University of Alberta

CHARACTERIZATION AND MODIFICATION OF OBLIQUELY DEPOSITED NANOSTRUCTURES

by

Kathleen Marion Krause

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of

Doctor of Philosophy.

in

Micro-Electrical-Mechanical Systems (MEMS) and Nanosystems

Department of Electrical and Computer Engineering

©Kathleen Marion Krause
Spring 2011
Edmonton, Alberta

Permission is hereby granted to the University of Alberta Libraries to reproduce single copies of this thesis and to lend or sell such copies for private, scholarly or scientific research purposes only. Where the thesis is converted to, or otherwise made available in digital form, the University of Alberta will advise potential users of the thesis of these terms.

The author reserves all other publication and other rights in association with the copyright in the thesis and, except as herein before provided, neither the thesis nor any substantial portion thereof may be printed or otherwise reproduced in any material form whatsoever without the author's prior written permission.

To my mom.
You gave me roots and wings.

Abstract

The glancing angle deposition (GLAD) technique is now used by over one hundred research groups, each requiring a fundamental understanding of and new techniques for modulating the properties of GLAD in order to optimize their results. In this thesis, the structural characteristics of nanostructured columnar films were therefore investigated and quantified using gas adsorption porosimetry, focused ion beam tomography, optical methods, scanning electron microscopy (SEM) image analysis. Questions such as "What is their surface area?", "How porous are they?", "How do the films evolve as they grow?", and "Can the structural characteristics be manipulated?" were answered. Surface areas, determined from krypton gas adsorption, were found to be high, making GLAD promising for applications requiring large and rough surface interfaces. Specifically, peak specific surface areas of 700 $\pm$ 150 m^2 g^{-1}, 325 $\pm$ 40 m^2 g^{-1}, 50 $\pm$ 6 m^2 g^{-1} were measured for silica (SiO_2), titania (TiO_2) and indium tin oxide (ITO), respectively. Broad pore distributions, with peaks in the low mesoporous regime of 2 nm to 5 nm, were also determined. The internal surface area may also be up to three times as high as that of the externally exposed surface. As well, despite the fact that GLAD column broaden as they grow, the surface area increases linearly with film thickness. Focused ion beam milling, with concurrent SEM imaging, was then employed to investigate and reconstruct the three-dimensional structure of GLAD films in the tens of nanometers regime not measurable by krypton gas adsorption porosimetry. The measured growth scaling trends agreed with previous findings, but were determined using only one sample, instead of multiple samples of increasing thickness. Mean column diameters, center-to-center spacings, void spacings, and column densities were found to scale with thickness as w = (9.4 $\pm$ 3.0) $t^{0.35\pm0.09}$ nm, c = (24.8 $\pm$ 5.2) $t^{0.31\pm0.08}$ nm, v = (15.2 $\pm$ 3.8) $t^{0.25\pm0.06}$ nm, and d = (3400 $\pm$ 2500) $t^{-0.65\pm0.15}$ columns μm^{-2}, respectively. Finally, spatially graded nanostructures were demonstrated by extending the GLAD technique to include macroscopic shadowing. Optically transparent, graded thickness and pitch helical films were fabricated with polarization selectivity over a spatial range of 30 mm, concurrent with 70 nm spectral tunability. These structures will be useful for tunable frequency photonic devices.

Acknowledgements

The completion of this thesis wouldn't have been possible without the support of numerous people. Thank-you to my mom and dad who have loved and supported me through all of the steps that have led to this point. Now that I am a mom I understand how much they did for me. Thanks also for the support of my husband, John, for his never failing encouragement of everything I attempt, including this most recent endeavor. Thank you to my little Rebecca who quietly napped through much of my writing and who also, by her daily discoveries, reminds me of how fascinating the world is! Thanks also to my brother David and sister-in-law Irene (and their sweet baby Emily) who both know what it's like to write a thesis and encouraged me throughout the process.

Thank you Dr Brett for providing an amazing research and learning opportunity. Your guidance and mentorship were invaluable. Thank you also to Mike Taschuk for providing insight on all aspects of my work. To the other members of the GLAD group, thank you for making my graduate studies experience fun, entertaining, and enriching.

I must acknowledge the invaluable help of my collaborators, coworkers, and mentors at the National Research Council (NRC) National Institute for Nanotechnology (NINT) and the University of Alberta. Thank you to Doug Vick, Mike Xia, Daniel Salamon, Marek Malac, Peng Li, Ken Harris, Dave Rider, Jillian Buriak, Mike Fleischauer, and Julie Qian. Thank you also to my collaborator at Quantachrome Inc, Dr. Matthias Thommes for our many discussions on the porosimetry work.

Thank you to the ladies at WISEST and WISER for providing a support and mentorship network that has sustained my enthusiasm for engineering and science throughout my research.

Finally, I would like to acknowledge the funding agencies who made this work possible. Thank you to the Natural Sciences and Engineering Research Council of Canada, Alberta Innovates: Technology Futures, the Informatics Circle of Research Excellence, Engineers Canada, the Senior Women Academic Administrators of Canada, Micralyne Inc., NRC-NINT, and the University of Alberta.

Table of Contents

1 Introduction **1**
1.1 Nanostructure Fabrication . . . 1
1.2 Nanostructure Characterization . . . 4
1.2.1 Morphological characterization by electron microscopy . . . 4
1.2.2 Measurements of surface roughness and porosity . . . 5
1.2.3 Optical characterization of nanostructures . . . 7
1.3 Overview of Glancing Angle Deposition . . . 8
1.4 Motivation and Scope of Research . . . 12
1.5 Major Contributions of this Work . . . 14

2 Nanostructure fabrication and analysis of fundamental properties **16**
2.1 Introduction . . . 16
2.1.1 Materials . . . 16
2.1.2 Structures . . . 17
2.2 Sample fabrication . . . 19
2.3 Sample volume . . . 30
2.4 Compensation for humidity infiltration . . . 30
2.4.1 Effective medium approximation . . . 31
2.4.2 Optical humidity measurements . . . 32
2.4.3 Estimation of adsorbed water volume into TiO_2 films . . . 36
2.5 Sample mass . . . 38
2.6 Density Calculations . . . 40
2.6.1 Measured Density . . . 40
2.6.2 Poxson fitting of vertical post densities . . . 42
2.6.3 Poxson fitting of helical column densities . . . 47
2.6.4 Densities of Multilayer films . . . 47
2.7 Chapter Summary . . . 50

3 Surface Area Characterization **51**
3.1 Introduction . . . 51
3.2 Surface area measurement . . . 52
3.2.1 GLAD surface area measurement . . . 52
3.2.2 Gas adsorption porosimetry . . . 54
3.2.3 BET analysis . . . 55
3.2.4 Krypton gas adsorption . . . 61
3.2.5 Cyclic voltammetry . . . 62
3.3 Experimental details . . . 63
3.4 Results and Discussion . . . 64
3.4.1 Calibration . . . 64
3.4.2 Surface area of GLAD films . . . 64
3.4.3 Thickness Study . . . 68
3.4.4 Helical film surface areas . . . 70
3.4.5 Confirmation by Cyclic voltammetry . . . 71
3.4.6 Multilayer films . . . 72
3.4.7 Aging study . . . 73
3.5 Chapter Summary . . . 76

4 Pore Size Distribution Measurement of GLAD Nanocolumns **78**
4.1 Introduction . . . 78
4.2 Pore size distribution measurement . . . 79
4.3 Kelvin method and BJH analysis . . . 82
4.4 Adsorption experimental details . . . 86
4.5 Inter-column porosity by analysis of SEM images . . . 87
4.6 Porosimetry analysis . . . 89
4.6.1 Isotherms . . . 89
4.6.2 BJH Analysis . . . 91
4.6.3 Empirical PSD calculations . . . 92
4.6.4 Pore volume estimation . . . 95
4.7 Chapter Summary . . . 98

5 Growth scaling and 3D reconstructions **99**
5.1 Introduction . . . 99
5.2 Focused ion beam tomography . . . 102
5.3 Experimental details . . . 107
5.3.1 Sample fabrication . . . 107
5.3.2 Tomography preparation . . . 108
5.3.3 FIB sectioning . . . 110
5.4 FIB cross-section analysis . . . 114
5.4.1 Alignment . . . 114
5.4.2 Stitching . . . 116
5.4.3 Image Post-processing . . . 117
5.4.4 Columnar properties and reconstruction . . . 120
5.5 Growth scaling . . . 124
5.5.1 Thresholding . . . 124
5.5.2 Column identification . . . 126
5.5.3 Nearest neighbor identification . . . 130
5.5.4 Spacing between columns . . . 133
5.5.5 Growth scaling . . . 133
5.5.6 Sensitivity Analysis . . . 137
5.5.7 Growth parameter relationships . . . 138
5.5.8 Surface area prediction . . . 140
5.6 Tracking individual columns . . . 142
5.7 Chapter Summary . . . 151

6 Modulation of growth properties **152**
6.1 Introduction . . . 152
6.2 Nature-inspired tunable optics . . . 152
6.3 Graded thickness method . . . 153
6.4 Experimental details . . . 154
6.4.1 Sample preparation . . . 154
6.5 Structural Characterization of Spatially Graded Helical Films . . . 159
6.5.1 Thickness Gradation prediction . . . 159
6.5.2 Graded thickness measurement . . . 163
6.6 Optical Characterization of Spatially Graded Helical Films . . . 165
6.7 Chapter Summary . . . 173

7 Summary and Conclusions **174**
7.1 Summary of thesis work . . . 174
7.2 Future Directions . . . 178
7.3 Final remarks . . . 179

Bibliography **180**

A Estimation of Density, Surface Area, and Porosity **193**
A.1 Code to estimate water adsorbed into TiO_2 and SiO_2 films . . . 193
A.2 Code to calculate Density and Surface Area of Vertical Post and Helical Films . . . 198
A.3 Code to estimate Density and Surface Area of Bragg and Rugate stacks . . . 210
A.4 Code to calculate Pore Size Distribution of films using BJH method . . . 218
A.5 Code to plot Pore Volumes . . . 223
A.6 Code to plot PSDs calculated using Quantachrome Empirical Method . . . 228
A.7 Code to estimate inter-column porosity by SEM image analysis . . . 233

B Optical relative humidity monitoring of GLAD films **239**

C Image processing of FIB-SEM image slices **255**
C.1 Code to track fiducial and image selection from image to image 255
C.2 Code to process and identify columns in each slice image selection 260
C.3 Code to Plot Column Diameters and to Find Nearest Neighbors 281
C.4 Code to Make Movies of FIB slices and to save binary file for analysis by JEOL TEMography software 291
C.5 Code to Determine Surface Area from Milled Images 293
C.6 Code to Determine Inter-Column Spacing 295
C.7 Code to Analyze Bifurcation of Individual Columns 305

D Spatially graded GLAD modeling **318**
D.1 Code to plot shadowed slanted post calibration film thicknesses 318
D.2 Code to simulated shadowed chiral film thicknesses 337

List of Tables

2.1 Design deposition and rotation angle parameters for rugate films. From each step to the next the change in thickness is monitored using the CTM, with the end of step target deposition and substrate rotation angles achieved by the interpolation routines of the deposition control software. 21
2.2 Properties of TiO_2 rugate films acquired from humidity study. 36
2.3 Properties of films deposited for surface area study. VP indicates a vertical post film structure. 41
2.4 Properties of multilayer films deposited for characterization study. 42
2.5 Density fits for TiO_2 thickness study films. 42

3.1 Verification of Porosimeter Functionality with Standard Samples. 64
3.2 TiO_2 vertical post surface area scaling with thickness (where t is in μm). 69

5.1 FIB-SEM slice series parameters. 114
5.2 Variable values for connected domain breaking algorithm. Connected domain areas must be larger than $f_{area} \cdot \overline{A}$. Pairs must closer than $f_{minBridge}$ pixels to possibly be spanning a bridge. Normal vector inflection values must be larger than f_{peak} deg pixel^{-1} to be possible indicators of a bridge. Pairs must be within f_{infl} pixels of large normal vector inflections to be considered. Pair within $f_{minPair}$ pixels of another pair are considered the same. 129

6.1 Properties of shadowed slanted post TiO_2 films used for calibration of shadowing simulations. Values for the target (t_{target}) and actual (t_{center}) thicknesses of substrates at the center of the deposition sample holder are provided. 155
6.2 Properties of graded chiral films. The target (t_{target}) thickness, actual thickness at 50 mm from the edge of the shadow block ($t_{(50mm)}$), number of turns (N_{turns}), thickness gradation measured at 5 mm to 50 mm from the shadow block edge ($\Delta\, t_{full}$), thickness gradation measured at 20 mm to 50 mm from the shadow block edge ($\Delta\, t_{upper}$), and pitch of the helical columns at 50 mm from the shadow block edge ($\Pi_{(50mm)}$) are provided. 156

List of Figures

1.1 Initial nucleation and effect of deposition angle on nuclei self-shadowing. At (a) normal incidence, impinging flux will deposit on and between nuclei forming a thin uniform film, while for (b) moderate deposition angles some or portions of small nuclei will be shadowed forming a densely packed columnar structure and (c) at oblique deposition angle whole nuclei will be shadowed by larger neighbors forming a columnar nanostructure. 9
1.2 GLAD apparatus employs physical vapor deposition to deposit a collimated vapor onto a tilted and rotating substrate. 10
1.3 Representative structures created using the GLAD technique. 11
1.4 Characterization and manipulation of GLAD thin films using three different techniques that cover a size scale from nanometers to millimeters. 13

2.1 Deposition chuck prepared for porosimetry study. 20
2.2 Photograph of deposition chuck is analyzed by image processing software to determine footprint area of sample pieces. 22
2.3 Cross-section and plan view SEM images of selected TiO_2 vertical post films deposited at (a), (b) 45^o, (c), (d) 55^o, (e), (f) 60^o, (g), (h) 65^o, (i), (j) 70^o, (k), (l) 75^o, (m), (n) 80^o, (o), (p) 85^o. 23
2.4 Cross-section and plan view SEM images of representative SiO_2 films deposited at (a), (b) 75^o and (c), (d) 85^o. 24
2.5 Cross-section and plan view SEM images of representative ITO films deposited at (a), (b) 45^o, (c), (d) 60^o, (e), (f) 70^o, (g), (h) 85^o. 25
2.6 Cross-section and plan view SEM images of selected helical TiO_2 films: (a) α = 70^o, (b) $\alpha = 75^o$, (c) $\alpha = 80^o$, (d) $\alpha = 85^o$. 26
2.7 Cross-section SEM images of TiO_2 rugate films deposited with α ranging from a) 30^o to 80^o, b) 45^o to 80^o, c) 50^o to 80^o, and d) 60^o to 80^o. 27
2.8 Cross-section SEM image of TiO_2 Bragg film, with bands deposited at $\alpha = 60^o$ to 80^o. a) Shows full thickness of deposited sample, while close-ups of the layers at the b) bottom and c) top of the film also shown. 28
2.9 TEM images of selected films. (a) and (b) TiO_2 $\alpha = 80^o$ TEM with inset diffraction pattern indicating an amorphous structure. (c) SiO_2 $\alpha = 80^o$ TEM, with amorphous structure verified by XRD measurements [93]. (d) ITO $\alpha = 80^o$ TEM with inset diffraction pattern indicating a polycrystalline structure. TEM images in (a) and (c) ©2008 IEEE. Reprinted with permission from IEEE Sensors, [93] 29
2.10 Humidity sensing chamber 34
2.11 Changing transmission spectra as a function of humidity for rugate films deposited with α ranging from (a) 30^o to 80^o, (b) 45^o to 80^o, (c) 50^o to 80^o, and (d) 60^o to 80^o. 35
2.12 Change in (a) blue edge and (b) red edge of stop-band as a function of relative humidity for the four rugate films (designated by densest layer α). Location of stop-band edges measured at 30% transmission. 35
2.13 Estimated volume of water adsorbed into a TiO_2 film as a percentage of total volume. 37
2.14 Estimated mass of water adsorbed into a TiO_2 film as a percentage of total sample mass. 39
2.15 Scatter of density measurements for individual SiO_2 $\alpha = 60^o$ 1 μm samples prepared for surface area analysis. Dashed line indicates average density. 42
2.16 Densities of TiO_2 vertical post films as a function of film thickness for a range of deposition angles. 43
2.17 Densities of TiO_2, SiO_2, and ITO vertical post films. 44

2.18 Uncertainties in density as a function of deposition angle for a) TiO_2, b) SiO_2, and c) ITO vertical post films. Stars indicate uncertainty calculated from measured mass, footprint area, and thickness values. Triangles indicate standard deviation of scatter from individual sample piece densities. Squares provide the difference in Poxson fit and measured densities. Diamonds are difference between densities measured using the VASE and those calculated from measured mass, footprint area, and thickness values. Sideways triangles indicate thickness study density uncertainties. 46
2.19 Densities of TiO_2 vertical post (VP) helical (He) films as a function of deposition angle, with Poxson fit curves indicated by solid and dashed lines respectively. . . . 47
2.20 Densities of TiO_2 rugate films. Circles indicate measured densities of these multilayer films. Squares indicate densities expected based on weighted averages of single deposition angle vertical post films. 49
2.21 Densities of TiO_2 Bragg stack films. Circles indicate measured densities of these multilayer films. Squares indicate densities expected based on weighted averages of single deposition angle vertical post films. 49

3.1 Diagram of gas adsorption porosimeter. Adsorbate is introduced to the instrument under high vacuum. The reference tube is used to adjust the system pressure before the sample volume is exposed to the adsorbate. 55
3.2 IUPAC Isotherms (adapted from [172]). 56
3.3 BET analysis is performed on an isotherm, such as the one shown in (a) in order to determine the surface area from analysis performed on isotherm linearization illustrated in (b). 57
3.4 BET-modeled surface of porous sample. Molecules collide with the sample surface where they adsorb at a rate determined by the sticking probability, temperature, pressure. Desorption also occurs at a rate dependent on temperature, activation energy of the system. Molecules are assumed to form layers where all layers except the first behave identically. 58
3.5 Phase diagram illustrates triple point temperature. 62
3.6 Krypton gas sorption isotherms at 77 K for (a) and (b) TiO_2, (c) and (d) SiO_2, (e) and (f) ITO vertical post GLAD films deposited at the indicated deposition angles. . 66
3.7 a) Relative surface area and b) specific surface area for TiO_2, SiO_2 and ITO vertical post films 68
3.8 Relative surface area of TiO_2 vertical post (VP) vs thickness for a range of deposition angles. Curves shown to guide the eye. 69
3.9 Relative surface areas of TiO_2 vertical post (VP) compared to those of helical (He) films for a range of deposition angles. 70
3.10 Relative surface areas of ITO vertical post films calculated by BET method and CV electro-chemical method. 71
3.11 a) Relative surface area and b) specific surface area for rugate films. Films were deposited from note α to 80^o. Circles indicate measured surface areas of these multilayer films. Squares indicate densities expected based on weighted averages of surface areas for single deposition angle vertical post films. 73
3.12 (a) Relative surface area and (b) specific surface area for Bragg stacks. Films were deposited from 60^o to 80^o. Circles indicate measured surface areas of these multilayer films. Squares indicate densities expected based on weighted averages of surface areas for single deposition angle vertical post films. 74
3.13 Specific surface decreases over time for (a) SiO_2 $\alpha = 60^o$, (b) TiO_2 $\alpha = 60^o$, and (c) TiO_2 $\alpha = 80^o$ set 1 and set 2 films. 75

4.1 Pore condensation (transition (a) to (b)) occurs when a critical adsorbate thickness is reached on the sides of the pore walls. Pore evaporation (transition (b) to (c)) occurs by a receding meniscus as the gas pressure is decreased. 80
4.2 H1 hysteresis is associated with well-defined cylindrical pore channels. H2 hysteresis is common with disordered, wide pore size distribution materials. H3 hysteresis occurs for samples with slit-like pores or plate-like particles. H4 hysteresis also characterizes slit-like pores, but in the micropore region. (Adapted from [171]) . . 81
4.3 Condensate in a pore of radius r_p forms a thin film of thickness (t_m) and a meniscus of radius (r_k). 83
4.4 Intercolumn porosity estimated by thresholding and analyzing planar view SEMs of 1 μm thick TiO_2 GLAD films at deposition angles of (a) 65^o, (b) 70^o, (c) 75^o, (d) 80^o, (e) 85^o. 87
4.5 Histograms of planar view TiO_2 α = (a) 65^o, (b) 70^o, (c) 75^o, (d) 80^o, (e) 85^o GLAD film SEM images reveal optimum threshold for separating columns from void spaces. 88

4.6 Pixel densities of thresholded planar view SEM images as a function of deposition angle. . . . 89
4.7 Distribution of void spacing distances between the columns of 1 μm thick TiO_2 GLAD films at deposition angles of (a) 65^o, (b) 70^o, (c) 75^o, (d) 80^o, (e) 85^o. Note the change in x-axis scale. . . . 90
4.8 Mean intercolumn void spacing distances between columns of TiO_2 GLAD films at 1 μm thickness. . . . 91
4.9 Krypton gas sorption isotherms at 87 K for (a) and (b) TiO_2, (c) and (d) SiO_2 GLAD films of 1 μm thickness deposited at the indicated deposition angles. . . . 91
4.10 Pore size distributions as determined by BJH analysis for (a) and (b) TiO_2, (c) and (d) SiO_2 1 μm thick GLAD films deposited at the indicated deposition angles. . . . 92
4.11 Pore size distributions as determined by empirical analysis for (a) and (b) TiO_2, (c) and (d) SiO_2 1 μm thick GLAD films deposited at the indicated deposition angles. . 93
4.12 Comparisons of peak pore size and pore spread for (a) and (b) TiO_2, (c) and (d) SiO_2 1 μm thick GLAD films as determined by BJH and Empirical methods. . . . 94
4.13 Total specific pore volume at 0.95 p_o for TiO_2 and SiO_2 1 μm thick GLAD films deposited over a range of deposition angles. . . . 96
4.14 Pore volume as a percentage of sample volume for TiO_2 and SiO_2 1 μm thick GLAD films deposited over a range of deposition angles. Triangles indicate measured porosity, the dashed line shows the porosity predicted by Tait [147], and the circles show the porosity determined by analysis of the 87 K krypton adsorption isotherms at 0.95 p_o. . . . 97

5.1 Schematic diagram of focused ion beam instrument. . . . 103
5.2 Schematic diagram of dual-beam FIB-SEM instrument. . . . 106
5.3 (a) Top-down and (b) side-view cross-sections of the original 5 μm TiO_2 GLAD nanostructured thin film deposited at α = 85^o. Side-view cross-section of the (c) once-filled and (d) twice-filled polymer-filled film. . . . 109
5.4 (a) Specimen orientation relative to FIB and SEM columns for the side-polishing procedure. (b) Side-polishing schematic. . . . 110
5.5 (a) Portion of sample is side-polished prior to sectioning. (b) Closer magnification reveals curtaining of silicon substrate that results as the FIB beam passes through the two-material GLAD film. . . . 110
5.6 (a) Side view of FIB-polished SEM cross-section of 5 μm polymer-filled GLAD TiO_2 film. Columns are tilted due to the polymer filling process. (b) Image is thresholded to isolate GLAD columns from void spaces such that (c) density analysis of the image can be carried out by finding the ratio of white to black pixels along horizontal lines through the thresholded SEM image from the substrate-film interface (0 nm) to the top of the film (5000 nm). . . . 111
5.7 Protective carbon layer deposited on side of GLAD film prior to planar sectioning as seen from (a) FIB view and (b) SEM view. . . . 111
5.8 (a) Specimen orientation relative to the FIB and SEM columns for the tomographic slicing process. (b) Tomographic slicing proceeds from the top of the film towards the substrate, exposing a planar cross-section of the GLAD film. . . . 112
5.9 (a) In FIB view, establish milling volume before beginning of second slice series. (b) SEM view of milling volume before slicing. (c) FIB view of milling volume after slicing. (d) SEM view of exposed GLAD film cross-section after milling. . . . 113
5.10 Representative cross-sectional slices shown for slice series 3 (a) before and (b) after the milling sequence has cut multiple slices from the sample volume. . . . 113
5.11 SEM images of the GLAD film (a) before and (b) after the execution of the first slice sequence. The dark region at the top of image (a) is the carbon protective layer on the side of the sample. The interface between this layer and the photoresist is clearly visible in frame (b). The bright region at the bottom of image (a) is the gold-coated top of the sample, removed by the FIB sectioning to reveal a top-down cross-sectional image of GLAD columns. The dashed outline in each figure indicates the location of the fiducial region used for image alignment. . . . 116
5.12 Representative progressive fiducial drift, measured relative to the initial fiducial image location, for slice series 4. Each data point represents one slice in the series. . . 117
5.13 The tracked region (c) in the last slice of series 3 (a) is stitched to the corresponding region (d) in the first slice of series 4. . . . 118
5.14 Image equalization is carried out in order to compensate for local charging and other artifacts in image (a) by breaking the image into tiles as shown in (b) and calculating the contrast equalization multiplier required to bring each tile to the average contrast. (c) Shows the filtered contrast equalization matrix and (d) shows the equalized image. 120

5.15 Top-down FIB cross sections of the 5 μm GLAD film at (a) 980 nm, (b) 1220 nm, (c) 3360 nm, (d) 4240 nm, (e) 4390 nm above the substrate. These cross sections illustrate the evolution of column diameter, spacing, and density. 121
5.16 GLAD columns emerging from the polymer medium for slices spaced at 16 nm. Image (a) is at the top of the film, while image (f) is down 96 nm from the surface. Due to volume penetration of the electrons, the actual location of the tips of the columns is likely several tens of nanometers below the first signal detected. 122
5.17 Final milling stages as substrate is reached. Slices are 4 nm apart. GLAD film is visible in entire milling region in images (a) and (b). At slices (c) and (d) the substrate is reached, but not uniformly. 123
5.18 Three-dimensional tomographic views of a stitched GLAD film showing (a) a representative volume at an oblique angle such that columnar features are visible and (b) a view from the top of image a down through the film. (c) An extracted volume where surface details are visible on the individual columns. 124
5.19 (a) Equalized top-down cross section of an FIB-sliced GLAD film in the second series (4240 nm from substrate), (b) thresholded image, (c) connected domains of clumped columns identified by unique color. 127
5.20 Connected domains, as illustrated in (b), are identified within each image (a) and broken into individual columns, as shown in (c). 129
5.21 Connected domain is broken into individual columns through grain detection algorithm. (a) Boundary of connected domain is identified, with boundary1 and boundary2 identifying travel along the shape edge in clockwise and counterclockwise directions. (b) Minimum distances between boundary1 and boundary2 point pairs, with only those on opposite edges of the domain kept in (c). Finally, in (d) only points with large concavity are used to break the connected domain up into (e) individual columns. 129
5.22 Local minima in the matrix of distances between points along the boundary of a shape help to identify bridges between connected columns. 130
5.23 Distribution of GLAD column diameters for three selected slices, at 4235 nm, 2773 nm, and 258 nm from the substrate-film interface. Fit lines represent Gaussian curve fits 131
5.24 Nearest neighbor columns are found along trajectories at 30 ^{o} increments. 131
5.25 Nearest neighbor columns are identified in a representative slice. (a) Columns are represented by circles centered at the locations of the actual columns. (b) Lines are then drawn from each column center. (c) The intersection of each line with all columns along that line are noted. (d) The nearest column along the given trajectory is found. 132
5.26 Percent difference in mean center to center column spacing (from the values calculated at trajectories of 5^o) for trajectories from 5^o to 90^o at selected film thickness (i.e., FIB milled slices). 133
5.27 Void spacing between columns in (a) and (d) the equalized image is estimated by (b) and (e) thresholding and then (c) and (f) finding the distances between column to void transitions (grey lines) and void to column transition (white lines) in the row and column directions. Shown here are examples for milled slices at (a) through (c) 4240 nm and (d) through (f) 1500 nm from the substrate. 135
5.28 (a) Mean column diameter scales as $\overline{w}$ = 9.5 $t^{0.35}$ [nm] ($\overline{w}_{act}$ =9.4 $t^{0.35}$ when the column tilt is compensated for), where t is in nanometers. (b) Column center-to-center spacing between columns scales as $\overline{c}$ =24.8 $t^{0.30}$ [nm] ($\overline{c}_{act}$ =24.5 $t^{0.30}$). (c) Void spacing scales as $\overline{v}$ = 15.4 $t^{0.25}$ [nm] ($\overline{v}_{act}$ =15.2 $t^{0.25}$). (d) Column density scales with thickness d= 3.4x10^3 $t^{-0.65}$ [column μm^{-2}]. The inset shows a magnified region of the column density plot. The dotted line through the data points shows the scaling fit. Dashed-dotted lines show the fit to one standard deviation of parameter distributions. 136
5.29 (a) Spread in mean column diameter values, (b) spread in column center-to-center spacing between columns and (c) spread in column density as grain detection algorithm parameters changed by up to $\pm$ 10%. 138
5.30 Difference between growth scaling empirical constants and scaling parameter for (a) and (d) mean column diameter values, (b) and (e) column center-to-center spacing between columns and (c) and (f) column density as grain detection algorithm parameters changed by up to $\pm$ 10%. 139
5.31 Closed packed array of nanocolumns is used to calculate expected column density and surface area scaling as a function of film thickness. 140
5.32 Measured column density (δ_{act}) versus that predicted for a closed packed array of nanocolumns (δ_{hcp}). 141

5.33 (a) Cumulative surface area enhancement (SAE_{FIB}) growth trend as a function of film thickness calculated from slice image column perimeter areas and differential thickness. (b) Comparison of FIB-calculated surface area enhancement to that measured using gas adsorption porosimetry (SAE_{GasAd}) for TiO_2 α = 85^o samples, shown in Figure 3.8. Prediction based on FIB-milled GLAD columns underestimates surface area and predicts a sublinear growth trend. 141
5.34 Tracking of four individual columns beginning at 842 nm from the substrate. (a) Initial configuration of the columns and (b) Evolution of these columns through several tens of nanometers. 142
5.35 Columns in (a) an SEM cross-section are manually separated by drawing lines between the columns (where the lines are thickened and colored white for emphasis) as shown in (b). The image is then thresholded as shown to isolate individual columns as shown in (c). These columns are filled as shown in (d) to be used as a mask to determine (e) the cross-sectional features of the nanocolumn being tracked. 143
5.36 (a) and (b) Columns in two consecutive slices are overlayed to determine whether they match. The RMS sum of the pixel differences is shown in image (c). 144
5.37 (a) Column diameters and (b) cumulative relative surface areas of individual neighboring columns evolve as a function of thickness. Only column 1 survives, while the others become extinct. 145
5.38 Tracking progress of column 1 bifurcation through several slices from 2556 nm to 2725 nm. 146
5.39 Tomographic view of four individual nanocolumns: from 840 nm to (1) 3080 nm, (2) 1510 nm, (3) 2170 nm, and (4) 2760 nm above substrate. 148
5.40 Number of branches per column for (a) column 1, (b) column 2, (c) column 3, (d) column 4. Column 1, which survives longer than its neighbors, has the largest peak number of branches. 149
5.41 Cumulative surface areas of column branches for (a) column 1, (b) column 2, (c) column 3, (d) column 4. Surface areas for all branches at a given film thickness have been cumulatively summed. Maximum surface area envelope is the total cumulative sum to the given thickness. 150

6.1 Shadowed GLAD method. (a) A shadow block placed in the center of the obliquely inclined substrate holder (deposition angle α) casts a shadow (from point δ_1 to δ_2 along radial axis of substrate) on the substrate. (b) This results in a graded thin porous film with structure defined by the substrate motion characteristics, such as rotation angle (ϕ). 155
6.2 Image of deposition chuck and post-deposition shadowed slanted-post GLAD film at α = 60^o. 156
6.3 Image of chuck, post-deposition, of shadowed chiral GLAD film at α = 80^o. 157
6.4 SEM images of shadowed slanted film deposited at α = 70^o. Measured along a line at a radial distance of d_r = 45 mm from the shadow block edge and at (a) d_n = 0 mm, t = 63 nm, (b) d_n = 27 mm, t = 240 nm, (c) d_n = 37 mm, t = 607 nm 158
6.5 Images of spatially graded chiral films deposited at two different oblique angles (a through c deposited at 70^o and d through f deposited at 85^o) and shown at selected distances of (a) 0.3 mm, (b) 13 mm, (c) 54 mm, (d) 1 mm, (e) 13 mm, and (f) 56 mm from the shadow block edge. 158
6.6 Slanted post calibration film thickness profiles at α = 30^o (a,b), 60^o (c,d), 70^o (e,f), and 85^o (g,h). Nominal thickness profile of slanted post film as a function of radial distance along the chuck is shown in the first column. Thickness profiles behind the shadowing block, at a number of distances from the shadow block edge, are shown in the second column. 160
6.7 Combined shadowed (right half of each image) and unshadowed (left half of each image) slanted post calibration film thickness profiles at (a) α = 30^o, (b) 60^o, (c) 70^o, and (d) 85^o, with red being the thickest film and blue being the thinnest. Shadowing block region at center of simulated substrate holder set to zero thickness. 161
6.8 Experimentally obtained (circles) graded chiral film thickness profiles at (a) α = 60^o, (b) 70^o, and (c) 85^o are compared against the simulated thickness profiles (line). Inset shows measurement direction along sample. 164
6.9 Apparatus for optical measurements. 165
6.10 Apparatus for translation of sample slide. 166

6.11 (a) LCP (solid-line) and RCP (dashed-line) transmission spectra 50 mm from edge of shadow block. RCP light interacts with the RCP structure, while LCP light is transmitted through the film with less attenuation. (b) Difference in transmission spectra for left (clockwise) minus right (counterclockwise) circularly polarized light (as viewed from the source) at 50 mm from the shadow block edge, for a graded chiral TiO_2 film deposited at a deposition angle of 70^o. 167
6.12 Photograph of a graded chiral TiO_2 film deposited at a deposition angle of 70^o on a glass substrate. Incident light in this case is unpolarized. 168
6.13 Frequency selectivity is demonstrated through LCP-RCP transmission spectrum measurements. Measurements were made at selected distances from the macroscopic shadow block edge, for graded chiral TiO_2 films deposited at deposition angles of (a) 60^o, (b) 65^o, (c) 70^o, (d) 75^o, (e) 80^o. 169
6.14 Frequency and polarization selectivity for a spatially graded chiral film. (a) Peak location and (b) polarization selective transmission as a function of radial distance from the shadow block edge, for a chiral TiO_2 film deposited at a deposition angle of 70^o. 170
6.15 Graded chiral LCP-RCP transmission peak location changes as a function of distance from the shadow block edge. Maximum peak location change, over a distance of 50 mm, ranges from 150 nm to 250 nm for films deposited at angles from 60^o to 80^o. 170
6.16 Selectivity at 50 mm, for films deposited at angles from 60^o to 80^o (line shown to guide the eye). Selectivity has been normalized by N_{turns} at a deposition angle of 70^o. 171
6.17 Estimated index of refraction at distances of 5 mm, 20 mm, and 50 mm from the edge of the shadow block, for films deposited at deposition angles from 60^o to 80^o. 172

B.1 Main user interface to LabVIEW program which monitors changes in GLAD film transmission spectra as a function of humidity. 240
B.2 Underlying LabVIEW code to user interface. 241
B.3 Main LabVIEW code step 0 for extracting and analyzing transmission spectrum. . 241
B.4 Main LabVIEW code step 1 for extracting and analyzing transmission spectrum. . 242
B.5 Main LabVIEW code step 2 for extracting and analyzing transmission spectrum. . 242
B.6 Main LabVIEW code step 3 for extracting and analyzing transmission spectrum. . 243
B.7 Main LabVIEW code step 4 for extracting and analyzing transmission spectrum. . 243
B.8 Main LabVIEW code step 4b for extracting and analyzing transmission spectrum. . 244
B.9 Main LabVIEW code step 5 for extracting and analyzing transmission spectrum. . 244
B.10 Main LabVIEW code step 6 for extracting and analyzing transmission spectrum. . 245
B.11 Main LabVIEW code step 7 for extracting and analyzing transmission spectrum. . 245
B.12 Main LabVIEW code step 8 for extracting and analyzing transmission spectrum. . 246
B.13 Main LabVIEW code step 9 for extracting and analyzing transmission spectrum. . 246
B.14 Main LabVIEW code to display transmission spectrum along with measured humidity. 247
B.15 Main LabVIEW code to calculate differences between sample and reference spectra. 248
B.16 LabVIEW interface for routine to get raw intensity data from spectrometer. 249
B.17 LabVIEW code step 0 for routine to get raw intensity data from spectrometer. . . . 249
B.18 LabVIEW code step 1 for routine to get raw intensity data from spectrometer. . . . 250
B.19 LabVIEW code step 2 for routine to get raw intensity data from spectrometer. . . . 250
B.20 LabVIEW interface for routine that identifies spectrometers. 251
B.21 LabVIEW interface for routine that gets wavelength array from spectrometer. . . . 252
B.22 LabVIEW code for routine that gets wavelength array from spectrometer. 252
B.23 LabVIEW interface for routine that sets spectrometer parameters. 253
B.24 LabVIEW code for routine that sets spectrometer parameters. 253
B.25 LabVIEW interface for routine that calculates transmission spectrum. 254
B.26 LabVIEW code for routine that calculates transmission spectrum. 254

List of Symbols and Abbreviations

α	Deposition angle
γ	Surface tension of the liquid
ϵ_r	Relative dielectric constant of a medium
ϵ_ρ	Error threshold for FIB-SEM image contrast adjustment routine
Θ	Substrate rotation angle
θ	Tilt imposed on GLAD film when filled with polymer
λ_{SB}	Optical interference stop-band center wavelength for a Rugate or Bragg structure
λ_{peak}	Peak location of LCP-RCP transmission
$\Delta\lambda_{peak}$	Change in helical LCP-RCP transmission peak location
$\Delta\lambda_{50\%RH}$	Shift in rugate stop-band edge as humidity changed from 0 to 50 % RH
μ_r	Relative permeability of a medium
μ_0	Mean pixel value of black level in thresholded grayscale image
μ_1	Mean pixel value of white level in thresholded grayscale image
$\mu(\Upsilon)$	First-order cumulative moment of grayscale image histogram up to Threshold level
μ_{Total}	Mean pixel value of grayscale image
Π	Period of multilayer film
ρ	Film density in pixels per unit area or white pixels per total number of pixels
ρ_{target}	Target pixel density of SEM image
$\Delta\rho$	Difference in target and SEM image pixel density
δ	Film density in g cm^{-3}
δ_{Tait}	Film density predicted by Tait *et al.*
σ_δ	Uncertainty in film density measurements
τ	Molecular thickness of adsorbate
Υ	Grayscale image threshold for determining black vs white pixels
Υ_c	Threshold of grayscale FIB-SEM slice image
$\Delta\Phi$	FIB specimen drift in x or y direction
ξ_{Fc}	Cross-sectional area of Ferrocene molecule
ξ_{cs}	Cross-sectional area of adsorbate
$\Psi_{rt,ct}(r,c)$	Multiplier to equalize contrast of a pixel
$\psi_{rt,ct}$	Multiplier to bring contrast threshold of sub-image tile to that of the whole image
ω_0	Probability of occurence of black level in thresholded grayscale image
ω_1	Probability of occurence of white level in thresholded grayscale image
$\omega(Th)$	Zeroth-order cumulative moment of grayscale image histogram up to Threshold level
a_i	Circumferential area of a single GLAD nanocolumn at a given cross section (i)
B	Number of branches for a given column as function of thickness
C_{ML}	Monolayer coverage factor for Cyclic Voltammetry method
C_{BET}	BET linearity constant
C	Linear-term constant in fit of density vs deposition angle trend
$C(\alpha)$	Graded film calibration profile
c	Column center-to-center spacing
$\overline{c}$	Mean of center-to-center spacing for all columns in a given slice
$\overline{c}_{avg}$	Mean of center-to-center spacing corrected for filling induced film tilt
$Co_{i,j}$	Column j in connected domain i of a top-down SEM image

Co_m	Column m in top-down SEM image
D	Pore diameter
d_{column}	Column density
d_{column_act}	Column density corrected for filling induced film tilt
d_n	Normal coordinate (perpendicular to radial direction from center of chuck) on deposition chuck
d_r	Radial coordinate on deposition chuck
d_{pore}	Peak pore diameter of pores on the GLAD column
d_{void}	Void spacing between columns in a GLAD film
E	Activation energy of an adsorbate-adsorbant system
F_{new}	Fiducial image region in SEM image of size r_{Nf} rows by c_{Nf} columns
f	Filter coefficient for filter SEM images
f_{area}	SEM image analysis: Connected domain areas must be larger than f_{area} times the mean area
$f_{minPair}$	Possible bridge pairs must be farther apart than this number of pixels
f_{peak}	Boundary normal vector inflections must be larger than this value
f_{infl}	Bridge pairs must be within this number of pixels of a normal vector inflection
f_{delta}	Normal vector change minima must be within this distance of a maxima for f_{peak} to be calculated
f_{peak}	The difference between local minima and maxima of the change in normal vector must be larger than this
FA	Footprint area of film deposited on a substrate
I	SEM image of size r_N rows by c_N columns
I_nc	Cross-sectional slice image of a nanocolumn
I_t	Filtered SEM slice image
I_s	Selected portion of SEM image of size r_{Ns} rows by c_{Ns} columns
$I_{s_rt,ct}$	Tile at location (rt = 1..n,ct = 1..n) of SEM image, when broken into n x n tiles
I_t	Thresholded SEM slice image
K	Quadratic-term constant in fit of density vs deposition angle trend
M	Molecular weight
M_Υ	Threshold multiplier to adjust image contrast of FIB-SEM slice images
$m_{initial_i}$	Initial mass of sample piece
m_{final_i}	Final mass of sample piece after deposition
m_{film}	Mass of deposited film
Δm	Difference in mass between substrate after and before deposition
m_{water}	Mass of water adsorbed onto deposited film
N_a	Avagadro's number
N_{branch}	Number of branches per nanocolumn in a given cross-sectional slice
N_δ	Number of points in density plot
N_{gray}	Number of grayscale pixel values
N_{layers}	Number of layers in a multilayer film
$N_{molecules}$	BET analysis: Number of adsorbate atoms stacked at a given site on a surface
N_{pieces}	Number of sample pieces used to characterize a given deposited film
N_{pore}	Number of steps (pores) in the BJH PSD calculation
N_{rad}	Number of points along a radial distance vector (in deposition simulation)
N_S	Number of connected domains (i.e., clumps of columns) in a top-down SEM image
N_{steps}	Number of thickness steps in a deposition file
N_{turns}	Number of turns in a helical film
n_0	Number of black pixels in an SEM image
n_1	Number of white pixels in an SEM image
n_{ad}	Total number of moles adsorbed onto a surface
n_{avg}	Average refractive index of a GLAD film
n_{Fc}	Number of ferrocene molecules transferred to a surface
n_i	Number of pixels with a given grayscale value in an image
n_{ml}	Monolayer capacity of a gas adsorbed onto a surface
n_t	Number of tiles that an SEM image is broken up into for contrast equalization
P	Characteristic growth scaling parameter for columns
P_{film}	Thin film porosity

p/p_o	Pressure relative to sublimation pressure
p_A	Percent closeness of branch areas for a tracked GLAD nanocolumn
p_d	Percent closeness of branch centers for a tracked GLAD nanocolumn
p_{bridge}	Bridge points between columns in a connected domain identified in a top-down SEM image
Q	Charge transferred in Cyclic Voltammetry technique
R	Gas constant
r_k	Radius of condensate in pore in a porous medium
r_p	Pore radius in a porous medium
S_c	Condensation coefficient for a gas hitting a surface
S_i	Connected domain (i.e., clump of columns) in a top-down SEM image
SA	Surface area as calculated by BET method
SAE	Surface area enhancement as calculated by BET method
SAE_{Fc}	Surface area enhancement as calculated by Cyclic Voltammetry
S_n	Slice number in a FIB series
s	Slope of a fit line
T	Temperature
T	Transmission spectra (percentage)
T_{LCP}	Left circularly polarized transmission spectra
T_{RCP}	Right circularly polarized transmission spectra
T_{peak}	Transmission value of peak in LCP-RCP transmission spectrum
t	Film thickness from substrate
Δt	Incremental film thickness of a given slice
t_m	Film thickness of condensed adsorbate on surface of porous medium
V_{gas}	Volume of gas adsorbed onto a surface
V_m	Molar volume of condensed adsorbate
V_{ml}	Volume of a monolayer of gas adsorbed onto a surface
V_o	Molecular volume of an adsorbate gas at standard temperature and pressure
V_{pore}	Total pore volume of a porous material
dV_p / d	Pore volume per differential pore diameter
dV_p / $log(d)$	Pore volume per log of differential pore diameter
V_{sample}	Volume of thin film sample used in porosimetry measurements
V_{water}	Volume of water adsorbed into thin film sample at 50 %RH
v	Void spacing between columns
$\overline{v}$	Mean void spacing between all columns in a give FIB slice
$\overline{v}_{act}$	Mean void spacing corrected for filling-induced film tilt
w	Cross sectional diameter of a column
$\overline{w}$	Mean cross sectional diameter of all columns in a given FIB slice
$\overline{w}_{act}$	Mean cross sectional diameter corrected for filling-induced film tilt
w_B	Diameter of branches for a column
w_o	Empirical material-dependent constant characterizing the growth of columnar diameter
x	Direction of FIB slice width (direction of raster)
y	Direction of FIB slice height (proportional to dwell time)
Z	Atomic number
z	Direction of FIB slice depth

AFM	Atomic force microscopy
AP	Atom probe
ASW	Amorphous solid water
BET	Brunauer-Emmett-Teller method
BF	Bright field
BJH	Barrett, Joyner and Helenda method
BSE	Backscattered electron
CCD	Charge coupled device
CNT	Carbon nanotube

CTM	Crystal thickness monitor
CVD	Chemical vapor deposition
DF	Dark field
DFT	Density functional theory
EDS	Energy dispersive spectrometers
EMA	Effective medium approximations
EsB	Energy selective backscatter
FIB	Focused ion beam
FWHM	Full-width half-maximum
GLAD	Glancing angle deposition
IUPAC	International union of pure and applied chemistry
IPA	Isopropyl alcohol
ITO	Indium tin oxide
LC	Liquid crystal
LCP	Left handed circularly polarized
LMIS	Liquid-metal ion sources
MEMS	Micro-electro-mechanical system
MCP	Microporous coordination polymer
MgO	Magnesium oxide
MOF	Metal oxygen framework
NEMS	Nano-electro-mechanical system
NIR	Near infrared
NLDFT	Non-local density functional theory
PDMS	Polydimethylsiloxane
PECVD	Plasma enhanced chemical vapor deposition
PSD	Pore size distribution
PVD	Physical vapor deposition
QCM	Quartz crystal microbalance
RCP	Right handed circularly polarized
RH	Relative humidity
RMS	Root mean square
SA	Surface area
SAE	Surface area enhancement
SAM	Self assembled monolayer
SAM	Scanning Auguer microprobe
SAW	Surface acoustic wave
SAXS	Small-angle x-ray scattering
SE	Secondary electron
SEM	Scanning electron microscope
SiO_2	Silicon dioxide (silica)
SnO_2	Tin oxide
SZM	Structure zone model
TCO	Transparent conductive oxide
TEM	Transmission electron microscope
TiC	Titanium carbide
TiO_2	Titanium dioxide (titania)
TLC	Thin layer chromotography
TPD	Temperature programmed desorption
UV	Ultra-violet
VASE	Variable angle spectroscopic ellipsometer
VLS	Vapor-Liquid-Solid
XRD	X-ray diffraction
ZnO	Zinc oxide

Chapter 1

Introduction

This thesis builds on the work of previous researchers who sought to understand and manipulate nanostructured thin films formed from obliquely deposited evaporated material. A number of tools have been used to characterize glancing angle deposition (GLAD) films, including microscropy, and ellipsometry, spectrometry and a range of chemical, electrical, and structural properties have been assessed. However, there remains much work to be done to answer remaining fundamental questions, including:

- What is the surface area of a GLAD nanostructure?
- How rough are GLAD films?
- What is the porosity of the structure?
- How do the GLAD columns grow and evolve?
- Are there ways that the thickness of these structures can be manipulated?

This thesis introduces and demonstrates several techniques that are new to the GLAD characterization toolbox and provides answers to the questions posed above. In this introductory chapter, a history and background into various nanostructure characterization and modification techniques is provided to set the stage for the work performed. The glancing angle deposition technique is also described. The chapter ends with an overview of the thesis as a whole.

1.1 Nanostructure Fabrication

Nanostructures are loosely defined as materials having been organized in zero, one, two, or three dimensions with features on the scale of nanometers to hundreds of nanometers. The natural world is replete with nanostructures, evolved over millions of years to provide their animal and plant hosts with protection, improved performance, and signalling characteristics. For example, the termite

wing consists of regularly spaced micron-size hairs, surrounded by a micro-array of star-shaped structures, which impart hydrophobicity and help to expel macro and micro droplets during flight through rainstorms [1]. Porous ridges and linking cross-ribs provide structural scaffolding and species-specific coloration for the wings of Lycaemodae family butterflies, while fine 180 nm diameter nanostructures on the sides of the hollow fibers which make up the surface of the Edelweiss flower help to reflect away harmful high-altitude ultra-violet (UV) radiation [2].

Biological nanostructures provide good models for researchers looking to mimic their optical, structural or functional properties. For example, the superhydrophobicity of the self-cleaning lotus leaf inspired one group to develop a technique for generating a heterogenous vertically aligned carbon nanotube (CNT) film with similar properties [3]. Another group created nanoscale pillars, coated with a mussel-mimetic polymer film, which replicated the strong, but reversible adhesion produced by the nano-spatulae covering the pads of gecko feet [4].

Beyond mimicking nature, nanostructures provide the basis for a range of systems and devices. For example, nanowires, with diameters from 2 nm to 200 nm and fabricated using techniques such as the adsorption and supersaturation of vapor onto a seed material have been employed in lasing, light-emitting diode and waveguiding applications [5]. Binary semiconductor wires, such as zinc oxide (ZnO), act as resonant cavities for light amplification, while nanowires fabricated from tin oxide (SnO_2) have shown to be excellent waveguides. Other groups have constructed micro-electro-mechanical and nano-electro-mechanical systems with nanotubes and nanoshells used for actuation and sensing feedback [6]. The size scales of nanostructures also make them excellent foundations for DNA (i.e., Deoxyribonucleic acid) and protein detection biodiagnostic devices. Arrays of functionalized metallic nanoparticles, as well as physically and chemically tailored organic CNT nanoforests, act as electrical sensors of captured target molecules [7], [8].

There are a range of methods available for fabrication of nanostructures. Since this thesis focuses on the development and characterization of columnar nanostructures, several techniques for creating similar structures are introduced here. Conventional optical and electron-beam lithography can be used to prepare micron and sub-micron scale structures such as pillars from a range of materials [9], [10]. To accomplish this goal, lithography uses a mask to apply a pattern to a sacrificial resist layer. The material layer of interest is etched through the resist pattern and this mask layer is then removed. The process can be repeated multiple times to create a three-dimensional (3D) structure or device. As an example, the gecko inspired surfaces described above were created through the electron-beam lithography of polydimethylsiloxane (PDMS) [4]. Other novel lithographic techniques for creating arrays of pillars from metals, semiconductors, and organics include patterning through surface nanosphere masks, extreme UV lithography, or nanoimprinting by hot-embossing or UV templating [11], [12]. These techniques produce high-aspect ratio structures with diameters

on the order of 100 nm, 50 nm and 10 nm respectively.

In addition to the use of lithographic techniques for forming structured nanofilms, self-assembly and chemical vapour deposition (CVD) have also been employed. Through the spontaneous organization and subsequent drying of solution-immersed soluble silica species, silica skeleton structures oriented parallel to the substrate surface have been prepared [13]. Other groups have produced self-assembled micron-high metal oxide columns with cross sectional diameters of tens to hundreds of nanometers [14]. Finally, CVD, which involves the reaction or decomposition at a substrate of one or more volatile precursors, has been used to produce porous columnar films and vertically-oriented nanotubes on patterned and unpatterned substrates [3], [15], [16].

Another process used for the creation of nanostructured materials, and a major focus for this thesis, is physical vapor deposition (PVD). As opposed to CVD, where a chemical reaction is used to form the thin film deposit, PVD involves the deposition of a thin film of material on a substrate surface through purely physical attachment of bombardment species [17]. These species are either evaporated or sputtered from a source. Resistively-heated thermal evaporation or electron-beam evaporation of a source material is usually conducted at low pressure in order to ensure line-of-sight, low-collision, and high purity deposition of the evaporated species. The disadvantage of evaporation-based PVD is that the component species of evaporated compounds may have different vapor pressures and thus different evaporation rates. To ensure stoichiometric deposition, a reactive gas of the deficient species may be introduced into the deposition chamber [18]. As an alternative, sputter-based PVD tends to preserve chemical composition. In many cases, this technique involves the ion bombardment of a source electrode target, also under vacuum conditions. The energy transfer from the impinging ions to the source material causes atoms to be ejected towards the substrate, upon which they deposit.

Movchan and Demchishin pioneered the structure zone model for evaporated PVD films, with Thornton extending the model to sputtered PVD films [19], [20]. They demonstrated that structure is inherent in crystalline and amorphous depositions when this technique is used. When deposited on substrates at temperatures (T) nearing the melting point (T_m) of the incident deposition material ($T/T_m > 0.5$) large grains are formed through bulk diffusion. As the substrate temperature is lowered ($0.3 < T/T_m < 0.5$) dense columnar grains are formed. Finally, at low substrate temperatures ($T/T_m < 0.3$) adatom diffusion is minimal and, because of shadowing mechanisms, any substrate roughness will result in tall columns separated by voids. Oblique deposition, which takes advantage of this shadowing behaviour at low substrate temperatures, will be discussed further as part of Section 1.3.

1.2 Nanostructure Characterization

Having created a nanostructured thin film, characterization of its properties is required. There are many techniques available to probe features such as surface structure, morphology, mechanical strength, electrical response and optical filtering. Several characterization techniques are described below. This thesis will make use of and build upon some of these techniques.

1.2.1 Morphological characterization by electron microscopy

One of the most widely used instruments for nanostructure characterization is the scanning electron microscope (SEM), used to image the faces of a three-dimensional specimen [21], [22], [23]. Originally demonstrated by Knoll in 1935, when an image of the individual grains of a polycrystalline sample were revealed, and developed into the "modern scanning electron microscope" by Sir Charles Oatley in the 1950's, the SEM harnesses the high resolution available from probing a surface with accelerated electrons [23], [24]. With an SEM, electrons are generated thermionically or through field-emission and accelerated, focused, and rastered over a specimen surface. Electron beam spot sizes of 1 nm - 10 nm are not uncommon, allowing for true characterization of nanostructured materials [22]. The aperture for this device is extremely small (on the order of milliradians), much smaller than that of an optical microscope, and allows for a large depth of focus even at low magnification (with magnification controlled by the raster size area). The extremely small spot size and large magnification range ensures that characterization of nanostructures over a range of size scales can be carried out.

The SEM now finds itself coupled with other instruments to provide visual characterization of a sample while a concurrent process is carried out. The SEM has been integrated with scanning tunneling microscopy, allowing for simultaneous imaging of nano or micro-scale surface features while resolving atomic scale topography [25]. Energy-dispersive spectrometer (EDS) units have also been added to SEM instruments to allow for the measurement of x-rays to evaluate the elemental constitution of multi-material samples [26]. Finally, later in this thesis we will discuss the integration of SEM and focused ion beam (FIB) instruments into a powerful dual-beam instrument for simultaneous manipulation and analysis of nano or microscopic samples [27].

Transmission electron microscopy (TEM) provides another imaging technique for nanostructured samples [28]. Because detection is of transmitted electrons as opposed to those which are scattered or reflected, the TEM samples must be extremely thin - ideally less than 100 nm. Samples are prepared by placing them on a conducting grid and then within the high-vacuum TEM column. In bright-field (BF) mode the electrons which travel directly through the sample and experience

limited scattering are detected at the image plane through an aperture located directly below the sample. Alternatively, in dark-field (DF) or Bragg diffraction mode, the detector collects diffracted electrons at a selected scatter angle. In addition, the TEM may also be used to collect the electron diffraction pattern through a region on the specimen by detecting the image at the back focal plane and not the image plane.

Recent advances in computing technology have allowed for the introduction of a new field: electron tomography [29], [30]. This technique involves using a TEM to acquire a series of two-dimensional projections from a rotated sample. A three-dimensional reconstruction is possible if the specimen was tilted about a single axis and accurate alignment to precision fiducial marks is performed. Some challenges to this technique include the fact that oversampling of low frequency data (e.g., the region directly surrounding the tilt axis), but undersampling of high frequency data must be compensated for with weighting filters. As well, while amorphous samples may employ bright-field imaging, dark-field imaging is required for crystalline samples which would otherwise exhibit diffraction artifacts [30].

1.2.2 Measurements of surface roughness and porosity

Techniques for determining the roughness and porosity of a nanostructured samples included atomic force microscopy (AFM), x-ray diffraction (XRD), and porosimetry. The atomic force microscope was introduced by Binnig *et al.* in 1986 and has been used extensively since then to study the surfaces of a range of materials with atomic resolution [31], [32], [33]. With AFM, the deflection between a feedback-controlled cantilever tip and the surface being rastered can be detected by assessing the location of a laser signal reflected off the back of the cantilever. AFM has the advantage of being usable under ambient conditions and on biological or organic samples, making analysis of natural porous nanostructures, such as the cellular scaffolds of algae, possible [29]. Additionally, AFMs can typically provide resolution to 1 nm, allowing them to image structures as detailed as DNA [29]. However, uncertainties in tip geometry and quality, as well as tip-sample interactions can make analysis of AFM images difficult.

Another technique used to assess the structural characteristics of a crystalline, polycrystalline, or amorphous sample is x-ray diffraction. X-rays were discovered in 1895 by Wilhelm Roentgen, a physicist investigating the nature of cathode rays. In a series of three publications he showed that, while x-rays are attenuated as they travel through a material and induce fluorescence in a luminescent screen, the ability to reflect, refract and diffract these rays remained inconclusive [34], [35]. Much later work by Max von Laue, Walter Friedrich and Paul Knipping in 1912 showed that, through careful shielding from ambient x-rays and those emitting from the glass walls of the x-ray tube, a

diffraction pattern could indeed be measured for a crystal specimen [36]. X-rays scatter from the electrons of the regularly spaced atoms in the crystal, interfering to produce a diffraction pattern indicative of the sample structure. Modern-day x-ray analysis consists of exposure of a mounted sample to x-rays from a range of angles. Intensity patterns are captured with a charge-coupled device (CCD) and analyzed by computer. To investigate specimen features on the nano- and micrometer scale, small-angle x-ray scattering (SAXS) is used. After traveling through a specimen, even one without crystallographic structure, some scattering occurs. The intensity pattern of the small-angle scattered X-rays reveals structural spacing information, such that particle shapes can be determined. SAXS has therefore become a fundamental tool for the study of molecular masses such as proteins [37].

As an alternative, porosimetry allows for the investigation of the overall roughness of specimens, regardless of aspect ratio. Porosimetry involves the adsorption of a gas or liquid onto a solid sample surface and the study of the resultant data. Indirect measurements of the testing environment or of a secondary parameter allow for an estimation of the adsorbate-adsorbant interactions, with further analysis resulting in a characterization of the porous nature of the sample. Nearly two centuries of adsorption research have enabled a thorough set of models for the investigation of porous nanostructures. A summary of the rich history of porosimetry is now provided.

Some of the earliest porosimetry studies investigated the adsorption of liquids into powders, such as water into dry sand. In 1822, Pouillet noted that this water-sand interaction was an exothermic reaction, with Gore (1894) proposing that the heat released was dependent on the surface area of the powder and Gurvich later (1915) suggesting that the nature of the liquid played a role [38], [39], [40], [41]. Others, such as Chappuis and Kayser in 1881, independently proposed a link between experimental chamber pressure and the amount of gas or vapor adsorbed into a sample [38]. This was the beginning of the recording of isotherms (gas adsorption versus chamber pressure at a constant temperature) for a variety of adsorbate-adsorbant pairs.

By 1910, scrutiny of these isotherms was beginning to yield insight into the nature of the solids that were being studied. Within the next decade, three researchers would provide profound insights into the nature of adsorption in porous materials. In 1911 Zsigmondy proposed the theory of capillary condensation, which theorized that small pores could fill with liquid at a pressure lower than the normal vapor pressure of the adsorbate [42]. From 1914 to 1916 a young student at the German Karlsruhe Technical Institute, Michael Polanyi, wrote papers on the attractive forces between the adsorbant surface and adsorbate gas [43]. It was finally in 1916 that Langmuir then proposed models for monolayer and multi-layer adsorption on a variety of different surfaces. It is his description of an adsorption isotherm for a single monolayer on a porous surface that has become a cornerstone of porosimetry [44].

One of the most important contributions to the field of porosimetry came in 1938 with the publication of the Brunauer-Emmett-Teller empirical model of multilayer adsorption on a surface [45]. They proposed a model to describe the interaction not only of the adsorbate gas with the surface upon which it binds, but also with the molecules within the multilayered adsorbate. This technique will be explored further in this thesis. Building on the work by Brunauer, Emmett, and Teller was the proposal by Barrett, Joyner and Helenda (BJH) in 1951 that isotherms could be analyzed to determine characteristics of the pores making up a permeable solid [46]. The BJH theory has been widely employed to study not only powders and thin films, but also structured solids such as nanofibers and nanotubes [47], [48]. This thesis will also explore the use of the BJH theory for nanostructured thin films.

1.2.3 Optical characterization of nanostructures

Nanostructured thin films exhibit interesting optical and dielectric properties which can be harnessed for photonic, solar energy conversion, plasmonics, and electrical sensing applications. For example, nanostructures are often birefringent, with different physical properties (adsorbance, refractive index) in different dimensions [49], [50]. Nanostructures also exhibit periodicity, as when composed of stacks of materials or composed of arrays of nanoparticles [51], [52]. These structures can therefore be used to create polarizers, beam splitters, novel optical filters, and liquid crystal supports [53], [54].

The ubiquitous spectrometer, applied in fields from astronomy to chemical analysis, has been employed numerous times for the optical characterization of various obliquely deposited nanostructures [55], [56]. The spectrometer (known in other incarnations as the spectroscope or spectrophotometer), originally developed by Joseph von Fraunhofer in 1814 for solar applications and Bunsen and Kirchhoff in 1859 to capture and analyze the emission spectra of various chemical processes, has evolved with the development of the camera and now charge coupled devices (CCDs) [57], [58]. Current spectrometers involve a light source, sample mount, diffraction grating (or two), and CCD detector for measurement of the near-IR, visible, and UV transmission and reflection frequency response of a sample [59], [60]. Optical components may also be added to the beam path to filter, alter the polarization of, attenuate, or split the incoming signal depending on the exact analysis required.

The transmission and reflection frequency responses of GLAD nanostructures not only provide indications of which applications these structures might be suited for, but also reveal structural properties. A variable angle spectroscopic ellipsometer (VASE) measures the response of polarized light, incident from a range of angles, over a broad band of wavelengths, to a material surface [61]. Using empirical models of common materials and structure types, the measured amplitude and phase

difference spectra are then processed to estimate sample properties such as film thickness, density, porosity, crystallinity and structure. Thin nanostructured films consisting of MgF_2, SiO_2, TiO_2, Si, ITO, have been characterized using the VASE technique [62], [63], [64], [65], [66].

1.3 Overview of Glancing Angle Deposition

Experimental parameters, such as deposition angle, substrate rotation, evaporation rate and chamber pressure, make the physical vapor deposition technique very versatile. As indicated by the work of Movchan and Demchishin and those that followed, the characteristics of materials deposited by PVD are strongly affected by deposition parameters such as substrate temperature, vapor flux rate, and vacuum conditions. What was realized as early as 1886 by Kundt, and followed by Knig and Helwig in 1950 and Holland in 1953, was that the geometry of the deposition system was also a key factor in determining the structure imparted to the deposited film [67], [68], [69]. These researchers experimented with PVD onto obliquely held substrates and observed that the resultant films exhibited birefringent and anisotropic behaviour. It was once Nieuwenhuizen and Haanstra confirmed that the obliquely deposited films were made up of uniquely slanted columnar structures that the link between the source-to-substrate geometry and nanostructured morphology was finally cemented [70].

The column growth seen when the substrate orientation is held at an oblique angle relative to the deposition flux is due to several factors. The first of these is the initial nucleation of the incident vapor flux. When, as described by Movchan and Demchishin, the substrate temperature is kept low relative to the melting point of the deposition material impinging atoms will 1) attach to the substrate, 2) have energy enough to move to an energetically favorable position such as a neighboring nuclei, and 3) experience little further adatom mobility. Secondly, through use of techniques such as thermal or electron beam evaporation coupled with long throw distances, a well-collimated, low-scattering angle beam of incoming atoms ensures extremely directional deposition. Finally, large nuclei will capture incoming flux preferentially, as regions behind these protrusions are shadowed. This line of site deposition is known as ballistic shadowing and ensures that film growth will occur predominantly on top of the non-shadowed nuclei. As film deposition progresses, nuclei will grow into columns, with the direction of growth tilted in the direction of the incoming evaporant. Figure 1.1 illustrates the concepts of initial nucleation, self-shadowing, and column growth that are charactersistics of this oblique angle deposition (OAD).

With the introduction of computer controlled substrate rotation, the columnar growth can be guided and shaped [71]. This glancing angle deposition technique enables the fabrication of three-

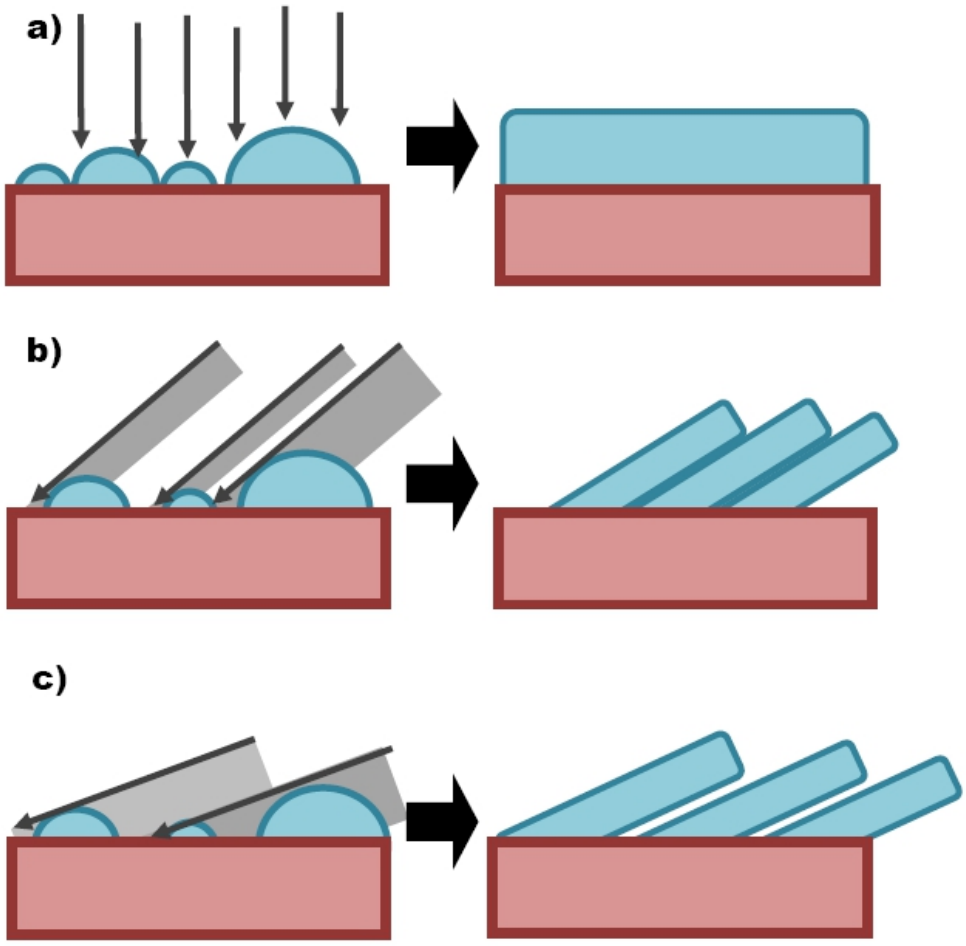


Figure 1.1: Initial nucleation and effect of deposition angle on nuclei self-shadowing. At (a) normal incidence, impinging flux will deposit on and between nuclei forming a thin uniform film, while for (b) moderate deposition angles some or portions of small nuclei will be shadowed forming a densely packed columnar structure and (c) at oblique deposition angle whole nuclei will be shadowed by larger neighbors forming a columnar nanostructure.

dimensional structures such as helices, zig-zags, and slanted posts by controlling the deposition angle (α) and the rotation angle (ϕ). A crystal thickness monitor (CTM) allows for the depositing film thickness to be used as feedback to the control system such that precise control of the motor angles is achieved. This controlled PVD technique is referred to as glancing or oblique angle deposition and is currently employed by over one hundred research groups. The apparatus used to create these obliquely deposited nanostructures is shown in Figure 1.2.

Using the GLAD method, thin films with a variety of nanostructures can be created. Examples of such structures are shown in Figure 1.3. As described above, by depositing film at an oblique angle, without rotating the substrate, slanted posts typically result. However, when substrate rotation is

employed, the flux hitting all areas of the substrate is averaged, resulting in upright columns. Slow substrate rotation results in the formation of helical structures, while faster substrate rotation allows these chiral columns to degenerate into vertical post structures.

Column evolution depends on the deposition material, initial nucleation conditions, and process parameters. Initial nucleation is a random process which results in both a variety of nuclei diameters, but also growth locations [72], [63]. Competition between nuclei means that only some survive to form columns. Once columns are established, further competition also occurs between columns, with some dying out while others grow [73]. Pre-seeding the substrate with an array of evenly spaced pseudo-nuclei through processes such as electron beam lithography, optical lithography, and deposited nanospheres can lead to a reduction in competition and a regular array of GLAD columns [74], [75], [76], [77]. In addition to column competition, cross-sectional anisotropy occurs when the incoming flux intersects the growing columns from predominantly one direction. To counter the broadening that results in fanlike shapes a technique known as PhiSweep was introduced by Jensen *et al.* [78]. In its simplest form, material is allowed to deposit at one angle until a set thickness is achieved, then the substrate is rotated by an angle (about the ϕ axis) such that material now impinges the growing columns from another angle. This process of sweeping the deposition angle, followed by discrete deposition, allows for a more uniform column cross-section to be achieved.

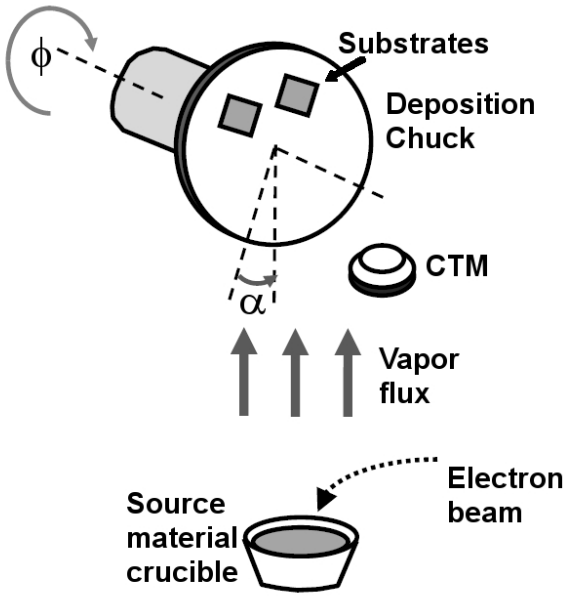


Figure 1.2: GLAD apparatus employs physical vapor deposition to deposit a collimated vapor onto a tilted and rotating substrate.

GLAD provides a nano/micro structure that allows for integration of nanoscale properties into macroscale devices. Examples of GLAD applications include:

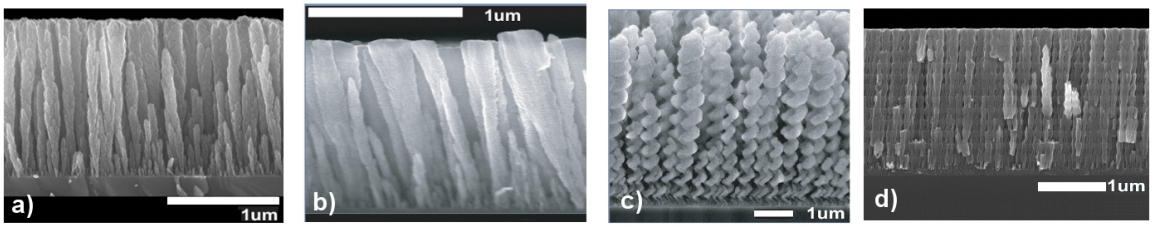


Figure 1.3: Representative structures created using the GLAD technique.

- **Photonic devices:** Control over the columnar properties in multiple directions allows for optical filters in two and three dimensions to be created. Exploitation of the column width, film density, and optical property relationship allows for a full range of photonic applications to be explored. Periodic and graded changes to the column widths as a function of column height can be tailored such that incident light interacts with the deposited nanostructure. For example, graded width SiO_2 columns impart a graded density profile (and thus graded refractive index) to the overall structure resulting in an antireflection coating [56], [79]. Techniques such as serial bideposition (SBD) can be used to stretch the columnar cross-sections and enhance the in-plane birefringence of the GLAD columns by employing rapid, 180^o stepwise substrate rotations [64], [80]. A Bragg relector, commonly fabricated by altering the density of deposited film, is made with the GLAD method by altering the widths of the deposited columns every several hundred nanometers [81]. By sinusoidally altering the column widths of an optically interactive material a rugate filter can be created that presents a bandpass filter to incoming light [82]. Another important structure, the helical column, is produced through slow substrate rotation. The interaction of light with the periodic rotating column produces a circular Bragg effect which can be used to create left or right circular polarizers [83], [84], [85]. In addition, a three-dimensional photonic crystal (PC) can be created by ensuring a periodicity not only in the film thickness direction but also in the substrate plane. This has been accomplished by pre-seeding the substrate prior to GLAD deposition and then fabricating square spirals [76].

- **Solar energy devices:** The high surface to footprint area ratio of GLAD films enable photovoltaics (PVs), photoelectrochemical (PEC) fuel cells, and battery devices to benefit from including these nanostructures in their design. Because of the large interfacial area provided by these nanorod arrays, high rate electron-hole conversion are achievable in GLAD-based solar cells. Vertically aligned nanoelectrodes of transparent and hole conducting indium tin oxide (ITO) have been embedded into active electron conducting layers to form polymer-based solar cells with enhanced power conversion efficiency [86]. Similarly, TiO_2 nanocolums have been used as the photoanode scaffolding layer to hold light-absorbing dye molecules in dye-sensitized solar cells (DSSC) [87]. The higher surface area provided by GLAD nanos-

tructures also enables less material to be used for fabrication of catalytic supports in fuel cells. For example, costly Pt has been sputtered on Ti nanoposts to obtain high current density electrocatalysts [88]. Additionally, portable electronics require microbatteries which maximize the electrode area relative to the footprint area. The GLAD process has been used to create nanostructured batteries with capacities double that of their thin film counterparts [89].

- **Microfluidic and Chromatographic channels:** The GLAD process is being applied in areas of chemical and biological research. The ability to pattern large (micron size) areas with seeds and subsequently deposit GLAD films enables regularly spaced arrays of nanocolumns to be constructed [75]. Patterned sealed channels of GLAD columns have been fabricated, with potential uses in chemical filtering, reaction chamber, and lab-on-chip applications [90]. The anisotropic properties of engineered GLAD films also provide a separation medium along channel-like tracks which enables its use in ultrathin-layer chromatography [91], [92].

- **Sensing:** The range of materials that can be used to fabricate nanostructures using the oblique angle deposition process mean that many different sensing applications can be targeted. For example, dielectric GLAD films have been fabricated for use as high speed capacitive sensors [93]. In addition, the morphological control enables GLAD films to be incorporated into mechanical sensing devices. To illustrate this, helical and zigzag nanorods with elastic properties have been used for pressure sensing [94]. Surface functionalization has also opened up the field of biosensing. Detection of proteins and large molecules is possible by coating columnar structures with biocompatible films [95].

1.4 Motivation and Scope of Research

One of the exciting properties of GLAD nanostructured films is that they exhibit features at the nanometer scale (crystallinity, porosity, roughness), at the micrometer scale (columnar features), and at the millimeter scale (substrate size). Given this large size range, it is natural that one would wish to not only understand the properties of the GLAD film over this range, but also to determine how to modify and control them. The following goals were therefore developed and explored in this thesis:

- To explore techniques to investigate and quantify the structural characteristics (such as surface area, roughness, porosity, and growth scaling) of GLAD thin films over a size range of nanometers to microns, and

- To manipulate these structural characteristics over a size range of microns to millimeters.

To address the first goal, two techniques were employed. The first of these was gas adsorption porosimetry, which enabled the measurement of surface characteristics of several porous structures. The second was focused ion beam tomography, which allowed for GLAD nanostructures to be segmented, imaged, and virtually reconstructured.

To address the second goal, a novel macroscopic shadowing technique was introduced to the GLAD toolbox which can be used to create graded height nanostructures for use in a range of applications. The size scale over which the structural characteristics of GLAD films were investigated and controlled is shown in Figure 1.4.

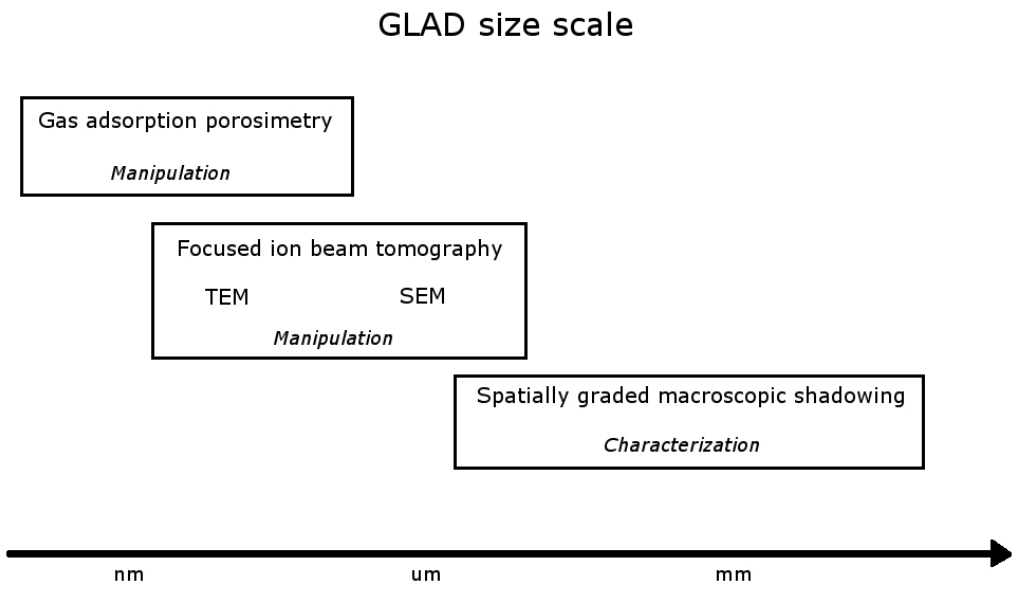


Figure 1.4: Characterization and manipulation of GLAD thin films using three different techniques that cover a size scale from nanometers to millimeters.

The remainder of this thesis begins with a description of my research into measuring the surface area of a GLAD film, a parameter not yet explored fully in the literature. Chapter 2 describes the fabrication and fundamental analysis of thin films fabricated as part of the density, surface area, porosity, and pore size distribution characterization studies. Chapter 3 then describes in detail how gas adsorption porosimetry is employed to estimate the surface area of porous powders and thin films. I investigate the relationship between film deposition parameters and surface characteristics. A description of common techniques for estimating the surface area is provided. Gas adsorption onto porous samples is discussed in detail and the theory behind estimating surface area from the acquired adsorption data is given. Following this, I discuss the measurements that I performed to calculate the surface area of a number of different GLAD samples.

In Chapter 4, porosimetry continues to be explored, but in this case it is its application to the measurement of pore size, roughness, and pore volume of structured samples which is described. The theory which allows for the pore size distribution to be gleaned from the adsorption data is given. This theory, along with empirical calibration data provided by my collaborator Dr Matthias Thommes of Quantachome Inc., is then applied to determine the pore size distribution and roughness of a range of GLAD thin films. A discussion of the implications of these findings concludes the chapter.

With Chapter 5, I continue to investigate the characteristics of nanostructures, but in this case a dual-beam FIB-SEM instrument is used to probe a representative GLAD sample. In this chapter some background and recent progress in the field of FIB tomography is described. Because of its ability to sequentially segment a sample with an accuracy of approximately 10 nm, the FIB is found to be especially appropriate for the investigation of GLAD samples. The procedure used to segment and image the GLAD nanostructure, carried out in collaboration with Dr Douglas Vick of the NRC-NINT, is provided. The chapter also describes the image processing, grain detection, and tomographic algorithms used to reconstruct and analyze the segmented data. The growth scaling properties of the sample are investigated.

Having investigated the surface and volumetric characteristics of GLAD samples, we turn our attention to techniques used to modulate the growth of GLAD samples. Chapter 6 describes a new technique used to alter the thickness of a GLAD sample across is breadth and width. This ability allows for devices with position dependent characteristics, such as the optical filter described herein, to be developed. The experimental technique is described, application results are discussed, and simulation work is provided to highlight further directions for which this technique could be used.

In the final chapter of this thesis, a summary of the work undertaken is provided. A discussion of the impact of the work performed here is given and future research directions are suggested.

1.5 Major Contributions of this Work

The major contributions of this thesis work include the following:

- Use of high-sensitivity krypton gas adsorption to measure, for the first time, surface areas and pore size distributions of GLAD metal-oxide nanostructures.
- The finding that peak surface areas occur in the mid-range of deposition angles, not at the most oblique deposition angles that normally characterize GLAD thin films.

- The determination that, despite columnar broadening as film thickness increases, the surface area of GLAD films rises linearly with thickness. This implies a constant differential surface area throughout the film.
- The finding that GLAD surface area can be tuned not only through choice of deposition angle and film thickness, but also from rotation characteristics.
- The measurement of inter- and intra-column porosity that extends to the nanometer range, for GLAD structures deposited from 45^o to the most oblique angles.
- The contribution of a focused ion beam milling procedure and series of image processing algorithms to slice and reconstruct a GLAD film.
- The determination, using a single sample, that GLAD column diameter, spacing, and density follow power-law scaling relationships. This had been previously determined from the construction and analysis of multiple samples.
- The tracking of the growth of individual GLAD columns, using the FIB milling data, and the finding that individual GLAD columns with large numbers of branches appear to survive longer than their neighbors.
- The construction of the first graded thickness helical GLAD nanostructures using a novel macroscopic shadowing technique.
- The demonstration of these graded thickness structures as frequency-tunable polarization-selective photonic filters.

Chapter 2

Nanostructure fabrication and analysis of fundamental properties

2.1 Introduction

This chapter describes the fabrication and fundamental analysis of thin films fabricated as part of the surface area, porosity, and pore size distribution characterization studies. Specifically, exploration of the thickness and density properties imparted to nanostructures deposited using glancing angle deposition were carried out on materials commonly used for photonic, sensing, and display applications. These parameters were investigated as they are essential parameters for the work discussed in the rest of the thesis. Nanostructures of amorphous titania (TiO_2), amorphous silica (SiO_2), and polycrystalline indium tin oxide (ITO) were fabricated over a range of deposition angles. Vertical posts, helical structures, and multilayer films were investigated. Correction for humidity in the mass measurements is also described. [1]

2.1.1 Materials

Titania was chosen as one of the primary materials of study in this thesis because of its promising properties and use in a number of applications. It is commonly used as a white pigment in toothpaste and paint because of its brightness in powder form. Titania also supports cell and tissue growth and is being explored as a biocompatible interface material for cells, DNA and proteins in cell-based assays, biosensors or microfabricated medical devices [96], [97]. Found in most physical blocker sunscreens, titania is also a strong absorber of UV rays [98]. Additionally, TiO_2 is also a photocatalyst, such that, when illuminated, the material will generate electron-hole pairs useful for reactions at its surface. It is therefore used for a number of environmental sensing, photovoltaic, and purification applications [99], [100]. Steel structures coated in TiO_2 are corrosion resistance due to the ceramic protective barrier and also to water-splitting (and corrosion reducing) protection imparted by the coating when illuminated by UV light [101]. Titania is also generally transparent

[1] Portions of the work documented in this chapter have been published in K.M. Krause, *et al.*, "Surface area characterization of obliquely deposited metal oxide nanostructured thin films", Langmuir 26 (2010) 4368-4376.

as an annealed thin film and has a high refractive index. This makes titania an excellent material for photonic devices such as optical interference filters and antireflection coatings [102], [103], [104], [105]. As well, humidity sensors are constructed from titania because of its hydrophilicity and material stability under exposure to water [93]. Finally, titania is a material of interest in a variety of processes because its crystal structure, and thus optical and material properties, can be controlled by deposition conditions and thermal annealing; it forms a non-crystalline thin film when deposited at low temperatures, but can form crystalline phases such as anatase or rutile when annealed at high temperatures [106]. For example, because of its high temperature stability, rutile is used in high temperature ($> 800\ ^{o}$C) automotive air-to-fuel ratio control oxygen gas sensors [107].

Another common metal oxide, silica, is found in a wide range of devices and applications. Silica is a common insulator, used in the semiconductor industry as the insulating layer between conducting and semiconducting layers. In nanostructured form, it finds itself used for scaffolding in the field of bioceramics. For example, new bone formation has been induced on silica-based porous surfaces, an exciting advance in a field where bone replacement from donors is problematic [108]. Silica scaffolding has also been used to align liquid crystals for display technologies and as a basis for thin layer chromotography (TLC) plates [109], [91]. Supported metal particles on porous silica have attracted attention as catalysts in applications such as the conversion of methane and carbon monoxide to higher hydrocarbons [110], [111]. Finally, thin film, porous and nanostructured SiO_2 have been used in resistance and capacitance based humidity sensing because of its dielectric sensitivity to water [112], [113].

ITO is composed of In_2O_3 and SnO_2 and is transparent in thin film form depending on fabrication conditions and post-processing annealing [114], [115]. In addition to transparency, it also possesses the useful property of being electrically conducting. This transparent conductor is therefore commonly used as the transparent electrode in light emitting diodes, flat panel displays, and electrically activated smart windows [115], [116]. It has also been found in flexible, plastic solar cells which use ITO as the metal electrode layer [117]. Alignment of liquid crystals, a requirement for liquid crystal displays, has also been achieved using ITO as transparent scaffolding [118]. In addition, ITO is also highly reflective to infra-red radiation, so is used in thin film heat mirrors [119].

2.1.2 Structures

The three structures studied for this work - vertical posts, helical films, and multilayer stacks - were selected for several reasons. The first is that all are produced using continual substrate rotation during the glancing angle deposition process. Films created without substrate rotation not only have an inherent slanted morphology, but also exhibit a thickness gradation across the deposition surface. Substrate rotation significantly reduces the thickness variation across the substrates affixed to the deposition chuck and ensures symmetric columnar structures about the vertical axis. The

second characteristic of note for these structures is that while vertical post films are perhaps the most widely studied and deposited of the GLAD nanostructures, they are actually a subset of helical films and multilayer stacks. Vertical post films are created by increasing the rate of substrate rotation with respect to deposition rate, causing otherwise chiral films to degenerate into straight columns. Multilayer stacks are vertical films for which the deposition angle is changed either continuously or discretely during the deposition process. A full study of vertical post film characteristics is not complete without some analysis of helical and multilayer films as well.

The multilayer structures studied here fall into two categories: 1) simple discrete stacks of alternating columns of different densities, each of which is deposited at a different deposition angle, and 2) undulating films composed by sinusoidally altering the density of the column through continuous modulation of the deposition angle. Both are interesting not only from a morphological, but also from an optical point of view. When composed of an optical transparent material, such as titania, the simple stack is known as a Bragg stack and the sinusoidally varied columnar structure is known as a rugate film. Both Bragg and rugate thin film filters, created using not only with oblique deposition techniques, but also electro-chemical etching and electro-chemical anodization, have been studied extensively for use in optical devices such as filters and reflectors [104], [120], [121], [122]. Optical gas sensors based on these structures have also been proven, with the concentration of gas detected by shifts in the optical properties of the structures. For example, Salem *et al.* has demonstrated ethanol detection by silicon rugate films [121]. Other groups have demonstrated that photonic bio-sensors are possible by functionalizing the multilayer surface with amine-based chemical groups [123].

The optical properties of Bragg and rugate films depend on the nature of the repeating refractive index profile. For obliquely deposited films, the refractive index profile in turn depends on the deposition angles selected during the fabrication process. The deposition angle determines the spacing between and cross-sectional area of the growing columns and therefore density of the overall film. As the deposition angle changes the density profile also changes, resulting in a refractive index variation. The resultant interference effects lead to a transmission stop band whose bandwidth depends on the pitch of the refractive index variation. The stop band center wavelength (λ_{SB}) depends on the average refractive index of the film (n_{avg}) and the pitch (Π) [124]:

$$\lambda_{SB} = 2n_{avg}\Pi. \tag{2.1}$$

The surface properties of these multilayer films were studied for use in future optical interference filter work as well as to determine the impact of humidity on the properties of GLAD metal oxide films. This humidity analysis is discussed in Section 2.4.

2.2 Sample fabrication

Nanostructured thin film samples were required for the characterization studies described in Chapters 3 and 4. Samples were required for SEM analysis, porosimetry measurement, optical testing, and electro-chemical investigation. A range of vertical post samples were fabricated at selected deposition angles. Helical films were also constructed to assess the effect of morphological features on surface properties. In addition, multilayer stacks, fabricated by modulating the deposition angle during the fabrication process, were deposited to compare their properties with those of nanostructures deposited at a single deposition angle.

To achieve sufficient total surface area for krypton adsorption measurements, samples with large footprint areas were prepared. For TiO_2 and SiO_2 films, a total of half of a $410"$ (100 mm) diameter factory polished silicon wafer (p-type $< 100 >$, University Wafer) was cleaved into N_{pieces} pieces of 1 cm x 1.5 cm prior to deposition (pieces of this size were required for the porosimetry measurements described in Chapters 3 and 4). For ITO films, a $110"$ x$110"$ ITO coated glass substrate, also cleaved into small pieces, was used as the deposition substrate (8-12 Ω / $\square$, Delta Technologies Ltd.). Prior to deposition, the weight of these sample pieces ($m_{initial_i}$, where i = 1 ... N_{pieces}) were individually measured on a Sartorius MC5 micro-balance and then attached with tape to a flat circular deposition chuck. Witness samples for SEM analysis where also attached to the chuck. In addition, in the case of films used for optical measurements, transparent fused silica squares of 1"x1" were included on the deposition chuck. The chuck and sample pieces were then photographed with a calibration ruler for use in subsequent footprint area calculations (at a resolution of approximately 10 mm pixel$^-1$). An image of a prepared chuck is shown in Figure 2.1.

Once sample preparation was complete, depositions were carried out on an electron-beam physical vapor deposition evaporation system (AXXIS, Kurt J Lesker). This system is computer controlled and allows for manipulation of two substrate orientation motors: one to set the deposition angle (α) of the substrate chuck with respect to the impinging evaporant flux and another to rotate the substrate (ϕ) as a function of deposition rate. Computer control relied on the deposition rate information provided by a quartz crystal microbalance oscillator (QCM) (Maxtex, SC-105 Aluminum at 6 MHz).

For these depositions TiO_2 (Cerac, 99.9% pure rutile), SiO_2 (Cerac, 99.99% pure), and ITO (Cerac, 91:9 In_2O_3/SnO_2 99.99% pure) were used. To promote the formation of stoichiometric TiO_2, the material was evaporated under partial pressure of O_2 gas; by varying the O_2 gas flow, the pressure was typically kept at 8x10^{-3} Pa. In contrast, SiO_2 and ITO were deposited without the addition of O_2 gas and were deposited at pressures of 5x10^{-4} Pa and 2x10^{-3} Pa, respectively. In all cases, the system base pressure was below 1x10^{-4} Pa. Substrates were not dynamically temperature

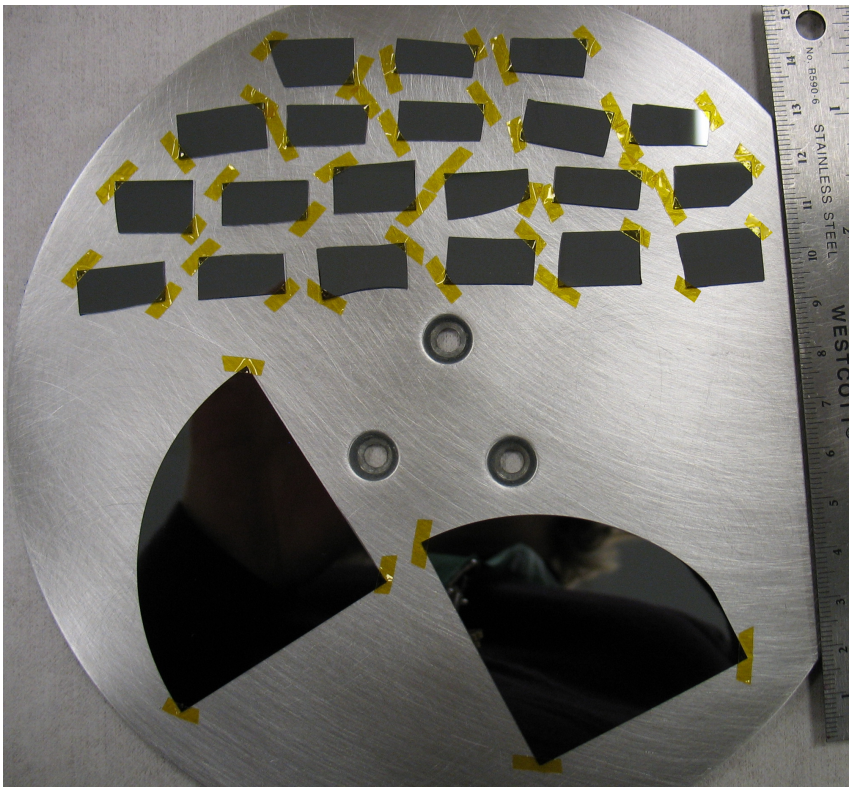


Figure 2.1: Deposition chuck prepared for porosimetry study.

controlled during the deposition process and reached a steady-state 70 ^{o}C during a deposition [125].

The morphology of the deposited films was determined by the deposition angle, rate of incident flux and rotation speed. Vertical post films of thickness 0.5 μm to 5 μm were deposited at α from 0^o to 85^o. A deposition rate of approximately 1 nm s^{-1} and a substrate rotation rate of 0.1 rev nm^{-1} of deposited film were used. In contrast to the vertical post films, a substrate rotation rate of 0.0033 rev nm^{-1} of deposited film was used to achieve helical films.

Multilayer stacks were also fabricated using TiO_2 as the deposition material. Rugate films with several deposition angle ranges were fabricated both to determine their surface properties, but also to be used for humidity calibration estimation. The four rugate films fabricated for this study had deposition angle ranges from 30^o to 80^o, 45^o to 80^o, 50^o to 80^o, and 60^o to 80^o. Bragg stacks of alternating layers, deposited at 60^o and 80^o were also fabricated. [2] As mentioned above, for these depositions, both silicon substrates (for porosimetry analysis) and fused silica substrates (which is transparent and therefore optimal for optical analysis) were used. Each rugate film consisted of 14 cycles of high to low to high refractive index change, with designed periods of 154 nm, 160 nm, 162 nm, 170 nm (actual periods measured using SEM images post-deposition). The deposition and

[2]These Bragg stacks were also used in the thesis work of Matthew Hawkeye (2010).

A) Rugate α = 30^o to 80^o			B) Rugate α = 45^o to 80^o			C) Rugate α = 50^o to 80^o			D) Rugate α = 60^o to 80^o		
t (nm)	α (^{o})	ϕ (^{o})	t (nm)	α (^{o})	ϕ (^{o})	t (nm)	α (^{o})	ϕ (^{o})	t (nm)	α (^{o})	ϕ (^{o})
0	80	0	0	80	0	0	80	0	0	80	0
20	77	703	20	78	703	20	79	703	20	80	703
35	66	1274	35	71	1275	35	72	1274	36	75	1274
48	50	1746	49	60	1746	49	62	1746	51	69	1746
60	36	2172	62	50	2172	62	53.8	2172	65	63	2172
72	30	2585	75	45	2585	75	50	2585	78	60	2585
84	36	3011	87	50	3011	88	54	3011	92	63	3011
97	50	3483	101	60	3483	102	62	4383	107	69	3483
113	66	4054	118	71	4054	199	72	4054	125	75	4054
132	77	4757	138	78	4757	140	79	4757	147	80	4757
154	80	5529	160	80	5529	162	80	5529	170	80	5529

Table 2.1: Design deposition and rotation angle parameters for rugate films. From each step to the next the change in thickness is monitored using the CTM, with the end of step target deposition and substrate rotation angles achieved by the interpolation routines of the deposition control software.

rotation angles as a function of design thickness, for one period, are provided in Table 2.1 (Note that the minimum α does not occur at exactly $\frac{1}{2}T$ because CTM-to-α deposition ratios have been incorporated into the design file). Design of these deposition parameters was guided by the work described in [126], [127]. Bragg stacks consisted of 12 cycles of α = 60^o and 80^o layers with designed periods of 233 nm (73 nm of a 60^o layer and 160 nm of a 80^o layer). Deposition rates of approximately 10 nm s^{-1} were targeted. With the addition of O_2 gas, the pressure was typically kept at 8x10^{-3} Pa.

Following deposition, the TiO_2 and ITO samples were annealed in order to ensure stoichiometric TiO_2 films with stable optical properties and ITO films with stable optical and electrical properties. The TiO_2 films were annealed at 100 ^{o}C for 24 hr, a low enough temperature that the crystallinity was not changed from amorphous to polycrystalline or crystalline [128]. The anneal was also used to enhance and stabilize the optical transmission of the film, an indication that oxygen deficient sites are eliminated and stoichiometry has been achieved [129], [130], [131]. ITO samples were annealed at 300 ^{o}C to 350 ^{o}C for 2 hr - 3 hr, with a 10 ^{o}C min^{-1} temperature ramp-up and down. These preparation procedures are standard practice in our laboratory.

Immediately following deposition in the case of SiO_2, and after annealing in the case of ITO and TiO_2, the final mass of each sample piece (m_{final_i}) was weighed in order to calculate the total film weight. From previous work done by Steele *et al.* the worst-case water adsorption in very obliquely deposited films was calculated to be 6% of the total film mass [132]. However, the amount of water adsorbed (m_{water}) onto the film surfaces over a full range of deposition angles was estimated and compensated for using an optical-based method that is described in Section 2.4. In addition to the uncertainty in mass due to water adsorption, a measurement repeatability error, calculated

by measuring the mass of a sample pieces ten times, of 0.02 mg per sample piece was taken into account.

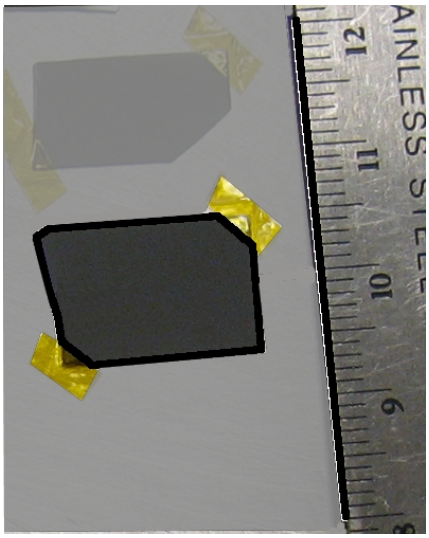


Figure 2.2: Photograph of deposition chuck is analyzed by image processing software to determine footprint area of sample pieces.

To calculate film densities (see Section 2.6), the film mass not only needed to be measured, but also sample footprint area (FA) and film thickness (t). Sample footprint areas were determined using the freeware ImageJ software package (1.36b) and the calibrated sample chuck photograph as illustrated in Figure 2.2 [133]. Repeatability error of 5% per sample piece footprint area was taken into account.

Witness samples for TiO_2, SiO_2, ITO film characterizations were deposited on silicon substrates. A conductive coating was required for imaging of the SiO_2 samples, but none was required for the TiO_2 and ITO samples. The witness sample thicknesses were measured using a Hitachi Field emission S-4800 Scanning electron microscope (SEM). Images of selected TiO_2 vertical post films are shown in Figure 2.3. Selected SiO_2 vertical posts, coated with a thin gold layer, are shown in Figure 2.4. Side and plan views of GLAD ITO films, deposited on Si wafers, are shown in Figure 2.5.

The images shown in Figures 2.3 through 2.5 reveal that as the deposition angle increases, the spacing and mesoporosity between the GLAD columns also increases, as does the diameter of the columns. At first glance, films deposited at low deposition angles appear to be quite dense; however, an analysis of top-down SEM images reveals that porosity and surface roughness are present above a deposition angle of 60^o. Equally important is the column broadening that is apparent at high deposition angles; this broadening, along with porosity, plays into the establishment of the specific surface area (discussed in Chapter 3). As illustrated in Figure 2.4, SiO_2 GLAD films do not appear

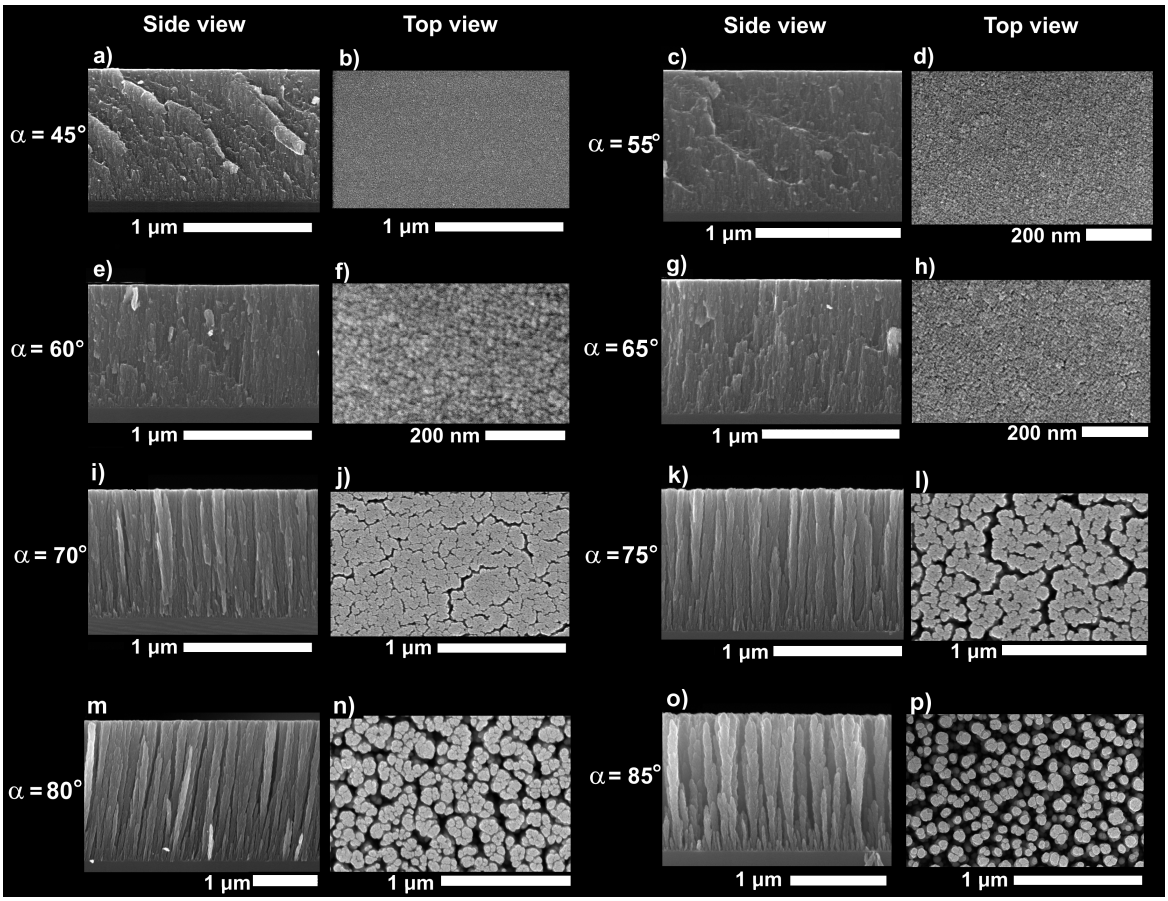


Figure 2.3: Cross-section and plan view SEM images of selected TiO_2 vertical post films deposited at (a), (b) 45^o, (c), (d) 55^o, (e), (f) 60^o, (g), (h) 65^o, (i), (j) 70^o, (k), (l) 75^o, (m), (n) 80^o, (o), (p) 85^o.

to be fundamentally different from TiO_2 and exhibit similar columnar properties as a function of deposition angle. In contrast, the ITO films are quite different: rougher column walls and bifurcations are characteristics of these structures as seen in Figure 2.5.

The helical GLAD films, fabricated from TiO_2 are shown in Figure 2.6. One can see that as the deposition angle is increased, both the spacing between the chiral columns and the width of the columns themselves increase. A comparison of the density and surface area of helical and vertical post films will be described below and in the following chapter.

SEM images of the multilayer rugate films are shown in Figure 2.7. As the initial deposition angle is increased from 30^o to 60^o the films become denser, with increased column definition. The Bragg stacks are shown in Figure 2.8 and exhibit some interesting properties at the top and bottom of the films as seen from the inset images. This will be discussed further in Section 2.6.4.

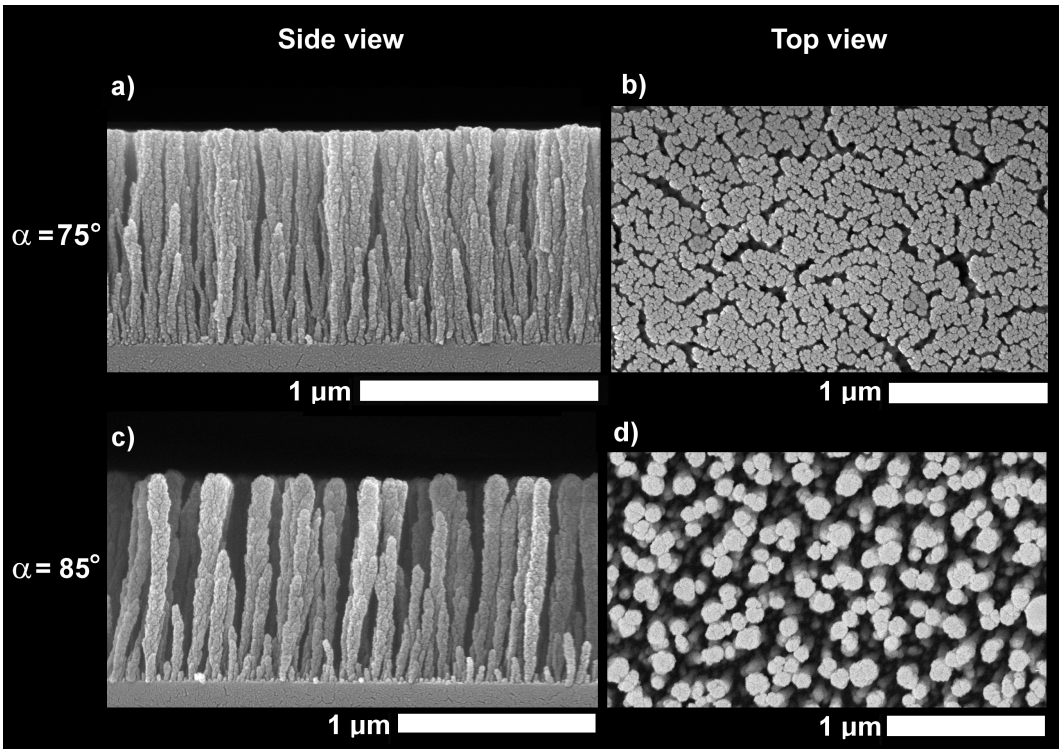


Figure 2.4: Cross-section and plan view SEM images of representative SiO_2 films deposited at (a), (b) 75^o and (c), (d) 85^o.

The uncertainty in film thickness was calculated based on the maximum and minimum thickness of the top-of-film envelope profile. Film thickness variability was estimated to be 4% of the average film thickness across the 8 cm chuck, based on measurements performed by colleague Nick Wakefield [134].

In addition to SEM analysis, transmission electron microscopy (TEM) was also used to interrogate individual GLAD columns. Further comparisons of the structure of the resultant GLAD films for the three oxides provide insight into the differences among TiO_2, SiO_2, and ITO films. TEM images of representative GLAD columns are shown in Figure 2.9, with inset electron diffraction patterns. For this study the GLAD columns were shaved off of sample substrates and sonicated in isopropyl alcohol (IPA), onto conductive grids and then imaged using a JEOL 2200FS TEM. These electron diffraction patterns show that TiO_2 is not crystalline but ITO is polycrystalline. Furthermore, individual TiO_2 and SiO_2 columns are made up of amorphous filaments with features of dimensions on the order of 10 nm, and ITO columns are made up of crystalline 5 nm to 10 nm sized particles.

Calculated film densities were corroborated using the optical Bruggeman effective medium optical model based on measurements taken by Mueller matrix ellipsometry [135], [136]. [1] The required

[1] VASE measurements on selected vertical post GLAD films from this surface area study were performed by Nicholas Wakefield as part of his PhD thesis work on GLAD film uniformity [134].

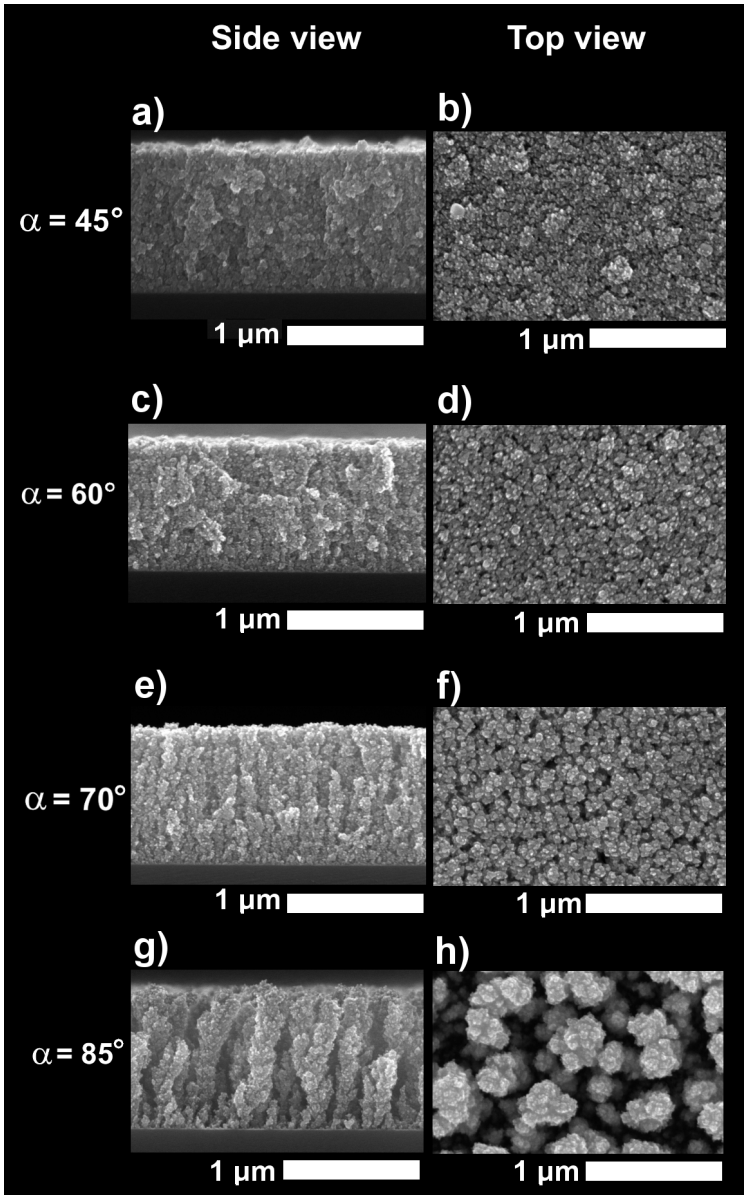


Figure 2.5: Cross-section and plan view SEM images of representative ITO films deposited at (a), (b) 45^o, (c), (d) 60^o, (e), (f) 70^o, (g), (h) 85^o.

optical data was collected in reflection between 400 nm and 1700 nm at 45^o, 55^o and 65^o angles of incidence using a variable angle spectroscopic ellipsometer (VASE, J.A. Woollam Co. Inc.) and modeled using the provided software (WVASE32, v3.517, J.A. Woollam Co. Inc.).

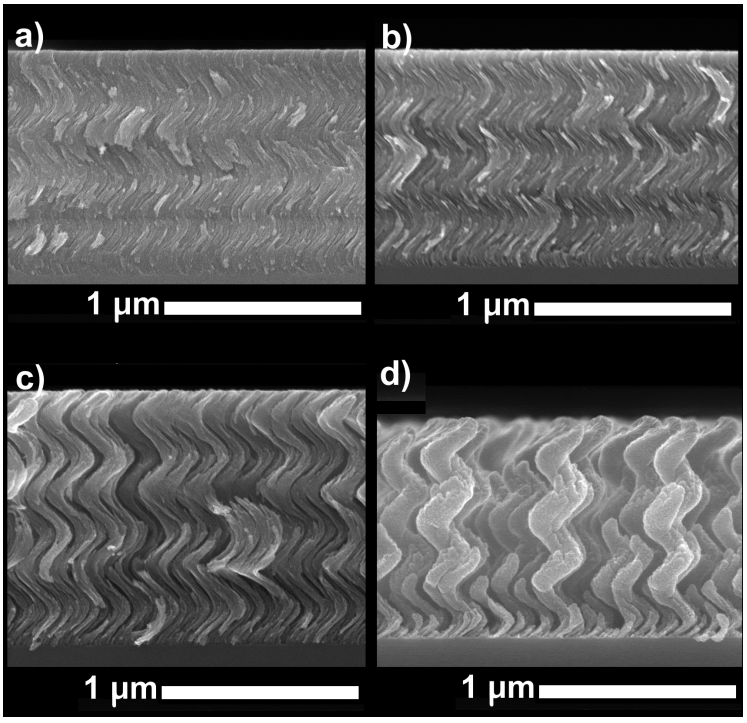


Figure 2.6: Cross-section and plan view SEM images of selected helical TiO_2 films: (a) $\alpha = 70^o$, (b) $\alpha = 75^o$, (c) $\alpha = 80^o$, (d) $\alpha = 85^o$.

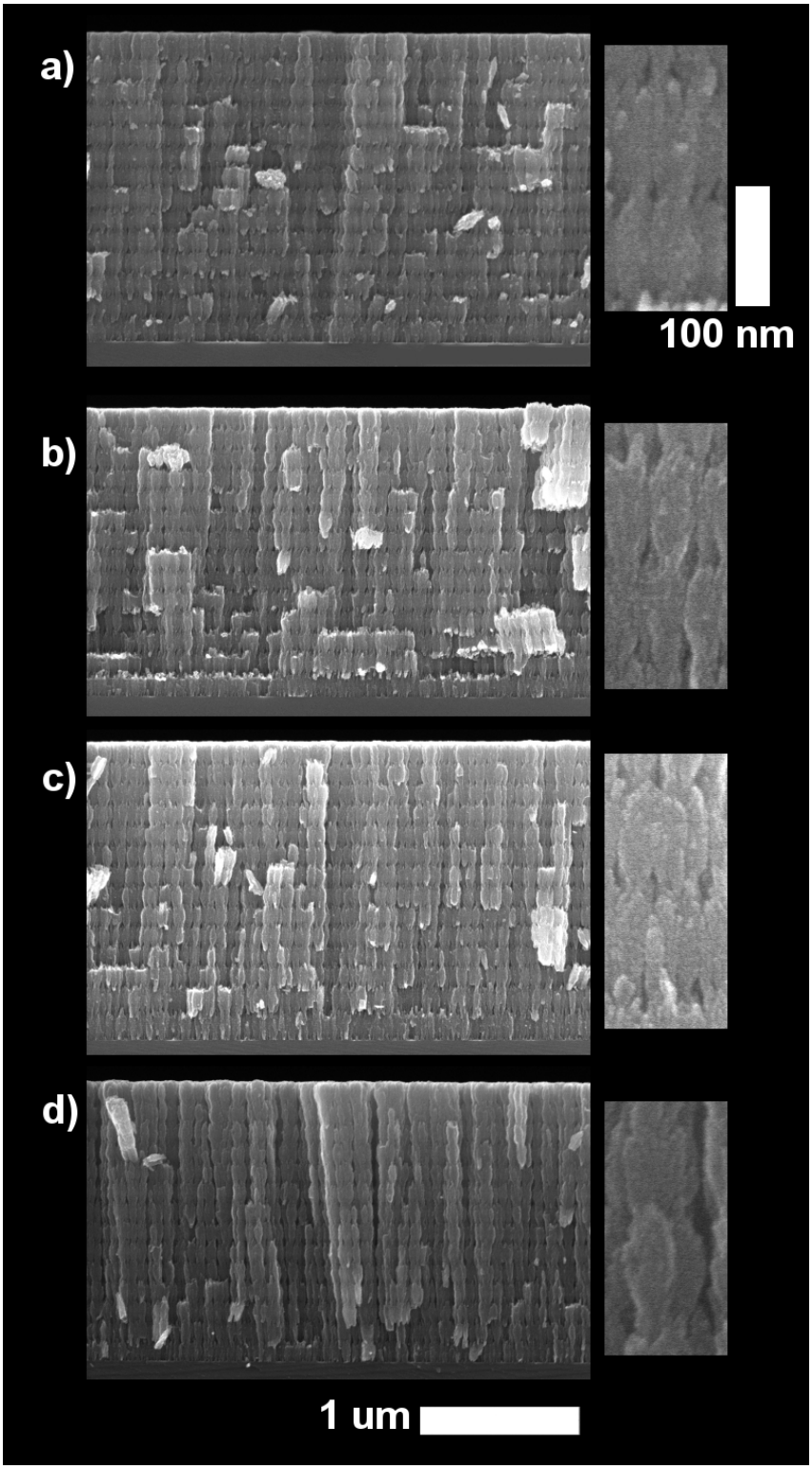


Figure 2.7: Cross-section SEM images of TiO_2 rugate films deposited with α ranging from a) 30^o to 80^o, b) 45^o to 80^o, c) 50^o to 80^o, and d) 60^o to 80^o.

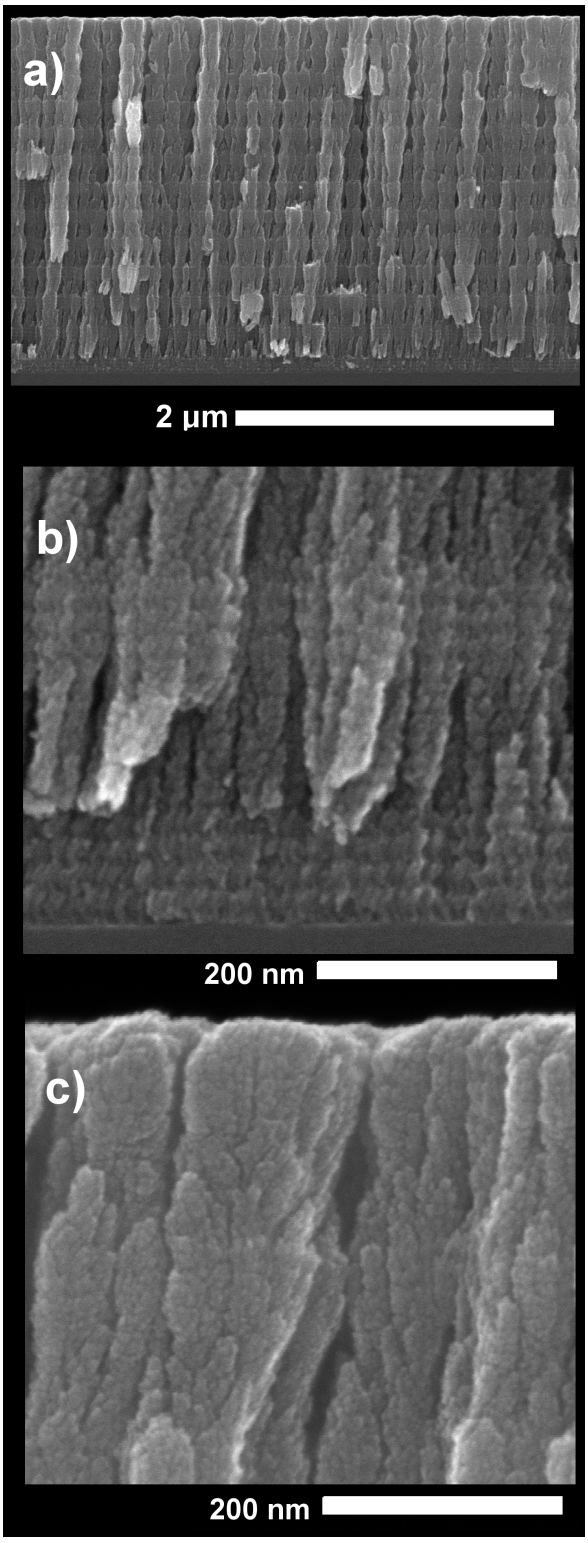


Figure 2.8: Cross-section SEM image of TiO_2 Bragg film, with bands deposited at α = 60^o to 80^o. a) Shows full thickness of deposited sample, while close-ups of the layers at the b) bottom and c) top of the film also shown.

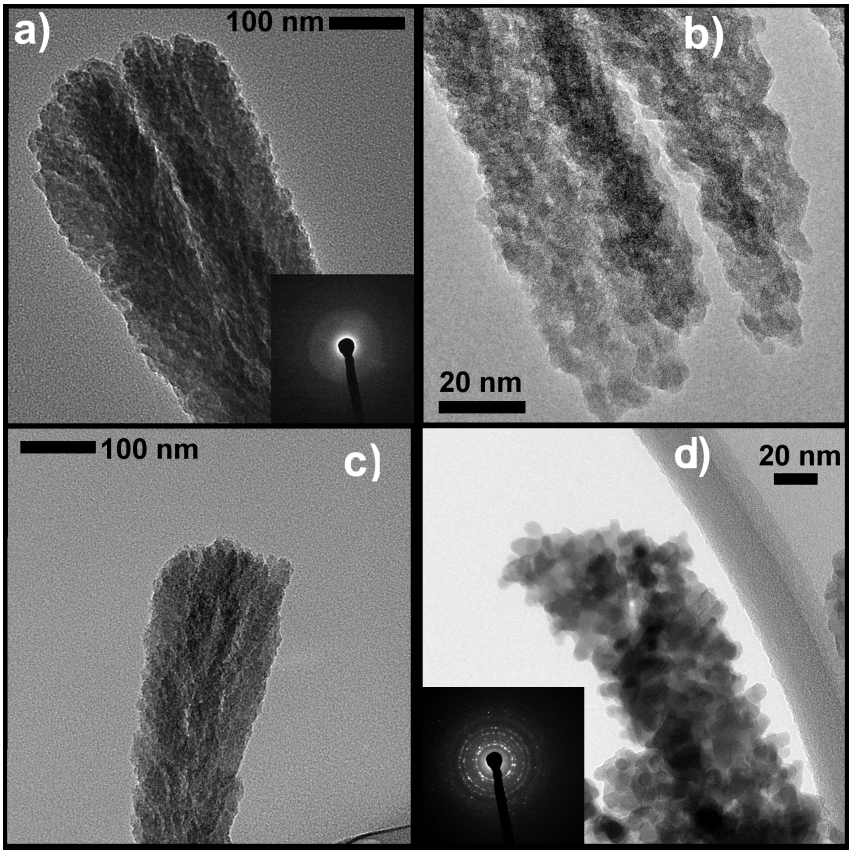


Figure 2.9: TEM images of selected films. (a) and (b) TiO_2 α = 80^o TEM with inset diffraction pattern indicating an amorphous structure. (c) SiO_2 α = 80^o TEM, with amorphous structure verified by XRD measurements [93]. (d) ITO α = 80^o TEM with inset diffraction pattern indicating a polycrystalline structure. TEM images in (a) and (c) ©2008 IEEE. Reprinted with permission from IEEE Sensors, [93]

2.3 Sample volume

For calculations of density, and to determine corrections for humidity infiltration, the sample volumes of all samples needed to be determined. The volume was determined from the footprint area and thickness of all pieces in the sample. The total footprint area of the sample set was found by summing the footprint area (determined using the ImageJ software program) of each piece making up the sample set (FA_i):

$$FA = \sum_{i=1}^{N_{pieces}} FA_i. \tag{2.2}$$

The uncertainty in footprint areas per sample piece (σ_{FAi}) was estimated to be approximately 5% of the sample's footprint area. This was estimated based on repeatability testing of the manual image processing identification method used to identify each sample's footprint area. Calculation of the overall uncertainty in the footprint area (σ_{FA}) for all pieces from a given deposition is given by:

$$\sigma_{FA} = \sqrt{\sum_{i=1}^{N_{pieces}} \left(\sigma_{FA_i}\right)^2}. \tag{2.3}$$

The sample thickness was measured from an SEM image of a deposited witness sample by taking the average of the maximum and minimum estimated thicknesses. The uncertainty in the film thickness of a sample was calculated by summing, in quadrature, the thickness uncertainty in the SEM measurement made on a witness sample and the thickness variation which was expected across the deposition chuck. The thickness uncertainty of the witness sample was found by subtracting the minimum measured thickness from the maximum of the captured SEM image σ_{tSEM} and varied by $\pm$ 3% of the estimated film thickness for a SiO_2 α = 85^o film to 14% for an ITO α = 30^o film. As mentioned above, the thickness variation σ_{tvar} was taken to be 4% of the estimated film thickness.

$$\sigma_t = \sqrt{\sigma_{t_{SEM}}^2 + \sigma_{t_{var}}^2}. \tag{2.4}$$

The total sample volume was then given by:

$$V_{sample} = FA \cdot t, \tag{2.5}$$

with uncertainty determined by:

$$\sigma_{V_{sample}} = V_{sample}\sqrt{\left(\frac{\sigma_{FA}}{FA}\right)^2 + \left(\frac{\sigma_t}{t}\right)^2}. \tag{2.6}$$

2.4 Compensation for humidity infiltration

While exposure to water, oxygen, organics, and contaminants, as well as structure and crystallinity impact the affinity of metal oxides to water, it has been found that generally as-deposited GLAD

metal oxide nanostructures are hydrophillic [137], [138], [139], [140]. Chemisorption, where water binds and dissociates to form hydroxyl groups on the metal oxide surface, and physisorption, weak hydrogen bonding by van der Waals interactions, occurs when the films are subjected to a humid atmosphere [107], [141], [142]. The stable surface hydroxyl bonds resulting from chemisorption are only removed at high temperatures (200 ^{o}C for ZnO, from 100 to 400 ^{o}C for TiO_2) [137], [141], [143], [144]. However, physisorption is reversible as the humidity in the environment decreases [141]. Indeed, this makes metal oxide nanostructures ideal as water sensing medium [93], [132].

Mass measurements performed under humid conditions are therefore expected to be impacted by the water adsorbed onto the surface. For the work performed for this thesis, under typical conditions, films exiting the deposition chamber and following post-deposition annealing were subject to an atmosphere of 50% relative humidity (RH). Water would therefore have adsorbed onto the surface such that mass measurements contained both a film mass component and an adsorbed water component. Ideally, the measurements of film mass would have been carried out under dry conditions (e.g., in a dry nitrogen chamber), but this was not the case. Concerns about film aging and its possible effects on mass prevented re-weighing of samples at a later date in a dry nitrogen chamber. In addition, re-heating samples to drive out water was not undertaken due to the same concerns about impacting film morphology and "aging" the samples. However, samples were initially weighed immediately after removal from the annealing chamber such that minimal water may have been adsorbed at the time of mass measurement. To estimate, and hence compensate for, the mass of water adsorbed onto thin film an analysis based on optical measurements was therefore carried out. The following sections provide a discussion of how water adsorption impacts optical performance, a description of how the amount of water adsorbed can be estimated, and the experimental results used to calculate the water adsorbed onto the GLAD films. However, because the film masses were measured soon after deposition and because the films were degassed at 150 ^{o}C prior to gas adsorption analysis, as described in the next chapter, only the physisorbed water component was considered.

2.4.1 Effective medium approximation

To determine the amount of water adsorbed onto the GLAD films, an optical method was used. For this analysis, the relationship between optical humidity response, dielectric constant, and water volume was determined. Because nanopillars fabricated by the glancing angle deposition method are complex, non-homogeneous, structures the relationship between dielectric constant and percentage of water by volume is be found by applying by the effective medium approximation (EMA) [135]. This technique proposes that the dielectric response of the specimen is best modeled as a medium filled with inclusions of a certain geometry. The medium could for example be a thin film of material perforated by a network of cracks, or a polycrystalline conglomerate for which grain boundaries

define the air inclusion regions. In this situation however, the medium is commonly air (with some percentage of water present) and the inclusions are the circular cross-sections of GLAD pillars.

There are several effective medium approximations. The Lorentz-Lorenz expression is applicable to composite materials on an atomic scale. The Maxwell Garnett EMA proposes a relationship between a material's dielectric constant and that of the host medium and embedded material. The Bruggeman expression is most applicable to GLAD films as it describes the case where the host medium and inclusions material are of similar proportions. The Bruggeman EMA is described by the following equation:

$$0 = f_a \frac{\epsilon_a - \epsilon}{\epsilon_a + 2\epsilon} + f_b \frac{\epsilon_b - \epsilon}{\epsilon_b + 2\epsilon}. \tag{2.7}$$

where f_a is the fraction of material *a* (the metal oxide GLAD film), f_b is the fraction of material *b* (air+water), ϵ_a is the dielectric constant of material *a*, ϵ_b is the dielectric constant of material *b*, and ϵ is the dielectric constant of the overall volume. The dielectric constant is related to the index of refraction, estimated using the method described in the next section, at optical frequencies:

$$\epsilon = \frac{n^2}{\mu_r}. \tag{2.8}$$

where μ_r is the relative permeability of the material, taken to be 1 for GLAD metal oxide films.

2.4.2 Optical humidity measurements

Humidity measurements on titania films were used to estimate the typical amounts of water adsorbed onto the GLAD films at the time when mass measurements were carried out (i.e., post-deposition, when exposed to atmosphere). By applying the effective medium theory to optical measurements made on TiO_2 rugate films, the amount of water adsorbed onto titania films deposited at different deposition angles was estimated. The amount of water adsorbed onto the SiO_2 films was then estimated based on this work as well as a complementary study performed in our lab using a quartz crystal thickness monitoring technique [145]. In addition, the water adsorption on ITO was taken to be similar to that of TiO_2.

The four TiO_2 rugate films (α = 30^o to 80^o (Film A), 45^o to 80^o (Film B), 50^o to 80^o (Film C), and 60^o to 80^o (Film D)), deposited on fused-silica coated substrates, were used for humidity measurements. As discussed in Section 2.1.2, rugate films are optical interference filters whose stop-band center wavelength is determined by the average refractive index of the air/GLAD/water system. Monitoring the stop-band characteristics as a function of relative humidity can therefore be used to determine, by applying Equation 2.7, the amount of water adsorbed into the film. Further, the amount of water adsorbed into the highest and lowest density layers of the film are estimated from the changing characteristics of the stop-band edges. This is because, for multilayer stacks, the blue

edge of the optical interference stop-band corresponds to the most porous, lowest index (n_{low}), and highest deposition angle (i.e., 80^o) regions of the film [132]. Likewise, the red edge corresponds to the densest, highest index (n_{high}), and lowest deposition angle (30^o, 45^o, 50^o, and 60^o, respectively) layers of the structure. When estimating the impact of water adsorption into the layers of the films, the densest and most porous layers were therefore treated separately and no assumption about even distribution of water adsorption in the layers of the film was made. We will see in the next section that the volume percentage of water does appear to depend on deposition angle.

$$\lambda_{blue} = 2n_{low}\Pi. \tag{2.9}$$

$$\lambda_{red} = 2n_{high}\Pi. \tag{2.10}$$

For each of the rugate films the thickness was determined from the SEM images shown in Figure 2.7. The period was found by dividing the thickness by the measured number of turns.

Following measurement of the direct path (no sample) reference spectrum, each film was placed inside of a custom-built humidity chamber, as shown in Figure 2.10. The experimental setup consisted of a white light source (Ocean Optics LS-1 Tungsten Halogen lamp) shining through a plastic window into a stainless steel humidity chamber. The 1"x1" sample was mounted in a teflon holder such that the incident light beam passed through the center and out through a second window. A spectrometer (Ocean Optics HR4000) was used to detect the reference and sample transmitted spectrum. The reference was subsequently subtracted from the sample spectrum to obtain the final transmission spectrum. The humidity in the chamber was adjusted using a nitrogen/water bubbler system built by a colleague for a previous set of experiments [146]. The nitrogen and water were mixed using a multi-channel flow controller (MKS PR4000) which controlled the flow of each input such that mixtures of different ratios could be achieved. A commercial humidity detector (Vaisala HMP100) was employed to monitor the relative humidity in the chamber. A LabVIEW (National Instruments) program, previously written for testing of GLAD-based capacitance sensors with the nitrogen bubbler system, was extended (see Appendix B) to interact with the Ocean Optics spectrometer [146] such that automated optical humidity testing could be carried out.

Using the automated LabVIEW program, humidity was first set to 0% RH and left for at least 4 hr in order to desorb the water vapor adsorbed on the GLAD nanostructures. The humidity was then ramped from approximately 0% to 100% RH in 5% steps (with relative humidity measured using Vaisala humidity detectors). A settling time of 10 min per step was enforced. The transmission spectra (T) at the end of each settling time was captured such that the change in spectra as a function of RH was monitored. As the humidity increased, the stop-band shifted to the red. The changing adsorption spectra for the four films is shown in Figure 2.11.

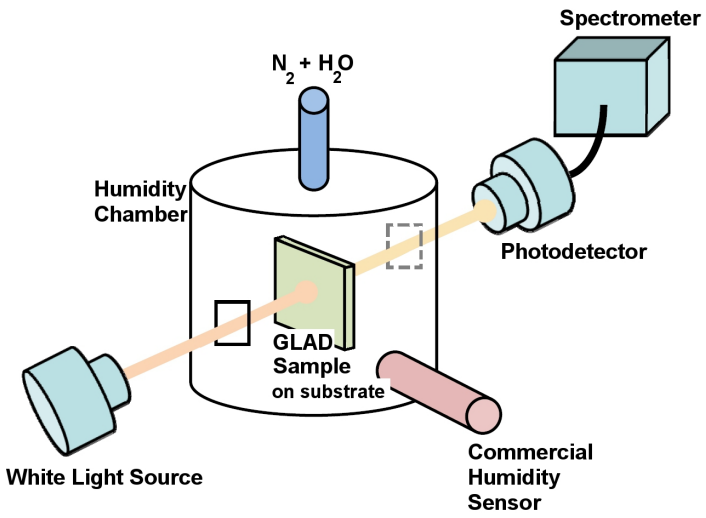


Figure 2.10: Humidity sensing chamber

As water is adsorbed onto the nanostructured columns the effective refractive index increases and the stop-band shifts towards the red. The shifting stop-band was therefore monitored in order to estimate the amount of water adsorbed into the film. To estimate the impact of water adsorption into the highest and lowest density regions of each each, the blue and red stop-band edge shifts of each film were the measured. The point monitored on each of the edges of the stopband was the 30% transmission point. The change in red and blue stop-band edge locations as a function of rising humidity are shown in Figure 2.12.

To estimate the amount of water adsorbed onto not only the rugate films, but all of the GLAD films discussed in this chapter, the data in Figure 2.12 was used. The shifts of the blue stop-band edges as a function of humidity were used to estimate the water adsorption into the most porous portions of the films (corresponding to 80^o). The shifts of the red stop-band edges were used to estimate the water adsorption into the densest layers of the films (corresponding to 30^o, 45^o, 50^o, and 60^o, respectively). To estimate the water adsorbed into the GLAD films at the point of film mass measurement, a relative humidity of 50% was used (the typical value in the laboratory during the months when the depositions were carried out). The changes in stop-band edges from 0% RH to 50% RH ($\Delta\lambda_{50\%RH}$) were therefore estimated from Figure 2.12.

Table 2.2 summarizes the wavelength shifts corresponding to the different deposition angles tested during humidity testing of the rugate films. Using equations 2.9 and 2.10 the refractive index change from 0% RH to 50% RH ($\Delta n_{50\%RH}$) was calculated for each of the stop-band edge corresponding deposition angles.

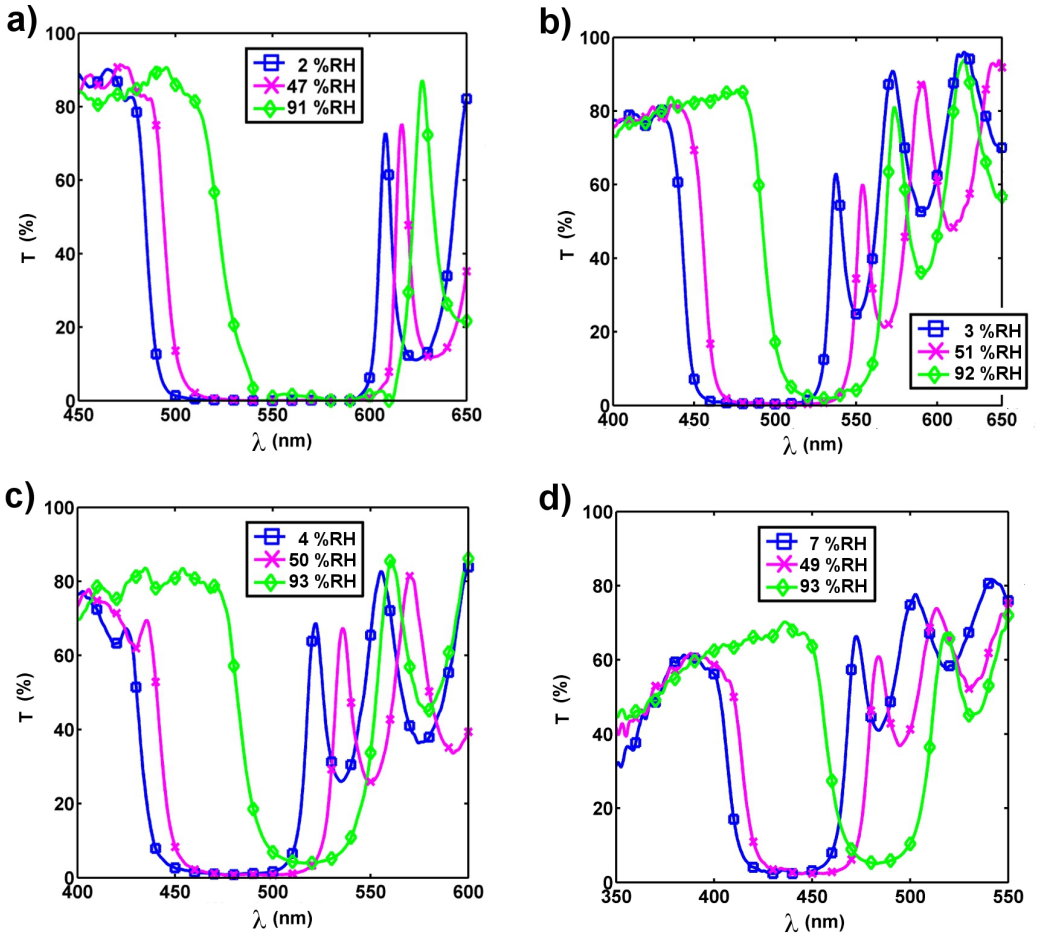


Figure 2.11: Changing transmission spectra as a function of humidity for rugate films deposited with α ranging from (a) 30^o to 80^o, (b) 45^o to 80^o, (c) 50^o to 80^o, and (d) 60^o to 80^o.

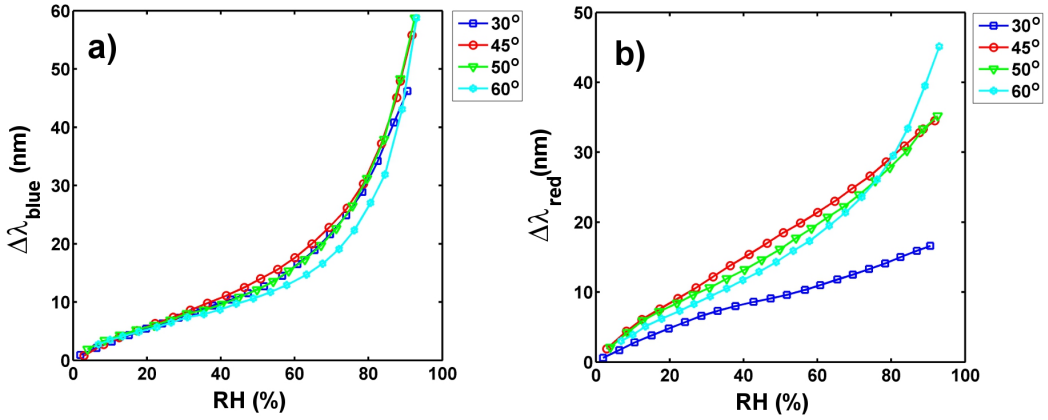


Figure 2.12: Change in (a) blue edge and (b) red edge of stop-band as a function of relative humidity for the four rugate films (designated by densest layer α). Location of stop-band edges measured at 30% transmission.

Film	Edge monitored	α (^{o})	t (μm)	N_{turns} (-)	Π (nm)	$\Delta\lambda_{50\%RH}$ (nm)	$\Delta n_{50\%RH}$
A	red	30	1.94	14	139	9.4	0.034
B	red	45	1.82	14	130	18.2	0.070
C	red	50	1.87	14	135	16.2	0.060
D	red	60	1.77	14	126	14.7	0.069
A	blue	80	1.94	14	126	12.2	0.044

Table 2.2: Properties of TiO_2 rugate films acquired from humidity study.

2.4.3 Estimation of adsorbed water volume into TiO_2 films

The shifts in red and blue stop-band edges from 0% to 50% RH were used to estimate the amount of water adsorbed into vertical post films deposited at the corresponding low and high α and therefore the refractive index change from zero to 50% RH. At each of the deposition angles for which this data had been captured (30^o, 45^o, 50^o, 60^o, and 80^o), the volume fraction of water in the air medium at 50% RH was estimated using the Bruggeman approximation. Given that the sides of the GLAD columns are rough, and that interior porosity is also likely present, this approximation, which assumes that the properties of the adsorbed water and titania columns are those of the bulk material, may be too simplistic. Modeling GLAD columns in this manner is currently the standard [136]. More detailed modeling of the water adsorption properties is left as a future exercise.

To estimate the fraction of TiO_2 in the volume a two-pass algorithm was used. [3] In the first pass the Tait model was used to estimate the relative density of the film [147]. In the second pass the estimated sample density based on the measured film mass, thickness, footprint area, and amount of water adsorbed onto the GLAD columns was used. The relationship proposed by Tait between density and deposition angle is given by:

$$f_{TiO_2} = \delta_{Tait} = \frac{2\cos(\alpha)}{1 + cos(\alpha)}. \tag{2.11}$$

The volume fraction of air at 0% RH was therefore:

$$f_{air_{0\%RH}} = 1 - f_{TiO_2}. \tag{2.12}$$

To calculate the estimated index of refraction of the sample at 0% RH ($n_{0\%RH} = \sqrt{\epsilon_{0\%RH}}$) the Bruggeman approximation (Equation 2.7) was used. In this case f_a and f_b were given by Equations 2.11 and 2.12 respectively. The refractive indices of air, water, and TiO_2 were $n_{air} = \sqrt{\epsilon_{air}}$ = 1, $n_{water} = \sqrt{\epsilon_{water}}$ = 1.33, and $n_{TiO2} = \sqrt{\epsilon_{TiO2}}$ = 2.4.

$$\epsilon_{0\%RH} = \frac{-b_0 - \sqrt{b_0^2 + 8\epsilon_{TiO_2}\epsilon_{air}}}{4}, \tag{2.13}$$

where

[3] Appendix A provided the MATLAB code used to estimate the water adsorbed into the GLAD films.

$$b_0 = 2\epsilon_{air} - \epsilon_{TiO_2} + 3f_{TiO_2}\epsilon_{TiO_2} - 3f_{TiO_2}\epsilon_{air}. \tag{2.14}$$

The effective refractive index and dielectric constant of the whole volume at 50% RH ($n_{50\%RH}$ and $\epsilon_{50\%RH}$) were therefore calculated as:

$$n_{50\%RH} = \sqrt{\epsilon_{50\%RH}} = n_{0\%RH} + \Delta n_{50\%RH}. \tag{2.15}$$

To calculate the volume fraction of water in the air-water medium at 50% RH the Bruggeman approximation was again applied. The effective dielectric constant of the water and air medium was calculated (ϵ_{air_water}):

$$\epsilon_{air_water} = -\frac{3f_{TiO_2}\epsilon_{TiO_2}\epsilon_{50\%RH} - 2\epsilon_{50\%RH}^2 - \epsilon_{TiO_2}\epsilon_{50\%RH}}{\epsilon_{TiO_2} - 3f_{TiO_2}\epsilon_{50\%RH} + 2\epsilon_{50\%RH}}, \tag{2.16}$$

such that the fraction of water in the air-water medium could be calculated:

$$f_{water50\%RH} = \frac{\frac{\epsilon_{air}-\epsilon_{air_water}}{\epsilon_{air}+2\epsilon_{air_water}}}{\frac{\epsilon_{air}-\epsilon_{air_water}}{\epsilon_{air}+2\epsilon_{air_water}} - \frac{\epsilon_{water}-\epsilon_{air_water}}{\epsilon_{water}+2\epsilon_{air_water}}}. \tag{2.17}$$

The volume fraction of water in the entire air/GLAD/water water was then found:

$$f_{water} = (1 - f_{TiO_2})f_{water50\%RH}. \tag{2.18}$$

The percentage of sample volume occupied by water ($V_{water\%}$) at 50% RH was calculated as:

$$V_{water\%} = 100 f_{water}. \tag{2.19}$$

This metric is shown in Figure 2.13. The percentage of water occupied by film regions deposited at 30^o is low, at 11%, while water in the more porous films appear to occupy a larger volume percentage.

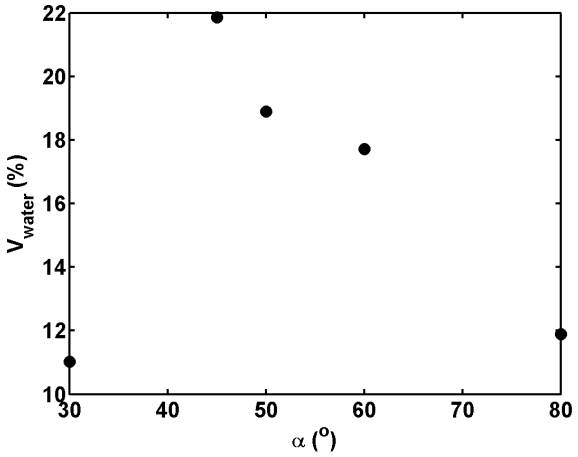


Figure 2.13: Estimated volume of water adsorbed into a TiO_2 film as a percentage of total volume.

2.5 Sample mass

To experimentally determine important GLAD film properties, such as density and specific surface area, the masses of the samples must be known. Determining film mass is not a straight-forward exercise since factors such as the substrate weight, humidity, and edge effects must be taken into account. This section describes the measurement of film mass, performed for each of the depositions in this study.

To determine the mass of each thin film sample, the weight of the deposition substrate was measured before and after deposition. Because many small samples were required to both achieve the large total surface area required for porosimetry measurements and fit into the porosimetry test vial, the total mass was calculated from the summation of film masses. Following this calculation, compensation for humidity was carried out.

For each of the samples deposited in this study (vertical posts, helical films etc.), the difference between final and initial sample piece masses determined Δm:

$$\Delta m = \sum_{i=1}^{N_{pieces}} m_{final_i} - m_{initial_i}. \tag{2.20}$$

For TiO_2 samples, where the amount of water in the films was estimated using the optical humidity measurements on rugate films, the mass of water was taken into account as shown in Equations 2.22 (where δ_{water} is the density of water (1 g cm^{-3})) and 2.25. The mass percentage of water adsorbed on the TiO_2 films is shown in Figure 2.14.

$$m_{water_{TiO2}} = f_{water}(\alpha) V_{sample} \delta_{water}. \tag{2.21}$$

$$m_{water\%_{TiO2}} = 100 \frac{m_{water_{TiO2}}}{\Delta m_{TiO2}}. \tag{2.22}$$

A least-squares fit to the data points, with an additional data point of 0% water mass at an α of 0^o, reveals a relationship between deposition angle and water mass percentage of:

$$\overline{m}_{water\%_{TiO2}}(\alpha) = (0.09 \pm 0.01)\alpha + (0.60 \pm 0.71). \tag{2.23}$$

To estimate the mass of water in SiO_2 samples, capacitance-based humidity measurements performed by colleague Josh Siewart on TiO_2 and SiO_2 films deposited at 81^o on quartz crystal microbalances were used as calibration [145]. In his work the relative amount of water mass adsorbed by the SiO_2 film was 2.94 times that of the TiO_2 film. The mass of water in SiO_2 samples was therefore taken to be:

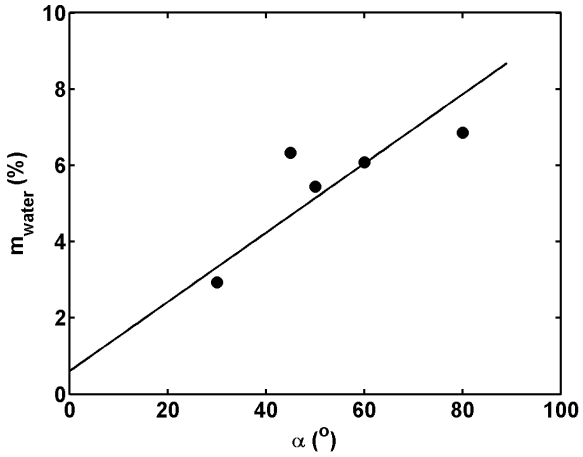


Figure 2.14: Estimated mass of water adsorbed into a TiO_2 film as a percentage of total sample mass.

$$m_{water_{SiO2}} = (2.94\overline{m}_{water\%_{TiO2}}(\alpha))\,\Delta m_{SiO2}. \tag{2.24}$$

$$m_{film} = \Delta m - m_{water}. \tag{2.25}$$

For ITO samples, where the amount of water adsorbed was not known, the uncertainty in mass measurement included a humidity term, but no water mass was subtracted and the mass of the film was taken to be Equation 2.20.

The measurement repeatability of the Sartorius micro-balance was calculated by weighing a sample piece ten times and calculating the scatter. The repeatability (σ_{m_i}) was found to be 0.02 mg per sample piece, with each piece being weighed before and after deposition. The uncertainty in film mass due to water infiltration was taken to be the percentage of water mass adsorbed onto the film. While this term should be additive only, it was included in both positive and negative error terms because of the uncertainty in humidity estimation technique. Finally, the uncertainty terms were added in quadrature to calculate the total uncertainty in film mass for each of the samples studied:

$$\sigma_m = \sqrt{\left[m_{film}\frac{m_{water\%}(\alpha)}{100}\right]^2 + 2\sum_{i=1}^{N_{pieces}}\sigma_{m_i}^2}. \tag{2.26}$$

The effects of deposition on the sides of the sample pieces and shadowing from the substrate attachment tape were not taken into account in this exercise. Film deposited on the sides of the sample is likely to be more dense than that on the surface. Film shadowed by the tape is expected to be less dense. The amount of mass impacted by these effects is on the order of several percent of the total film mass.

2.6 Density Calculations

This section describes the calculation of sample densities based on the humidity-compensated film masses. The relationship between density and deposition angle is then fit based on a standard technique called the Poxson method [148]. Densities are also confirmed using VASE measurements. Calculations of uncertainties in density estimates are provided.

2.6.1 Measured Density

The density for a given sample was calculated as the ratio of total mass to total sample volume:

$$\delta = \frac{m_{film}}{V_{sample}}. \tag{2.27}$$

To estimate the uncertainty in density measurement ($\sigma_{\delta meas}$), the errors in mass, footprint area, and thickness were taken into account:

$$\sigma_{\delta_{meas}} = \delta\sqrt{\left(\frac{\sigma_m}{m_{film}}\right)^2 + \left(\frac{\sigma_{V_{sample}}}{V_{sample}}\right)^2}. \tag{2.28}$$

Table 2.3 shows the properties of the vertical and helical post samples deposited for the surface area study. Table 2.4 provides the film properties for the multilayer stacks which were investigated. Uncertainties in density are as calculated in Equation 2.28.

To corroborate these calculated density uncertainties, the sample density uncertainty (Equation 2.28) was compared to that obtained from the density scatter in the densities of the individual sample pieces (δ_i), as well as to the difference to a standard fit curve (described in Section 2.6.2), the VASE measurements, and to scatter in densities calculated for as a function of α but at different thicknesses. To illustrate the scatter observed for a typical sample set, the densities of the individual pieces making up the SiO_2 vertical post film of 1 μm thickness, deposited at 60^o are shown in Figure 2.15. The root mean square (RMS) scatter in sample piece densities ($\sigma_{\delta scatter}$) is given by Equation 2.30. For the sample case shown here the RMS scatter is 0.05 g cm^{-3} (vs $\sigma_{\delta meas}$ of 0.22 g cm^{-3}) for a mean density of 1.47 g cm^{-3}.

$$\sigma_{\delta_{scatter}} = \sqrt{\frac{1}{N_{piece}}\sum_{i=1}^{N_{piece}}(\delta_i - \delta)}. \tag{2.29}$$

To determine the scatter between densities at a given deposition angle, a thickness study was carried out. Multiple films of different thicknesses were deposited. The density of each film was calculated from the humidity-compensated mass measurements and is plotted in Figure 2.16. A least-squares fit to densities calculated for each α was carried out. With the exception of the $\alpha = 80^o$ case all densities rise slightly as the thickness increases. The parameter $\sigma\delta_{thickness}$ was calculated

Material	Structure	α (^{o})	t (μm)	FA (mm^2)	m$_{film}$ (mg)	δ (g cm^{-3})
ITO	VP	30	0.41 ± 0.07	2740 ± 50	6.54 ± 0.25	5.83 ± 1.09
		40	0.57 ± 0.05	2600 ± 50	8.61 ± 0.29	5.83 ± 0.52
		45	0.52 ± 0.03	1910 ± 40	6.70 ± 0.25	6.72 ± 0.51
		50	0.52 ± 0.03	2280 ± 40	7.68 ± 0.28	6.51 ± 0.46
		55	0.53 ± 0.02	2410 ± 50	7.49 ± 0.26	5.87 ± 0.35
		60	0.46 ± 0.02	2490 ± 50	7.25 ± 0.27	6.3 ± 0.38
		65	0.55 ± 0.02	2250 ± 50	6.37 ± 0.23	5.13 ± 0.31
		70	0.5 ± 0.02	2350 ± 50	6.36 ± 0.23	5.41 ± 0.34
		75	0.53 ± 0.02	2540 ± 60	5.3 ± 0.21	3.94 ± 0.25
		80	0.51 ± 0.02	2450 ± 50	4.65 ± 0.31	3.69 ± 0.3
		85	0.5 ± 0.02	2090 ± 50	3.16 ± 0.28	3.03 ± 0.32
SiO_2	VP	0	3.06 ± 0.12	2350 ± 90	17.06 ± 1.71	2.37 ± 0.27
		30	1.13 ± 0.05	1960 ± 50	4.15 ± 0.43	1.88 ± 0.22
		40	1.03 ± 0.05	1490 ± 40	2.75 ± 0.29	1.79 ± 0.21
		45	0.9 ± 0.05	3230 ± 70	4.88 ± 0.51	1.68 ± 0.2
		50	0.91 ± 0.04	1360 ± 40	2.06 ± 0.23	1.67 ± 0.21
		55	0.93 ± 0.04	1500 ± 40	2.08 ± 0.27	1.5 ± 0.21
		60	0.87 ± 0.04	2550 ± 50	3.12 ± 0.44	1.41 ± 0.21
		65	0.93 ± 0.04	2570 ± 50	3.05 ± 0.46	1.28 ± 0.2
		70	0.97 ± 0.04	4200 ± 140	3.88 ± 0.62	0.95 ± 0.16
		75	0.95 ± 0.04	2570 ± 100	2.09 ± 0.36	0.86 ± 0.16
		80	0.88 ± 0.04	2580 ± 100	1.63 ± 0.3	0.72 ± 0.14
		85	0.99 ± 0.05	3130 ± 110	1.16 ± 0.24	0.37 ± 0.08
TiO_2	VP	0	1.55 ± 0.06	2440 ± 100	14.03 ± 0.57	3.71 ± 0.26
		30	1.09 ± 0.04	1360 ± 40	5.38 ± 0.24	3.64 ± 0.24
		40	1.06 ± 0.04	1270 ± 40	4.63 ± 0.21	3.43 ± 0.23
		45	1.03 ± 0.04	2440 ± 60	8.28 ± 0.35	3.29 ± 0.21
		50	1.02 ± 0.04	2310 ± 50	7.76 ± 0.33	3.29 ± 0.21
		55	0.97 ± 0.04	2660 ± 60	8.11 ± 0.35	3.14 ± 0.2
		60	0.96 ± 0.04	2040 ± 50	5.34 ± 0.25	2.73 ± 0.18
		65	0.95 ± 0.04	2840 ± 60	7.22 ± 0.32	2.68 ± 0.17
		70	1 ± 0.04	2600 ± 50	5.95 ± 0.28	2.29 ± 0.15
		70	2.11 ± 0.08	2590 ± 60	14.14 ± 0.58	2.59 ± 0.16
		70	4.94 ± 0.2	2380 ± 50	30.69 ± 1.23	2.61 ± 0.16
		75	0.67 ± 0.03	2830 ± 60	3.54 ± 0.19	1.86 ± 0.13
		75	1.09 ± 0.08	2560 ± 60	4.55 ± 0.28	1.63 ± 0.16
		75	2.39 ± 0.1	2660 ± 60	13.20 ± 0.54	2.08 ± 0.13
		75	5.34 ± 0.21	2580 ± 60	28.22 ± 1.14	2.05 ± 0.13
		80	0.66 ± 0.03	2950 ± 60	3.38 ± 0.19	1.73 ± 0.13
		80	1.3 ± 0.05	2200 ± 50	4.93 ± 0.33	1.72 ± 0.14
		80	2.18 ± 0.09	3990 ± 140	13.90 ± 0.56	1.60 ± 0.11
		80	5.25 ± 0.21	2450 ± 50	22.20 ± 0.9	1.72 ± 0.11
		85	0.47 ± 0.03	2420 ± 50	1.05 ± 0.13	0.92 ± 0.13
		85	0.75 ± 0.04	2920 ± 100	2.35 ± 0.12	1.07 ± 0.09
		85	1.49 ± 0.06	2890 ± 60	4.82 ± 0.35	1.12 ± 0.10
		85	2.64 ± 0.11	3960 ± 130	11.97 ± 0.49	1.14 ± 0.08
		85	5.07 ± 0.21	3170 ± 60	20.17 ± 0.82	1.25 ± 0.08
	Helical	50	1.11 ± 0.05	2370 ± 60	8.78 ± 0.37	3.33 ± 0.21
		55	1.02 ± 0.04	2630 ± 60	8.87 ± 0.38	3.31 ± 0.21
		60	1.35 ± 0.05	2420 ± 50	10.07 ± 0.42	3.08 ± 0.19
		65	1.06 ± 0.04	2770 ± 60	8.79 ± 0.37	2.99 ± 0.19
		70	1.14 ± 0.05	2630 ± 60	7.62 ± 0.33	2.54 ± 0.16
		75	1.17 ± 0.05	2920 ± 60	7.29 ± 0.32	2.14 ± 0.14
		80	1.29 ± 0.05	2840 ± 60	6.59 ± 0.3	1.8 ± 0.11
		85	1.06 ± 0.04	2330 ± 50	2.03 ± 0.15	0.82 ± 0.07

Table 2.3: Properties of films deposited for surface area study. VP indicates a vertical post film structure.

Material	Structure	α (^{o})	t (μm)	FA (mm^2)	m$_{film}$ (mg)	δ (g cm^{-3})
TiO_2	Rugate	30 to 80	1.94 ± 0.10	1393 ± 56	7.07 ± 0.45	2.61 ± 0.27
		45 to 80	1.81 ± 0.09	2749 ± 110	13.28 ± 0.85	2.67 ± 0.28
		50 to 80	1.89 ± 0.09	256 ± 118	13.68 ± 0.88	2.45 ± 0.26
		60 to 80	1.76 ± 0.09	3022 ± 121	12.11 ± 0.78	2.28 ± 0.24
	Bragg	60 to 80	2.24 ± 0.11	2688 ± 108	14.26 ± 0.92	2.37 ± 0.25
		60 to 80	2.20 ± 0.11	2689 ± 108	12.75 ± 0.83	2.16 ± 0.23

Table 2.4: Properties of multilayer films deposited for characterization study.

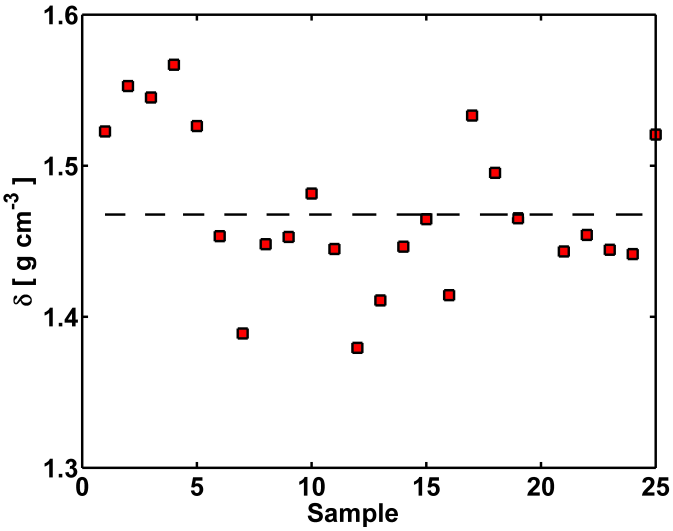


Figure 2.15: Scatter of density measurements for individual SiO_2 α = 60^o 1 μm samples prepared for surface area analysis. Dashed line indicates average density.

as the maximum of the difference between the measured densities (δ_i) and the fit densities (δ_{fiti}), for each deposition angle. Fit parameters are provided in Table 2.5.

$$\sigma_{\delta_{thickness}} = \max |\delta_i - \delta_{fit_i}|. \tag{2.30}$$

2.6.2 Poxson fitting of vertical post densities

Once the densities for all samples sets had been calculated, the relationship between density, thickness, and deposition angle could be determined for each of the materials studied. To parameterize the GLAD film densities the Poxson method has been applied [148]. Poxson *et al.* propose an analytic model which predicts the porosity of nanoporous vertical post films grown using the GLAD tech-

α (^{o})	$\delta_{fit}(\alpha)$ = m α + b (g cm^{-3})	$\sigma\delta_{thickness}$ (g cm^{-3})
70	$\delta = (0.069 \pm 0.034)\ \alpha + (2.31 \pm 0.11)$	0.10
75	$\delta = (0.064 \pm 0.037)\ \alpha + (1.75 \pm 0.11)$	0.14
80	$\delta = (0.000 \pm 0.016)\ \alpha + (1.69 \pm 0.05)$	0.06
85	$\delta = (0.057 \pm 0.014)\ \alpha + (0.98 \pm 0.04)$	0.05

Table 2.5: Density fits for TiO_2 thickness study films.

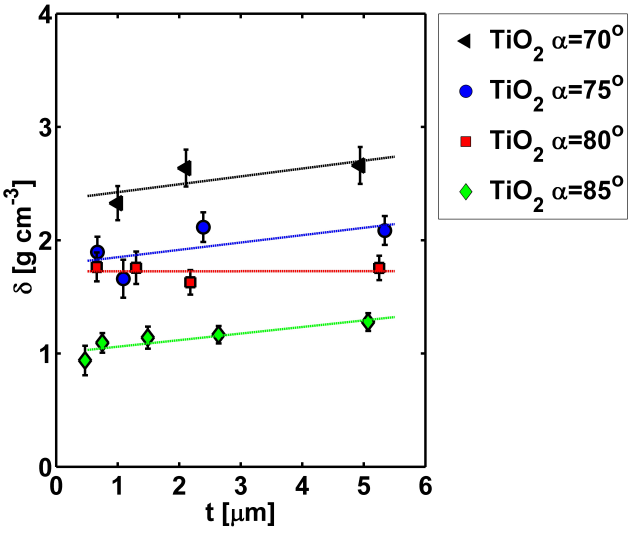


Figure 2.16: Densities of TiO_2 vertical post films as a function of film thickness for a range of deposition angles.

nique. A single fitting parameter (C) takes into account the geometry of the nanostructure and, along with the deposition angle, predicts the porosity (P_{film}) of the deposited thin film. The porosity, a unitless quantity, indicates the fraction of void in the film is given by Poxson's original equation:

$$P_{film} = \frac{\alpha \tan \alpha}{C + \alpha \tan \alpha}. \tag{2.31}$$

The normalized or relative density of a film (δ_{rel}) is the fraction of material in a film:

$$\delta_{rel} = 1 - P_{film}. \tag{2.32}$$

The specific density is found by multiplying the bulk or material density (δ_{bulk}) by the relative density:

$$\delta_{poxson} = \delta_{bulk} \frac{C}{C + \alpha \tan \alpha}. \tag{2.33}$$

The fitting parameter describes the relationship between the volume of material in a nanocolumn (V_m), the cross-sectional area of the nanocolumn (A_{cs}) and the height of the initial nucleation site (h_n):

$$C = \frac{\pi V_m}{2 A_{cs} h_n}. \tag{2.34}$$

The measured densities of vertical post films fabricated from each of the three different materials are shown in Figure 2.17. Error bars in Figure 2.17 represent the calculated uncertainty in density

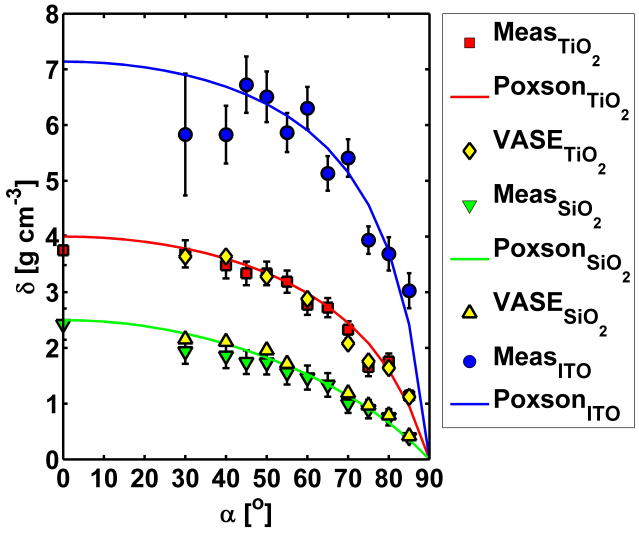


Figure 2.17: Densities of TiO_2, SiO_2, and ITO vertical post films.

measurements, as estimated in Equation 2.28. Equation 2.33 was used to fit Poxson curves to each of the density plots using a least-squares technique.

The least-squares fitting of the density curves involved fitting a line (with optimal slope $\frac{1}{C}$) to the inverse of the normalized density points (δ_i/δ_{bulk}, i = 1..N_δ) shown in Figure 2.17. The best fit line is given by:

$$y(\alpha) = \frac{\alpha \tan \alpha}{C} \tag{2.35}$$

with minimum summed square error (X^2) :

$$X^2 = \min \left[\sum_{i=1}^{N_\delta} \left(\frac{\delta_{bulk}}{\delta_i} - \frac{\alpha_i \tan \alpha_i}{C} \right)^2 \right]. \tag{2.36}$$

Uncertainty in the fit parameter is calculated from the following equation [149]:

$$\sigma_c = \frac{\sqrt{X^2}}{(N-1)\sqrt{\sum_{i=1}^{N_\delta} (\alpha_i \tan \alpha_i)^2}}. \tag{2.37}$$

Quantitatively, one can see that there is good agreement between Poxson's fit and the densities measured, as well as those estimated using the VASE technique. For the three sets of vertical post films, the application of Equation 2.31 results in Poxson fitting parameters of 5.10 $\pm$ 0.37 (R^2 = 0.86), 2.60 $\pm$ 0.11 (R^2 = 0.95), and 8.70 $\pm$ 0.85 (R^2 = 0.712) for TiO_2, SiO_2, and ITO, respectively. In Poxson's original paper, density fitting parameters of 3.17 $\pm$ 0.11 (R^2 = 0.993) and 3.55 $\pm$ 0.21 (R^2 = 0.995) were given for SiO_2 and ITO, respectively. There is good agreement between

our results and those of Poxson for SiO_2. The higher-fitting parameters required for the ITO data presented here are due to somewhat higher densities, especially in the mid-deposition angle range of 50^o - 75^o. These higher densities are most likely due to differences in deposition conditions, such as substrate temperature, crucible-to-substrate distance, and deposition rate. The substrate rotation and film thickness rate may also play roles in the density trend, as discussed below in 2.6.3. Film thickness and broadening may explain some of the differences between our density fitting parameters and those provided by Poxson. In the original Poxson paper, all film thicknesses were 200 nm whereas ours are at least twice as thick. Columnar competition and extinction are dominant factors in shorter films. Also, thicker GLAD films, especially the oxide films, exhibit film broadening as thickness increases. Although film densities appear to be fairly uniform with thickness, as shown in Figure 2.16 for TiO_2 GLAD films of increasing thickness, there may be some density dependence on film thickness when they are less than 1 μm thick.

The difference between the Poxson fit and measured density ($\sigma_{\delta poxson}$) is given by:

$$\sigma_{\delta_{poxson}} = |\delta_{meas} - \delta_{poxson}|. \tag{2.38}$$

With this comparison of measured density and Poxson fit completed, the analysis of uncertainty in measured density can be finalized. In Figure 2.18, the uncertainty in the measured densities of all vertical post samples are compared to several other metrics which quantify the error in density measurement. The first is the standard deviation of the scatter in individual sample piece measurements, such as those shown in Figure 2.15. The second is the difference between measured densities and those predicted by the Poxson fit, as given by Equation 2.38. Third is the difference between the VASE density measurements and those calculated in Equation 5.17. Finally, the uncertainties calculated from the thickness study measurements are provided. These plots show good agreement between the uncertainties calculated by all methods and justify the use of the measured uncertainties, calculated in Equation 2.28, for the error bars in Figure 2.17 and related density plots.

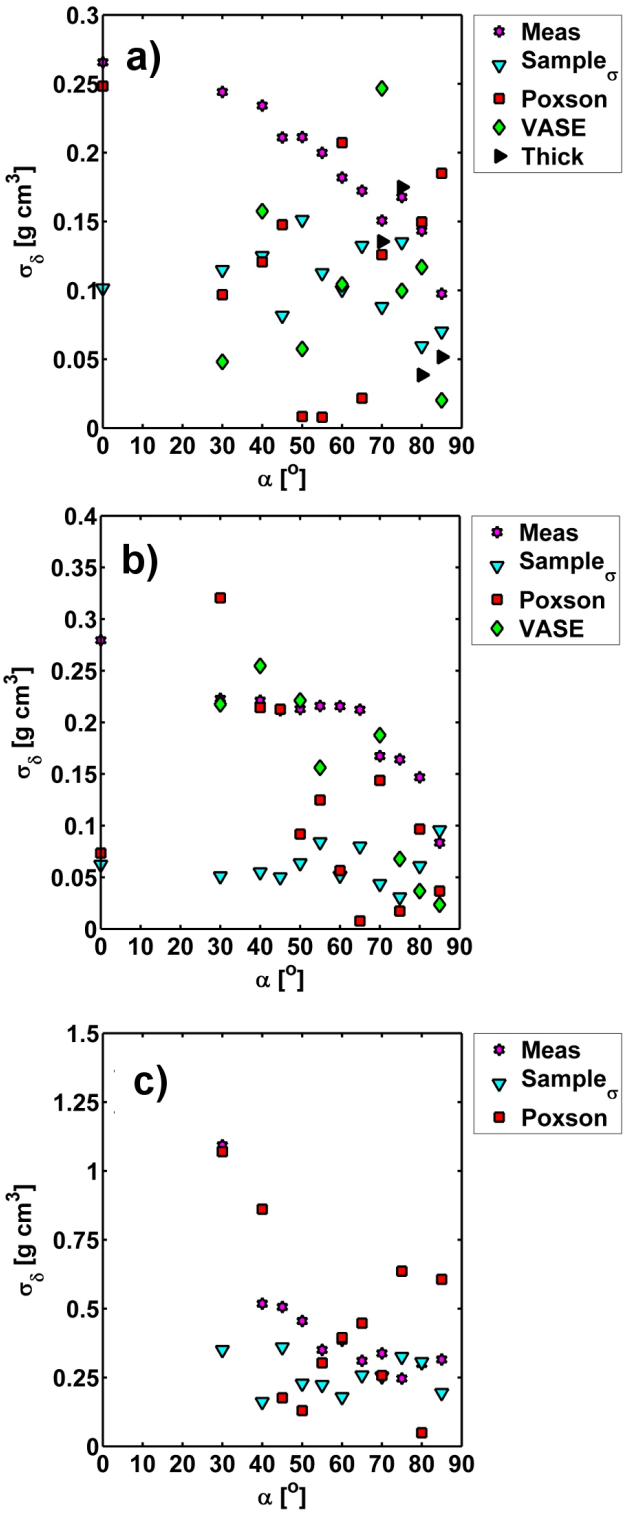


Figure 2.18: Uncertainties in density as a function of deposition angle for a) TiO_2, b) SiO_2, and c) ITO vertical post films. Stars indicate uncertainty calculated from measured mass, footprint area, and thickness values. Triangles indicate standard deviation of scatter from individual sample piece densities. Squares provide the difference in Poxson fit and measured densities. Diamonds are difference between densities measured using the VASE and those calculated from measured mass, footprint area, and thickness values. Sideways triangles indicate thickness study density uncertainties.

2.6.3 Poxson fitting of helical column densities

In addition to evaluating the density trends of vertical post films, the densities of helical GLAD films were tracked as a function of deposition angle. The Poxson theory was also applied. Figure 2.19 shows the density profile of TiO_2 helical GLAD films, as compared to that of vertical post films, deposited over a range of deposition angles. The application of Equation 2.33, results in a Poxson fitting parameter of 5.24 $\pm$ 0.37 with an R^2 value of 0.95 for the helical films. While the densities of the helical films, and the fitting parameter, are slightly higher than those of the vertical post films, they are within error of each other. The higher densities of the helical films suggest a somewhat higher cross-sectional area, perhaps due to an elliptical broadening of the film normal to the direction of flux with a corresponding maintenance of cross-section along the flux direction.

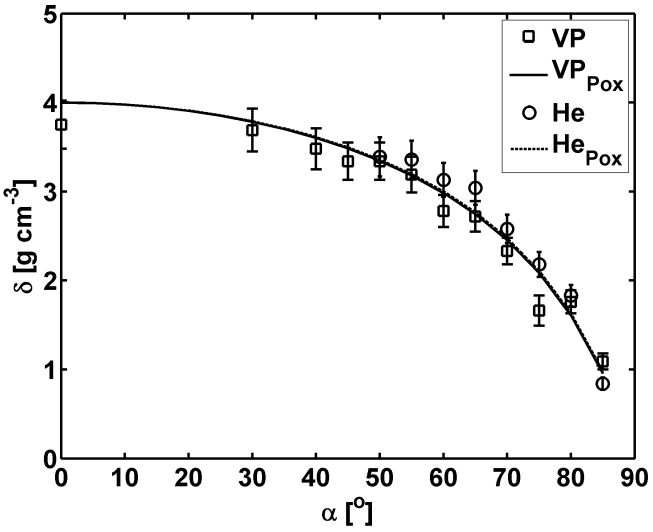


Figure 2.19: Densities of TiO_2 vertical post (VP) helical (He) films as a function of deposition angle, with Poxson fit curves indicated by solid and dashed lines respectively.

2.6.4 Densities of Multilayer films

Two sets of TiO_2 multilayer films were fabricated: rugate and Bragg stacks. Calculation of the multilayer film density required two calibration steps. The first step involved scaling the expected thickness of each layer ($\Delta t_{expect j}$, where j = 1 ... N_{layers}) by the ratio of actual film thickness (t_{act}) to expected thickness (t_{expect}):

$$\Delta t_{est_j} = \Delta t_{expect_j} \frac{t_{act}}{t_{expect}} \quad [nm]. \tag{2.39}$$

The second step involved compensating the estimated mass of the film by the amount of water expected to be present in each of the layers of the TiO_2 films. The estimated mass of water as a function of deposition angle, calculated in Equation 2.23, was used to estimate the percentage of water in each layer (m_{waterj}) and therefore the average percentage of weighed film mass (Δm) made up of water ($m_{water\%avg}$). The mass of film (m_{film}) was then calculated.

$$m_{water\%_j} = \overline{m}_{water\%}(\alpha) \quad [\%], \tag{2.40}$$

$$m_{water\%_{avg}} = \sum_{j=1}^{N_{layers}} m_{water\%_j} \frac{\Delta t_{est_j}}{t_{act}} \quad [\%]. \tag{2.41}$$

$$m_{multilayer} = \Delta m \left(1 - \frac{m_{water\%_{avg}}}{100}\right) \quad [mg]. \tag{2.42}$$

The density of the multilayer film (δ_{meas}) was then calculated from the sample volume and humidity-compensated film mass. This was compared to the density that was expected based on a weighted average of the expected densities of each layer of the film ($\delta_{expected}$).

$$\delta_{meas} = \frac{m_{multilayer}}{V_{sample}}. \tag{2.43}$$

$$\delta_{expected} = \sum_{j}^{N_{layers}} \delta(\alpha) \frac{\Delta t_{est_j}}{t_{act}}. \tag{2.44}$$

Figure 2.20 shows the measured and expected densities for the rugate films, while Figure 2.21 provides this for the Bragg films. Despite the difference in appearance of layers from the bottom to top of the films (as highlighted in Figure 2.8, the densities agree within error terms. While individual layers may not be entirely uniform, the cumulative impact on density appears to be small. The non-uniformity of the layers in the multi-stack films may however impact other parameters, such as surface area which will be explored in the following chapter (Section 3.4.6).

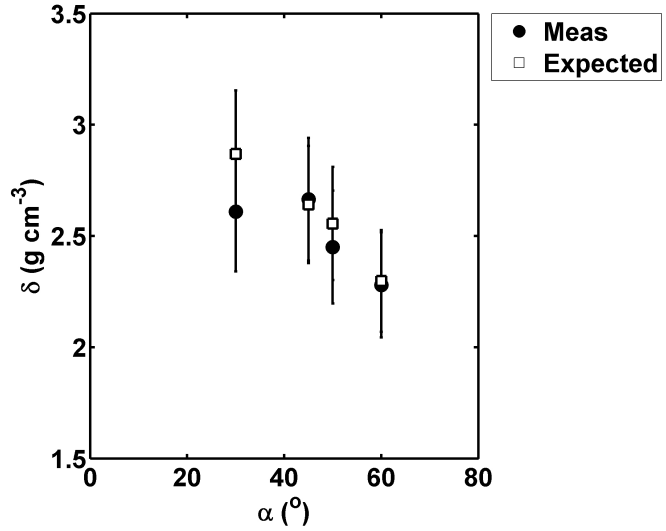


Figure 2.20: Densities of TiO_2 rugate films. Circles indicate measured densities of these multilayer films. Squares indicate densities expected based on weighted averages of single deposition angle vertical post films.

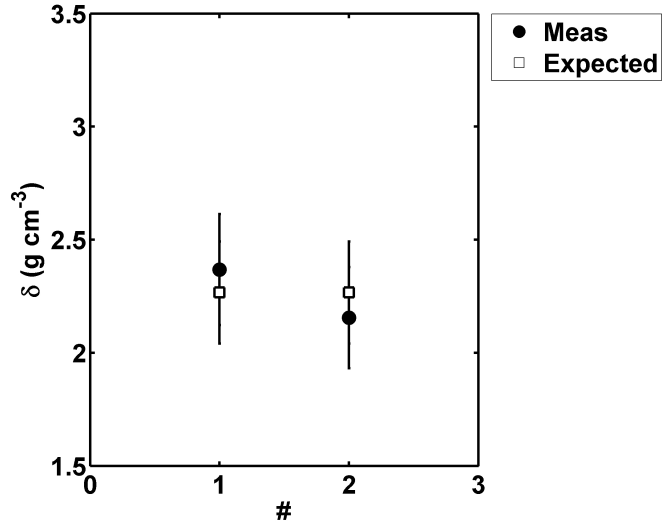


Figure 2.21: Densities of TiO_2 Bragg stack films. Circles indicate measured densities of these multilayer films. Squares indicate densities expected based on weighted averages of single deposition angle vertical post films.

2.7 Chapter Summary

This chapter described the fabrication and basic characterization of obliquely deposited nanostructured samples used in density, surface area, and porosity analysis. Using measurements of mass, footprint area, thickness, and estimates of the impact of humidity, the densities of amorphous titania, amorphous silica and polycrystalline indium tin oxide GLAD films have been characterized and found to depend on oblique deposition angle, structure and material. Upright, vertical post film densities followed the predicted Poxson model, while helical films appeared to be slightly more porous as a function of deposition angle. A study of vertical post films deposited at several different thicknesses and deposition angles revealed that the densities were weakly dependent on the film thickness. In addition, study of the densities of multilayer films showed that the measured densities were within acceptable error limits of those predicted from weighted average calculations. The results presented in this chapter will be use in the next chapters when analyzing the specific surface area, pore size, and pore volume as a function of deposition angle.

Chapter 3

Surface Area Characterization

3.1 Introduction

The surface of a material is its interface to the environment. Chemical reactions are determined by the surface chemistry of a material. Complex electro-magnetic wave interactions depend on the optical properties of an object's surface layers. Passing molecules, and liquids change their flow patterns depending on the roughness and porosity of the support structure. High surface area provides more interface area per volume, thus enhancing properties of applications which depend on the medium/surface interface to function. Enhanced surface areas thus increase catalytic activity, speed separation processes, and boost sensor detection sensitivities. [1]

High surface area materials have surface areas from 300 m^2g^{-1} to over 5000 m^2g^{-1}. Typically, these materials are generated either from a composite of nanosized particles or fibers, by depositing a sample volume that is porous and rough, or through molecular self-assembly. High surface areas are found for both disordered and ordered materials. Traditional high surface materials, such as zeolites and activated carbons exhibited surface areas as high as 904 m^2g^{-1} and 2030 m^2g^{-1} [150], [151]. The first material to achieve a surface area above 4000 m^2g^{-1} was MOF-177 (metal oxygen framework), crystals of metal oxide carboxylate rings [152]. A similar synthesis mechanism forms microporous coordination polymers (MCPs); a reaction of metal (zinc) ions with organic linkers creates materials with surface areas as high as 5200 m^2g^{-1} [153].

Typical nanostructured metal oxide films such as nanowires, nanotubes, and nanocolumns exhibit somewhat lesser, although still high, surface areas. For example, Jiu *et al.* reported nanocrystalline titania nanorod arrays with surface areas of up to 55 m^2g^{-1} [154]. Still others have fabricated titania nanorod meshes with surface areas of 100 m^2g^{-1} to 250 m^2g^{-1} [155], [156].

[1] Portions of the work documented in this chapter have been published in K.M. Krause, *et al.*, "Surface area characterization of obliquely deposited metal oxide nanostructured thin films", Langmuir 26 (2010) 4368-4376.

A quick glance at the SEM image of a GLAD nanostructured film reveals that it is a high surface area material. This high surface area has allowed GLAD films to be used in high capacitance devices, humidity sensors, microbatteries, catalyst supports, and solar cells [89], [93], [157], [158], [159], [160]. To optimize the performance of such devices, the surface area characterization of GLAD films is vital. To address this need, simulation and experimental work has been carried out by several groups as described below.

In this chapter, the GLAD thin film surface area characterization provided by previous groups is built upon. The surface areas of GLAD nanostructures, fabricated with several different morphologies and from several different materials, are investigated using the Brunauer, Emmett, and Teller (BET) technique. Comparison with cyclic voltammetry (CV)-based surface area estimation done by colleague David Rider is also included. This work will provide valuable insight into the structure of obliquely deposited thin films.

3.2 Surface area measurement

Gas adsorption porosimetry, introduced in Section 1.2, is probably the most common technique employed to measure the surface properties of porous materials and is the focus of the present work. This technique relies on the measurement of system pressure changes as the introduced gas species adsorbs onto the surface of the material under study. Variations on this method employ slightly different metrics, such as sample mass changes, to monitor the surface adsorption of the gas. Still others rely on monitoring chemical reactions that occur at the surface when a gas or liquid species is introduced to the host material. An overview of these alternative techniques is now provided, with reference to how they have been employed to estimate the surface area of GLAD films. What follows is a description of gas adsorption porosimetry, a review of the BET surface area analysis technique, and a discussion of the use of krypton gas for high sensitivity adsorption measurements.

3.2.1 GLAD surface area measurement

Several techniques have been used to estimate the surface area of nanostructured obliquely deposited films. These techniques have been primarily used to determine the surface area of the most obliquely deposited films (e.g., $\alpha > 80^o$), although several groups have assessed the surface area of selected slanted post films over a wider range of deposition angles. This present work documents the first measurements of the surface area as a function of deposition angle and thickness for vertical GLAD metal oxide structures. Vertical structures were chosen because of their low inherent thickness gradation and uniformity of structure across a wide substrate area [134].

Surface acoustic wave (SAW) technologies were among the first methods used to measure the surface area of a GLAD film. For example, Broughton *et al.* used a SAW technique to measure the surface area enhancement of manganese GLAD films to be up to 9 m^2 m^{-2} [161]. Additionally, Kiema *et al.* used a SAW technique to estimate the surface area enhancement of carbon GLAD films, deposited at 85^o, to be 63 m^2m^{-2} [162]. For these experiments, the researchers deposited the GLAD film onto the piezoelectric substrate, between the input and output interdigitated transducers previously fabricated on the surface. Transduction from electric oscillations in the input transducer to mechanical waves traveling through the film and back to electric oscillations at the output transducer allowed for mass changes in the film to be detected. Gas adsorbed onto the surface of the film, over a range of system gas pressures, induced the required mass changes. Analysis of the resultant adsorption isotherms, using the BET methodology outlined below, allowed for an estimation of surface area to be made.

Another technique, temperature programmed desorption (TPD), has been used by the Kay and Mullins groups to estimate the surface area of obliquely deposited slanted post Pd, TiO_2, TiC, MgO and amorphous solid water (ASW) films [163], [164], [165], [166], [167]. TPD measurements are carried out under ultra-high vacuum conditions following the deposition of the thin film under investigation. A jet of gas such as N_2 or C_2H_6 was aimed at the surface of the thin film for a short time such that the thin film adsorbs or chemically reacts with the gas. The temperature of the sample substrate is then increased and the amount and type of desorbed molecules is monitored with a mass spectrometer. The surface characteristics can be estimated from the spectra that are captured as a function of temperature. Using this technique, the specific surface areas of obliquely deposited Pd, TiO_2, TiC, MgO and ASW films were found to be 120 m^2g^{-1}, 100 m^2 g^{-1}, 840 m^2g^{-1}, 1000 m^2g^{-1}, and 3000 m^2g^{-1}, respectively.

In addition to experimental methods for estimating the surface area of obliquely deposited thin films, simulations of gas adsorption into a porous GLAD film have been carried out to predict surface characteristics. Suzuki *et al.* provide a 3D ballistic deposition model for predicting the surface area as a function of deposition angle [168]. The simulated columns consist of aggregated cubic particles which deposit on growing columns and are allowed to diffuse over the film surface. Sticking probabilities and diffusion coefficients are based on simulated surface energy, temperature, and deposition conditions. Using this model they estimate a surface area enhancement of up to 20 to 40 times for simulated films deposited at angles from 45^o to 82^o.

Finally, other groups have employed gas adsorption porosimetry, the technique used for the present work, to evaluate the surface area of selected obliquely deposited films. Demirel measured a surface area enhancement of 392 m^2m^{-2} for an obliquely deposited organic poly(p-xylylene) film using nitrogen gas adsorption [169]. Li *et al.* also used nitrogen adsorption to determine that tita-

nium GLAD films, deposited at oblique angles of 65^o to 80^o, had a maximum specific surface area of 25 m^2g^{-1} [170].

3.2.2 Gas adsorption porosimetry

Physical gas adsorption is a standard method employed to measure the surface area and pore size distribution of powders, thin films, and other porous structures [38], [171]. One advantage of this technique is that it is *ex situ*: characterization can be performed on samples outside of the deposition chamber. Annealing, functionalization, and aging tests are therefore possible. As well, each film under investigation can be fully characterized in terms of not only its absolute surface area but also its specific surface area, since its mass, footprint area, and thicknesses can all be measured following deposition. It is a physical adsorption technique whereby inert gases such as nitrogen and argon are adsorbed onto the surface of a sample, without chemically reacting, for the purpose of estimating its surface properties.

The first porosimetry measurements were carried out using a mercury manometer to monitor the pressure changes in a gas adsorbate-adsorbant system. This type of gas adsorption apparatus was used by Emmett and Brunauer in 1937 and allows for a volumetric analysis to be carried out [38]. A more modern apparatus employs one or more pressure transducers, enabling a pressure-based analysis. A gas adsorption manometer is shown in Figure 3.1.

With gas porosimetry, the sample material is kept within an evacuated sample holder. The adsorbate gas is introduced into this holder at an initial pressure. An equilibrium time is allowed to pass and the steady-state pressure (p) is measured. More adsorbate is introduced, increasing the pressure within the fixed-volume sample holder. Equilibrium is again established and the steady-state pressure is again measured. This adsorption process continues up until the saturation vapor pressure (p_o) of the adsorbate gas is reached. The adsorbate is now slowly removed from the sample chamber, with an equilibrium time allowed between subsequent desorption steps, until the sample holder is again evacuated. During this process, the intrinsic bulk saturation pressure of the adsorbate is continuously monitored to ensure that the saturation relative pressure is always known. The ad- and de-sorption isotherms that result can be analyzed to determine a number of sample material parameters.

Sorption isotherms manifest differently depending on the characteristics of the porous sample under investigation. Macropores are so wide that they can act as flat surfaces in terms of the surface-adsorbate interaction. Micropores, with widths on the order of molecular diameters, quickly fill as

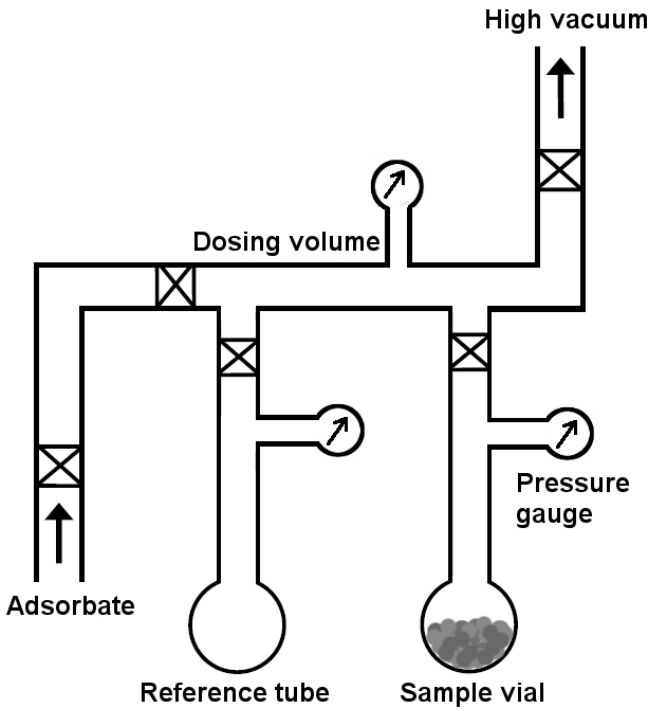


Figure 3.1: Diagram of gas adsorption porosimeter. Adsorbate is introduced to the instrument under high vacuum. The reference tube is used to adjust the system pressure before the sample volume is exposed to the adsorbate.

the sorption pressure increases. Mesopores exist in the intermediate region, filling only as the adsorbate pressure increases to sufficient levels. The International Union of Pure and Applied Chemistry (IUPAC) has classified and described six reference sorption isotherms as shown in Figure 3.2 [172]. These isotherms can be used to estimate the nature of the pores in a porous sample for which a porosimetry analysis has been carried out. For example, a type I sorption isotherm rises to a plateau and exhibits no hysteresis, indicating that the material being tested is microporous. As another example, Type IV isotherms exhibit hysteresis and typically do not reach a stable plateau, indicating that the sample material is macroporous and has pore diameter (d) larger than those which can be measured with the porosimetry parameters of the given experiment.

3.2.3 BET analysis

The Brunauer-Emmett-Teller (BET) technique is the standard method used to estimate the specific surface area (SA) of a porous sample. The BET method uses the adsorption isotherm, measured on a porous sample, to estimate the monolayer capacity of the material. Building on the Langmuir model, which assumed that only a single monolayer formed on a porous material, the BET method proposed that multiple layers of adsorbate could form on the surface. The BET analysis is carried out

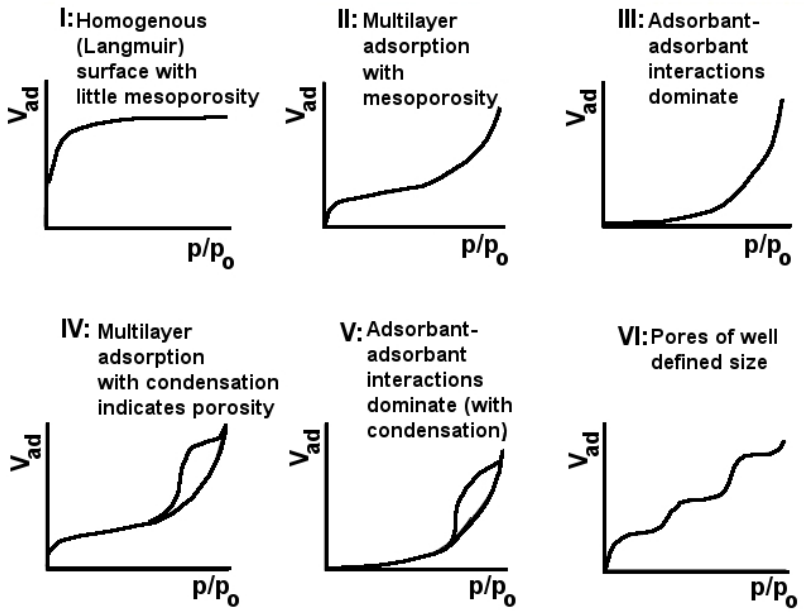


Figure 3.2: IUPAC Isotherms (adapted from [172]).

in the low pressure, initial adsorption regime, just beyond the "knee" (or point "B") of the adsorption isotherm, where monolayer adsorption has essentially completed and multilayer adsorption on the sample surface has begun. Figure 3.3a shows a sample isotherm and illustrates the data points over which the BET is calculated. Here V_{gas} is the volume of gas adsorbed onto the surface of the porous GLAD film. The BET calculation region is approximately 0.05 to 0.3 of the saturation vapor pressure [38], [45]. [2]

An illustration of the adsorbant gas encountering the BET-modeled surface is shown in Figure 3.4. The BET model makes the following assumptions:

- The surface is energetically homogeneous,
- The first layer of adsorbed gas molecules is energetically different than the other layers,
- Multiple layers of adsorbed molecules is possible, but not all need to be filled,
- In all layers except the first the evaporation-condensation conditions are identical, and
- At $p = p_o$, the adsorbate condenses to be a flat liquid along the surface of the solid.

The BET model describes the surface of the sample as hosting multiple layers of adsorbed gas molecules. The surface is modeled as a grid of host sites where a stack of zero, one, or more molecules may be present. The surface is described by the fraction of sites (θ_j) with j = 1..$N_{molecules}$.

[2]The code used to calculate BET surface areas for the GLAD films described here is provided in Appendix A.

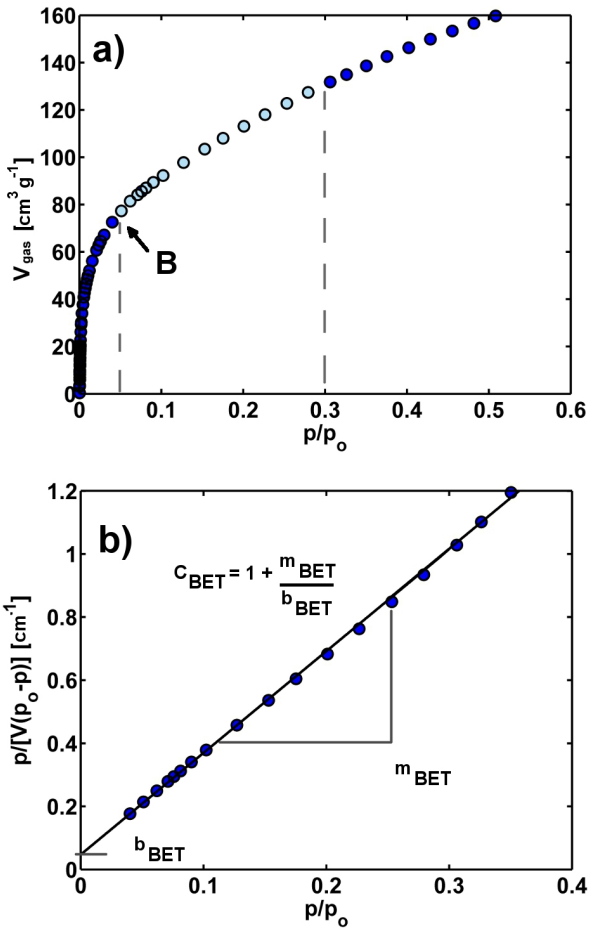


Figure 3.3: BET analysis is performed on an isotherm, such as the one shown in (a) in order to determine the surface area from analysis performed on isotherm linearization illustrated in (b).

Some sites on the surface have no adsorbate molecule (θ_0), some have one molecules (θ_1), others two (θ_1), ... and a few have $N_{molecules}$ ($\theta_{Nmolecules}$) stacked on top of each other. The sum of these fractions is 1:

$$\sum_{j=1}^{N_{molecules}} \theta_j = 1. \tag{3.1}$$

On a surface of area SA_t (denoted with the subscript t to indicate total non-specific surface area) which has z_m sites per unit area, the total number of molecules adsorbed onto the surface is given by z. For example, in the two dimensional configuration shown in Figure 3.4 there are 20 host sites. If, for arguments sake, SA_t is set to 1 nm then z_m = 20 sites nm^{-1} (note that this means that there are 20 molecules in one monolayer). At each site there are 0 to $N_{molecules}$ (where $N_{molecules}$ = 5 in this example) molecules present ($\frac{1}{20}$ of these have zero molecules $\frac{11}{20}$ have one molecule, $\frac{3}{20}$ have two molecules, $\frac{3}{20}$ have three molecules, $\frac{1}{20}$ has four molecules, and $\frac{1}{20}$ has five molecules). The

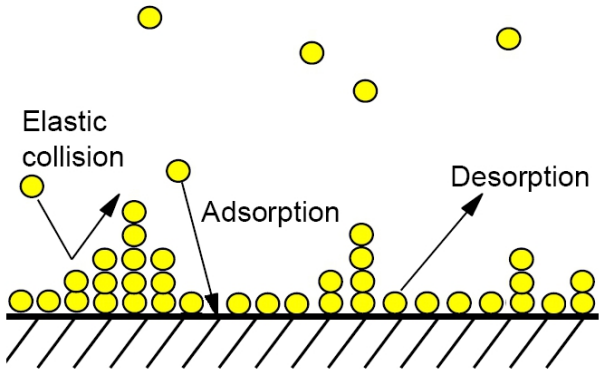


Figure 3.4: BET-modeled surface of porous sample. Molecules collide with the sample surface where they adsorb at a rate determined by the sticking probability, temperature, pressure. Desorption also occurs at a rate dependent on temperature, activation energy of the system. Molecules are assumed to form layers where all layers except the first behave identically.

total number of molecules adsorbed onto the surface is therefore z = 35.

$$z = SA_t \, z_m \sum_{j=1}^{N_{molecules}} j\theta_j. \tag{3.2}$$

By extension, the total number of moles adsorbed onto the surface is given by n_{ad}.

$$\begin{aligned} n_{ad} &= \frac{SA_t \, z_m}{N_a} \sum_{j=1}^{N_{molecules}} j\theta_j \\ &= n_{ml} \sum_{j=1}^{N_{molecules}} j\theta_j, \end{aligned} \tag{3.3}$$

where N_a is Avagadro's constant. The parameter n_{ad} is the measurable quantity. The term $n_{ml} = \frac{SA \, z_m}{N_a}$ is the number of molecules in one monolayer. The goal of the BET exercise is therefore to determine n_{ml} (and effectively the surface area of the sample under test) by measuring n_{ad} and estimating θ_j for all j. To estimate these stacking fractions kinetic theory is applied and some assumptions about the adsorption interactions are assumed.

Kinetic theory says that there is a direct relationship between the properties of a gas (e.g., molecular weight, M), the pressure (p) and temperature (T) in a volume (V), and the rate of collisions ($\frac{dN_{ad}}{dt}$) of the gas with the walls of the volume:

$$\frac{dN_{ad}}{dt} = \frac{N_a \, p}{\sqrt{2\pi MRT}}, \tag{3.4}$$

where R is the gas constant. To calculate the rate that impinging molecules adsorb to the sample surface, both the fraction of unoccupied surface sites and the condensation (or sticking) coefficient (S_c) are taken into account:

$$\begin{aligned}\frac{dN_{ad}}{dt} &= k\ p\ \theta_0\ S_c \\ &= k_a\ \theta_0,\end{aligned} \tag{3.5}$$

where $k = \frac{N_a}{\sqrt{2\pi MRT}}$ and k_a is the adsorption coefficient for the system. Similarly, the rate of molecules desorbing from the surface ($\frac{dN_{de}}{dt}$) depends on the activation energy (E) of the adsorbate-adsorbant system, the fraction of sites that have one molecule on them (θ_1), and a characteristic desorption coefficient (k_d):

$$\frac{dN_{de}}{dt} = k_d\ \theta_1\ e^{\frac{-E}{RT}}. \tag{3.6}$$

In the BET analysis, the system is assumed to be at equilibrium such that the rate of adsorption at one layer is assumed to be equal to the rate of desorption from the next. Because all but the first layer of adsorbate are considered to be energetically identical, the adsorption coefficient is k_a at the first layer and k'_a for the subsequent layers. Similarly, the desorption is k_d at the first layer and k'_d for the subsequent layers. As well, the activation energy at the first layer (E_1) is assumed to be different than that of all subsequent layers (E_2);

$$\begin{aligned} k_a p\theta_0 &= k_d\ \theta_1\ e^{\frac{-E_1}{RT}}, \\ k'_a p\theta_1 &= k'_d\ \theta_2\ e^{\frac{-E_2}{RT}}, \\ &\dots \\ k'_a p\theta_{j-1} &= k'_d\ \theta_j\ e^{\frac{-E_2}{RT}}. \end{aligned} \tag{3.7}$$

By rearranging the above and setting

$$D = \frac{p\ k'_a}{k'_d}\ e^{\frac{E2}{RT}}, \tag{3.8}$$

$$A = \frac{p\ k_a}{k_d}\ e^{\frac{E1}{RT}} \tag{3.9}$$

the following relationships are obtained:

$$\begin{aligned} \theta_1 &= \theta_0\ A, \\ \theta_2 &= \theta_1\ D, \\ \theta_3 &= \theta_2\ D = \theta_1\ D^2, \\ \theta_4 &= \theta_3\ D = \theta_1\ D^3, \\ &\dots \\ \theta_j &= \theta_1\ D^{j-1} = \theta_0\ D^j\ \frac{A}{D} = \theta_0\ D^j\ C_{BET}, \end{aligned} \tag{3.10}$$

where C_{BET} is the BET constant. The BET constant therefore describes all adsorbate-adsorbant interactions including sticking coefficients and activation energies.

By substituting Equations 3.1 and 3.3 into Equation 3.10, letting $D = \frac{p}{p_o}$, and from the relationship $\sum jx^j = \frac{x}{(1-x)^2}$, the linearization of the adsorption equation is obtained:

$$\frac{n_{ad}}{n_{ml}} = \sum jD^j C_{BET}\theta_0, \tag{3.11}$$

$$\frac{n_{ml}}{n_{ad}} = \frac{C_{BET}}{\left(1 - \frac{p}{p_o}\right)\left(1 + [C_{BET} - 1]\frac{p}{p_o}\right)}\frac{p}{p_o}, \tag{3.12}$$

The number of molecules in one monolayer is also equal to the volume of gas in one monolayer (V_{ml}) divided by the the molecular volume (V_o) of the adsorbed gas at standard temperature and pressure:

$$n_{ml} = \frac{V_{ml}}{V_o}, \tag{3.13}$$

such that:

$$\frac{p}{V_{gas}(p_o - p)} = \frac{C_{BET} - 1}{C_{BET}V_{ml}}\frac{p}{p_o} + \frac{1}{C_{BET}V_{ml}}. \tag{3.14}$$

Plotting Equation 3.14 results in a linear plot (with $x_i = \frac{p_i}{p_o}$, $y_i = \frac{p_i}{V_{gas_i}(p_o - p_i)}$, i = 1 .. N_{BET}) whose slope ($m_{BET} = \frac{C_{BET}-1}{C_{BET}V_{ml}}$) and intercept ($b_{BET} = \frac{1}{C_{BET}V_{ml}}$) can be used to determine the BET constant, the monolayer capacity (n_{ml}) of the film and finally the surface area. An example of this BET linearization is shown in Figure 3.3(b). A least squares fit to the BET data points allows for the calculation of the BET constant to be calculated:

$$C_{BET} = 1 + \frac{m_{BET}}{b_{BET}}. \tag{3.15}$$

From this, the monolayer capacity of the material can be calculated:

$$V_{ml} = \frac{1}{C_{BET}b_{BET}}. \tag{3.16}$$

By substituting Equation 3.16 into 3.12, and using the cross-sectional area of the adsorbate ($\xi_{cs} = \frac{1}{z_m}$), the total surface area of the sample can be calculated as:

$$SA_t = \frac{V_{ml}}{V_o}N_a\xi_{cs}. \tag{3.17}$$

The error in the surface area (σ_{BET}), used later in the calculation of error in specific surface area, is calculated from the uncertainties in the slope (db_{BET}) and intercept (dm_{BET}) of the BET linear fit, as well as the scatter in y values [149]:

$$\sigma_{BET} = SA_t \frac{\sqrt{(db_{BET}^2 + dm_{BET}^2)}}{(b_{BET} + m_{BET})}, \tag{3.18}$$

where

$$db_{BET} = \sqrt{dy_{BET}} \cdot \sqrt{\frac{\sum_{i=1}^{N_{BET}} x}{N_{BET} \sum_{i=1}^{N_{BET}} x_i^2 - \left(\sum_{i=1}^{N_{BET}} x_i\right)^2}}, \tag{3.19}$$

$$dm_{BET} = \sqrt{dy_{BET}} \cdot \sqrt{\frac{N_{BET}}{N_{BET} \sum_{i=1}^{N_{BET}} x_i^2 - \left(\sum_{i=1}^{N_{BET}} x_i\right)^2}}, \tag{3.20}$$

$$dy_{BET} = \frac{1}{N_{BET} - 2} \cdot \sum_{i=1}^{N_{BET}} y_i - (m_{BET} \cdot x_i + b_{BET}) \,. \tag{3.21}$$

3.2.4 Krypton gas adsorption

To measure the surface characteristics of the GLAD metal oxide films studied here, high sensitivity krypton gas physical adsorption was used. By employing highly accurate volumetric adsorption equipment, it is possible to measure absolute surface areas as low as 0.5 m^2 - 1 m^2 with nitrogen (at 77 K) or argon (at 87 K) [38], [171]. In order to measure even lower surface areas the number of molecules trapped in the void volume of the sample cell needs to be reduced. This can be achieved by applying krypton adsorption at liquid nitrogen temperature for the surface area analysis.

Krypton at 77 K is 38.5 K below the triple point temperature (T_r = 115.35 K), as shown in Figure 3.5, and it sublimates at 1.6 Torr (at 77 K). However, it has become customary to adopt the saturation pressure of supercooled liquid krypton for the application of the BET equation [171], [173], [174]. In this case, one assumes that despite the fact that the sorption measurement is performed far below the bulk triple point temperature, the adsorbed krypton layer is a liquid on the surface of and in the pores of the sample [175], [176]. The saturation pressure of the supercooled liquid krypton is 2.63 Torr, i.e., the number of molecules in the free space of the sample cell is significantly reduced (to 1/300th) compared to the conditions of nitrogen adsorption at liquid nitrogen temperature. Hence, krypton adsorption at 77 K is much more sensitive, and can be applied to assess surface areas down to at least 0.05 m^2 . This method was therefore used to determine the surface areas of GLAD TiO_2, SiO_2 and ITO films and the results were compared to those found in the literature.

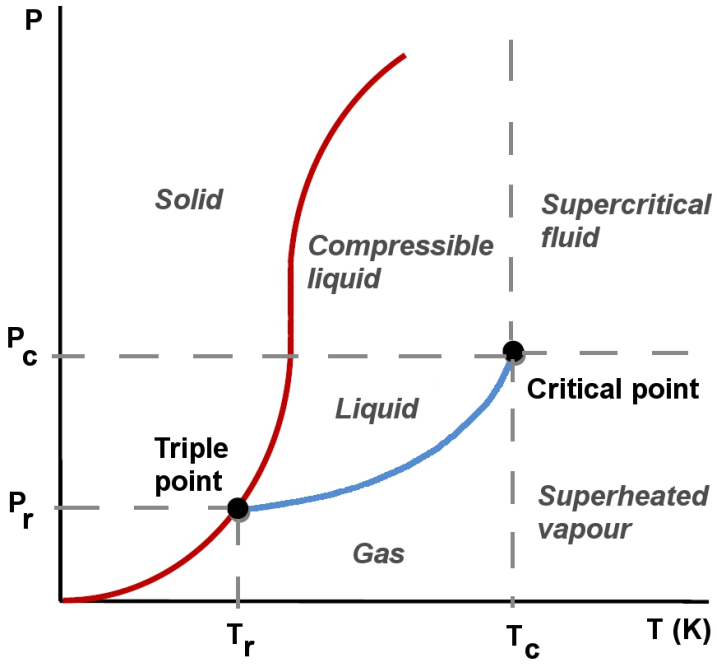


Figure 3.5: Phase diagram illustrates triple point temperature.

3.2.5 Cyclic voltammetry

Because of the electrochemically active nature of ITO, this material provides an opportunity to use a completely different analytical tool for surface area measurement. Previous work has shown that a ferrocene derivative, $Fc(COOH)_2$ forms a submonolayer on ITO surfaces through simple immersion at room temperature [177]. Immersion of the functionalized ITO nanostructure in a conducting electrolyte solution, with a low potential applied across the solution (reference electrode) and GLAD ITO film (acting as the working electrode) will cause each ferrocene molecule to undergo a one-electron oxidation reaction. The resultant current is detected between the working electrode (ITO film) and a counter electrode (in solution). By sweeping the potential from 0 V to a maximum potential (where no more current is detected), all ferrocene molecules that are attached to the ITO surface become completely oxidized. Integration of the current-voltage curve to obtain the anodic peak area (Q) provides a measure of how many ferrocene molecule sites are present on the ITO surface:

$$n_{Fc} = \frac{Q}{e \cdot FA}, \tag{3.22}$$

where e is the elementary charge constant and FA is the footprint area of the sample. The surface area enhancement (SAE_{Fc}) can then be calculated:

$$SAE_{Fc} = n_{Fc}\xi_{Fc}C_{ML}, \tag{3.23}$$

where a cross-sectional area (ξ_{Fc}) of 4x10^{-10} mol cm^{-2} is attributed per Fc(COOH)$_2$, and C_{ML} is the monolayer coverage factor found when applying this technique to air plasma-treated commercially deposited ITO (31%) [178], [179].

3.3 Experimental details

Once all required depositions and density measurements had been performed as described in Chapter 2, the sample pieces were transferred to 12 mm diameter sample bulbs (Quantachrome) in order to carry out physical adsorption tests. Testing was performed on all samples described in Table 2.3 and included TiO_2, SiO_2, and ITO samples. Prior to performing an adsorption test, the glass holder containing the sample pieces was degassed at 150 ^{o}C for 16 hr - 24 hr under vacuum in one of the degassing stations of the Quantachrome Autosorb-1MP (with cold trap cooled by liquid N_2). Only once the outgassing rate fell below 10x10^{-3} Torr min^{-1} was the outgassing taken to be sufficient (as recommended by Quantachrome).

To acquire data for BET analysis, Kr adsorption (Praxair, 99.999% pure) at liquid nitrogen (Praxair) temperature (77 K), were acquired over pressures from 0.05 p_o to 0.3 p_o, where p_o was 2.63 mmHg (the saturation pressure of the super cooled liquid). In some cases, full adsorption and desorption isotherms were also collected, up to 0.55 p_o (above which the bulk saturation, i.e., sublimation, pressure is reached). To verify that the total film surface areas were low enough to warrant the use of Kr gas, N_2 (Praxair, 99.9999% pure) sorption isotherms at 77 K were taken from 0.05 p_o to 1 p_o, where p_o was approximately 720 Torr. The level of liquid in the coolant bath was detected using a sensor (Quantachrome, 00080-AR-RTD and 00080-LN2-RTD, for liquid Ar and N_2 respectively). Captured adsorption data was processed using the BET method as described below.

Aging tests were carried out over several months on samples stored while covered under ambient conditions. These tests were performed in order to assess whether surface area changes occurred over time and could be expected on devices fabricated from these materials.

To complement the BET analysis, cyclic voltammetry was carried out on functionalized GLAD ITO samples by colleague Dr David Rider [177], [180].[3] ITO-coated substrates (area = 5.25 cm^2) were deposited with GLAD ITO films at the same time that samples were fabricated for density and BET surface area testing (see Chapter 2). Prior to functionalization, the ITO GLAD film samples were cleaned with air plasma (Harrick model PDC-32G) for 10 min, followed with a rinse with ethanol and acetonitrile. Functionalization of the ITO nanostructure surfaces with Fc(COOH)$_2$

[3] Functionalization and cyclic voltammetry tests on ITO films were carried out by David Rider as part of [181].

Sample	Reference Surface Area (mg^{-1})	Adsorbant Gas Gas	Measured Surface Area (mg^{-1})
Quantachrome SARM 2003	0.508 $\pm$ 0.36	Krypton	0.528 $\pm$ 0.007
Quantachrome SARM 2005	108.1 $\pm$ 6.6	Nitrogen	111.3 $\pm$ 0.6
Quantachrome SARM 2012	461.8 $\pm$ 36.5	Krypton	457 $\pm$ 16

Table 3.1: Verification of Porosimeter Functionality with Standard Samples.

(Aldrich Chemicals) was then carried out. The cyclic voltammmetry experiments were performed with a Princeton Applied Research (model 2273) instrument with potentiodynamic scans from -0.50 V to 1.75 V at a scan rate of 0.10 V s^{-1} in a conducting electrolyte solution (0.1 M tetrabutylammonium hexafluorophosphate). Princeton Applied Research software (Power Suite v. 2.58) was used to analyze the reaction results and calculate the anodic peak area required for estimation of the surface area of the $Fc(COOH)_2$-functionalized ITO GLAD films.

3.4 Results and Discussion

3.4.1 Calibration

To verify the accuracy of the porosimeter, calibration samples of known surface area were also tested. Adsorption testing at 77 K was carried out on three commercially available surface area reference material samples from Quantachrome. The SARM2003 sample was an aluminum oxide powder sample with a low surface area, while the SARM2012 sample had a high surface area. These two sample were used to verify the function of the machine for 77 K krypton gas adsorption. The SARM2005 sample was an alumina sample with a moderate surface area. This sample was used to verify the performance of the Quantachrome Autosorb porosimeter for 77 K N_2 gas adsorption. BET analysis carried out on these samples to verify the expected surface areas. Table 3.1 shows the difference between calculated and expected surface areas. The measured surface areas were all within acceptable error range and suggest that there is no systematic error due to the instrument.

3.4.2 Surface area of GLAD films

Selected adsorption and desorption isotherms are shown in Figure 3.6. These isothems exhibit a range of characteristics. TiO_2 and SiO_2 GLAD film isotherms demonstrate similar trends, suggesting similar morphology. In both cases, isotherms gradually rise with increased pressure. There are no vertical condensation steps, indicating that pore size distributions are likely broad for all samples under investigation [38]. These TiO_2 and SiO_2 films also have very low krypton adsorption volumes at deposition angles from 0^o to 40^o, signifying non-porous films, whereas large adsorbate volumes above this indicate significant porosity for films deposited at oblique angles. The maximum volume

of krypton adsorbed is achieved at deposition angles from 60^o to 70^o and then falls off at extremely oblique deposition angles. The lack of hysteresis and the rise to a plateau for the 45^o TiO_2 and SiO_2 IUPAC Type I isotherms indicates that these samples are essentially microporous. The isotherms of the 60^o films are Type H2, with large hysteresis, verifying pore condensation for krypton below the triple point and perhaps indicating the presence of ink-bottle shaped mesopores or a network of pores [172]. The 85^o films exhibit Type II isotherms, with the beginnings of hysteresis occurring at high relative pressures, suggesting that large mesopores are present in obliquely deposited TiO_2 and SiO_2 films. Because of bulk solidification at the sublimation pressure, sorption isotherms cannot be acquired up to the vapor pressure of super cooled liquid krypton at 77 K and thus a full description of the porosity for the most oblique films is not possible. However because the BET theory is only applied up to 0.3 of the vapor pressure, the acquired data is more than sufficient for calculating surface areas.

The isotherms of ITO GLAD films are very different from those of TiO_2 and SiO_2. For example, the condensation step prior to sublimation, for 55^o to 65^o deposition angle films, is sharper than the gradual rise seen for the films fabricated from the other two materials. This indicates that a sharper pore size distribution may be present for the ITO films. This may be expected from the TEM image of ITO films, shown in Figure 2.9. The polycrystalline nature of these GLAD ITO columns may result in smoother surface texture, with pores defined by the crystal plane intersections. In contrast, those of the amorphous TiO_2 and SiO_2 films are likely as a result of more disordered junctions and likely result in a broader pore size distribution. This will be explored in the next chapter.

Hysteresis is present for all ITO isotherms, even at a deposition angle of 45^o, indicating pore condensation and the presence of mesoporosity in all of these films. Hysteresis widens as the deposition angle increases and then drops off at oblique angles, reaching its largest value at approximately 60^o. As with the TiO_2 and SiO_2 films, krypton solidification at high relative pressure limits the range over which data could be collected for the ITO samples.

To quantify the surface characteristics of the films under investigation, the BET technique was applied over the low pressure region of each sampled isotherm (from approximately 0.05 to 0.3 of the bulk vapor pressure p_o) to estimate the surface area (SA) of a porous GLAD film. Equation 3.17 was applied (krypton cross-sectional area of 20.2 $\mathring{A}^2$ molecule^{-1}) [38], [171], [182].

Specific (found with respect to humidity-corrected mass (see Section 2.4, in m^2 g^{-1}) and relative surface areas (or surface area enhancement (SAE), found with respect to footprint area and film thickness, in m^2 m^{-2} μm^{-1}) were found for all of the GLAD films described in Tables 2.3 and 2.4:

$$SA = \frac{SA_t}{m} \quad [m^2 g^{-1}], \tag{3.24}$$

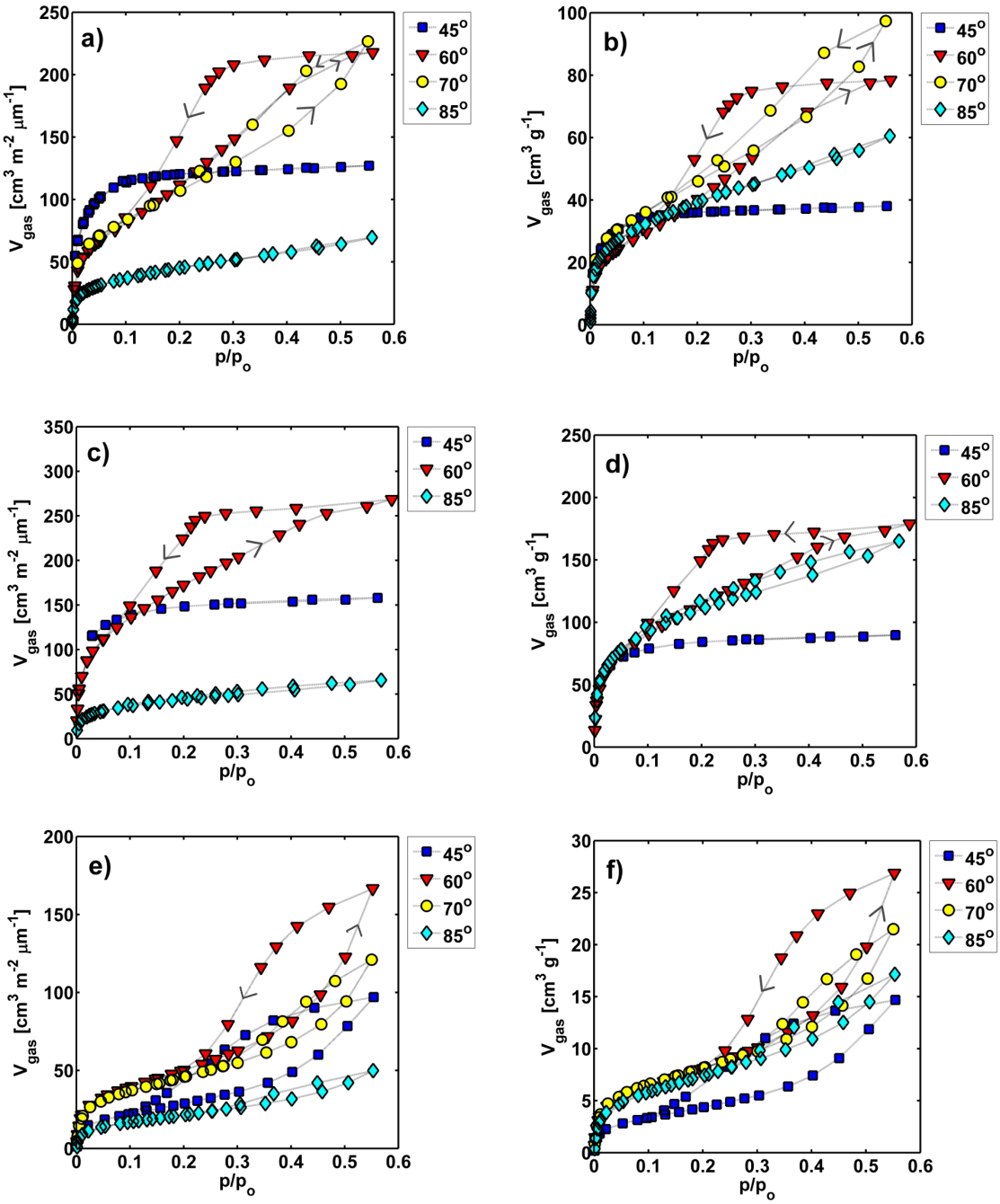


Figure 3.6: Krypton gas sorption isotherms at 77 K for (a) and (b) TiO_2, (c) and (d) SiO_2, (e) and (f) ITO vertical post GLAD films deposited at the indicated deposition angles.

$$SAE = \frac{SA_t}{FA \cdot t} \quad [m^2 m^{-2} \mu m^{-1}]. \tag{3.25}$$

These specific and relative surface areas are shown in Figure 3.7a and b. The highest surface areas are achieved for SiO_2. Peak specific surface areas occur at 60^o for TiO_2, 65^o for SiO_2, and 75^o for

ITO. Peak surface area enhancement occurs at 60^o for TiO_2 and SiO_2, with nearly 1000 times the surface area as footprint area. For the ITO thin films, peak surface area enhancement occurs at a deposition angle of 65^o for ITO, with a much lower peak enhancement of 210 times. Peak specific surface areas and enhancements in the mid-range of deposition angles indicate that even though GLAD films appear dense and unstructured at these angles, they are likely constituted of columns and vertical nanopores. Uncertainties were calculated based on the RMS sum of uncertainties in density measurements (described in Chapter 2, scatter in BET fit (σ_{BET}), and spread in the fit of surface area enhancements calculated in the thickness study (σ_{meas}) described in Section 3.4.3, as described in Equations 3.26 through 3.28.

$$\sigma_{SA_t} = SA_t\sqrt{{\sigma_{meas}}^2 + {\sigma_{BET}}^2} \quad [m^2], \tag{3.26}$$

$$\sigma_{SA} = SA\sqrt{\frac{{\sigma_{SA_t}}^2}{SA_t} + \frac{{\sigma_m}^2}{m}} \quad [m^2 g^{-1}], \tag{3.27}$$

$$\sigma_{SAE} = SAE\sqrt{\frac{{\sigma_{SA_t}}^2}{SA_t} + \frac{{\sigma_{FA}}^2}{FA} + \frac{{\sigma_t}^2}{t}} \quad [m^2 m^{-2} \mu m^{-1}]. \tag{3.28}$$

Surface area enhancements for the TiO_2, SiO_2 and ITO GLAD films studied here are high relative to those found in the literature [161], [162], [168], [169], [170]. The work by Demirel shows that organic materials are smooth and have low surface area. The magnetron sputtered Ti films deposited by Li *et al.* were estimated to have a specific surface area that was an order of magnitude less than that of the oxide films analyzed here [170]. In addition, the maximum surface area was found for deposition angles of 80^o to 85^o, values that are significantly shifted from the trend found here. Choice of material, as well as the lack of substrate rotation during deposition may have resulted in the shift relative to Li's results.

One of the significant contributions to the analysis of metal oxide surface area for GLAD-like films has been performed by the Kay group. Kay's temperature programmed desorption work on TiO_2 films, using ultra-high vacuum ballistic oblique deposition, resulted in an estimate of the maximum specific surface area of 100 m^2 g^{-1} (at a substrate temperature of 50 K) [163]. This value is half of the specific surface areas measured here using the krytpon BET method. Additionally, the surface area that this group obtained at high temperatures is even less at the temperature at which we deposited our films (340 K). Despite differences in approaches (ours were measured directly ex situ, while theirs were based on indirect measurements), the film density measurements suggest that the densities of our films are less than those of the Kay group and that our films may also have a rougher surface. While both structures were found to be amorphous, slightly different deposition conditions may have led to different morphological properties. For example, we employed substrate rotation, resulting in columns with vertical morphologies; slanted post films may exhibit lower surface areas.

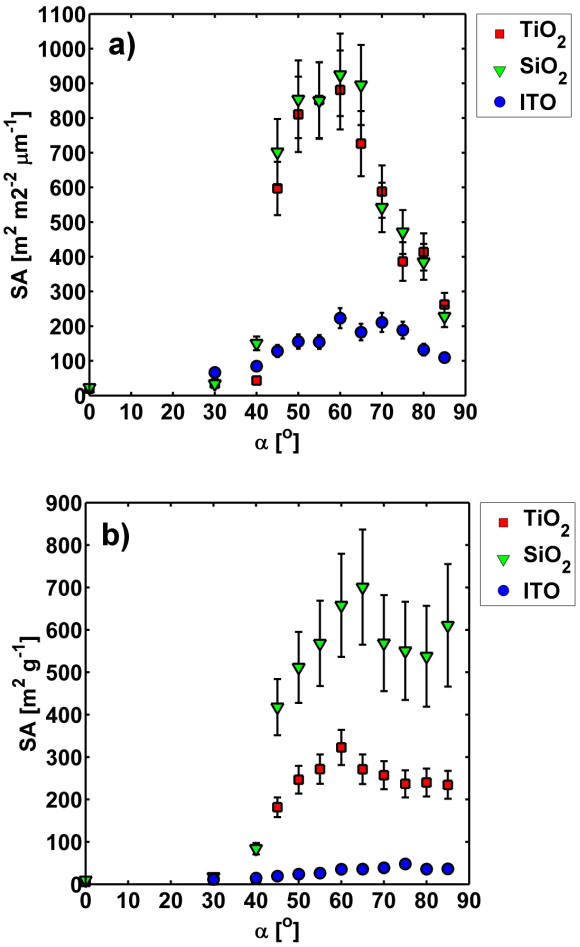


Figure 3.7: a) Relative surface area and b) specific surface area for TiO_2, SiO_2 and ITO vertical post films .

Additionally, our deposition process was different in that we evaporated TiO_2 in the presence of O_2 gas, while they evaporated Ti. This may have led to different surface properties due to differences in Ti and O_2 adatom mobilities as they reached the substrate surface. Finally, due to the increased pressure of O_2 gas, our deposition pressure was higher than that used in the other study. Collisions between the evaporant flux and gas flow have been found to result in lessened mobility of impinging particles and therefore a lower thin film density [183].

3.4.3 Thickness Study

In addition to exploring the relationship between deposition angle and surface area, we also investigated the impact that film thickness has on the surface morphology. We carried out a thickness study for TiO_2 GLAD films deposited at oblique angles from 70^o to 85^o. The results shown in Figure 3.8 indicate that surface area to footprint area ratio rises linearly with film thickness. Least-squares fits

α (^{o})	m_t (m^2 m^{-2} μm^{-1})	b_t (m^2 m^{-2})	$\frac{\sigma_y}{b_t}$ (%)
70	732 $\pm$ 35	-7 $\pm$ 95	18
75	488 $\pm$ 15	3 $\pm$ 40	20
80	318 $\pm$ 16	12 $\pm$ 43	5
85	235 $\pm$ 12	-9 $\pm$ 30	6

Table 3.2: TiO_2 vertical post surface area scaling with thickness (where t is in μm).

with an additional forced data point of 1 m^2 m^{-2} at 0 μm thickness and weighted to the reciprocal of absolute surface area measurement errors (with error of additional forced data point set to that of the first true data point) resulted in the calculation of slopes (m_t) and intercepts (b_t) as shown in Table 3.2. Slopes decrease from 732 m^2 m^{-2} μm^{-1} to 235 m^2 m^{-2} μm^{-1} over deposition angles from 70^o to 85^o. Intercepts are all through the origin within error. The average of the scatter in data points from the fit line is 12% of the fit line value.

The linear increase in surface area indicates that, despite the column broadening that has been noted by various GLAD groups, the differential surface area is a constant as a function of film thickness [78], [184]. This means that the same surface area that is exhibited by the many, small, initial GLAD columns is found for the broad columns that survive towards the top of the film. This has implications in terms of column roughness and internal surface area. To achieve the same surface area, at the tops of the broad GLAD columns as at the bottom of the film, the large column tops must be quite rough and contain significant surface area in their inner volume.

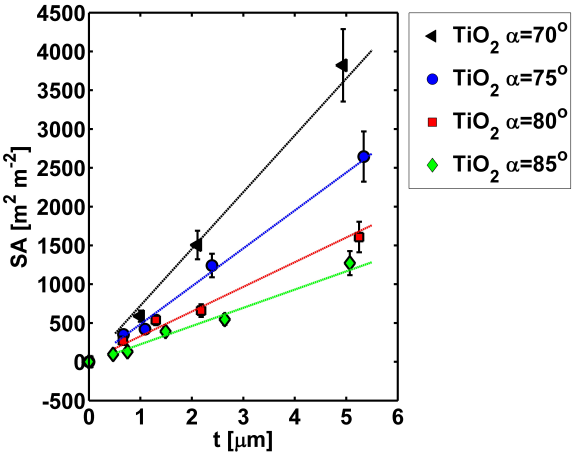


Figure 3.8: Relative surface area of TiO_2 vertical post (VP) vs thickness for a range of deposition angles. Curves shown to guide the eye.

3.4.4 Helical film surface areas

Control of substrate rotation rate influences the morphology and type of structure that is deposited. Slow rotation results in helical films, while faster rotation leads to upright vertical posts. The surface areas of helical GLAD films were measured and compared against the surface area obtained for vertical post films. As shown in Figure 3.9, the surface area enhancements of the two film types follow similar trends and reach nearly the same peak surface area. The fact that the peak value is the same suggests that there is not a strong dependence of peak surface area on substrate rotation rate. A modest shift towards higher surface areas at higher deposition angles is however seen for the helical films. The peak surface area is 65^o for helical films versus 60^o for vertical post films. There is a cross-over point at 80^o, where both helical and vertical post films have the same surface areas. By 85^o, helical films have lower surface area. To some extent, these differences mirror those seen in the density plot shown in Figure 2.19, where the densities for the helical films were shifted upwards. The difference in surface area may be because of the subtle difference in growth conditions for the vertical versus helical post films; the tops of the helical posts always face the incident flux, while only one side of the vertical post faces the flux direction at a given time. There may be slight differences in column top temperature of the vertical post column from shadowed to unshadowed side which facilitate diffusion of recently impinged atoms, thus smoothing the surface slightly and decreasing the surface area [185]. Further study into the effect of substrate rotation rate during deposition is required to determine the full impact of rotation speed on film morphology [172].

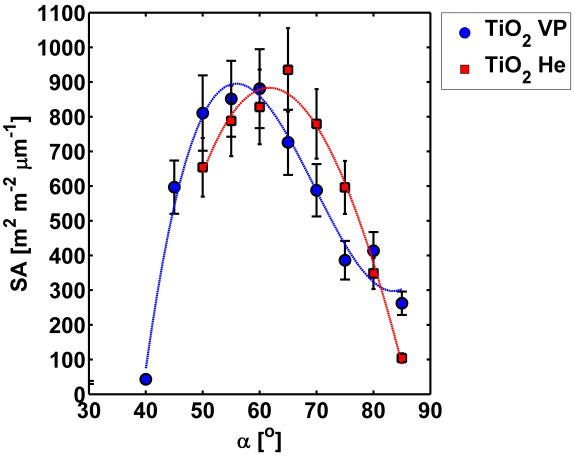


Figure 3.9: Relative surface areas of TiO_2 vertical post (VP) compared to those of helical (He) films for a range of deposition angles.

3.4.5 Confirmation by Cyclic voltammetry

A comparison of two techniques for measuring surface area enhancement was made for the ITO nanostructured samples. SAEs, as calculated using the BET method, were compared to those determined by colleague David Rider using cyclic voltammetry [181]. This comparison is shown in Figure 3.10.

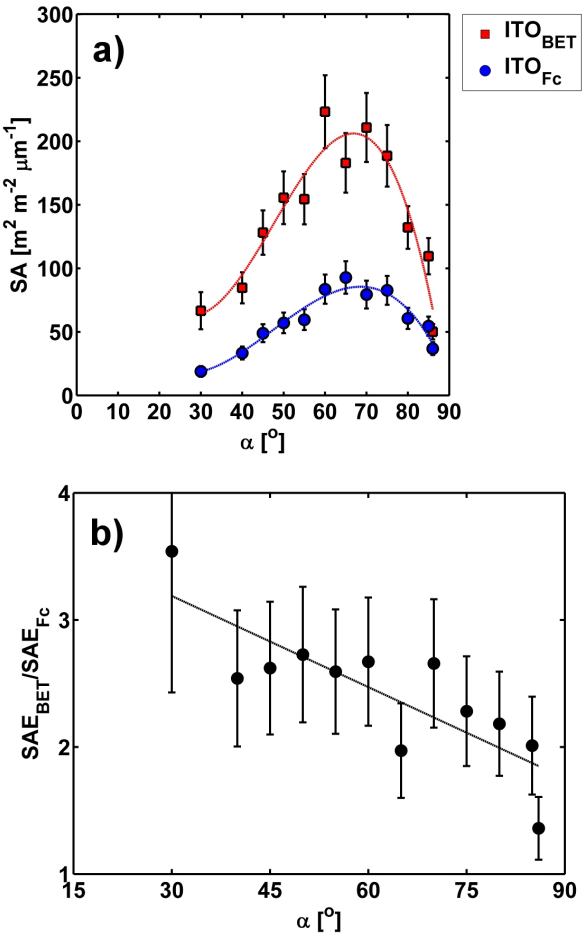


Figure 3.10: Relative surface areas of ITO vertical post films calculated by BET method and CV electro-chemical method.

Uncertainties in the estimation of the number of ferrocene molecules, coupled with the uncertainty in footprint area and deposited ITO film thickness measurement resulted in an uncertainty in SAE_{Fc} of 12%. While the BET and cyclic voltammetry SAE trends are similar, with a peak surface area at a deposition angle of 60^o to 65^o, the BET method reveals a surface area which is approximately twice to three times that predicted by cyclic voltammetry. This discrepancy may arise from a larger fraction of electrochemically inert surface area, resulting in a lower measured degree of incorporation of $Fc(COOH)_2$ [178], [179]. The electrochemically active surface areas of different ITO

structures is known to vary widely with often subtle differences in the synthetic procedures (i.e., DC sputtering versus evaporation), coating and cleaning techniques, and aging [138]. Another cause of this discrepancy could be that the relatively large ferrocene molecules are not able to penetrate into the interior surface of the GLAD columns. The difference in reported surface area for the two methods could be an indication of the relative ratios of interior to exterior surface area. More investigation, including probing the surface of GLAD films with differently size molecules, is required to determine whether this is the case. In addition, altering the parameters of the CV procedure, such as sweep rate, could probe the relative interior to exterior diffusion rates of the electrolyte solution and thus be used to estimate this surface area ratio.

3.4.6 Multilayer films

The multilayer films shown in Figures 2.7 and 2.8 were also characterized by gas adsorption. The surface areas of these films were measured using krypton at 77 K and the BET method. These surface areas were then compared to the expected results obtained from a weighted average of the areas measured for individual vertical posts at constant deposition angles ($SA(\alpha)$). These expected specific and relative surface areas were calculated using Equations 3.24 and 3.25.

$$SA_{expected} = \sum_{j}^{N_{layers}} SA_j(\alpha)\frac{\Delta t_{est_j}}{t_{act}} \quad [m^2 g^{-1}], \tag{3.29}$$

$$SAE_{expected} = \sum_{j}^{N_{layers}} SAE_j(\alpha)\frac{\Delta t_{est_j}}{t_{act}} \quad [m^2 m^{-2} \mu m^{-1}]. \tag{3.30}$$

Figure 3.11 shows a comparison of expected vs measured specific and relative surface areas for the rugate films. The surface areas are within error, but diverge for the films with higher base deposition angle. This is likely because the characteristics of the individual film layers making up the rugate stack are not equivalent to those of entire vertical posts deposited at single deposition angles. For example, one can see from a comparison of Figures 2.7d (Rugate, α = 60^o to 80^o) and 2.3e (Vertical post, α = 60^o), that the rugate film does not reach the compactness expected for the densest α = 60^o layer. It may be that, by changing α quickly, the rate of structural characteristic changes is reduced and discrete layers do not reach the morphological characteristics of the corresponding vertical post columns.

A comparison of the expected and measured surface areas for the Bragg multilayer structures is shown in Figure 3.13. The results are similar to those seen in the case of the rugate multilayer film. The expected surface area is higher than that achieved. The actual surface area appears to be more similar to a high deposition angle, lower surface area film. This has implications when applying

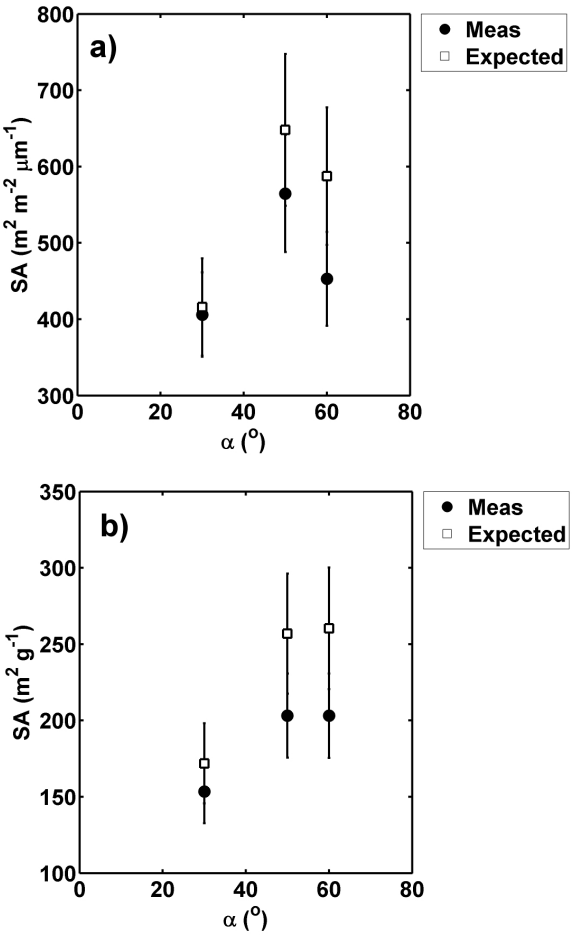


Figure 3.11: a) Relative surface area and b) specific surface area for rugate films. Films were deposited from note α to 80^o. Circles indicate measured surface areas of these multilayer films. Squares indicate densities expected based on weighted averages of surface areas for single deposition angle vertical post films.

the surface area versus deposition angle results shown in Figure 3.7 to complex, multilayer films. Devices fabricated from these stacks may not achieve the surface area expected.

3.4.7 Aging study

The surface area of several films were monitored over time to determine whether aging related changes occurred. This exercise was motivated by evidence from TiO_2 GLAD film optical and humidity characterization that TiO_2 samples experience changes in optical and electrical properties over time (some of which are reversed through exposure to UV light) [100]. Three nanostructured samples, deposited as part of the studies described above in this chapter, were therefore monitored over the course of several years. Two TiO_2 samples were monitored, as well as one SiO_2 sample. The SiO_2 film was deposited at a deposition angle of 60^o and thickness of 1 μm. The TiO_2 samples were deposited at α = 60^o and 80^o (thickness of 1 μm and 2 μm, respectively). In addition, the

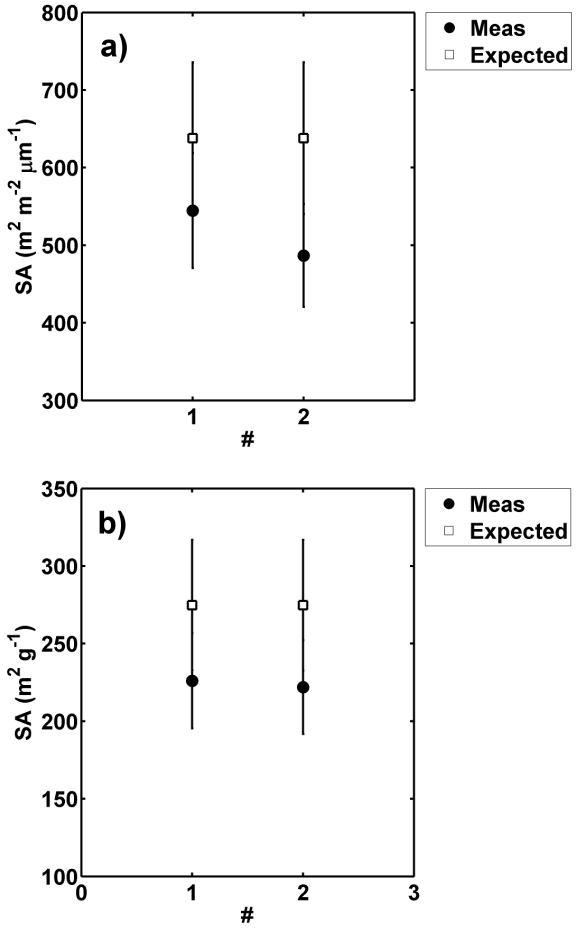


Figure 3.12: (a) Relative surface area and (b) specific surface area for Bragg stacks. Films were deposited from 60^o to 80^o. Circles indicate measured surface areas of these multilayer films. Squares indicate densities expected based on weighted averages of surface areas for single deposition angle vertical post films.

TiO_2 α = 80^o sample was divided into two sample sets (set 1, mass = 14.16 mg and set 2, mass = 8.73 mg).

Krypton gas adsorption at 77 K was carried out repeatedly on these samples, with the sorption isotherm analyzed by the BET method to obtain the surface area of the samples after each test. In between the tests the sample vials were removed from the Quantachrome Autosorb test station, backfilled with helium, and covered by a plastic membrane. The sample vials were stored under ambient conditions. Prior to each subsequent adsorption test the samples were degassed. However, to assess the impact of the offgassing step (at 150 ^{o}C for generally 12 hr - 16 hr), in some cases two tests were carried out, one following the other, with no degassing step in between. As well, some tests were carried out on sequential days with only a degassing step (and no storage) in between. Other tests were carried out weeks and months apart in order to determine the impact of aging. In

one case (for the TiO_2 α = 80^o set 1 at day 318) the sample pieces were removed from the sample vial and exposed to UV light (8 W lamp, UVP-MRL-58, held 5 cm above the samples) to determine whether this would have an impact on the measured surface area.

While more work remains to be done to fully characterize the aging of GLAD samples, the results shown in Figure 3.13 are interesting. The first result of note is that the TiO_2 samples appear to age over time to a greater degree than the SiO_2 sample. The surface area of the SiO_2 sample drops by 25% over two years, although the large uncertainties make the exact change in SiO_2 surface area as a function of time difficult to confirm. In contrast, the surface area of the TiO_2 samples drop by half.

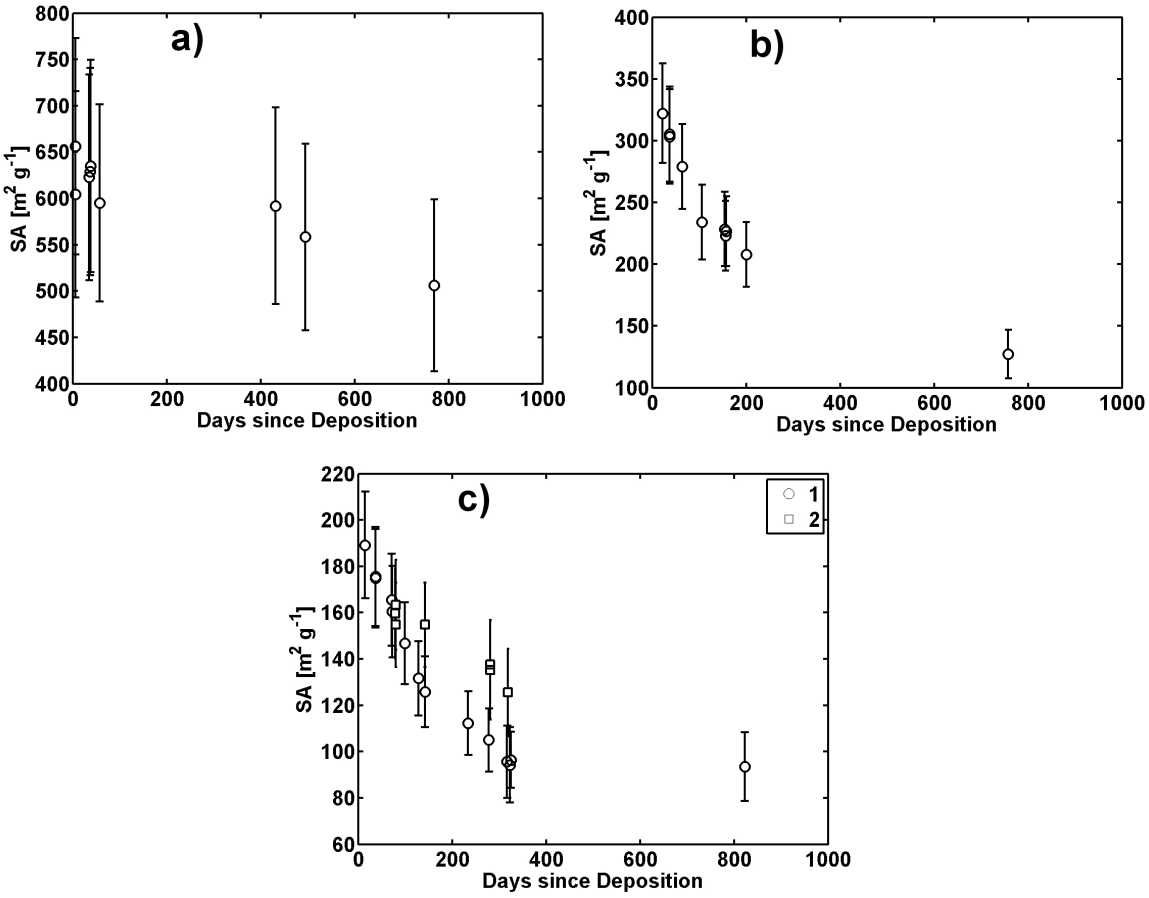


Figure 3.13: Specific surface decreases over time for (a) SiO_2 α = 60^o, (b) TiO_2 α = 60^o, and (c) TiO_2 α = 80^o set 1 and set 2 films.

The impact of sequential measurement, with or without degassing in between, appears to be small. Repeatedly measured surface areas over the course of several days remained the same. This implies little effect of the outgassing process on adsorption isotherms performed immediately afterwards. The heating of the sample during the outgassing process could however trigger a slower, longer term effect.

The results for sample sets 1 and 2 of the TiO_2 $\alpha = 80^o$ film are inconclusive. The surface areas for sample set 1 were first measured within days of the deposition. Adsorption testing began for sample set 2 at day 77. At day 77 the measured surface areas were nearly identical. However, following this day the surface area trend was different for the two samples. The surface area for sample set 2 decreased, but at a slower rate than that of sample set 1. More measurements were carried out for set 1 than set 2, implying that the measurement process itself (most likely the sample heating during the offgassing) impacts the structure of the samples. This finding requires further investigation.

The impact of UV exposure on the surface area of the TiO_2 $\alpha = 80^o$ sample (set 1) was negligible. On day 317 the sample was degassed and then the surface area was measured to be 96 ± 15 m^2 g^{-1}. On day 323 the sample was again degassed and the surface area was measured at 94 ± 16 m^2 g^{-1}. On day 325 the sample pieces were removed from the vial, placed in a glass dish and exposed under UV light for 24 hr. The pieces were then returned to the sample vial, degassed and the BET surface area was acquired. The surface area was measured to be 96 ± 12 m^2 g^{-1}, essentially unchanged. This implies that, while UV exposure has been shown to remove organic contaminants from TiO_2 surfaces, and may alter the surface chemistry, it has little effect on the surface area of aged samples. The aging process does not appear to be a contamination issue and is most likely a gradual (and perhaps accelerated by heating) change in film morphology.

The results provided here indicate that aging, whatever the cause, has a significant impact on the surface areas of GLAD films. Surface areas should therefore be measured within days of film deposition such that the as-deposited characteristics can be obtained with some accuracy. Subsequent measurements of surface area, concurrent with other characterizations relevant to the device or application, should also be carried out to assess how the surface area changes over time. Changes in film crystallinity and grain size should be probed. Further investigations are required for other GLAD films, with more heating and storage conditions tested.

3.5 Chapter Summary

Surface areas were determined by applying the classical BET method to acquired krypton gas adsorption isotherms at 77 K. Amorphous TiO_2 and SiO_2 films have higher surface area enhancements, with maximum enhancements of 880 ± 110 and 925 ± 125 at midrange deposition angles, than the polycrystalline ITO samples with 210 ± 30 surface area enhancement. Cast in terms of the specific surface area, the maximum values for TiO_2, SiO_2, and ITO films are 325 ± 40 m^2 g^{-1}, 700 ± 150 m^2 g^{-1}, and 50 ± 6 m^2 g^{-1}, respectively high relative to surface areas measured by other groups for nanostructured films.

Surface area enhancements for the ITO thin films were corroborated against ferrocene dicarboxylic acid surface-modified cyclic voltammetry results. Although the trends presented by both methods are similar, incomplete monolayer coverage by the ferrocene molecules of up to 40% - 50% could explain the discrepancy between absolute surface areas. An alternative, exciting, explanation is that the large ferrocene molecules are not able to probe the internal structure of the GLAD films. The discrepancy may instead be an indication of the relative interior to external surface area of GLAD nanostructures.

The effects of column broadening on film surface area are minimal. There is a linear relationship between surface area and film thickness for obliquely deposited TiO_2 GLAD films, with slopes of 730 ± 35 m^2m^{-2} μm^{-1} to 235 ± 10 m^2m^{-2} μm^{-1} for vertical post films deposited at angles from 70^o to 85^o. In contrast to the small impact of column broadening, the impact of column morphology on surface area was found to be more pronounced. Although helical TiO_2 films had nearly the same peak surface area enhancement, their surface area versus deposition angle curve was shifted with respect to the vertical post curve. This suggests that substrate rotation rate can be used to tailor the surface area of GLAD films.

Multilayer films were studied and found to have surface areas less than those that would have been expected based on modeling the films as stacks with individual layer surface areas equal to those of the equivalent vertical post films. This must be taken into consideration when designing nanostructured films with more than one deposition angle.

The surface area of thin film samples were seen to change over time. The effect was especially pronounced for TiO_2 samples. More work is needed to understand the underlying mechanism responsible for this change. Temperature-induced stress may alter the microscopic film structure or surface diffusion over time may smooth out the initially extremely rough surface. An analysis of the effect of annealing temperature, time, and exposure to different atmospheric conditions, coupled with investigation by high resolution TEM or AFM, is required.

The results presented here will be useful for better understanding and optimizing the properties of thin film sensors, microfluidic devices, and functionalized coatings fabricated from TiO_2, SiO_2, and ITO. Coupled with the additional findings on surface roughness and porosity, discussed in the next chapter, this new understanding of the impact of film morphology on surface properties will be especially useful when designing the next generation of obliquely deposited nanostructured devices. In addition to directing future work on sensors and microfluidic instruments, improvements in devices such as catalyst supports, energy harvesting devices and optimized liquid-crystal scaffolding structures, all of which depend on the surface characteristics of the underlying films for optimum performance, will be possible with the surface analysis provided.

Chapter 4

Pore Size Distribution Measurement of GLAD Nanocolumns

4.1 Introduction

The surface area characterization of nanostructures, discussed in the previous chapter, provides an estimation of the interface area available and thus efficiencies and reaction rates possible for processes employing these surfaces. Reaction and adsorption process parameters that depend on the available number of adsorption sites, such as catalytic and photochemical reactions, can therefore be predicted. An extension of this surface area analysis is porosimetry, a field of study aimed at revealing details about the three-dimensional and volumetric properties of a material. Characteristics such as void fraction, roughness, pore size, and feature spacing are studied. Porosimetry allows for estimates of microscopic and macroscopic interactions to be made. Areas of study include fluid incorporation, movement of biological materials through porous structures, light scattering from rough surfaces, and alignment of molecules within a porous sample volume. [1]

While properties such as inter-column spacing have been examined for several obliquely deposited nanostructures, mesoporosity has been investigated by very few groups and no data exists in the literature to describe the meso-porosity of amorphous metal oxide GLAD films [63], [186], [187]. In this current work, the understanding of the basic properties of titania and silica GLAD films was expanded by investigating their porosity. The focus of this work is on TiO_2 and SiO_2 because of their wide use in applications such as microfluidic supports, optical coatings, and humidity sensors [55], [91], [93]. The understanding and control of porosity for metal oxide devices is important for controlling light scatter in optical devices, as well as for facilitating gas diffusion in sensors and molecule detection of various sizes in microfluidic instruments. To accomplish the goal of understanding the relationship between porosity and GLAD metal oxide deposition parameters, scanning electron processing image analysis and krypton gas adsorption porosimetry have

[1] Portions of the work documented in this chapter are part of the manuscript: K.M. Krause, M. Thommes, M.J. Brett, "Pore analysis of obliquely deposited nanostructures by krypton gas absorption at 87 K", submitted to Microporous and Mesoporous Materials (Dec 2010) 30 manuscript pages.

been employed. High sensitivity krypton gas porosimetry was carried out at 87 K (instead of the more typical 77 K) in order to allow for measurements over a wide mesoporous pore size range to be carried out. Analysis of the resultant isotherms was carried out using both the classical Barrett, Joyner and Helenda (BJH) methodology as well as by applying empirical calibration based on 87 K krypton reference isotherms of highly ordered (and well-defined pore size) mesoporous molecular sieves provided by collaborator Dr Matthias Thommes of Quantachrome Inc..

4.2 Pore size distribution measurement

The determination of the micro and nano structure of a nanostructured powder or thin film sample can be carried out a number of ways. For example, x-ray diffraction (XRD) and ellipsometry have been used to study the porous nature of obliquely deposited thin films [63], [87], [136], [186]. Physical gas adsorption is another technique that can be used to determine not only the surface area, as described in Chapter 3, but also the pore size distribution (PSD) of porous samples [38], [171]. Gas adsorption porosimetry allows for the classification of a sample as having micropores (< 2 nm), mesopores (2 nm to 50 nm), and macropores (> 50 nm). Non-polar nitrogen is the most common gas-based physical adsorption probe used to determine porosity of a sample [188], [189], [190]. In addition, probes such as argon and krypton have begun to be used for samples with low total surface area since their vapor pressures are low and thus allow for highly sensitive pressure change measurements to be carried out [176], [191]. Other adsorbates, such as polar water, have also been employed for the investigation of surface chemistry and structure [187].

Much work has been carried out to analyze the structure of porous powder samples using physical gas adsorption techniques and some characterizations of thin films have also been reported. For example, argon gas adsorption has been used to measure the pore size distribution of low surface area crystalline WO_3 thin film samples [192]. This gas adsorption analysis revealed an interesting relationship between the porosity and the film deposition conditions. Similarly, the surface area properties of columnar titania films, fabricated with plasma enhanced chemical vapor deposition (PECVD), have also been characterized with water vapor and toluene [187]. Finally, obliquely deposited thin films of nanocrystalline titanium carbide have had their pore size distribution characterized as a function of deposition angle (from 60^o to 80^o) by toluene adsorption [186].

Analysis of gas sorption isotherms, such as those described in the previous chapter, provide information about the porosity of a substance. The nature of the rise in adsorbed gas volume at low pressures is indicative of the microporosity of the sample. Filling of these micropores is generally believed to be a continuous process of layer by layer deposition of adsorbant [171]. In contrast,

the behavior of the isotherm at moderate and high pressures is a function of the condensation process within mesopores. Hysteresis in the isotherm is an especially strong indication of significant mesoporosity. Hysteresis is believed to originate from a difference in metalstable pore fluid states between the condensation and evaporation processes. This is illustrated in Figure 4.1. As the vapor pressure above a pore increases, the condensate forms layers until the liquid collapses into a condensate. During the evaporation process, a gradual evaporation from the meniscus of the fluid is thought to occur, with a sudden evaporation from the pore leaving a thinly coated pore wall [171].

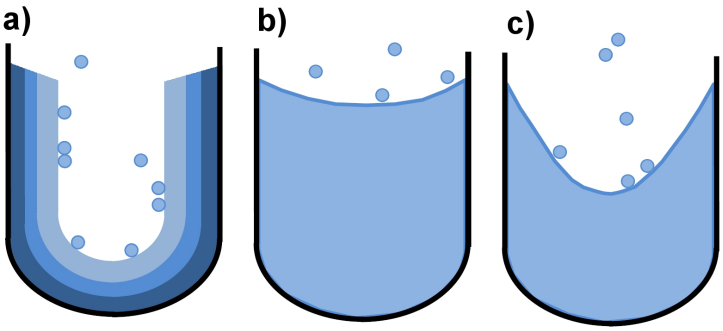


Figure 4.1: Pore condensation (transition (a) to (b)) occurs when a critical adsorbate thickness is reached on the sides of the pore walls. Pore evaporation (transition (b) to (c)) occurs by a receding meniscus as the gas pressure is decreased.

The classification of hysteresis loops are given by the international union of pure and applied chemistry (IUPAC) [172]. Figure 4.2 shows several common hysteresis classifications. The H1 hysteresis type occurs for materials that are made up of compacted spheres or have well-defined cylindrical pores. H2 hysteresis, on the other hand, characterizes disordered materials. H3 and H4 hysteresis are associated with slit-like pores or materials with plate-like particles. However, understanding whether pores are single and isolated vs connected in a three dimensional network is difficult and requires complementary methods of analysis such as SEM, TEM, AFM, or ellipsometry.

Different methods exist to analyze the gas sorption isotherms for quantitative characterization of pore size distribution. Classical methods, such as the application of the Kelvin equation using the Barrett, Joyner and Helenda (BJH) method are among the most popular [46]. This technique will be described in detail in Section 4.3. The Kelvin method applies macroscopic properties such as surface tension to a process occurring in microscopic pores. In very narrow pores the condensation of the adsorbate gas may not be fully described by classical fluid properties. Kelvin equation-based methods have therefore been found to lead to large systematic errors (underestimation of true pore size by up to 20% - 30%) in the narrow microporous and low mesoporous range [193], [194].

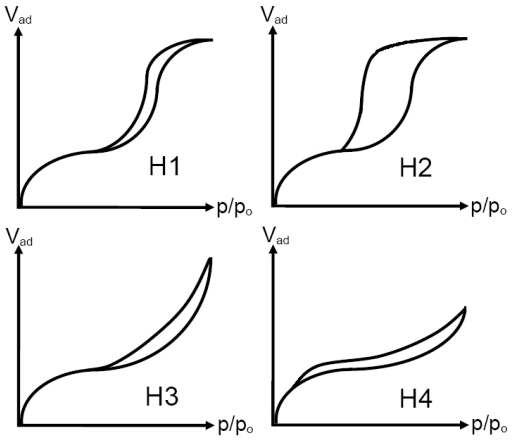


Figure 4.2: H1 hysteresis is associated with well-defined cylindrical pore channels. H2 hysteresis is common with disordered, wide pore size distribution materials. H3 hysteresis occurs for samples with slit-like pores or plate-like particles. H4 hysteresis also characterizes slit-like pores, but in the micropore region. (Adapted from [171])

Over the last decade or so progress has been made in porous materials characterization by gas adsorption. Advances have been made in areas such as experimental adsorption techniques, the development and application of microscopic approaches such as the density functional theory (DFT), and the development of methods based on molecular simulation. The DFT method, and its extension non-local density functional theory (NLDFT), provide a means to estimate the pore size distribution of a substance by modeling the more complex fluid-surface and fluid-fluid interactions. The adsorbate is not treated as a hetergeneous liquid, but instead as varying density substance based on statistical mechanics [195]. These techniques require the interactions between the adsorbate and adsorbant to be well understood and are computationally intensive. The discovery of novel ordered nanoporous materials (e.g., MCM-41, SBA-15) which exhibit a uniform pore structure and morphology have allowed for the testing and calibration of new and existing gas adsorption theories [196]. This has allowed for the development of empirical models to describe porous materials over the complete range of micro and mesopores (i.e., from 0.35 nm up to 100 nm) [174].

Despite the advances in this area, the characterization of the porosity of thin porous films remains difficult because convenient gas adsorption techniques involving nitrogen and argon at 77 and 87 K cannot be easily applied to this task [197], [198]. Such films are typically several hundred nanometers thick and, as discussed in Chapter 3, the application of nitrogen and argon adsorption is not sensitive enough to detect the small pressure changes which occur during adsorption of the gas onto the thin film surface. Krypton gas adsorption is therefore an option because of its high sensitivity, however it has not been routinely used for pore size analysis at these temperatures because the sorption and phase behavior of krypton far below its triple point temperature is still under investigation. The nature of the phase transition (e.g., gas to liquid or gas to solid) as the krypton adsorbs onto the

sample surface depends on the temperature and pore size since there is a shift in the phase diagram due to confinement of the adsorbate in very narrow meso and micropores [199], [200], [201]. For instance, for argon at 77 K (which is 6.5 K below its triple point) pore condensation does not occur if the pore diameter exceeds 15 nm [175]. Similar sorption experiments with krypton at 77 K and 87 K revealed that pore filling cannot be observed anymore in cases where the pore diameter exceeds 6 and 9 nm - 10 nm respectively [174], [191], [202].

While BJH methods have been applied to study krypton adsorption in micro and mesoporous materials [176], [191], [202] recently work has been completed by Dr Matthias Thommes of Quantachrome Inc. on an empirical calibration curve to more accurately describe the relationship between sorption isotherm and pore size for krypton adsorption at 87 K [174]. Adsorption at this temperature is less problematic than that at 77 K since at 87 K there is no evidence for gas to solid phase transitions (but rather gas to liquid) over the complete pressure range (up to solidification at 9 nm - 10 nm) [174]. The krypton calibration curve was prepared by Dr. Thommes by obtaining nitrogen (at 77 K), argon (at 87 K), and krypton (at 87 K) isotherms on ordered and high surface area silica mesoporous molecular sieves (MCM-41, SBA-15, MCM-48) which covered the pore size range from 2 nm to 10 nm. Pore size calibration curves could therefore be established by comparing the krypton results to those of nitrogen and argon. From previous work, the density of the confined krypton was taken to be 2.6 $\pm$ 0.1 g cm^{-1}, allowing for pore volume estimates to be made using krypton adsorption at 87 K [174]. This empirical calibration curve has been applied in this thesis for comparison against the results obtained by the BJH method.

4.3 Kelvin method and BJH analysis

The classic technique for obtaining pore size distributions of a porous sample is the BJH method. The BJH method uses the Kelvin equation, given by Equation 4.1 to analyze the sorption isotherm and estimate the pore diameters associated with each pressure step. [2]

$$ln\left(\frac{p}{p_o}\right) = \frac{-2\gamma V_m}{r_k RT}, \tag{4.1}$$

where γ is the surface tension of the liquid, V_m is the molar volume of the adsorbate, r_k is the radius of the condensate, R is the gas constant, and T is the temperature. The Kelvin equation is derived from applying the ideal gas law and the Laplace equation which describes the relationship between pressure difference between a liquid (p_l, condensate) and gas (p_g, vapor above the porous surface) phase and the liquid meniscus radius of curvature:

[2]The code used to calculate BJH pore size distributions and pore volumes for the GLAD films described here is provided in Appendix A.

$$p_g - p_l = \frac{2\gamma}{r_k}, \tag{4.2}$$

Equation 4.1 can be used to solve for the condensate radius. The surface tension of the supercooled liquid is extrapolated down to 87 K to be 0.0224 N m^{-1} [203]. The molar volume is 32.23 cm^3 mol^{-1}, calculated using a liquid density of 2.6 g cm^{-3} [174].

The actual pore radius of the pore is determined by assuming that, in addition to the condensate, there is a thin film of liquid that coats the walls of the pore, as described by Figure 4.3. This thickness (t_m) is also dependent on the pressure. The thickness can be estimated from the monolayer capacity of a non-porous material of the same type. In this work the Halsey equation was applied (Equation 4.3 to estimate this thickness [171]).

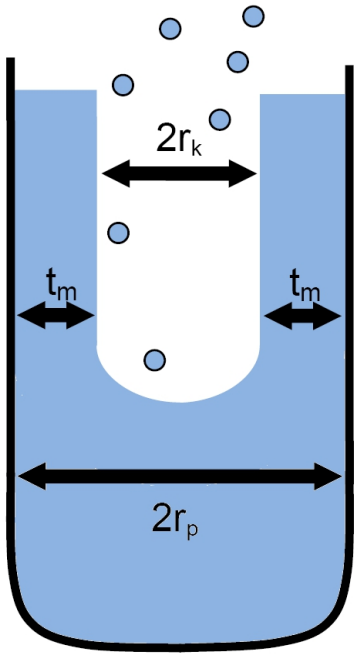


Figure 4.3: Condensate in a pore of radius r_p forms a thin film of thickness (t_m) and a meniscus of radius (r_k).

$$t_m = \tau \left[\frac{5}{ln\left(\frac{p}{p_o}\right)} \right]^{1/3}, \tag{4.3}$$

where τ is the molecular thickness of the adsorbate (2.65 Å for krypton). The pore radius (r_p) then becomes this film thickness plus the radius of the condensate:

$$r_p = r_k + t_m. \tag{4.4}$$

Having transformed the relative pressure axis of the isotherm into pore volume, the BJH technique can be applied. With the BJH technique, the desorption isotherm (or the adsorption isotherm analyzed from largest to smallest pressure) is assumed to describe two processes: 1) the sequential removal of condensate from pores of increasing smaller diameter and 2) adsorbate film thinning. At each desorption step both processes occur: condensate is removed from the pore with radius greater than that calculated in Equation 4.4 and the remaining thin film of adsorbate which coats all larger pores becomes thinner. At each subsequent step, the volume of adsorbate removed for each pore larger than that specified for the current pressure by Equation 4.1 is determined by multiplying the calculated surface area of these large pores by the change in absorbate film thickness, as calculated by Equation 4.3. The remainder of the volume removed at the given desorption step is attributed to the pore whose radius matches Equation 4.1; thus, its specific surface area can be calculated and from it the total volume of condensed adsorbate in pores of its size.

In the first stage of the BJH algorithm, the change in thickness of the adsorbed thin film of krypton (Δt_m), the mean condensate radius ($\overline{r_k}$), the mean pore radius ($\overline{r_p}$), and the change in specific condensed adsorbate volume (ΔV_{liq}, the total volume associated with all pores of a given size) is calculated for each pressure step (i = (N_{pore}-1) [highest pressure] .. 1 [lowest pressure]):

$$\Delta t_m(i) = t_m(i+1) - t_m(i), \tag{4.5}$$

$$\overline{r}_p(i) = \frac{r_p(i+1) + r_p(i)}{2}, \tag{4.6}$$

$$\overline{r}_k(i) = \frac{r_k(i+1) + r_k(i)}{2}, \tag{4.7}$$

$$\Delta V_{liq}(i) = \frac{V_{gas}(i+1) - V_{gas}(i)}{V_m}, \tag{4.8}$$

where V_m is the molar volume of the gas.

In the second stage, the specific volume of condensate in all pores (k = j .. 1) is calculated for each pressure step (j = N_{pores} .. 1), from the highest to lowest pressure. At the first, highest pressure step, the specific volume of the largest pore (dV_p) is equal to the condensate removed in that step. The condensate removed in the given step does not include the thin film of krypton still adsorbed to the surface of the entire porous sample (and still coating the sides of the largest pore). If pores are assumed to be cylindrical then the specific pore volume of the largest pore (i.e., the volume of all pores of the largest size, k=N_{pores}) is calculated from the ratio of the removed condensate volume, the pore radius and the condensate radius:

$$dV_p(k = N_{pores}) = \Delta V_{liq}(j = N_{pores}) \left[\frac{\overline{r_p}(k = N_{pores})}{\overline{r_k}(j = N_{pores})} \right]^2 . \tag{4.9}$$

The specific length of this largest pore (k = N_{pores}) can then be determined from the following equation. Note that this length is the total cumulated length of all pores of the given size (k).

$$l_p(k) = \frac{\Delta V_p(k)}{\pi \overline{r_p}^2}. \tag{4.10}$$

The pore (k), still being coated by a thin adsorbate film ($t_m(j)$), has an exposed specific inside area ($A(j,k)$) given by:

$$A_p(j,k) = 4\pi \; l_p(k) \; [\overline{r_k}(k) + t_m(j)]. \tag{4.11}$$

The algorithm now decrements the pressure by one step (j = N_{pores} - 1). The largest pore (k = N_{pores}) again visited. A small amount ($\Delta t_m(j)$) of adsorbate is peeled away from the inside of this largest pore, leaving a new exposed inside area ($A(j,k)$), as calculated by Equation 4.11. The change in specific volume of the condensate ($\Delta V_c(j,k)$) at this pressure step for this large pore (k = N_{pores}) is calculated:

$$\Delta V_c(j,k) = A_p(j+1,k)\Delta t_m(j). \tag{4.12}$$

A variable ($\Delta V_{larger}(j)$) tracks the volume of adsorbate removed from all pores at a given pressure step (j). Since only the largest pore has so far been visited, at this time $\Delta V_{larger}(j) = \Delta V_c(j, k = N_{pores})$.

$$\Delta V_{larger}(j) = \Delta V_{larger}(j) + \Delta V_c(j,k). \tag{4.13}$$

The second largest pore is now visited. All condensate of radius $\overline{r_k}$ is simulated to be removed at this point, plus some of the thin adsorbate film removed from the surface of the sample ($\Delta t_m(j)$). The specific volume associated with pores of this size is calculated as:

$$dV_p(k) = (\Delta V_{liq}(N_{pores}) - \Delta V_{greater}(j)) \left[\frac{\overline{r_p}(k)}{\overline{r_k}(j) + \Delta t_m(j)} \right]^2 . \tag{4.14}$$

The specific length of this second largest pore is calculated by applying Equation 4.10, while the exposed specific inside area of the pore is calculated from Equation 4.11.

This process is continued for each pressure step (j) down to the smallest step, visiting all pores from largest to that aligned with the given pressure step (k = N_{steps} .. j). The change in volume, due to the thinning of the adsorbate film, of the adsorbate remaining in each pore from k = N_{steps} .. j +1 (i.e., all pores except the one corresponding to j) is calculated from Equation 4.12. The cumulated volume of condensate removed for all pores at the given pressure is calculated from Equation 4.13.

At the pore corresponding to the given pressure step the specific length of the pore is found from Equation 4.10. The specific volume at the given pore size is found by applying Equation 4.14 and the exposed inside area from Equation 4.11. Once the specific pore volumes have been calculated for all pores, it is customary to present this as a differential pore volume, divided by the log of the pore radius ($\frac{dV_p}{log(d)}$) such that small features in the pore volume curve can be revealed.

The total volume of the pores in the sample, up to the point which can be measured using the adsorption technique (10 nm in the case of krypton at 87 K) can also be calculated from the adsorption isotherms. The final volume of the gas adsorbed can be recast into a liquid volume as given by the following equation:

$$V_{pore_total} = \frac{V_{gas}(j = N_{pore})}{V_m}. \tag{4.15}$$

Uncertainties in the total pore volume are calculated by taking the RMS sum of the uncertainty in the mass measurement (see Table 2.3) and the uncertainty in the molar volume measurement (4% from [174]).

$$\delta V_{pore_total} = V_{pore_total}\sqrt{\frac{\delta m}{m}^2 + \frac{\delta V_m}{V_m}^2}. \tag{4.16}$$

4.4 Adsorption experimental details

Selected samples from the surface area analysis, as described in Chapter 2, were tested not only at 77 K, but also at 87 K for porosity analysis. The 1 μm thick TiO_2 and SiO_2 films were used for this analysis. These samples were left attached to the Quantachrome test station following krypton adsorption at 77 K (and not exposed to the air or degassed). The coolant bath was then switched to liquid argon (Praxair, purity 99.998%) at 87 K. Full adsorption and desorption isotherms were acquired, from 1 x 10^{-4} to 0.999 (adsorption cycle) to 0.01 (desorption cycle) p_o, where p_o was the solidification pressure of the krypton in the pores of the samples under investigation (1.1 x 10^6 Pa). The level of liquid in the coolant bath was detected using a sensor (Quantachrome, 00080-AR-RTD).

The captured adsorption isotherms were analyzed in order to calculate the pore size distributions using both the Quantachrome software (AS1Win, v2.11), which applied the empirical krypton curve described above [174], and Matlab routines which implemented the BJH method. Total pore volumes (the volume of all pores filled to 10 nm) were also calculated from Equation 4.15. Once this data had been generated, Matlab routines were employed to compare the results obtained for the range of nanostructured films studied here. Noise in the acquired PSDs was smoothed out by applying a three point moving average filter. A log-normal fit was made to the curve. The peak in the fit was taken as the peak pore size in the distribution.

4.5 Inter-column porosity by analysis of SEM images

To complement the analysis of GLAD nanostructure mesoporosity, the inter-column macro-porosity was also analyzed by SEM image analysis. For this analysis, planar top-down views of 1 μm thick TiO_2 GLAD films, as shown in Figure 4.4, were studied. Because films below a deposition angle of 65^o appear too dense in the acquired SEM images to make any estimation of macro-porosity, only films above a deposition angle of 65^o were analyzed. Noise was removed from each image by applying a five point moving average filter in first the x and then y directions. Thresholding was then carried out to visually separate columns (white pixels) from void regions (black pixels). The Otsu method, which obtains the optimal threshold for each image based on image histogram distributions was employed by calling on the Matlab *graythresh* function for all cases except for α = 65^o, where a slightly higher threshold was employed to achieve a visually more reasonable thresholded image [204]. The histograms of the images are shown in Figure 4.5, where the dashed line indicates the threshold. The thresholded images are shown in Figure 4.4(a) through (e).

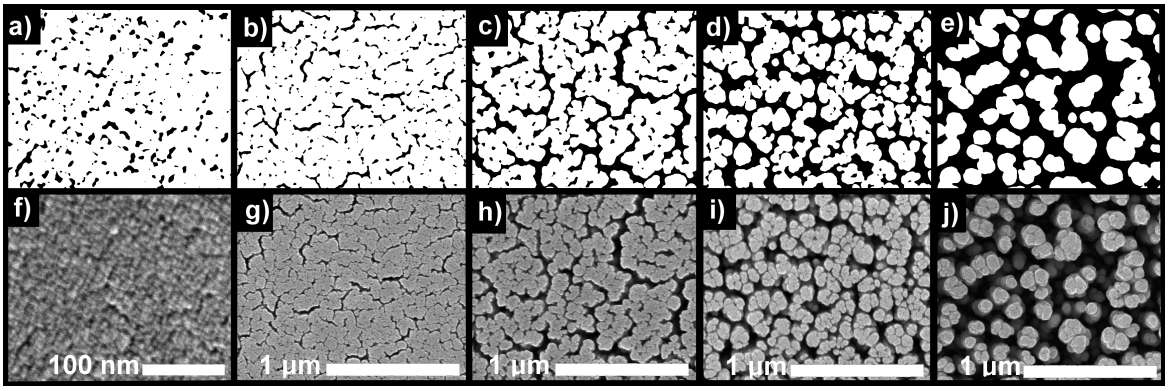


Figure 4.4: Intercolumn porosity estimated by thresholding and analyzing planar view SEMs of 1 μm thick TiO_2 GLAD films at deposition angles of (a) 65^o, (b) 70^o, (c) 75^o, (d) 80^o, (e) 85^o.

Figure 4.6 shows the difference between the achieved densities and those predicted by Tait and highlights that the densities predicted by thresholding the SEM images does not take into account the mesoporosity of the samples [147]. The thresholded density of the α = 65^o sample is lower than the trend because the SEM image is at a higher resolution, allowing for smaller pores to be imaged.

Once thresholded, the black void distances (d_{void}) between white columns were determined along the x, y, +45^o, and -45^o axes of the image. These void distances are the lengths between edges of adjacent columns. Ideally one would find the distances between adjacent columns along all trajectories, but to ensure that the algorithm would run in a tractable amount of time, only the four

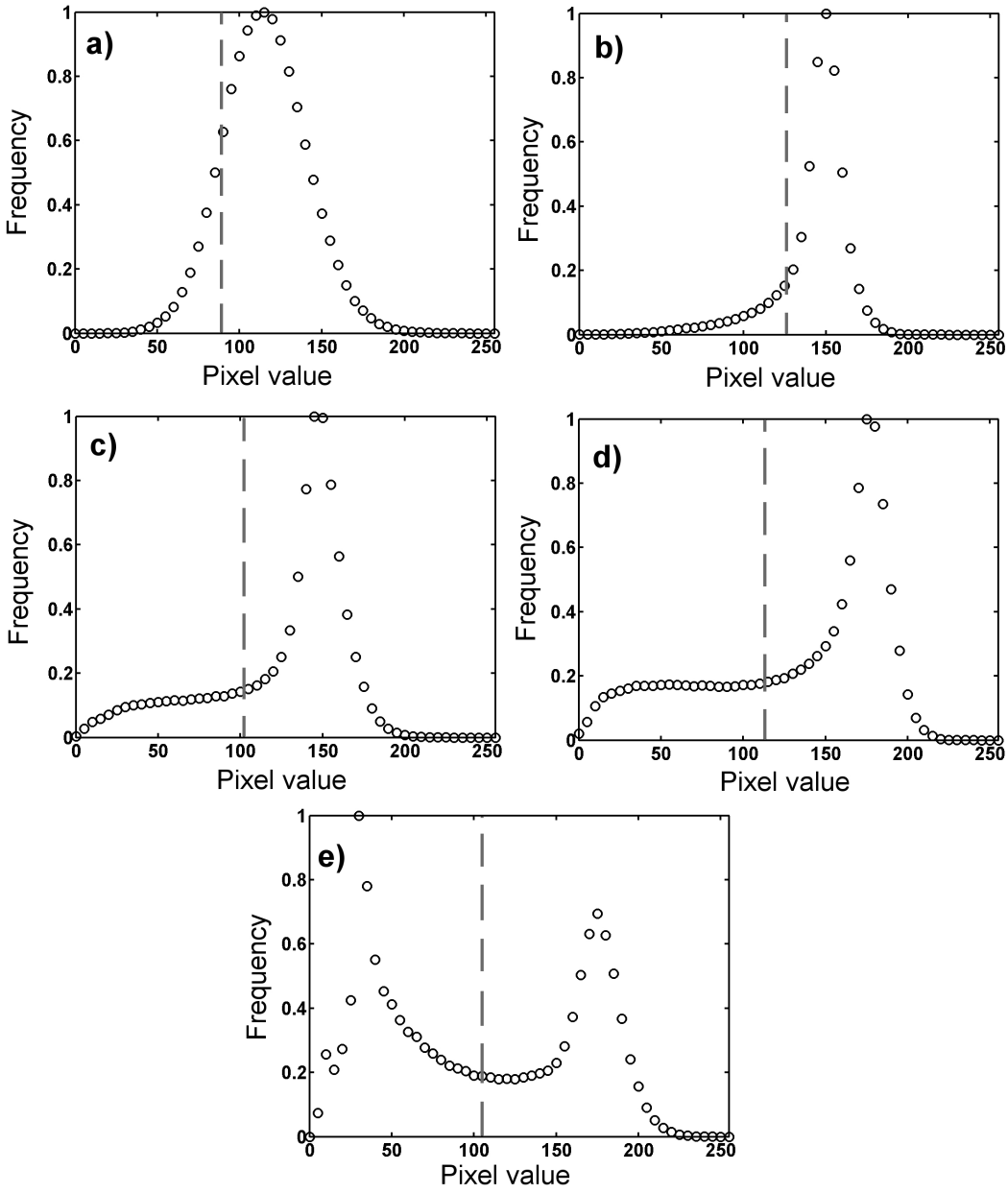


Figure 4.5: Histograms of planar view TiO_2 α = (a) 65^o, (b) 70^o, (c) 75^o, (d) 80^o, (e) 85^o GLAD film SEM images reveal optimum threshold for separating columns from void spaces.

vectors noted were traversed. The mean of the distributions of all distances between columns along the four noted trajectories was then calculated and the relationship between this void spacing and deposition angle was found. The distributions of the void spacings are shown in Figure 4.7 and illustrate an interesting features of the GLAD films: that both the peak spacing between columns and the spread of spacings increase with deposition angle.

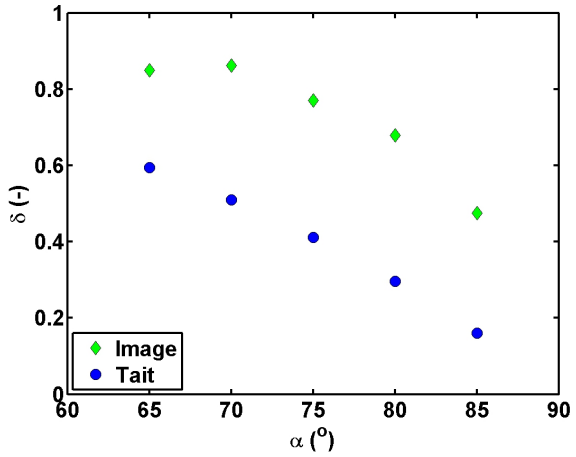


Figure 4.6: Pixel densities of thresholded planar view SEM images as a function of deposition angle.

The average void distance ($\overline{d}_{void}$) between columns was calculated from these distributions and is shown in Figure 4.8. The average void spacing drops from 128 nm for TiO_2 nanocolumnar films deposited at 85^o to 4 nm at 65^o. The spacing drops sharply from that of the most obliquely deposited film and then falls linearly for moderate deposition angle films. These results compare well with the findings of Buzea *et al.* who used two-dimensional Fourier transform analysis of top-down SEM images to estimate the void spacing between columns of 200 nm silicon vertical post films deposited at angles from 75^o to 89^o. They found that at 200 nm height the column spacing was 40 nm at 75^o and 75 nm at 85^o [205]. Further work by their group estimated that at a deposition angle of 87^o and thickness of 1000 nm the void spacing of silicon films would be 200 nm [63].

4.6 Porosimetry analysis

4.6.1 Isotherms

To determine the mesoporosity of GLAD films, both for the obliquely deposited films and those deposited at moderate angles, krypton sorption isotherms were collected for the TiO_2 and SiO_2 films. Representative adsorption and desorption isotherms are shown in Figure 4.9 and appear to be fairly similar for both materials. All isotherms were measured up to a pressure where solidification of the bulk fluids occurred (indicated by the steep step in the isotherms at p/p_o = 1). For example, in both cases, the isotherm for the $\alpha = 45^o$ sample represents a type I isotherm (according to the IUPAC classification [172]) which is typical for a microporous material. The adsorption isotherms at α = 60^o is type IV which indicates mesoporosity. The H2 type hysteresis of this isotherm is typically caused by the existence of irregular pores which give rise to pore blocking or cavitation induced evaporation (of the pore liquid) during desorption. In the case of isotherms obtained on materials

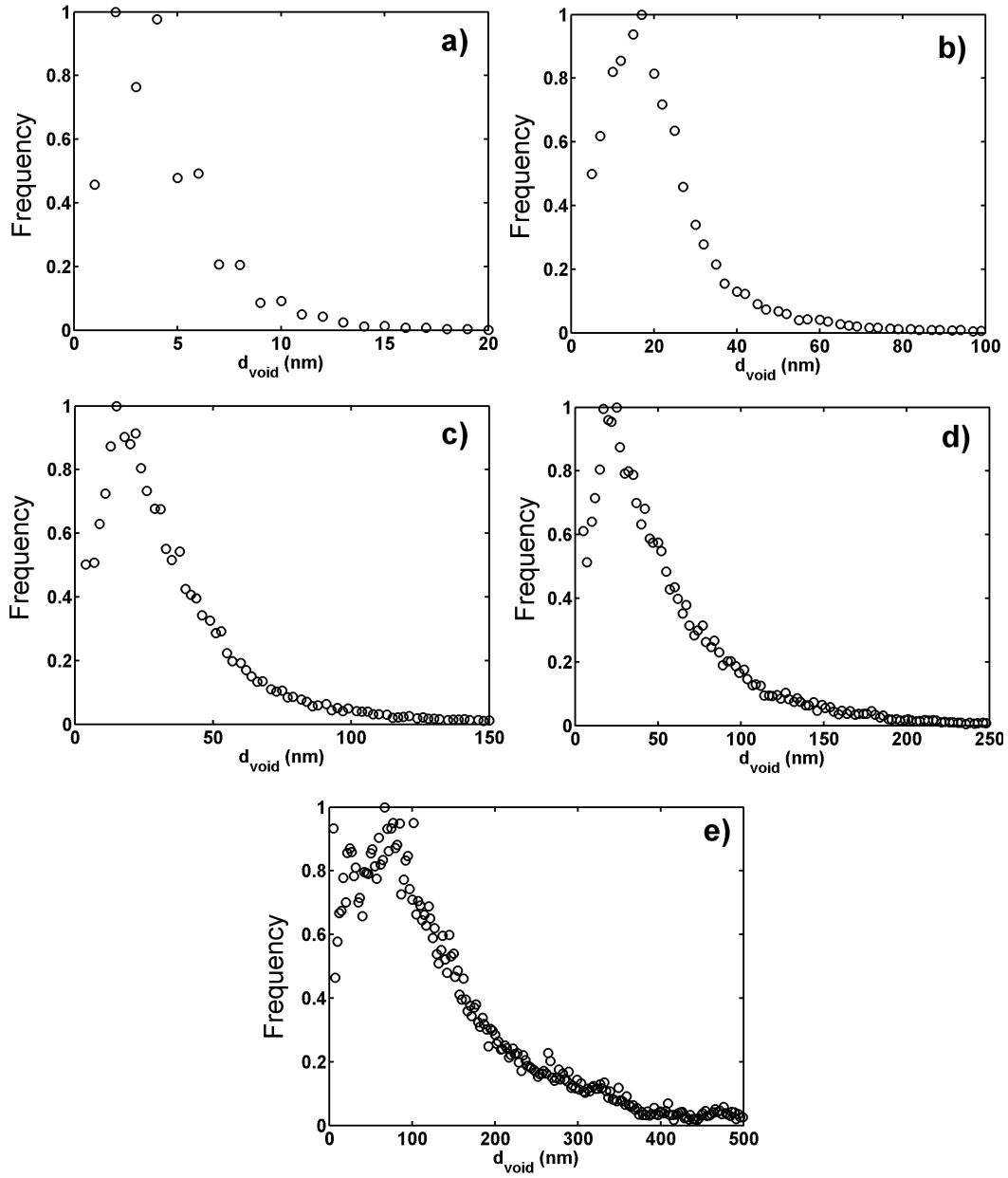


Figure 4.7: Distribution of void spacing distances between the columns of 1 μm thick TiO_2 GLAD films at deposition angles of (a) 65^o, (b) 70^o, (c) 75^o, (d) 80^o, (e) 85^o. Note the change in x-axis scale.

with a deposition angle of 70^o to 85^o one can no longer observe complete pore filling because bulk solidification occurs. Hence, the krypton adsorption method indicates that samples with obvious columnar structure have pores of width larger than 9 nm to 10 nm (i.e., the upper measurement limit of the krypton 87 K method) [173], [174].

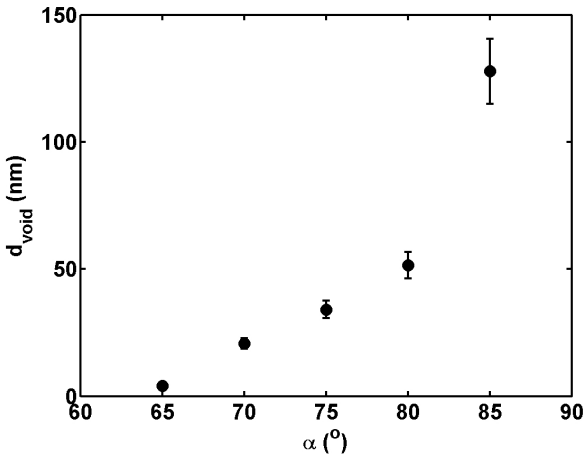


Figure 4.8: Mean intercolumn void spacing distances between columns of TiO_2 GLAD films at 1 μm thickness.

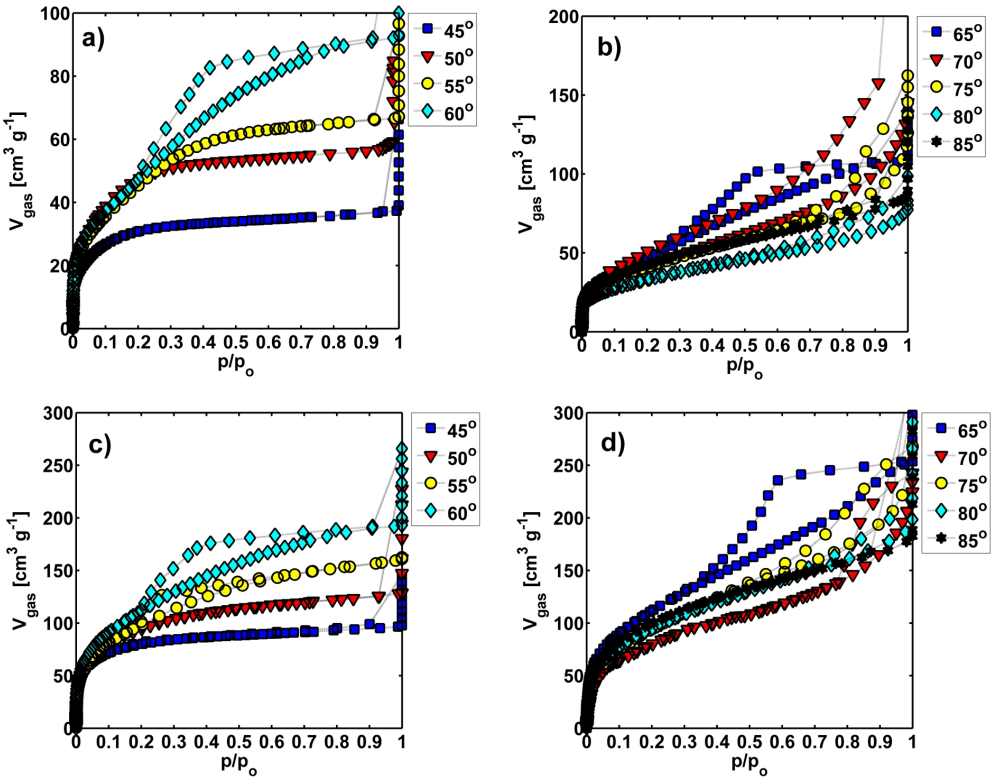


Figure 4.9: Krypton gas sorption isotherms at 87 K for (a) and (b) TiO_2, (c) and (d) SiO_2 GLAD films of 1 μm thickness deposited at the indicated deposition angles.

4.6.2 BJH Analysis

The isotherms shown in Figure 4.9 provide a starting point for pore size analysis. Using the BJH method outlined in Section 4.3 the pore size distributions were calculated. The differential PSDs (covering a pore size range of 2 nm to 10 nm) are shown in Figure 4.10. The obtained PSDs curves

for the TiO_2 and SiO_2 films are quantitatively similar, although it appears that the PSDs for the TiO_2 samples are better defined. Here one can clearly see that with increasing deposition angle, the maximum of the relatively wide pore size distribution curves (which is associated with so-called mode pore diameter) is shifted to larger values.

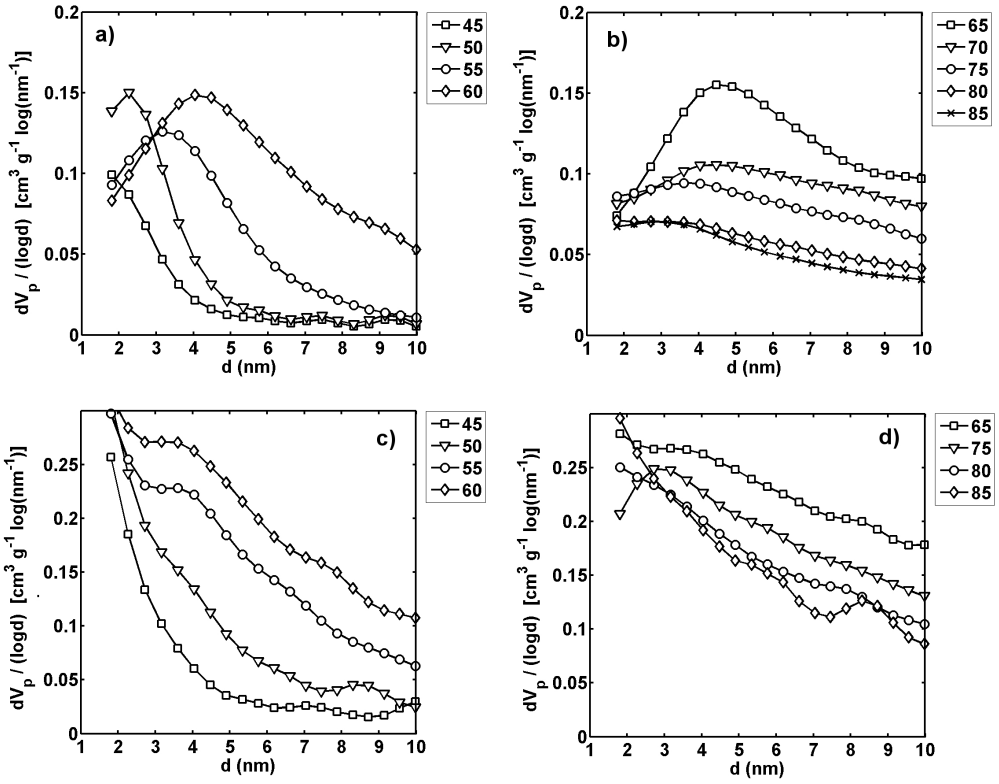


Figure 4.10: Pore size distributions as determined by BJH analysis for (a) and (b) TiO_2, (c) and (d) SiO_2 1 μm thick GLAD films deposited at the indicated deposition angles.

4.6.3 Empirical PSD calculations

The isotherms shown in Figure 4.9 were analyzed using the calibration data provided in the Quantachrome AS1 software. The resultant PSD are shown in Figure 4.11. The distributions are quite similar to those shown in Figure 4.10, although shifted towards higher pore sizes as expected from [174] and the fact that the BJH method is known to underestimate pore sizes by 20% to 30%.

An analysis of the distributions provided in Figure 4.11 now follows. For both TiO_2 and SiO_2 cases, the majority of the porosity is accounted for by micropores. As the deposition angle is increased to 50^o, the porosity is attributed to narrow mesopores and some micropores: the maximum of the pore size distribution is approximately 2.2 nm for TiO_2. In the case of the α = 50^o SiO_2

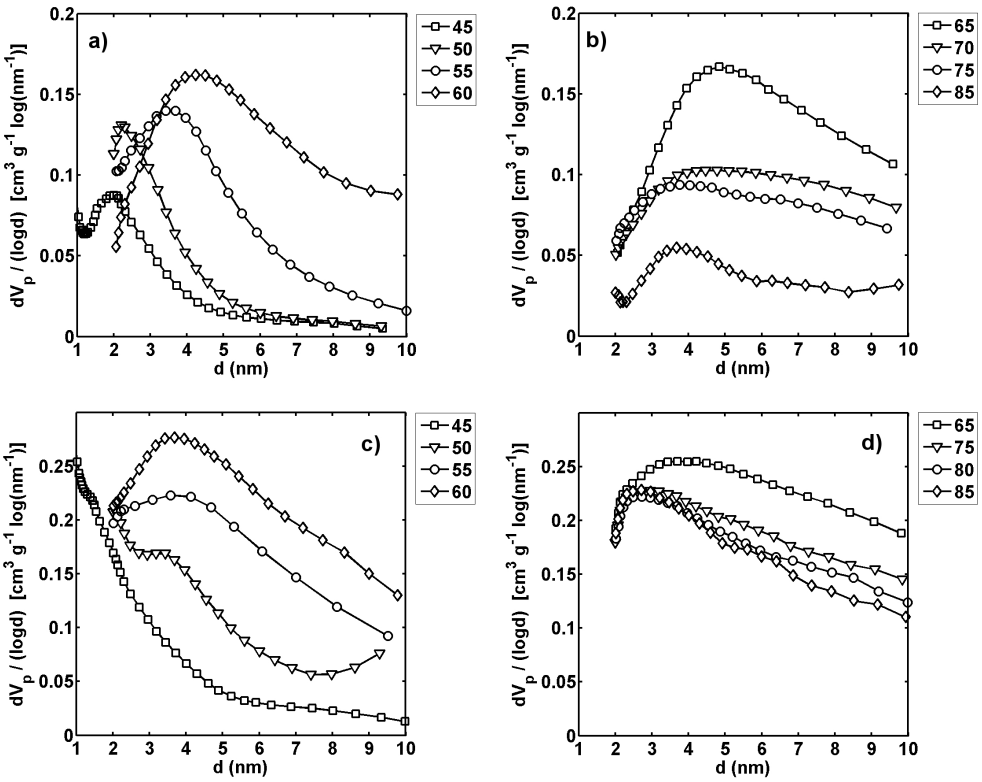


Figure 4.11: Pore size distributions as determined by empirical analysis for (a) and (b) TiO_2, (c) and (d) SiO_2 1 μm thick GLAD films deposited at the indicated deposition angles.

sample, the peak of the pore size distribution appears to be in the micropore range below 2 nm, although there is still significant mesoporosity with a peak of 3.3 nm. Both of the 55^o samples show mode pore diameters of 3 nm to 4 nm. By a deposition angle of 65^o the films reveals significant mesoporosity with a maximum in the PSD curve of nearly 5 nm for TiO_2 and 4 nm for SiO_2. This peak mesopore diameter for TiO_2 is in line with the inter-column porosity of 4 nm predicted in Figure 4.8 from the top-down SEMs shown in Figure 4.4. For samples obtained with larger deposition angles, the pore size distribution curves become much wider and less defined. As discussed earlier, for the oblique films deposited at $\alpha > 65^o$ the adsorption isotherms indicate the existence of pores larger than 9 nm to 10 nm (which cannot be detected by the krypton 87 K adsorption method), but the obtained pore size distribution curves shown in Figure 4.11b and d indicate the presence of some narrow mesopores in the range of 2 nm to 6 nm. Figure 4.12a and c summarizes the peak diameters of the PSD curves from 2 nm to 10 nm and highlights the rise in mean pore diameter with increasing deposition angle up until 65^o. The peak diameters for the films deposited beyond 65^o indicate the significant presence of mesoporosity but, because of the broadness of the distributions and the fact that there is still significant porosity indicated for the regime beyond 10 nm, this should be interpreted as columnar roughness and not a full description (i.e., does not describe inter-column spacing) of the sample porosity. Hence, for samples with such large deposition angles the krypton

method does not record details about the full porosity of the samples and underestimates the total film porosity. This is explored further in Section 4.6.4.

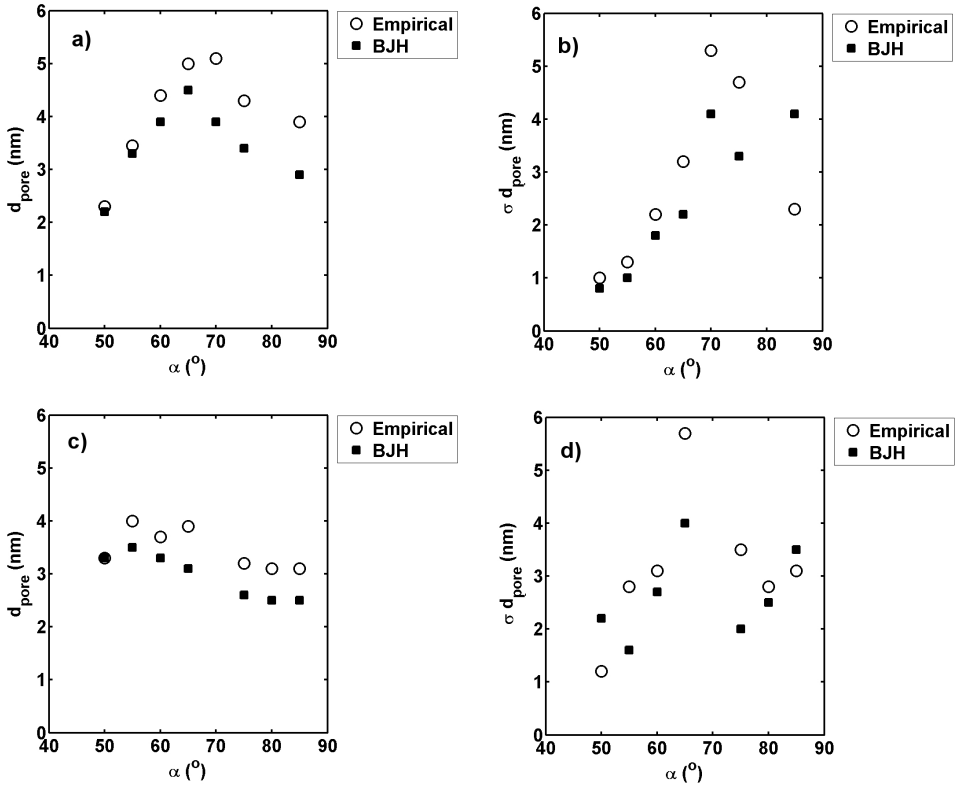


Figure 4.12: Comparisons of peak pore size and pore spread for (a) and (b) TiO_2, (c) and (d) SiO_2 1 μm thick GLAD films as determined by BJH and Empirical methods.

A comparison of the empirical and BJH methods shows that both produce results which are similar. Peak pore diameters for the BJH method are up to 30% less than those calculated using the empirical calibration data, as expected. The spread in pore distributions are also similar. This indicates that the BJH method may be applied to krypton 87 K adsorption data to obtain approximate peak pore diameters of GLAD films.

Despite the fact that the total film porosity is underestimated for the very obliquely deposited samples, the agreement between the predicted void distances (from the SEM technique) and the mesoporous peak locations suggest that the mesoporosity of the midrange deposited films may be not only an indication of sidewall roughness but also of the spacing between closely packed micro-columns. For the more obliquely deposited films the mesoporous distribution data provided in Figure 4.12 describes the roughness and porosity along and within the sides of the well-defined columns. This sidewall roughness has previously been seen in TEMs of titania GLAD films and highlights a valuable feature of obliquely deposited GLAD films - that they exhibit a porosity in not

only the macroporous regime, but also in the meso and microporous ranges [93].

Figure 4.12a and c indicate a rise and then fall of peak pore size with deposition angle. This is most pronounced in the case of the TiO_2 films. This may indicate a shift from one pore model to another. At low deposition angles the porosity is likely composed of longer channels between the densely packed columns, while at the higher deposition angles the spacing between columns broadens and the porosity is better described by smaller cavities and roughness coating the walls. Further analysis of the nature of the columnar porosity and roughness as a function of deposition angle is required; one method that could be used is a FIB-TEM analysis, discussed briefly in the following section.

Figure 4.12b and d describes the spread in pore size distributions as measured down from the peak by 25% (the 1.5 dB point). The spread in PSDs increases significantly for both TiO_2 and SiO_2 films from the low deposition angle, reaching a peak at 70^o and 65^o respectively. At oblique deposition angles the spread is large, indicating that the adsorption method does not adequately describe the porosity (or more appropriately, the porosity between columns).

4.6.4 Pore volume estimation

Specific pore volumes, calculated at 0.95 p_o in the isotherms shown in Figure 4.9, are plotted in Figure 4.13. Error bars indicate the uncertainty in the assumed supercooled density of krypton at 87 K, as well as uncertainty in film mass measurements. The pore volumes can only be calculated with reasonable accuracy for the nanocolumns deposited to 65^o. Beyond 10 nm the mesoporosity is so significant that any calculation made is not indicative of a total pore volume. Up until 65^o however, the rise in pore volume is linear with deposition angle (with a slope of 0.01 $cm^3\ g^{-1}\ deg^{-1}$ for TiO_2 and 0.005 $cm^3\ g^{-1}\ deg^{-1}$ for SiO_2).

These trends were also cast in terms of the percentage void porosity out of the total sample volume and compared to the percentage porosity calculated from the weighed density of the samples (relative to the non-porous 0^o deposition angle densities). These comparisons are shown in Figure 4.14. Here the percent pore volume is shown for films deposited at angles greater than 65^o to show how the adsorption technique underestimates the total porosity of these obliquely deposited films. The percent pore volumes calculate from the 87 K krypton adsorption data are denoted by "Meso" in this figure. These values were calculated by dividing V_{pore} by the sample volume (footprint area times thickness, given in Table 2.3). The percent pore volumes denoted by "Meas" in the figure were calculated from the measured densities given in Table 2.3 divided by the density of the non-porous film. The error in this measured porosity was calculated by taking into account the uncertainties

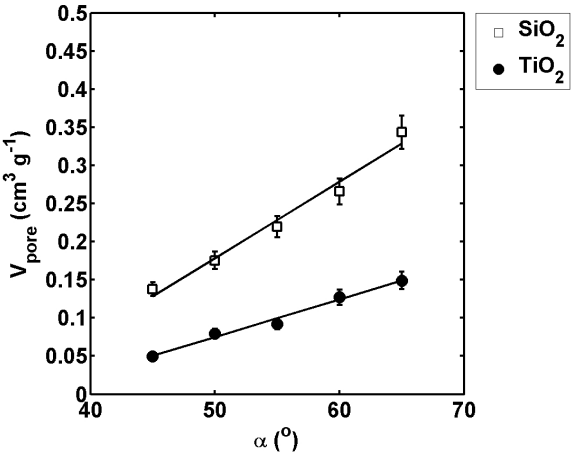


Figure 4.13: Total specific pore volume at 0.95 p_o for TiO_2 and SiO_2 1 μm thick GLAD films deposited over a range of deposition angles.

of the film thickness, footprint area, film mass and non-porous sample density. For films deposited at angles up to 65^o, the trends agree within error. The agreement is especially good for the silica films, as expected from the fact that the empirical data used to estimate pore volume was based on silica sample measurements. Differences in pore volumes, for films deposited at angles less than 65^o, are likely due to inaccuracies in the estimation of water adsorbed onto the film during mass measurement or to the choice of non-porous density for use in comparison to porous film density. The results indicate that the total porosity of the GLAD samples, deposited at angles from 45^o to 65^o, can be estimated using the krypton adsorption technique developed here. For films deposited at angles greater than 65^o, analysis of the pore volume by krypton adsorption at 87 K provides an estimate of the mesoporous pore volume for pores of diameter less than 10 nm. While a limited result, this still provides useful information about the roughness of these most obliquely deposited films. SEM analysis or other methods are required (as described above in Section 4.5) to study the nature of the inter-column pores.

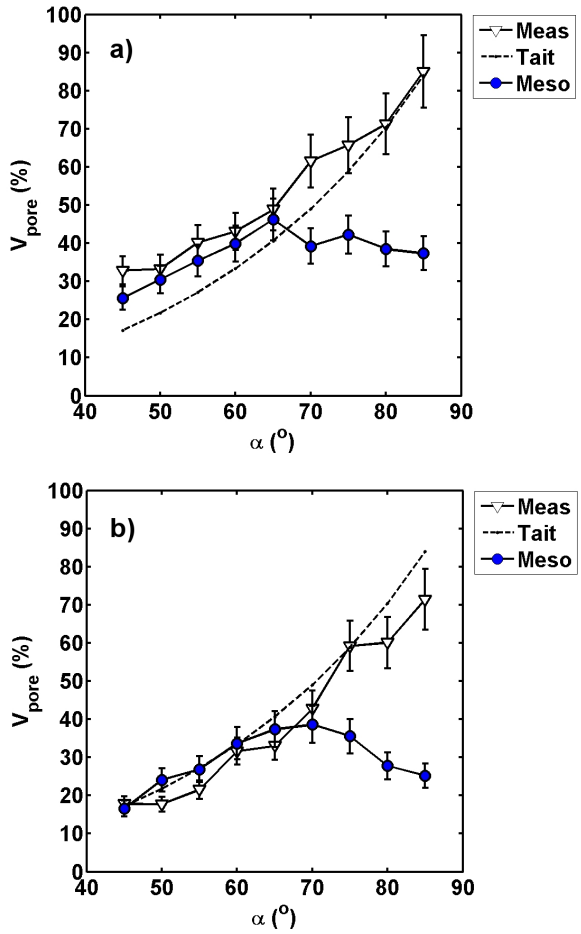


Figure 4.14: Pore volume as a percentage of sample volume for TiO_2 and SiO_2 1 μm thick GLAD films deposited over a range of deposition angles. Triangles indicate measured porosity, the dashed line shows the porosity predicted by Tait [147], and the circles show the porosity determined by analysis of the 87 K krypton adsorption isotherms at 0.95 p_o.

4.7 Chapter Summary

The understanding of the porosity of metal oxide GLAD films has been expanded through use of krypton gas adsorption and SEM image analysis. The krypton gas adsorption data for TiO_2 and SiO_2 GLAD films has revealed that they exhibit significant mesoporosity in the 2 nm to 10 nm range, while examination of SEM images has provided an estimate of their macroporosity in the 10 nm to $>$ 100 nm range. Investigation of top-down SEM images permits prediction that the mean void distance between columns is greater than 100 nm for 1 μm thickness films deposited at the oblique angle of 85^o, with spacing dropping to tens of nanometers for angles from 70^o to 80^o. By a deposition angle of 65^o, the films are quite dense and mesoporosity dominates. Krypton adsorption has also allowed for the determination of the pore size distributions of the titania and silica films. These PSDs have a log normal distribution, with peak pore diameters from 2 nm - 3 nm at 50^o to 4 nm - 5 nm at 65^o. In addition, the analysis of the adsorption isotherms for both silica and titania obliquely deposited films (i.e., those above 70^o) reveals that these films are not only columnar on a macro scale, but also consist of a mesoporous roughness and porosity.

The current work will allow for better design of nanostructured sensors, microfluidic devices, and scaffolding solutions by taking into account the meso and macro porosity of GLAD films deposited over a range of deposition angles. Further investigation into other materials and morphologies is planned. The analysis provide here also motivates for the work documented in the next chapter: the porous nature of GLAD films are investigated in full three dimensional detail using a dualbeam FIB/SEM device.

Chapter 5

Growth scaling and 3D reconstructions

5.1 Introduction

GLAD nanostructures typically exhibit columnar structures with morphology that is determined in part by the source material and in part by the deposition conditions. Through oblique deposition and controlled substrate rotation, this physical vapor deposition process enables fabrication of a range of columnar nanostructured films that have been employed in a variety of applications. Understanding how these films grow and the characteristics of their columnar structure is therefore essential not only for optimizing the technique itself but also for advancing the consequent applications. Without knowledge of the exact column structure, the column growth processes and competition, device optimization is either difficult or hit-and-miss. Application of a dual beam (focused ion beam (FIB) and scanning electron microscopy (SEM)) instrument to segment and then reconstruct a GLAD columnar array has allowed us to better analyze and understand the growth dynamics and morphological characteristics of these nanostructures. [1]

Important properties of GLAD columnar arrays, such as their high surface area and column roughness have been investigated by various groups using gas adsorption porosimetry and transmission electron microscopy (TEM) studies [165], [166], [169], [206]. Other groups have used atomic force microscopy (AFM) to image the tips of GLAD columns to measure parameters associated with roughness [207]. Optical ellipsometry and scanning electron microscopy are often used to investigate the morphological properties of these films, such as their height, width, and density as a function of column height [104]. Each of these approaches has yielded important information about GLAD columnar structure; however, each has its limitations and recovery of the film's three-dimensional structure of the film is challenging. The advantage of SEM imaging is that it offers significant depth of focus. A drawback is that the detected electron signal can contain a significant

[1] Portions of the work documented in this chapter have been published in: K.M. Krause, D.W. Vick, M. Malac, M.J. Brett, "Taking a little off the top: Nanorod array morphology and growth studied by focused ion beam tomography", Langmuir 26 (2010) 17558-17567.

component that is generated in the near-surface volume from secondary electron collisions rather than thin surface layer of the film [22]. AFM images are subject to distortion due to the interaction of the probe with the sample, as well as the finite size of the probe. The accuracy of porosimetry is limited by assumptions that must be made about the characteristics of the gas adsorbate and its interactions with the surface. Use of the TEM allows for individual columns to be imaged with sub-nanometer resolution, but the spatial relationships between individual columns are not preserved and measurements such as column spacing as a function of column height are not possible. A technique which provides a full three-dimensional view of the columnar structure is therefore required.

A method to probe the full three-dimensional properties of GLAD nanocolumnar arrays would be useful for analyzing one of the most important parameters associated with basic structural characterization: the growth scaling of the column diameter, spacing, and density as a function of film thickness. The growth scaling of these GLAD films has been investigated by a number of groups, with the outcome that the evolution of the columnar cross sectional diameter ($w(t)$), at a given height (t) above the substrate, is described by the following power law:

$$w(t) = w_o t^P. \tag{5.1}$$

where P is the characteristic growth parameter and w_o is an empirical material-dependent constant describing a characteristic width at a unit height. SEM image analysis by Mukherjee *et al.* showed that the growth exponent for metal GLAD films varies from 0.3 to 0.7 depending on the substrate temperature relative to the melting point of the deposited metal [208]. Kaminska *et al.* studied the SEM images of silicon films deposited at 87^o with different thicknesses and determined that the column diameter growth parameter was 0.5 for these films [63]. Buzea then went on to show that the scaling exponent appears to drop from 0.6 to 0.3 as the glancing angle is varied from 75^o to 89^o for silicon films [205]. Additionally, Taschuk *et al.* studied metal oxide GLAD films deposited over a range of deposition angles and thicknesses to find that the growth parameter increases with deposition angle from approximately 0.3 to 0.6 for angles from 70^o to 89^o [209]. To date, the investigation of this parameter has required the growth of multiple GLAD films, each of a different thickness, in order to determine the dependence of the scaling relationship on column height. A method to study these nanostructures in full three dimensional form, without relying on multiple depositions, would therefore be beneficial.

Depending on the material and deposition conditions, GLAD columns also exhibit an important property: bifurcation. This property is manifested as a single column gradually splitting into two or more individual columnar structures as film growth proceeds. Several researchers have investigated branching and bifurcation in these structures [210], [211]. One of their interesting findings is that the number of branches per column evolves as the column grows. This compliments the shad-

owing competition that occurs between adjacent columns and suggests some competition between branches within one column or between a given branch on a column and the neighboring column. Understanding and controlling bifurcation is essential to fabrication of uniform structures of consistent architecture and pore size distribution (e.g., in photonic crystal and optical filtering applications) where the uniformity of the columns greatly impacts the optical performance [76].

To improve the understanding of the three dimensional characteristics of GLAD columns, the imaging and sectioning capabilities of a dual beam FIB/SEM instrument have therefore been employed. Tomographic reconstruction techniques have been used to create a three-dimensional model of the GLAD film and image analysis routines have been used to assess its growth scaling and bifurcation behavior.

With dual beam tomography, multiple slices are made into the sample volume. For each slice, the focused ion beam removes a thin layer of material of fixed thickness, adjustable down to a few nanometers, allowing for detailed analysis of samples. Conditions (e.g., SEM beam energy) can be adjusted such that only a thin volume at the top of the sample contributes to the detected SEM signal.

FIB tomography has been used by a number of groups to slice, image, and reconstruct micro and nanostructured volumes. Some of the earliest work in this area was performed by Sakamoto *et al.*. This group developed a dual FIB and scanning Auger microprobe (SAM) instrument to simultaneously mill and probe the three-dimensional characteristics of a microstructured volume [212]. The FIB beam, oriented along the face of the sample, was used to shave off sample cross-sections while the Auger beam was directed normal to the surface for concurrent imaging. Their experiment proved the concept of a milling and imaging system on a test sample of a several micron sized SiO_2 particle embedded in an indium substrate. A milling resolution of several hundred nanometers was achieved and eighteen cross-sectional images were taken over the course of several hours.

Early three-dimensional reconstruction was carried out by Kubis, Hull, and team. By employing secondary electron and ion detection as the FIB milling occurred, a series of cross-sectional images were acquired on a heterostructure consisting of nickel, aluminum, and chromium components [213]. Alignment of the images were performed manually using Adobe Photoshop and then displayed using MATLAB three-dimensional plotting routines. They then went on to demonstrate their technique on 150 nm diameter Ge quantum dot stacks buried in Si films. They were able to achieve a 20 nm milling resolution and used their technique to show how the number and diameter of the quantum dots in the columnar stacks changed as a function of stack height, a set of metrics not unlike those that this present thesis explores [214].

Dual SEM-FIB instruments were subsequently used for reconstructions and characterizations of grain boundaries and crack propagation in Ni-based superalloys [215], [216]. Milling resolutions of 250 nm to 300 nm were used, with pixel resolutions of several tens of nanometers. Properties such as grain area, number of neighboring grains, grain edge shape were assessed using in-house developed codes and manual processing. For example, to study crack growth through a volume, Holzapfel and coworkers marked the crack plane in each slice image manually and then used custom routines to analyze characteristics of the volume on either side of the crack. Technology improvements have allowed for increased 3D voxel resolution, and other groups have studied similar samples with resolutions on the order of 15 x 15 x 15 nm^3 [217].

In this chapter, the development of a FIB/SEM tomography technique for sectioning and 3D reconstruction of arrays of GLAD nanocolumns is described. The initial FIB milling stage, carried out in collaboration with Dr Doug Vick of the NRC-NINT, is also documented. In addition, the growth scaling and bifurcation characteristics of a representative metal oxide GLAD nanostructured thin film has been investigated. Individual columns within the volume have been isolated through new sectioning and imaging routines such that bifurcation and scaling behavior were tracked. This is the first direct investigation of properties of a single GLAD column as a function of thickness. These findings and techniques should enable improved optimization and understanding of the growth of GLAD nanostructures.

5.2 Focused ion beam tomography

The focused ion beam instrument was developed following research into its keystone technology, the liquid-metal ion sources (LMIS), in 1961 [218]. The instrument was subsequently commercialized in the 1980's as a sample etching tool for use in the microelectronics industry [27]. The focused ion beam provided a faster, and fewer charge-induced distortions, alternative to electron beam lithography. It did not achieve the high resolution offered by electron beam bombardment, but did have the advantage of high mass atomic penetration for removing material quickly. The FIB has since gone on to be a powerful tool in the field of nanotechnology, used for milling, sample preparation, and nanostructure fabrication.

The core of the instrument, the FIB column, is shown in Figure 5.1. The purpose of the column is to generate and focus a beam of high energy ions onto a specimen surface. The ion beam is generated by the liquid metal ion gun, emission tip, and extraction electrodes. In most FIB systems, heavy metal gallium (Ga+) ions are used because of their low melting point (29.7 ^oC) and the ease at which they can be ionized. This low melting point allows the liquid phase to be maintained in the metal reservoir, where it undergoes little interdiffusion into the tungsten (melting point 3422 ^oC) needle tip. Gallium also has a high boiling point (2477 K) and low vapor pressure ($9.31x10^{-36}$ Pa at

29.9 ^{o}C), ensuring purity during storage in the reservoir, and is viscous, allowing for wicking along the tungsten needle tip [27]. These heavy atoms also efficiently penetrate into a sample, removing material and generating the secondary electrons required for basic imaging.

The liquid gallium metal is drawn from the reservoir onto a tungsten needle with an apex of approximately 5 nm in diameter. The potential difference between the tungsten tip and the aperture (on the order of 10^{10} V m^{-1}), as well as surface tension of the liquid metal, pulls these atoms down to the tip of the needle, where they are ionized by the high field near the tip (known as field evaporation), evaporated, repelled from the tip, and accelerated down the column. Typical acceleration voltages are 30 keV [219].

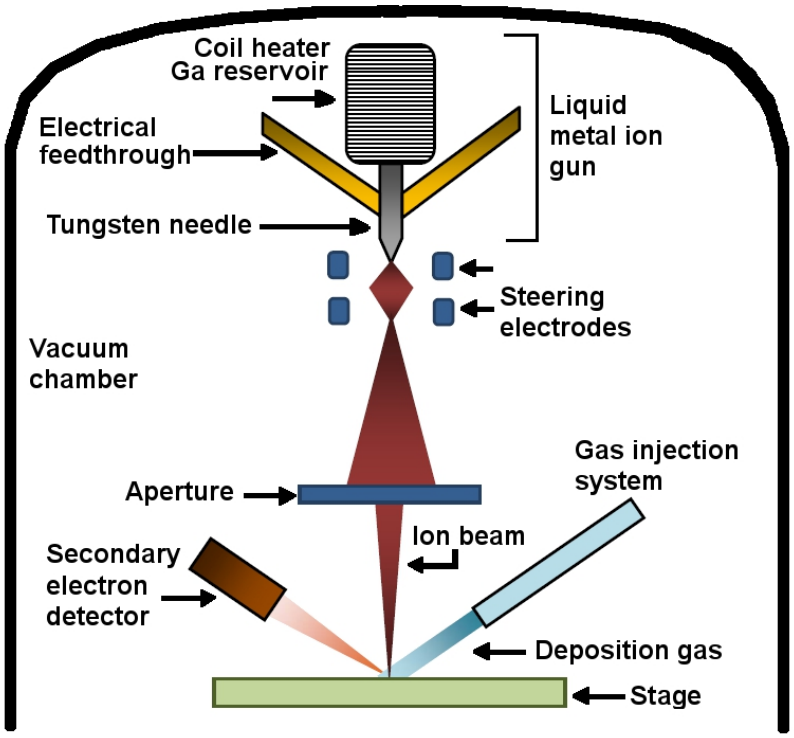


Figure 5.1: Schematic diagram of focused ion beam instrument.

Focussing and scanning electrostatic lenses steer the beam. As opposed to an SEM, which uses magnetic fields to steer the traveling electrons, electrostatic lenses are used for the FIB. The heavy ions require a force larger than that exerted by a magnetic field over the length of a typical "bench top" column (several feet tall) so that instead electrodes running along the ion beam are used. However, aberrations, even more severe than those seen in SEMs, are common in ion beam instruments. Spherical aberration, due to fringing effects at the edges of the focussing lenses, causes ions to deflect from their ideal trajectory. Careful lens design and the use of physical apertures can limit this effect (a range of aperture window sizes is typically provided with each instrument). Chromatic

aberration, a spread in focal points for the beam, is the result of spread of ion energies. This is a fundamental property of the source and is also corrected somewhat with the use of lenses and apertures. Finally, asymmetry in the focusing field results in a skew to the cross-section of the beam. This astigmatism is corrected dynamically by the user during instrument operation with a set of electromagnetic coils which reshape the beam.

The focused ion beam may be used to deposit material onto a substrate surface using the gas injection system. Nanofabrication is possible when the interaction between the high energy ion beam and a surface-adsorbed layer of gas is taken advantage of. The adsorbed gas molecules are decomposed by the incident ions into a volatile species which leaves the surface and a nonvolatile product which remains deposited. For example, a common deposition material in FIB processing is tungsten. This deposition employs the precursor $W(CO)_6$ which decomposes, when struck by the ion beam, into W and the volatile C and O components. For all such depositions, a balance must be struck between the deposition and ion-induced surface sputtering rates. This is accomplished through optimization of ion beam current, dwell time, overlap, and scanning area.

Similarly, gas assisted etching of a surface (such as Si) is possible. In the presence of the ion beam, a gas precursor (such as XeF_2) decomposes and free radicals (in this case, F-) react with the surface. Ideally the gas precursor does not react readily with the surface, only doing so when bombarded with the ion beam. Volatile products (e.g., Xe(g) and SiF_4(g)) then leave the surface, resulting in localized etching. This technique is useful for targeted etching of regions composed of certain materials.

FIB milling can also be carried out for micromachining and lithographic purposes. While conventional optical or e-beam lithography produce high-resolution features, they are limited to soft materials (such as photo or e-beam resists) [27]. The patterning achieved with FIB techniques depends primarily on the energy and size of the beam such that it can be used to mill both soft and hard materials. E-beam lithography can also be a slow process as the electrons are not as efficient as ions at extracting material from the sample. Both direct write and projection printing are possible with the FIB. Direct writing is the normal mode for the FIB and uses a beam deflector just above the sample to control the trajectory of the ions on the substrate. Projection printing uses one of two mask technologies to transfer a pattern onto a surface which is coated with a ion beam resist. The first type of mask employs silicon regions of varying thicknesses to act as absorber and ion-transparent membranes. The second, more conventional, mask uses an etched metal foil which is open to the passing ions.

In this present work the direct write type of milling is used. This technique allows for three dimensional regions to be milled from a sample volume with precision on the order of tens of nanometers.

However, milling artifacts must be dealt with in order to ensure optimal performance. For example, implantation of gallium and sputtered atoms onto the surface and sides of the milled volume is common [219]. To reduce their impact on the etched volume, trapezoidal milling is sometimes employed such that the sides of the volume do not angle inwards after extensive sectioning. In addition, ion damage can occur in softer materials. This must be taken into consideration when choosing subjects for FIB milling. Finally, curtaining occurs when differential milling occurs on heterogeneous samples (an example of this will be seen later in Section 5.3.2). This often occurs when a material is composed of both soft (low atomic number) and hard (high atomic number) portions or when the surface topology is not flat. The resultant FIB milled surface will be grooved and will resemble a curtain. This is often reduced by repeated lower energy passes of the FIB beam (i.e., repeated polishing steps) or through the pre-milling deposition of a "smoothing" layer of carbon or metal to the face of the sample. This layer helps not only to smooth out any surface roughness, aids in controlling the rate of removal of the material below [220], [221].

Preparation of TEM and atom probe (AP) specimens is possible with FIB milling. Atom probes are nanometer-scale needle-shaped sample specimens which are evaporated layer by layer to determine the structure of the original sample. The probes must be needle-shaped to induce a large enough electric field such that the atoms at the tip will evaporate and ionize with the application of an applied voltage [222]. Time of flight measurements are then performed to determine the composition, by mass to charge ratio, of the original sample. To create the needle-like AP samples the FIB mills an increasingly smaller diameter ring with increasingly lower ion current into the specimen. The sample is then bonded to a micromanipulator via a platinum deposit. Final cuts are made from the initial sample, and the needle is lifted away for use in the AP experiment [223]. Similarly, ultra-thin TEM samples are prepared by milling a sample from two sides, exposing only a thin lamella [224]. TEM cross-sections tens of nanometers thick can be milled from a range of materials using the FIB. Unlike using lithography and lift-off for TEM sample creation and electro-polishing for AP preparation, heterogeneous samples of organic and metal and semiconductor mixes are possible with the FIB due to its ability to cut through any material [225].

The advantage of the FIB is that it can not only be used for milling and nanofabrication, but can also be used to image the exposed sample surface. The secondary electrons that are emitted as the ion beam interacts with the surface may be detected to infer properties of the surface. Several drawbacks to this usage have, however, led to the development of dual-beam FIB-SEM instruments. For example, the additional dwell time required for imaging by the single-beam instrument results in damage to the surface. In addition, charging effects, resulting from the cumulative impact of both the positive ions bombarding the surface and ejection of negative electrons, make imaging problematic and induce beam deflection. Finally, the characteristic X-rays and backscattered electrons that add valuable quantitative information about a sample are not generated with the surface-ion beam

interactions [219].

The dual-beam FIB/SEM instrument combines a FIB column and an imaging SEM column in one instrument, as illustrated in Figure 5.2. The SEM is held at a fixed angle relative to the FIB column, typically between 45^o to 54^o. This allows for both columns to see the sample surface as it is being milled. The two beams are controlled with a single software interface to ensure that co-focussing on a single coincidence point can be accomplished. Sample stage movement and rotation can therefore be carried out without changing the setup of the columns. Software can also be used to compensate for imaging foreshortening of the SEM images, due to imaging the sample surface from an angle.

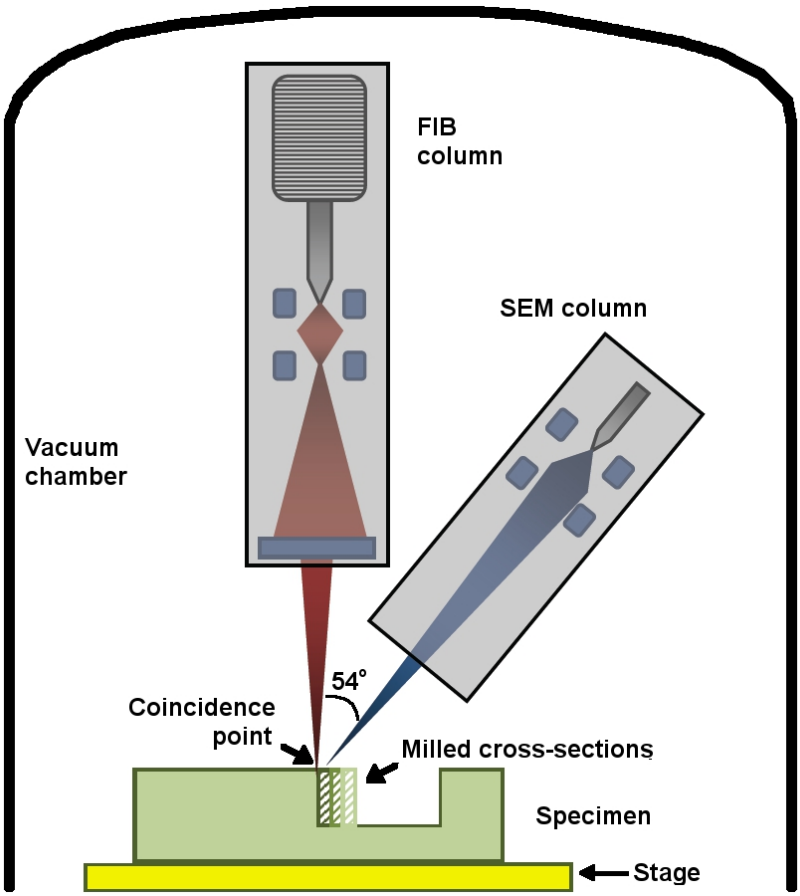


Figure 5.2: Schematic diagram of dual-beam FIB-SEM instrument.

The advantage of using the SEM for imaging during concurrent FIB milling is the increased signal to noise ratio achievable from detecting either secondary electrons (SEs) or backscattered electrons (BSEs). The low energy SEs are generated through multiple collision events within the sample volume. Secondary electrons are pulled by the strong magnetic fields in the column towards the InLens detector [22]. The trapped electrons travel on spiral trajectories around the magnetic flux lines towards a luminescent scintillator, where photons are generated. A light waveguide then directs

the generated photons outside of the instrument's vacuum chamber to a detector. Positive or negative biased Everhart-Thornley (ET) detectors are also employed to capture off-axis SEs and high energy BSEs, respectively, useful for creating images with enhanced topographic contrast [24]. This ET detector consists of a high voltage scintillator, excited by the incoming ionizing electrons, within a low voltage Faraday cage. The detector traps nearby electrons and then accelerates them towards the luminescent scintillator. As with the InLens detector, the generated photons are guided outside of the chamber to a detector. A third alternative is an energy selective backscatter (EsB) detector, used to capture the high energy BSEs backscattered through elastic collisions near the sample surface. This EsB detector provides images with strong Z (atomic number) contrast and is therefore used when sample composition information is desired.

The dual FIB-SEM instrument enables three-dimensional tomographic recreation of a sample volume. The layer by layer imaging that occurs as the sample is milled allows for a volumetric data set to be created. This in turn makes possible the reconstruction of the sliced-and-imaged volume using custom or commercial software packages. To enable high spatial accuracy, fiducial or reference marks must also be milled into the sample volume prior to slicing [213]. Image processing routines are then used to normalize image contrast and ensure feature alignment. Quantitative information about surface area, volume and porosity can be then determined, making this instrument an incredibly powerful tool.

5.3 Experimental details

5.3.1 Sample fabrication

Several steps were required to prepare the specimens for tomography. The first was to deposit an amorphous 5080 $\pm$ 50 nm TiO_2 (material source, Cerac; 99.9 % pure rutile) vertical column film by electron-beam physical vapor deposition evaporation (AXXIS, Kurt J. Lesker). For this analysis a titania GLAD film was studied because of the wide use in a number of applications. As discussed previously, titania is an excellent material for antireflection coatings, photonic devices, humidity sensors, and photovoltaics so provides a good test subject for this first application of FIB tomography to GLAD films [93], [104], [105], [99], [100].

The film was deposited at an incident vapor angle of 85^o, with continuous substrate rotation. Stoichiometric TiO_2 was ensured by the introduction of O_2 gas flow to maintain a deposition pressure of 7 x 10^{-5} Torr, followed by postdeposition annealing in air at 100 ^{o}C for 24 hr [128], [129], [130], [131]. During the deposition, a typical deposition rate onto the film substrate of 5 Å s^{-1} was

achieved. Substrates were not dynamically temperature controlled during the deposition process and reached a steady-state 70 ^{o}C during a deposition [125].

To ensure that only the exposed surface would be imaged during FIB slicing, a filling step was carried out. This minimized the detection of columnar features from deeper into the film volume. Following deposition, the porous structure was filled with a polymer-based photoresist (HPR504, Arch Chemicals) using a standard spin-in process repeated twice to ensure photoresist infiltration: (i) 500 rev min^{-1} spin for 10 s, (ii) 3000 rev min^{-1} spin for 40 s, and (iii) a soft bake at 115 ^{o}C for 90 s [226]. This filling also ensured good contrast between the titania nanostructure and the surrounding void region.

The sample was imaged before and after polymer filling by a Hitachi cold field emission S-4800 SEM, as shown in Figure 5.3. An artifact of the filling process is that the columns become slightly inclined. This is likely due to the spinning step or to internal stresses during curing. The resultant inclination angle for the present sample was approximately 10^o, reducing the film thickness to 5000 $\pm$ 50 nm.

5.3.2 Tomography preparation

In preparation for tomography, the twice-filled GLAD sample was cleaved from the wafer substrate, coated with a thin (20 nm) coating of gold to ensure good conductivity, and mounted onto specimen holders with an inclined mounting surface using heat-bondable silver epoxy. The sample was then mounted onto the six-axis (three translation, three rotation) FIB stage of a Zeiss NVision 40 Crossbeam Workstation, which features a vertically aligned SEM column and an FIB column oriented at 54^o with respect to the vertical. Microscopy was performed at the edge of the sample; the inclined orientation of the sample allowed edge milling to be performed from two orthogonal directions, as described below. [2]

The sample was first oriented such that the FIB probe was normal to the wafer substrate, as shown in Figure 5.4a, b. A high current probe FIB probe was used to mill into the sample edge approximately 5 μm, exposing a fresh surface of the sample and removing any edge defects introduced during the cleaving step. The exposure typically extended 50 μm along the sample edge and at least 10 μm deep. The SEM was then used to select an appropriate site for the tomography step. This portion of the exposure was then subjected to a sequence of polishing steps using FIB probe currents from 3 nA down to 150 pA. The objective of this step was to planarize the edge by removing as much as possible the curtaining effects that resulted from the different milling rates of the TiO_2

[2]The FIB milling was performed with Doug Vick of the NINT-NRC as part of the work published in [227]

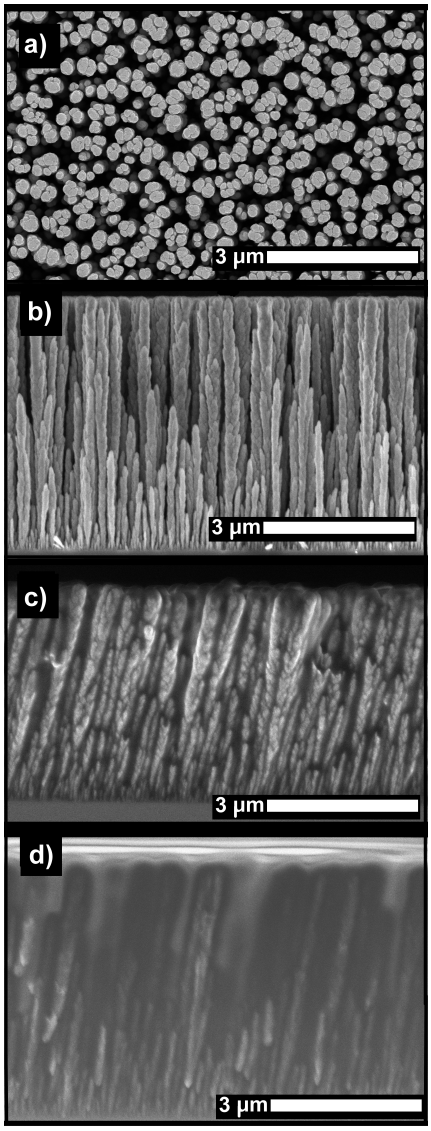


Figure 5.3: (a) Top-down and (b) side-view cross-sections of the original 5 μm TiO_2 GLAD nanostructured thin film deposited at α = 85^o. Side-view cross-section of the (c) once-filled and (d) twice-filled polymer-filled film.

and photoresist. Figure 5.5 shows the side of the polished GLAD film. Figure 5.6a shows a close-up of the FIB-sliced GLAD film.

In the final step prior to serial slicing, the sample was returned to the first orientation and ion-beam-assisted deposition was used to create a protective layer of carbon on the surface that was polished in the previous step. A phenanthrene precursor was introduced onto the sample surface using a gas injection system and dissociated by the 300 pA FIB probe and by locally generated secondary electrons, thereby producing a carbon layer of approximately 1 μm thickness. Figure 5.7 shows this protective carbon layer.

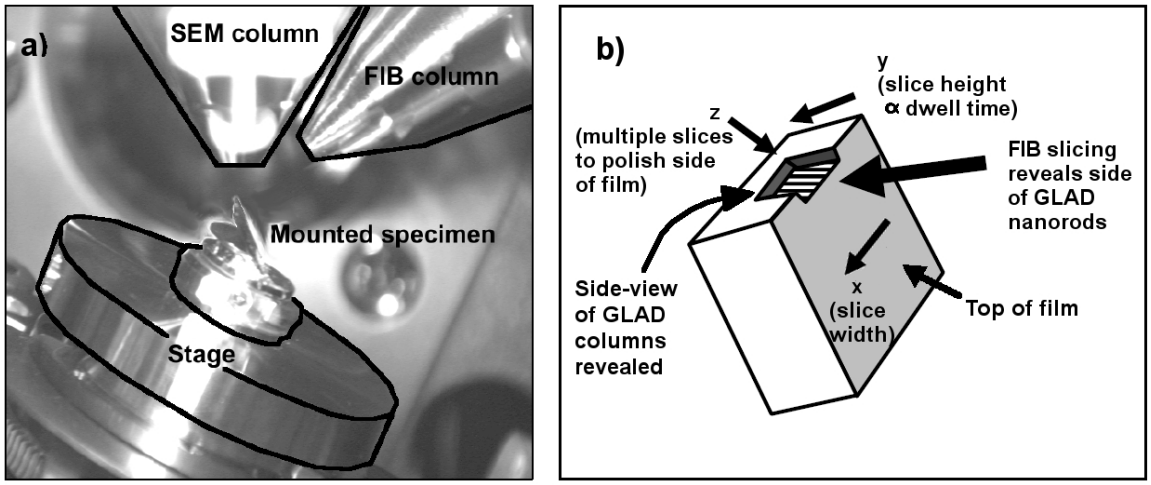


Figure 5.4: (a) Specimen orientation relative to FIB and SEM columns for the side-polishing procedure. (b) Side-polishing schematic.

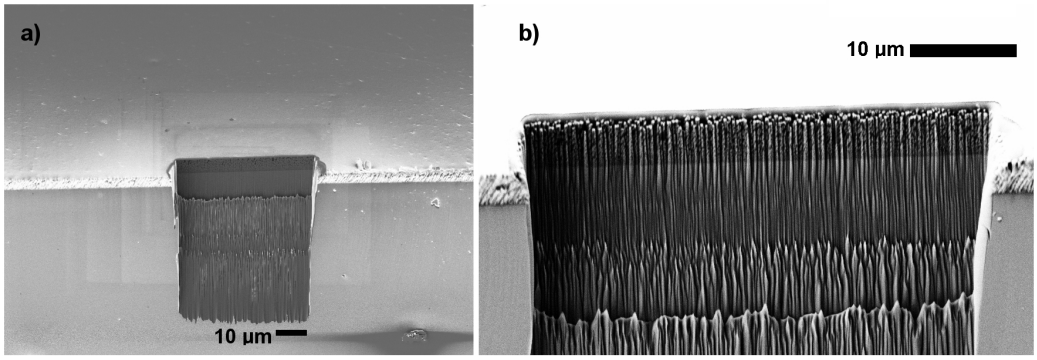


Figure 5.5: (a) Portion of sample is side-polished prior to sectioning. (b) Closer magnification reveals curtaining of silicon substrate that results as the FIB beam passes through the two-material GLAD film.

5.3.3 FIB sectioning

During the FIB tomography stage, serial slicing progressed from the photoresist capping layer through the nanostructured film and toward the silicon substrate, exposing surfaces parallel to the substrate plane, as shown in Figure 5.8a, b. After each slice was milled, an SEM image was acquired with the InLens detector and an electron beam energy of 3.0 keV. The SEM images were scanned at a resolution of 2048 pixels x 1536 pixels, with tilt-correction enabled in the NVision software to compensate for foreshortening such that the pixel sizes in the x and y directions were identical. The working distance to the initial slicing plane was kept constant throughout the milling. An image subdomain with typical dimensions of 1000 pixels x 1000 pixels was selected from the original im-

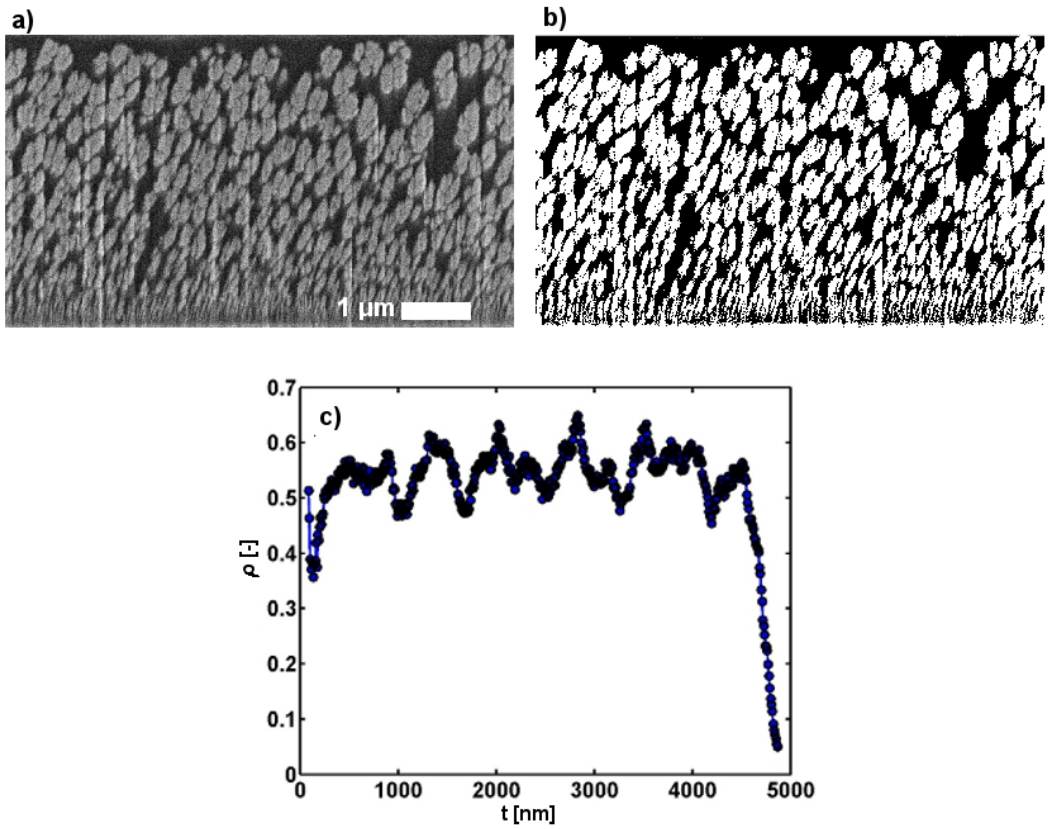


Figure 5.6: (a) Side view of FIB-polished SEM cross-section of 5 μm polymer-filled GLAD TiO_2 film. Columns are tilted due to the polymer filling process. (b) Image is thresholded to isolate GLAD columns from void spaces such that (c) density analysis of the image can be carried out by finding the ratio of white to black pixels along horizontal lines through the thresholded SEM image from the substrate-film interface (0 nm) to the top of the film (5000 nm).

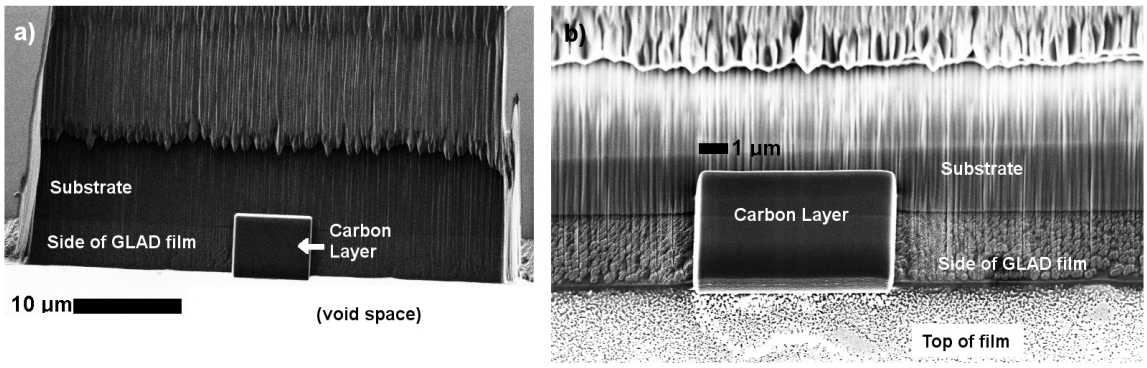


Figure 5.7: Protective carbon layer deposited on side of GLAD film prior to planar sectioning as seen from (a) FIB view and (b) SEM view.

age for tomographic reconstruction and morphological analysis. During the milling procedure, the FIB and SEM magnifications were locked and the FIB imaging/milling resolution set to 512 pixels x 384 pixels. Consequently, the volume elements of the reconstructed image series were not cubes

but rectangular elements with relative dimensions of 1 x 1 x 4 (width x length x height). A 300 pA milling probe was used for the first two slice series, and a 150 pA probe was used for the remaining high-resolution series. The probe spots contained 50% of the beam energy within diameters of 38 nm and 29 nm, respectively. The choice of probes ensured that the pixel fill factor during milling exceeded 100%, in accordance with standard milling practice.

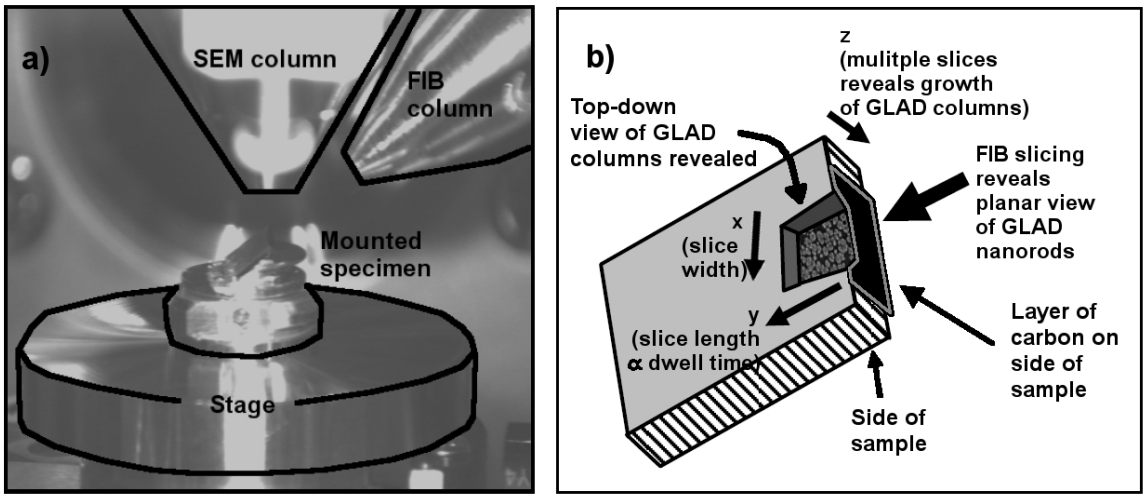


Figure 5.8: (a) Specimen orientation relative to the FIB and SEM columns for the tomographic slicing process. (b) Tomographic slicing proceeds from the top of the film towards the substrate, exposing a planar cross-section of the GLAD film.

The NVision GUI allowed for the automation of the milling procedure after appropriate focal, scanning, and detector settings had been selected. Because of the inherent scaling of the film (with column diameters growing larger as one progresses from the substrate-film interface toward the top of the film), the slicing was broken into several slice sequences, using progressively higher magnification, smaller fields of view and smaller milling volumes for each successive layer. The FIB was instructed to mill a trapezoidal volume for each slice series (i.e., a wider milling surface was used at the beginning of the given slice series than at its end in order to allow for redeposition of milled material to occur without impacting the milled volume side walls). This is illustrated in Figure 5.9. The multiple slice sequences also helped to manage stage, sample, and probe drift by realigning the sample at the beginning of each sequence. The typical slice sequence involved the execution of roughly 100 slice and image captures. With the exception of one slice series, all series were carried out at the same location on the sample. Due to an instrument error data for one slice series (Series 6), slicing was carried out at another location on the same film (with initial and final thicknesses taken relative to the film-substrate interface, as with the other slice series), with slightly different resolution parameters than were used for the initial slice series. While this did not allow for a single reconstruction of the 5 μm volume to be reconstructed, it did still allow for growth scaling parameters to be measured along the entire film thickness.

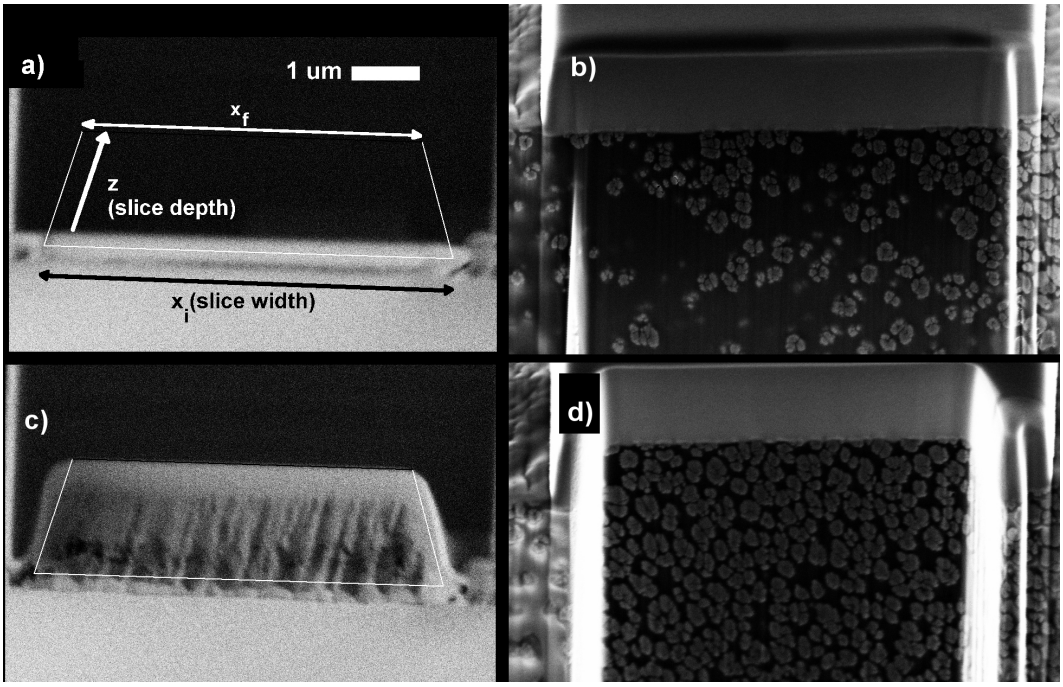


Figure 5.9: (a) In FIB view, establish milling volume before beginning of second slice series. (b) SEM view of milling volume before slicing. (c) FIB view of milling volume after slicing. (d) SEM view of exposed GLAD film cross-section after milling.

Table 5.1 shows the parameters for each of the slice series. SEM magnification settings were chosen such that the pixel size and slice thickness of the initial sequence were 4.0 nm and 16.0 nm, respectively. For the final sequence, the corresponding dimensions were 1.0 nm and 4.0 nm. The cross-sectional area was constant for all slices in a given series but decreased from 15.0 μm^2 for the first series to 0.14 μm^2 for the last. The total volume imaged was approximately 30 μm^3. Figure 5.10 shows the initial and final slice images of a representative slice series.

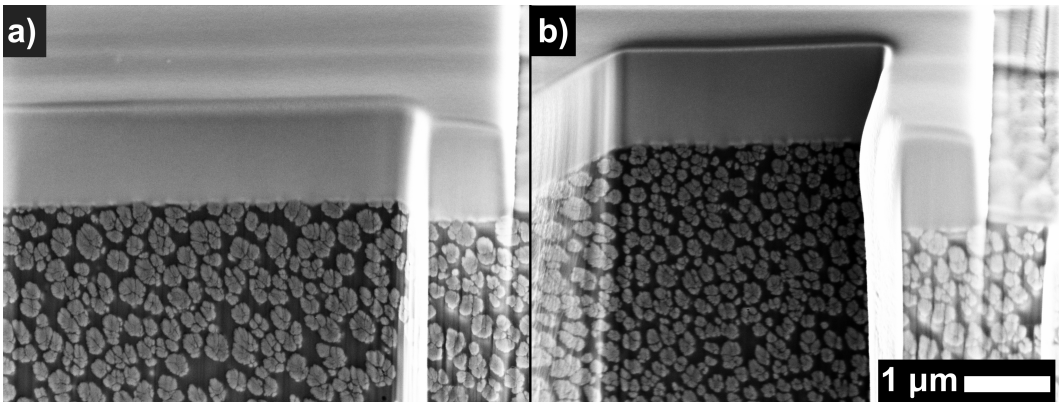


Figure 5.10: Representative cross-sectional slices shown for slice series 3 (a) before and (b) after the milling sequence has cut multiple slices from the sample volume.

Series	Slices	Thickness (nm)	Image resolution, R_{image} (nm pixel^{-1})	Slice resolution, R_{slice} (nm slice^{-1})	FIB beam current (pA)	EHT (keV)	(x,y) Fiducial shift (nm)	y-direction Advancement compensation (nm)
1	3	4928 to 4894	4	16	300	3	(-24, 0)	-50
2	93	4857 to 3390	4	16	300	3	(160, -336)	-1272
3	133	3364 to 1772	3	12	150	3	(-768, 102)	-720
4	71	1767 to 1215	2	8	150	3	(-1048, 272)	-40
5	40	1212 to 983	1.5	6	150	3	(-360, -188)	-353
6	45	903 to 177	4	16	150	1	(-128, 0)	-624
7	25	94 to 0	1	4	150	3	(-264, -42)	-150

Table 5.1: FIB-SEM slice series parameters.

5.4 FIB cross-section analysis

Once the complete set of the slice series was acquired, the images were postprocessed in preparation for 3D reconstruction and tomographic analysis. This section describes the alignment, stitching, equalization, and grain detection algorithms created for this processing. Full code for all algorithms described here is provided in Appendix C.

5.4.1 Alignment

Postprocessing of the milled slice images involved image registration and alignment to compensate for image drift. To accomplish this, a rectangular fiducial region, milled prior to serial slicing and common to all images within a given slice series, was used to align the images, as illustrated in Figure 5.11. In the algorithm provided below, $F_{old}(1..row_{Nf}, 1..col_{Nf})$ is the set of pixels representing the fiducial mark in the previous slice image, centered at ($row_{fiducial}$,$col_{fiducial}$). The current slice image is described by $I(1..row_N, 1..col_N)$. For the first slice image the fiducial location and size is defined by the user.

The drift of this fiducial frame was tracked through the image sequence using a routine, described in Algorithm 5.1 and in more detail in Appendix C, which minimized the sum of the squared differences in pixel values between the detected fiducial region in a given image and the fiducial region in the first image (ΔF_{min}). The new fiducial mark is given by $F_{new}(1..row_{Nf}, 1..col_{Nf}) = I(row_{m1}..row_{m2}, col_{m1}..col_{m2})$. To optimize the algorithm, a coarse pass was first completed to find the approximate location of the new fiducial mark (with row_{max}, col_{max} = 40 $pixels$ and row_{step}, col_{step} = 5 $pixels$). This was followed by a fine pass (with row_{max}, col_{max} = 5 $pixels$ and row_{step}, col_{step} = 1 $pixel$).

Algorithm 5.1 Tracking of fiducial frame from image slice to image slice

For each (row_o) from $-row_{max} : row_{step} : row_{max}$
  For each (col_o) from $-col_{max} : col_{step} : col_{max}$
    Shift the proposed location of the fiducial mark over by (row_o, col_o).
    Calculate the new upper (left) and lower (right) corners of the proposed fiducial mark:
      $row_1 = row_{fiducial} - row_{size}/2 + row_o$
      $row_2 = row_{fiducial} + row_{size}/2 + row_o$
      $col_1 = col_{fiducial} - col_{size}/2 + col_o$
      $col_2 = col_{fiducial} + col_{size}/2 + col_o$

    Isolate the proposed fiducial mark from the image:
      $F = I(row_1..row_2, col_1..col_2)$

    Calculate the difference in pixel values between the fiducial mark in the previous image and this current one:
      $\Delta F = \sqrt{\sum_{row=row_1}^{row_2} \sum_{col=col_1}^{col_2} [F(row, col) - F_{old}(row, col)]^2}$

    If the difference (ΔF) is less than ΔF_{min} then store the coordinates of the proposed fiducial mark:
      $row_{m1} = row_1$
      $row_{m2} = row_2$
      $col_{m1} = col_1$
      $col_{m2} = col_2$
    EndIf
  EndFor
EndFor

An additional translation of the frame (row_{adv}) in the positive y direction was required to compensate for the advancement of the milling surface into the sample. Here, the contrast step across the interface between the sample and the protective carbon layer was used to monitor the progress of the milling surface. Total shifts in the row and column directions were typically several hundred pixels, translating into several hundred nanometers to micrometers of spatial shift. These shifts are presented in Table 5.1. A plot of the shift in fiducial location (($\Delta\Phi$)) through the slices (S) of a single series is shown in Figure 5.12.

Once the drift of fiducial mark was calculated for each slice image, a sub-selection

$$(I_s(1..row_{Ns}, 1..col_{Ns}) = I(row_{s1}..row_{s2}, col_{s1}..col_{s2}))$$

of the overall image was extracted from row_{s1} to row_{s2} and col_{s1} to col_{s2}. This sub-selection included just the region that had been sliced by the FIB. The exact location of the sub-selected image tracked the drift in the fiducial mark. This image was used for all further processing, analysis of column growth parameters, and 3D reconstruction.

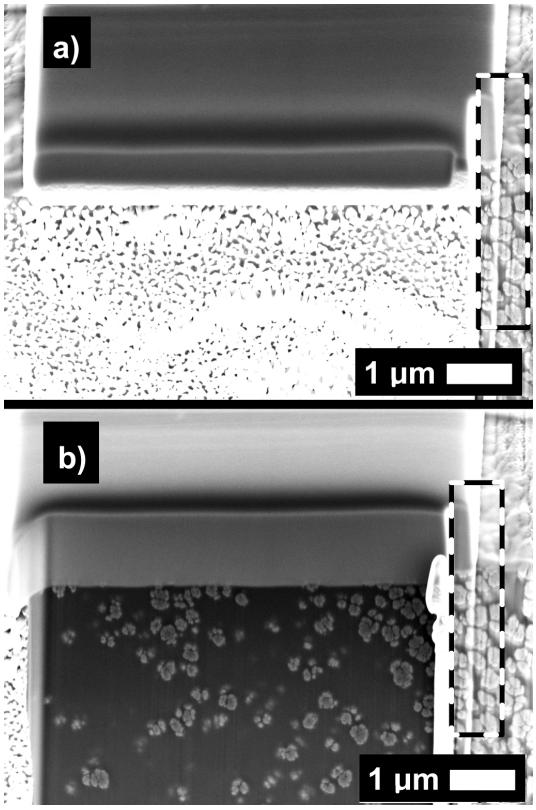


Figure 5.11: SEM images of the GLAD film (a) before and (b) after the execution of the first slice sequence. The dark region at the top of image (a) is the carbon protective layer on the side of the sample. The interface between this layer and the photoresist is clearly visible in frame (b). The bright region at the bottom of image (a) is the gold-coated top of the sample, removed by the FIB sectioning to reveal a top-down cross-sectional image of GLAD columns. The dashed outline in each figure indicates the location of the fiducial region used for image alignment.

5.4.2 Stitching

To reconstruct a larger volume of the GLAD film than that covered by just one slice series, several slice series layers had to be stitched together. To tackle differences in alignment from series to series, stitching was carried out by correlating similar features of the last image of a given slice series to those in the first image of the next series as illustrated in Figure 5.13. This stitching at the series to series boundary was done using a summed square pixel difference minimization technique which was essentially the same as that used for image registration (described above in Section 5.4.1). For all other images within the slice series image alignment was carried out by using the fiducial tracking technique described above. The stitching of consecutive series was carried out from the substrate toward the top of the film. This was done because the slices further into the film were captured at a higher magnification and a lower absolute area: this magnification and the selected region of interest dimensions at the bottom of the film had to be carried through for all images in the stitched series. A consequence of this is that 2D image interpolation had to be carried out for lower-magnification slices. This interpolation entailed using the MATLAB *imresize* function to

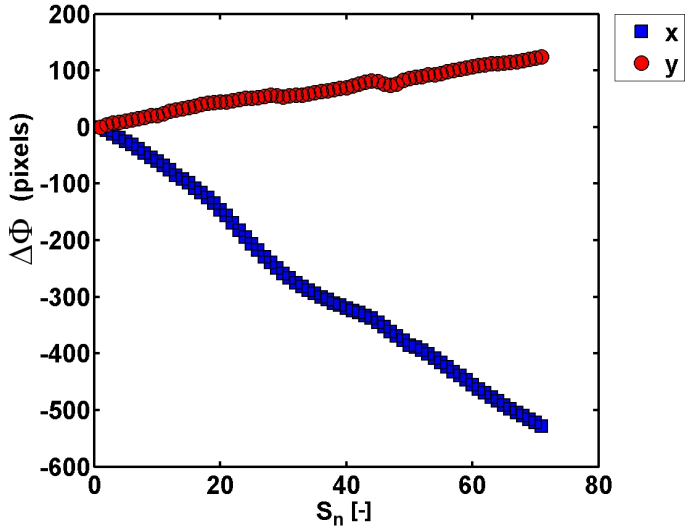


Figure 5.12: Representative progressive fiducial drift, measured relative to the initial fiducial image location, for slice series 4. Each data point represents one slice in the series.

interpolate all images from each of the slicing series to the same grid size and resolution. Once stitched together, a large 3D volume of a GLAD film, consisting of equal side grayscale voxels, could be reconstructed. This reconstruction was performed by stacking together the 2D images. Because the slicing resolution was different for each of the series, multiple copies of slices were stacked together to achieve uniform scaling within the reconstructed volume.

5.4.3 Image Post-processing

For each selected image region (I_s), local charging and other artifacts were compensated for through a custom equalization routine similar to adaptive histogram-based equalization algorithms implemented in MATLAB and by other groups [228]. To accomplish this, the selected image ($I_s(1..row_{Ns}, 1..col_{Ns})$) was broken into a grid of n_t x n_t tiles (typically n_t = 2 or 3). Each tile is described by $I_{srt,ct}$, where r_t and c_t denote the tile index along the row and column directions respectively:

$$
\begin{array}{lll}
I_{s1,1} = I_s(1..\frac{rol_{Ns}}{n_t}, & I_{s1,2} = I_s(1..\frac{rol_{Ns}}{n_t}, & I_{s1,3} = I_s(1..\frac{rol_{Ns}}{n_t}, \\
\quad 1..\frac{col_{Ns}}{n_t}) & \quad [\frac{col_{Ns}}{n_t}+1]..[2\frac{col_{Ns}}{n_t}]) & \quad [2\frac{col_{Ns}}{n_t}+1]..col_{Ns}) \\
I_{s2,1} = I_s([\frac{row_{Ns}}{n_t}+1]..[2\frac{row_{Ns}}{n_t}], & I_{s2,2} = I_s([\frac{row_{Ns}}{n_t}+1]..[2\frac{row_{Ns}}{n_t}], & I_{s2,3} = I_s([\frac{row_{Ns}}{n_t}+1]..[2\frac{row_{Ns}}{n_t}], \\
\quad 1..\frac{col_{Ns}}{n_t}) & \quad [\frac{col_{Ns}}{n_t}+1]..[2col_{Ns}/n_t]) & \quad [2\frac{col_{Ns}}{n_t}+1]..col_{Ns}) \\
I_{s3,1} = I_s([2\frac{row_{Ns}}{n_t}+1]..row_{Ns}, & I_{s3,2} = I_s([2\frac{row_{Ns}}{n_t}+1]..row_{Ns}, & I_{s3,3} = I_s([2\frac{row_{Ns}}{n_t}+1]..row_{Ns}, \\
\quad 1..\frac{col_{Ns}}{n_t}) & \quad [\frac{col_{Ns}}{n_t}+1]..[2\frac{col_{Ns}}{n_t}]) & \quad [2\frac{col_{Ns}}{n_t}+1]..col_{Ns})
\end{array}
$$

For each tile, the contrast equalization multiplier required to achieve the image-average black-white threshold value was then calculated, as shown in Figure 5.14 a, b. To calculate the black-white

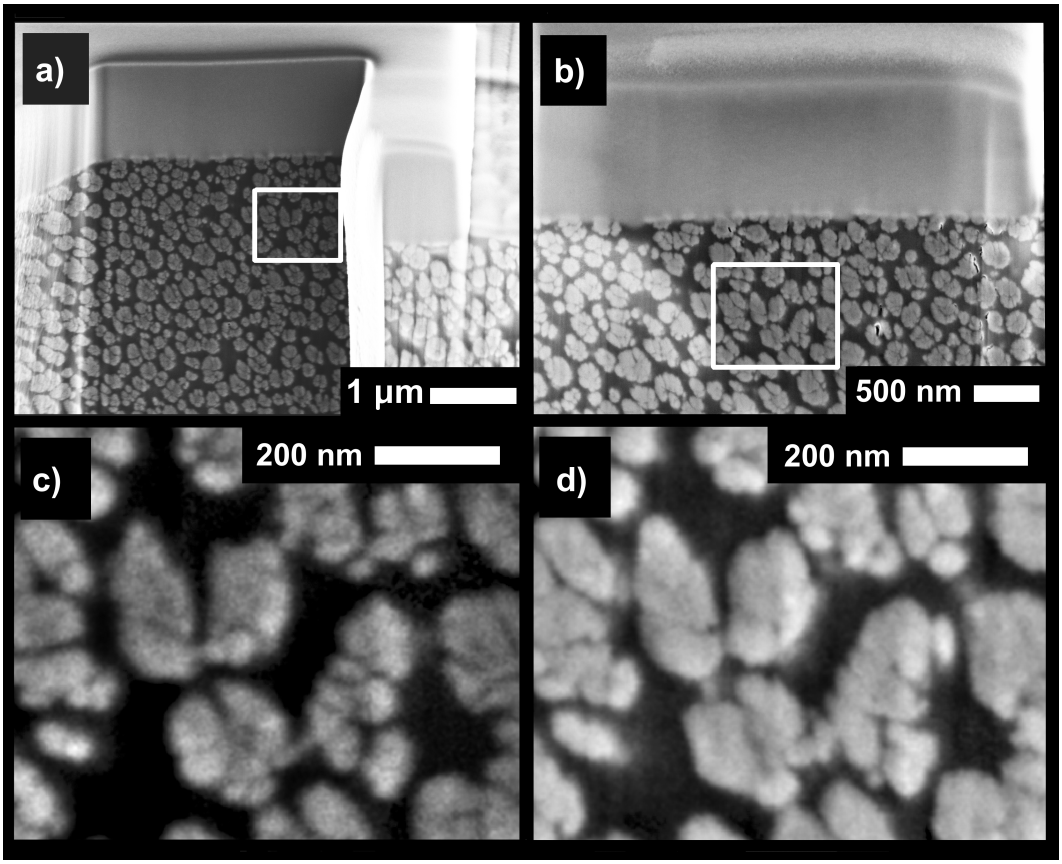


Figure 5.13: The tracked region (c) in the last slice of series 3 (a) is stitched to the corresponding region (d) in the first slice of series 4.

threshold of the entire grayscale image and the separate tiles, the Otsu method was applied [204]. This algorithm is implemented in the *graythresh* MATLAB routine. To find the optimum threshold for a grayscale image, Otsu's method casts the histogram of grey pixel values as a probability distribution (where the number and fraction of pixels at grey level $i = 1..N_{gray}$ is given by n_i and $p_i = n_i/N_{gray}$ respectively). Note that the number of grayscale values, N_{gray}, is 256 for these images. With thresholding, all gray pixels at a level below the threshold become black pixels while pixels with a level above the threshold become white pixels. The probability of pixels being black (i.e., below the threshold, Υ) is given by:

$$\omega_0 = \omega(\Upsilon) = \sum_{i=1}^{\Upsilon} p_i, \tag{5.2}$$

while the probability that pixels are white is given by:

$$\omega_1 = 1 - \omega(\Upsilon) = \sum_{i=\Upsilon+1}^{N_{gray}} p_i. \tag{5.3}$$

The means of the black and white levels (μ_0, μ_1) are related to the total mean level of the picture

(μ_{Total}) and the first order cumulative moment of the threshold ($\mu(\Upsilon)$):

$$\mu_0 = \frac{\mu(\Upsilon)}{\omega(\Upsilon)} = \sum_{i=1}^{\Upsilon} \frac{ip_i}{w_0}, \tag{5.4}$$

$$\mu_1 = \frac{\mu(\Upsilon)}{1-\omega(\Upsilon)} = \sum_{i=\Upsilon+1}^{N_{gray}} \frac{ip_i}{w_0}, \tag{5.5}$$

$$\mu(\Upsilon) = \sum_{i=1}^{\Upsilon} ip_i, \tag{5.6}$$

$$\mu_{Total} = \sum_{i=1}^{N_{gray}} ip_i. \tag{5.7}$$

The optimal threshold maximizes the separation of the black and white distributions (i.e., the distance between means) by maximizing the between-class variance (σ_B). This is done by calculating σ_B at each possible threshold (from $\Upsilon = 1..N_{gray}$):

$$\begin{aligned} \sigma_B^2 &= \omega_0\sigma_0^2 + \omega_1\sigma_1^2 & (5.8)\\ &= \omega_0\omega_1(\mu_1-\mu_0)^2 & (5.9)\\ &= \frac{\left[\mu_{Total}\omega(\Upsilon) - \mu(\Upsilon)\right]^2}{\omega(\Upsilon)\left[1-\omega(k)\right]}. & (5.10) \end{aligned}$$

The thresholds for each tile ($\Upsilon_{rt,ct}$) and the entire image (Υ_s) were therefore calculated using the above method. The multiplier required to bring the tile threshold to that of the entire image (a simple way to adjust the contrast of the tile) was then calculated:

$$\psi_{rt,ct} = \frac{\Upsilon_{rt,ct}}{\Upsilon_s}. \tag{5.11}$$

The multipliers were then assigned to each pixel in each corresponding tile ($\Psi_{rt,ct}(row, col) = \psi_{rt,ct}$ for $row = 1..row_s/n_t$, $col = 1..col_s/n_t$) and then compiled together into a multiplication matrix (Ψ) the size of the image.

This contrast equalization matrix was filtered in both dimensions with a moving average filter of length half the tile size ($f_c = row_s/(2n_t)$) to ensure no discontinuities across tile boundaries. Filter coefficients are therefore given by $a_1 = (f_c + 1)$, $a_2 = 0$, .. $a_j = 0$, .., $a_{fc} = 0$, $b_1 = 1$, $b_2 = 1$, .., $b_j = 1$, .., $b_{fc} = 1$. The MATLAB *filtfilt* function was used to apply this filter, first in the row direction and then in the column direction.

$$\Psi'(row, col) = -\sum_{j=1}^{f_c} a_j\Psi'(row-j, col) + \sum_{j=0}^{f_c} b_j\Psi(row-j, col). \tag{5.12}$$

$$\Psi''(row, col) = -\sum_{j=1}^{f_c} a_j\Psi''(row, col-j) + \sum_{j=0}^{f_c} b_j\Psi'(row, col-j). \tag{5.13}$$

The filtered matrix (Ψ'') was then applied to the original image, as shown in Figure 5.14 c and d to achieve the equalized image I_{eq}:

$$I_{eq}(row, col) = \Psi''(row, col) \cdot I_s(row, col), \qquad row = 1..row_{Ns}, col = 1..col_{Ns}. \tag{5.14}$$

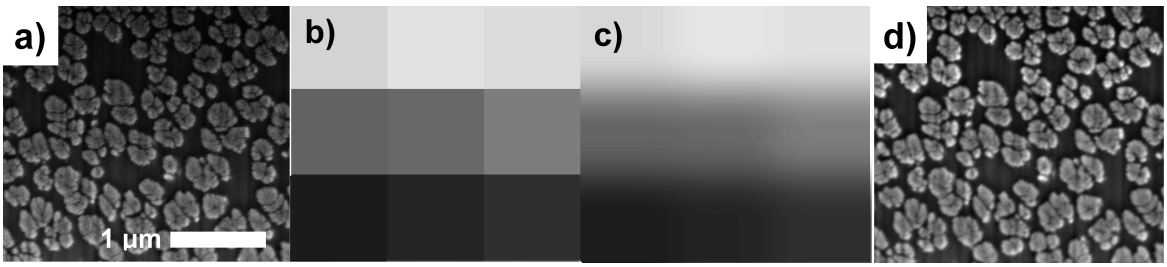


Figure 5.14: Image equalization is carried out in order to compensate for local charging and other artifacts in image (a) by breaking the image into tiles as shown in (b) and calculating the contrast equalization multiplier required to bring each tile to the average contrast. (c) Shows the filtered contrast equalization matrix and (d) shows the equalized image.

Once equalized, the image (such as the one shown in Figure 5.14d) was filtered in both dimensions with a moving average filter to remove noise. A filter length of f = 3 was used, with coefficients of $a_1 = 4$, $a_2 = 0$, $a_3 = 0$, $b_1 = 1$, $b_2 = 1$, $b_3 = 1$. The MATLAB *filtfilt* function was used to apply this filter, first in the row direction (to obtain I'_s) and then in the column direction to obtain the final equalized and filtered image (I_f) which was ready for analysis. Section 5.5.6 includes a sensitivity analysis of the GLAD columnar growth scaling properties to the choice of filtering coefficient.

$$I'_s(row, col) = -\sum_{j=1}^{f} a_j I_s^{eq}(row - j, col) + \sum_{j=0}^{f} b_j I_{eq}(row - j, col). \tag{5.15}$$

$$I_f(row, col) = -\sum_{j=1}^{f} a_j I_f(row, col - j) + \sum_{j=0}^{f} b_j I'_s(row, col - j). \tag{5.16}$$

5.4.4 Columnar properties and reconstruction

Representative cross sections of the GLAD columnar array are shown in Figure 5.15. Toward the substrate, the film is composed of a large number of small, closely spaced columns that are tens of nanometers wide. This is the region where initial nucleation of deposited material become the first GLAD columns. Some of these become extinct through competitive growth processes, and as we move upward toward the top of the film, the rods broaden and the average spacing between the column centers increases. After several micrometers of film deposition, the rod diameters reach several hundred nanometers. At the very top of the film, shown in Figure 5.15a, only the tips of the tallest rods can be seen when a cross-sectional FIB slice is taken.

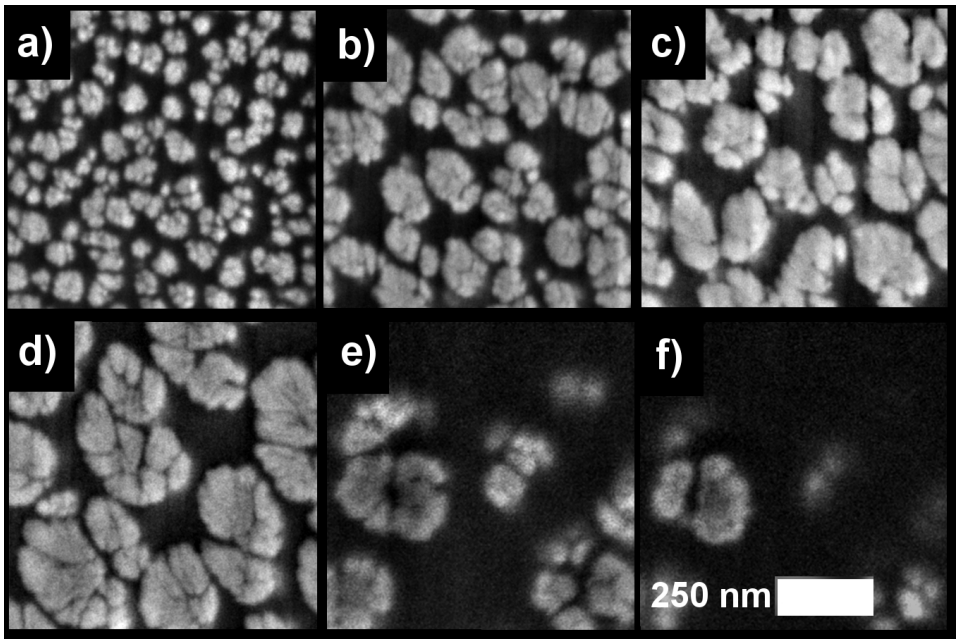


Figure 5.15: Top-down FIB cross sections of the 5 μm GLAD film at (a) 980 nm, (b) 1220 nm, (c) 3360 nm, (d) 4240 nm, (e) 4390 nm above the substrate. These cross sections illustrate the evolution of column diameter, spacing, and density.

From Figure 5.16 one can observe the first manifestation of the tops of GLAD columns emerging from the polymer matrix during the first few slices into the top of the film. Each image, separated by 16 nm, reveals the tops of additional emerging columns. What this figure illustrates is how difficult it is to determine the exact point at which a column has actually come to the exposed surface. The initial manifestation of the columns may in fact be due to electrons penetrating to a not insignificant depth into the matrix, thus imaging column features that have not yet surfaced. Using Equation 3.4 and Figure 3.16 in [22] the effective penetration and subsequent backscatter depth of electrons (e_{range}) can be estimated:

$$e_{range} = \frac{d \cdot 27.6 \cdot A \cdot Eo^{1.67}}{Z^{0.89} \cdot \delta}. \tag{5.17}$$

where d is the fraction of electrons that return to the surface and are detected (0.4), A is the atomic weight of the photoresist (taken to be composed primarily of ethyl lactate $C_5H_{10}O_3$, for a weight of 6.56 g mole^{-1}), Eo is the beam energy (3 keV), Z is the atomic number of the material (3.56), and δ is the density of the material (1.03 g cm^{-3}). This results in an estimation that the depth to which the BSEs interact with the photoresist matrix and return for detection is approximately 80 nm. However, because an InLens detector was used and because the beam energy is low, secondary electrons dominate the detected signal. Surface features and the atomic number contrast are therefore the principal characteristics imaged. However, while the interaction depth of secondary electrons in metal is only a few nanometers, it has been measured up to 50 nm in insulators [22]. This means

that the electron interaction with still-buried TiO_2 rods is still likely on the order of several tens of nanometers.

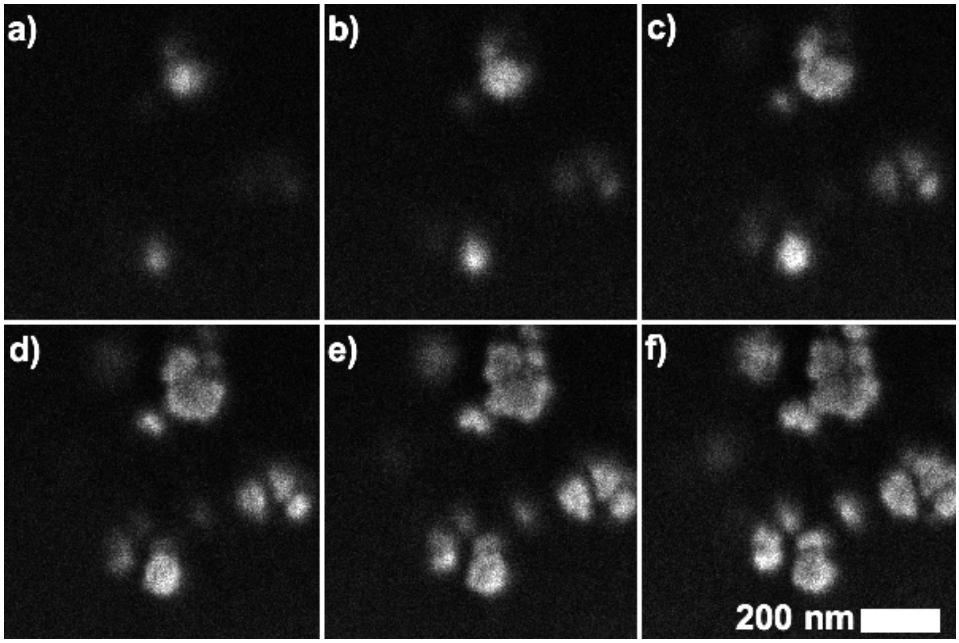


Figure 5.16: GLAD columns emerging from the polymer medium for slices spaced at 16 nm. Image (a) is at the top of the film, while image (f) is down 96 nm from the surface. Due to volume penetration of the electrons, the actual location of the tips of the columns is likely several tens of nanometers below the first signal detected.

The final milling stages are shown in Figure 5.17, where images are separated by 4 nm. The high density of nucleation sites is apparent in image a), which is on the order of ten nanometers from the substrate surface. These initial nuclei will compete to form the beginnings of the GLAD columns; some nuclei will be extinguished from shadowing by slightly larger neighbors while others will continue to grow. Also shown in Figure 5.17 is the transition from these nuclei to the substrate as the FIB milling progresses through the substrate. It is apparent that the FIB slicing made a slightly non-parallel cut with the substrate surface. It appears that this slant is approximately 10 nm over 1 μm (a slant of less than 1^o). Although fairly insignificant for most slicing performed on the film, this gradation matters when features towards the bottom of the film, whose characteristic dimensions are in the range of 10 nm, are imaged. In this case the column sizes sampled by a milling step will range from small towards the top of the image to larger towards the bottom, where the columns are further from the substrate. This effect is visible in Figure 5.17a. The impact of this milling slope on the analysis and reconstruction of volumes towards the bottom of the film therefore needed to be taken into account.

Another feature visible in Figure 5.17 is the curtaining visible at the top of each image (at the boundary between the GLAD film and carbon layer). This curtaining is due to imperfect polishing

of the side face of the film during the initial pre-carbon deposition preparation step. The curtain ripple is 40 nm peak-to-peak at the substrate to film interface. Due to similar differential slicing mechanisms as the FIB probes into the photoresist and TiO_2 heterogeneous solid, this suggests a thickness ripple in the exposed film surface on the order of several tens of nanometers.

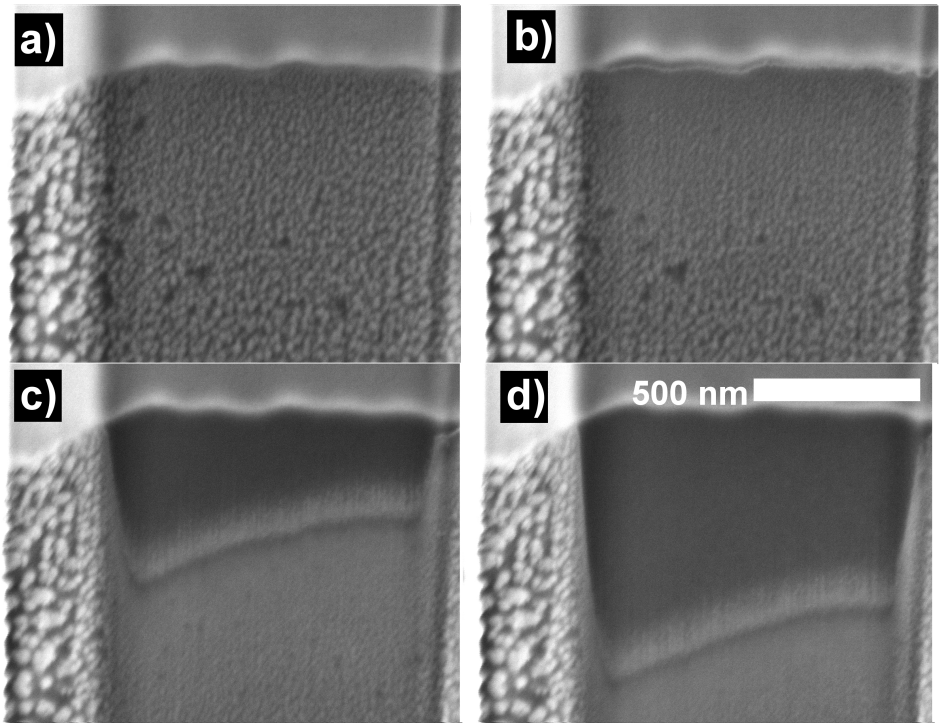


Figure 5.17: Final milling stages as substrate is reached. Slices are 4 nm apart. GLAD film is visible in entire milling region in images (a) and (b). At slices (c) and (d) the substrate is reached, but not uniformly.

Once images from consecutive slice series were stitched together into a single data cube, a reconstructed volume was created. This 3D data cube was imported into the JEOL TEMography Visualizer software (v 2.2, 2006) for imaging. This program allowed the volume to be viewed, rotated and magnified while parameters such as opacity and contrast were controlled. Representative 3D reconstructions of a GLAD columnar array are shown in Figure 5.18. These show a subset of the milled and imaged volume. Images a and b are different views of the same volume. They show that the features of individual columns are apparent, as well as the relationship between adjacent columns. Image c reveals details of a magnified region of this volume. Discrete slices are visible in this image, as are some details of the surface of the GLAD columns. The columns can be seen to be rough and made up of clumps of bifurcated arms, verifying previous reports from two-dimensional SEM data that GLAD films appear to bifurcate as they grow.

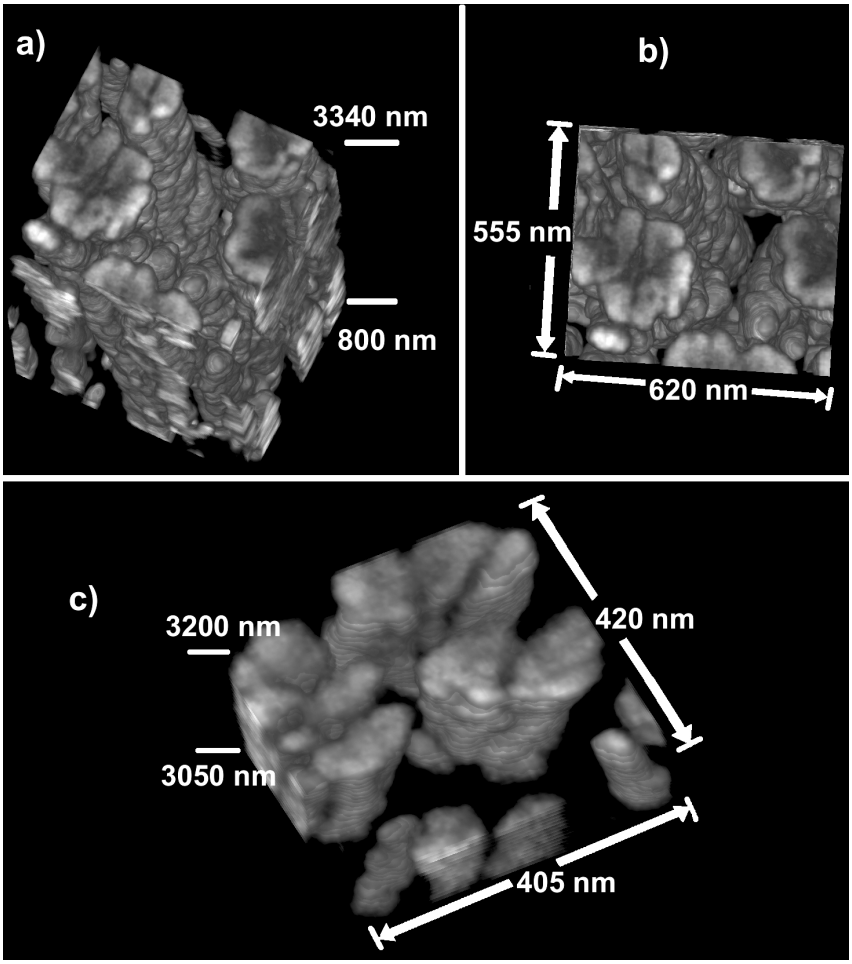


Figure 5.18: Three-dimensional tomographic views of a stitched GLAD film showing (a) a representative volume at an oblique angle such that columnar features are visible and (b) a view from the top of image a down through the film. (c) An extracted volume where surface details are visible on the individual columns.

5.5 Growth scaling

To assess the growth scaling property of the GLAD film being analyzed, the sectioned slice images were assessed as a function of the height of the film from the substrate. Column diameter, density, and spacing were all determined by employing custom image processing algorithms. These algorithms are described in detail in the following sections.

5.5.1 Thresholding

Prior to performing any operations requiring thresholding of the image (e.g., for column identification and quatitative analysis of slice properties) the thresholding target was adjusted such that images were effectively normalized to the same pixel density, as described in the following algorithm. This

density target (ρ_{target}) was the average of the pixel density profile described in Figure 5.6c. This density profile is quasi-stable as a function of film thickness, with the oscillations that are observed consistent with simulation work described in the literature [184]. These oscillations may be a consequence of self-shadowing competition between the nanocolumns; columns grow large until periodically a certain percentage die out at the expense of others. Toward the substrate, the density appears to decrease, although image resolution makes it difficult to determine the density of the film nucleation region precisely. Because the location above the substrate-film interface was not known precisely enough for each FIB slice, no attempt was made to match the oscillatory nature of the actual density profile.

The routine to adjust the thresholded pixel density of an image was iterative. The optimal black-white threshold (Υ) for the equalized and filtered image (I_f) was calculated using the MATLAB *graythresh* routine (Otsu's method). The density (ρ) of the image was calculated as described by Equation 5.18.

$$\rho = \frac{n_1}{n_1 + n_0}, \tag{5.18}$$

where n_0 and n_1 were the number of black and white pixels calculated from the image histogram (described by the distribution, p_i):

$$n_0 = (row_{Ns} \cdot col_{Ns}) \sum_{i=1}^{\Upsilon_c} p_i, \tag{5.19}$$

$$n_1 = (row_{Ns} \cdot col_{Ns}) - n_0. \tag{5.20}$$

The threshold level (Υ_c) was then stepped up or down (by m_s = 0.01) from the optimal value until the relative pixel density reached the target (ρ_{target}) within an error of $\epsilon_\rho = 0.002$. This threshold adjustment algorithm is described in Algorithm 5.2 and with full code provided in Appendix C. Section 5.5.6 includes a sensitivity analysis of the columnar growth properties to choice of this density target.

Once the required threshold level had been determined, the equalized and filtered image was thresholded to obtain a black and white image:

$$I_t(row, col) = white, \qquad if \;\; I_f(row, col) > \Upsilon_c \tag{5.21}$$

$$= black, \qquad if \;\; I_f(row, col) \leq \Upsilon_c \tag{5.22}$$

for all rows from 1..row_{Ns} and columns from 1..col_{Ns}.

Algorithm 5.2 Optimizing density of thresholded image

A) Calculate the difference between current thresholded image density and the target:

$\Delta\rho = \rho - \rho_{target}$

B) Adjust the threshold by a factor of M_Υ in order to reach the required density.
(M_Υ is initially set to 1)

C) While $|\Delta\rho| \leq \epsilon_\rho$

1) Step the new threshold multiplier up or down:
$M_\Upsilon = M_\Upsilon +$ sign $(\Delta\rho) \cdot m_s$

2) Calculate the new threshold:
$\Upsilon_c = \Upsilon_c \cdot M_\Upsilon$

3) Calculate the density of the thresholded image:
$n_0 = (row_{Ns} \cdot col_{Ns}) \sum_{i=1}^{\Upsilon_c} p_i$
$n_1 = (row_{Ns} \cdot col_{Ns}) - n_0$
$\rho = \frac{n_1}{n_1+n_0}$

4) Calculate the difference between current thresholded image density and the target:
$\Delta\rho = \rho - \rho_{target}$

EndWhile

5.5.2 Column identification

A feature of top-down SEM images of GLAD films is that columns often appear to clump together. By eye these columns can be separated into individual columns, but with automated computer algorithms the task is more difficult. Grain-detection algorithms, which identify individual columns, have been used by other groups to analyze the top-down SEM images of obliquely deposited films [63], [205]. For this present work, a custom grain-detection algorithm was also implemented in order to analyze each of the equalized and thresholded cross-sectional images acquired during the FIB slicing process. A sample equalized and thresholded cross-sectional slice is shown in Figure 5.19 a,b. Identification of individual columns and connected regions of columns was done using the MATLAB *bwlabel* routine, which labels connected pixels as single objects. The individual columns and connected regions of a representative cross-sectional slice of the GLAD film are shown in Figure 5.19c.

Connected regions (S_i, i = 1 .. N_S) typically consisted of two to ten columns. To break these connected objects, such as those shown in Figure 5.20, into individual columns the algorithm described in Algorithm 5.3 was used (with code provided in Appendix C). This algorithm seeks to find bridges in the connected domains that join one column to another. While general, it requires several variables that must be set individually for each slice series. Despite filtering to remove noise and equalization routines to adjust contrast, differences in factors such as resolution, sharpness, noise, and brightness for each series required the fine-tuning of these variables. Setting of the variable values was done by visually inspecting the results obtained by running the algorithm on the first

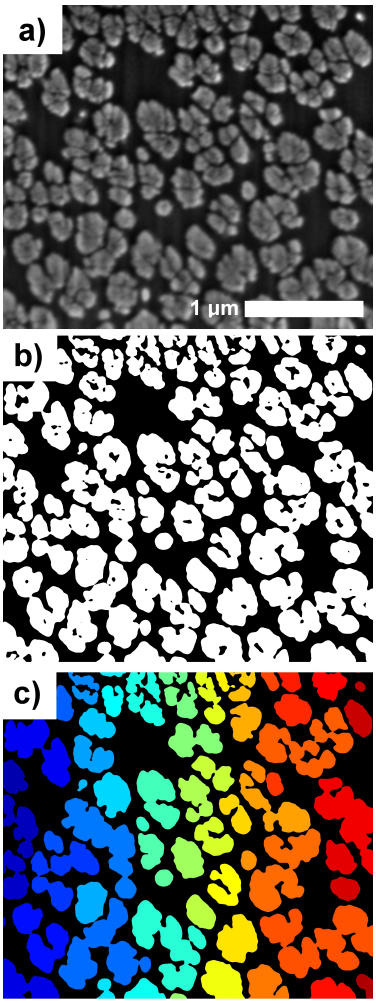


Figure 5.19: (a) Equalized top-down cross section of an FIB-sliced GLAD film in the second series (4240 nm from substrate), (b) thresholded image, (c) connected domains of clumped columns identified by unique color.

and last slice image of each series. Each parameter was then adjusted until the columns had been individually identified in a manner similar to that which would have occurred had one manually identified each column in the slice images. The series-specific values for each variable described in the following algorithm are provided in Table 5.2. Section 5.5.6 includes a sensitivity analysis of the columnar growth properties to the values of the parameters described in this algorithm.

Once all columns had been identified in a given slice, the diameters of each column (w_k, k = 1 .. $N_{columns}$) were calculated. The diameter was calculated by assuming a circular cross-sectional area for each column. A circular cross-section was assumed for the GLAD columns to simplify analysis. Improvements to future image processing schemes could include use of a more appropriate (e.g. elliptical) geometry. This columnar area was in turn calculated from the total number of pixels in an identified column ($Co_k.pixels$) times the image resolutions (R_{image}).

Algorithm 5.3 Identification of individual columns within an SEM image by breaking connected domains

I) Calculate the mean of all domain areas in the given slice:

$$\overline{A} = \frac{1}{N_S} \sum_{i=1}^{N_S} area(S_i)$$

II) For each of the identified connected domains S_i, i = 1 .. N_S

If the connected domain has an area greater than $\overline{A} \cdot f_{area}$ then

A) Determine the inner and outer boundaries (B_n, n = 1 .. $N_{boundaries}$) along the outlines of the domain using the MATLAB *bwboundaries* routine.

B) Starting with the outer boundary (as shown in Figure 5.21 a)

Boundary 1 (B_1) is in the clockwise direction,

Boundary 2 (B_2) is in the counterclockwise direction.

1) Compute the distance matrix ($dist$) between the points along B_1 and B_2:

($b1 = 1..length(B_1)$, $b2 = 1..length(B_2)$)

A sample distance matrix is shown in Figure 5.22.

$$dist(b1, b2) = \sqrt{[B_1(b1).row - B_2(b2).row]^2 + [B_1(b1).col - B_2(b2).col]^2}$$

2) Find the local minima of the distance matrix using a MATLAB opensource routine *extrema*. These identify the possible bridge point pairs ($p_{bridge}(k), k = 1..N_{pairs}$) between connected columns in the domain. This is illustrated in Figure 5.21 b.

3) Reject all point pairs that are adjacent to each other.

4) Reject all point pairs that are greater than $f_{minBridge}$ pixels apart.

5) A virtual line is drawn between the two points of a possible bridge. Point pairs with a void space between them are rejected, as illustrated in Figure 5.21 c.

6) Identify the points along the boundaries where there is a large inflection.

a) The normal vector to each boundary (n_1, n_2) is calculated.

$$v_1(b1) = \arctan(\left| \frac{[B1(b1+1).col - B1(b1).col]}{[B1(b1+1).row - B1(b1).row]} \right|)$$

$$n_1(b1) = v_1(b1) + 90 \cdot quadrant[v_1(b1)]$$

b) The change in the normal vector (Δn_1, Δn_2) is calculated.

$$\Delta n_1(b1) = n_1(b1 + 1) - n_1(b1)$$

c) Find local minima of the normal vector changes (MATLAB *extrema* routine).

d) Determine if these local minima correspond to any possible bridge point pairs (within f_{infl} pixels).

e) Reject all point pairs without local minima associated with them.

f) For each point pair ($p_{bridge}(k), k = 1..N_{pairs}$), if inflection (i.e., the change in normal vector is too small ($\leq f_{peak}$ [deg pixel^{-1}])) then reject this pair as illustrated in Figure 5.21 d.

7) Merge any point pairs that are near to each other (within $f_{minPair}$ pixels) into a single point pair.

8) Draw a virtual cut line between all remaining point pairs, as shown in Figure 5.21 e.

C) Repeat the above algorithm with all inner and outer boundary combinations to determine appropriate cut lines between inner and outer boundaries of the connected domain.

D) Break the domain along each of the cut lines using the MATLAB *bwlabel* routine.

E) Store the columnar cross-sections from the broken connected domain ($Co_{i,j}$, j = 1 .. $N_{columns_i}$).

Else

Store the domain as a columnar cross-section ($Co_{i,1}$)

EndIf

For all columnar cross-sections, $Co_{i,j}$, i = 1 .. N_S, j = 1 .. $N_{columns_i}$

Calculate metrics such as diameter, area, and center location

EndFor

EndFor

Series	f_{area} (pixels)	$f_{minBridge}$ (pixels)	f_{peak} (deg pixel^{-1})	f_{infl} (pixels)	$f_{minPair}$ (pixels)
1	1.3	40	5	10	20
2	0.8	55	15	10	20
3	1.0	40	15	10	20
4	1.0	30	15	15	20
5	1.0	30	15	15	20
6	0.8	15	5	10	5
7	0.8	45	5	10	20

Table 5.2: Variable values for connected domain breaking algorithm. Connected domain areas must be larger than $f_{area} \cdot \overline{A}$. Pairs must closer than $f_{minBridge}$ pixels to possibly be spanning a bridge. Normal vector inflection values must be larger than f_{peak} deg pixel^{-1} to be possible indicators of a bridge. Pairs must be within f_{infl} pixels of large normal vector inflections to be considered. Pair within $f_{minPair}$ pixels of another pair are considered the same.

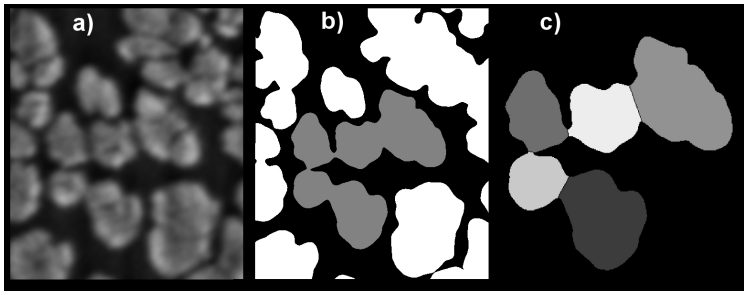


Figure 5.20: Connected domains, as illustrated in (b), are identified within each image (a) and broken into individual columns, as shown in (c).

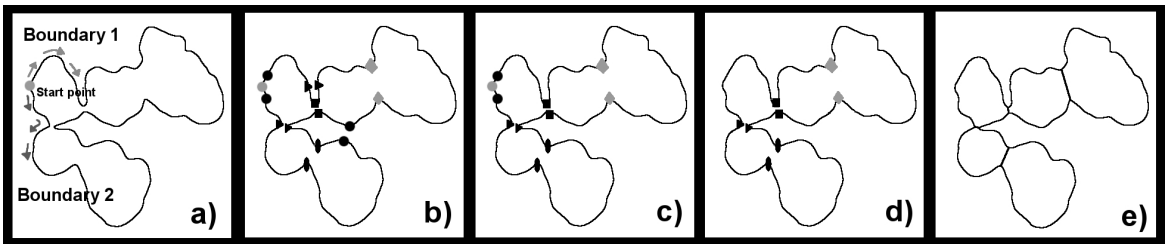


Figure 5.21: Connected domain is broken into individual columns through grain detection algorithm. (a) Boundary of connected domain is identified, with boundary1 and boundary2 identifying travel along the shape edge in clockwise and counterclockwise directions. (b) Minimum distances between boundary1 and boundary2 point pairs, with only those on opposite edges of the domain kept in (c). Finally, in (d) only points with large concavity are used to break the connected domain up into (e) individual columns.

$$w_k = \frac{\sqrt{\sum Co_k.pixels \cdot R_{image}}}{\pi}. \tag{5.23}$$

At each slice, the mean cross-sectional diameter was calculated:

$$\overline{w} = \frac{\sum_{k=1}^{N_{columns}} w_k}{N_{columns}}. \tag{5.24}$$

As shown in Figure 5.23, the mean of the diameters increased with height from the substrate-film

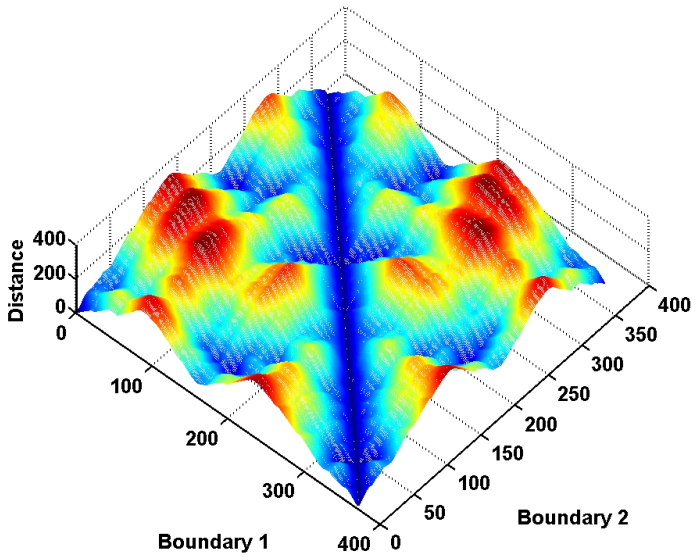


Figure 5.22: Local minima in the matrix of distances between points along the boundary of a shape help to identify bridges between connected columns.

interface towards the top of the film.

The column density per slice image (d_{column}) was the calculated by counting the number of columns per cross-sectional image area (given by the number of pixels in the image $N_{row}N_{col}$ multiplied by the resolution per square pixel).

$$d_{column} = \frac{N_{columns}}{N_{row}N_{col} \cdot {R_{image}}^2}. \tag{5.25}$$

5.5.3 Nearest neighbor identification

Once the individual columns had been identified within each image, they were represented by a column center point (row_k, col_k) and a circle of diameter w_k, as determined in Algorithm 5.3. A representation of these columns in shown in Figure 5.24. One can see that because the columns are represented as circles some columns appear to overlap. This does not impede the detection of nearest neighbors, an algorithm now discussed.

The nearest neighbors of each column were identified along trajectories at specific increments. For each slice, the distance from the center of a given column to the center of the nearest neighbor column (c) was calculated. Columns that bordered the edge of the image were excluded from this analysis. Algorithm 5.4 provides the details of this routine (with code provided in Appendix C).

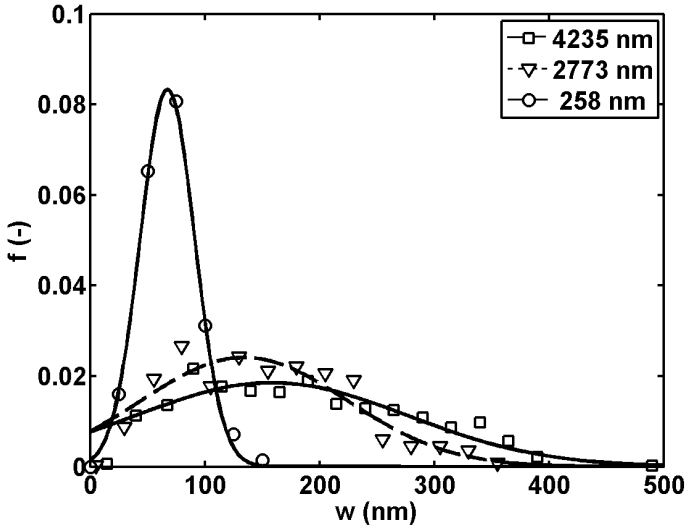


Figure 5.23: Distribution of GLAD column diameters for three selected slices, at 4235 nm, 2773 nm, and 258 nm from the substrate-film interface. Fit lines represent Gaussian curve fits

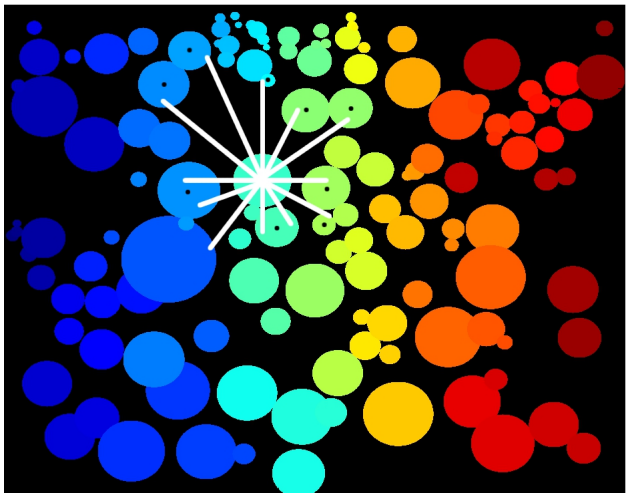


Figure 5.24: Nearest neighbor columns are found along trajectories at 30 ^{o} increments.

The mean center to center column spacing was then calculated at each slice:

$$\overline{c} = \frac{\sum_{k=1}^{N_{columns}} \left[\frac{\sum_{\tau=1}^{N_{\tau k}} c_{k,\tau}}{N_{\tau k}} \right]}{N_{columns}}. \tag{5.26}$$

Due to the nature of this computationally intensive algorithm, trajectories at 30^o increments were traversed, as illustrated in Figure 5.24. While this algorithm will not find all nearest neighbors, on average it will locate most neighbors and will calculate the mean nearest neighbor spacing within several percent of that calculated when a finer set of trajectory angles is used. Figure 5.26 shows the percent difference (from the values calculated at trajectories of 5^o) of the calculated mean center to

Algorithm 5.4 Identification of nearest columnar neighbors

Represent each column (Co_k) by a circle of diameter w_k, centered at (row_k,col_k) (as illustrated in Figure 5.25a).

For each column (Co_k)

For each trajectory τ from 0 ^{o} to 330 ^{o}, in 30 ^{o} increments

1) Draw a line from the center of the column along the given trajectory, as illustrated in Figure 5.25b. (This figure shows the lines from the centers of all columns along the 0 ^{o} trajectory).

2) Find the intersection of the line with each other column along that line, as illustrated in Figure 5.25c.

3) Identify the first column along the line ($Co_{k,\tau}$), as illustrated in Figure 5.25d.

4) If this neighboring column has previously been identified along another trajectory then disregard it. Otherwise calculate the distance between column centers:

$$c_{k,\tau} = \sqrt{(row_k - row_\tau)^2 + (col_k - col_\tau)^2}$$

EndFor

EndFor

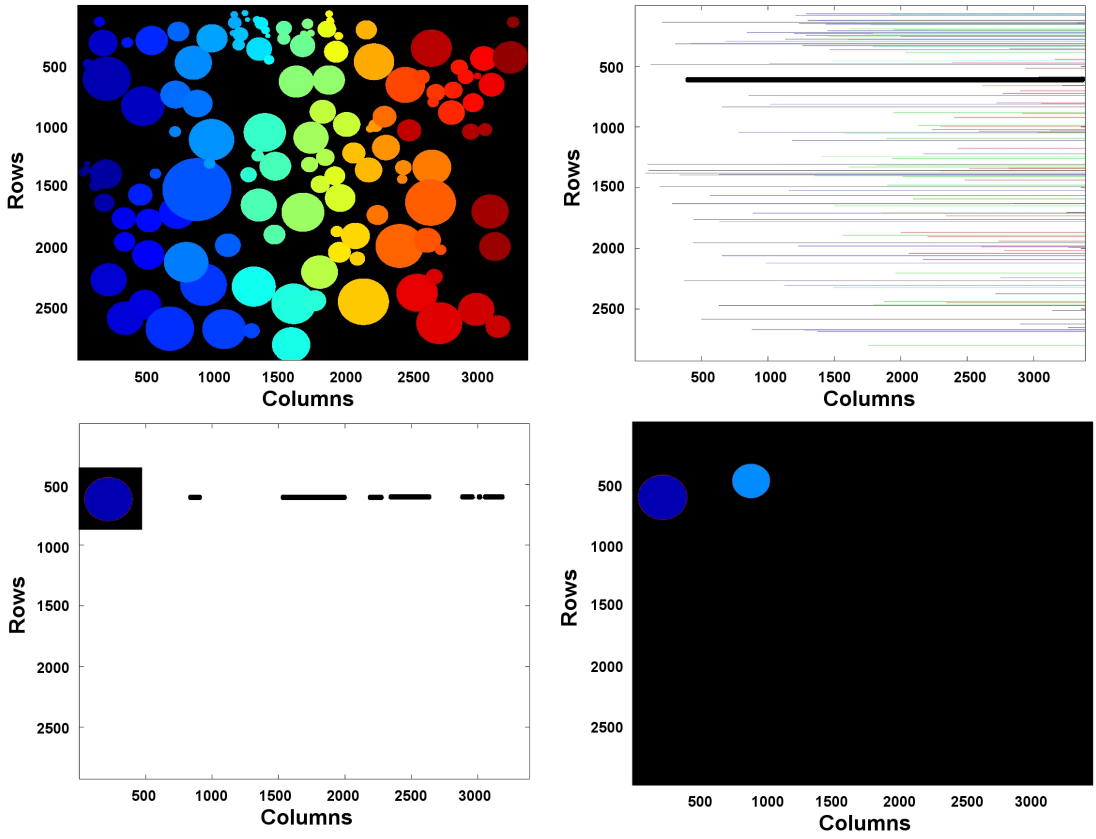


Figure 5.25: Nearest neighbor columns are identified in a representative slice. (a) Columns are represented by circles centered at the locations of the actual columns. (b) Lines are then drawn from each column center. (c) The intersection of each line with all columns along that line are noted. (d) The nearest column along the given trajectory is found.

center column spacings, for three selected FIB slices, for trajectories from 5^o to 90^o. Results for the 30^o trajectories are within 1% of those calculated at a trajectory of 5^o.

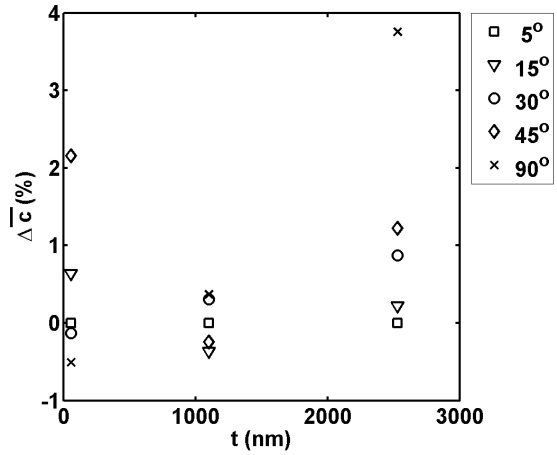


Figure 5.26: Percent difference in mean center to center column spacing (from the values calculated at trajectories of 5^o) for trajectories from 5^o to 90^o at selected film thickness (i.e., FIB milled slices).

5.5.4 Spacing between columns

To determine the average spacing between adjacent columns ($\overline{v}$), a metric closely related to the column center to center spacing, the distance between adjacent column edges (i.e., the length of void space between the columns) was found for each slice image. The equalized and thresholded images, such as those shown in Figure 5.27a and b were used for this analysis. Algorithm 5.5 describes the routine used to calculate this metric (see Appendix C for full code).

The average void spacing between columns was calculated at each slice, as shown in Equation 5.27. This finds not only the mean void distance between nearest neighbor columns, but also includes some large gaps between columns. Any gaps within columns or clumped columns are also included in the calculation, such that all void spaces are tracked.

$$\overline{v} = \frac{sum_{void}}{N_{void}}. \tag{5.27}$$

5.5.5 Growth scaling

The growth scaling parameters of mean column diameter, void spacing, center-to-center spacing, and density are shown in Figure 5.28. The different symbols shown in these plots represent the separate slice series. Scatter within each slice series and between slice series is due to the sensitivity

Algorithm 5.5 Determination of void spacing between columns

A) Find the edges of columnar structures
(i.e., the transitions from column to void (white (1) to black (0)))
as shown in Figure 5.27c.
For each row
For each col
$t_r(row, col) = I_t(row, col) - I_t(row, col - 1)$
If $t_r(row, col) == -1$ then
The transition is from column to void
Elseif $t_r(row, col) == 1$ then
The transition is from void to column
EndIf
EndFor
EndFor
For each col
For each row
$t_r(row, col) = I_t(row, col) - I_t(row - 1, col)$
If $t_r(row, col) == -1$ then
The transition is from column to void
Elseif $t_r(row, col) == 1$ then
The transition is from void to column
EndIf
EndFor
EndFor

B) Determine distances between columns
(i.e., length of void spaces)
sum_{void} = 0
N_{void} = 0
For each row
Identify all black to white transitions (btw_i)
Identify all white to black transitions (wtb_i)
For each transition
$l_{void} = btw_i.col - wtb_i.col$
$sum_{void} = sum_{void} + l_{void}$
EndFor
EndFor
For each col
Identify all black to white transitions (btw_i)
Identify all white to black transitions (wtb_i)
For each transition
$l_{void} = btw_i.row - wtb_i.row$
$sum_{void} = sum_{void} + l_{void}$
$N_{void} = N_{void} + 1$
EndFor
EndFor

of the column (grain) detection routines to image contrast nonuniformity, resolution, nearness of adjacent columns, and bifurcation characteristics of individual columns. The sensitivity of the predicted growth scaling characteristics to the grain detection routine parameters, and thus estimates of uncertainty in these growth scaling exponents, is explored below in Section 5.5.6.

Figure 5.28a illustrates quantitatively the column diameter growth scaling from tens to hundreds of nanometers over a thickness of several micrometers. This column diameter curve shows the

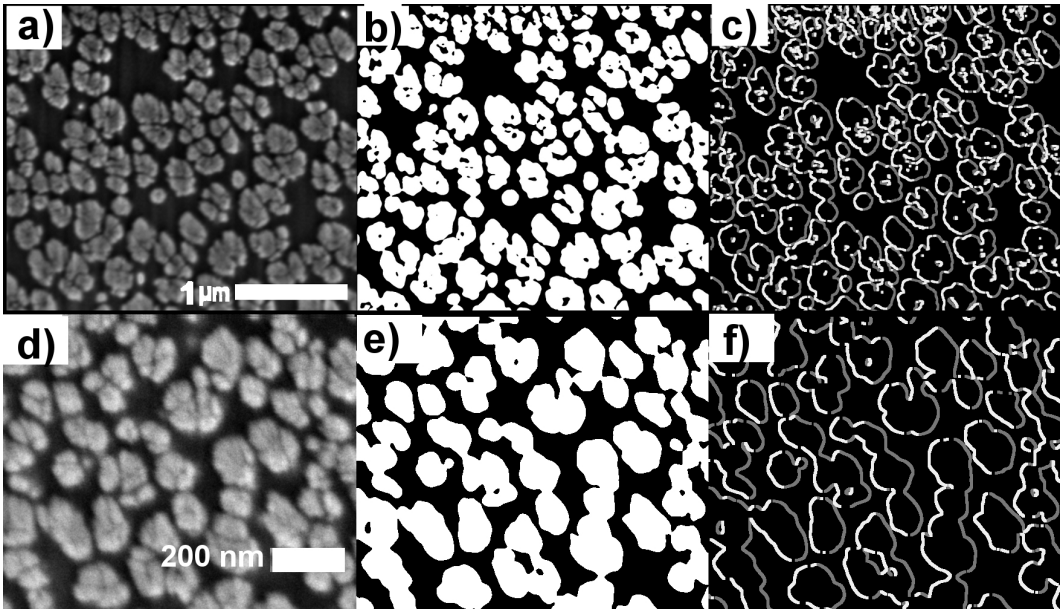


Figure 5.27: Void spacing between columns in (a) and (d) the equalized image is estimated by (b) and (e) thresholding and then (c) and (f) finding the distances between column to void transitions (grey lines) and void to column transition (white lines) in the row and column directions. Shown here are examples for milled slices at (a) through (c) 4240 nm and (d) through (f) 1500 nm from the substrate.

characteristic self-similar scaling seen for GLAD films. An abrupt dip in the mean diameter curve occurs at the maximum film thickness because only the very tips of the tallest columns are imaged at this point. A power law fit to the mean diameter curve reveals a relationship of $\overline{w}$ = 9.5 $t^{0.35}$ [nm], with t being valid from approximately 100 nm to 5000 nm. (Data from the very bottom and top of the film were disregarded.) Compensation for the $\theta = 10^o$ tilt in the filled film results in a stretching of the thickness axis from a maximum thickness of 5000 nm to 5080 nm and a reduction in mean width by $(\cos\theta)^{1/2}$ to yield a relationship between the actual diameter ($\overline{w}_{act}$) and thickness of $\overline{w}_{act}$ = 9.4 $t^{0.35}$ [nm]. The lines above and below the data points represent a fit to one standard deviation from the mean of the column diameter distributions (i.e., the standard deviations were calculated from each Gaussian distribution, such as the ones shown in Figure 5.23, and fit to a power-law scaling curve). The width of the distributions is of the same order as that of the mean diameter, indicating a wide range of diameters at every stage of column growth.

The exponential factor of 0.35 describing the growth scaling of the column diameter is in line with the findings of previous groups for metal oxide films, where P values ranging from 0.3 to 0.5 have been found for metal oxide GLAD films deposited at an oblique angle of 85^o [209]. It has also been speculated that column growth, which is dominated by shadowing, should result in a P value of 0.5 whereas large adatom diffusion should result in a scaling factor of 0.3 [229]. Our scaling factor of 0.35 therefore suggests that adatom diffusion is a significant factor in the growth of TiO_2

films under the deposition conditions described in Section 5.3.1.

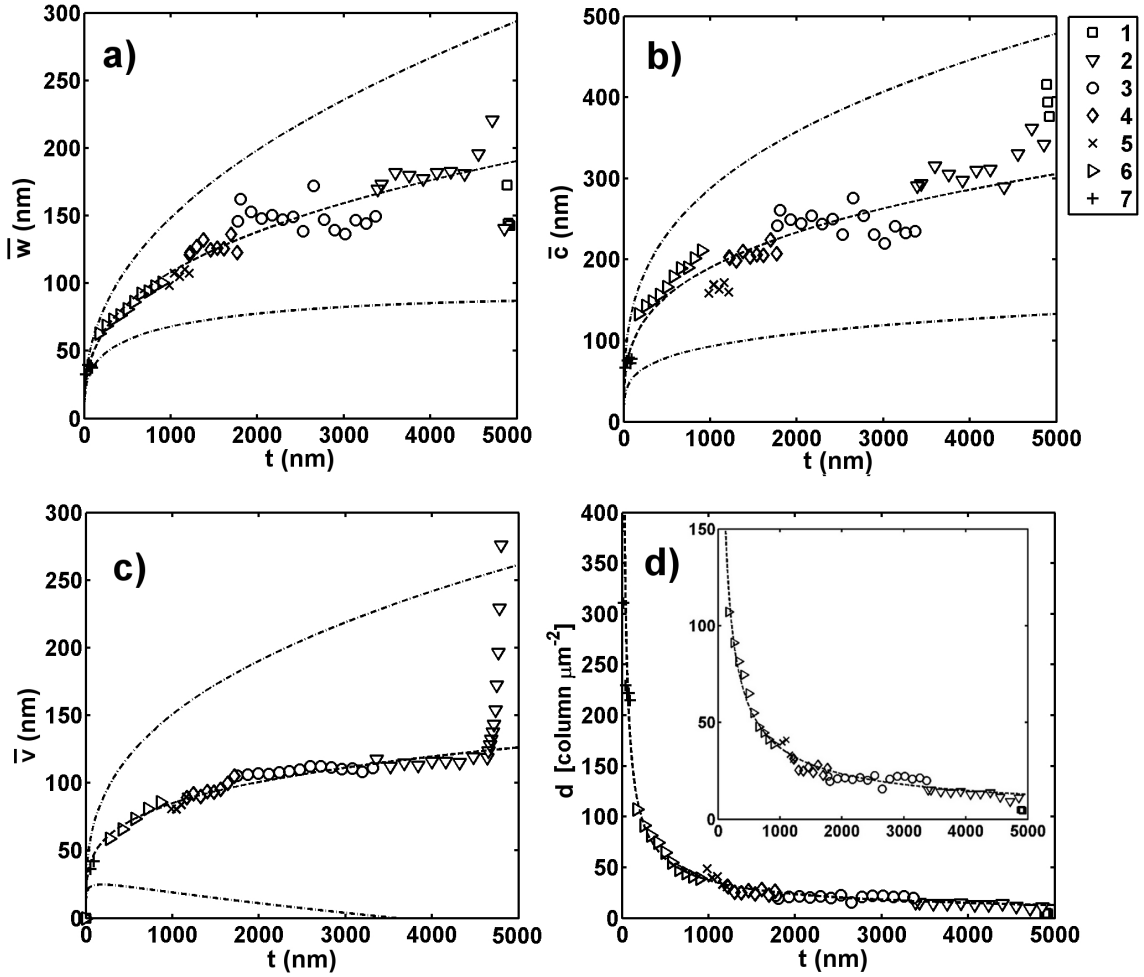


Figure 5.28: (a) Mean column diameter scales as $\overline{w}$ = 9.5 $t^{0.35}$ [nm] ($\overline{w}_{act}$ =9.4 $t^{0.35}$ when the column tilt is compensated for), where t is in nanometers. (b) Column center-to-center spacing between columns scales as $\overline{c}$ =24.8 $t^{0.30}$ [nm] ($\overline{c}_{act}$ =24.5 $t^{0.30}$). (c) Void spacing scales as $\overline{v}$ = 15.4 $t^{0.25}$ [nm] ($\overline{v}_{act}$ =15.2 $t^{0.25}$). (d) Column density scales with thickness d= 3.4x10^3 $t^{-0.65}$ [column μm^{-2}]. The inset shows a magnified region of the column density plot. The dotted line through the data points shows the scaling fit. Dashed-dotted lines show the fit to one standard deviation of parameter distributions.

The void spacing between columns also follows a self-similar growth pattern. Figure 5.28c reveals a power-law scaling trend of mean void spacing versus a thickness of $\overline{v}$ =15.4 $t^{0.25}$ [nm]. The tail at 5000 nm is due to the large gaps between the very tips of the tallest columns at the top of the film. Compensation for film tilt results in a void spacing relationship of $\overline{v}_{act}$ = 15.2 $t^{0.25}$ nm. Similarly, the columnar nearest-neighbor center-to-center spacing grows with a power-law relationship of $\overline{c}$ = 24.8 $t^{0.30}$ nm. This curve is shown in Figure 5.28b. By compensating for the tilt in the film, the spacing relationship becomes $\overline{c}_{act}$ = 24.5 $t^{0.30}$ nm. Interestingly, it can be seen that the center-to-center and

inter column void spacing grow at rates slower than that of the column diameter. This indicates that whereas the absolute spacing between the columns grows and is significant after a thickness of several micrometers, columns are becoming relatively closer to each other as they grow. The self-shadowing effect appears to change with increasing film thickness and diameter, being significant for small initial nuclei and the beginnings of columns but less so as columns grow. This indicates that the growth scaling of intentionally seeded GLAD films may follow a slightly different initial growth trajectory than those that naturally nucleate. The growth scaling properties of both column diameter and spacing will also have to be taken into account for applications employing GLAD structures, such as microfluidic separation devices for which inter column spacing may be a critical parameter [91].

The column density is plotted as a function of thickness in Figure 5.28d. As expected, the density of columns per unit area decreases as the columns broaden. There are a large number of rods close to the film-substrate interface, but as these nanocolumns grow and compete, only the largest survive. A fit to the trend reveals an inverse power law scaling of d = 3400 $t^{-0.65}$ columns μm^{-2} for t from approximately 100 nm to 5000 nm. Compensation for the actual thickness of the film results in a revised fit of d_{act} = 3402 $t^{-0.65}$ columns μm^{-2}. Because this parameter is cast in terms of area, the inverse doubling of the column diameter power-law exponent is expected because the image contrast was set for each image to achieve a constant pixel density. The rate of density decrease is also as expected from the rate of growth in nanocolumn diameter. As the columns grow in size, the number per area will obviously decrease.

5.5.6 Sensitivity Analysis

The sensitivity of the grain detection routines to changes in algorithm parameter values was found such that the errors in growth scaling factors could be estimated. The filtering coefficient (f) used to filter the images (as described in Equation 5.16), the target density (ρ_{target}) used to normalize the thresholding for all images (as described in Algorithm 5.2) and the grain detection parameter values shown in Table 5.2 were adjusted in tandem, each by a random amount. A sub-selection of slice images were processed ten times, each time with an randomly altered grain detection algorithm parameter set. In the first iteration, all parameter values were set to nominal values. In the nine subsequent iterations, all parameters were randomly adjusted by up $\pm$ 10% of their nominal value, a change felt to reflect a reasonable range of adjustment. The only exception to this scheme was that the filter coefficient which was allowed to take on values from 3 (nominal) to 5.

Figure 5.29 shows the distribution of column diameter, center to center spacing, and density values found for the selected slice images. At thicknesses near the top of the film, the spread in es-

timated diameters and spacing widens. Conversely, near the substrate-film interface, the uncertainty in column density increases.

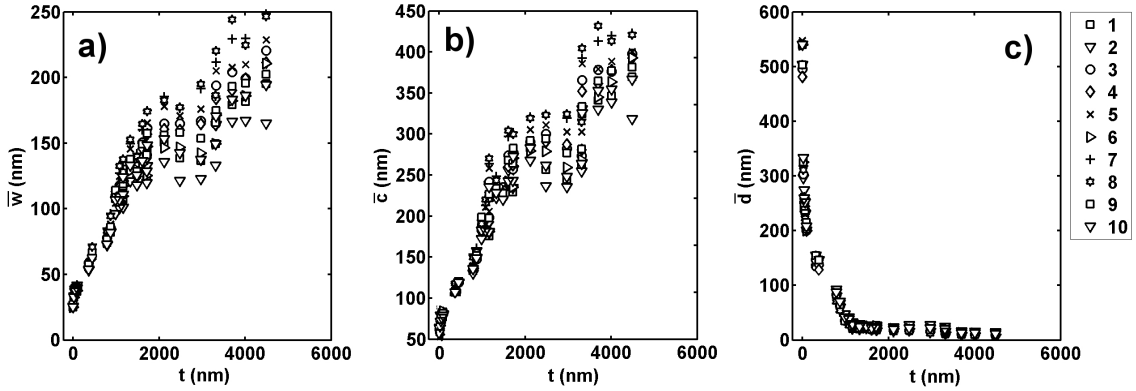


Figure 5.29: (a) Spread in mean column diameter values, (b) spread in column center-to-center spacing between columns and (c) spread in column density as grain detection algorithm parameters changed by up to $\pm$ 10%.

The difference in scaling constants (Δw, Δc, Δd) and factor (Δp_w, Δp_c, Δp_d) for the three parameters identified using the grain detection algorithm (column diameter, column center to center spacing, and column density) from the nominal value for each sensitivity test was calculated. The spread in scaling constant and factor differences are shown in Figure 5.30. This spread was then used to estimate the sensitivity of the growth scaling parameters to changes in algorithm variable values. Due to only testing over ten sample sets, the error in growth scaling parameter was taken as the maximum $|\Delta p|$. The errors in column diameter, center to center spacing, and column density scaling parameters were therefore estimated to be $\pm$ 0.09, 0.09, and 0.15 respectively. Errors in respective scaling constants were 3.0 nm, 5.8 nm, and 2500 columns μm^{-2}. This resulted in scaling parameters of $\overline{w}_{act}$ = (9.4 $\pm$ 3.0) $t^{0.35\pm0.09}$ [nm], $\overline{c}_{act}$ = (24.8 $\pm$ 5.8) $t^{0.31\pm0.09}$ [nm], and d_{act} = (3400 $\pm$ 2500) $t^{-0.65\pm0.15}$ [columns μm^{-2}]. The error in the void spacing was estimated to be on the order of that found for the column center to center spacing (25%), such that $\overline{v}_{act}$ = (15.2 $\pm$ 3.8) $t^{0.25\pm0.06}$ [nm]. Changing column identification and analysis parameters by up to $\pm$ 10% therefore has an impact of up to 25% on the growth scaling (p) parameters and 25% to 30% on the empirical growth constants for column diameter and spacing. An even larger impact on column density growth constant is the result of the wide distribution of densities at the high density near-nucleation region of the film and the sensitivity of the fit to changes in estimated column density in this region of the curve.

5.5.7 Growth parameter relationships

To verify that the thickness scaling relationships of the mean columnar width, void spacing, and center-to-center spacing agreed, a quick check was performed. Because the cross-sectional density

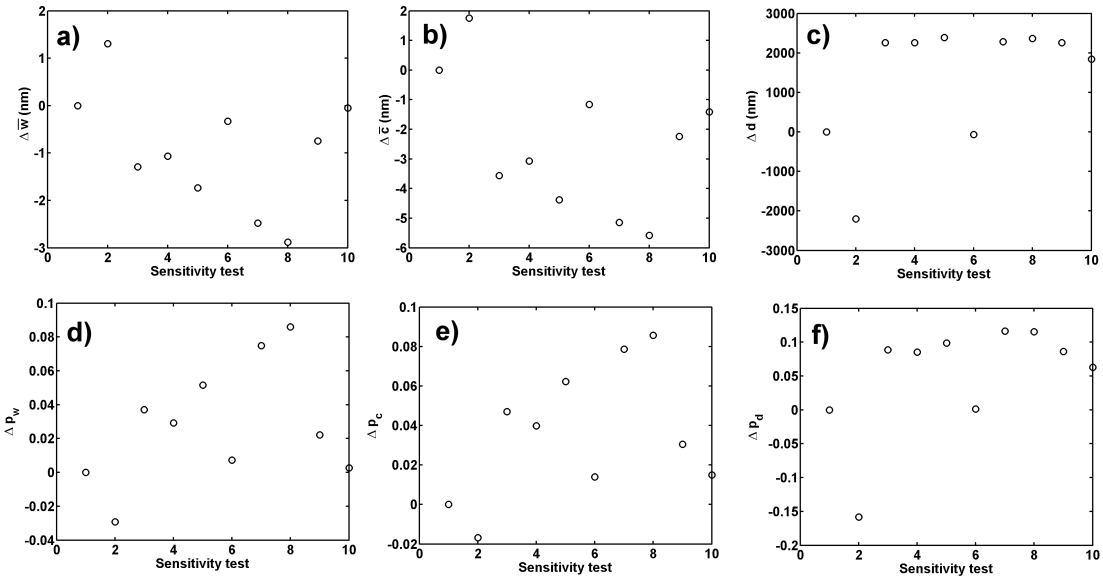


Figure 5.30: Difference between growth scaling empirical constants and scaling parameter for (a) and (d) mean column diameter values, (b) and (e) column center-to-center spacing between columns and (c) and (f) column density as grain detection algorithm parameters changed by up to $\pm$ 10%.

is roughly constant throughout the thickness of the film (with the exception of the initial growth region and the region at the top of the film), the column center-to-center spacing ($\overline{c}_{est}$) should be roughly equal to the sum of one column diameter and the void spacing between columns:

$$\overline{c}_{est} = \overline{w} + \overline{v}. \tag{5.28}$$

Point by point summation of the column diameter and void spacing data points in Figure 5.28a and c, with subsequent fitting to a power law relationship, reveals an expected column center spacing of $\overline{c}_{est}$ = 23.3 $t^{0.31}$ [nm], which is close to that determined from Figure 5.28b.

An additional area to explore is the relationship between column density and column spacing. While GLAD nanostructures fabricated from metal oxide films are not regularly arrayed (some organic GLAD structures are quite regular), as can be seen in Figure 5.3a, it is of interest to understand how the growth parameters found above scale as compared to those of an ideal closed packed arrangement. The regularly arrayed structure shown in Figure 5.31 was used to calculate the expected column density (δ_{hcp}) scaling given the calculated center to center spacing calculated above ($\overline{c_{act}}$).

For this exercise, the effective number of columns ($N_{columns}$ = 3) in a footprint area (FA) surrounding one closed packed column was calculated as follows:

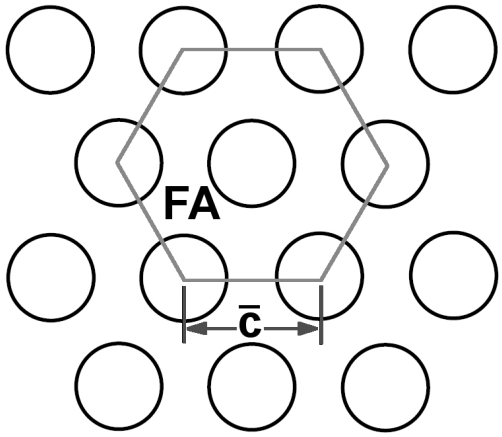


Figure 5.31: Closed packed array of nanocolumns is used to calculate expected column density and surface area scaling as a function of film thickness.

$$\begin{aligned}
\delta_{hcp} &= \frac{N_{columns}}{FA} && (5.29)\\
&= \frac{3}{6\frac{\overline{c_{act}}}{2}\left(\tan(60)\frac{\overline{c_{act}}}{2}\right)} && (5.30)\\
&= \frac{2}{\sqrt{3}\overline{c_{act}}^2} && (5.31)\\
&= \frac{2}{\sqrt{3}\left(24.5t^{0.30}\right)^2} && (5.32)\\
&= 0.0019t^{-0.60} \quad [columns\ nm^{-2}] && (5.33)\\
&= 1900t^{-0.60} \quad [columns\ \mu m^{-2}]. && (5.34)
\end{aligned}$$

The expected column density, had the GLAD nanostructure been closely packed, is somewhat less compact than that found from the analysis of the FIB milled cross-sections of the 5 μm film (d_{act} = (3400 $\pm$ 2500) $t^{-0.65\pm0.15}$ [columns μm^{-2}]), as can be seen in Figure 5.32. However, they are within error of each other and predict similar column densities by a film thickness of 5 μm (13.5 columns μm^{-2} for the actual film and 11.5 columns μm^{-2} for the closed packed arrangement).

5.5.8 Surface area prediction

To complement the work done in Chapter 3, where the surface area enhancement of GLAD nanostructures was assessed using gas adsorption porosimetry, the surface area was estimated from the post-processed milled images. The FIB-based surface area enhancement (SAE_{FIB}) was calculated for a given slice image by multiplying the perimeter area of each column by the differential thickness of the given slice. The cumulative surface area, as a function of film thickness, was then determined.

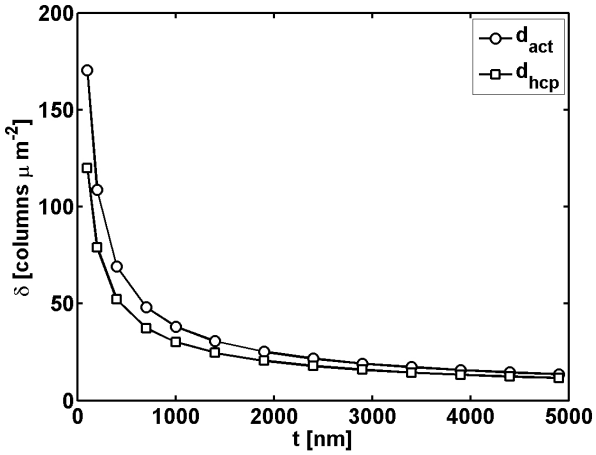


Figure 5.32: Measured column density (δ_{act}) versus that predicted for a closed packed array of nanocolumns (δ_{hcp}).

The code used to calculate the differential and cumulative surface areas is provided in Appendix C. The cumulative surface area is shown in Figure 5.33a.

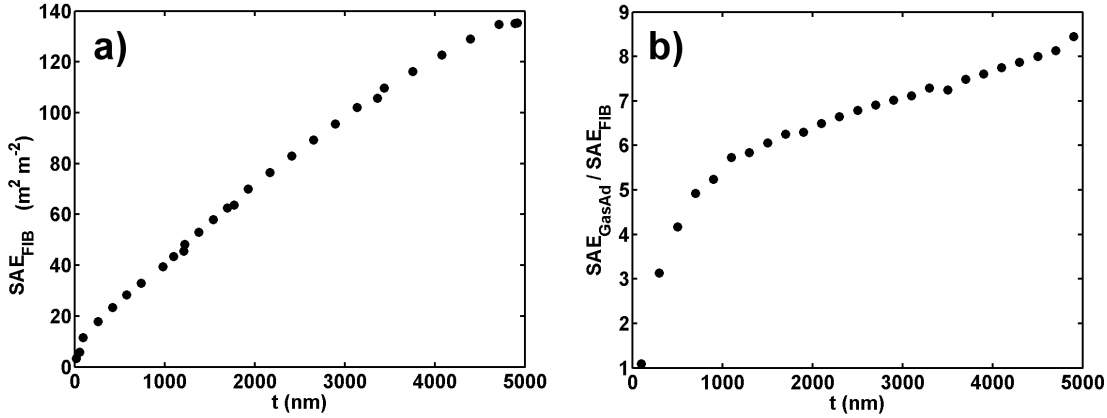


Figure 5.33: (a) Cumulative surface area enhancement (SAE_{FIB}) growth trend as a function of film thickness calculated from slice image column perimeter areas and differential thickness. (b) Comparison of FIB-calculated surface area enhancement to that measured using gas adsorption porosimetry (SAE_{GasAd}) for TiO_2 α = 85^o samples, shown in Figure 3.8. Prediction based on FIB-milled GLAD columns underestimates surface area and predicts a sublinear growth trend.

The surface area growth, predicted from the FIB-milled images, is sublinear. This is in contrast to the linear growth trend ($SAE = 0.235t$ for the α = 85^o films, with t in [nm]) calculated by the gas adsorption technique and seen in Figure 3.8. In addition, the predicted surface area enhancement is several times lower than that expected from Figure 3.8, as shown in Figure 5.33b. The sublinear growth trend predicted by the FIB method is not unexpected. Details, such as columnar roughness, are not visible in the acquired SEM images. Because the image resolution is only several to tens

of nanometers, the GLAD columns appear smoother than they actually are. The sublinear growth of the smooth GLAD columns is to be expected. To achieve a linear surface area trend, column roughness and bifurcation must be taken into account.

5.6 Tracking individual columns

In addition to calculating average properties for each slice image of the GLAD film, several individual columns were tracked. These columns, identified in the initial slice image shown in Figure 5.34a, were tracked from 840 nm above the substrate toward the top of the film at 3080 nm. Because of stage drifting during the FIB slicing process, this is as far as the columns could be tracked. Only one nanocolumn (column 1) continues beyond 3080 nm but could not be followed further because it reached the edge of the FIB-polished sample side face.

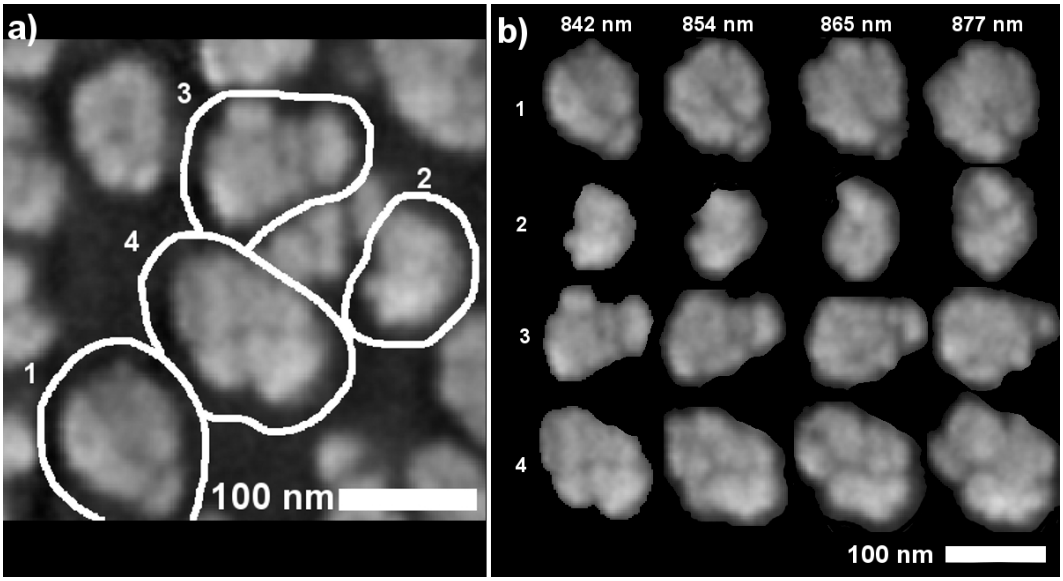


Figure 5.34: Tracking of four individual columns beginning at 842 nm from the substrate. (a) Initial configuration of the columns and (b) Evolution of these columns through several tens of nanometers.

Tracking of these nanocolumns involved matching column cross sections from image to image using custom MATLAB routines and the Gimp (GNU image manipulation program, v2.6.8) image processing program. Figure 5.34b shows how the four tracked nanocolumns evolve over the first few slices. One can see that there is significant variation from slice to slice as a single column grows. This variation is partly due to the random growth of the columns as deposited atoms land and diffuse, but also due to the complex inter-column shadowing that occurs as the flux impinges on the film surface. Variations in deposition flux rate, local temperature, as well as the movement

of the substrate as deposition angle and rotation angle are altered lead to a dynamic deposition environment. In this section the impact of this complex and dynamic system on the individual column morphology is investigated.

Each column was identified within a given cross-section using a thresholding algorithm similar to that described in Section 5.5.1. Each cross-section was thresholded to separate the columnar regions from the inter-column void spaces. However, for some slices such as those shown in Figure 5.35a, neighboring columns were so close that they touched. Automatic thresholding and identification of individual columns was problematic. The simplest approach to isolating individual columns was, in this case, to manually separate the columns by drawing lines between the columns being tracked, using the Gimp image editing software. This is illustrated in Figure 5.35b. The MATLAB *bwlabel* routine was then used to label connected pixels as single domains (i.e., columns). Figure 5.35c shows the thresholded isolated column. The thresholded domains were then filled, as shown in Figure 5.35d, and used as masks to isolate individual columns (shown in Figure 5.35e).

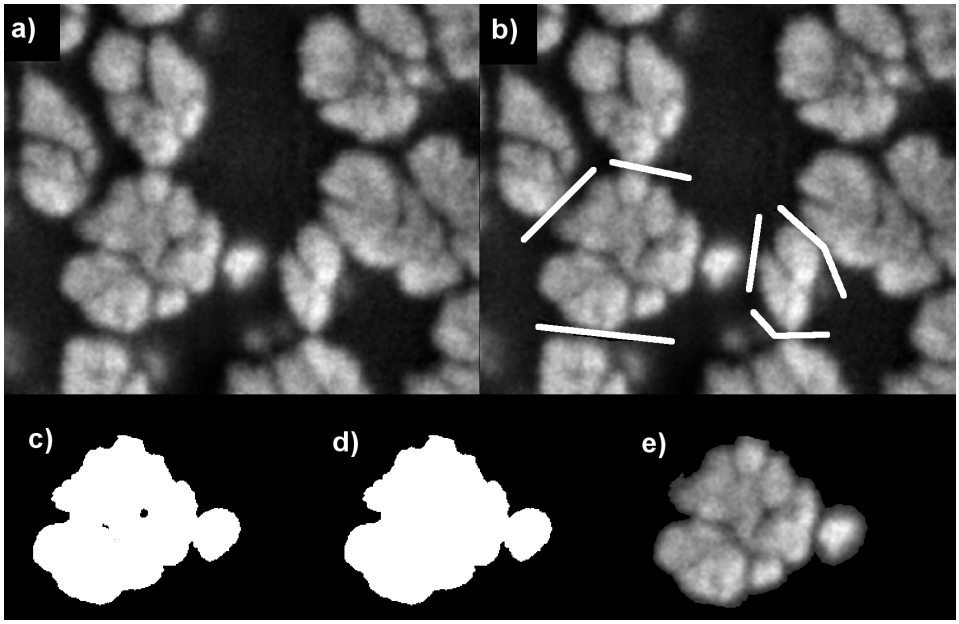


Figure 5.35: Columns in (a) an SEM cross-section are manually separated by drawing lines between the columns (where the lines are thickened and colored white for emphasis) as shown in (b). The image is then thresholded as shown to isolate individual columns as shown in (c). These columns are filled as shown in (d) to be used as a mask to determine (e) the cross-sectional features of the nanocolumn being tracked.

The matching of isolated nanocolumn cross-sections from slice to slice is illustrated in Figure 5.36. To locate the cross-section of a given nanocolumn in a given slice image, the cross-sectional image from the previous slice (Figure 5.36a) was compared to the region most likely containing that same nanocolumn (Figure 5.36b) in the current slice image. The likely region was

moved by a number of pixels in the row and column directions until the location of the best match (centered at (row_b,col_b) in image I_s) between the overlayed cross-sections was achieved. Using a routine similar to that described in Algorithm 5.1, the best match was determined from the RMS sum of the pixel differences between the two images (illustrated in Figure 5.36c). To isolate the cross-section of the nanocolumn in the current slice this portion of the image, centered at the best match location, was then thresholded as described above to isolate the required column. The process continued for the subsequent slice cross-sections.

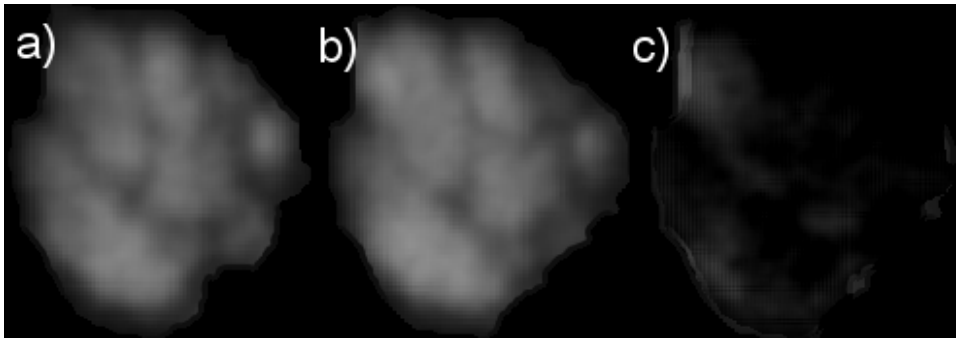


Figure 5.36: (a) and (b) Columns in two consecutive slices are overlayed to determine whether they match. The RMS sum of the pixel differences is shown in image (c).

To investigate the relative behavior of the four nanocolumns under investigation, several metrics were monitored as a function of column height (t). The diameter (w_i) of each of these columns (i) is plotted in Figure 5.37a. What is immediately apparent is that the diameter of column 2 rises and modulates for several hundred nanometers and then falls off as the structure disappears. In contrast, for several hundred nanometers the diameters of columns 1, 3, and 4 rise together, with their diameter growing to several hundred nanometers. The diameter of column 3 then levels off and begins to drop. Subsequently, despite following a growth trajectory until 2000 nm, column 4 dies out by 2760 nm. Only column 1 sees its diameter continue to rise without receding. A comparison of Figure 5.37a with Figure 5.28a may reveal why either column 1 or 4 survives at the expense of the other. By 2000 nm, columns 1 and 4 both have a diameter of approximately 200 nm. This exceeded the mean diameter expected at this film height. Two neighboring columns that exceed the mean diameter predicted by the growth scaling model compete with each other such that one must expire.

Having calculated the surface area growth trend expected from the average growth scaling parameters, the cumulative surface area ($SA_c(i,j)$) of each individual column (i) was of interest. The surface area, up to a given cross section (j), can be calculated from the following equation:

$$SA_c(i,j) = SA_c(i,j-1) + a_{i,j}\Delta t_j \quad [\mu m^2]. \tag{5.35}$$

where $a_{i,j}$ is the perimeter circumferential area of the nanocolumn and Δt_j is the thickness of the slice. The surface area relative to the footprint area (FA) under the four initial columns is given by:

$$A(i,j) = \frac{SA_c(i,j)}{FA} \qquad [\mu m^2 \mu m^{-2}]. \tag{5.36}$$

The cumulative relative surface area is plotted in Figure 5.37b. This plot reveals that as a column reaches its height limit, the surface area curve inflects and levels off. Interestingly, while the column continues to grow and before it levels off, its surface area rises in a slight superlinear trend.

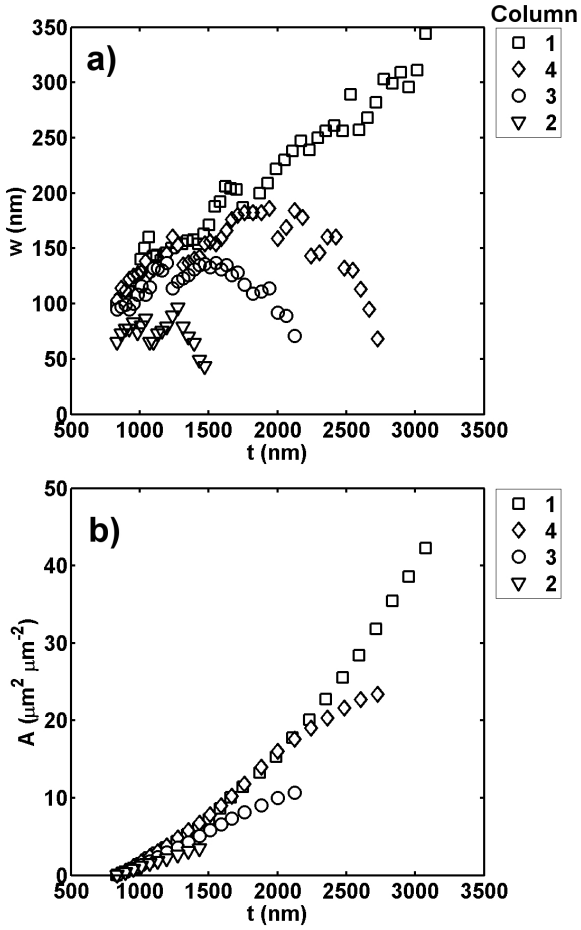


Figure 5.37: (a) Column diameters and (b) cumulative relative surface areas of individual neighboring columns evolve as a function of thickness. Only column 1 survives, while the others become extinct.

The contrast between the superlinear trend in surface area evolution for column 1, the column which survives at the expense of the others, and the linear growth in surface area seen in Figure 3.8 of Chapter 3 and the sublinear growth predicted for a modelled smooth GLAD nanostructure (Section 5.5.8) is striking. This superlinear growth for a surviving column implies that the linear growth

trend of the macroscopic average of surface area is not shared by all columns. While the total surface area of a GLAD nanostructured sample is composed of the surface area of all columns, some of those columns are growing at a rate which will inevitably cause them to die out. The composite surface area is made up of superlinear columns which will survive (at least for some time) and the remaining incremental surface area growth is made up by the columns which become extinct. For example, the surface area of column 1 grows at approximately $t^{2.0\pm0.5}$. Close examination of the surface area curve for column 4, the column which survives second-longest, reveals that its growth trend is also superlinear with a similar exponent up to 2000 nm. Survivors may require a growth trend with an exponent that exceeds a certain value. Any reduction from this superlinear trend, by the shadowing of neighbors, the extinction of a branch or random chance may cause a column to die out.

To examine the bifurcation of the tracked columns, the branches were identified for each cross-sectional slice of each column. As an example, Figure 5.38 illustrates how the branches of one nanocolumn evolved over several hundred nanometers. For each of the columns tracked (i), the number of branches ($B_{i,j}$) and the diameter of each branch ($w_{B_i,j,k}$) was also captured as a function of film thickness (i.e., slice number j and branch k). To determine the location of the nanocolumn branches, each columnar cross-sectional image was manually assessed using the Gimp software. The thresholded cross-sections were used as a guide to the eye in determining where branches existed in the grayscale slice image of the given nanocolumn (I_{nc}). Any columnar region with a hole, cut, or line of significance running through it was taken to be a branch. Using the Gimp software, some manual "connecting of the dots/lines" was performed to isolate branches by drawing black lines on the isolated cross-sectional columnar images were appropriate. In addition, in the MATLAB routine used to assess the branching properties, branches that became well-separated from the main body of the column (such as the branch to the right of the main column at 2556 nm in Figure 5.38) were noted as connected to the given column.

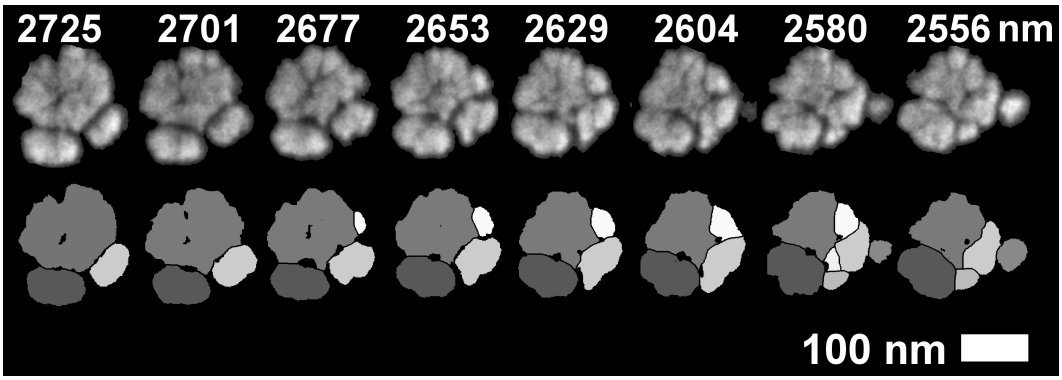


Figure 5.38: Tracking progress of column 1 bifurcation through several slices from 2556 nm to 2725 nm.

Once branches had been identified, Algorithm 5.6 (with code provided in Appendix C) was used to assess the properties of each branch. In this algorithm branches are tracked through the slices by assessing their similarity (closeness in area, p_A = 20%) and location (closeness in center location, p_d = 20%).

Algorithm 5.6 Columnar branching properties

For each nanocolumn, $i = 1..4$
For each cross-sectional slice image of the nanocolumn
($I_{nc_i,j}(row = 1..row_{Nnc}, col = 1..col_{Nnc}), j = 1..N_{nc}$)
A) Threshold and isolate the branches of the nanocolumn using the MATLAB *bwlabel* routine
B) Record the number of branches, $B_{i,j}$
C) Iterate through each branch and calculate metrics

For each branch ($I_{branch_i,j,k}(row = 1..row_{Nb}, col = 1..col_{Nb}), k = 1..B_{i,j}$)
1) Record the center row and column ($row_{center_i,j,k}, col_{center_i,j,k}$)
of the branch in the original image ($I_{nc_i,j}$)

2) Calculate the area of the branch,
$A_{B_i,j,k} = \sum_{row=1}^{row_{Nb}} \sum_{col=1}^{col_Nb} I_{branch_i,j,k}(row, col) \cdot R_{image}$

3) Calculate the diameter of the branch,
$w_{B_i,j,k} = \frac{\sqrt{A_{B_i,j,k}}}{pi}$

4) Determine the quadrant of the branch,
$\Delta row = \frac{row_{Nb}}{2} - row_{center_i,j,k}$
$\Delta col = \frac{row_{Nb}}{2} - row_{center_i,j,k}$
If ($\Delta row \geq 1$ AND $\Delta col \leq 1$) then $q_{B_i,j,k}$ = 1
ElseIf ($\Delta row \geq 1$ AND $\Delta col > 1$) then $q_{B_i,j,k}$ = 2
ElseIf ($\Delta row < 1$ AND $\Delta col > 1$) then $q_{B_i,j,k}$ = 3
Else $q_{B_i,j,k}$ = 4
EndFor

D) Track the progression of branches. Match the current branch to one in the previous slice
For each branch ($I_{branch_i,j,k}(row = 1..row_{Nb}, col = 1..col_{Nb}), k = 1..B_{i,j}$)
For each branch in the previous slice ($k' = 1..Bi, j - 1$)
Calculate the percentage difference in area
$\Delta A = 100 \cdot \frac{A_{B_i,j,k} - A_{B_i,j-1,k'}}{A_{B_i,j,k}}$
Calculate the percentage difference in location
$\Delta d = 100 \cdot \frac{\sqrt{(row_{center_i,j,k} - row_{center_i,j-1,k'})^2 + (col_{center_i,j,k} - col_{center_i,j-1,k'})^2}}{w_{B_i,j,k}}$
Calculate the difference in quadrant
$\Delta q = q_{B_i,j,k} - q_{B_i,j-1,k'}$

If ($\Delta A < p_A$) and ($\Delta d < p_d$) and ($\Delta q == 0$) then
Possible match. Store the deltas for this branch match.
EndFor

If found match
If there is more than one possible match then the current branch
gets the same label as the minimum distance (Δd) branch in the previous slice.
Else
Consider this to be the start of a new branch. Create a new branch label.
EndFor
EndFor
EndFor

Figure 5.39 shows the 3D reconstruction of these four nanocolumns. The smallest of the columns tracked, column 2, survives only 700 nm from the initial tracking depth. In addition, despite having an initially similar cross-sectional area, columns 3 and 4 die out before column 1, at heights of 2170 nm and 2760 nm, respectively. Although it was not surprising that a column with a smaller diameter than that of neighboring columns would not persist, the survival of column 1 at the expense of columns 3 and 4 was worth exploring.

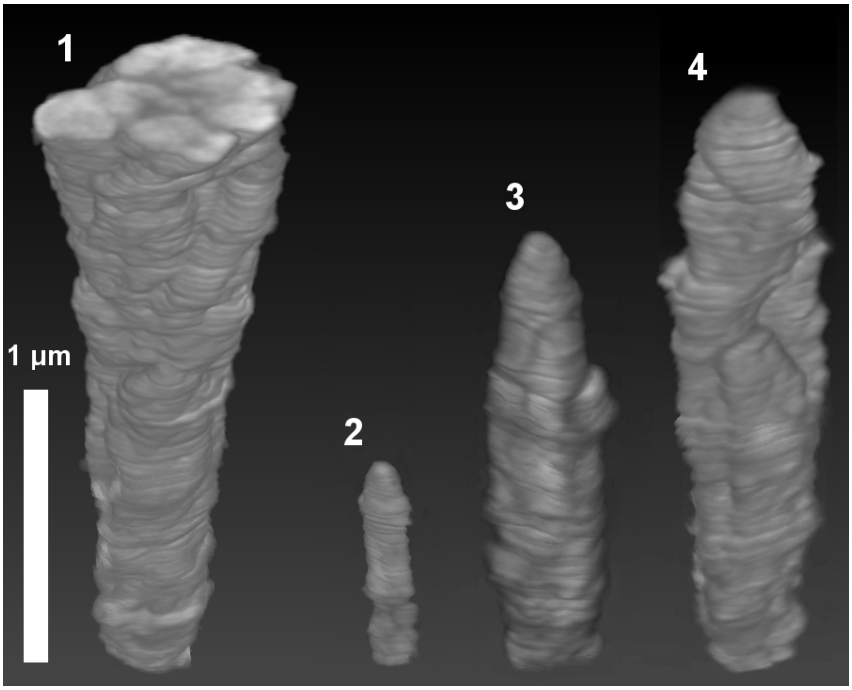


Figure 5.39: Tomographic view of four individual nanocolumns: from 840 nm to (1) 3080 nm, (2) 1510 nm, (3) 2170 nm, and (4) 2760 nm above substrate.

Figures 5.40 and 5.41 show the number and cumulative surface areas (SA_{cum}) of the individual column as a function of column height. The branches that were tracked are those with a diameter greater than 10 nm. The resolution of the acquired SEM images was not sufficient to study branching and bifurcation on a smaller scale than 10 nm. Further work with the TEM of FIB-sliced GLAD cross sections is planned to investigate the interesting nanometer-sized structural characteristics that have been reported for TEMs of individual GLAD columns [214]. Despite the resolution limitations, the data shown in Figure 5.40 may reveal some additional information about why some columns successfully out-compete their neighbors. Columns 2 and 3, which die out the earliest, have very few branches discernible at the resolution available. Columns 1 and 4, which survive to the tallest height, generally maintain many side arms. It may be that nanocolumns that grow branches of significant area are able to shadow their neighbors and steal incoming evaporant flux, thus ensuring their survival and growth at the expense of their neighbors. Investigation over a range

of image resolutions may yield more information.

Figure 5.41 shows the stacked cumulative surface areas of the individual branches. At each slice the surface areas of each branch was measured from the perimeter area of the branch multiplied by the differential thickness of the slice. The surface areas for all branches in the given slice were then cumulatively summed and presented in a stacked manner at the corresponding thickness. Individual symbols represent the surface area of one branch, summed with the branch surface area of any symbols shown below it in the figure. Consecutive identical symbols show the surface area evolution of an individual branch. As branches appear and disappear the cumulative surface area of remaining branches becomes adjusted. The maximum surface area presented at each thickness point is the total cumulative summation of the column surface area to that thickness. This figure shows that as the columns grow, their cumulative surface area increases, but the surface areas of the individual branches rise and fall as the columns evolve.

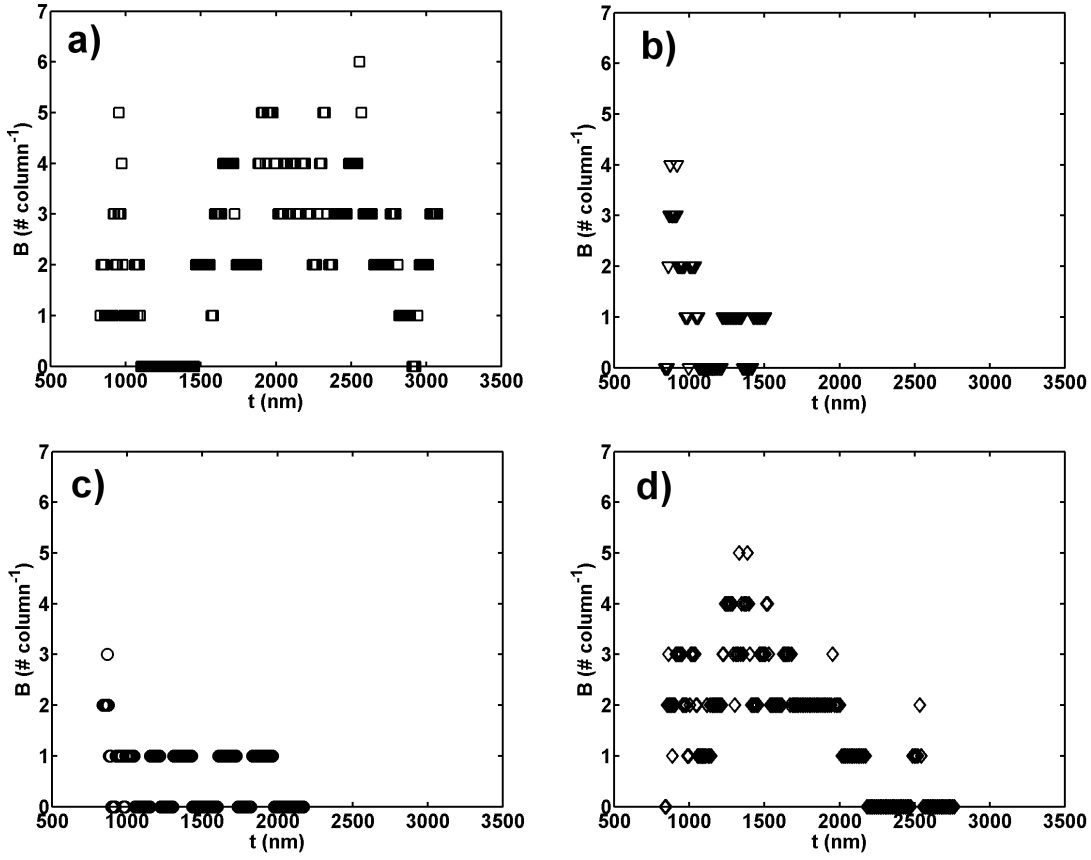


Figure 5.40: Number of branches per column for (a) column 1, (b) column 2, (c) column 3, (d) column 4. Column 1, which survives longer than its neighbors, has the largest peak number of branches.

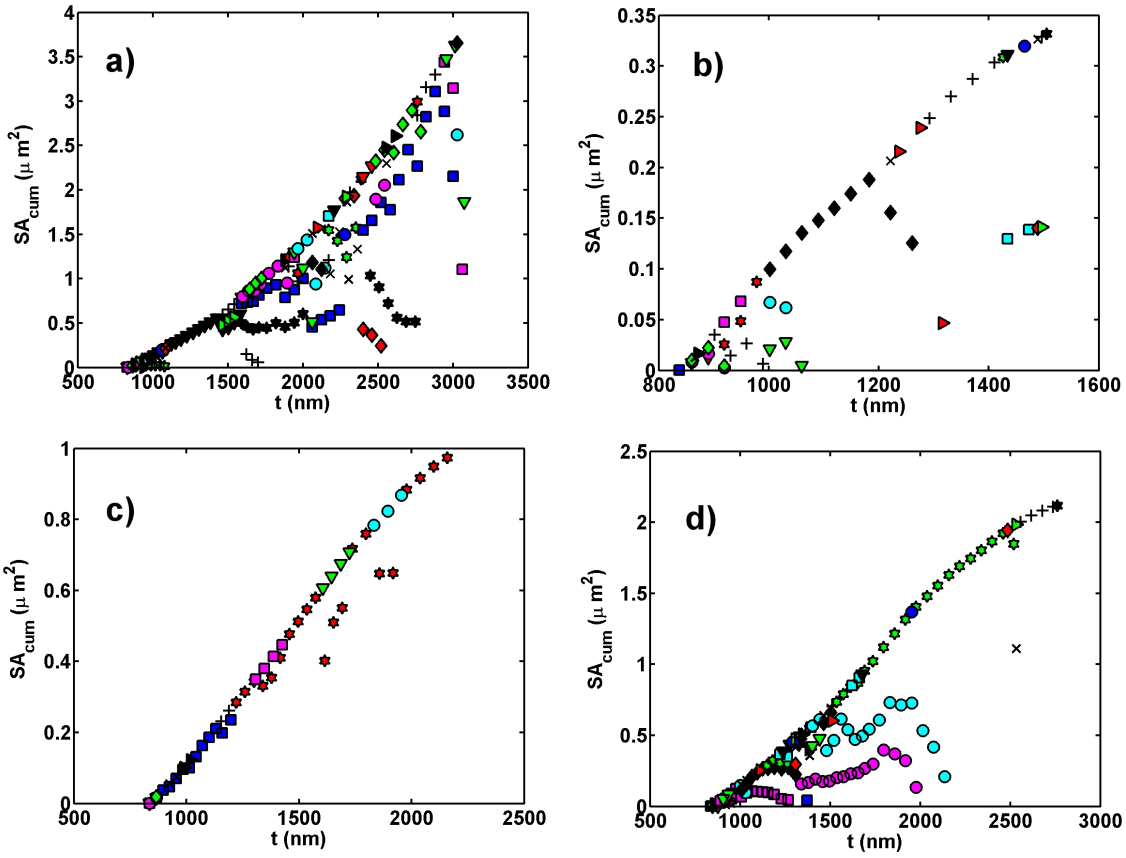


Figure 5.41: Cumulative surface areas of column branches for (a) column 1, (b) column 2, (c) column 3, (d) column 4. Surface areas for all branches at a given film thickness have been cumulatively summed. Maximum surface area envelope is the total cumulative sum to the given thickness.

5.7 Chapter Summary

FIB milling has been used to slice and image a porous nanostructure fabricated using glancing angle deposition. A technique used previously for lithography on GLAD, spin-in polymer filling, was successfully applied for FIB milling. The polymer supported the nanocolumnar array during milling and also provided contrast between the columnar material and background. Deposition of a carbon protection layer prior to milling appeared to minimize curtaining effects, as the final milled slices were uniform to within a few tens of nanometers. By performing multiple slicing sequences, each at an increasing magnification and resolution, the structure of a representative TiO_2 GLAD film was measured over a range of sizes.

Following postprocessing, which included image alignment and sequence stitching, a 3D reconstruction of the film was carried out. This reconstruction, the first of its kind, has allowed for the examination of the porous, columnar, and bifurcated nature of the structure. The identification of individual nanorods within the acquired cross sections has allowed for the measurement of column diameter and inter-nanocolumn spacing characteristics as a function of the GLAD film thickness. These parameters were found to follow a self-similar growth pattern, with the large columns that out-compete their neighbors resembling the small columns that initially grow. The growth scaling factors that were determined indicate a faster growth rate of nanorod diameters than nanorod spacing. Using these growth scaling trends, an estimation of surface area evolution was also made, with sublinear growth predicted for smooth columns where roughness was not taken into account. This was an interesting but not unexpected contrast to the linear trend seen in Chapter 3. Additionally, tracking of individual nanorods revealed some interesting trends in surface area, bifurcation and column survival. For example, there appears to be a link between number of branches per column and maximum height. Finally, the surface area trends of individual columns which survive their extinct neighbors appears to grow with superlinear trend.

Application of the FIB sectioning technique to other GLAD structures is recommended. Use of the FIB to section and view other nanocolumnar thin film structures will be useful for understanding not only growth scaling and bifurcation properties but also porosity and roughness. In addition, TEM probing of a thin nanostructured cross-section would allow not only the probing of individual column properties such as porosity, but also the intercolumn relationships.

FIB tomography provides a means for viewing the evolution of GLAD nanostructure properties not only as a function of film height, but also as individual columns. The ability to observe the modulation of columnar properties as they grow prompted the investigation described in the next chapter. Modulation of the columnar height, not as a function of thickness, but as a function of position on the substrate is next explored.

Chapter 6

Modulation of growth properties

6.1 Introduction

Having studied the fundamental characteristics of GLAD nanostructured films, an experiment in modulating their properties was undertaken. Nanostructures with a gradation to their thickness were created using a new shadowing technique. This chapter describes the technique and demonstrates its usefulness in creating tunable optical components. [1]

6.2 Nature-inspired tunable optics

In the field of photonics there is a need for, and many examples of, tunable optical components which select certain frequencies of light [230]. Many optical applications also require devices which provide circularly polarized incident light. Methods for creating these polarizing optical filters include oblique angle deposition techniques, [83], [85], [122], stacking chiral polymer films, [231] and liquid crystal fabrication processes [232]. It is, however, difficult to simultaneously provide circular polarization and frequency selectivity. This difficulty has provided the motivation for exploring natural systems which exhibit these properties. In addition, the expansion of fabrication techniques which might allow for the realization of structures that provide this dual polarization and tunable frequency functionality have been undertaken here.

One method commonly employed to naturally and artificially create structures with interesting photonic properties is the multiple layering of materials. Multilayered interference structures are found in a large range of natural and fabricated systems. By alternating layers of materials with high and low indices of refraction, thus creating a Bragg reflector, these structures impart light filtering, anti-reflection, and polarizing functionality to a material [233]. Properties such as material composition, porosity, thickness, and repeated structural pitch, all determine the characteristics of the spectral features and allow for the design of complex reflectance spectra that are useful for a

[1] Portions of the work documented in this chapter have been published in: K.M. Krause, M.J. Brett, "Spatially graded nanostructured chiral films as tunable circular polarizers", Advanced Functional Materials 18 (2008) 3111-3118.

variety of photonics applications [234]. The Bragg wavelength,

$$\lambda_{avg} = n_{avg}\Pi. \tag{6.1}$$

where n_{avg} is the average refractive index of the multilayer structure and Π is the pitch, describes the relationship between optical performance and multilayer structure.

An additional complexity that can be introduced to nanostructured multilayer films is thickness gradation, useful for applications such as compact spectrometers and optical components [60], [235]. Gradation of thickness allows for a variation in color across photonic structures. Several methods have been employed to manufacture spatially graded films. For example, sliding masks are commonly employed during sputtering [235], [236], molecular-beam epitaxial growth [237], and chemical deposition [238], to provide a thin film with a sloping thickness. An alternative method is to exploit the dependency of the film thickness on the distance and angle of incidence between substrate and evaporant source [239]. While each of these techniques can create a thin film filter with frequency selectivity, concurrent polarization discrimination has not yet been achieved.

Nature has provided the inspiration for exploring the creation of both frequency and polarization selective thin films. A host of animals employ complex multilayer structures on their skins, feathers, wings, and exoskeletons for communication and protection [240], [241], [242], [243]. Many species of butterfly owe their coloration to multilayer structures and some also exhibit remarkable polarization phenomena [243], [244], [245], [246]. The nanostructured exocuticle of scarab beetles has been found to impart both coloration and polarization [247]. In these beetles, helically ordered chitin micro-fibrils result in the circular polarization of reflected light and coloration due to the exocuticle layer spacing and structure [248], [249]. There are also examples of birds and beetles, such as Phasianida (pheasants) and Plateumaris sericea (leaf beetles) that exhibit a color variation along their skin and shells, respectively, which depend on the thickness of their multilayer structures [250], [251]. Mimicking this frequency variation, while coupling it with the polarization selectivity found in certain animal specimens, would be very useful for building polarization responsive and tunable frequency optical filters, sources, and detectors for applications such as multichannel lightwave systems [252], [253].

6.3 Graded thickness method

Using the GLAD method, thin films with a variety of nanostructures can be created. By depositing a film at an oblique angle, without rotating the substrate, slanted posts typically result. The characteristic columnar slant occurs in the direction of impinging flux. With slanted post films, a thickness

gradation is also typical due to the fact that the side of the substrate chuck closest to the evaporant source receives a higher density of flux than the side farthest from the source [134]. However, when substrate rotation is employed, the flux hitting all areas of the chuck is averaged such that the thickness is fairly even. Also, the flux impingement effectively occurs from all sides of the growing column, resulting in more upright columns. Slow substrate rotation results in the formation of helical structures, while faster substrate rotation allows these chiral columns to degenerate into vertical post structures. In the case of chiral structures the handedness is determined by the substrate rotation direction, while the pitch is controlled by the speed of rotation and deposition rate.

Here, a variation of GLAD is introduced that allows for a graded thickness nanostructure to be achieved. This thickness gradation is accomplished by adding a macroscopic shadowing object to the substrate. The shadowing object modulates the microscopic structure of the film and creates a spatially graded chiral film with graded optical properties. As illustrated in Figure 6.1, deposition within the macroscopic shadowing region will be affected and a graded thickness profile will be achieved. By employing substrate motion, the shadowed regions area and duty cycle can be controlled, thus controlling the thickness gradient of the deposited structured thin film.

For this work, the shadowing object was several centimeters high, thus casting a shadow that was on the order of several centimeters long. This was done to create a polarizing device that had a thickness gradation and frequency response that varied across a length typical of the size of commercially available polarizing filters. In the experiments described below, substrates with lengths on the order of several centimeters were used in order to measure gradation effects imposed by the shadowing block.

6.4 Experimental details

6.4.1 Sample preparation

To explore the creation and use of macroscopically shadowed GLAD for optical devices, several titanium oxide (TiO_2) nanostructured thin films were created. TiO_2 was selected as the deposition material because of its high refractive index, low absorption in the visible regime, and its successful previous use in creating helical films with strong optical activity [83], [122]. Of these shadowed thin films, the most interesting were those fabricated using slow substrate rotation in order to achieve spatially graded helical films. However, in order to predict the thickness variation that would be achieved as a result of the macroscopic shadowing block, a set of calibration films were first fabricated.

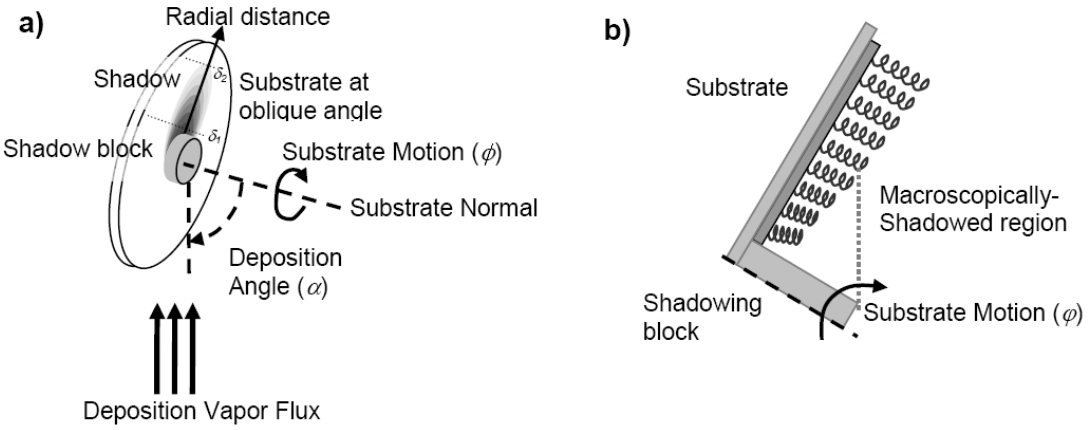


Figure 6.1: Shadowed GLAD method. (a) A shadow block placed in the center of the obliquely inclined substrate holder (deposition angle α) casts a shadow (from point δ_1 to δ_2 along radial axis of substrate) on the substrate. (b) This results in a graded thin porous film with structure defined by the substrate motion characteristics, such as rotation angle (ϕ).

For all films, the deposition material TiO_2 (Cerac, Inc., 99.9% pure rutile) was evaporated under partial pressure of O_2 gas. The addition of O_2 gas flow promotes the formation of stoichiometric TiO_2. By varying the O_2 gas flow, the pressure was typically kept at $8x10^{-3}$ Pa (system base pressure below $1x10^{-4}$ Pa). All films were grown on substrates of fused-silica and polished silicon that had been placed on a flat circular substrate holder. The deposition rate was monitored using a quartz crystal oscillator. Substrates were not dynamically temperature controlled during the deposition process [125]. At the center of the substrate holder was a shadowing block, as shown in Figure 6.1a. The shadowing block was circular with a 24 mm radius and 20 mm height.

For the set of slanted post calibration films, the target (t_{target}) thicknesses of substrates at the center of the deposition sample holder (chuck) are provided in Table 6.1 (with t_{center} described below). For the graded thickness helical films, the target (t_{target}) thickness and the number of turns (N_{turns}) are provided in Table 6.2. Other parameters are discussed below.

α (^{o})	t_{target} (nm)	t_{center} (nm)
30	1000	1100 ± 50
60	1000	900 ± 50
70	1000	1000 ± 50
85	1000	1300 ± 50

Table 6.1: Properties of shadowed slanted post TiO_2 films used for calibration of shadowing simulations. Values for the target (t_{target}) and actual (t_{center}) thicknesses of substrates at the center of the deposition sample holder are provided.

α (^{o})	t_{target} (nm)	$t_{(50mm)}$ (nm)	N_{turns} (nm)	Δt_{full} = $t_{(50mm)} - t_{(5mm)}$ (nm)	Δt_{upper} = $t_{(50mm)} - t_{(20mm)}$ (nm)	$\Pi_{(50mm)}$ (nm/turn)
60	3600	3300 $\pm$ 10	8.5	820 $\pm$ 20	440 $\pm$ 20	388 $\pm$ 1
65	3600	3150 $\pm$ 10	8.0	880 $\pm$ 20	520 $\pm$ 20	394 $\pm$ 1
70	3600	3350 $\pm$ 10	8.7	780 $\pm$ 20	330 $\pm$ 20	395 $\pm$ 1
75	3600	2900 $\pm$ 10	6.8	720 $\pm$ 20	270 $\pm$ 20	436 $\pm$ 1
80	3600	3800 $\pm$ 10	8.5	810 $\pm$ 20	290 $\pm$ 20	450 $\pm$ 1
85	3600	3600 $\pm$ 10	7.8	700 $\pm$ 20	210 $\pm$ 20	461 $\pm$ 1

Table 6.2: Properties of graded chiral films. The target (t_{target}) thickness, actual thickness at 50 mm from the edge of the shadow block ($t_{(50mm)}$), number of turns (N_{turns}), thickness gradation measured at 5 mm to 50 mm from the shadow block edge ($\Delta\ t_{full}$), thickness gradation measured at 20 mm to 50 mm from the shadow block edge ($\Delta\ t_{upper}$), and pitch of the helical columns at 50 mm from the shadow block edge ($\Pi_{(50mm)}$) are provided.

The calibration films were a set of slanted post structures, deposited at selected fixed oblique deposition angles and employing no substrate rotation. The target thickness for these films was 1000 nm, however because no substrate rotation was employed there was an inherent thickness gradation to the film. In addition, the shadow block casts a shadow behind it such that a region behind it received little or no flux, thus reducing the thickness in that region. The chucks were oriented within the deposition chamber such that one test substrate was directly behind the shadowing block (to measure the shadowed thickness profile), one was to the side (to measure the unshadowed radial thickness profiles), and one was below (to measure unshadowed thickness profiles).

The deposition substrate holder used for the deposition of the α = 60^o calibration film is shown in Figure 6.2. This images show the impact that the shadowing block has on the deposited film. There is an obvious shadow profile, cast by the shadowing block, which minimizes the deposited flux behind the shadowing block.

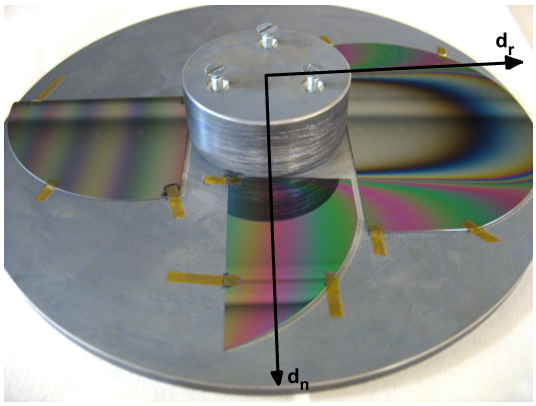


Figure 6.2: Image of deposition chuck and post-deposition shadowed slanted-post GLAD film at α = 60^o.

The graded chiral structures were achieved by applying clockwise substrate rotations (as seen from the evaporant source), such that the helix was right-handed. A nominal pitch of 450 nm was targeted by controlling the substrate rotation rate based on the monitored deposition rate. A representative chuck for the 80^o chiral deposition is shown in Figure 6.3. Bands of different coloration in the light reflecting from the substrates indicates that a thickness gradation is present.

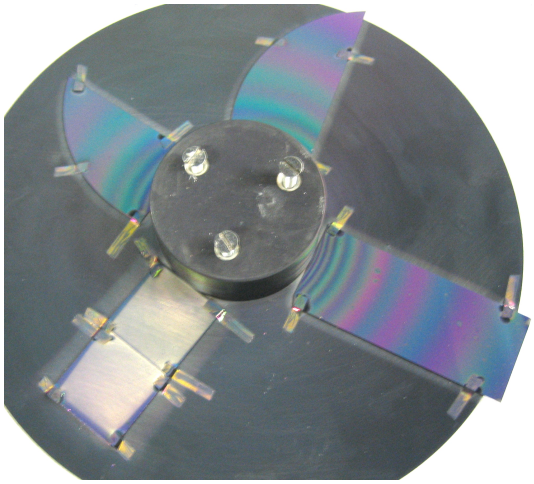


Figure 6.3: Image of chuck, post-deposition, of shadowed chiral GLAD film at α = 80^o.

The thicknesses of the thin films were measured with a Hitachi Field emission S-4800 SEM. No conductive coating was required for the measurements of these TiO_2 films. Both the calibration and graded chiral films, deposited on silicon wafer substrates, were cleaved into long pieces such that an edge along the radial direction from the center of the substrate holder was exposed.

As is described below, the thickness profiles of the calibration films were used to predict the thickness profiles of the helical films. Figure 6.4 shows several representative SEM images taken of the α = 70^o slanted post calibration film. These images were acquired at a distance (d_r) 9 mm behind the shadow block edge, and then at selected points normal to this radial vector. Thicknesses at the center of the chuck (t_{center}), at a radial distance of 0 mm, were measured from the SEM data and are provided in Table 6.1. In addition, the full thickness profiles of these calibration films are provided in Section 6.5.1. One can see that, even behind the shadowing block (at a), there is some deposition.

The cross-sectional images of the graded helical films, along the radial direction towards the top of the deposition chuck, were used to determine the thickness of the film as a function of distance from the edge of the shadow block. Representative SEM images of several graded chiral films are shown in Figure 6.5. Using these SEM images, parameters describing the characteristics of the graded helical films, such as the actual thickness at 50 mm from the edge of the shadow block

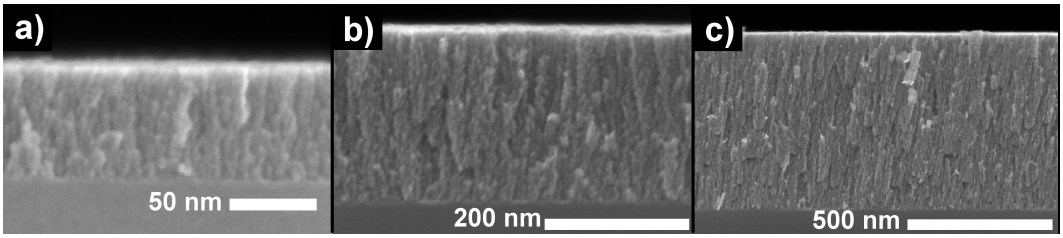


Figure 6.4: SEM images of shadowed slanted film deposited at α = 70^o. Measured along a line at a radial distance of d_r = 45 mm from the shadow block edge and at (a) d_n = 0 mm, t = 63 nm, (b) d_n = 27 mm, t = 240 nm, (c) d_n = 37 mm, t = 607 nm

($t_{(50mm)}$), thickness gradation measured at 5 mm to 50 mm from the shadow block edge ($\Delta\ t_{full}$), thickness gradation measured at 20 mm to 50 mm from the shadow block edge ($\Delta\ t_{upper}$), and pitch of the helical columns at 50 mm from the shadow block edge ($\Pi_{(50mm)}$) were determined. Values for these parameters are provided in Table 6.2.

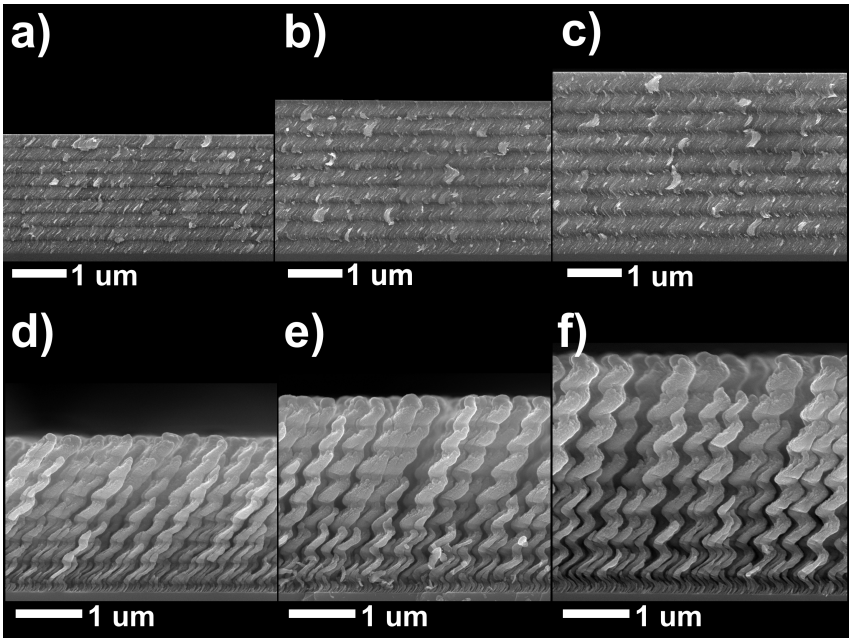


Figure 6.5: Images of spatially graded chiral films deposited at two different oblique angles (a through c deposited at 70^o and d through f deposited at 85^o) and shown at selected distances of (a) 0.3 mm, (b) 13 mm, (c) 54 mm, (d) 1 mm, (e) 13 mm, and (f) 56 mm from the shadow block edge.

6.5 Structural Characterization of Spatially Graded Helical Films

6.5.1 Thickness Gradation prediction

The characterization of the calibration films allowed for the nominal shadowing profile, cast by the macroscopic shadow block, to be determined for a range of glancing deposition angles. Calibration film thicknesses were measured at selected coordinates on the film and plotted versus normal (perpendicular to direction of evaporant flux) and radial (along direction of evaporant flux) directions. Once these nominal shadowing profiles had been found, a model was then constructed which could be used to predict the thickness profile which would result from depositions involving not only modulated deposition angle, but also substrate rotation.

The thickness measurements of the slanted post calibration films allowed for the shadow block shadowing profile to be determined as a function of deposition angle. In general, the shadowing profiles were found to be parabolic in shape, with the shadow reach set by the height of the shadow block and orientation angle of the substrate. The shadow did not abruptly end, but was instead a penumbra and yielded a radially graded film thickness profile. This was likely due to the finite area of the evaporant source, and the increased scattering of the evaporant flux due to the increased background pressure created by the addition of O_2 during deposition. Directly behind the shadow block there was a region (at approximately δ_1 of Figure 6.1) where little or no evaporant flux could deposit. Farther away (at approximately δ_2 of Figure 6.1), more flux was able to deposit and the thickness of the deposited film was increased. At the edges of the shadow the film thickness reached that of the nominal profile. As has been seen in [134], the nominal thickness profile of slanted post structures is graded due to the nonuniform flux deposition on the nearly vertically oriented substrate. The thickness profiles of the calibration films are shown in Figure 6.6. Only calibration films with deposition angles greater than 60^o were used to predict the thickness profiles of the graded chiral films described in this thesis, but the thickness profile for the shadowed calibration film at $\alpha = 30^o$ is shown for comparison. Here d_r is the radial distance from bottom (-100 mm) to top (+100 mm) of the angle deposition chuck. Thicknesses at $d_r = 0$ were measured alongside the shadowing block, along the normal vector. The normal vector is described by d_n.

To predict the thickness profile that would result from depositing a constant rotation (i.e., helical) film on a shadowed chuck, at one of the calibration deposition angles, the thickness profiles described in Figure 6.7 were integrated into a MATLAB simulation routine. Simulated thickness profiles for chiral films were then generated by modeling the deposition of material on a substrate fixed at the required deposition angle, while substrate rotation was modeled. The rotation of the chuck was performed at a simulated rate of 450 nm per rotation to match the experimental parameter.

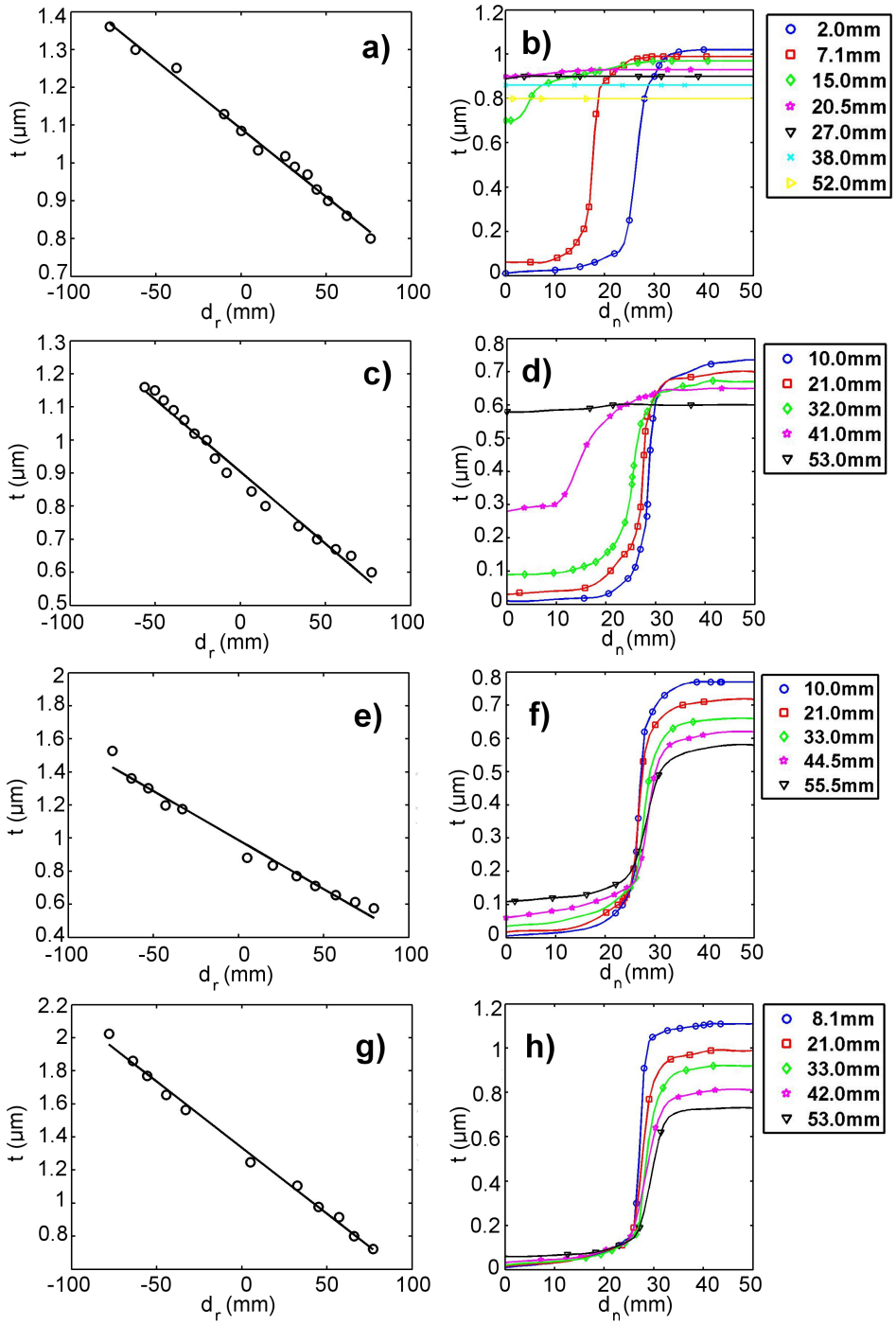


Figure 6.6: Slanted post calibration film thickness profiles at $\alpha = 30^o$ (a,b), 60^o (c,d), 70^o (e,f), and 85^o (g,h). Nominal thickness profile of slanted post film as a function of radial distance along the chuck is shown in the first column. Thickness profiles behind the shadowing block, at a number of distances from the shadow block edge, are shown in the second column.

The simulation involved the virtual deposition of a discrete amount of film at each grid point on the simulated substrate for each incremental change in rotation angle. The amount deposited at a given grid point depended on its current location (in normal d_n and radial d_r coordinates). The

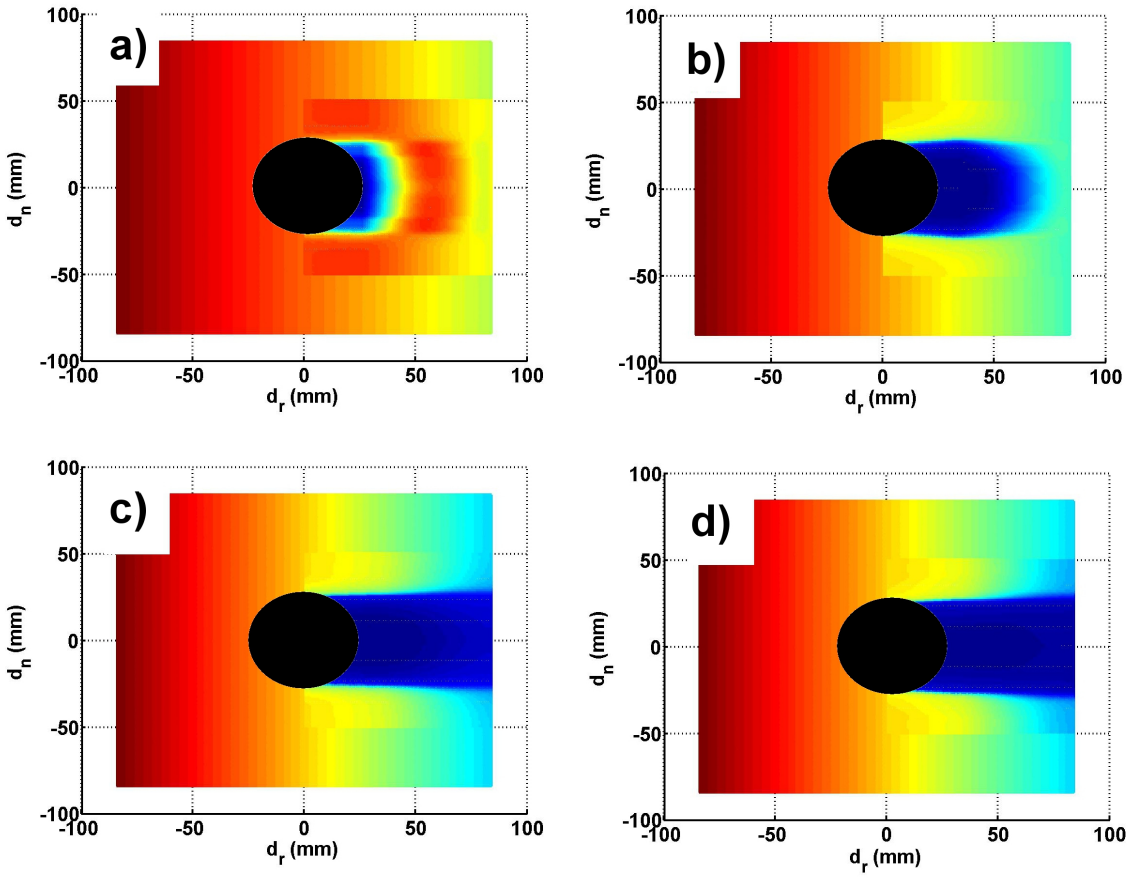


Figure 6.7: Combined shadowed (right half of each image) and unshadowed (left half of each image) slanted post calibration film thickness profiles at (a) α = 30^o, (b) 60^o, (c) 70^o, and (d) 85^o, with red being the thickest film and blue being the thinnest. Shadowing block region at center of simulated substrate holder set to zero thickness.

thickness profile shown in Figure 6.7 was overlayed over the substrate to determine the amount of flux received by each point on the substrate grid. For example, if a given grid point was behind the simulated shadowing block ($d_r > 0$ mm, $d_n < 25$ mm) then little flux was acquired. As this grid point moved out from behind the shadow block, it began to acquire flux, until it became fully exposed to the simulated evaporant flux ($d_r < 0$ mm). This process continued until the required number of substrate rotations was reached. The simulation algorithm is shown in Algorithm 6.1. [2]

[2]The code used to simulate the graded thickness profile for helical films is provided in Appendix D.

Algorithm 6.1 Graded thickness profile simulation

A) Define the shadow block radius (r_{sb} = 25 mm)
B) Define the substrate length (l_s = 50 mm)

C) Read in the calibrated shadowing profiles from the vertical post depositions $C(\alpha)$. These are an array of thicknesses (shown in Figure 6.7) on a grid in the radial [-d_{r_max}:d_{step}:d_{r_max}] and normal [-d_{n_max}:d_{step}:d_{n_max}] directions.

D) Read in deposition parameter file used in actual deposition of graded chiral films (Sample file shown here)

t (nm)	α (^{o})	ϕ (^{o})
0	60	0
450	60	360
900	60	720

E) Interpolate t, α, and ϕ to a finer grid of N_{steps}

F) Define a radial line along the substrate (s), divided into N_{rad} segments, each representing a growing GLAD nanocolumn, from s_1 = r_{sb} to s_{Nrad} = r_{sb} + l_s. This line of nanocolumns will be tracked during the simulated deposition process

G) For each thickness step, t_i, i = 1..N_{steps}
1) Determine the current change in thickness (Δt), deposition angle (α_i) and rotation angle (ϕ_i)

2) For each nanocolumn at points along the radial substrate slice, s_j, j = 1..N_{rad}
a) Determine the radial and normal location of the column relative to the calibration shadowing profile for the given α and ϕ

$$d_r = \sin(\phi) \cdot s_j$$
$$d_n = \cos(\phi) \cdot s_j$$

b) Perform weighted interpolation of grid points nearest to d_r, d_n on the calibration shadowing profile, $C(\alpha)$, to determine the scaling that should be applied to the incremental thickness for this deposition step ($\overline{C}$)

c) Add the incremental thickness to the total thickness (tc) for the given nanocolumn (j):

$$tc_j = \overline{C} \cdot \Delta t + tc_j$$

EndFor
EndFor

H) Compare the simulated total thickness of the nanocolumns along the radial vector to those measured for actual deposited film.

6.5.2 Graded thickness measurement

The second set of fabricated films was a set of spatially graded helical films. Optically transparent helical films of TiO_2 were fabricated at deposition angles of 60^o, 65^o, 70^o, 75^o, 80^o, and 85^o to demonstrate the macroscopically shadowed GLAD process and the ability to design films with varying frequency response and circular polarization sensitivity. The nominal pitch of these films was 450 nm per turn and the target number of turns (N_{turns}) was seven or eight. Such a high number of chiral rotations was chosen because of results from [85] indicating that a high circular birefringence (and thus high polarization selectivity) is achieved by adding more turns to the helical columnar structure. Helical films deposited at a deposition angle of approximately 70^o were of special interest because these have previously been shown to have minimal diffuse optical scattering and low film stress [254], [255]. These films were used to study the optical properties of graded helical films. The 85^o deposition angle was selected because such obliquely deposited thin films demonstrate visually the chiral nature of the process and can be used to inspect the impact of macroscopic shadowing on the graded helical film. Table 6.2 summarizes the properties of these graded helical films, with emphasis on the thickness and pitch of the film 50 mm from the shadow block [t (50 mm) and Π (50 mm), respectively] and the thickness gradation measured over a distance of 5 mm to 50 mm (Δt_{full}) and 20 mm to 50 mm (Δt_{upper}) from the edge of the shadowing block.

Selected cross-sectional scanning electron microscopy images, taken along the radial sections of representative graded helical films, are shown in Figure 6.5. As expected from [255], the film deposited at 70^o is quite dense, while the film deposited at 85^o is porous and clearly illustrates the helical turns of the film. Figure 6.8 demonstrates that there is substantial film thickness gradation in the radial thickness profiles for the two films. The 85^o deposition angle film is obviously quite porous and could provide an excellent scaffold for a spatially graded liquid crystal application [85].

Using the calibration data, simulated thickness profiles for macroscopically shadowed chiral films were produced. This was done by sweeping the simulated substrate through the respective calibration shadowing profile during a simulated deposition under the same deposition angle and rotation rate conditions. There is good agreement between the experimental and simulated thickness profiles, as demonstrated in Figure 6.8. The thickness gradation for the helical films is therefore predictable and can be simulated by first calibrating the shadowing properties for a particular shadow block.

Close examination of the SEM images shown in Figure 6.5 reveals that there is a tilt to the chiral structure of the deposited film, a consequence of using the shadow block to create the graded film. Far from the shadow block, where the substrate is not shadowed (at d_2 of Figure 6.1), a vertical chiral structure is created. Within the macroscopically shadowed region (at d_1 of Figure 6.1) flux cannot impinge from all directions equally and the chiral structure becomes slightly tilted. The

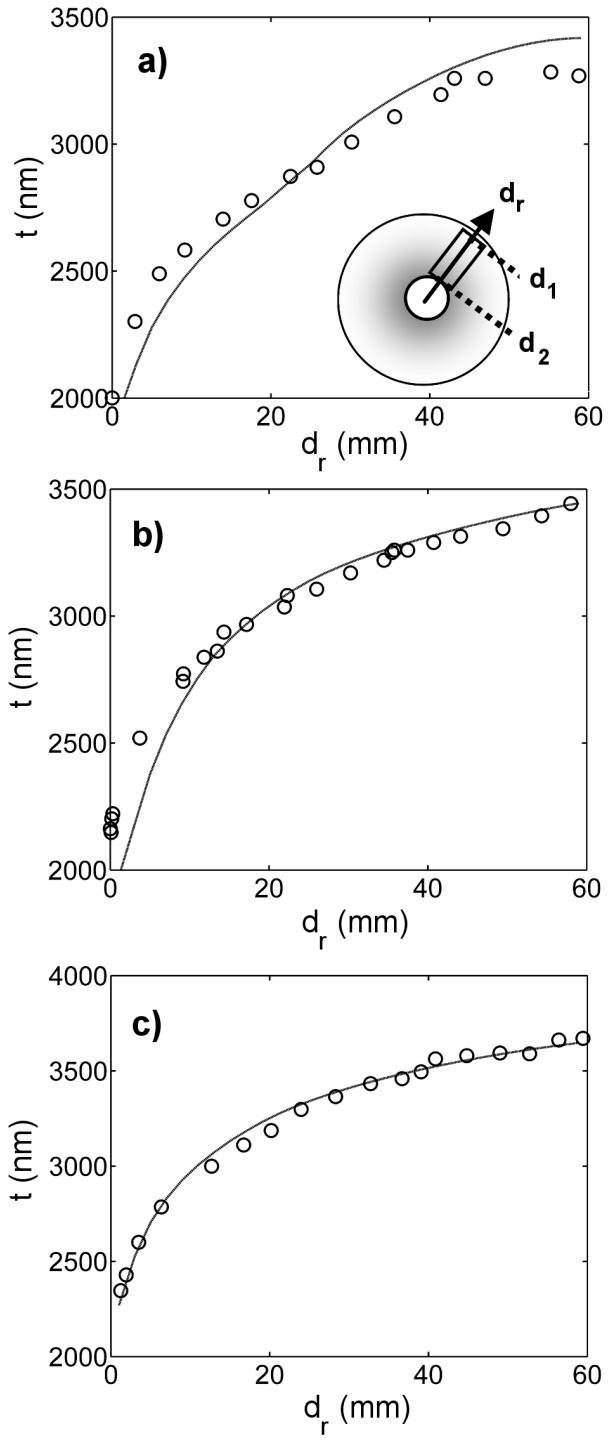


Figure 6.8: Experimentally obtained (circles) graded chiral film thickness profiles at (a) $\alpha = 60^o$, (b) 70^o, and (c) 85^o are compared against the simulated thickness profiles (line). Inset shows measurement direction along sample.

helical columns lean away from the block and toward the direction from which the majority of the flux is incident. The tilt angle closest to the shadowing block edge is approximately 30^o, larger than

the tilt of 7^o which has been observed for substrates placed at the very edge of the deposition chuck [85]. The effect of this tilt is discussed in the next section.

6.6 Optical Characterization of Spatially Graded Helical Films

To measure the optical transmission properties of the graded helical films, a simple optical transmission setup was used. This apparatus is shown in Figure 6.9. White light (Ocean Optics LS-1 Tungsten Halogen lamp) was circularly polarized using a linear polarizer (Thorlabs) set at an angle of 45^o (to produce RCP light) or -45^o (to produce LCP light) followed by a quarter wave plate (Thorlabs). A pinhole aperture minimized the spot size of the light. The transmitted spectral intensity, through the chiral thin film on a fused-silica substrate, was detected by a compact CCD spectrometer (Ocean Optics HR4000). The spatial gradation of each LCP and RCP transmission spectrum was measured using a translation stage, with total travel of 50 mm. Measurements were made in steps of 1 mm. A photograph of the translation stage is shown in Figure 6.10.

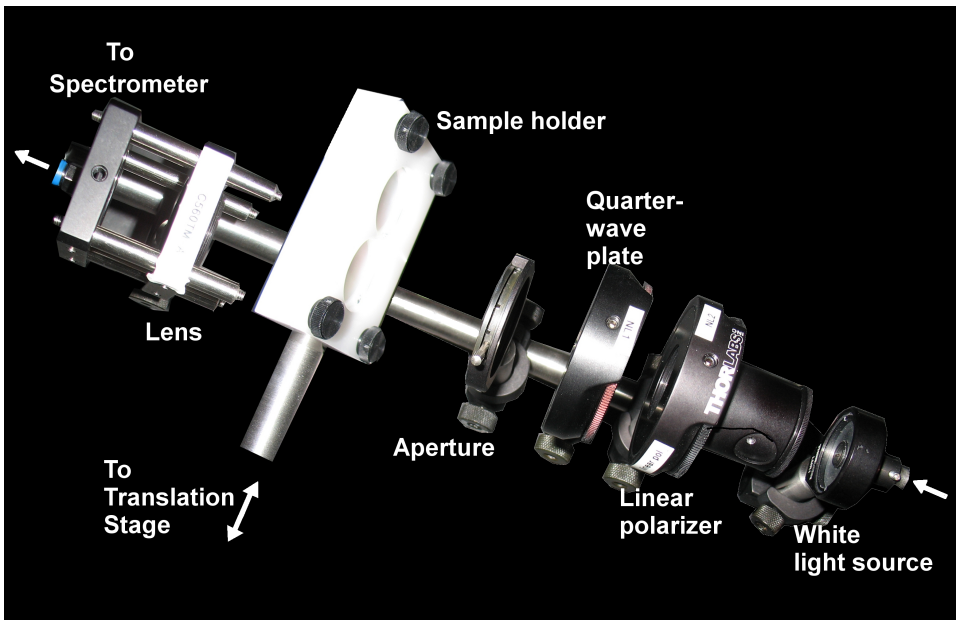


Figure 6.9: Apparatus for optical measurements.

Selective circular polarization transmission and reflection is expected from the inherent birefringence of the helical film [85]. The helical structures fabricated for this work are right-handed and therefore preferentially transmit left-hand circularly polarized (LCP) light (clockwise when viewed from the optical light source) and reflect right-hand circularly polarized (RCP) light. Circular polarized transmission spectra were acquired for both RCP and LCP light and the difference between

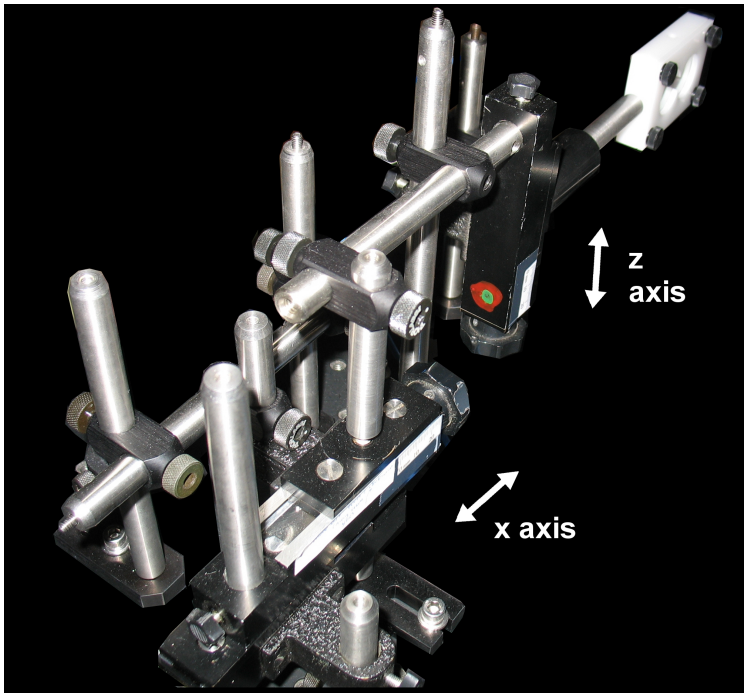


Figure 6.10: Apparatus for translation of sample slide.

these two spectra was calculated in order to determine the polarization selectivity of the film. In order to detect how this selective transmission changed across the graded film a translation stage was employed and multiple measurements were made at set distances along the film from the edge of the shadow block.

We first focus on one of the spatially graded helical films fabricated at a deposition angle of 70^o. As expected from [255] the polarization selectivity of this film is strong. The LCP and RCP transmission spectra (T_{LCP}, T_{RCP}) for the 70^o deposition angle helical film, at a distance of 50 mm from the shadow block edge, are shown in Figure 6.11. For much of the wavelength region under investigation there is overlap between the LCP and RCP spectra. At the location on the substrate shown in Figure 6.11a, there is a region near the optical wavelength of 690 nm that exhibits circular birefringence. The RCP transmission decrease at this wavelength is due to the coupling between the aligned and identical-pitch incident RCP light and the RCP chiral structure. At this wavelength, the LCP light passes through the substrate with little interaction with the film and therefore exhibits high transmission. Some cross-polarization of RCP to LCP light may also result in an increase in the measured high transmission for the LCP light [84]. From Figure 6.11b, the birefringent chiral selectivity between LCP and RCP transmission spectra can be seen to be greater than 25% (or equivalently an extinction ratio of 1:4). While significantly higher selectivity is seen as compared to commercially available circular polarizers (e.g., those used in the photographic industry, where extinction ratios can be as high as 1:4000 [256] for filters based on a combination of linear polarizer

and quarter wave plate), and for thin film polarizers fabricated from more complex multistep photolithography methods (e.g., 1:10 to 1:45 in [257]), this result is as expected, based on previously obtained results for similarly fabricated TiO_2-based chiral GLAD thin films [255]. The selectivity could be improved by using other materials, as has been shown by constructing GLAD chiral nanostructures from the organic light-emitting diode (OLED) material tris(8-hydroxyquinoline) aluminum (Alq_3) with selectivity of greater than 60% [258].

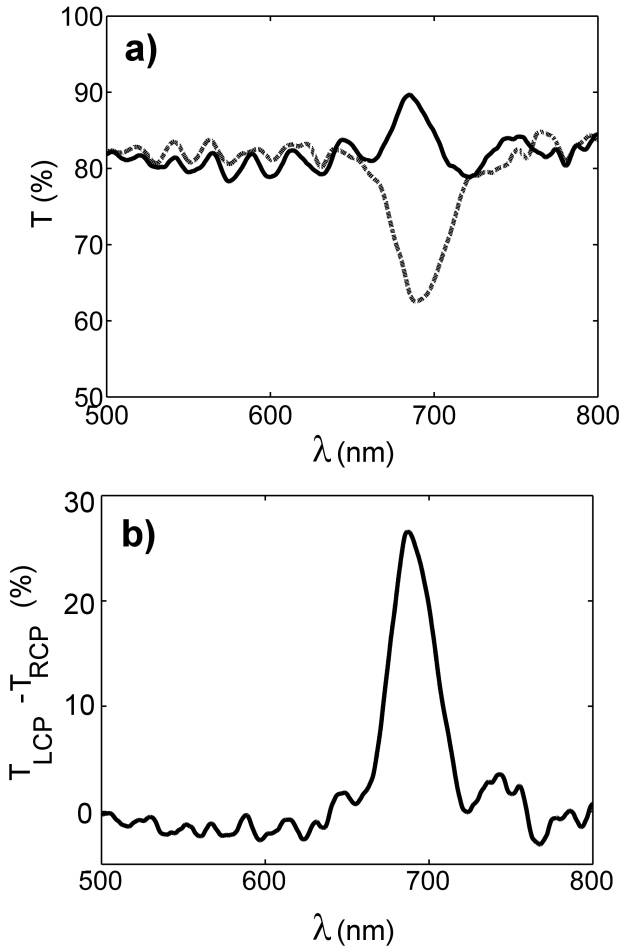


Figure 6.11: (a) LCP (solid-line) and RCP (dashed-line) transmission spectra 50 mm from edge of shadow block. RCP light interacts with the RCP structure, while LCP light is transmitted through the film with less attenuation. (b) Difference in transmission spectra for left (clockwise) minus right (counterclockwise) circularly polarized light (as viewed from the source) at 50 mm from the shadow block edge, for a graded chiral TiO_2 film deposited at a deposition angle of 70^o.

As the thickness of the graded films changes with radial distance from the shadow block, the pitch of the film also changes. This change in pitch allows for a gradation in optical properties across the film length, as illustrated in Figure 6.12 and 6.13. Figure 6.12 shows the color gradation that appears when unpolarized white light is reflected from a graded chiral film. The effect of the

graded multilayer coating pitch is apparent in the color transition from shorter blue to longer red wavelengths as the distance from the edge of the substrate most shadowed by the shadowing block is increased. Figure 6.13 shows the circular polarization selective transmission spectra, for several positions along the radial direction of the 60^o to 80^o helical film. These selective transmission spectra are measured at noted distances from the edge of the shadow block. The wavelength of the LCP-RCP transmission peak is seen to change with changing position along the graded film. The change in the position of this LCP-RCP transmission peak is believed to be primarily due to the change in helical pitch of the film across its length. However, a small portion of the peak shift from low wavelength to high could be due to the increase in symmetry of the multilayer structure [241]. The absolute circular polarization selectivity of the film is also seen to increase along this radial distance. The change in the polarization discrimination is not likely due to the change in thickness or pitch of the film, but may instead be due to the increased alignment of the helical structures with normally incident light and increasing symmetry of the multilayer film, as seen in Figure 6.5.

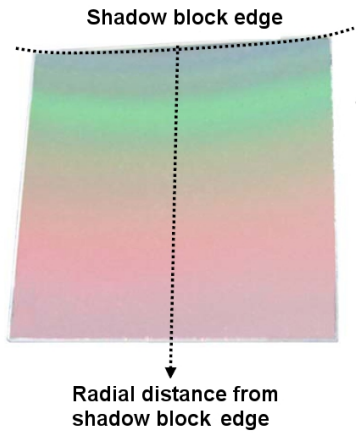


Figure 6.12: Photograph of a graded chiral TiO_2 film deposited at a deposition angle of 70^o on a glass substrate. Incident light in this case is unpolarized.

Figure 6.14 demonstrates the frequency and circular polarization selectivity of the 70^o deposition angle graded chiral film. Figure 6.14a shows the polarization selective Bragg transmission peak location (λ_{peak}) as the radial distance from the shadow block increases. For a distance of 5 mm to 50 mm from the shadowing block edge, the peak shift is 157 nm. The preferential circular polarization transmission values (T_{peak}) are shown in Figure 6.14b, rising from 5% near to the shadowing block to greater than 25% at a distance of 50 mm from the shadowing block edge. The low polarization selectivity near to the shadowing block may be due to a variation from the ideal helical structure; helical column tilt has been seen to decrease from birefringence [255]. However, over a 30 mm wide region 20 mm to 50 mm from the block edge, the filter has a strong selectivity

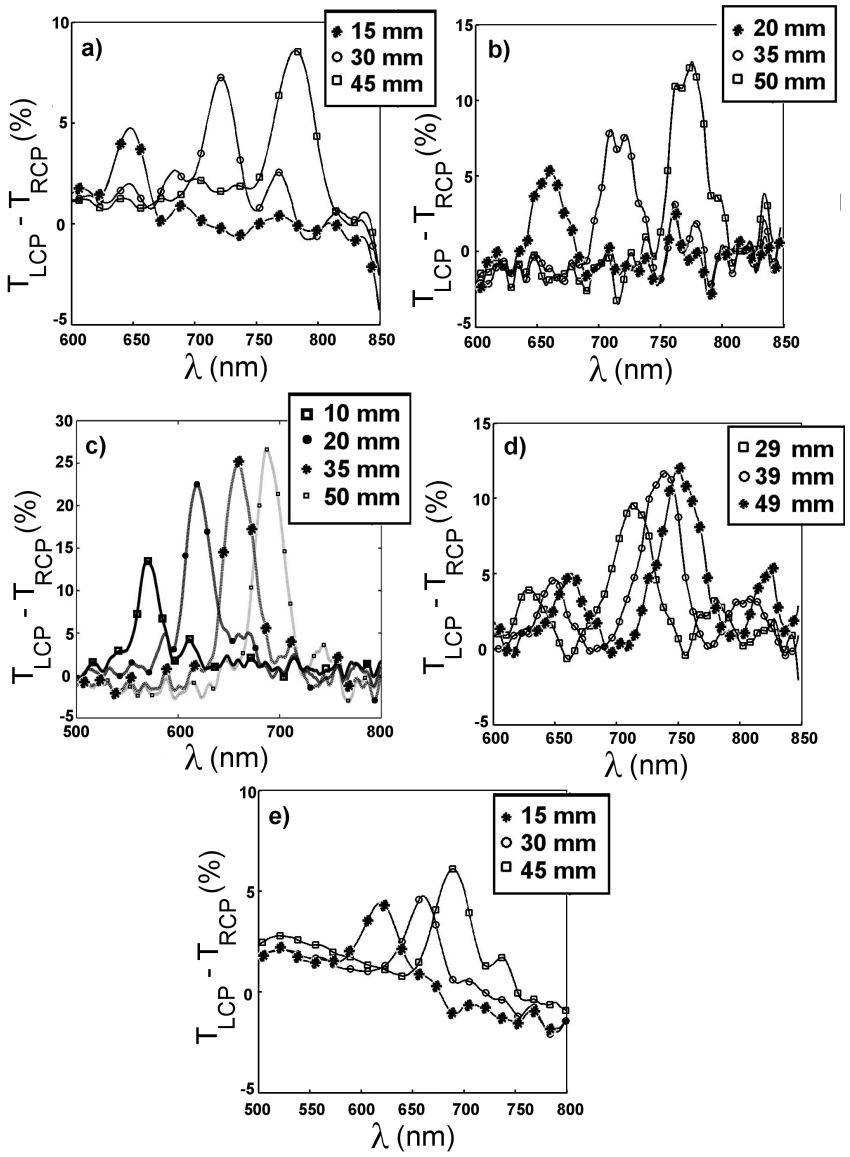


Figure 6.13: Frequency selectivity is demonstrated through LCP-RCP transmission spectrum measurements. Measurements were made at selected distances from the macroscopic shadow block edge, for graded chiral TiO_2 films deposited at deposition angles of (a) 60^o, (b) 65^o, (c) 70^o, (d) 75^o, (e) 80^o.

and significant peak shift of 70 nm.

A comparison of the selective optical performance achieved by each of these spatially graded helical films, deposited at angles from 60^o to 80^o, is shown in Figure 6.15. In each case, the spatially graded films exhibit graded optical response. The peak Bragg wavelength of the LCP-RCP chiral polarization selectivity shifts from its initial point ($\Delta\lambda_{peak}$) as a function of distance across the substrate. Films fabricated at the lower deposition angles of 60^o and 65^o achieve a larger peak location change across a distance of 50 mm. This is likely because the shadow cast by the shadowing

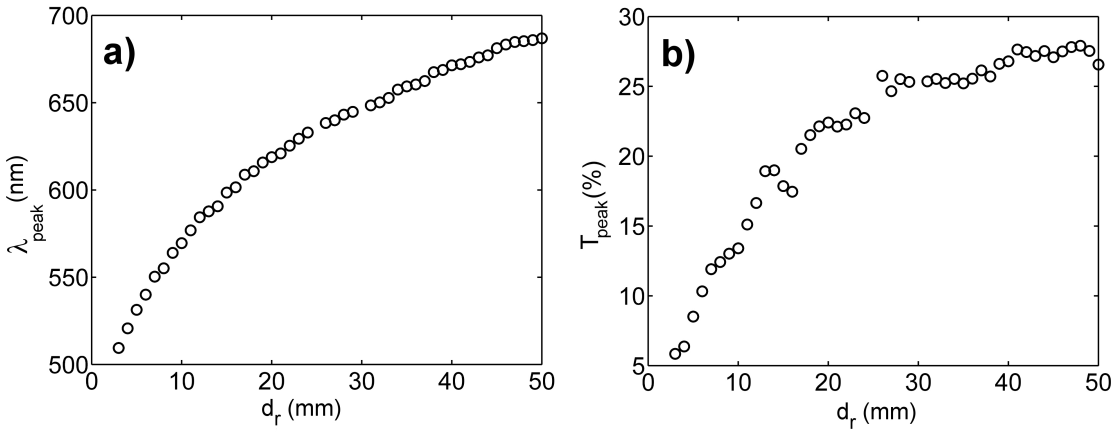


Figure 6.14: Frequency and polarization selectivity for a spatially graded chiral film. (a) Peak location and (b) polarization selective transmission as a function of radial distance from the shadow block edge, for a chiral TiO_2 film deposited at a deposition angle of 70^o.

block is not as long as in the case of the more obliquely angled depositions; the edge of the film is at the penumbral edge of the shadow in this case.

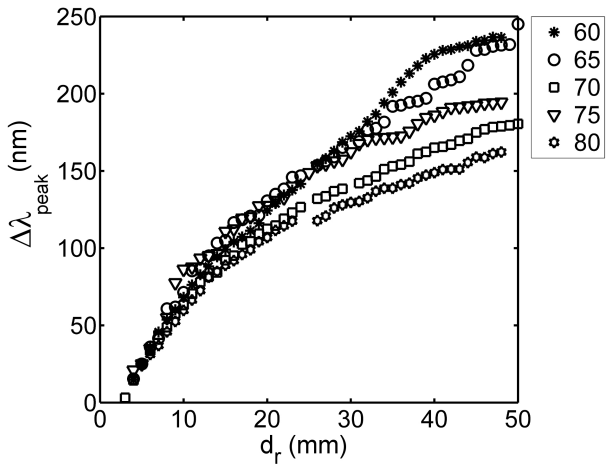


Figure 6.15: Graded chiral LCP-RCP transmission peak location changes as a function of distance from the shadow block edge. Maximum peak location change, over a distance of 50 mm, ranges from 150 nm to 250 nm for films deposited at angles from 60^o to 80^o.

The graded chiral film deposited at a deposition angle of 85^o was not optically analyzed due to its opacity. It has previously been seen that, for helical GLAD films fabricated at oblique deposition angles beyond 80^o, diffuse scattering becomes significant. As the helical structures transition from densely packed to more separated structures, scattering begins to occur. Scattering effects have been seen to increase from negligible amounts at low deposition angles to 8% for a three-turn film, deposited at a deposition angle of 85^o [255]. This diffuse scattering becomes significant for helical

films with greater than five turns, where the circular polarization peak selective transmission begins to level off for films deposited at deposition angles of 85^o [85]. Beyond this number of turns transmission measurements have been found to be difficult due to the opacity of the films.

The chiral polarization selectivity (i.e., the maximum LCP-RCP peak transmission value, T_{peak}), for the films deposited at angles from 60^o to 80^o, is compared in Figure 6.16. For this comparison the selectivity was normalized by the ratio of N_{turns} for the given film to the N_{turns} for the 70^o deposition angle film. The 70^o deposition angle film shows the greatest selectivity. Below this deposition angle the selectivity falls off, due to the fact that the anisotropies of the film decreases [259]. Several effects are predicted to lessen the chiral polarization selectivity of helical films. Above a deposition angle of 70^o the film becomes more porous. According to effective medium theory, it becomes less birefringent due to the increased contribution of air to the average refractive index. Diffuse scattering has also been seen to increase at large oblique deposition angles [255]. Also, increased columnar broadening with film thickness causes the chiral structure to lose definition and thus reduces the chiral selectivity [85].

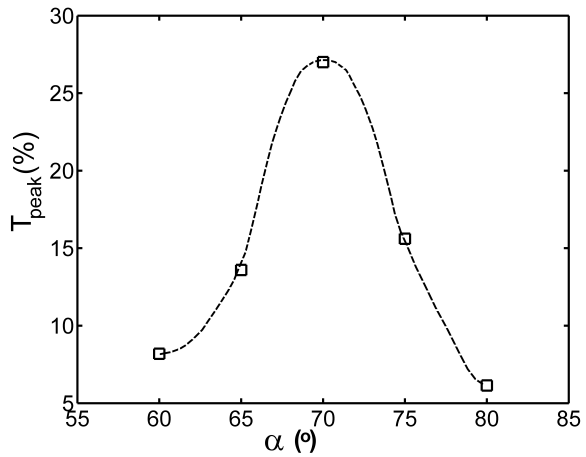


Figure 6.16: Selectivity at 50 mm, for films deposited at angles from 60^o to 80^o (line shown to guide the eye). Selectivity has been normalized by N_{turns} at a deposition angle of 70^o.

The result shown in Figure 6.16 is similar to the result seen in [255] where the maximum selectivity was seen to be for a film fabricated at a deposition angle of 65^o. However, in that case films were fabricated with three turns each, instead of eight. The maximum chiral selectivity exhibited, in the present case (14%), for the 65^o deposition angle film matches well with the simulated result presented in [259] for an eight turn film (15%) and the measured result for an 11 turn film (17%). As mentioned above, in addition to the increase in diffuse scattering for thick films deposited at angles above 70^o, the long shadow cast by the shadow block at these highly oblique deposition angles means that even at a distance of 50 mm from the shadow block edge, there is still shadowing of the

film. This results in lower selectivities for the films deposited at 75^o and 80^o deposition angles.

Using the Bragg relationship (Equation 6.1), the measured pitch of each film, and the chiral polarization selective transmission peak location, the average refractive index at each deposition angle can be estimated. It should be noted that by average refractive index it is meant that the combination of the three primary indices of refraction for the case of normally incident light is used [53]. A plot of refractive index versus deposition angle over a range of distances from the shadow block edge allows for the correlation between thickness and optical performance gradation to be shown. The estimated refractive indices are shown in Figure 6.17. The refractive indices at distances of 5 nm, 20 nm, and 50 mm, from the edge of the shadow block, are plotted. This figure shows that there is a fairly constant refractive index, regardless of location along the graded film. This indicates a constant relationship between thickness and optical selectivity along the spatially graded film. Because the Bragg relationship is maintained across the length of the film, this also indicates that the dominant source of the frequency variation in the polarization selective peak is due to change in helical pitch.

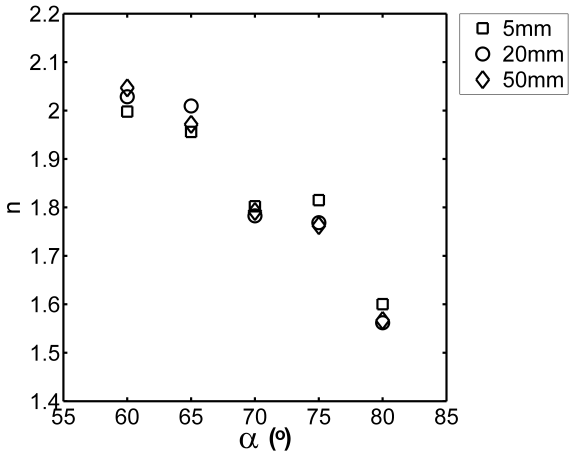


Figure 6.17: Estimated index of refraction at distances of 5 mm, 20 mm, and 50 mm from the edge of the shadow block, for films deposited at deposition angles from 60^o to 80^o.

The refractive indices can also be compared to the expected values for TiO_2 GLAD films. The refractive index of a porous film can be estimated from effective medium theory to be a density weighted average of the refractive indices of air (1) and film material (2.49 for TiO_2 in anatase form) [260]. The density (δ) of the films can be estimated from Tait's rule, as described in Equation 2.11. From this approximation the effective refractive index of film deposited at 60^o and 80^o are expected to be 2.0 and 1.4, respectively. These are in-line with previously obtained results that show that the index of refraction decreases significantly with deposition angle for GLAD films [136]. These values are also quite close to the values shown in Figure 6.17.

6.7 Chapter Summary

By taking advantage of macroscopic shadowing, the GLAD process can be enhanced to create spatially graded chiral nanostructured thin films. Graded helical films, fabricated as a test subject for this new process, were manufactured from transparent TiO_2 and as such were optically multifunctional. Because of the thickness gradation, films exhibited both polarization and frequency selectivity. The helical structures had a graded pitch which allowed for frequency tuning by varying the spatial position of the incident light. For the film deposited at a deposition angle of 70^o, strong circular polarization selectivity occurred over a spatial range of 30 mm, concurrent with a spectral tunability from approximately 620 nm to 690 nm in terms of wavelength. Films deposited at angles from 60^o to 80^o, with lesser polarization selectivity, but similar frequency tunability. As with nature, where nanostructured coloration is often used for signaling, the spatially graded structures could be used as components in optical communication systems.

Potential applications beyond those described above include position and velocity sensors, where light transmitted through the chiral thin film would reveal motion through changes in the received selective polarization peak. It may also be possible to adapt the macroscopic shadowing technique to fabricate structures such as spatially graded linear birefringent thin films. Graded films with spectral hole defects are also possible, for use in tunable optical components where polarization properties may not be as critical. Finally, these graded nanostructured films may also be of use in liquid crystal and microfluidic applications requiring structural support that has a spatial gradation. In the case of microfluidic devices, the graded helical support might allow for interesting new separation techniques which exploit the spatially varying pitch and density of the graded nanostructure.

Chapter 7

Summary and Conclusions

7.1 Summary of thesis work

In this thesis, the properties of nanostructured thin films, fabricated using the glancing angle deposition method, were probed and manipulated over a range of size scales. The structural characteristics were investigated and quantified using gas adsorption porosimetry, focused ion beam tomography, optical methods, and SEM image analysis. The manipulation of these structural characteristics has been carried out using a novel shadowing technique.

Fundamental properties of the samples fabricated for this thesis were first determined. Metal oxide nanostructures at several thicknesses, deposition angles, and geometries were manufactured. Using measurements of the mass, footprint area and thickness of these nanostructure samples, their density was determined. To compensate for the humidity present in the air during mass measurements, the adsorbed water was estimated using optical measurements and an effective medium approximation. The densities of vertical post films, deposited from 0.5 μm to 5 μm at oblique angles from 70^o to 85^o, were found to be constant as a function of thickness. As a function of deposition angle, the densities of vertical post films decreased with decreasing deposition angle, following the trend predicted by Poxson [148]. Poxson fitting parameters of 5.10 ± 0.38, 2.60 ± 0.11 and 8.70 ± 0.85 were found for TiO_2, SiO_2, and ITO. The fit parameter of the Poxson equation applied to the ITO density scaling was high. This was likely due to differences in deposition conditions between the work presented here and that described by Poxson. Helical films were found to follow a density profile with a Poxson fit of 5.44 ± 0.41, slightly higher than that of the vertical post films. Investigation by techniques such as dual beam FIB-SEM milling and imaging may help to elucidate the differences in morphology between these two structures and determine why differences in parameters such as density occur.

Having measured the basic properties of the metal oxide nanostructured films, their surface area and pore size distribution could be found. It had long been clear that GLAD films would have high surface area, since SEMs reveal high aspect ratio columnar structures with bifurcation, branch-

ing, and roughness common for many materials. Gas adsorption onto the studied films allowed for quantification of the surface area parameter, as well as the mesoporosity of the films. Using high sensitivity krypton gas, which enabled surface characterization of the low total surface area film samples, sorption isotherms were measured at 77 K and 87 K for the nanostructured films for deposition angles from 0^o to 85^o. The isotherms for the ITO films were quite different from those of the TiO_2 and SiO_2 films. This is likely partially due to the difference in crystallinity between the two sample sets (the ITO films were polycrystalline whereas the TiO_2 and SiO_2 were amorphous), and also because of morphological characteristics. The isotherms for the low deposition angle, dense and non-columnar TiO_2 and SiO_2 films are typical Type I isotherms, with no hysteresis. As deposition angle increased the isotherms evolved both hysteresis and a high pressure rise in adsorbed volume which did not reach a steady-state level for the most obliquely deposited films. This mirrored the evolution of nanostructure from a densely packed conglomerate of nanorods at moderate deposition angles to the more widely arrayed columns characteristic of GLAD films. In contrast, even low deposition angle ITO films are columnar and exhibit hysteresis from low deposition angles onward.

The isotherms captured at 77 K were assessed using BET theory in the pressure range from 0.05 to 0.3 of the vapor pressure. Plots of relative surface areas versus deposition angle show an interesting and somewhat unexpected feature of GLAD films: the highest surface area is achieved at deposition angles far below the oblique angles typical of GLAD depositions. This may be because the column density is higher at lower deposition angles. For SiO_2 and TiO_2, the highest surface area was reached at $\alpha = 60^o$, with surface area enhancements of nearly 900 $m^2\ m^{-2}\ \mu m^{-1}$ ($700 \pm 150\ m^2 g^{-1}$ and $325 \pm 40\ m^2 g^{-1}$, respectively). Similarly, the peak ITO surface area was in the range of $\alpha = 60^o$ to 70^o, at approximately 200 $m^2\ m^{-2}\ \mu m^{-1}$ (50 $m^2 g^{-1}$). These values, high for nanostructured materials, make GLAD promising for applications requiring large surface interfaces.

The impacts of deposition conditions on surface properties were determined. As one might expect, relative surface areas increase linearly with thickness. Interestingly however, multilayer structures consisting of stacks of different deposition angles did not reach the surface areas expected. Surface areas were within calculated uncertainties, but significantly less than those predicted from a weighted average of single α vertical post films. This was an interesting finding, given that the densities of the multilayer films were as predicted. This suggests that the peak exterior columnar roughness, achieved at the mid-range deposition angles (e.g., $\alpha = 60^o$), is not present in the multilayer films. In contrast, the helical film surface areas did reach roughly the same peak values as those of the vertical post films, although the trend with respect to deposition angle was different. This suggests subtle differences in adatom diffusion during deposition of the two structures. Again, FIB-SEM imaging or TEM analysis of a range of structures, deposited at different rotation rates and deposition angles, may reveal the reason for differences in surface characteristics between films of dissimilar morphologies.

The surface area having been measured for selected GLAD films, a second related characteristic was determined. The pore size distributions of the vertical post TiO_2 and SiO_2 films were calculated from the 87 K krypton gas adsorption isotherms. Revealing details about roughness and porosity, these distributions were calculated using standard classical BJH analysis, with a subsequent empirical calibration based on measurement performed on ordered mesoporous molecular materials. The GLAD nanostructured films exhibited broad pore distributions, with peaks in the low mesoporous regime of 2 nm to 5 nm. The peak pore diameters of 4 nm for SiO_2 and 5 nm for TiO_2 were reached at the mid-range deposition angles of 60^o to 70^o, the same location as the peak in surface area curves. This is an interesting finding and suggests that although films deposited at these angles appear to be quite dense, they are in fact quite porous. Indeed, analysis of top-down SEM images of these same GLAD films reveals detectable columnar structure down to 65^o. These findings hint at columnar structure that extends down to at least 60^o.

Focused ion beam milling is a relatively new technique which was employed to investigate the structure of GLAD films in the tens of nanometers regime not measurable by krypton gas adsorption porosimetry. The dual beam FIB-SEM allowed for a "slice and view" analysis to be carried out on a 5 μm TiO_2 nanocolumnar film. The sample volume was sliced by the FIB into several hundred cross-sections, imaged concurrently by the SEM. Post-processing by custom equalization, alignment, and grain detection algorithms revealed details about the growth of the GLAD sample under investigation. Column diameters scaled with thickness as $\overline{w}_{act}$ = (9.4 $\pm$ 3.0) $t^{0.35\pm0.09}$ [nm]. Similarly the column center to center spacing and mean void spacing between columns grew as $\overline{c}_{act}$ = (24.8 $\pm$ 5.2) $t^{0.31\pm0.08}$ [nm] and $\overline{v}_{act}$ = (15.2 $\pm$ 3.8) $t^{0.25\pm0.06}$ [nm]. The density of columns per unit area decreased with thickness, following a trend of d_{act} = (3400 $\pm$ 2500) $t^{-0.65\pm0.15}$ [columns μm^{-2}]. These trends agree with previous findings, but were determined using only one sample, instead of multiple samples of increasing thickness. In addition to probing nanostructure growth properties, the first three dimensional reconstructions of a GLAD film and of individual columns were carried out. The competition of neighboring columns was therefore observable for the first time, with the column with the most branches appearing to out-compete its neighbors. Analysis of aggregate film parameters, such as column density and mean column diameter enabled an assessment of surface area evolution as a function of film growth. By modeling columns as smooth, the surface area growth was predicted to be sublinear. This was at odds with the linear trend seen from gas adsorption measurements, indicating that features such as column roughness and bifurcation contribute significantly to the surface area of metal oxide (and other material) GLAD films.

A technique to modulate the properties of GLAD films was next described. By harnessing the very tool (shadowing) that enables GLAD spatially graded nanostructures were created. By using a special shadow block in the center of the deposition chuck the thickness of the deposited film could be varied. As a test of this method optically transparent graded helical films were fabricated.

Because of the graded pitch of the chiral structures, the films exhibited both polarization and frequency selectivity. Structures with moderate to strong circular polarization selectivity occurred over a spatial range of 30 mm, concurrent with frequency tunability over a wavelength range of 620 nm to 690 nm. These will potentially be useful in applications such as optical filters where tunable frequency properties may be useful or microfluidic devices where the graded helical supports might allow for interesting new separation techniques.

To summarize, the conclusions of this work are listed below.

- Fundamental properties, including density and estimated water adsorption, were found for the studied GLAD films.
- High-sensitivity krypton gas adsorption was used to measure surface areas and pore size distributions of GLAD metal-oxide nanostructures.
- The surface areas of the amorphous TiO_2 and SiO_2 GLAD films were found to be higher than those of the polycrystalline ITO films.
- Peak surface areas occurred in the mid-range of deposition angles, not at the most oblique deposition angles that normally characterize GLAD thin films.
- Despite columnar broadening as film thickness increases, the surface area of GLAD films rises linearly with thickness. This implies a constant differential surface area throughout the film.
- GLAD surface area can be tuned not only through choice of deposition angle and film thickness, but also from rotation characteristics.
- Inter- and intra-column porosity that extends to the nanometer range, for GLAD structures deposited from 45^o to the most oblique angles.
- A focused ion beam milling procedure for segmenting GLAD films was demonstrated. Additionally, a series of image post-processing algorithms were implemented to reconstruct and analyze a GLAD film.
- Using a single sample, GLAD column diameter, spacing, and density were found to follow power-law scaling relationships.
- Individual GLAD columns were tracked using the FIB milling data. It was found that individual GLAD columns with large numbers of branches appear to survive longer than their neighbors.
- The first graded thickness helical GLAD nanostructures were constructed using a novel macroscopic shadowing technique.

- These graded thickness structures were used as frequency-tunable polarization-selective photonic filters.

7.2 Future Directions

There is great potential for several avenues of research following the findings given in this thesis. The diverse set of nanostructures that can be created using the glancing angle deposition technique means that many years of further characterization and modification are possible by applying the processes described in this work. For example, the surface area characterization of vertical posts, helical films, and multilayer stacks should now be extended to slanted post arrays, serial bi-deposited (SBD) columns, and even three-dimensional square photonic crystal structures. Krypton adsorption at 87 K also shows promise and porosimetry on metals, organics, and hybrid materials would be an interesting future area of study.

The dual-beam FIB-SEM tomography technique developed for GLAD films could now be applied to not only vertical post structures, but also helical, slanted post, multilayer, and multimaterial columnar arrays. Assessment of the full three dimensional structure of helical films will help to confirm properties such as birefringence, cross-sectional geometry, helical pitch as a function of deposition angle, and growth properties. A complement to the measurements made using gas adsorption porosimetry, the porosity and surface characteristics of multilayer nanostructures could be verified with three-dimensional reconstruction. Understanding the growth of multi-material films have also been of interest in the GLAD community, particularly for energy applications requiring frequency selective light scavenging and electron/hole transport layers.

The focused ion beam milling technique, proven for analysis of a large GLAD volume, should now be extended such that more detailed analysis of isolated individual and clusters of columns can be carried out. Having been proven in other applications, a technique to mill a small lamella from a larger sample volume for use in TEM studies is therefore now under development. Used to assess obliquely deposited films of not only glancing angles greater than 80^o, this technique could help to confirm the nature of porosity in the denser lower deposition angle films studied in the porosimetry section of this thesis.

Future directions for the spatially graded GLAD work are already being undertaken [261]. Graded thickness ultrathin layer chromatography (UTLC) plates have been fabricated using the shadowed GLAD technique. These produce anisotropic films with liquid separation properties that

are dependent on the radial distance from the shadow block. This technique shows promise for mass-identification of multiple liquid samples.

7.3 Final remarks

Conceived within the last twenty years, the application of the glancing angle deposition technique has exploded to include over one hundred research groups, each probing, expanding, and utilizing the method for a range of applications and products. The fundamental characterization presented here of GLAD over a range of size scales, accompanied by new methods for modulating the properties of nanostructures fabricated with this technique, will therefore be valuable to a large research community and should provide essential information for those seeking to optimize their results and achieve the best device performance possible.

Bibliography

[1] G.S. Watson, B.W. Cribb, J.A. Watson. How micro/nanoarchitecture faciliates anti-wetting: An elegant hierarchical design on the termite wing. *ACS Nano*, 4:129 – 136, 2010.

[2] K. Kertesz, Z. Balint, Z. Vertesy, G.I. Mark, V. Lousse, J.-P. Vigneron, L.P. Biro. Photonic crystal type structures of biological origin: Structural and spectral characterization. *Curr. Appl. Phys.*, 6:252 – 258, 2006.

[3] T. Sun, L. Feng, X. Gao, L. Jiang. Bioinspired surfaces with special wettability. *Acc. Chem. Res.*, 38:644 – 652, 2005.

[4] H. Lee, B.P. Lee, P.B. Messersmith. A reversible wet/dry adhesive inspired by mussels and geckos. *Nature*, 448:338 – 342, 2007.

[5] R. Yan, D. Gargas, P. Yang. Nanowire photonics. *Nat. Photonics*, 3:569 – 576, 2009.

[6] V.Y. Prinz, V.A. Seleznev, A.V. Prinz, A.V. Kopylov. 3D heterostructures and systems for novel MEMS/NEMS. *Sci. Tech. Adv. Mater.*, 10:034502, 2009.

[7] N.L. Rosi, C.A. Mirkin. Nanostructures in biodiagnostics. *Chem. Rev.*, 105:1547 – 1562, 2005.

[8] J. Wang. Carbon-nanotube based electrochemical biosensors: A review. *Electroanalysis*, 17:7 – 14, 2005.

[9] J.R. Heflin Jr M. Di Ventra, S. Evoy. *Introduction to nanoscale science and technology*. Kluwer Academic Publishers, Massachusettes, 2004.

[10] M.D.B. Charlton, G.J Parker. Fabrication of high aspect ratio silicon microstructures by anodic etching. *J. Micromech. Microeng.*, 7:155 – 158, 1997.

[11] W. Li, W. Zhao, P. Sun. Fabrication of highly ordered metallic arraysand silicon pillars with controllable size using nanosphere lithography. *Physica E*, 41:1600 – 1603, 2009.

[12] H. Schift. Nanoimprint lithography: An old story in modern times? A review. *J. Vac. Sci. Technol. B*, 26:458 – 480, 2008.

[13] C.J. Brinker, Y. Lu, A. Sellinger, H. Fan. Evaporation-induced self-assembly: Nanostructures made easy. *Adv. Mat.*, 11:579 – 585, 1999.

[14] L. Liao, J.C. Li, D.H. Liu, C. Liu, D.F. Wang, W.Z. Song, Q. Fu. Self-assembly of aligned ZnO nanoscrews: Growth, configuration, and field emission. *Appl. Phys. Lett.*, 86:083106, 2005.

[15] D. Barreca, A. Gasparotto, C. Maccato, C. Maragno, E. Tondello, E. Cornini, G. Sberveglieri. Columnar CeO2 nanostructures for sensor applications. *Nanotechnology*, 18:125502, 2007.

[16] S. Fan, M.G. Chapline, N.R. Franklin, T.W. Tornbler, A.M. Cassell, H. Dai. Self-oriented regular arrays of carbon nanotubes and their field emission properties. *Science*, 283:512 – 514, 1999.

[17] K. Reichelt, X. Jiang. The preparation of thin films by physical vapour deposition methods. *Thin Solid Films*, 191:91 – 126, 1990.

[18] R. Thieslsch, A. Gatto, J. Heber, N. Kaiser. A comparitive study of the UV optical and structural properties of SiO_2, Al_2O_2, and HfO_2 single layers deposited by reactive evaporation, ion-assisted deposition and plasma ion-assisted deposition. *Thin Solid Films*, 410:86 – 93, 2002.

[19] B.A. Movchan, A.V. Demchistin. Study of the structure and properties of thick vacuum condensates of nickel, titanium, tungsten, aluminium oxide and zirconium dioxide. *Phys. Metals. Metallogr.*, 28:83 – 90, 1969.

[20] J.A. Thornton. High rate thick film growth. *Ann. Rev. Mater. Sci.*, 7:239 – 260, 1977.

[21] L. Reimer. *Scanning electron microscopy: Physics of image formation and microanalysis, 2nd ed.* Springer, Berlin, 1998.

[22] J. Goldstein, et al. *Scanning electron microscopy and x-ray microanalysis, 3rd ed.* Kluwer Academic/Plenum Publishers, New York, 2003.

[23] O.C. Wells, D.C. Joy. The early history and future of the SEM. *Surf. Interface Anal.*, 28:1738 – 1742, 2006.

[24] T.E. Everhart, R.F.M. Thornley. Wide-band detector for micro-microampere low-energy electron currents. *J. Sci. Instrum.*, 37:246 – 248, 1960.

[25] M. Troyon, H.N. Lei, A. Bourhettar. Integration of an STM in an SEM. *Ultramicroscopy*, 42-44:1564 – 1568, 1992.

[26] J.J. Pagano, T. Bansagi Jr, O. Steinbock. Bubble-templated and flow-controlled synthesis of macroscopic silica tubes supporting zinc oxide nanostructures. *Angew. Chem. Int. Ed.*, 47:9900 – 9903, 2008.

[27] N. Yao. *Focused ion beam systems*. Cambridge University Press, Cambridge, 20079.

[28] H. Kohl L. Reimer. *Transmission electron microscopy: Physics of image formation, 5th ed.* Springer, New York, 2008.

[29] M. Weyland, P.A. Midgley. Electron tomography. *Mater. Today*, 7:32 – 40, 2004.

[30] C. Kubel, A. Voigt, R. Schoenmakers, M. Otten, D. Su, T.-C. Lee, A. Carlsson, J. Bradley. Recent advances in electron tomography: TEM and HAADF-STEM tomography for materials science and semiconductor applications. *Micro. Microanal.*, 11:378 – 400, 2005.

[31] G. Binnig, C.F. Quate, Ch. Gerber. Atomic force microscope. *Phys. Rev. Lett.*, 56:930 – 933, 1986.

[32] R.W. Carpick, M. Salmeron. Scratching the surface: Fundamental investigations of tribology with atomic force microscopy. *Chem. Rev.*, 97:1163 – 1194, 1997.

[33] F.J. Giessibl. Advances in atomic force microscopy. *Rev. Mod. Phys.*, 75:949 – 983, 2003.

[34] W.K. Roentgen. On a new kind of ray (translated from German). *Nature*, 53:274 – 276, 1896.

[35] P.P. Ewald. *Fifty years of x-ray diffraction*. International Union of Crystallography, 1999.

[36] M. von Laue. Concerning the detection of x-ray interference. *Nobel Lecture*, pages 347 – 355, 1915.

[37] D.I. Svergun, M.H.J. Koch. Advances in structure analysis using small-angle scattering in solution. *Curr. Opin. Struc. Biol.*, 90:5599 – 5605, 2001.

[38] K. Sing F. Rouquerol, J. Rouquerol. *Adsorption by powders and porous solids: Principles, methodologies and applications*. Academic Press, San Diego, 1999.

[39] M.C.S. Pouillet. Discussion of a new heat production phenomena (in French). *Ann. Chim. Phys.*, 20:141, 1822.

[40] G. Gore. Changes of temperature caused by contact of liquids with powdered silica. *Phil. Mag.*, 37:306 – 346, 1894.

[41] L.G. Gurvich. On the physicochemical attracting force. *J. Russ. Phys. Chim.*, 47:805 – 827, 1915.

[42] A. Zsigmondy. On the structure of silica. Theory of adsorption (in German). *Z. Anorg. Chem.*, 71:356 – 377, 1911.

[43] R.A. Hodgkin E.P. Wigner. *Michael Polanyi, 12 March 1891- 22 February 1976 in Biographical memoirs of fellows in the Royal Society, Vol. 23*. The Royal Society, London, 1977.

[44] H. Talyor. *Irving Langmuir, 1881-1957 in Biographical memoirs of fellows in the Royal Society, Vol. 4*. The Royal Society, London, 1958.

[45] S. Brunauer, P.H. Emmett, E. Teller. Adsorption of gases in multimolecular layers. *J. Am. Chem. Soc.*, 60:309 – 319, 1938.

[46] E.P. Barrett, L.G. Joyner, P.P. Halenda. The determination of pore volume and area distribution in porous substances. I. Computation from nitrogen isotherms. *J. Am. Chem. Soc.*, 62:373 – 380, 1951.

[47] X. Song, Z. Wang, Y. Liu, C. Wang, L. Li. A highly sensitive ethanol sensor based on mesoporous ZnO-SnO_2 nanofibers. *Nanotechnology*, 20:075501, 2009.

[48] M.-H. Seo, M. Yuasa, T. Kida, J.-S. Hug, K. Shimanoe, N. Yamazoe. Gas sensing characteristics and porosity control of nanostructured films composed of TiO_2 nanotubes. *Sens. Act. B*, 137:513 – 520, 2009.

[49] N. Kunzner, J. Diener, E. Gross, D. Kovalev, V. Yu. Timoshenko, M. Fujii. Form birefringence of anisotropically nanostructured silicon. *Phys. Rev. B*, 71:195304, 2005.

[50] D. Kovalev, G. Polisski, J. Diener, H. Heckler, N. Kunzner, V. Yu. Timoshenko. Strong in-plane birefringence of spatially nanostructured silicon. *Appl. Phys. Lett.*, 78:1343476, 2001.

[51] K. Busch, G. von Freymann, S. Linden, S.F. Mingaleev, L. Tkeshelashvili, M. Wegener. Periodic nanostructures for photonics. *Phys. Rep.*, 444:101 – 202, 2007.

[52] S. Colodrero, M. Ocana, H. Miguez. Nanoparticle-based one-dimensional photonic crystals. *Langmuir*, 24:4430 – 4434, 2008.

[53] I.J. Hodgkinson, Q.H. Wu, S. Collett. Dispersion equations for vacuum-deposited tilted-columnar biaxial media. *Appl. Opt.*, 40:452 – 457, 2001.

[54] K. Robbie, D.J. Broer, M.J. Brett. Chiral nematic order in liquid crystals imposed by an engineered inorganic nanostructure. *Nature*, 399:764 – 766, 1999.

[55] S. Wang, G. Xia, H. He, K. Yi, J. Shao, Z. Fan. Structural and optical properties of nanostructured TiO_2 thin films fabricated by glancing angle deposition. *J. Alloys Comp.*, 431:287 – 291, 2007.

[56] K. Kaminska, T. Brown, G. Beydaghyan, K. Robbie. Vacuum evaporated porous silicon photonic interference filters. *Appl. Opt.*, 20:4212 – 4219, 2003.

[57] J.S. Ames. *Prismatic and diffraction spectra: Memoirs by Joseph von Fraunhofer*. Harper and Brothers Publishers, New York, 1898.

[58] G. Kirchhoff, R. Bunsen. Chemical analysis by observation of spectra. *Annalen der Physik und der Chemie*, 110:161 – 189, 1860.

[59] W.L. Wolfe. *Introduction to imaging spectrometers*. SPIE, Bellingham, Washington, 1997.

[60] J.P. Coates. New microspectrometers. *Spectroscopy*, 15:21 – 27, 2000.

[61] C. M. Herzinger, B. Johs, W. A. McGahan, J. A. Woollam, W. Paulson. Ellipsometric determination of optical constants for silicon and thermally grown silicon dioxide via a multi-sample, multi-wavelength, multi-angle investigation. *J. Appl. Phys.*, 83:3323, 1998.

[62] J. Gospodyn, M.J. Brett, J.C. Sit. Characterization by variable angle spectroscopic ellipsometry of dielectric columnar thin films produced by glancing angle deposition. *Proc. Mat. Res. Soc.*, 797:W5 19.1 – 6, 2004.

[63] K. Kaminska, A. Amassian, L. Martinu, K. Robbie. Growth of vacuum evaporated ultra-porous silicon studied with spectroscopic ellipsometry and scanning electron microscopy. *J. Appl. Phys.*, 97:013511, 2005.

[64] A.C. van Popta, J. Cheng, J.C. Sit, M.J. Brett. Birefringence enhancement in annealed TiO_2 thin films. *J. Appl. Phys.*, 102:013517, 2007.

[65] J.-Q. Xi, M.F. Schubert, J.K. Kim, E.F. Schubert, M. Chen, S.-Y. Liu, W. Liu, J.A. Smart. Optical thin-film materials with low refractive index for broadband elimination of Fresnel reflection. *Nat. Photonics*, 1:176 – 179, 2007.

[66] C.H. Chang, P. Yu, C.S. Yang. Broadband and omnidirectional antireflection from conductive indium-tin-oxide nanocolumns prepared by glancing-angle deposition with nitrogen. *Appl. Phys. Lett.*, 94:051114, 2009.

[67] A. Kundt. On the double refraction of light in metal layers which have been prepared by cathode sputtering (in German). *Ann. Phys. Chem.*, 27:59 – 71, 1886.

[68] H. Konig, G. Helwig. The structure of obliquely evaporated films and their influence on the formation of submicroscopic surface irregularities (in German). *Optik.*, 6:111 – 124, 1950.

[69] L. Holland. The effects of vapour incidence on the structure of evaporated aluminum films. *J. Opt. Soc. Am.*, 43:376 – 380, 1953.

[70] J.M. Nieuwenhuizen, H.B. Haanstra. Microfractography of thin films. *Philips Tech. Rev.*, 27:87 – 91, 1966.

[71] K. Robbie, M.J. Brett, A. Lakhtakia. Chiral sculptured thin films. *Nature*, 384:616, 1996.

[72] T. Karabacak, G.-C. Wang, T.-M. Lu. Quasi-periodic nanostructures grown by oblique angle deposition. *J. Appl. Phys.*, 94:7723, 2003.

[73] A.-L. Barabasi, H.E. Stanley. *Fractal concepts in surface growth*. Cambridge University Press, Cambridge, 1995.

[74] M. Malac, R.F. Egerton, M.J. Brett, B. Dick. Fabrication of submicrometer arrays of pillars and helices. *J. Vac. Sci. Technol. B.*, 17:2671, 1999.

[75] M.J. Brett M.O. Jensen. Functional pattern engineering in glancing angle deposition thin films. *J. Nanoscience Nanotech.*, 5:723 – 728, 2005.

[76] M.A. Summers, M.J. Brett. Optimization of periodic column growth in glancing angle deposition for photonic crystal fabrication. *Nanotechnology*, 19:415203, 2008.

[77] S.V. Kesapragada, D. Gall. Two-component nanopillar arrays grown by glancing angle deposition. *Thin Solid Films*, 494:234 – 239, 2005.

[78] M.O. Jensen, M.J. Brett. Porosity engineering in glancing angle deposition thin films. *Appl. Phys. A.*, 80:763 – 768, 2005.

[79] S.R. Kennedy, M.J. Brett. Porous broadband antireflection coating by glancing angle deposition. *Appl. Optics*, 42:4573 – 4579, 2003.

[80] I.J. Hodgkinson, Q.H. Wu. Serial bideposition of anisotropic thin films with enhanced linear birefringence. *Appl. Opt.*, 38:3621 – 3625, 1999.

[81] M.F. Schubert, J.-Q. Xi, J.K. Kim, E.F. Schubert. Distributed bragg reflector consisting of high- and low-refractive-index thin film layers made of the same material. *Appl. Phys. Lett.*, 90:2720269, 2007.

[82] A.C. van Popta, M.M. Hawkeye, J.C. Sit, M.J. Brett. Gradient-index narrow-bandpass filter fabricated with glancing-angle deposition. *Opt. Lett.*, 29:2545 – 2547, 2004.

[83] Q. Wu, I.J. Hodgkinson, A. Lakhtakia. Circular polarization filters made of chiral sculptured thin films: experimental and simulation results. *Opt. Eng.*, 39:3140 – 3149, 2000.

[84] I.J. Hodgkinson, Q.H. Wu, M. Arnold, M.W. McCall, A. Lakhtakia. Chiral mirror and optical resonator designs for circularly polarized light: suppression of cross-polarized reflectances and transmittances. *Opt. Comm.*, 210:201 – 211, 2002.

[85] A.C. van Popta, J.C. Sit, M.J. Brett. Optical properties of porous helical thin films. *Appl. Opt.*, 43:3632 – 3639, 2004.

[86] P. Yu, C.-H. Chang, M.-S. Su, M.-H. Hsu, K.-H. Wei. Embedded indium-tin-oxide nanoelectrodes for efficiency and lifetime enhancement of polymer-based solar cells. *Appl. Phys. Lett*, 96:153307, 2010.

[87] H.-Y. Yang, M.-F. Less, C.-H. Huang, Y.-S. Lo, Y.-J. Chen, M.-S. Wong. Glancing angle deposited titania films for dye-sensitized solar cells. *Thin Solid Films*, 518:1590 – 1594, 2009.

[88] A. Bonakdarpour, M.D. Fleischauer, M.J. Brett, J.R. Dahn. Columnar support structures for oxygen reduction electrocatalysts prepared by glancing angle deposition. *Appl. Catal. A*, 349:110 – 115, 2008.

[89] M.D. Fleischauer, J. Li, M.J. Brett. Columnar thin films for three-dimensional microbatteries. *J. Electrochem. Soc.*, 156:A33 – A36, 2009.

[90] M. Seto, K. Westra, M.J. Brett. Arrays of self-sealed microchambers and channels. *J. Mater. Chem.*, 12:2348 – 2351, 2002.

[91] L.W. Bezuidenhout, M.J. Brett. Ultrathin layer chromatography on nanostructure thin films. *J. Chromo. A*, 1183:179 – 185, 2008.

[92] S.R. Jim, M. T. Taschuk, G.E. Morlock, L.W. Bezuidenhout, W. Schwack, M.J. Brett. Engineered anisotropic microstructures for ultrathin-layer chromatography. *Anal. Chem.*, 82:5349 – 5356, 2010.

[93] J.J. Steele, M.T. Taschuk, M.J. Brett. Nanostructured metal oxide thin films for humidity sensors. *IEEE Sens.*, 8:1422 – 1429, 2008.

[94] S.V. Kesapragada, P.Victor, O. Nalamasu, D. Gall. Nanospring pressure sensors grown by glancing angle deposition. *Nano. Lett.*, 6:854 – 857, 2006.

[95] W. Zhang, N. Ganesh, I.D. Block, B.T. Cunningham. High sensitivity photonic crystal biosensor incorporating nanorod structures for enhanced surface area. *Sens. Act. B*, 131:279 – 284, 2007.

[96] R. Carbone, et al. Biocompatibility of cluster-assembled nanostructured TiO_2 with primary and cancer cells. *Biomaterials*, 27:3221 – 3229, 2006.

[97] J.X. Liu, D.-Z. Yang, F. Shi, Y.-J. Cai. Sol-gel deposited TiO_2 film on NiTi surgical alloy for biocompatibility improvement. *Thin Solid Films*, 429:225 – 230, 2003.

[98] A. Jaroenworaluck, W. Sunsaneeyametha, N. Kosachan, R. Stevens. Characteristics of silica-coated TiO_2 and its UV absorption for sunscreen cosmetic applications. *Surf. Interface Anal.*, 38:473 – 477, 2006.

[99] A. Fujishima, T.N. Rao, D.A. Tryk. Titanium dioxide photocatalysis. *J. Photochem. Photobio. C*, 1:1 – 21, 2000.

[100] M.J. Taschuk, J.J. Steele, A.C. van Popta, M.J. Brett. Photocatalytic regeneration of interdigitated capacitor relative humidity sensors fabricated by glancing angle deposition. *Sens. Act. B*, 134:666 – 671, 2008.

[101] G.X. Shen, Y.C. Chen, C.J. Lin. Corrosion protection of 316 L stainless steel by a TiO_2 nanoparticle coating prepared by sol-gel method. *Thin Solid Films*, 489:130 – 136, 2005.

[102] K. Bange, C.R. Ottermann, O. Anderson, U. Jeschkowski, M. Laube, R. Feile. Investigations of TiO_2 films deposited by different techniques. *Thin Solid Films*, 197:279 – 285, 1991.

[103] S. Schiller, G. Beister, W. Sieber, G. Schirmer, E. Hacker. Influence of deposition parameters on the optical and structural properties of TiO_2 films produced by reactive D.C. plasmatron sputtering. *Thin Solid Films*, 83:239 – 245, 1981.

[104] M.M. Hawkeye, M.J. Brett. Glancing angle deposition: Fabrication, properties, and applications of micro- and nanostructured thin films. *J. Vac. Sci. Technol. A.*, 25:1317 – 1335, 2007.

[105] A. Lakhatakia, R. Messier. *Sculptured thin films: nanoengineered morphology and optics*. SPIE Press, Bellingham WA, 2004.

[106] M.J. Colgan, B. Djufors, D.G. Ivey, M.J. Brett. Effects of annealing on titanium dioxide structured thin films. *Thin Solid Films*, 466:92 – 96, 2004.

[107] G. Eranna, B.C. Joshi, D.P. Runthala, R.P. Gupta. Oxide materials for development of integrated gas sensors - a comprehensive review. *CRC Cr. Rev. Sol. State.*, 29:111 – 188, 2004.

[108] I. Izquierdo-Barba, M. Colilla, Maria Vallet-Regi. Nanostructured mesoporous silicas for bone tissue regeneration. *J. Nanomater.*, 2008:1 – 14, 2008.

[109] S.R. Kennedy, J.C. Sit, D.J. Broer, M.J. Brett. Optical activity of chiral thin films and liquid crystal hybrids. *Liq. Cryst.*, 28:1799 – 1803, 2001.

[110] P. Ferreira-Aparicio, I. Rodriguez-Ramos, A. Guerrero-Ruiz. Methane interaction with silica and alumina supported metal catalysts. *App. Catal. A*, 148:343 – 356, 1997.

[111] B.K. Min, A.K. Santra, D.W. Goodman. Understanding silica-supported metal catalysts: Pd/silica as a case study. *Catal. Today*, 85:113 – 124, 2003.

[112] K.D. Harris, A. Huizinga, M.J. Brett. High-speed porous thin film humidity sensors. *Electrochem. Solid St.*, 5:H27 – H29, 2002.

[113] E. Traversa. Ceramic sensors for humidity detection: the state-of-the-art and future developments. *Sens. Act. B*, 23:135 – 156, 1995.

[114] M.J. Alam, D.C. Cameron. Optical and electrical properties of transparent conductive ITO thin films deposited by sol-gel process. *Thin Solid Films*, 377-378:455 – 459, 2000.

[115] C.J. Granqvist, A. Hultaker. Transparent and conducting ITO films: new developments and applications. *Thin Sold Films*, 411:1 – 5, 2002.

[116] M.G. Mason, L.S. Hung, C.W. Tang, S.T. Lee, K.W. Wong, M. Wang. Characterization of treated indium-tin-oxide surfaces used in electroluminescent devices. *J. Appl. Phys.*, 86:1688, 1999.

[117] C.J. Brabec, N.S. Sariciftci, J.C. Hummelen. Plastic solar cells. *Adv. Funct. Mater.*, 11:15 – 26, 2001.

[118] K.D. Harris, A.C. van Popta, J.C. Sit, D.J. Broer, M.J. Brett. A birefringent and transparent electrical conductor. *Adv. Func. Mat.*, 18:2147 – 2153, 2008.

[119] T. Bauer, J. S. Kolb, T. Loffler, E. Mohler, H. G. Roskos, U.C. Pernisz. Indium-tin-oxide-coated glass as dichroic mirror for far-infrared electromagnetic radiation. *J. Appl. Phys.*, 92:2210, 2002.

[120] N. Ishikura, M. Fujii, K. Nishida, S. Hayashi, J. Diener. Dichroic rugate filters based on birefringent porous silicon. *Opt. Expr.*, 16:15331 – 15339, 2008.

[121] M.S. Salem, M.J. Sailor, K. Fukami, T. Sakka, Y.H. Ogata. Sensitivity of porous silicon rugate filters for chemical vapor detection. *J. Appl. Phys.*, 103:083516, 2008.

[122] I.J. Hodgkinson, Q.H. Wu, A. Lakhtakia, M.W. McCall. Spectral-hole filter fabricated using sculptured thin-film technology. *Opt. Comm.*, 15:79 – 84, 2000.

[123] E. Guillermain, V. Lysenko, R. Orobtchouk, T. Benyattou, S. Roux, A. Pillonnet, O. Perriat. Bragg surface wave device based on porous silicon and its application for sensing. *Appl. Phys. Lett.*, 90:241116, 2007.

[124] B.G. Bovard. Rugate filter theory: an overview. *Appl. Opt.*, 32:5427 – 5442, 1993.

[125] M.R. Kupsta. Advanced Methods for GLAD Thin Films. Master's thesis, University of Alberta, 2010.

[126] M.M. Hawkeye, M.J. Brett. Narrow bandpass optical filters fabricated with one-dimensionally periodic inhomogeneous thin films. *J. Appl. Phys.*, 100:044322, 2006.

[127] M.M. Hawkeye, R. Joseph, J.C. Sit, M.J. Brett. Coupled defects in one-dimensional photonic crystal films fabricated with glancing angle deposition. *Opt. Express*, 18:13220 – 13226, 2010.

[128] A.C. van Popta, J.C. Sit, M.J. Brett. Optical properties of porous helical thin films and the effects of postdeposition annealing. *Proc. SPIE - Int. Soc. Opt. Eng.*, 5464:3632 – 3639, 2004.

[129] K.N. Rao, R.S. Mohan. Optical properties of electron-beam evaporated TiO_2 films deposited in an ionized oxygen medium. *J. Vac. Sci. Tech. A*, 8:3260 – 3264, 1990.

[130] K.N. Rao. Influence of deposition parameters on optical properties of TiO_2 films. *Opt. Eng.*, 41:2357 – 2364, 2002.

[131] W.-H. Wang, S. Chao. Annealing effects on ion-beam-sputtered titanium dioxide films. *Opt. Lett.*, 23:1417 – 1419, 1998.

[132] J.J. Steele, A.C. van Popta, M.H. Hawkeye, J.C. Sit, M.J. Brett. Nanostructured gradient index optical filter for high-speed humidity sensing. *Sens. Actuators B*, 120:213 – 219, 2006.

[133] W.S. Rasband. ImageJ. *National Institutes of Health: USA*, v1.36b, 2006.

[134] N.G. Wakefield, J.C. Sit. On the uniformity of thin films fabricated by glancing angle deposition. *J. Appl. Phys.*, (submitted):(27 manuscript pages), 2010.

[135] D.E. Aspnes. Optical properties of thin films. *Thin Solid Films*, 89:249 – 262, 1982.

[136] J. Gospodyn, J.C. Sit. Characterization of dielectric columnar thin films by variable angle Mueller matrix and spectroscopic ellipsometry. *Opt. Mater.*, 29:318 – 325, 2006.

[137] B. Fubini, V. Bolis, M. Bailes, F.S. Stone. The reactivity of oxides with water vapor. *Solid State Ionics*, 32/33:258 – 272, 1989.

[138] M. Brumbach, P.A. Venernan, F.S. Marrikar, T. Schulmeyer, A. Simmonds, W. Xia, P. Lee, N.R. Armstrong. Surface composition and electrical and electrochemical properties of freshly deposited and acid-etched indium tin oxide electrodes. *Langmuir*, 23:11089 – 11099, 2007.

[139] S. Tsoi, E. Fok, J.C. Sit, J.G.C. Veinot. Superhydrophobic, high surface area, 3-D SiO_2 nanostructures through siloxane-based surface functionalization. *Langmuir*, 20:10771 – 10774, 2004.

[140] S. Tsoi, E. Fok, J.C. Sit, J.G.C. Veinot. Surface functionalization of porous nanostructured metal oxide thin films fabricated by glancing angle deposition. *Chem. Mater.*, 18:5260 – 5266, 2006.

[141] T. Morimoto, M. Nagao, F. Tokuda. Relation between the amounts of chemisorbed and physisorbed water on metal oxides. *J. Phys. Chem.*, 73:243 – 248, 1969.

[142] U. Diebold. The surface science of titanium dioxide. *Surf. Sci. Reports*, 48:53 – 229, 2003.

[143] M. Nagao. On physisorption of water on zinc oxide surface. *J. Phys. Chem.*, 75:3822 – 3828, 1971.

[144] M.A. Henderson. The interaction of water with solid surfaces: fundamental aspects revisited. *Surf. Sci. Rep.*, 46:1 – 308, 2002.

[145] J. Siewart. Characterizing humidity interaction with GLAD films. *Final presentation*, EE 497:1 – 13, 2010.

[146] J.J. Steele. *Nanostructured Thin Films for Humidity Sensing*. PhD thesis, University of Alberta, 2010.

[147] R.N. Tait, T. Smy, M.J. Brett. Modelling and characterization of columnar growth in evaporated films. *Thin Solid Films*, 226:196 – 201, 1993.

[148] D.J. Poxson, F.W. Mont, M.F. Schubert, J.K. Kim, E.F. Schubert. Quantification of porosity and deposition rate of nanoporous films grown by oblique-angle deposition. *Appl. Phys. Lett.*, 93:101914, 2008.

[149] J.R. Taylor. *An Introduction to Error Analysis: The study of uncertainties in physical measurements, 2nd Ed.* University Science Books, Sausalito CA, 1997.

[150] A.W. Chester, P. Clement, S. Han. Faujasite zeolitic materials. US patent 6136219A, Oct 2000.

[151] M.G. Nijkamp, J.E.M.J. Raaymakers, A.J. van Dillen, K.P. de Jong. Hydrogen storage using physisorption - materials demands. *Appl. Phys. A*, 72:619 – 623, 2001.

[152] H.K. Chae, D.Y. Siberio-Perez, J. Kim, Y.B. Go, M. Eddaudi, A.J. Matzger, M. O'Keeffe, O.M. Yaghi. A route to high surface area porosity inclusion of large molecules in crystals. *Nature*, 427:523 – 527, 2004.

[153] K. Koh, A. G. Wong-Foy, A.J. Matzger. A porous coordination copolymer with over 5000 m^2/g BET surface area. *J. Am. Chem. Soc.*, 191:4184–4185, 2009.

[154] J. Jiu, S. Isoda, F. Wang, M. Adachi. Dye-sensitized solar cells based on a single-crystalline TiO_2 nanorod film. *J. Phys. Chem. B*, 110:2087 – 2092, 2006.

[155] S.H. Kang, S.-H. Choi, M.-S. Kang, J.-Y. Kim, H.-S. Kim, T. Hyeon, Y.-E. Sung. Nanorod-based dye-sensitized solar cells with improved charge collection efficiency. *Adv. Mater.*, 20:54 – 58, 2007.

[156] M. Adachi, Y. Murata, J. Takao, J. Jiu, M. Sakamoto, F. Wang. Highly efficient dye-sensitized solar cells with a titania thin-film electrode composed of a network structure of single-crystal-like TiO_2 nanowires made by the oriented attachment mechanism. *J. Am. Chem. Soc.*, 126:14943 – 14949, 2004.

[157] Y.-J. Jen, M.-J. Lin, J.-H. Chao. Single dielectric columnar thin film as a humidity sensor. *Sens. Act. B*, 149:67 – 70, 2010.

[158] B. Djufors, J.N. Broughton, M.J. Brett, D.G Ivey. Microstructural characterization of porous manganese thin films for electrochemical supercapacitor applications. *J. Mater. Sci.*, 38:4817 – 4830, 2003.

[159] X.-J. Tang, G. Zhang, Y.-P. Zhao. Electrochemical characterization of silver nanorod electrodes prepared by oblique angle deposition. *Nanotechnology*, 17:4439, 2006.

[160] D.A. Rider, B.J. Worfolk, K.D. Harris, A. Lalany, K. Shahbazi, M.D. Fleischauer, M.J. Brett, J.M. Buriak. Stable inverted polymer/Fullerene solar cells using a cationic polythiophene modified PEDOT:PSS cathodic interface. *Adv. Funct. Mater.*, 20:2404 – 2415, 2010.

[161] J.N. Broughton, M.J. Brett. Electrochemical capacitance in manganese thin films with chevron microstructure. *Electrochem. Solid. St.*, 5:A279 – A282, 2002.

[162] G.K. Kiema, M.J. Brett. Electrochemical characterization of carbon films with porous microstructures. *J. Electrochem. Soc.*, 150:E342 – E347, 2003.

[163] J. Kim, Z. Dohnalek, B.D. Kay. Structural characterization of nanoporous pd films grown via ballistic deposition. *Surf. Sci.*, 586:137 – 145, 2005.

[164] D.W. Flaherty, Z. Dohnalek, A. Dohnalkova, B.W. Arey, D.E. McCready, N. Ponnusamy, C.B. Mullins, B.D. Kay. Reactive ballistic deposition of porous TiO_2 films: growth and characterization. *J. Phys. Chem. C*, 11:4765 – 4773, 2007.

[165] D.W. Flaherty, N.T. Hahn, D. Ferrer, T.R. Engstrom, P.L. Tanaka, C.B. Mullins. Growth and characterization of high surface area titanium carbide. *J. Phys. Chem. C*, 113:12742 – 12752, 2009.

[166] Z. Dohnalek, G.A. Kimmel, D.E. McCready, J.S. Young, A. Dohnalkova, R.S. Smith, B.D. Kay. Structural and chemical characterization of aligned crystalline nanoporous MgO films grown via reactive ballistic deposition. *J. Phys. Chem. B*, 106:3526 – 3529, 2002.

[167] K.P. Stevenson, G.A. Kimmel, Z. Dohnalek, R.S. Smith, B.D. Kay. Controlling the morphology of amorphous solid water. *Science*, 283:1505 – 1507, 1999.

[168] M. Suzuki, Y. Taga. Numerical study of the effective surface area of obliquely deposited thin films. *J. Appl. Phys.*, 90:5599 – 5605, 2001.

[169] M.C. Demirel. Emergent properties of spatially organized poly(p-xylylene) films fabricated by vapor deposition. *Colloid. Surface A*, 321:121 – 124, 2008.

[170] C.-C. Li, J.-L. Huang, R.-J. Lin, D.-F. Lii, C.-H. Chen. Performance characterization of nonevaporable porous Ti getter films. *J. Vac. Sci. Technol. A*, 25:1373 – 1380, 2007.

[171] S. Lowell, J.E. Shields, M.A. Thomas, M. Thommes. *Characterization of porous solids and powders: Surface area, pore size and density*. Springer, The Netherlands, 2006.

[172] K.S.W. Sing, D.H. Everett, R.A.W. Haul, L. Moscou, R.A. Pierotti, J. Rouquerol, T. Siemieniewska. Reporting physisorption data for gas/solid systems with special reference to the determination of surface area and porosity. *Pure Appl. Chem.*, 57:603 – 619, 1985.

[173] M.A. Thommes. *Chapter 11: Physical adsorption characterization of ordered and amorphous mesoporous materials in Nanoporous Materials: Science and Engineering*. (G. Q. Lu and X.S. Zhao, Eds.) Imperial College Press, London, 2004.

[174] M.A. Thommes, N.Nishiyama, S. Tanaka. *Aspects of a novel method for the pore size analysis of thin silica films based on krypton adsorption at liquid argon temperature (87.3 K) in Rec. Prog. Mesostruct. Mat., Proc. 5th Int. Mesostruct. Mat. Symp.* Elsevier, Shanghai, 2006.

[175] M.A. Thommes, R. Kohn, M. Froba. Sorption and pore condensation behavior of pure fluids in mesoporous MCM-48 silica, MCM-41 silica, SBA-15 silica and controlled-pore glass at temperatures above and below the bulk triple point. *Appl. Surf. Sci.*, 196:239 – 249, 2002.

[176] C.-Y. Chiu, A.S.T. Chiang, K.-J. Chao. Mesoporous silica powders and films - Pore size characterization by krypton adsorption. *Micropor. Mesopor. Mat.*, 91:244 – 253, 2006.

[177] G. Zotti, G. Schiavon, S. Zecchin. Adsorption of ferroscene compounds on indium-tin-oxide electrodes. Enhancement of adsorption by decomposition of ferrocenium molecules by oxygen. *Langmuir*, 14:1728 – 1733, 1998.

[178] C. Donley, D. Dunphy, D. Paine, C. Carter, K. Nebesny, P. Lee, D. Alloway, N.R. Armstrong. Characterization of indium-tin-oxide interfaces using x-ray photoelectron spectroscopy and redox processes of a chemisorbed probe molecule: effect of surface pretreatment conditions. *Langmuir*, 18:450 – 457, 2002.

[179] Y.-H. Liau, N.F. Scherer, K.Rhodes. Nanoscale electrical conductivity and surface spectroscopic studies of indium tin oxide. *J. Phys. Chem. B*, 105:3282 – 3288, 2001.

[180] J. Li, L. Wang, J. Liu, G. Evmenenko, P. Dutta, T.J. Marks. Characterization of transparent conducting oxide surfaces using self-assembled electroactive monolayers. *Langmuir*, 24:5755 – 5765, 2008.

[181] K.M. Krause, M.T. Taschuk, K.D. Harris, D.A. Rider, N.G. Wakefield, J.C. Sit, J.M. Buriak, M. Thommes, M.J. Brett. Surface area characterization of obliquely deposited metal oxide nanostructured thin films. *Langmuir*, 26:4368 – 4376, 2010.

[182] M. De Podesta. *Understanding the properties of matter, 2nd ed.* Taylor and Francis, London, 2002.

[183] J. Xu, J. Umehara, I. Kojima. Effect of deposition parameters on composition, structures, density and topography of CrN films deposited by R.F. magnetron sputtering. *Appl. Surf. Sci.*, 201:208 – 218, 2002.

[184] T. Smy, D. Vick, M.J. Brett, S.K. Dew, A.T. Wu, J.C. Sit. Three-dimensional simulation of film microstructure produced by glancing angle deposition. *J. Vac. Sci. Technol. A*, 18:2507 – 2512, 2000.

[185] B. Dick, M.J. Brett, T. Smy. Investigation of substrate rotation at glancing incidence on thin-film morphology. *J. Vac. Sci. Technol. B*, 21:2569 – 2575, 2003.

[186] D.W. Flaherty, R.A. May, S.P. Berglund, K.J. Stevenson, C.B. Mullins. Low temperature synthesis and characterization of nanocrystalline titanium carbide with tunable porous architectures. *Chem. Mat.*, 22:319 – 329, 2010.

[187] A. Borras, J.R. Sanchez-Valencia, J. Garrido-Molinero, A. Barranco, A.R. Gonzalez-Elipe. Porosity and microstructure of plasma deposited TiO_2 thin films. *Micopor. Mesopor. Mat.*, 118:314 – 324, 2009.

[188] G.-S. Shao, X.-J. zhang, Z.-Y. Yuan. Preparation and photocatalytic activity of hierarchically mesoporous-macroporous $TiO_2 - xN_2$. *Appl. Catal. B*, 82:208 – 218, 2008.

[189] Y. Ao, J. Xu, D. Fu, C. Yuan. Prepation of porous titania thin film and its photocatalytic activity. *Appl. Surf. Sci.*, 255:3137 – 3140, 2008.

[190] K.S.W. Sing, R.T. Williams. Review: The use of molecular probes for the characterization of nano-porous adsorbents. *Part. Part. Syst. Char.*, 21:71 – 79, 2004.

[191] F.R. Hung, S. Bhattacharya, B. Coasne, M. Thommes, K.E. Gubbins. Argon and krypton adsorption on template mesoporous silicas: molecular simulation and experiment. *Adsorption*, 13:425 – 437, 2007.

[192] Y. Shen, T. Yamazaki, Z. Liu, D. Meng, T. Kikuta, N. Nakatani. Influence of effective surface area on gas sensing properties of WO_3 sputtered thin films. *Thin Solid Films*, 517:2069 – 2072, 2009.

[193] M. Kruk, M. Jaroniec, A. Sayari. Adsorption study of surface and structural properties of MCM-41 materials of different pore sizes. *J. Phys. Chem. B*, 101:583 – 589, 1997.

[194] P.I. Ravikovitch, S.C.O. Domhnaill, A.V. Neimark, F. Schuth, K.K. Unger. Capillary hysteresis in nanopores: Theoretical and experimental studies of nitrogen adsorption on MCM-41. *Langmuir*, 11:4765– 4772, 1995.

[195] N.A. Seaton, J.P.R.B. Walton, N. Quirke. A new analysis method for the determination of the pore size distribution of porous carbons from nitrogen adsorption measurements. *Carbon*, 27:853 – 861, 1989.

[196] C.T. Kresge, M.E. Leonowicz, W.J. Roth, J.C. Vartull, J.S. Beck. Ordered mesoporous molecular sieves synthesized by a liquid-crystal template mechanism. *Nature*, 359:710 – 712, 1992.

[197] C.J. Martinez, B. Hockey, C.B. Montgomery, S. Semanik. Porous tin oxide nanostructured microspheres for sensor applications. *Langmuir*, 21:7937 – 7944, 2005.

[198] E.T.H. Tan, G.W. Ho, A.S.W. Wong, S. Kawi, A.T.S. Wee. Gas sensing properties of tin oxide nanostructures synthesized via a solid-state reaction method. *Nanotechnology*, 19:255706, 2008.

[199] L.D. Gelb, K.E. Gubbins, R. Radhakrishnan, M. Sliwinska-Bartkowiak. Phase separation in confined systems. *Rep. Prog. Phys.*, 62:1573 – 1639, 1999.

[200] C.G.V Burgess, D.H. Everett, S. Nuttall. Adsorption hysteresis in porous materials. *Pure Appl. Chem.*, 61:1845 – 1853, 1989.

[201] M.A. Thommes, G.H. Findenegg. Pore condensation and critical-point shift of a fluid in controlled-pore glass. *Langmuir*, 10:4270 – 4277, 1994.

[202] A. Zukal. Adsorption and pore condensation of krypton on mesoporous silicas at 77 K. *Micropor. Mesopor. Mat.*, 92:220 – 226, 2006.

[203] NIST Chemistry Webbook. Thermophysical properties of fluid systems, URL: http://webbook.nist.gov/chemistry/fluid/, "Accessed: Jan" 2010.

[204] N. Otsu. A threshold selection method from gray-level histograms. *IEEE Trans. Syst. Man. Cyber.*, 9:62 – 66, 1979.

[205] C. Buzea, G. Beydaghyan, C. Elliot, K. Robbie. Control of power law scaling in the growth of silicon nanocolumns pseudo-regular arrays deposited by glancing angle deposition. *Nanotechnology*, 16:1986 – 1992, 2005.

[206] M. Malac, R.F. Egerton. Observation of the microscopic growth mechanism of pillars and helices formed by glancing-angle thin-film deposition. *J. Vac. Sci. Technol. A.*, 19:158 – 166, 2001.

[207] D. Vick, T. Smy, M.J. Brett. Growth behavior of evaporated porous thin films. *J. Mater. Res.*, 17:2904–2911, 2002.

[208] S. Mukherjee, D. Gall. Anomalous scaling during glancing angle deposition. *Appl. Phys. Lett.*, 9:173106, 2009.

[209] M.T. Taschuk, K.M. Krause, J.J. Steele, M.A. Summers, M.J. Brett. Growth scaling of metal oxide columnar thin films deposited by glancing angle deposition. *J. Vac. Sci. Technol. B*, 27:2106 – 2111, 2009.

[210] J. Wang, H. Huang, S.V. Kesapragada, D. Gall. Growth of y-shaped nanorods through physical vapor deposition. *Nano Lett.*, 5:2505 – 2508, 2005.

[211] C.M. Zhou, D. Gall. Branched Ta nanocolumns grown by glancing angle deposition. *Appl. Phys. Lett*, 88:203117, 2006.

[212] T. Sakamoto, Z. Cheng, M. Takahaski, M. Owari, Y. Nihei. Development of an ion and electron dual focused beam apparatus for three-dimensional microanalysis. *Jpn. J. Appl. Phys.*, 37:2051 – 2056, 1998.

[213] A.J. Kubis, G.J. Shiflet, D.N. Dunn, R. Hull. Focused ion-beam tomography. *Metell. Mater. Trans. A.*, 35A:1935 – 1943, 2004.

[214] A.J. Kubis, T.E. Vandervelde, J.C. Bean, D.N. Dunn, R. Hull. Analysis of the three-dimensional ordering of epitaxial Ge quantum dots using focused ion beam tomography. *Appl. Phys. Lett.*, 88:263103, 2006.

[215] M.A. Groeber, B.K. Haley, M.D. Uchic, D.M. Dimiduk, S. Ghosh. 3D reconstruction and characterization of polycrystalline microstructures using a FIB-SEM systems. *Mater. Charact.*, 57:259 – 273, 2006.

[216] C. Holzapfel, W. Schaf, M. Marx, H. Vehoff, F. Mucjlich. Interaction of cracks with precipitates and grain boundaries: Understanding focused ion beam tomography. *Scripta. Mat.*, 56:697 – 700, 2007.

[217] M.D. Uchic, M. De Graef, R. Wheeler, D.M. Dimiduk. Microstructural tomography of a $Ni_{70}Cr_{20}Al_{10}$ superalloy using focused ion beam microscopy. *Ultramicroscopy*, 109:1229 – 1235, 2009.

[218] V.E. Krohn. Liquid metal droplets for heavy particles propulsion. *Progr. Astonaut. Rocketry*, 51:73 – 80, 1961.

[219] P.R. Munroe. The application of focused ion beam microscopy in the material sciences. *Mater. Charact.*, 60:2 – 13, 2009.

[220] B. Van Leer, Y.-C. Wang, L.A. Giannuzzi. Protective carbon deposition for superior FIB prepared (S)TEM specimens. *Microsc. Microanal.*, 15 (S02):336 – 337, 2009.

[221] S.M. Schwarz, B.W. Kepshall, L.A. Giannuzzi, M.R. McCartney. Avoiding the curtaining effect: backside milling by FIB INLO. *Microsc. Microanal.*, 9 (S02):116 – 117, 2003.

[222] M.K. Miller, R.G. Forbes. Atom probe tomography. *Mater. Charact.*, 60:461 – 469, 2009.

[223] M.K. Miller, K.F Russell, G.B. Thompson. Strategies for fabricating atom probe specimens with a dual beam fib. *Ultramicroscopy*, 102:287 – 298, 2005.

[224] H.-J. Kang, J.H. Kim, J. won Oh, T. S. Back, H. J. Kim. Ultra-thin TEM sample preparation with advanced backside FIB milling method. *Microsc. Microanal.*, 16 (S2):170 – 171, 2010.

[225] R.M. Langford. Focused ion beam techniques for nanomaterials characterization. *Microsc. Res. Techniq.*, 69:538 – 549, 2006.

[226] K.D. Harris, K.L. Westra, M.J. Brett. Fabrication of perforated thin films with helical and chevron pore shapes. *Electrochem. Solid St.*, 4:C39 – C42, 2001.

[227] K.M. Krause, D.W. Vick, M. Malac, M.J. Brett. Taking a little off the top: Nanorod array morphology and growth studied by focused ion beam tomography. *Langmuir*, 26:17558 – 17567, 2010.

[228] K. Zuiderveld. *Contrast limited adaptive histograph equalization in Graphic Gems IV*. Academic Press Professional, San Diego, 1994.

[229] T. Karabacak, J.P. Singh, Y.-P. Zhao, G.-C. Wang, T.-M. Lu. Scaling during shadowing growth of isolated nanocolumns. *Phys. Rev. B*, 68:125408, 2003.

[230] D. Sadot, E. Biomovich. Tunable optical filters for dense WDM networks. *IEEE Comm. Mag.*, 36:50 – 55, 1998.

[231] Y. Huang, Y. Zhou, S.-T. Wu. Broadband circular polarizer using stacked chiral polymer films. *Opt. Exp.*, 15:6414 – 6419, 2007.

[232] Z. Zhuang, J.S. Patel, Y.J. Kim. Behavior of the cholesteric liquid-crystal Fabry-Perot cavity in the Bragg reflection band. *Phys. Rev. Lett.*, 84:1168 – 1171, 2000.

[233] E. Wolf M. Born. *Principles of Optics 7th Edition*. Cambridge University Press, Cambridge, 1999.

[234] A. Thelen. Multilayer filters with wide transmittance bands. *J. Opt. Soc. Am.*, 53:1266 – 1270, 1963.

[235] A. Piegari, J. Buir. Variable narrowband transmission filters with a wide rejection band for spectrometry. *Appl. Opt.*, 45:3768 – 3773, 2006.

[236] D.W. Dodds, I. Gershkovich, B.T. Cunningham. Fabrication of a graded-wavelength guided-mode resonance filter photonic crystal. *Appl. Phys. Lett.*, 89:123113, 2006.

[237] B.S. Krusor, D.B. Fenner, D.K. Biegelsen, R.D. Yingling, R.D. Bringans. Description and applications of a graded-thickness growth tehnique for molecular-beam epitaxy. *J. Vac. Sci. Technol. B*, 8:172 – 175, 1989.

[238] Q. Wang, H. Moutinho, B. To, J. Perkins, D. Ginley, H.M. Branz, L.R. Tessler, D. Han. Combinatorial approach to thin-film silicon materials and devices. *Proc, MRS Spring Meeting*, Symposium A, 2003.

[239] A. Thelen. Circularly wedged optical coatings. *Appl. Opt.*, 4:977 – 985, 1965.

[240] A.R. Parker. 515 million yars of structural colour. *J. Opt. A.*, 2:R15 – R28, 2000.

[241] S. Kinoshita, S. Yoshioka. Structural colors in nature: the role of regularity and irregularity in the structure. *Chem. Phys. Chem.*, 6:1442 – 1459, 2005.

[242] M. Sarikaya, C. Tamerler, A.K.-Y. Schulten, F. Baneyx. Molecular biomimetrics: nanotechnology through biology. *Nat. Mat.*, 2:577 – 585, 2003.

[243] P. Vukusic, J.R. Sambles. Photonic structures in biology. *Nature*, 424:852 – 855, 2003.

[244] R. Won. On the wing of a butterfly. *Nat. Photonics*, 2:130, 2007.

[245] S. Kinoshita, S. Yoshioka, K. Kawgoe. Mechanisms of structural colour in the Morpho butterfly: cooperation of regularity and irregularity in an irridescent scale. *Proc. Royal Soc. B*, 269:1417 – 1421, 2002.

[246] P. Vukusic, J.R. Sambles, C.R. Lawrence. Colour mixing in wing scales of a butterfly. *Nature*, 404:457, 2000.

[247] S. Lowrey, L.D. De Silva, I. Hodgkinson, J. Leader. Observation and modeling of polarized light from scarab beetles. *J. Opt. Soc. Amer. A*, 24:2418–2425, 2007.

[248] P. Vukusic. *Dekker Encyclopedia of Nanoscience and Nanotechnology, Vol 5 (Eds: J.A. Schwartz, C.I. Contescu, K. Putyera)*. Marcel Dekker, New York, 2004, p. 3716.

[249] D.J. Brink, N.G. van der Berg, L.C. Prinsloo, I.J. Hodgkinson. Unusual colouration in scarabaeid beetles. *J. Phys. D*, 40:2189 – 2196, 2007.

[250] R.O. Prum, R. Torres. Structural colouration of avian skin: convergent evolution of coherently scattering dermal collagen arrays. *J. Exp. Biol.*, 206:2409 – 2429, 2003.

[251] M. Kurachi, Y. Takaku, Y. Komiya, T. Hariyama. The origin of extensive colour polymorphism in Plateumaris sericea (Chrysomelidae, Coleoptera) The origin of extensive colour polymorphism in Plateumaris sericea (Chrysomelidae, Coleoptera). *Naturwissenschaften*, 89:295 – 298, 2002.

[252] A.Z. Genack, V.I. Kopp. Chiral fiber Bragg gratings. *Proc. SPIE*, 5508:57 – 64, 2004.

[253] L.-W. Chen J.-Y. Chen. Polarization-dependent filters based on chiral photonic structures with defects. *J. Opt. A*, 7:558 – 566, 2005.

[254] A.C. van Popta, K.R. van Popta, J.C. Sit, M.J. Brett. Sidelobe suppression in chiral optical filters by apodization of the local form birefringence. *J. Opt. Soc. Am. A*, 24:3140 – 3149, 2007.

[255] J.B. Sorge, A.C. van Popta, J.C. Sit, M.J. Brett. Circular birefringence dependence on chiral film porosity. *Opt. Exp.*, 14:10550 –10557, 2006.

[256] Schneider Kreuznach Optics. Industrial Filters (Specifications Document). *http://www.schneiderkreuznach.com/industrial_filters/industrial_filters.html*, Accessed July 17, 2008.

[257] P.C. Deguzman, G.P. Nordin. Stacked subwavelength gratings as circular polarization filters. *Appl. Opt.*, 40:5731 – 5737, 2001.

[258] P.C.P. Hrudey, K.L. Westra, M.J. Brett. Highly ordered organic Alq_3 chiral luminescent thin films fabricated by glancing-angle deposition. *Adv. Mater.*, 18:224 – 228, 2006.

[259] A.C. van Popta, M.J. Brett, J.C. Sit. Double-handed circular bragg phenomena in polygonal helix thin films. *J. Appl. Phys.*, 98:083517, 2005.

[260] D.R. Lide, Ed. *CRC Handbook of Chemistry and Physics, 88th Ed.* CRC Press, Boca Raton, 88th edition, 20072-2008.

[261] S.R. Jim, L.W. Bezuidenhout, M.T. Taschuk, M.J. Brett. Porous separation media engineered through glancing angle deposition. *LOC World Congress*, (Poster):San Francisco, 2009.

Appendix A

Estimation of Density, Surface Area, and Porosity

A.1 Code to estimate water adsorbed into TiO_2 and SiO_2 films

```
% Code to estimate the amount of water adsorbed into films
% Uses spectral shifts observed in humidity testing on Rugate films
% Most data for TiO2 films, but calibrates to SiO2 films based
% on work that Josh Siewart and Mike Taschuk did on GLAD+QCMs


close all;
clear all;
clc;

fontSize = 20;
lineWidth = 2;
markerSize = 10;
colours = 'brmgckbrmgckbrmgckbrmgck';
shapes = 'svodx>+hsvodx>+hsvodx>';
lineStyles = {'-','--',':','-.','-','--',':','-.','-','--',':','-.','-','--',':','-.'};



% Dielectric constants
n_air = 1;
n_tio2 = 2.4;
n_water = 1.33;
% Fixed.  Correct relationship between dielectric constant and refractive index!
e_air = n_air^2;
e_tio2 = n_tio2^2;
e_water = n_water^2;

% Assume humidity = 50%

% From My_Documents\Research\Research_Areas\Humidity_Sensors\Matlab_Code\ParseHumidityData\
%  plotDir_Shifted_Cycle1_ad, plotDir_Shifted_Cycle1_de,
%  plotDir_Shifted_Cycle2_ad
% Rugate films with different initial alpha (30,45,50,60) to a final alpha of 80
% Humidity tests performed from 0 to 100 % Relative Humidity
% Stop band movement tracked as a function of humidity
% Change in blue edge corresponds to changing properties of the 80 degree
% layer, while change in red edge corresponds to changing properties of the
% initial alpha layer

% At 50 % RH
% Change in red and blue edges found from humidity sensor plots
% extrapolated to 0% RH
hc_alphas =       [     30      45     50      60     80];
wvlChange =       [    9.4    18.2   16.2    14.7   12.2];
thickness =       [   1944    1813   1888    1758   1944];
Nturns =          [     14      14     14      14     14];


% Result = 139  129.5  135  126  126
% (obtained from SEM thicknesses)
```

```
periodicity = thickness./Nturns;


% Measured values
% Here we are assuming that the "tio2_mass_mg" consists of both TiO2 and adsorbed water
tio2_alphas =              [  30    45    50   55    60   65    70   75    80    85];
tio2_mass_mg =             [5.56  8.69  8.18 8.59  5.68 7.72  6.39 4.91 15.09 13.05];
tio2_footprint_area_mm2 = [1356  2445  2313 2664  2037 2837  2601 2559  3988  3963];
tio2_thickness_um =        [1.09  1.03 1.019 0.97 0.958 0.95 0.999 1.09 2.183  2.64];
sio2_alphas =              [  30    45    50   55    60    65    70   75    80    85];
sio2_mass_mg =             [4.60  5.66  2.43 2.49  3.79  3.77  4.88 2.67  2.12  1.53];
sio2_footprint_area_mm2 = [1956  3230  1356 1496  2549  2565  4200 2566  2580  3127];
sio2_thickness_um =        [1.10  0.90  0.91 0.93  0.86  0.93  0.96 0.95  0.88  0.99];



% Surface areas
tio2_sa_m2_m2_um =         [  32  597  810  852   881  726   588  386   303   207];
sio2_sa_m2_m2_um =         [  39  702  854  850   924  895   542  471   377   219];

% Bulk density
tio2_bulk_density_g_cm3 = 4;
sio2_bulk_density_g_cm3 = 2.5;


%============================================================

% Fit data at humdity compensation alphas
sample_i = [];
for count = 1:length(hc_alphas)
    curr_i = find(tio2_alphas == hc_alphas(count));
    sample_i = [sample_i curr_i];
end

sample_mass_mg = tio2_mass_mg(sample_i);
sample_footprint_area_mm2 = tio2_footprint_area_mm2(sample_i);
sample_thickness_um = tio2_thickness_um(sample_i);
sample_sa_m2_m2_um = tio2_sa_m2_m2_um(sample_i);


% Calculate change in index of refraction given the change in wavelength
 indexChange = wvlChange / 2 ./ periodicity;


disp(['For humidity measurements on TiO2 samples:']);
disp(['  At alpha = ',num2str(hc_alphas)]);
disp(['  Index change = ',num2str(0.001*round(1000*indexChange))]);


% Calculate densities
 density_g_cm3 = (sample_mass_mg/1000) ./ ...
                 ( (sample_footprint_area_mm2 / 10^2) .* (sample_thickness_um /10000));
 density_rel = density_g_cm3 ./ tio2_bulk_density_g_cm3;

% Use Tait model to estimate fraction of volume that is tio2 and what is air
 tait_model = 2 * cos(hc_alphas*pi/180) ./ (1 + cos(hc_alphas*pi/180));

 %fraction_tio2 = tait_model;
 fraction_tio2 = [0.91304     0.81     0.82074     0.68371     0.40392];

 fraction_air_zeroRH = 1 - fraction_tio2;

% Calculate the expected index of refraction for the film at zero %RH
  % Effective medium approximation
  % Bruggeman method
    % At 0 % RH
    a = -2;
    b = 2 * e_air  - e_tio2 + 3 * fraction_tio2 * e_tio2 - 3* fraction_tio2 * e_air;
    c = e_tio2 * e_air;
    e_zeroRH = (-b - sqrt(b.^2 - 4*a*c)) / (2*a);


    index_zeroRH = sqrt(e_zeroRH);


% Calculate the estimated index of refraction at 50% RH
    index_fiftyRH = index_zeroRH + indexChange;
    e_fiftyRH =index_fiftyRH.^2;

% Calculate the fraction of water in the medium
    % Effective medium approximation
```

```
    % Bruggeman method
      % At 50 % RH
      e_air_water = -(3* fraction_tio2 * e_tio2 .* e_fiftyRH - ...
                           2 * e_fiftyRH.^2 - e_fiftyRH * e_tio2) ./ ...
                     (e_tio2 - 3*fraction_tio2.*e_fiftyRH + 2*e_fiftyRH) ;

      water_term = (e_water - e_air_water) ./ (e_water + 2 * e_air_water);
      air_term = (e_air - e_air_water) ./ (e_air + 2 * e_air_water);

      fraction_water_in_voidspace = air_term ./ (air_term - water_term);
      fraction_water_fiftyRH = (1 - fraction_tio2) .* fraction_water_in_voidspace;
      fraction_air_fiftyRH = (1 - fraction_tio2) .* (1 - fraction_water_in_voidspace);


%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
% Fit of sloped line to water fraction data
% doesn't appear to work, since 30 deg data is far off
% Use only alpha > 45 deg data and just find average
% This fits ok with SA data, which shows that only films with alpha >=45
% have significant porosity
 n_points = length(hc_alphas);
 gt45 = find(hc_alphas >= 45);
 lt45 = find(hc_alphas < 45);
 p = polyfit(hc_alphas(gt45),fraction_water_fiftyRH(gt45),0);
 b_fraction = p(1);
 scatter_fraction = fraction_water_fiftyRH - b_fraction;
 uncertainty_y_fraction = sqrt(sum(scatter_fraction.^2)/(n_points));
 fit_fraction_water_fiftyRH = [fraction_water_fiftyRH(lt45) b_fraction * ones(1,length(gt45))];

 fit_fraction_air_fiftyRH = (1 - fraction_tio2 - fit_fraction_water_fiftyRH);

% Calculate sample volume
 volume_sample_cm3 = (sample_footprint_area_mm2 / 10^2) .* (sample_thickness_um * 1e-4);

% Now calculate volume of water, tio2, air
 volume_water_cm3 = fit_fraction_water_fiftyRH .* volume_sample_cm3;
 volume_tio2_cm3 = fraction_tio2 .* volume_sample_cm3;
 volume_air_cm3 = fit_fraction_air_fiftyRH .* volume_sample_cm3;


% Water adsorbed per m2
 tio2_sa_m2 = sample_sa_m2_m2_um .* (sample_footprint_area_mm2 / 1000^2) .* sample_thickness_um;

% Calculate the mass of water
density_water_g_cm3 = 1;
mass_water_mg = volume_water_cm3 * density_water_g_cm3 * 1000;

% Calculate the actual mass of film
mass_film_actual_mg = sample_mass_mg - mass_water_mg;

actual_film_density_g_cm3 = (mass_film_actual_mg/1000) ./ ...
                            ( (sample_footprint_area_mm2 / 10^2) .* (sample_thickness_um /10000));
actual_density_rel = actual_film_density_g_cm3 ./ tio2_bulk_density_g_cm3;

% How much of the original weighed mass is this
mass_water_percentage = mass_water_mg ./ sample_mass_mg * 100;

% Display this value
disp(['  Water Volume (percent) = ',num2str(100*fraction_water_fiftyRH)]);
disp(['  Fit Water Volume (percent) = ',num2str(100*fit_fraction_water_fiftyRH)]);
disp(['  Water Mass (percent) = ',num2str(0.01*round(100*mass_water_percentage))]);
disp(['  Measured Mass (mg) = ',num2str(0.01*round(100*sample_mass_mg))]);
disp(['  Water Mass (mg) = ',num2str(0.01*round(100*mass_water_mg))]);
disp(['  Actual film mass (mg) = ',num2str(0.01*round(100*mass_film_actual_mg))]);
disp(['  Measured density (rel) = ',num2str(density_rel)]);
disp(['  Predicted density (rel) = ',num2str(fraction_tio2)]);
disp(['  Actual film density (rel) = ',num2str(actual_density_rel)]);

% For humidity measurements on TiO2 samples:
% NEW: correct index of refraction
%   At alpha = 30  45  50  60  80
%   Index change = 0.034        0.07        0.06       0.059       0.044
%   Water Volume (percent) = 11.0199       21.851      18.8979      17.7126      11.8842
%   Fit Water Volume (percent) = 11.0199       21.851      18.8979      17.7126      11.8842
%   Water Mass (percent) = 2.93        6.33        5.45        6.09        6.86
%   Measured Mass (mg) = 5.56         8.69         8.18         5.68        15.09
%   Water Mass (mg) = 0.16        0.55        0.45        0.35        1.03
%   Actual film mass (mg) = 5.4         8.14        7.73        5.33        14.06
%   Measured density (rel) = 0.94043     0.86267     0.86765     0.72767     0.43333
%   Predicted density (rel) = 0.91304     0.80785     0.82074     0.68371     0.40392
%   Actual film density (rel) = 0.91288     0.80804      0.8204     0.68338     0.40362
```

```
%--------------------------------------------------------------------------
% Fit line to water mass percentage data
% TODO: Force through zero
 x = [0 hc_alphas];
 y = [0 mass_water_percentage];
 n_points = length(x);
 p = polyfit(x,y,1);
 m_mass = p(1);
 b_mass = p(2);
 scatter_mass = y - (b_mass + m_mass*x);
 uncertainty_y_mass_water_percentage = sqrt(sum(scatter_mass.^2)/(n_points));
 hc_fit_alphas = 0:1:89;
 hc_mass_water_percentage = m_mass*hc_fit_alphas + b_mass;


% Calculate water-compensated TiO2 masses from fit line
 gt45 = find(tio2_alphas >= 45);
 lt45 = find(tio2_alphas < 45);
 tio2_fraction_water_fiftyRH = [fit_fraction_water_fiftyRH(1)*ones(1,length(lt45)) ...
                                b_fraction * ones(1,length(gt45))];


 % If calculated from fit of fraction:
 % But, use fit of mass
  % tio2_volume_sample_cm3 = (tio2_footprint_area_mm2 / 10^2) .* (tio2_thickness_um * 1e-4);
  % tio2_volume_water_cm3 = tio2_fraction_water_fiftyRH .* tio2_volume_sample_cm3;
  % tio2_mass_water_mg = tio2_volume_water_cm3 * density_water_g_cm3 * 1000;

 tio2_mass_water_percentage = tio2_alphas * m_mass + b_mass;
 tio2_mass_water_mg = tio2_mass_mg .* tio2_mass_water_percentage/100;
 tio2_mass_film_actual_mg = tio2_mass_mg - tio2_mass_water_mg;
 tio2_uncertainty_mass_water_mg = tio2_mass_mg .* uncertainty_y_mass_water_percentage/100;
 tio2_meas_density_g_cm3 = (tio2_mass_mg/1000) ./ ...
                           ( (tio2_footprint_area_mm2 / 10^2) .* (tio2_thickness_um /10000));
 tio2_meas_density_rel = tio2_meas_density_g_cm3 ./ tio2_bulk_density_g_cm3;
 tio2_actual_film_density_g_cm3 = (tio2_mass_film_actual_mg/1000) ./ ...
                                  ( (tio2_footprint_area_mm2 / 10^2) .* (tio2_thickness_um /10000));
 tio2_actual_density_rel = tio2_actual_film_density_g_cm3 ./ tio2_bulk_density_g_cm3;


% Display this value
 disp(['After fitting water percentage curve, masses for TiO2 films:']);
 disp(['  Slope = ',num2str(m_mass),', intercept = ',num2str(b_mass)]);
 disp(['  For alphas ',num2str(tio2_alphas)]);
 disp(['  Water Volume (percent) = ',num2str(100*tio2_fraction_water_fiftyRH)]);
 disp(['  Water Mass (percent) = ',num2str(0.01*round(100*tio2_mass_water_percentage))]);
 disp(['  Measured Mass (mg) = ',num2str(0.01*round(100*tio2_mass_mg))]);
 disp(['  Water Mass (mg) = ',num2str(0.01*round(100*tio2_mass_water_mg))]);
 disp(['  Actual film mass (mg) = ',num2str(0.01*round(100*tio2_mass_film_actual_mg))]);
 disp(['  Uncertainties (mg) = ',num2str(0.01*round(100*tio2_uncertainty_mass_water_mg))]);
 disp(['  Measured density (rel) = ',num2str(tio2_meas_density_rel)]);
 disp(['  Actual film density (rel) = ',num2str(tio2_actual_density_rel)]);



% Plot mass water percentage
 figure('Visible','off');
 plot(hc_alphas, mass_water_percentage, 'ko',  'LineWidth', lineWidth, ...
         'MarkerSize', markerSize, 'MarkerFaceColor', 'k', 'MarkerEdgeColor', 'k');
 hold on;
 plot(hc_fit_alphas, hc_mass_water_percentage, 'k-',  'LineWidth', lineWidth, ...
        'MarkerSize', markerSize, 'MarkerFaceColor', 'k', 'MarkerEdgeColor', 'k');
 xlabel(['\alpha (^o)'],'FontSize',fontSize,'FontWeight','bold');
 ylabel(['m_{water} (%)'],'FontSize',fontSize,'FontWeight','bold');
 set(gca,'LineWidth',lineWidth,'FontSize',fontSize,'FontWeight','bold');
 print('-dtiff',['TiO2_water_mass_percentage']);
 close



%--------------------------------------------------------------------------

% Plot fraction water
 figure('Visible','off');
 plot(hc_alphas, fraction_water_fiftyRH * 100, 'ko',  'LineWidth', lineWidth, ...
         'MarkerSize', markerSize, 'MarkerFaceColor', 'k', 'MarkerEdgeColor', 'k');
 hold on;
 plot(hc_alphas, fit_fraction_water_fiftyRH * 100, 'k-',  'LineWidth', lineWidth, ...
```

```
        'MarkerSize', markerSize, 'MarkerFaceColor', 'k', 'MarkerEdgeColor', 'k');
 xlabel(['\alpha (^o)'],'FontSize',fontSize,'FontWeight','bold');
 ylabel(['V_{water} (%)'],'FontSize',fontSize,'FontWeight','bold');
 set(gca,'LineWidth',lineWidth,'FontSize',fontSize,'FontWeight','bold');
 print('-dtiff',['TiO2_water_volume_percentage']);
 close


%------------------------------------------------------------------------
% From Josh Siewart's work: Volume percentage of water adsorbed for SiO2 is
% 66/40 times more water adsorbed (by volume) for SiO2 than for TiO2
% But, SiO2 film thickness = 1.5 um (vs 1.6 um)
% And SA may be a bit larger for TiO2 than SiO2 at 80 degrees for the ~1um film
% but from the fit they are fairly similiar, so use the same SA
% density diff = 4/2.5;
% (66/40)*(1.6/1.5)*(4/2.5) = 2.82
% But, when taking into account total mass of water + film
% multiplicative factor is 2.94

 sio2_volume_sample_cm3 = (sio2_footprint_area_mm2 / 10^2) .* (sio2_thickness_um * 1e-4);

 sio2_mass_water_percentage = tio2_mass_water_percentage * 2.94;
 sio2_mass_water_mg = sio2_mass_mg .* sio2_mass_water_percentage / 100;
 sio2_volume_water_cm3 = sio2_mass_water_mg / density_water_g_cm3 / 1000;
 sio2_volume_water_percentage = sio2_volume_water_cm3 ./ sio2_volume_sample_cm3 * 100;

 sio2_mass_film_actual_mg = sio2_mass_mg - sio2_mass_water_mg;
 sio2_meas_density_g_cm3 = (sio2_mass_mg/1000) ./ ...
                           ( (sio2_footprint_area_mm2 / 10^2) .* (sio2_thickness_um /10000));
 sio2_meas_density_rel = sio2_meas_density_g_cm3 ./ sio2_bulk_density_g_cm3;
 sio2_actual_film_density_g_cm3 = (sio2_mass_film_actual_mg/1000) ./ ...
                                  ( (sio2_footprint_area_mm2 / 10^2) .* (sio2_thickness_um /10000));
 sio2_actual_density_rel = sio2_actual_film_density_g_cm3 ./ sio2_bulk_density_g_cm3;

% Display this value
disp(['Estimated SiO2 water adsorbed using 66/40 increased water adsorption ', ...
      'for SiO2 over TiO2 - from QCM plot']);
disp(['  For alphas ',num2str(sio2_alphas)]);
disp(['  Water Volume (percent) = ',num2str(sio2_volume_water_percentage)]);
disp(['  Water Mass (percent) = ',num2str(0.01*round(100*sio2_mass_water_percentage))]);
disp(['  Measured Mass (mg) = ',num2str(0.01*round(100*sio2_mass_mg))]);
disp(['  Water Mass (mg) = ',num2str(0.01*round(100*sio2_mass_water_mg))]);
disp(['  Actual film mass (mg) = ',num2str(0.01*round(100*sio2_mass_film_actual_mg))]);
disp(['  Measured density (rel) = ',num2str(sio2_meas_density_rel)]);
disp(['  Actual film density (rel) = ',num2str(sio2_actual_density_rel)]);



% Plot mass water percentage
 figure('Visible','off');
 plot(sio2_alphas, sio2_mass_water_percentage, 'ko',  'LineWidth', lineWidth, ...
        'MarkerSize', markerSize, 'MarkerFaceColor', 'k', 'MarkerEdgeColor', 'k');
 hold on;
 xlabel(['\alpha (^o)'],'FontSize',fontSize,'FontWeight','bold');
 ylabel(['m_{water} (%)'],'FontSize',fontSize,'FontWeight','bold');
 set(gca,'LineWidth',lineWidth,'FontSize',fontSize,'FontWeight','bold');
 print('-dtiff',['SiO2_water_mass_percentage']);
 close



% ------------------------------------------------------------------------
% Josh Siewart work using a QCM to detect humidity mass change
% Titania:
% At alpha = 81deg
% Adsorbed water = 0.4e-4 cc / m2
% SA of my 80deg TiO2 film =  3988 mm2 x (2.183 um)
qcm_tio2.ad_water_cc_m2 = 0.4e-4;
% Josh used 300 m2/m2 as his SAE, but he had a 1.6 um film and
% should probably have used 350 m2/m2 for an 81 deg film
qcm_tio2.correction_factor = (6.3/3.5)*(0.007^2/0.004^2)*(1.6*350/300);
qcm_tio2.volume_film_m2_um = (3988/1000^2)*(2.183);
qcm_tio2.mass_film_mg = 15.09;
qcm_tio2.sa_m2_m2_um = 350;
qcm_tio2.sa_m2 = qcm_tio2.sa_m2_m2_um * qcm_tio2.volume_film_m2_um;
qcm_tio2.volume_water_cc = (qcm_tio2.ad_water_cc_m2*qcm_tio2.correction_factor) * qcm_tio2.sa_m2;
qcm_tio2.water_mass_mg = qcm_tio2.volume_water_cc * 1000;
qcm_tio2.mass_film_actual_mg = qcm_tio2.mass_film_mg - qcm_tio2.water_mass_mg;
qcm_tio2.water_mass_percentage = qcm_tio2.water_mass_mg / qcm_tio2.mass_film_mg *100;

% Display this value
```

```
disp(['For QCM measurements:']);
disp(['  At alpha = 80 for TiO2']);
disp(['  Water Mass (mg) = ',num2str(0.01*round(100*qcm_tio2.water_mass_mg))]);
disp(['  Water Mass (percent) = ',num2str(0.01*round(100*qcm_tio2.water_mass_percentage))]);
disp(['  Actual film mass (mg) = ',num2str(0.01*round(100*qcm_tio2.mass_film_actual_mg))]);



% Silica:
% At alpha = 81deg
% Adsorbed water = 0.66e-4 cc / m2
% SA of my 80deg SiO2 film =  2580 mm2 x (0.85+0.05 um)
qcm_sio2.ad_water_cc_m2 = 0.66e-4;
% Josh used 300 m2/m2 as his SAE, but he had a 1.5 um film
% and should probably have used 320 m2/m2 for an 81 deg film
qcm_sio2.correction_factor = (6.3/3.5)*(0.007^2/0.004^2)*(1.5*350/300);
qcm_sio2.volume_film_m2_um = (2580/1000^2)*(0.9);
qcm_sio2.mass_film_mg = 2.12;
qcm_sio2.sa_m2_m2_um = 350;
qcm_sio2.sa_m2 = qcm_sio2.sa_m2_m2_um * qcm_sio2.volume_film_m2_um;
qcm_sio2.volume_water_cc = (qcm_sio2.ad_water_cc_m2*qcm_sio2.correction_factor) * qcm_sio2.sa_m2;
qcm_sio2.water_mass_mg = qcm_sio2.volume_water_cc * 1000;
qcm_sio2.mass_film_actual_mg = qcm_sio2.mass_film_mg - qcm_sio2.water_mass_mg;
qcm_sio2.water_mass_percentage = qcm_sio2.water_mass_mg / qcm_sio2.mass_film_mg *100;

% Display this value
disp(['  At alpha = 80 for SiO2']);
disp(['  Water Mass (mg) = ',num2str(0.01*round(100*qcm_sio2.water_mass_mg))]);
disp(['  Water Mass (percent) = ',num2str(0.01*round(100*qcm_sio2.water_mass_percentage))]);
disp(['  Actual film mass (mg) = ',num2str(0.01*round(100*qcm_sio2.mass_film_actual_mg))]);
```

A.2 Code to calculate Density and Surface Area of Vertical Post and Helical Films

```
%-------------------------------------------------------------------------
% Program to calculate BETs from the krypton gas adsorption isotherms
% measured for TiO2, SiO2, and ITO films.  Mostly Vertical Post films,
% but also some helical films measured.
% Error analysis is also performed.
% ITO surface area is estimated from work that Dave Rider performed
% (electrochemical analysis)
% VASE work performed by Nick Wakefield

close all;
clear all;
clc;



% -------------------------------------------------------------------------
% Data parameters
%material = 'TiO2';    % ITO or SiO2 or TiO2
%temp_K = 77;          % 77 or 87
%alpha = 85;           % 0 to 85
%gas = 'Krypton';      % Krypton or Nitrogen

rsq_limit = 0.9980;   % BET Rsq must be above this

% -------------------------------------------------------------------------
% Constants
R_J_per_K_per_mol = 8.314472;      % Constant
Na = 6.022e23;                     % Constant
q = 1.6e-19;

krypton.cross_sectional_area_nm2 = 0.202;   % Krypton - From Matthias Thommes blue book, page 80
krypton.atomic_weight_g_per_mol = 83.798;   % Krypton - CRC Handbook

tio2_bulk_density_g_cm3 = 4.0; % Anatase = 3.9,  Rutile = 4.0
sio2_bulk_density_g_cm3 = 2.5; % At alpha = 0
ito_bulk_density_g_cm3 = 7.14;    % Wikipedia

% -------------------------------------------------------------------------
```

```
% Error percentages
error_factors


% ------------------------------------------------------------------------
% Data from the VASE (from Nick)
% VASE 1 = TiO2
% VAES 2 = SiO2

vase(1).alpha =         [  30    40     50     60    70    75    80     85];
vase(1).density_g_cm3 = [ 0.91  0.91  0.82   0.72  0.52  0.44   0.41  0.28] * tio2_bulk_density_g_cm3;
vase(2).alpha =         [  30    40     50   55     70    75    80     85];
vase(2).density_g_cm3 = [ 0.98  0.96  0.89  0.78  0.54  0.44   0.36  0.19] * 2.2;

% ------------------------------------------------------------------------
% Data from Fc tests

ito_fc.alpha              = [   30    40    45    50    55    60    65    70    75    80    85    86];
ito_fc.anodicpeakarea.val_uC  = [  484  1160  1598  1777  2545  2422  3136  2488  2813  1554  1707   801];
ito_fc.error_uCmeas = 1.3;  % Percent - Dave Rider's estimate

ito_fc.sample_area.val_cm2    = [ 5.25  5.13  5.25  5.04  6.75  5.25  5.13  5.25  5.38  4.18  5.25  5.25];
ito_fc.sample_area.e_pos_cm2 = sqrt(2*(0.05^2));
% Error in sample area is 0.05cm in each dimension
ito_fc.sample_area.e_neg_cm2 = sqrt(2*(0.05^2));
% Error in sample area is 0.05cm in each dimension

ito_fc.closedpacked_coverage_mol_cm2 = 4e-10  ;
% Langmuir, 24, 2008 pg 5757, 50% coverage est by Thin Solid Films 445, 2003

ito_fc.commercial_coverage = 0.31;
ito_fc.commercial_coverage_err = 0.04;


ito_fc.cross_sectional_area_A2_molecule = 41.51;

bet_alpha86.thickness.mean_um = 0.347;
bet_alpha86.SA_m2_m2_um = 50.04;
bet_alpha86.dthickness = 0.04;   % Percent
bet_alpha86.dSA_meas_m2 = 0.12;  % Percent



% ------------------------------------------------------------------------
% Comparisons
max_structureLength = 8;
max_materialLength = 5;

poxsonCompareCounts = [1 2 3];
poxson(1).bulk_density_g_cm3 = tio2_bulk_density_g_cm3;
poxson(2).bulk_density_g_cm3 = sio2_bulk_density_g_cm3;
poxson(3).bulk_density_g_cm3 = ito_bulk_density_g_cm3;

poxsonHelicalCompareCounts = [4];
poxson_helical.bulk_density_g_cm3 = tio2_bulk_density_g_cm3;

thicknessScatterCounts = [5 6 7 8];

vaseCompareCounts = [1 2];

compare(1).material = 'TiO2';
compare(1).structure = 'VP';
compare(1).alpha = 'ALL';
compare(1).thickness_cat_um = '1um';

compare(2).material = 'SiO2';
compare(2).structure = 'VP';
compare(2).alpha = 'ALL';
compare(2).thickness_cat_um = '1um';

compare(3).material = 'ITO';
compare(3).structure = 'VP';
compare(3).alpha = 'ALL';
compare(3).thickness_cat_um = '05um';

compare(4).material = 'TiO2';
compare(4).structure = 'Helical';
compare(4).alpha = 'ALL';
compare(4).thickness_cat_um = '1um';

compare(5).material = 'TiO2';
```

```
compare(5).structure = 'VP';
compare(5).alpha = '70';
compare(5).thickness_cat_um = 'ALL';

compare(6).material = 'TiO2';
compare(6).structure = 'VP';
compare(6).alpha = '75';
compare(6).thickness_cat_um = 'ALL';

compare(7).material = 'TiO2';
compare(7).structure = 'VP';
compare(7).alpha = '80';
compare(7).thickness_cat_um = 'ALL';

compare(8).material = 'TiO2';
compare(8).structure = 'VP';
compare(8).alpha = '85';
compare(8).thickness_cat_um = 'ALL';



% ------------------------------------------------------------------------
% File specific data
plotDir = 'Plots_WaterAdComp';
dataDir = 'C:\My_Documents\Research\Research_Areas\Porosity\Analysis\Porosity_Data_Sheets';
fileExtension = 'csv';

mkdir(plotDir);
density_plotDir = [plotDir,'\SampleDensities']; mkdir(density_plotDir);
bet_plotDir = [plotDir,'\BETs']; mkdir(bet_plotDir);
iso_plotDir = [plotDir,'\Isotherms']; mkdir(iso_plotDir);
SA_plotDir = [plotDir,'\SurfaceAreas']; mkdir(SA_plotDir);
sumDen_plotDir = [plotDir,'\SummaryDensities']; mkdir(sumDen_plotDir);

fileNames = dir([dataDir,'\*.',fileExtension]);
numFiles = length(fileNames);
numCompare = length(compare);


% ------------------------------------------------------------------------
% NOTES:

% Use errors calculated from mass, footprint area, and thickness
% for density.
% These are in-line with the scatter seen in the Poxson fits and
% also in line with the standard deviation of the fit through the
% individual densities of the sample pieces


% ------------------------------------------------------------------------
% MAIN CODE:

% ------------------------------------------------------------------------
% Read in data files and do preliminary calculations
disp('Reading in data files and performing calculations...');
for fileCount = 1:numFiles
    %disp(['  ... ',num2str(fileCount),': ',fileNames(fileCount).name]);
    data = readFile([dataDir,'\',fileNames(fileCount).name]);
    foundException = 0;
    for excCount = 1:length(error_factor.exception)
        if (isequal(error_factor.exception(excCount).material, data.material) && ...
            isequal(error_factor.exception(excCount).alpha, num2str(data.alpha)) && ...
            isequal(error_factor.exception(excCount).structure, num2str(data.structure)) && ...
            isequal(error_factor.exception(excCount).thickness_cat_um, data.thickness.cat_um))

            use_error = error_factor.exception(excCount);
            foundException = 1;
        end
    end
    if (foundException == 0)
        switch data.material
            case 'TiO2'
                use_error = error_factor.tio2;
            case 'SiO2'
                use_error = error_factor.sio2;
            case 'ITO'
                use_error = error_factor.ito;
        end
    end

    % Correction of measured masses for humidity
```

```
        switch data.material
        case 'TiO2'
            humidityCorrection = -(0.08249 * data.alpha - 0.44324);

        case 'SiO2'
            humidityCorrection = -(2.94 * 0.08249 * data.alpha - 0.44324) ;


        case 'ITO'
            humidityCorrection = 0;
        end


    data = calculateFilmProperties(data, use_error, humidityCorrection, 0);
    data = calculateFilmDensity(data);
    data = calculateSurfaceAreaProperties(data, use_error, krypton, Na);
    data = findSurfaceAreas(data, rsq_limit, 'Krypton', 77);


    % From thickness study scatter
    switch data.material
        case 'TiO2'
            dSA_meas_m2 = 0.12;
        case 'SiO2'
            dSA_meas_m2 = 0.12;
        case 'ITO'
            dSA_meas_m2 = 0.12;
    end

    data.SA.dSA_meas_m2 = sqrt(data.SA.dSA_m2^2 + (data.SA.SA_m2 * dSA_meas_m2)^2);
    data.SA.dSA_pos_m2_g = data.SA.SA_m2_g * ...
                           sqrt((data.SA.dSA_meas_m2/data.SA.SA_m2)^2 + ...
                                (data.mass.e_pos_mg/data.mass.total_mg)^2);
    data.SA.dSA_neg_m2_g = data.SA.SA_m2_g * ...
                           sqrt((data.SA.dSA_meas_m2/data.SA.SA_m2)^2 + ...
                                (data.mass.e_neg_mg/data.mass.total_mg)^2);
    data.SA.dSA_pos_m2_m2 = data.SA.SA_m2_m2 * ...
                             sqrt((data.SA.dSA_meas_m2/data.SA.SA_m2)^2 + ...
                                  (data.footprint_area.e_pos_mm2/data.footprint_area.total_mm2)^2);
    data.SA.dSA_neg_m2_m2 = data.SA.SA_m2_m2 * ...
                             sqrt((data.SA.dSA_meas_m2/data.SA.SA_m2)^2 + ...
                                  (data.footprint_area.e_neg_mm2/data.footprint_area.total_mm2)^2);
    data.SA.dSA_pos_m2_m2_um = data.SA.SA_m2_m2_um * ...
                               sqrt((data.SA.dSA_pos_m2_m2/data.SA.SA_m2_m2)^2 + ...
                                    (data.thickness.e_pos_um/data.thickness.mean_um)^2);
    data.SA.dSA_neg_m2_m2_um = data.SA.SA_m2_m2_um * ...
                               sqrt((data.SA.dSA_neg_m2_m2/data.SA.SA_m2_m2)^2 +  ...
                               (data.thickness.e_neg_um/data.thickness.mean_um)^2);


    disp(['    ',data.material,', ',num2str(data.alpha),', ',data.structure,', ', ...
          num2str(data.thickness.mean_um),' +/- ',num2str(data.thickness.e_pos_um),' um, ', ...
          num2str(data.footprint_area.total_mm2),' +/- ', ...
          num2str(data.footprint_area.e_pos_mm2),' mm2, ', ...
          num2str(data.mass.total_mg),' + ',num2str(data.mass.e_pos_mg),' - ', ...
          num2str(data.mass.e_neg_mg),' mg (', ...
          num2str(data.mass.e_pos_mg / data.mass.total_mg * 100),' %), ', ...
          num2str(data.density.mean_g_cm3),' + ',num2str(data.density.e_pos_g_cm3), ...
          ' - ',num2str(data.density.e_neg_g_cm3),' g/cm^3 (', ...
          num2str(data.density.e_pos_g_cm3/data.density.mean_g_cm3 * 100),' %), SA = ', ...
          num2str(data.SA.SA_m2_m2_um),' +/- ',num2str(data.SA.dSA_pos_m2_m2_um),' m2/m2/um ', ...
          num2str(data.SA.SA_m2_g),' +/- ',num2str(data.SA.dSA_pos_m2_g),' m2/g ', ...
          ]);


    depData(fileCount) = data;

end



for fileCount = 1:numFiles
    data = depData(fileCount);

    disp([' & & ',num2str(data.alpha),' & ',num2str(round(data.thickness.mean_um*100)/100), ...
          ' $\pm$ ',num2str(round(data.thickness.e_pos_um*100)/100), ...
          ' & ',num2str(round(data.footprint_area.total_mm2/10)*10), ...
          ' $\pm$ ',num2str(round(data.footprint_area.e_pos_mm2/10)*10), ...
          ' & ',num2str(round(data.mass.total_mg*100)/100),' $\pm$ ', ...
          num2str(round(data.mass.e_pos_mg*100)/100), ...
```

```
          ' & ',num2str(round(data.density.mean_g_cm3*100)/100),' $\pm$ ', ...
          num2str(round(data.density.e_pos_g_cm3*100)/100), ...
          ' \\ cline{3-7}']);

end

% Sort data into required comparison structure
disp('Sorting data ...');
[compare] = sortData(compare,depData, numFiles, numCompare, max_structureLength, max_materialLength);



% ------------------------------------------------------------------------
% Density calculations and plots

% Plot density scatter for each experiment
disp('Plotting density scatter for samples ... ');
sampleNums = [1:1:length(depData)];
plotDensitySamples(depData, sampleNums, 5, 0.1, density_plotDir)
close all

% Calculate and plot Poxson fits
disp('Poxson fits ... ');
[poxson, poxson_scatter] = calculatePoxsonFits(compare, poxson, poxsonCompareCounts);
plotPoxsonDensity(compare, poxson, vase, poxsonCompareCounts, sumDen_plotDir);
close all


% Outcomes of TiO2, SiO2, ITO poxson fits:
% Poxson fits ...
%  Poxson model:  c = 5.3191 +/- 0.37601, Rsq = 0.87052, scatter in density = 0.19158
%  Poxson model:  c = 2.8329 +/- 0.11239, Rsq = 0.96351, scatter in density = 0.15322
%  Poxson model:  c = 8.6957 +/- 0.84855, Rsq = 0.71218, scatter in density = 0.56765

% ------------------------------------------------------------------------


%Calculate diff in VASE density
[vase] = calculateVASEdiff(compare, vase, vaseCompareCounts);



% ------------------------------------------------------------------------

% Calculate and plot Thickness study fits
disp('Scatter Calculations for TiO2 films of different thicknesses ...');
[thickness_scatter] = calculateThicknessStudyFits(compare, thicknessScatterCounts);
plotThicknessStudyDensity(thickness_scatter, sumDen_plotDir);
close all


% Outcomes of Thickness study:
% Scatter Calculations for TiO2 films of different thicknesses ...
%       70 : uy = 0.09877, m = 0.069468 +/- 0.034377, b = 2.3544 +/- 0.10845, uy/b = 0.041951
%       75 : uy = 0.13748, m = 0.064978 +/- 0.037627, b = 1.7851 +/- 0.11267, uy/b = 0.077015
%       80 : uy = 0.056501, m = 0.00025843 +/- 0.016047, b = 1.7254 +/- 0.047086, uy/b = 0.032746
%       85 : uy = 0.05253, m = 0.058144 +/- 0.014058, b = 1.0018 +/- 0.037552, uy/b = 0.052434
%       Overally uy = 0.096892, uy/b = 0.057966
% So, density scatter is approximately 3 to 8%
% This matches the density error calculated from estimating mass, footprint
% area and thickness errors


% ------------------------------------------------------------------------

% Plot density scatter as calculated by a few different methods
disp('Plotting density scatter as calculated by a few different methods ...');
plotDensityScatter(compare, poxson_scatter, thickness_scatter, vase, sumDen_plotDir)
close all


% Chiral films - calculate Poxson fits
disp('Fitting poxson model to helical data ...');
[poxson_helical, poxson_helical_quad, scatter_helical, scatter_helical_quad] = ...
            calculatePoxsonFits_Helical(compare, poxson_helical, poxsonHelicalCompareCounts);
plotHelicalDensities(compare, poxson_helical, poxson_helical_quad, ...
            poxsonHelicalCompareCounts, sumDen_plotDir);

close all
```

```
% ------------------------------------------------------------------------
% Outcomes of Helical Poxson fit:
% Fitting poxson model to helical data ...
%  Poxson model:  c = 5.4348 +/- 0.40968, Rsq = 0.94764, scatter in density  = 0.16805

%  Modified Poxson model:  b = 0.04765, c = 8.1661, Rsq = 0.9959, scatter in density = 0.26107
%     K = 1/b


% ------------------------------------------------------------------------
% Surface area calculations and plots

disp('Plotting isotherms ...');
plotIsotherms(depData, iso_plotDir);
close all

disp('Plotting BETs... ');
plotBETs(depData, bet_plotDir);
close all

disp('Surface area error calculations ...');

[thickness_scatter] = calculateThicknessStudySAFits(compare, thicknessScatterCounts);
plotThicknessStudySurfaceAreas(thickness_scatter, rsq_limit, SA_plotDir);  %compare, rsq_limit, plotDir)
close all


% ------------------------------------------------------------------------
% TiO2 thickness study
% Outcomes:
%  ... SA (m2/g)
%       70 : uy = 15.8732, b = 276.0946, uy/b = 0.057492
%       75 : uy = 17.7947, b = 252.7483, uy/b = 0.070405
%       80 : uy = 27.8892, b = 211.0875, uy/b = 0.13212
%       85 : uy = 26.5596, b = 199.9231, uy/b = 0.13285
%       Overally uy = 24.8722, uy/b = 0.11582
%  ... SA (m2/m2/um)
%       70 : uy = 77.3556, b = 691.5753, uy/b = 0.11185
%       75 : uy = 56.3643, b = 481.7616, uy/b = 0.117
%       80 : uy = 54.0691, b = 358.4832, uy/b = 0.15083
%       85 : uy = 32.5372, b = 219.6774, uy/b = 0.14811
%       Overally uy = 58.3381, uy/b = 0.14466
%  ... SA (m2/m2)
%       70 : uy = 128.0642, m = 731.5457 +/- 34.6562, b = -7.0921 +/- 94.6813, uy/b = -18.0573
%       75 : uy = 63.2322, m = 488.6193 +/- 14.9653, b = -3.0995 +/- 40.0807, uy/b = -20.4006
%       80 : uy = 67.5092, m = 318.0174 +/- 16.4676, b = 12.2834 +/- 43.2195, uy/b = 5.496
%       85 : uy = 51.9673, m = 235.1741 +/- 12.3936, b = -9.2126 +/- 30.2219, uy/b = -5.6409
%       Overally uy = 83.1947, uy/b = 14.3897
%
% TiO2 VP errors (due to mass, footprint area, thickness errors + BET instrument error):
% are about 10-15%
%
% Use SA (m2/g) thickness study uncertainty to estimate the uncertainty in
% the SA (m2) error due to changing testing conditions (e.g. humidity,
% pressure, bath temperature, degas conditions, aging etc...)
%
% dSA_meas_m2 = 12% as seen above  (uy/b = 12%)
%
% Use this value for TiO2, SiO2 AND ITO films
%
% For ITO films: a=65 and a=75 have two films that were grown on different
% days.  a=65 films have different densities, but similar SAs
% s=75 films have different densities and different SAs.  Average
% delta in SA is on par with 12%, so use this value for ITO films too
%
% a = 65: SA = 31.5 vs 36.2 m2/g,  187.2 vs 185.7 m2/m2/um
% a = 75: SA = 36.9 vs 48.6 m2/g,  163.8 vs 191.4 m2/m2/um
% ** Using the data for the a=65 and a=75 films that were used to test
%    the Ferroscene work (deposited in Jan 2009)



% ------------------------------------------------------------------------
disp('Plotting Surface Areas...');
plotSurfaceAreas(compare, rsq_limit, SA_plotDir);
plotHelicalSurfaceAreas(compare, rsq_limit, SA_plotDir);
close all

% ------------------------------------------------------------------------
% ITO: BET vs Fc
```

```
[ito_fc bet_alpha86] = calculateITOFcCoverage(ito_fc, compare(3).parameters, bet_alpha86, Na, q);
plotITOSurfaceAreas(compare(3).parameters, ito_fc, bet_alpha86, rsq_limit, SA_plotDir);
close all

% ------------------------------------------------------------------------
% All done
disp('Done');
close all;



%==========================================================================

function [data] = calculateFilmProperties(data, error_percent, humidityCorrection, thicknessCorrection)

    % Calculation of total film mass
    data.sample.filmMass_mg.nom = (data.sample.finalMass_mg - data.sample.initMass_mg) * ...
                                  (1 + humidityCorrection/100);
    data.mass.total_mg = sum(data.sample.selected .* data.sample.filmMass_mg.nom);

    % Calculation of mass errors - repeatibility for positive error
    % 2 * mass^2 because there are two mass measurements - one of the
    % substrate before deposition and one after (with the film on it)
    data.sample.filmMass_mg.e_pos = ones(1,data.sampleCounter)* ...
                                    sqrt( 2*(error_percent.mass_repeatibility_mg^2) );


    % Calculation of total mass error
    % Take into account water adsorbed into sample
    data.mass.e_pos_mg = sqrt( sum((data.sample.selected .* data.sample.filmMass_mg.e_pos ).^2 ) + ...
                               (data.mass.total_mg * error_percent.mass_humidity / 100).^2);

    data.mass.e_neg_mg = data.mass.e_pos_mg;


    % Calculation of total footprint area
    data.footprint_area.total_mm2 = sum(data.sample.selected .* data.sample.footprintArea_mm2.nom);

    % Calculation of footprint area errors
    data.sample.footprintArea_error_mm2.e_pos = data.sample.footprintArea_mm2.nom * ...
                                                error_percent.area_repeatibility / 100;
    data.sample.footprintArea_error_mm2.e_neg = data.sample.footprintArea_mm2.nom * ...
                                                error_percent.area_repeatibility / 100;

    % Calculation of total footprint area error
    data.footprint_area.e_pos_mm2 = sqrt(sum( (data.sample.selected .* ...
                                        data.sample.footprintArea_error_mm2.e_pos ).^2 ));
    data.footprint_area.e_neg_mm2 = data.footprint_area.e_pos_mm2;



    % Correct thickness with VASE correction (necessary for SiO2 samples)
    data.thickness.mean_um = data.thickness.mean_um + thicknessCorrection;


    % Calculation of thickness error
    data.thickness.e_pos_um = sqrt(data.thickness.error_um.^2 + ...
                                   (data.thickness.mean_um * ...
                                        error_percent.thickness_variation / 100).^2);
    data.thickness.e_neg_um =  data.thickness.e_pos_um;



%==========================================================================

function [data] = calculateFilmDensity(data)

    % Calculation of density
    data.density.mean_g_cm3 = (data.mass.total_mg/1000) / ...
                              (data.footprint_area.total_mm2/10^2) / ...
                              (data.thickness.mean_um/10^4);

    % Calculation of density error
    data.density.e_pos_g_cm3 = data.density.mean_g_cm3 * ...
                                 sqrt( (data.mass.e_pos_mg/data.mass.total_mg)^2 + ...
                                 (data.footprint_area.e_pos_mm2/data.footprint_area.total_mm2)^2 + ...
                                       (data.thickness.e_pos_um/data.thickness.mean_um)^2 );
    data.density.e_neg_g_cm3 = data.density.mean_g_cm3 * ...
                                 sqrt( (data.mass.e_neg_mg/data.mass.total_mg)^2 + ...
```

```
                                (data.footprint_area.e_neg_mm2/data.footprint_area.total_mm2)^2 + ...
                                    (data.thickness.e_neg_um/data.thickness.mean_um)^2 );

    % Calculation of density scatter
    data.sample.density_g_cm3 = (data.sample.filmMass_mg.nom/1000) ./ ...
                                (data.sample.footprintArea_mm2.nom/10^2) / ...
                                (data.thickness.mean_um/10^4);
    data.density.stddev_g_cm3 = std(data.sample.density_g_cm3(find(data.sample.selected==1)));
    data.density.max_g_cm3 = max(data.sample.density_g_cm3(find(data.sample.selected==1)));
    data.density.min_g_cm3 = min(data.sample.density_g_cm3(find(data.sample.selected==1)));
    data.density.delta_g_cm3 = max( (data.density.max_g_cm3 - data.density.mean_g_cm3), ...
                                    (data.density.mean_g_cm3 - data.density.min_g_cm3) );
    data.density.numstd = floor(data.density.delta_g_cm3/data.density.stddev_g_cm3);


%=========================================================================

function [data] = calculateSurfaceAreaProperties(data, error_percent, element, Na)

    for isoCounter = 1:data.isothermCounter

        data.isotherm(isoCounter).p_mmHg = data.isotherm(isoCounter).p_po * ...
                                           data.isotherm(isoCounter).po_mmHg;
        data.isotherm(isoCounter).y = data.isotherm(isoCounter).p_mmHg ./ ...
                                      (data.isotherm(isoCounter).vol_cc .* ...
                                      (data.isotherm(isoCounter).po_mmHg - ...
                                           data.isotherm(isoCounter).p_mmHg));


        ranges = [[0.05 0.1];[0.05 0.2];[0.05 0.3]];
        numRanges = size(ranges,1);

        for rngCounter = 1:numRanges
            tmp = find(data.isotherm(isoCounter).p_po >= ranges(rngCounter,1));
            if (isempty(tmp))
                % Error!
                index1 = 1;
            else
                index1 = tmp(1);
            end
            tmp = find(data.isotherm(isoCounter).p_po >= ranges(rngCounter,2));
            if (isempty(tmp))
                % Error!
                index2 = length(data.isotherm(isoCounter).p_mmHg);
            else
                index2 = tmp(1);
            end

            data.isotherm(isoCounter).p_po_1(rngCounter) = data.isotherm(isoCounter).p_po(index1);
            data.isotherm(isoCounter).p_po_2(rngCounter) = data.isotherm(isoCounter).p_po(index2);
            data.isotherm(isoCounter).p1_mmHg(rngCounter) = data.isotherm(isoCounter).p_mmHg(index1);
            data.isotherm(isoCounter).p2_mmHg(rngCounter) = data.isotherm(isoCounter).p_mmHg(index2);

            n = index2 - index1 + 1;
            x = data.isotherm(isoCounter).p_po(index1:index2);
            y = data.isotherm(isoCounter).y(index1:index2);
            p = polyfit(x,y,1);
            m = p(1);
            b = p(2);


            y2 = m*x + b;

            %figure;plot(x,y,'bo');hold on;
            %plot(x,y2,'r-');


            xsq = x.^2;
            xy = x .* y;


            yerr = y - y2;
            yerrsq = yerr.^2;

            sum_x = sum(x);
            sum_xsq = sum(xsq);
            sum_xy = sum(xy);

            dy = sqrt(sum(yerrsq)/n);

            db = dy * sqrt( sum_xsq / (n*sum_xsq - sum_x.^2));
```

```
            dm = dy * sqrt( n / (n*sum_xsq - sum_x.^2));

            c = 1 + m / b;
            Vml = 1/(c*b);
            nml = Vml / 22400;

            data.isotherm(isoCounter).SA_m2(rngCounter) = nml * Na * ...
                                        element.cross_sectional_area_nm2 * 1e-18;
            data.isotherm(isoCounter).dSA_m2(rngCounter) = data.isotherm(isoCounter).SA_m2(rngCounter) * ...
                                                    sqrt(db^2 + dm^2) / (b+m);

            data.isotherm(isoCounter).rsq(rngCounter) = 1 - sum((y - y2).^2) / sum((y - mean(y)).^2);


            data.isotherm(isoCounter).SA_m2_g(rngCounter) = ...
                     data.isotherm(isoCounter).SA_m2(rngCounter) ./ (data.mass.total_mg/1000);
            data.isotherm(isoCounter).SA_m2_m2(rngCounter) = ...
                     data.isotherm(isoCounter).SA_m2(rngCounter) ./ ...
                         (data.footprint_area.total_mm2/1000^2);
            data.isotherm(isoCounter).SA_m2_m2_um(rngCounter) = ...
                     data.isotherm(isoCounter).SA_m2(rngCounter) ./ ...
                           (data.footprint_area.total_mm2/1000^2) ./ ...
                           (data.thickness.mean_um);

            data.isotherm(isoCounter).dSA_pos_m2_g(rngCounter) =  ...
                     data.isotherm(isoCounter).SA_m2_g(rngCounter) * ...
                           sqrt( ...
                             (data.isotherm(isoCounter).dSA_m2(rngCounter)/ ...
                                    data.isotherm(isoCounter).SA_m2(rngCounter))^2 + ...
                             (data.mass.e_pos_mg / data.mass.total_mg)^2);
            data.isotherm(isoCounter).dSA_neg_m2_g(rngCounter) =  ...
                    data.isotherm(isoCounter).SA_m2_g(rngCounter) * ...
                           sqrt( ...
                             (data.isotherm(isoCounter).dSA_m2(rngCounter)/ ...
                                  data.isotherm(isoCounter).SA_m2(rngCounter))^2 + ...
                             (data.mass.e_neg_mg / data.mass.total_mg)^2);

            data.isotherm(isoCounter).dSA_pos_m2_m2(rngCounter) = ...
                    data.isotherm(isoCounter).SA_m2_m2(rngCounter) * ...
                          sqrt( ...
                             (data.isotherm(isoCounter).dSA_m2(rngCounter)/ ...
                                  data.isotherm(isoCounter).SA_m2(rngCounter))^2 + ...
                             (data.footprint_area.e_pos_mm2 / data.footprint_area.total_mm2)^2);

            data.isotherm(isoCounter).dSA_neg_m2_m2(rngCounter) =  ...
                    data.isotherm(isoCounter).SA_m2_m2(rngCounter) * ...
                         sqrt( ...
                             (data.isotherm(isoCounter).dSA_m2(rngCounter)/ ...
                                  data.isotherm(isoCounter).SA_m2(rngCounter))^2 + ...
                             (data.footprint_area.e_neg_mm2 / data.footprint_area.total_mm2)^2);

            data.isotherm(isoCounter).dSA_pos_m2_m2_um(rngCounter) =  ...
                   data.isotherm(isoCounter).SA_m2_m2_um(rngCounter) * ...
                         sqrt( ...
                             (data.isotherm(isoCounter).dSA_pos_m2_m2(rngCounter)/ ...
                                  data.isotherm(isoCounter).SA_m2_m2(rngCounter))^2 + ...
                             (data.thickness.e_pos_um / data.thickness.mean_um)^2);

            data.isotherm(isoCounter).dSA_neg_m2_m2_um(rngCounter) =  ...
                   data.isotherm(isoCounter).SA_m2_m2_um(rngCounter) * ...
                         sqrt( ...
                              (data.isotherm(isoCounter).dSA_neg_m2_m2(rngCounter)/ ...
                                  data.isotherm(isoCounter).SA_m2_m2(rngCounter))^2 + ...
                              (data.thickness.e_neg_um / data.thickness.mean_um)^2);

        end
    end


%=========================================================================

function [data] = findSurfaceAreas(data, rsq_limit, gas_match, temp_match);

    % Use the first 77K Krypton isotherm to calculate surface area from
    found_77K_Kr_isotherm = 0;

    for currIsotherm = 1:data.isothermCounter
        if (isequal(data.isotherm(currIsotherm).gas, gas_match) && ...
           (data.isotherm(currIsotherm).temp_K == temp_match) && (found_77K_Kr_isotherm == 0))
            found_77K_Kr_isotherm = 1;
```

```
            numRange =  length(data.isotherm(currIsotherm).rsq);

            if (data.isotherm(currIsotherm).rsq(numRange) >= rsq_limit)
                % Use range from 0.05 to 0.3 po
                data.SA.range = numRange;
            else
                [maxVal, maxRange] = max(data.isotherm(currIsotherm).rsq);
                data.SA.range = maxRange;
            end



            data.SA.rsq = data.isotherm(currIsotherm).rsq(data.SA.range);
            data.SA.p_po_1 = data.isotherm(currIsotherm).p_po_1(data.SA.range);
            data.SA.p_po_2 = data.isotherm(currIsotherm).p_po_2(data.SA.range);
            data.SA.p1_mmHg = data.isotherm(currIsotherm).p1_mmHg(data.SA.range);
            data.SA.p2_mmHg = data.isotherm(currIsotherm).p2_mmHg(data.SA.range);
            data.SA.SA_m2 = data.isotherm(currIsotherm).SA_m2(data.SA.range);
            data.SA.dSA_m2 = data.isotherm(currIsotherm).dSA_m2(data.SA.range);
            data.SA.SA_m2_g = data.isotherm(currIsotherm).SA_m2_g(data.SA.range);
            data.SA.dSA_pos_m2_g = data.isotherm(currIsotherm).dSA_pos_m2_g(data.SA.range);
            data.SA.dSA_neg_m2_g = data.isotherm(currIsotherm).dSA_neg_m2_g(data.SA.range);
            data.SA.SA_m2_m2 = data.isotherm(currIsotherm).SA_m2_m2(data.SA.range);
            data.SA.dSA_pos_m2_m2 = data.isotherm(currIsotherm).dSA_pos_m2_m2(data.SA.range);
            data.SA.dSA_neg_m2_m2 = data.isotherm(currIsotherm).dSA_neg_m2_m2(data.SA.range);
            data.SA.SA_m2_m2_um = data.isotherm(currIsotherm).SA_m2_m2_um(data.SA.range);
            data.SA.dSA_pos_m2_m2_um = data.isotherm(currIsotherm).dSA_pos_m2_m2_um(data.SA.range);
            data.SA.dSA_neg_m2_m2_um = data.isotherm(currIsotherm).dSA_neg_m2_m2_um(data.SA.range);
        end
    end


%=========================================================================

function [vase] = calculateVASEdiff(compare, vase, vaseCompareCounts)
    alphaStep = 5;

    poxsonCount = 0;
    for compareCount = vaseCompareCounts
        disp(['compareCount = ',num2str(compareCount)]);
        for vaseCount = 1:length(vase(compareCount).alpha)
            vaseAlpha = vase(compareCount).alpha(vaseCount);
            for measCount = 1:length(compare(compareCount).data)
                measAlpha = compare(compareCount).data(measCount).alpha;
                if (vaseAlpha == measAlpha)
                    disp(['   Meas alpha = ',num2str(vaseAlpha)]);
                    measDensity = compare(compareCount).parameters.density.mean_g_cm3(measCount);
                    vaseDensity = vase(compareCount).density_g_cm3(vaseCount);
                    vase(compareCount).scatter_g_cm3(vaseCount) = abs(measDensity - vaseDensity);
                end
            end
        end
    end


%=========================================================================


function [thickness_scatter] = calculateThicknessStudyFits(compare, thicknessScatterCounts)

    counter = 0;
    for compareCount = thicknessScatterCounts
        counter = counter + 1;
        thickness_scatter.alphas(counter) = [str2num(compare(compareCount).alpha)];
        thickness_scatter.entry(counter).x = compare(compareCount).parameters.thickness.mean_um;
        thickness_scatter.entry(counter).y = compare(compareCount).parameters.density.mean_g_cm3;
        thickness_scatter.entry(counter).y_e_pos = compare(compareCount).parameters.density.e_pos_g_cm3;
        thickness_scatter.entry(counter).y_e_neg = compare(compareCount).parameters.density.e_neg_g_cm3;

    end
    forceIntercept_to_Mean = 0;
    [thickness_scatter] = scatter_calcs(thickness_scatter, forceIntercept_to_Mean);



%=========================================================================

function [thickness_scatter] = calculateThicknessStudySAFits(compare, thicknessScatterCounts)
```

```
    disp(' ... SA (m2/g)');
    counter = 0;
    for compareCount = thicknessScatterCounts
        counter = counter + 1;
        thickness_scatter.SA_m2_g.alphas(counter) = [str2num(compare(compareCount).alpha)];
        thickness_scatter.SA_m2_g.entry(counter).x = compare(compareCount).parameters.thickness.mean_um;
        thickness_scatter.SA_m2_g.entry(counter).y = compare(compareCount).parameters.SA_m2_g;
        thickness_scatter.SA_m2_g.entry(counter).y_e_pos = compare(compareCount).parameters.dSA_pos_m2_g;
        thickness_scatter.SA_m2_g.entry(counter).y_e_neg = compare(compareCount).parameters.dSA_pos_m2_g;

    end
    forceIntercept_to_Mean = 1;
    [thickness_scatter.SA_m2_g] = scatter_calcs(thickness_scatter.SA_m2_g, forceIntercept_to_Mean);

    disp(' ... SA (m2/m2/um)');
    counter = 0;
    for compareCount = thicknessScatterCounts
        counter = counter + 1;
        thickness_scatter.SA_m2_m2_um.alphas(counter) = [str2num(compare(compareCount).alpha)];
        thickness_scatter.SA_m2_m2_um.entry(counter).x = ...
              compare(compareCount).parameters.thickness.mean_um;
        thickness_scatter.SA_m2_m2_um.entry(counter).y = ...
              compare(compareCount).parameters.SA_m2_m2_um;
        thickness_scatter.SA_m2_m2_um.entry(counter).y_e_pos = ...
              compare(compareCount).parameters.dSA_pos_m2_m2_um;
        thickness_scatter.SA_m2_m2_um.entry(counter).y_e_neg = ...
              compare(compareCount).parameters.dSA_pos_m2_m2_um;

    end
    forceIntercept_to_Mean = 1;
    [thickness_scatter.SA_m2_m2_um] = scatter_calcs(thickness_scatter.SA_m2_m2_um, forceIntercept_to_Mean);

    disp(' ... SA (m2/m2)');
    counter = 0;
    for compareCount = thicknessScatterCounts
        counter = counter + 1;
        thickness_scatter.SA_m2_m2.alphas(counter) = [str2num(compare(compareCount).alpha)];
        thickness_scatter.SA_m2_m2.entry(counter).x = ...
                 [0 compare(compareCount).parameters.thickness.mean_um];
        thickness_scatter.SA_m2_m2.entry(counter).y = ...
                 [1 compare(compareCount).parameters.SA_m2_m2];
        thickness_scatter.SA_m2_m2.entry(counter).y_e_pos = ...
                 [5 compare(compareCount).parameters.dSA_pos_m2_m2];
        thickness_scatter.SA_m2_m2.entry(counter).y_e_neg = ...
                 [5 compare(compareCount).parameters.dSA_pos_m2_m2];

    end
    [thickness_scatter.SA_m2_m2] = scatter_calcs_weighted_error(thickness_scatter.SA_m2_m2);
    %[thickness_scatter.SA_m2_m2] = scatter_calcs(thickness_scatter.SA_m2_m2,0);



%=========================================================================

function [fitData] = scatter_calcs(fitData, forceMean)

x70 = fitData.entry(1).x;
x75 = fitData.entry(2).x;
x80 = fitData.entry(3).x;
x85 = fitData.entry(4).x;
y70 = fitData.entry(1).y;
y75 = fitData.entry(2).y;
y80 = fitData.entry(3).y;
y85 = fitData.entry(4).y;


n_points70 = length(x70);
n_points75 = length(x75);
n_points80 = length(x80);
n_points85 = length(x85);

xsq70 = x70.^2;
xsq75 = x75.^2;
xsq80 = x80.^2;
xsq85 = x85.^2;

if forceMean
    m70 = 0;
    m75 = 0;
    m80 = 0;
```

```
    m85 = 0;
    b70 = mean(y70);
    b75 = mean(y75);
    b80 = mean(y80);
    b85 = mean(y85);
else

    p70 = polyfit(x70,y70,1);
    p75 = polyfit(x75,y75,1);
    p80 = polyfit(x80,y80,1);
    p85 = polyfit(x85,y85,1);

    m70 = p70(1);
    m75 = p75(1);
    m80 = p80(1);
    m85 = p85(1);

    b70 = p70(2);
    b75 = p75(2);
    b80 = p80(2);
    b85 = p85(2);
end

scatter70 = y70 - (b70 + m70*x70);
scatter75 = y75 - (b75 + m75*x75);
scatter80 = y80 - (b80 + m80*x80);
scatter85 = y85 - (b85 + m85*x85);


uncertainty_y70 = sqrt(sum(scatter70.^2)/(n_points70));%-2));
uncertainty_y75 = sqrt(sum(scatter75.^2)/(n_points75));%-2));
uncertainty_y80 = sqrt(sum(scatter80.^2)/(n_points80));%-2));
uncertainty_y85 = sqrt(sum(scatter85.^2)/(n_points85));%-2));

del70 = n_points70 * sum(xsq70) - (sum(x70)).^2;
del75 = n_points75 * sum(xsq75) - (sum(x75)).^2;
del80 = n_points80 * sum(xsq80) - (sum(x80)).^2;
del85 = n_points85 * sum(xsq85) - (sum(x85)).^2;

uncertainty_b70 = uncertainty_y70 * sqrt( sum(xsq70) / del70 );
uncertainty_b75 = uncertainty_y75 * sqrt( sum(xsq75) / del75 );
uncertainty_b80 = uncertainty_y80 * sqrt( sum(xsq80) / del80 );
uncertainty_b85 = uncertainty_y85 * sqrt( sum(xsq85) / del85 );

uncertainty_m70 = uncertainty_y70 * sqrt( n_points70 / del70 );
uncertainty_m75 = uncertainty_y75 * sqrt( n_points75 / del75 );
uncertainty_m80 = uncertainty_y80 * sqrt( n_points80 / del80 );
uncertainty_m85 = uncertainty_y85 * sqrt( n_points85 / del85 );

fitData.m = [m70 m75 m80 m85];
fitData.b = [b70 b75 b80 b85];
fitData.scatter_y = [max(scatter70) max(scatter75) max(scatter80) max(scatter85)];
fitData.unc_y = [uncertainty_y70 uncertainty_y75 uncertainty_y80 uncertainty_y85];
fitData.unc_m = [uncertainty_m70 uncertainty_m75 uncertainty_m80 uncertainty_m85];
fitData.unc_b = [uncertainty_b70 uncertainty_b75 uncertainty_b80 uncertainty_b85];


n_points = n_points70 + n_points75 + n_points80 + n_points85;

uncertainty_y_overall = sqrt((sum(scatter70.^2) + sum(scatter75.^2) + ...
                              sum(scatter80.^2) + sum(scatter85.^2))/(n_points - 2));
uncertainty_yb_overall = sqrt((sum((scatter70/b70).^2) + ...
                               sum((scatter75/b75).^2) + sum((scatter80/b80).^2) + ...
                               sum((scatter85/b85).^2))/(n_points - 2));

if forceMean
    disp(['      70 : uy = ',num2str(uncertainty_y70),', b = ', ...
          num2str(b70),', uy/b = ',num2str(uncertainty_y70/b70)]);
    disp(['      75 : uy = ',num2str(uncertainty_y75),', b = ', ...
          num2str(b75),', uy/b = ',num2str(uncertainty_y75/b75)]);
    disp(['      80 : uy = ',num2str(uncertainty_y80),', b = ', ...
          num2str(b80),', uy/b = ',num2str(uncertainty_y80/b80)]);
    disp(['      85 : uy = ',num2str(uncertainty_y85),', b = ', ...
          num2str(b85),', uy/b = ',num2str(uncertainty_y85/b85)]);
    disp(['      Overally uy = ',num2str(uncertainty_y_overall), ...
         ', uy/b = ',num2str(uncertainty_yb_overall)]);
else
    disp(['      70 : uy = ',num2str(uncertainty_y70), ...
          ', m = ',num2str(m70),' +/- ',num2str(uncertainty_m70), ...
          ', b = ',num2str(b70),' +/- ',num2str(uncertainty_b70), ...
          ', uy/b = ',num2str(uncertainty_y70/b70)]);
```

```
    disp(['      75 : uy = ',num2str(uncertainty_y75), ...
          ', m = ',num2str(m75),' +/- ',num2str(uncertainty_m75), ...
          ', b = ',num2str(b75),' +/- ',num2str(uncertainty_b75), ...
          ', uy/b = ',num2str(uncertainty_y75/b75)]);
    disp(['      80 : uy = ',num2str(uncertainty_y80), ...
          ', m = ',num2str(m80),' +/- ',num2str(uncertainty_m80), ...
          ', b = ',num2str(b80),' +/- ',num2str(uncertainty_b80), ...
          ', uy/b = ',num2str(uncertainty_y80/b80)]);
    disp(['      85 : uy = ',num2str(uncertainty_y85), ...
          ', m = ',num2str(m85),' +/- ',num2str(uncertainty_m85), ...
          ', b = ',num2str(b85),' +/- ',num2str(uncertainty_b85), ...
          ', uy/b = ',num2str(uncertainty_y85/b85)]);
    disp(['      Overally uy = ',num2str(uncertainty_y_overall), ...
          ', uy/b = ',num2str(uncertainty_yb_overall)]);
end


%=========================================================================
```

A.3 Code to estimate Density and Surface Area of Bragg and Rugate stacks

```
% Code to estimate the density and surface area of stacked films
% (ie. Bragg and Rugate stacks) to see whether the weighted
% averages of density and SA match the measured parameters

close all;
clear all;
clc;
warning off;

plotParam.fontSize = 20;
plotParam.lineWidth = 2;
plotParam.markerSize = 10;
plotParam.colours = 'brmgckbrmgckbrmgckbrmgck';
plotParam.shapes = 'svodx>+hsvodx>+hsvodx>';
plotParam.lineStyles = {'-','--',':','-.','-','--',':','-.','-','--',':','-.','-','--',':','-.'};

inputDir = 'Data';
outputDir = 'Plots'; mkdir(outputDir);
doCalPlots = 0;


% Use average percent errors from previous
% density and SA work on Vertical Post films
percent_error_thickness = 5;
percent_error_area = 4;
percent_error_mass = 6;
percent_error_density_g_cm3 = 7;  % mean(ddensity_g_cm3 ./ density_g_cm3)
percent_error_thickness_ratio = sqrt(2*percent_error_thickness^2);


% Two test cases:
%   1) Rugate films
%   2) Bragg stacks


testCase = 2;

switch testCase
    case 1
        testType = 'Rugate';
        tags = {'30to80','50to80','60to80'}; %{'30to80','45to80','50to80','60to80'};
        xaxis_nums = [30, 50, 60];
        xaxis_text = ['\alpha (^o)'];
        axisLimits_density = [0, 80, 1.5, 3.5];
        axisLimits_relSA = [0, 80, 300, 800];
        axisLimits_specificSA = [0, 80, 100, 350];
```

```
        depFiles = {'rugate30-80_14.txt', ...
                    ... %'rugate_45_80_14_new.txt', ...
                    'rugate_50_80_14_new.txt', ...
                    'rugate_60_80_14_new.txt'};

        betFiles = {'2008_10_03_TiO2_Rugate_a30to80Sarah_July2908_77K_BET.raw', ...
                    ... %'2008_11_10_TiO2_a45to80Rugate_77K_BET.raw', ...
                    '2008_11_26_TiO2_Rugate_a50to80_Aug2608_77K_adData.raw', ...
                    '2008_11_25_TiO2_Rugate_a60to80_Aug2608_77K_adData.raw'};

        meas.mass_mg =             [7.50  14.64  13.03]; %[7.50  14.19  14.64  13.03];
        meas.dmass_mg =            meas.mass_mg .* percent_error_mass/100;
        meas.footprint_area_mm2 =  [1393   2749   2956   3022]; %[1393   2749   2956   3022];
        meas.dfootprint_area_mm2 = meas.footprint_area_mm2 .* percent_error_area/100;
        meas.thickness_um =        [1.95   1.82   1.87   1.77]; %[1.95   1.82   1.87   1.77];
        meas.dthickness_um =       meas.thickness_um .* percent_error_thickness/100;

        interp_points = 2;

    case 2
        testType = 'Bragg';
        tags = {'1','2'};
        xaxis_nums = [1, 2];
        xaxis_text = '#';
        axisLimits_density = [0, 3, 1.5, 3.5];
        axisLimits_relSA = [0, 3, 300, 800];
        axisLimits_specificSA = [0, 3, 100, 350];


        depFiles = {'bragg_60_80_12_v1.txt', ...
                    'bragg_60_80_12_v2.txt'};

        betFiles = {'2009_03_11_TiO2_Bragg1_a60to80_77K_adData.raw', ...
                    '2009_03_12_TiO2_Matt_Bragg2_a60to80_77_adData.raw'};

        meas.mass_mg =             [15.33 13.71];
        meas.dmass_mg =            meas.mass_mg * percent_error_mass/100;
        meas.footprint_area_mm2 =  [2688 2689];
        meas.dfootprint_area_mm2 = meas.footprint_area_mm2 .* percent_error_area/100;
        meas.thickness_um =        [2.24 2.20];
        meas.dthickness_um =       meas.thickness_um .* percent_error_thickness/100;

        interp_points = 0;
end


tio2_bulk_density_g_cm3 = 4.0;
po_mmHg = 2.63;
dsa_thicknessStudy_m2 = 0.12;  % From thickness study



numFiles = length(tags);

% Read in dep file
disp('Reading in dep files');
for count = 1:numFiles
    fileName = [inputDir,'\',char(depFiles(count))];
    depData(count) = readDepFile(fileName);
end

% Read in BET data
disp('Reading in BET data');
for count = 1:numFiles
    fileName = [inputDir,'\',char(betFiles(count))];
    betData(count) = readBETFile(fileName);
end


% Establish calibration tables
disp('Establishing calibration data');
[cal.depRatio] = find_depRatios(doCalPlots, outputDir, plotParam);
[cal.sa] = find_SAcal(doCalPlots, outputDir, plotParam);
[cal.hc] = find_humidityCal(doCalPlots, outputDir, plotParam);
[cal.density] = find_densityCal(tio2_bulk_density_g_cm3, doCalPlots, outputDir, plotParam);



disp('Processing data for each file');
for count = 1:numFiles
    disp([' ',char(tags(count))]);
```

```
%-------------------------------------------------------------
% Use gradually changing alpha to more finely divide film
% into small thickness slices

 if (interp_points > 1)
    disp('   Interpolating deposition file data to more steps');
    nsteps_orig = length(depData(count).thickness_nm);
    steps = [1:1:nsteps_orig];
    spline_steps = [1/interp_points:1/interp_points:nsteps_orig];

    depData(count).interp_thickness_nm = spline(steps,depData(count).thickness_nm,spline_steps);
    depData(count).interp_alpha = spline(depData(count).thickness_nm, ...
                                  depData(count).alpha,depData(count).interp_thickness_nm);
    depData(count).interp_phi = spline(depData(count).thickness_nm, ...
                                  depData(count).phi,depData(count).interp_thickness_nm);
 else
    depData(count).interp_thickness_nm = depData(count).thickness_nm;
    depData(count).interp_alpha = depData(count).alpha;
    depData(count).interp_phi = depData(count).phi;
 end

 % Calculate the change in thickness of each small slice
 depData(count).delta_thickness_nm = depData(count).interp_thickness_nm(2:end) - ...
                                     depData(count).interp_thickness_nm(1:end-1);
 expected.delta(count).alpha = depData(count).interp_alpha(1:end-1);


 %-------------------------------------------------------------
 % For each small slice calculate the percentage water (by mass),
 % the density, and the Surface area
 for count2 = 1:length(expected.delta(count).alpha)
    calIndex = find(cal.depRatio.alpha == round(expected.delta(count).alpha(count2)));
    depCal(count).cal_thickness(count2) = cal.depRatio.ratio(calIndex);

    calIndex = find(cal.hc.alpha == round(expected.delta(count).alpha(count2)));
    depCal(count).water_mass_percentage(count2) = cal.hc.water_mass_percentage(calIndex);
    depCal(count).density_rel(count2) = cal.density.density_rel(calIndex);
    depCal(count).density_g_cm3(count2) = cal.density.density_g_cm3(calIndex);

    calIndex = find(cal.sa.alpha == round(expected.delta(count).alpha(count2)));
    depCal(count).sa_m2_m2_um(count2) = cal.sa.sa_m2_m2_um(calIndex);
    depCal(count).sa_m2_g(count2) = cal.sa.sa_m2_g(calIndex);
 end


%-------------------------------------------------------------
% Correct thickness bands by dep ratio
% Use actual thickness measured by SEM to normalize results
 disp('   Dep Ratio Thickness Calibration');
 depData(count).total_thickness_nm_v1 = sum(depData(count).delta_thickness_nm);
 depData(count).total_thickness_nm_v2 = depData(count).thickness_nm(end);
 depRatio_cal_delta_thickness_nm = depData(count).delta_thickness_nm .* ...
                                   depCal(count).cal_thickness;
 depRatio_cal_total_thickness_nm = sum(depRatio_cal_delta_thickness_nm);
 thickness_cal = meas.thickness_um(count)*1000 / depRatio_cal_total_thickness_nm;
 expected.delta(count).thickness_nm = depRatio_cal_delta_thickness_nm * thickness_cal;

 expected.thickness_um(count) = sum(expected.delta(count).thickness_nm / 1000);

 % Error in thickness is the measured thickness error since they are
 % essentially the same value
 expected.dthickness_um(count) = sqrt(meas.dthickness_um(count)^2*2);

 % Calculate thickness ratio for use in finding weighted average of
 % surface area and density
 % Error due to thickness ratio should probably be twice the measured
 % error
 depCal(count).thickness_ratio = (expected.delta(count).thickness_nm / ...
                                 (expected.thickness_um(count)*1000));


 disp(['     t_meas = ',num2str(meas.thickness_um(count)*1000), ...
       '    t_dep = ',num2str(depData(count).total_thickness_nm_v1), ...
       ' or ',num2str(depData(count).total_thickness_nm_v2), ...
       ' nm, t_depRatioCal = ',num2str(depRatio_cal_total_thickness_nm), ...
       ' nm, t_expected = ',num2str(expected.thickness_um(count)*1000), ...
       ', tcal = ',num2str(thickness_cal)]);
```

```
%-------------------------------------------------------------
% Find mass based on humidity compensation - use thickness ratio
 disp('   Humidity Compensation');

   depCal(count).average_water_mass_percentage = sum(depCal(count).thickness_ratio  .* ...
                                                     depCal(count).water_mass_percentage);


  % Remove water mass from weighed mass
   expected.mass_mg(count) = meas.mass_mg(count) .* ...
                             (1 - depCal(count).average_water_mass_percentage/100);
  % Error in measured mass already took into account humidity error
   expected.dmass_mg(count) = meas.dmass_mg(count);

   mass_diff_mg = expected.mass_mg(count) - meas.mass_mg(count);
   mass_diff_percent = 100 * mass_diff_mg / meas.mass_mg(count);

  disp(['     Average water mass = ',num2str(depCal(count).average_water_mass_percentage),' %']);
  disp(['     Mweighed = ',num2str(meas.mass_mg(count)), ...
        ' +/- ',num2str(meas.dmass_mg(count)), ...
        ' mg, Mexpected = ',num2str(expected.mass_mg(count)), ...
        ' +/- ',num2str(expected.dmass_mg(count)), ...
        ' mg, dM = ',num2str(mass_diff_mg),' mg (',num2str(mass_diff_percent),' %)']);


%-------------------------------------------------------------
% Calculate and display density
 disp('   Calculating density');

  meas.density_g_cm3(count) = (expected.mass_mg(count)/1000) ./ ...
                               (meas.footprint_area_mm2(count) / 10^2) ./ ...
                               (expected.thickness_um(count) / 10^4);

  meas.ddensity_g_cm3(count) = meas.density_g_cm3(count) * sqrt( ...
                              (expected.dmass_mg(count)/expected.mass_mg(count))^2 + ...
                              (meas.dfootprint_area_mm2(count)/meas.footprint_area_mm2(count))^2 + ...
                              (expected.dthickness_um(count)/expected.thickness_um(count))^2);

  expected.density_g_cm3(count) = sum(depCal(count).density_g_cm3 .* ...
                                      depCal(count).thickness_ratio);
  expected.ddensity_g_cm3(count) = expected.density_g_cm3(count) * sqrt( ...
                                   (percent_error_density_g_cm3/100)^2 + ...
                                   (percent_error_thickness_ratio/100)^2);

  density_diff_g_cm3 =   expected.density_g_cm3(count) - meas.density_g_cm3(count);
  density_diff_percent = 100 * density_diff_g_cm3 / meas.density_g_cm3(count);

  disp(['     Density est from humidity corrected mass = ', ...
        num2str(meas.density_g_cm3(count)), ...
        ' +/- ',num2str(meas.ddensity_g_cm3(count)), ...
        ' g/cm3, from weighted average of densities = ', ...
        num2str(expected.density_g_cm3(count)), ...
        ' +/- ',num2str(expected.ddensity_g_cm3(count)), ...
        ' g/cm3.  Diff = ',num2str(density_diff_g_cm3), ...
        ' g/cm3 (',num2str(density_diff_percent),' %)']);


 %-------------------------------------------------------------
 % Estimate SA from BET plot
  disp('   Calculating SA from BET data');
  expected.footprint_area_mm2 = meas.footprint_area_mm2;

  [meas.sa_m2(count) meas.dsa_m2(count), meas.sa_rsq] = calculateSurfaceAreas(betData(count), po_mmHg);

  % Include thickness study error
   meas.dsa_m2(count) = sqrt(meas.dsa_m2(count)^2 + (meas.sa_m2(count) * dsa_thicknessStudy_m2)^2);

  % Calculate specific surface areas
   meas.sa_m2_g(count) = meas.sa_m2(count) ./ (expected.mass_mg(count)/1000);
   meas.sa_m2_m2(count) = meas.sa_m2(count) ./ (meas.footprint_area_mm2(count)/1000^2);
   meas.sa_m2_m2_um(count) = meas.sa_m2_m2(count) ./ (expected.thickness_um(count));

   meas.dsa_m2_g(count) =  meas.sa_m2_g(count) * ...
                           sqrt( (meas.dsa_m2(count)/meas.sa_m2(count))^2 + ...
                                 (meas.dmass_mg(count) / expected.mass_mg(count))^2);

   meas.dsa_m2_m2(count) =  meas.sa_m2_m2(count) * ...
                            sqrt( (meas.dsa_m2(count)/meas.sa_m2(count))^2 + ...
                                  (meas.dfootprint_area_mm2(count) / ...
                                     meas.footprint_area_mm2(count))^2);
```

```
      meas.dsa_m2_m2_um(count) = meas.sa_m2_m2_um(count) * ...
                                 sqrt( (meas.dsa_m2_m2(count)/meas.sa_m2_m2_m2(count))^2 + ...
                                       (meas.dthickness_um(count) / ...
                                          expected.thickness_um(count))^2);


    %-------------------------------------------------------------
    % Use SA calibration data from VP investigation to calculate estimated SA
     disp('   Calculating expected Surface Area based on dep file info');

     expected.sa_m2_m2_um(count) = sum(depCal(count).sa_m2_m2_um .* depCal(count).thickness_ratio);
     expected.dsa_m2_m2_um(count) = expected.sa_m2_m2_um(count) * sqrt( ...
                                     (meas.dsa_m2_m2_um(count)/meas.sa_m2_m2_um(count))^2 + ...
                                      (percent_error_thickness_ratio/100)^2);

     expected.sa_m2_g(count) = sum(depCal(count).sa_m2_g .* depCal(count).thickness_ratio);
     expected.dsa_m2_g(count) = expected.sa_m2_g(count) * sqrt( ...
                                     (meas.dsa_m2_g(count)/meas.sa_m2_g(count))^2 + ...
                                      (percent_error_thickness_ratio/100)^2);

     sa_diff_m2_m2_um = expected.sa_m2_m2_um(count) - meas.sa_m2_m2_um(count);
     sa_diff_m2_m2_um_percent = 100 * sa_diff_m2_m2_um / meas.sa_m2_m2_um(count);
     sa_diff_m2_g = expected.sa_m2_g(count) - meas.sa_m2_g(count);
     sa_diff_m2_g_percent = 100 * sa_diff_m2_g / meas.sa_m2_g(count);

     disp(['     SA_meas = ',num2str(meas.sa_m2_m2_um(count)), ...
           ' +/- ',num2str(meas.dsa_m2_m2_um(count)), ...
           ' m2/m2/um, SA_expected = ',num2str(expected.sa_m2_m2_um(count)), ...
           ' +/- ',num2str(expected.dsa_m2_m2_um(count)), ...
           ' m2/m2/um, dSA = ',num2str(sa_diff_m2_m2_um), ...
           ' m2/m2/um (',num2str(sa_diff_m2_m2_um_percent),' %)']);
     disp(['     SA_meas = ',num2str(meas.sa_m2_g(count)), ...
           ' +/- ',num2str(meas.dsa_m2_g(count)), ...
           ' m2/g, SA_expected = ',num2str(expected.sa_m2_g(count)), ...
           ' +/- ',num2str(expected.dsa_m2_g(count)), ...
           ' m2/g, dSA = ',num2str(sa_diff_m2_g),' m2/g (',num2str(sa_diff_m2_g_percent),' %)']);


end

%-------------------------------------------------------------
% Plots comparing SAs
% Plots comparing densities
disp('Plotting summary plots');


figure('Visible','off');
errorbar(xaxis_nums,meas.density_g_cm3,meas.ddensity_g_cm3 ,'ko','MarkerSize',plotParam.markerSize, ...
         'LineWidth',plotParam.lineWidth, 'MarkerFaceColor', 'k', 'MarkerEdgeColor', 'k');
hold on;
errorbar(xaxis_nums,expected.density_g_cm3,expected.ddensity_g_cm3 , ...
        'ks','MarkerSize',plotParam.markerSize, ...
        'LineWidth',plotParam.lineWidth, 'MarkerFaceColor', 'w', 'MarkerEdgeColor', 'k');
xlabel([xaxis_text],'FontSize',plotParam.fontSize,'LineWidth',plotParam.lineWidth,'FontWeight','bold')
ylabel(['\delta (g cm^{-3})'],'FontSize',plotParam.fontSize, ...
       'LineWidth',plotParam.lineWidth,'FontWeight','bold')
set(gca,'LineWidth',2,'FontSize',plotParam.fontSize,'FontWeight','bold');
legend('Meas','Expected',-1);
axis(axisLimits_density);
print('-dtiff','-r300',[outputDir,'\',testType,'_densityComparison.tiff']);
close;

figure('Visible','off');
errorbar(xaxis_nums,meas.sa_m2_m2_um,meas.dsa_m2_m2_um ,'ko', ...
          'MarkerSize',plotParam.markerSize,'LineWidth',plotParam.lineWidth, ...
          'MarkerFaceColor', 'k', 'MarkerEdgeColor', 'k');
hold on;
errorbar(xaxis_nums,expected.sa_m2_m2_um,expected.dsa_m2_m2_um ,'ks', ...
         'MarkerSize',plotParam.markerSize,'LineWidth',plotParam.lineWidth, ...
         'MarkerFaceColor', 'w', 'MarkerEdgeColor', 'k');
xlabel([xaxis_text],'FontSize',plotParam.fontSize,'LineWidth',plotParam.lineWidth, ...
        'FontWeight','bold')
ylabel(['SA (m^2 m^{-2} \mum^{-1})'],'FontSize',plotParam.fontSize, ...
        'LineWidth',plotParam.lineWidth,'FontWeight','bold')
set(gca,'LineWidth',2,'FontSize',plotParam.fontSize,'FontWeight','bold');
legend('Meas','Expected',-1);
axis(axisLimits_relSA);
print('-dtiff','-r300',[outputDir,'\',testType,'_relSAComparison.tiff']);
close;

figure('Visible','off');
```

```
errorbar(xaxis_nums,meas.sa_m2_g,meas.dsa_m2_g ,'ko','MarkerSize',plotParam.markerSize, ...
        'LineWidth',plotParam.lineWidth, 'MarkerFaceColor', 'k', 'MarkerEdgeColor', 'k');
hold on;
errorbar(xaxis_nums,expected.sa_m2_g,expected.dsa_m2_g ,'ks', ...
        'MarkerSize',plotParam.markerSize,'LineWidth',plotParam.lineWidth, ...
        'MarkerFaceColor', 'w', 'MarkerEdgeColor', 'k');
xlabel([xaxis_text],'FontSize',plotParam.fontSize,'LineWidth',plotParam.lineWidth,'FontWeight','bold')
ylabel(['SA (m^2 g^{-1})'],'FontSize',plotParam.fontSize,'LineWidth',plotParam.lineWidth, ...
        'FontWeight','bold')
set(gca,'LineWidth',2,'FontSize',plotParam.fontSize,'FontWeight','bold');
legend('Meas','Expected',-1);
axis(axisLimits_specificSA);
print('-dtiff','-r300',[outputDir,'\',testType,'_specificSAComparison.tiff']);
close;

disp('Done');



%====================================================================
function [depRatio] = find_depRatios(doPlots, outputDir, plotParam)
% Dep Ratio Calibration

alpha = [0 30 60  70   81   84   86   87];
ratio = [1  1  1 0.8 0.53 0.43 0.40 0.38];
tmp = find(ratio < 1);
firstIndex = tmp(1) - 1;
p = polyfit(alpha(firstIndex:end), ratio(firstIndex:end), 1);
m = p(1);
b = p(2);
fit_alpha = [alpha(firstIndex):1:85];
fit_ratio = fit_alpha*m + b;
depRatio.alpha = [0:1:85];
depRatio.ratio = [ones(1,(alpha(firstIndex))) fit_ratio];

if (doPlots)
    figure('Visible','off');
    plot(alpha,ratio,'ko','MarkerSize',plotParam.markerSize,'LineWidth',plotParam.lineWidth);
    hold on;
    plot(depRatio.alpha,depRatio.ratio,'k-', ...
          'MarkerSize',plotParam.markerSize,'LineWidth',plotParam.lineWidth);
    xlabel(['\alpha (^o)'],'FontSize',plotParam.fontSize,'LineWidth',plotParam.lineWidth,'FontWeight','bold')
    ylabel(['r (-)'],'FontSize',plotParam.fontSize,'LineWidth',plotParam.lineWidth,'FontWeight','bold')
    set(gca,'LineWidth',2,'FontSize',plotParam.fontSize,'FontWeight','bold');
    print('-dtiff','-r300',[outputDir,'\cal_depRatios.tiff']);
    close;
end

%====================================================================
function [sa] = find_SAcal(doPlots, outputDir, plotParam)
% Interpolate Surface Areas

alpha =        [ 0 30 40  45  50  55  60  65  70  75  80  85];
sa_m2_m2_um =  [20 34 43 597 810 852 881 726 588 386 414 262];
dsa_m2_m2_um = [ 3  4  6  77 109 109 114  96  76  56  54  34];
sa_m2_g =      [ 5  9 12 179 243 267 317 267 253 233 236 230];
dsa_m2_g =     [ 1  1  2  23  33  35  41  34  32  31  32  32];
tmp = find(sa_m2_m2_um > 100);
firstIndex = tmp(1);
fit_alpha = [alpha(firstIndex):1:85];
fit_sa_m2_m2_um = spline(alpha(firstIndex:end),sa_m2_m2_um(firstIndex:end),fit_alpha);
fit_dsa_m2_m2_um = spline(alpha(firstIndex:end),dsa_m2_m2_um(firstIndex:end),fit_alpha);
fit_sa_m2_g = spline(alpha(firstIndex:end),sa_m2_g(firstIndex:end),fit_alpha);
fit_dsa_m2_g = spline(alpha(firstIndex:end),dsa_m2_g(firstIndex:end),fit_alpha);

sa.alpha = [0:1:85];
sa.sa_m2_m2_um = [mean(sa_m2_m2_um(1:(firstIndex-1)))*ones(1,(alpha(firstIndex))) fit_sa_m2_m2_um];
sa.dsa_m2_m2_um = [mean(dsa_m2_m2_um(1:(firstIndex-1)))*ones(1,(alpha(firstIndex))) fit_dsa_m2_m2_um];
sa.sa_m2_g = [mean(sa_m2_g(1:(firstIndex-1)))*ones(1,(alpha(firstIndex))) fit_sa_m2_g];
sa.dsa_m2_g = [mean(dsa_m2_g(1:(firstIndex-1)))*ones(1,(alpha(firstIndex))) fit_dsa_m2_g];

if (doPlots)
    figure('Visible','off');
    errorbar(alpha,sa_m2_m2_um,dsa_m2_m2_um ,'ko', ...
             'MarkerSize',plotParam.markerSize,'LineWidth',plotParam.lineWidth);
    hold on;
    plot(sa.alpha,sa.sa_m2_m2_um,'k-', ...
          'MarkerSize',plotParam.markerSize,'LineWidth',plotParam.lineWidth);
    xlabel(['\alpha (^o)'],'FontSize',plotParam.fontSize, ...
              'LineWidth',plotParam.lineWidth,'FontWeight','bold')
    ylabel(['SA (m^2 m^{-2} \mum^{-1})'],'FontSize',plotParam.fontSize, ...
```

```
            'LineWidth',plotParam.lineWidth,'FontWeight','bold')
    set(gca,'LineWidth',2,'FontSize',plotParam.fontSize,'FontWeight','bold');
    print('-dtiff','-r300',[outputDir,'\cal_relSAs.tiff']);
    close;

    figure('Visible','off');
    errorbar(alpha,sa_m2_g,dsa_m2_g ,'ko', ...
             'MarkerSize',plotParam.markerSize,'LineWidth',plotParam.lineWidth);
    hold on;
    plot(sa.alpha,sa.sa_m2_g,'k-', ...
          'MarkerSize',plotParam.markerSize,'LineWidth',plotParam.lineWidth);
    xlabel(['\alpha (^o)'],'FontSize',plotParam.fontSize, ...
             'LineWidth',plotParam.lineWidth,'FontWeight','bold')
    ylabel(['SA (m^2 g^{-1})'],'FontSize',plotParam.fontSize, ...
             'LineWidth',plotParam.lineWidth,'FontWeight','bold')
    set(gca,'LineWidth',2,'FontSize',plotParam.fontSize,'FontWeight','bold');
    print('-dtiff','-r300',[outputDir,'\cal_specificSAs.tiff']);
    close;
end

%====================================================================
function [hc] = find_humidityCal(doPlots, outputDir, plotParam)

% From humidity compensation calculations
% Percentage of water by mass

m = 0.09073;
b = 0.60106;

hc.alpha = [0:1:85];
hc.water_mass_percentage = m * hc.alpha + b;

if (doPlots)
    figure('Visible','off');
    plot(hc.alpha,hc.water_mass_percentage ,'ko', ...
          'MarkerSize',plotParam.markerSize,'LineWidth',plotParam.lineWidth);
    hold on;
    xlabel(['\alpha (^o)'],'FontSize',plotParam.fontSize, ...
            'LineWidth',plotParam.lineWidth,'FontWeight','bold')
    ylabel(['m_{water} (%)'],'FontSize',plotParam.fontSize, ...
            'LineWidth',plotParam.lineWidth,'FontWeight','bold')
    set(gca,'LineWidth',2,'FontSize',plotParam.fontSize,'FontWeight','bold');
    print('-dtiff','-r300',[outputDir,'\cal_humidityComp.tiff']);
    close;
end


%====================================================================
function [density] = find_densityCal(bulk_density_g_cm3, doPlots, outputDir, plotParam)

% Use humidity-compensated densities

alpha =          [    0   30   40   45   50   55   60   65   70   75   80   85];
density_g_cm3 =  [ 3.75 3.69 3.48 3.34 3.34 3.19 2.78 2.72 2.33 1.66 1.76 1.14];
ddensity_g_cm3 = [ 0.27 0.24 0.23 0.21 0.21 0.20 0.18 0.17 0.15 0.17 0.14 0.10];


%  Poxson model:  c = 5.3191 +/- 0.37601, Rsq = 0.87052, scatter in density = 0.19158
c = 5.3191;
density.alpha = [0:1:85];
density.density_rel = c  ./ (c + (density.alpha/180*pi) .* tan(density.alpha/180*pi));
density.density_g_cm3 = bulk_density_g_cm3 * c  ./ ...
                        (c + (density.alpha/180*pi) .* tan(density.alpha/180*pi));



if (doPlots)
    figure('Visible','off');
    errorbar(alpha,density_g_cm3,ddensity_g_cm3 ,'ko', ...
              'MarkerSize',plotParam.markerSize,'LineWidth',plotParam.lineWidth);
    hold on;
    plot(density.alpha,density.density_g_cm3 ,'k-', ...
           'MarkerSize',plotParam.markerSize,'LineWidth',plotParam.lineWidth);
    xlabel(['\alpha (^o)'],'FontSize',plotParam.fontSize, ...
             'LineWidth',plotParam.lineWidth,'FontWeight','bold')
    ylabel(['\delta (g cm^{-3})'],'FontSize',plotParam.fontSize, ...
             'LineWidth',plotParam.lineWidth,'FontWeight','bold')
    set(gca,'LineWidth',2,'FontSize',plotParam.fontSize,'FontWeight','bold');
    print('-dtiff','-r300',[outputDir,'\cal_density.tiff']);
    close;
end
```

```
%====================================================================

function [best_SA_m2, best_dSA_m2, best_rsq] = calculateSurfaceAreas(betData, po_mmHg)


rsq_limit = 0.9980;   % BET Rsq must be above this

% ------------------------------------------------------------------------
% Constants
R_J_per_K_per_mol = 8.314472;      % Constant
Na = 6.022e23;                     % Constant
q = 1.6e-19;

krypton.cross_sectional_area_nm2 = 0.202;   % Krypton - From Matthias Thommes blue book, page 80
krypton.atomic_weight_g_per_mol = 83.798;   % Krypton - CRC Handbook


p_mmHg = betData.relPressure * po_mmHg;
y_full = p_mmHg ./ (betData.volume_cc .* (po_mmHg - p_mmHg));

ranges = [[0.05 0.1];[0.05 0.2];[0.05 0.3]];
numRanges = size(ranges,1);

for rngCounter = 1:numRanges
    tmp = find( betData.relPressure >= ranges(rngCounter,1));
    if (isempty(tmp))
        % Error!
        index1 = 1;
    else
        index1 = tmp(1);
    end
    tmp = find( betData.relPressure >= ranges(rngCounter,2));
    if (isempty(tmp))
        % Error!
        index2 = length(p_mmHg);
    else
        index2 = tmp(1);
    end

    p_po_1(rngCounter) = betData.relPressure(index1);
    p_po_2(rngCounter) = betData.relPressure(index2);
    p1_mmHg(rngCounter) = p_mmHg(index1);
    p2_mmHg(rngCounter) = p_mmHg(index2);

    n = index2 - index1 + 1;
    x = betData.relPressure(index1:index2);
    y = y_full(index1:index2);
    p = polyfit(x,y,1);
    m = p(1);
    b = p(2);


    y2 = m*x + b;

    %figure;plot(x,y,'bo');hold on;
    %plot(x,y2,'r-');


    xsq = x.^2;
    xy = x .* y;


    yerr = y - y2;
    yerrsq = yerr.^2;

    sum_x = sum(x);
    sum_xsq = sum(xsq);
    sum_xy = sum(xy);

    dy = sqrt(sum(yerrsq)/n);

    db = dy * sqrt( sum_xsq / (n*sum_xsq - sum_x.^2));
    dm = dy * sqrt( n / (n*sum_xsq - sum_x.^2));

    c = 1 + m / b;
    Vml = 1/(c*b);
    nml = Vml / 22400;

    SA_m2(rngCounter) = nml * Na * krypton.cross_sectional_area_nm2 * 1e-18;
```

```
    dSA_m2(rngCounter) = SA_m2(rngCounter) * sqrt(db^2 + dm^2) / (b+m);

    rsq(rngCounter) = 1 - sum((y - y2).^2) / sum((y - mean(y)).^2);

    disp(['     c = ',num2str(c),', Vml = ',num2str(Vml), ...
          ', SA = ',num2str(SA_m2(rngCounter)),' +/- ', ...
          num2str(dSA_m2(rngCounter)),' m2, Rsq = ',num2str(rsq(rngCounter))]);
end

tmp = find(rsq > rsq_limit);
bestIndex = tmp(end);

best_SA_m2 = SA_m2(bestIndex);
best_dSA_m2 = dSA_m2(bestIndex);
best_rsq = rsq(bestIndex);
```

A.4 Code to calculate Pore Size Distribution of films using BJH method

```
% Program to open isotherms (ad or de) and estimate Pore Size
% Distributions from each.  87K or 77K data may be analyzed.
% The surface tension of Krypton at these temperatures
% is extrapolated.
% The density of super-cooled Krypton is taken from M. Thommes's
% extrapolation (presented in his text book).
%
% SiO2 and TiO2 masses compensated for water adsorption
% (see humidityCompensation.m in Analysis\HumidityCompensation)
%
% Smooth calculated PSDs as is done for Quantachrome SW
% calculated PSDs (using M.Thommes Empirical method)
% such that the two methods can be compared.
%


% ------------------------------------------------------------------------
% Data parameters
close all;
clear all;
clc;

material = 'TiO2';    % ITO or SiO2 or TiO2
temp_K = 87;          % 77 or 87
alpha = 75;           % 40 to 85
sorption = 'ad';      %'ad' or 'de'


fontSize = 20;
lineWidth = 2;
markerSize = 10;
colours = 'brmgckbrmgckbrmgckbrmgck';
shapes = 'svodx>+hsvodx>+hsvodx>';
lineStyles = {'-','--',':','-.','-','--',':','-.','-','--',':','-.','-','--',':','-.'};


% ------------------------------------------------------------------------
% Constants

R_J_per_K_per_mol = 8.314472;      % Constant
Na = 6.022e23;                     % Constant

cross_sectional_area_nm2 = 0.202;   % From Matthias Thommes blue book, page 80
atomic_weight_g_per_mol = 83.798;   % CRC Handbook


% ------------------------------------------------------------------------
% File specific data

if (isequal(material,'TiO2'))

    if (temp_K == 77)
```

```
        if (isequal(sorption,'ad'))
            r_k_nm_1 = 3;
            r_k_nm_2 = 0.4;
        else
            r_k_nm_1 = 2.5;
            r_k_nm_2 = 0.5;
        end

        switch alpha
            case 45
                raw_directory = 'C:\My_Documents\Research\Research_Areas\Porosity\Analysis\', ...
                  'Text_data_from_77K_BET_and_87K_isotherm_tests\TextData_TiO2_VP_a45_May13_08';
                raw_file = '2008_06_05_TiO2_a45_May1308dep_77K_adData_2.txt';
                mass_g = 0.00828; %0.00869;
            case 50
                raw_directory = 'C:\My_Documents\Research\Research_Areas\Porosity\Analysis\', ...
                  'Text_data_from_77K_BET_and_87K_isotherm_tests\TextData_TiO2_VP_a50_Mar3_08';
                raw_file = '2008_06_23_TiO2_a50_Mar308_77K_adData.txt';
                mass_g = 0.00776; %0.00818;

            % ...
            % Note: raw_directory, raw_file, mass_g definitions removed to save space

        end

    elseif (temp_K == 87)
        if (isequal(sorption,'ad'))
            r_k_nm_spline = 4.7:-bin_size_nm:0.4; %4.5:-bin_size_nm:0.5;
        else
            r_k_nm_spline = 4:-bin_size_nm:0.5;
        end

        switch alpha
            case 30
                raw_directory = 'C:\My_Documents\Research\Research_Areas\Porosity\Analysis\', ...
                  'Text_data_from_77K_BET_and_87K_isotherm_tests\TextData_TiO2_VP_a30_April09_08';
                raw_file = '2008_04_28to29_TiO2_a30_Apr908_dep_87K_micropore_isotherm.txt';
                mass_g = 0.00538; %0.00556;
            case 40
                raw_directory = 'C:\My_Documents\Research\Research_Areas\Porosity\Analysis\', ...
                  'Text_data_from_77K_BET_and_87K_isotherm_tests\TextData_TiO2_VP_a40_April09_08';
                raw_file = '2008_04_29to30_TiO2_a40_Apr908dep_micropore_isotherm.txt';
                mass_g = 0.00463; %0.00484;

            % ...
            % Note: raw_directory, raw_file, mass_g definitions removed to save space

        end

    end
elseif (isequal(material,'SiO2'))

    % Note: raw_directory, raw_file, mass_g definitions removed to save space


elseif (isequal(material,'ITO'))

    % Note: raw_directory, raw_file, mass_g definitions removed to save space


end



% ------------------------------------------------------------------------
% Calculations

surface_tension_N_per_m = - 2.0802e-4*temp_K + 4.0494e-2;
                                % Extrapolated from NIST Chemistry Webdoc

%liquid_density_g_per_cm3 = - 7.6710e-3*temp_K + 3.3350;
                                % Extrapolated from NIST Chemistry Webdoc
liquid_density_g_per_cm3 =  2.6;    % From Matthias Thommes blue book, page 79

liquid_po_torr = 16.2;  %  1.8449e-3*(temp_K)^2 - 3.8220e-1*temp_K + 2.0073e1;



cross_sectional_area_cm2_per_mol = cross_sectional_area_nm2 * (10^(-7))^2 * Na;
molar_volume_cm3_per_mol = atomic_weight_g_per_mol/ liquid_density_g_per_cm3;
molecular_thickness_m = (molar_volume_cm3_per_mol / cross_sectional_area_cm2_per_mol)/100;
```

```
doSpline = 1;
test_name = [num2str(temp_K),'K_',material,'_',num2str(alpha),'_',sorption,'sorption'];
plot_directory = ['Data_WaterAdComp\Plots_',material,'_', ...
                  num2str(temp_K),'K\',test_name]; mkdir(plot_directory);

disp([test_name]);


% Read in data file
disp('Reading in file');
[p_ratio, volume_gas_cm3, lastPo_torr] = readFile([raw_directory,'\',raw_file]);



% Plot volume gas adsorbed vs p_ratio
figure('Visible','off');
plot(p_ratio, volume_gas_cm3,'ko','MarkerSize',markerSize,'LineWidth',lineWidth)
hold on;
set(gca,'LineWidth',lineWidth,'FontSize',fontSize,'FontWeight','bold');
xlabel(['p/p_o'],'FontSize',fontSize,'LineWidth',lineWidth,'FontWeight','bold')
ylabel(['V_{gas} [cm^3]'],'FontSize',fontSize,'LineWidth',lineWidth,'FontWeight','bold')
print('-dtiff','-r300',[plot_directory,'\','sorption_isotherm_', ...
                 material,'_',num2str(alpha),'.tiff']);
close;



if (isequal(sorption,'na'))
else
    tmp = find(p_ratio == max(p_ratio));
    if (isequal(sorption,'ad'))
        p_ratio = p_ratio(1:tmp(1));
        volume_gas_cm3 = volume_gas_cm3(1:tmp(1));
        p_ratio = p_ratio(end:-1:1);
        volume_gas_cm3 = volume_gas_cm3(end:-1:1);
    else
        p_ratio = p_ratio(tmp(end)+1:end);
        volume_gas_cm3 = volume_gas_cm3(tmp(end)+1:end);
    end
end


num_pores = length(p_ratio);


disp('Calculating specific volume of gas adsorbed');
% Calculate the specific volume of gas adsorbed
volume_gas_cm3_per_g = volume_gas_cm3 / mass_g;

figure('Visible','off');
plot(p_ratio, volume_gas_cm3_per_g,'ko','MarkerSize',markerSize,'LineWidth',lineWidth)
hold on;
set(gca,'LineWidth',lineWidth,'FontSize',fontSize,'FontWeight','bold');
xlabel(['p/p_o'],'FontSize',fontSize,'LineWidth',lineWidth,'FontWeight','bold')
ylabel(['V_{gas} [cm^3 g^{-1}]'],'FontSize',fontSize,'LineWidth',lineWidth,'FontWeight','bold')
print('-dtiff','-r300',[plot_directory,'\',sorption, ...
          'sorption_isotherm_',material,'_',num2str(alpha),'.tiff']);
close;

disp('Calculating kelvin radii');
% Kelvin equation describes the radius of the pore
% Except that, we assume that the pore has some film in it
% (of thickness t), so the radius of the pore is actually the
% radius of the hole in the fluid which coats the pore
r_k_m = - 2 * surface_tension_N_per_m * (molar_volume_cm3_per_mol/100^3) ./ ...
         ( log(p_ratio) * R_J_per_K_per_mol * temp_K );
r_k_cm = r_k_m * 100;



figure('Visible','off');
plot(p_ratio, r_k_cm/100 * 10^9,'ko','MarkerSize',markerSize,'LineWidth',lineWidth)
hold on;
set(gca,'LineWidth',lineWidth,'FontSize',fontSize,'FontWeight','bold');
xlabel(['p/p_o'],'FontSize',fontSize,'LineWidth',lineWidth,'FontWeight','bold')
ylabel(['r_k [nm]'],'FontSize',fontSize,'LineWidth',lineWidth,'FontWeight','bold')
print('-dtiff','-r300',[plot_directory,'\',sorption, ...
                'sorption_kelvin_radius_',material,'_',num2str(alpha),'.tiff']);
close;
```

```
% The thickness of the fluid in the pore
% Described by a common curve for a non-porous surface
% (the Halsey equation)
t_m = molecular_thickness_m * (5 ./ log(1./p_ratio)) .^(1/3);
t_cm = t_m * 100;

% The actual pore radius
r_p_cm = r_k_cm + t_cm;


% Limit r_k to reasonable range
    tmp = find(r_k_cm*10^7 < r_k_nm_2);
    tmp2 = find(r_k_cm*10^7 > r_k_nm_1);
    index1 = tmp(1);
    index2 = tmp2(end);
    r_k_cm = r_k_cm(index1:index2);
    volume_gas_cm3_per_g = volume_gas_cm3_per_g(index1:index2);
    r_p_cm = r_p_cm(index1:index2);
    t_cm = t_cm(index1:index2);
    num_pores = length(r_k_cm);

    figure('Visible','off');
    plot(r_k_cm * (10^7), volume_gas_cm3_per_g,'ko','MarkerSize',markerSize,'LineWidth',lineWidth)
    hold on;
    set(gca,'LineWidth',lineWidth,'FontSize',fontSize,'FontWeight','bold');
    xlabel(['r_k [nm]'],'FontSize',fontSize,'LineWidth',lineWidth,'FontWeight','bold')
    ylabel(['V_{gas} [cm^3 g^{-1}]'],'FontSize',fontSize,'LineWidth',lineWidth,'FontWeight','bold')
    xlim([r_k_nm_spline(end),r_k_nm_spline(1)]);
    print('-dtiff','-r300',[plot_directory,'\',sorption,'sorption_rkisotherm_',material,'_',num2str(alpha),'.tiff']);
    close;



disp('Calculating step to step changes in radii, thicknesses, etc...');
% Calculate the change in thickness at each pressure step
delta_t_cm = -(t_cm(2:end) - t_cm(1:end-1));

% Work with r_k and r_p at the half-way points between steps
r_k_mean_cm = (r_k_cm(2:end) + r_k_cm(1:end-1))/2;
r_p_mean_cm = (r_p_cm(2:end) + r_p_cm(1:end-1))/2;

% Calculate the change in volume of gas adsorbed at each step
delta_volume_gas_cm3_per_g = -(volume_gas_cm3_per_g(2:end) - volume_gas_cm3_per_g(1:end-1));

% Estimate the equivalent volume of liquid in the pores
% At standard temperature and pressure
% (is this valid?)
% divide the volume of gas by 224000 * molecular thickness (in A)
delta_volume_liq_cm3_per_g = delta_volume_gas_cm3_per_g / ...
                                22400 * molar_volume_cm3_per_mol;


% Calculate the volume of liquid adsorbed at each step
%volume_liq_cm3 = volume_gas_cm3 / 22440 * molar_volume_cm3_per_mol;
%volume_liq_g = volume_liq_cm3 * liquid_density_g_per_cm3;

volume_liq_cm3_per_g  = volume_gas_cm3_per_g / 22400 * molar_volume_cm3_per_mol;

disp(['Total pore volume = ',num2str(volume_liq_cm3_per_g(1)), ...
         ' cm^3/g at ',num2str(p_ratio(1))]);




% ------------------------------------------------------------------------
% BJH Method implementation

disp('Calculate pore volumes');
delta_volume_larger_p_cm3_per_g = zeros(1,(num_pores-1));
delta_volume_p_cm3_per_g = zeros((num_pores-1),(num_pores-1));
volume_p_cm3_per_g = zeros(1,(num_pores-1));
length_p_cm_per_g = zeros(1,(num_pores-1));
r_k_new_cm = zeros((num_pores-1),(num_pores-1));
inside_area_p_cm2_per_g = zeros((num_pores-1),(num_pores-1));
% For each pressure step
for j = 1: (num_pores-1)
    % For each pore
    delta_volume_larger_p_cm3_per_g(j) = 0;
    for k = 1:j
```

```
        if (j == k)
            if (j == 1)
                delta_volume_p_cm3_per_g(j,k) = delta_volume_liq_cm3_per_g(j);
            else
                delta_volume_p_cm3_per_g(j,k) = delta_volume_liq_cm3_per_g(j) - ...
                                                delta_volume_larger_p_cm3_per_g(j);
            end

            % Calculate the volume of the pore based on the new volume
            % of the pore after it has lost some delta thickness out
            % of its inside
            % Assume that the pore started out with a thick layer
            % of fluid (of thickness t_m(k=j)) and a radius of the inner
            % core of air in the fluid of r_k_mean_m(k=j)
            % At this step an amount of fluid of thickness delta_t_m
            % is removed.
            % We don't know the (specific) length of the pore,
            % but can know the ratio of the volume of the pore to the
            % volume of the hollow core.
            if (j == 1)
                volume_p_cm3_per_g(k) = delta_volume_p_cm3_per_g(j,k) * ...
                                        (r_p_mean_cm(k) / (r_k_mean_cm(j)))^2;
            else
                volume_p_cm3_per_g(k) = delta_volume_p_cm3_per_g(j,k) * ...
                                        (r_p_mean_cm(k) / (r_k_mean_cm(j)+ delta_t_cm(j)))^2;
            end

            % Calculate the (specific) length of the pore
            % The volume of the pore divided by the area across the
            % pore
            length_p_cm_per_g(k) = volume_p_cm3_per_g(k) / (pi * (r_p_mean_cm(k))^2);


            % Calculate the new r_k of the core of the pore liquid
            r_k_new_cm(j,k) = r_k_mean_cm(k) + delta_t_cm(j)/2;

            % Calculate the area of the inner core of the fluid within the
            % pore
            inside_area_p_cm2_per_g(j,k) = 4 * pi * r_k_new_cm(j,k) * length_p_cm_per_g(k);

        else
            % Change in volume from removal of last thickness
            delta_volume_p_cm3_per_g(j,k) = inside_area_p_cm2_per_g(j-1,k) * delta_t_cm(j);
            delta_volume_larger_p_cm3_per_g(j) = delta_volume_larger_p_cm3_per_g(j) + ...
                                                 delta_volume_p_cm3_per_g(j,k);

            r_k_new_cm(j,k) = r_k_new_cm(j-1,k) + delta_t_cm(j);

            inside_area_p_cm2_per_g(j,k) = 4 * pi * r_k_new_cm(j,k) * length_p_cm_per_g(k);

        end
    end
end


%------------------------------------------------------------------------
% Figure plotting

disp('Plotting and saving final data');
r_p_nm = r_p_cm * 10^7;
d_p_nm = r_p_nm * 2;
d_p_mean_nm = r_p_mean_cm * 2 * 10^7;
delta_r_p_nm = r_p_nm(1:end-1) - r_p_nm(2:end);
delta_d_p_nm = d_p_nm(1:end-1) - d_p_nm(2:end);
delta_log_d_p = log(d_p_nm(1:end-1) ./ d_p_nm(2:end));

DVd = volume_p_cm3_per_g ./ delta_d_p_nm;
DVlogd = 2 * volume_p_cm3_per_g ./ delta_log_d_p;

figure('Visible','off');
plot(d_p_mean_nm, DVd,'ko','MarkerSize',markerSize,'LineWidth',lineWidth)
hold on;
    set(gca,'LineWidth',lineWidth,'FontSize',fontSize,'FontWeight','bold');
title([material,' \alpha=',num2str(alpha)]);
xlabel(['d_p [nm]'],'FontSize',fontSize,'LineWidth',lineWidth,'FontWeight','bold')
ylabel(['V / \Delta d_p [cm^3 g^{-1} A^{-1}]'],'FontSize',fontSize, ...
            'LineWidth',lineWidth,'FontWeight','bold')
print('-dtiff','-r300',[plot_directory,'\',sorption,'sorption_PSD_', ...
            material,'_',num2str(alpha),'_d.tiff']);
close
```

```
figure('Visible','off');
plot(d_p_mean_nm, DVlogd,'ko','MarkerSize',markerSize,'LineWidth',lineWidth)
hold on;
    set(gca,'LineWidth',lineWidth,'FontSize',fontSize,'FontWeight','bold');
title([material,' \alpha=',num2str(alpha)]);
xlabel(['d_p [nm]'],'FontSize',fontSize,'LineWidth',lineWidth,'FontWeight','bold')
ylabel(['V / \Delta log(d_p) [cm^3 g^{-1}]'],'FontSize',fontSize, ...
         'LineWidth',lineWidth,'FontWeight','bold')
print('-dtiff','-r300',[plot_directory,'\',sorption,'sorption_PSD_', ...
         material,'_',num2str(alpha),'_logd.tiff']);
close


cum_volume_p_cm3_per_g = cumsum(volume_p_cm3_per_g(end:-1:1));

figure('Visible','off');
plot(d_p_mean_nm(end:-1:1), cum_volume_p_cm3_per_g,'ko','MarkerSize', ...
          markerSize,'LineWidth',lineWidth)
hold on;
    set(gca,'LineWidth',lineWidth,'FontSize',fontSize,'FontWeight','bold');
title([material,' \alpha=',num2str(alpha)]);
xlabel(['d_p [nm]'],'FontSize',fontSize,'LineWidth',lineWidth,'FontWeight','bold')
ylabel(['V_{cum} [cm^3 g^{-1}]'],'FontSize',fontSize, ...
           'LineWidth',lineWidth,'FontWeight','bold')
print('-dtiff','-r300',[plot_directory,'\',sorption,'sorption_CumVol_', ...
           material,'_',num2str(alpha),'.tiff']);
close



fid = fopen([plot_directory,'\',sorption,'sorption_PSD.txt'],'w');
fprintf(fid,'Mean Diameter (nm), DV/d, DV/log(d) \n');
for count = 1:num_pores-1
    fprintf(fid,'%6.3f, %8.6f, %6.4f \n',d_p_mean_nm(count), DVd(count), DVlogd(count));
end
fclose(fid);

close all;

disp('Done');
```

A.5 Code to plot Pore Volumes

```
% Code to take pore volumes, calculated by Quantachrome AS1 software
% (and same as BJH PSD code) and plot
% Also, compare porosity calculated this way to that calculated
% from densities of samples


close all;
clear all;
clc;


warning off all

fontSize = 20;
lineWidth = 2;
markerSize = 10;
colours = 'brmgckbrmgckbrmgckbrmgck';
shapes = 'svodx>+hsvodx>+hsvodx>';
lineStyles = {'-','--',':','-.','-','--',':','-.','-','--',':','-.','-','--',':','-.'};

homework = 1;


switch homework
    case 1
        beginDir = ['C:\My_Documents\'];
    case 2
        beginDir = ['H:\private\My Documents\'];
end
```

```
outputDir = [beginDir,'Research\Research_Areas\Porosity\Analysis\PoreVolume\Plots_Volume'];
mkdir(outputDir);

alphaInd = '_a';
fileSuffix = '.txt';



%-----------------------------------------------------------------------
% Determining total pore volume
disp('Determing total pore volume');

% Mesoporosity - pore volume at ~10 nm
% Determined from Quatachrome AS1 software
% (But also calculated in BJH PSD code - volume of liquid in largest pores)
tio2_alphas = [45    50    55    60    65    70    75    80   85];
tio2_meso_porevolume_cc_per_g = [0.0519    0.0694    0.0918    0.1259    ...
                                 0.1472    0.179    0.1600   0.1008  0.1160];
sio2_alphas = [45    50    55    60    65   70    75    80    85];
sio2_meso_porevolume_cc_per_g = [0.1247    0.1663    0.2008    0.2389    ...
                                 0.3055   0.2607   0.3011    0.2224    0.209];

% Density measured with balance
% There has been much discussion about which bulk density to use!
% Anatase = 3.8, Rutile = 4.2 g cm-3 (CRC)
% Amorphous = somewhere in between?
% At 0 degrees our bulk density is ~ 4
tio2_bulk_density_g_cm3 = 4.0;

% Corrected for water adsorption (see Humidity Compensation code):
tio2_mass_mg = [8.28 7.76 8.11  5.34 7.22 5.95 4.55 13.90 11.97];

% Original measured masses - not compensated for water adsorption
tio2_mass_mg_weighed = [8.69 8.18 8.59 5.68 7.72 6.39 4.91 15.09 13.05];
tio2_meso_porevolume_cc = tio2_meso_porevolume_cc_per_g .* (tio2_mass_mg_weighed / 1000);

% Correct for mass originally inputted in Quantachrome software
tio2_meso_porevolume_cc_per_g = tio2_meso_porevolume_cc_per_g .* tio2_mass_mg_weighed ./ tio2_mass_mg;

tio2_footprint_area_mm2 = [2445 2313 2664 2037 2837 2601 2559 3988 3963];
tio2_thickness_um = [1.03 1.019 0.97 0.958 0.95 0.999 1.09 2.183 2.64];



% The highest SiO2 bulk density reported is 2.65 (CRC, Quartz)
% But, amorphous SiO2 is 2.2 g cm-3
% For our deps, at 0 degrees the density is 2.5 g cm-3
% Use the amorphous value
sio2_bulk_density_g_cm3 =  2.5;

% Correct for water adsorption
sio2_mass_mg = [4.88  2.06 2.08 3.12 3.05 3.88 2.09 1.63 1.16];

% Original measured masses - not compensated for water adsorption
sio2_mass_mg_weighed = [5.66 2.43 2.49 3.79 3.77 4.88 2.67 2.12 1.53];

sio2_meso_porevolume_cc = sio2_meso_porevolume_cc_per_g .* (sio2_mass_mg_weighed / 1000);

% Correct for mass originally inputted in Quantachrome software
sio2_meso_porevolume_cc_per_g = sio2_meso_porevolume_cc_per_g .* sio2_mass_mg_weighed ./ sio2_mass_mg;

sio2_footprint_area_mm2 = [3230 1356 1496 2549 2565 4200 2566 2580 3127];
sio2_thickness_um = [0.90 0.91 0.93 0.86 0.93 0.96 0.95 0.88 0.99];


%-----------------------------------------------------------------------
curr_outputDir = [outputDir,'\summary_dt40_ds25']; mkdir(curr_outputDir);


%-----------------------------------------------------------------------
tait_density_rel = 2 * cos(tio2_alphas*pi/180) ./ (1 + cos(tio2_alphas*pi/180));
tait_density_percent = 100 * tait_density_rel;
tait_porosity_percent = 100 - tait_density_percent;


%-----------------------------------------------------------------------
% See SummaryFilmProperties.txt in BET directory
sio2_density_error_percent = 12;
tio2_density_error_percent = 8;
sio2_mass_error_percent = 10;
```

```
tio2_mass_error_percent = 5;

% Error in weighed density is sensitive to bulk density
% Range of bulk density is ~ 2.2 to 2.7 g/cm3
% Impact on weighed_volume_cc_per_g is 20%
% because of 1/actual_density - 1/bulk_density relationship
sio2_bulk_density_error_percent = 20;
% Range of bulk density is ~ 3.8 to 4.2 g/cm3
tio2_bulk_density_error_percent = 20;

% Using estimate that there is a 4% error in supercooled krypton density
% (2.6 +/- 0.1 g/cm3)
porevolume_error_percent = 4;

% Repeatibility of Quantachrome Autosorb
% Estimate to be 12% from Surface Area thickness scatter study
autosorb_repeatibility_error_percent = 12;



%-------------------------------------------------------------------------
% TiO2

% Calculate errors in density and porosity for these measurements
tio2_weighed_density_g_cm3 = (tio2_mass_mg/1000) ./ ...
                             (tio2_footprint_area_mm2 / 10^2) ./ ...
                             (tio2_thickness_um / 10^4);
tio2_weighed_density_error_g_cm3 = tio2_weighed_density_g_cm3 * tio2_density_error_percent / 100;

tio2_weighed_density_rel = tio2_weighed_density_g_cm3 ./ tio2_bulk_density_g_cm3;
tio2_weighed_density_error_rel = tio2_weighed_density_rel .* ...
                                 sqrt( (tio2_density_error_percent/100).^2 + ...
                                       (tio2_bulk_density_error_percent/100).^2);
tio2_weighed_density_percent = 100 * tio2_weighed_density_rel;
tio2_weighed_density_error_percent = 100 * tio2_weighed_density_error_rel;
tio2_weighed_porosity = 1 - tio2_weighed_density_rel;
tio2_weighed_porosity_percent = 100 - tio2_weighed_density_percent;
tio2_weighed_porosity_error_percent = tio2_weighed_porosity_percent .* ...
                                      ( tio2_weighed_density_error_rel ./ ...
                                        tio2_weighed_density_rel);


% porosity_volume = sample_volume * (1 - actual_density / bulk_density) / mass
%                 = 1/actual_density - 1/bulk_density
tio2_weighed_volume_cc_per_g = (tio2_weighed_porosity_percent/100) ./ ...
                               (tio2_mass_mg/1000) .* ...
                               (tio2_footprint_area_mm2 / 10^2) .* ...
                               (tio2_thickness_um / 10^4);
tio2_weighed_volume_err_cc_per_g = tio2_weighed_volume_cc_per_g .* ...
                                   sqrt( (tio2_density_error_percent/100).^2 + ...
                                         (tio2_bulk_density_error_percent/100).^2);




%-------------------------------------------------------------------------
% SiO2:

sio2_weighed_density_g_cm3 = (sio2_mass_mg/1000) ./ ...
                              (sio2_footprint_area_mm2 / 10^2) ./ (sio2_thickness_um / 10^4);
sio2_weighed_density_error_g_cm3 = tio2_weighed_density_g_cm3 * tio2_density_error_percent / 100;
sio2_weighed_density_rel = sio2_weighed_density_g_cm3 ./ sio2_bulk_density_g_cm3;
sio2_weighed_density_error_rel = sio2_weighed_density_rel .* ...
                                 sqrt( (sio2_density_error_percent/100).^2 + ...
                                       (sio2_bulk_density_error_percent/100).^2);
sio2_weighed_density_percent = 100 * sio2_weighed_density_rel;
sio2_weighed_density_error_percent = 100 * sio2_weighed_density_error_rel;
sio2_weighed_porosity = 1 - tio2_weighed_density_rel;
sio2_weighed_porosity_percent = 100 - sio2_weighed_density_percent;
sio2_weighed_porosity_error_percent = sio2_weighed_porosity_percent .* ...
                                      ( sio2_weighed_density_error_rel ./ ...
                                          sio2_weighed_density_rel);

sio2_weighed_volume_cc_per_g = (sio2_weighed_porosity_percent/100) ./ ...
                               (sio2_mass_mg/1000) .* ...
                               (sio2_footprint_area_mm2 / 10^2) .* ...
                               (sio2_thickness_um / 10^4);
sio2_weighed_volume_err_cc_per_g = sio2_weighed_volume_cc_per_g .* ...
                                   sqrt( (sio2_density_error_percent/100).^2 + ...
                                         (sio2_bulk_density_error_percent/100).^2);
```

```
%-------------------------------------------------------------------------
% Here we are seeing what the pore volume is relative to a fully packed volume
% It is not quite the same comparison as the above
% where we compared the deposited density (actual) to
% a bulk density that is in itself not completely filling the volume
%

% Here the bulk density has an impact of about 10% in pore volume
% percentage sensitivity calculations
tio2_bulk_density_error_percent = 10;
sio2_bulk_density_error_percent = 10;


tio2_samplevolume_cc = ((tio2_footprint_area_mm2 / 10^2) .* (tio2_thickness_um / 10^4));

% A thin (non-porous) film would still have a density
% which is due to the atoms not being next each other (as closely
% packed as possible in a crystal, and not quite so packed in an
% amorphous material)
v_bulk_cm3 = (tio2_mass_mg / 1000) ./ tio2_bulk_density_g_cm3;
v_pores_cm3 = tio2_meso_porevolume_cc_per_g .* (tio2_mass_mg / 1000);
tio2_meso_porevolume_cc_per_cc = tio2_meso_porevolume_cc ./ ...
                                 (v_bulk_cm3 + tio2_meso_porevolume_cc);
tio2_meso_porevolume_percent = 100 * tio2_meso_porevolume_cc_per_cc;
tio2_meso_porevolume_error_cc_per_cc = tio2_meso_porevolume_cc_per_cc .* ...
                                    sqrt( (autosorb_repeatibility_error_percent/100).^2 + ...
                                           (porevolume_error_percent/100).^2 + ...
                                           (tio2_mass_error_percent/100).^2 + ...
                                           (tio2_bulk_density_error_percent/100).^2 );
tio2_meso_porevolume_error_percent = tio2_meso_porevolume_error_cc_per_cc  *100;
tio2_meso_porevolume_rem_macro = tio2_weighed_porosity_percent - tio2_meso_porevolume_percent;

tio2_meso_porevolume_error_cc_per_g = tio2_meso_porevolume_cc_per_g .* sqrt( ...
                                    (autosorb_repeatibility_error_percent/100).^2 + ...
                                    (porevolume_error_percent/100).^2 + ...
                                           (tio2_mass_error_percent/100).^2 );

tio2_airvolume_cc = tio2_samplevolume_cc .* ...
                     (1 - tio2_weighed_density_g_cm3 ./ tio2_bulk_density_g_cm3);
tio2_intercolumnvolume_cc = tio2_airvolume_cc - tio2_meso_porevolume_cc;
tio2_intercolumnvolume_rel = tio2_intercolumnvolume_cc ./ tio2_samplevolume_cc;
tio2_intercolumnvolume_percent = 100 * tio2_intercolumnvolume_rel;

disp([num2str(tio2_intercolumnvolume_percent)]);



sio2_samplevolume_cc = ((sio2_footprint_area_mm2 / 10^2) .* ...
                         (sio2_thickness_um / 10^4));
v_bulk_cm3 = (sio2_mass_mg / 1000) ./ sio2_bulk_density_g_cm3;
sio2_meso_porevolume_cc_per_cc = sio2_meso_porevolume_cc ./ ...
                                 (v_bulk_cm3 + sio2_meso_porevolume_cc);
sio2_meso_porevolume_percent = 100 * sio2_meso_porevolume_cc_per_cc;
sio2_meso_porevolume_error_cc_per_cc = sio2_meso_porevolume_cc_per_cc .* ...
                                    sqrt( (autosorb_repeatibility_error_percent/100).^2 + ...
                                           (porevolume_error_percent/100).^2 + ...
                                           (sio2_mass_error_percent/100).^2 + ...
                                           (sio2_bulk_density_error_percent/100).^2 );
sio2_meso_porevolume_error_percent = sio2_meso_porevolume_error_cc_per_cc * 100;
sio2_meso_porevolume_rem_macro = sio2_weighed_porosity_percent - sio2_meso_porevolume_percent;

sio2_meso_porevolume_error_cc_per_g = sio2_meso_porevolume_cc_per_g .* sqrt( ...
                                    (autosorb_repeatibility_error_percent/100).^2 + ...
                                    (porevolume_error_percent/100).^2 + ...
                                           (sio2_mass_error_percent/100).^2 );


sio2_airvolume_cc = sio2_samplevolume_cc .* sio2_weighed_porosity;
sio2_intercolumnvolume_cc = sio2_airvolume_cc - sio2_meso_porevolume_cc;
sio2_intercolumnvolume_rel = sio2_intercolumnvolume_cc ./ sio2_samplevolume_cc;
sio2_intercolumnvolume_percent = 100 * sio2_intercolumnvolume_rel;

disp([num2str(sio2_intercolumnvolume_percent)]);


%-------------------------------------------------------------------------

figure('Visible','off');
```

```
tmp = find(sio2_alphas <= 65);
errorbar(sio2_alphas(tmp), sio2_meso_porevolume_cc_per_g(tmp), ...
         sio2_meso_porevolume_error_cc_per_g(tmp), 'ks', ...
         'LineWidth', lineWidth, 'MarkerSize', markerSize, ...
         'MarkerFaceColor', 'w', 'MarkerEdgeColor', 'k');
hold on;
tmp = find(tio2_alphas <= 65);
errorbar(tio2_alphas(tmp), tio2_meso_porevolume_cc_per_g(tmp), ...
         tio2_meso_porevolume_error_cc_per_g(tmp), 'ko', ...
         'LineWidth', lineWidth, 'MarkerSize', markerSize, ...
         'MarkerFaceColor', 'k', 'MarkerEdgeColor', 'k');
xlabel('\alpha (^o)','FontSize',fontSize,'FontWeight','bold');
ylabel('V_{pore} (cm^3 g^{-1})','FontSize',fontSize,'FontWeight','bold');
set(gca,'LineWidth',lineWidth,'FontSize',fontSize,'FontWeight','bold');
legend('SiO_2','TiO_2',-1);
tmp = find(sio2_alphas <= 65);
% For SiO2 bulk density of 2.5 g/cm3, m = 0.0099g/cm3/deg, b = -0.31 g/cm3
p = polyfit(sio2_alphas(tmp), sio2_meso_porevolume_cc_per_g(tmp),1);
fit_y = sio2_alphas(tmp)*p(1) + p(2);
plot(sio2_alphas(tmp), fit_y,'k-', 'LineWidth', lineWidth);
tmp = find(tio2_alphas <= 65);
% For TiO2 bulk density of 3.0 g/cm3, m = 0.0051g/cm3/deg, b = -0.18 g/cm3
p = polyfit(tio2_alphas(tmp), tio2_meso_porevolume_cc_per_g(tmp),1);
fit_y = tio2_alphas(tmp)*p(1) + p(2);
plot(tio2_alphas(tmp), fit_y,'k-', 'LineWidth', lineWidth);
axis([40 70 0 0.45]);
set(gca,'YTick',[0:0.05:0.45]);
print('-dtiff',[curr_outputDir,'\pore_volume_per_mass_tio2_sio2_10nm_psd']);
close




figure('Visible','off');
plot(tio2_alphas, tio2_meso_porevolume_percent, 'ko-', 'LineWidth', lineWidth, ...
     'MarkerSize', markerSize, 'MarkerFaceColor', 'k', 'MarkerEdgeColor', 'k');
hold on;
plot(sio2_alphas, sio2_meso_porevolume_percent, 'ks-', 'LineWidth', lineWidth, ...
     'MarkerSize', markerSize, 'MarkerFaceColor', 'w', 'MarkerEdgeColor', 'k');
xlabel('\alpha (^o)','FontSize',fontSize,'FontWeight','bold');
ylabel('V_{pore} (%)','FontSize',fontSize,'FontWeight','bold');
set(gca,'LineWidth',lineWidth,'FontSize',fontSize,'FontWeight','bold');
legend('TiO_2','SiO_2',-1);
axis([40 90 0 60]);
set(gca,'YTick',[0:5:60]);
print('-dtiff',[curr_outputDir,'\pore_volume_percent_tio2_sio2_10nm']);
close



figure('Visible','off');
errorbar(tio2_alphas, tio2_weighed_porosity_percent, tio2_weighed_porosity_error_percent, ...
         'kv-', 'LineWidth', lineWidth, 'MarkerSize', markerSize, ...
         'MarkerFaceColor', 'w', 'MarkerEdgeColor', 'k');
hold on;
plot(tio2_alphas, tait_porosity_percent, 'k.--', 'LineWidth', lineWidth, ...
         'MarkerSize', markerSize, 'MarkerFaceColor', 'r', 'MarkerEdgeColor', 'k');
errorbar(tio2_alphas, tio2_meso_porevolume_percent, tio2_meso_porevolume_error_percent, ...
         'ko-', 'LineWidth', lineWidth, 'MarkerSize', markerSize, ...
         'MarkerFaceColor', 'b', 'MarkerEdgeColor', 'k');
xlabel('\alpha (^o)','FontSize',fontSize,'FontWeight','bold');
ylabel('V_{pore} (%)','FontSize',fontSize,'FontWeight','bold');
set(gca,'LineWidth',lineWidth,'FontSize',fontSize,'FontWeight','bold');
legend('Meas','Tait','Meso',-1);
axis([40 90 0 100]);
set(gca,'YTick',[0:10:100]);
set(gca,'XTick',[40:10:90]);
print('-dtiff',[curr_outputDir,'\pore_volume_macro_tio2']);
close

figure('Visible','off');
errorbar(sio2_alphas, sio2_weighed_porosity_percent, sio2_weighed_porosity_error_percent, ...
         'kv-', 'LineWidth', lineWidth, 'MarkerSize', markerSize, ...
         'MarkerFaceColor', 'w', 'MarkerEdgeColor', 'k');
hold on;
plot(sio2_alphas, tait_porosity_percent, 'k.--', 'LineWidth', lineWidth, ...
      'MarkerSize', markerSize, 'MarkerFaceColor', 'r', 'MarkerEdgeColor', 'k');
errorbar(sio2_alphas, sio2_meso_porevolume_percent, sio2_meso_porevolume_error_percent, ...
          'ko-', 'LineWidth', lineWidth, 'MarkerSize', markerSize, ...
          'MarkerFaceColor', 'b', 'MarkerEdgeColor', 'k');
```

```
xlabel('\alpha (^o)','FontSize',fontSize,'FontWeight','bold');
ylabel('V_{pore} (%)','FontSize',fontSize,'FontWeight','bold');
set(gca,'LineWidth',lineWidth,'FontSize',fontSize,'FontWeight','bold');
legend('Meas','Tait','Meso',-1);
axis([40 90 0 100]);
set(gca,'YTick',[0:10:100]);
set(gca,'XTick',[40:10:90]);
print('-dtiff',[curr_outputDir,'\pore_volume_macro_sio2']);
close


figure('Visible','off');
tmp = find(sio2_alphas <= 65);
errorbar(sio2_alphas(tmp), sio2_meso_porevolume_percent(tmp), ...
         sio2_meso_porevolume_error_percent(tmp), 'ks', 'LineWidth', lineWidth, ...
         'MarkerSize', markerSize, 'MarkerFaceColor', 'w', 'MarkerEdgeColor', 'k');
hold on;
errorbar(sio2_alphas(tmp), sio2_weighed_porosity_percent(tmp), ...
         sio2_weighed_porosity_error_percent(tmp),'ko', 'LineWidth', lineWidth, ...
         'MarkerSize', markerSize, 'MarkerFaceColor', 'k', 'MarkerEdgeColor', 'k');
xlabel('\alpha (^o)','FontSize',fontSize,'FontWeight','bold');
ylabel('V_{pore} (%)','FontSize',fontSize,'FontWeight','bold');
set(gca,'LineWidth',lineWidth,'FontSize',fontSize,'FontWeight','bold');
legend('SiO_2 PSD  ','SiO_2 Weigh',-1);
tmp = find(sio2_alphas <= 65);
% For a bulk density of 2.5 g/cm3, m = 0.98%/deg, b =  -17.5%
p = polyfit(sio2_alphas(tmp), sio2_meso_porevolume_percent(tmp),1);
fit_y = sio2_alphas(tmp)*p(1) + p(2);
%plot(sio2_alphas(tmp), fit_y,'k-', 'LineWidth', lineWidth);
p = polyfit(sio2_alphas(tmp), sio2_weighed_porosity_percent(tmp),1);
fit_y = sio2_alphas(tmp)*p(1) + p(2);
%plot(sio2_alphas(tmp), fit_y,'k--', 'LineWidth', lineWidth);
axis([40 70 0 60]);
set(gca,'YTick',[0:10:60]);
print('-dtiff',[curr_outputDir,'\pore_volume_percent_sio2_10nm_weighed_psd']);
close

figure('Visible','off');
tmp = find(tio2_alphas <= 65);
errorbar(tio2_alphas(tmp), tio2_meso_porevolume_percent(tmp), ...
         tio2_meso_porevolume_error_percent(tmp), 'ks', 'LineWidth', lineWidth, ...
         'MarkerSize', markerSize, 'MarkerFaceColor', 'w', 'MarkerEdgeColor', 'k');
hold on;
errorbar(tio2_alphas(tmp), tio2_weighed_porosity_percent(tmp), ...
         tio2_weighed_porosity_error_percent(tmp),'ko', 'LineWidth', lineWidth, ...
         'MarkerSize', markerSize, 'MarkerFaceColor', 'k', 'MarkerEdgeColor', 'k');
xlabel('\alpha (^o)','FontSize',fontSize,'FontWeight','bold');
ylabel('V_{pore} (%)','FontSize',fontSize,'FontWeight','bold');
set(gca,'LineWidth',lineWidth,'FontSize',fontSize,'FontWeight','bold');
legend('TiO_2 PSD  ','TiO_2 Weigh',-1);
tmp = find(tio2_alphas <= 65);
% For a bulk density of 4.0 g/cm3, m = 1.04%/deg, b = -29.5 %
p = polyfit(tio2_alphas(tmp), tio2_meso_porevolume_percent(tmp),1);
fit_y = tio2_alphas(tmp)*p(1) + p(2);
%plot(tio2_alphas(tmp), fit_y,'k-', 'LineWidth', lineWidth);
p = polyfit(tio2_alphas(tmp), tio2_weighed_porosity_percent(tmp),1);
fit_y = tio2_alphas(tmp)*p(1) + p(2);
%plot(tio2_alphas(tmp), fit_y,'k--', 'LineWidth', lineWidth);
axis([40 70 0 60]);
set(gca,'YTick',[0:10:60]);
print('-dtiff',[curr_outputDir,'\pore_volume_percent_tio2_10nm_weighed_psd']);
close


close all;

disp('Done');
```

A.6 Code to plot PSDs calculated using Quantachrome Empirical Method

```
% Code to read in PSDs, as calculated by Quantachrome AS1 software
% (with Empirical method by M. Thommes built-in to SW) and plot
% Filtering of the PSDs is performed
% The peak of the fit PSDs is calculated
```

```
clear all
close all
clc

warning off all

fontSize = 20;
lineWidth = 2;
markerSize = 10;
colours = 'brmgckbrmgckbrmgckbrmgck';
shapes = 'svodx>+hsvodx>+hsvodx>';
lineStyles = {'-','--',':','-.','-','--',':','-.','-','--',':','-.','-','--',':','-.'};


homework = 1;
averaging = 0;
material = 'TiO2'; %'SiO2';

doPlotOriginalData = 1;
doPlotLogData = 1;
doPlotNormalizedData = 1;

doFilterPSDdata = 1;
doPlotLogFiltData = 1;

doSummaryPeakLocPlots = 1;


switch homework
    case 1
        beginDir = ['C:\My_Documents\'];
    case 2
        beginDir = ['H:\private\My Documents\'];
end

inputDir = [beginDir,'Research\Research_Areas\Porosity\Analysis\PSD_EmpiricalMethod\PSDTables'];

switch averaging
    case 1
        outputDir = [beginDir,'Research\Research_Areas\Porosity\Analysis\PSD_EmpiricalMethod\Plots_5ptMA'];
        filePrefix = 'tab_Kr87_PSD_5ptMA';
    case 0
        outputDir = [beginDir,'Research\Research_Areas\Porosity\Analysis\PSD_EmpiricalMethod\Plots_noAvg'];
        filePrefix = 'tab_Kr87_PSD';
end

alphaInd = '_a';
fileSuffix = '.txt';


switch material
    case 'TiO2'
        switch averaging
            case 1
                startFile = 1;
                sample_plots = [[0 0 2 3 4];[0 1 2 3 4];[0 5 6 7 8];[1 3 5 7 8];[0 0 1 4 8]];
            case 0
                startFile = 3;
                sample_plots = [[0 0 4 5 6];[0 3 4 5 6];[0 7 8 9 11];[3 5 7 9 11];[0 0 3 6 11]];
        end

        alpha = [ 45  50  55  60  65  70  75  80  85];
        mass_mg = [8.28 7.76 8.11  5.34 7.22 5.95 4.55 13.90 11.97];
        % Original measured masses - not compensated for water adsorption
        mass_mg_weighed = [8.69 8.18 8.59 5.68 7.72 6.39 4.91 15.09 13.05];
        footprint_area_mm2 = [2445 2313 2664 2037 2837 2601 2559 3988 3963];
        thickness_um = [1.03 1.019 0.97 0.958 0.95 0.999 1.09 2.183 2.64];

        plotStart = [12, 15,20, 20, 23];

        zoomsOrig_x = [[0.6 2];[1 10]];
        zoomsOrig_y = [[0 0.02];[0 3e-3]];
        zoomsOrig_x_tick = [0.2 1];
        zoomsLog_x = [[1 10]];
        zoomsLog_y = [[0 0.2]];
        zoomsLog_x_tick = [1];
        zoomsNorm_x = [[0.6 2];[1 10]];
        zoomsNorm_y = [[0 0.02];[0 2.5e-3]];
        zoomsNorm_x_tick = [0.2 1];
```

```
        zoomsOrigSummary_x = [[0.6 2];[1 10]];
        zoomsOrigSummary_y = [[0 0.02];[0 3e-3]];
        zoomsOrigSummary_x_tick = [0.2 1];
        zoomsLogSummary_x = [[1 10]];
        zoomsLogSummary_y = [[0 0.2]];
        zoomsLogSummary_x_tick = [1];
        zoomsNormSummary_x = [[0.6 2];[1 10]];
        zoomsNormSummary_y = [[0 0.02];[0 2.5e-3]];
        zoomsNormSummary_x_tick = [0.2 1];


        cutBackAlphas = [50 55 60 65 70 75 85];
        boundary = [2 10]; %[1 10]; %[2 10];


    case 'SiO2'
        switch averaging
            case 1
                startFile = 2;
                sample_plots = [[0 0 3 4 5]; [0 2 3 4 5];[0 6 8 9 10];[2 4 6 8 10];[0 0 2 5 10]];
            case 0
                startFile = 3;
                sample_plots = [[0 0 4 5 6]; [0 3 4 5 6];[0 7 9 10 11];[3 5 7 9 11];[0 0 3 6 11]];
        end

        alpha = [ 45  50  55  60  65  70  75  80  85];
        mass_mg =         [4.88  2.06 2.08 3.12 3.05 3.88 2.09 1.63 1.16];
        % Original measured masses - not compensated for water adsorption
        mass_mg_weighed = [5.66 2.43 2.49 3.79 3.77 4.88 2.67 2.12 1.53];
        footprint_area_mm2 = [3230 1356 1496 2549 2565 4200 2566 2580 3127];
        thickness_um = [0.90 0.91 0.93 0.86 0.93 0.96 0.95 0.88 0.99];

        plotStart = [12, 15, 20, 20, 23];

        zoomsOrig_x = [[0.6 2];[1 10]];
        zoomsOrig_y = [[0 0.02];[0 0.02]];
        zoomsOrig_x_tick = [0.2 1];
        zoomsLog_x = [[1 10]];
        zoomsLog_y = [[0 0.3]];
        zoomsLog_x_tick = [1];
        zoomsNorm_x = [[0.6 2];[1 10]];
        zoomsNorm_y = [[0 0.01];[0 3e-3]];
        zoomsNorm_x_tick = [0.2 1];

        zoomsOrigSummary_x = [[0.6 2];[1 10]];
        zoomsOrigSummary_y = [[0 0.02];[0 0.02]];
        zoomsOrigSummary_x_tick = [0.2 1];
        zoomsLogSummary_x = [[1 10]];
        zoomsLogSummary_y = [[0 0.3]];
        zoomsLogSummary_x_tick = [1];
        zoomsNormSummary_x = [[0.6 2];[1 10]];
        zoomsNormSummary_y = [[0 0.02];[0 3e-3]];
        zoomsNormSummary_x_tick = [0.2 1];


        cutBackAlphas = [50 55 60 65 75 80 85];
        boundary = [2 10];

end


%-----------------------------------------------------------------------
% Start program
disp(['Material ',material]);

fileNames = dir([inputDir,'\',filePrefix,'_',material,alphaInd,'*',fileSuffix]);
outputDir = [outputDir, '_',material,'_2to10nm']; mkdir(outputDir);

% Get data
disp('Reading in data files');
for count = 1:length(fileNames)
    curr_fileName = fileNames(count).name;
    tmp = findstr(curr_fileName,alphaInd);
    tmp2 = findstr(curr_fileName,fileSuffix);
    curr_alpha = str2num(curr_fileName(tmp+2:tmp2-1));
    disp([' ... alpha: ',num2str(curr_alpha)]);

    psd(count).alpha = curr_alpha;
    alphas(count) = curr_alpha;
```

```
    fid = fopen([inputDir,'\',curr_fileName]);
    mydata = textscan(fid,'%s %s %s %s','headerlines',1);
    fclose(fid);

    diameter_st = char(mydata{1});
    porevolume_st = char(mydata{2});
    dv_d_st = char(mydata{3});
    dv_logd_st = char(mydata{4});
    for count2 = 2:size(diameter_st,1)
        psd(count).diameter(count2-1) = str2num(diameter_st(count2,:));
        psd(count).porevolume_cc_g(count2-1) = str2num(porevolume_st(count2,:));
        psd(count).dv_d_cc_g(count2-1) = str2num(dv_d_st(count2,:));
        psd(count).dv_logd_cc_g(count2-1) = str2num(dv_logd_st(count2,:));
    end

    sampleCount = find(alpha == curr_alpha);

    if ~isempty(sampleCount)
        psd(count).porevolume_cc_g = psd(count).porevolume_cc_g * ...
                                     mass_mg_weighed(sampleCount) / mass_mg(sampleCount);
        psd(count).dv_d_cc_g = psd(count).dv_d_cc_g * ...
                                     mass_mg_weighed(sampleCount) / mass_mg(sampleCount);
        psd(count).dv_logd_cc_g =  psd(count).dv_logd_cc_g * ...
                                     mass_mg_weighed(sampleCount) / mass_mg(sampleCount);
    end
end

%-----------------------------------------------------------------------
% Remove some data points (microporous region)
disp('Culling data points')
for count = 1:length(fileNames)
    if (find(cutBackAlphas == alphas(count)))
        tmp = find(psd(count).diameter/10 >= boundary(1) & psd(count).diameter/10 <= boundary(2));
        psd(count).diameter = psd(count).diameter(tmp);
        psd(count).porevolume_cc_g = psd(count).porevolume_cc_g(tmp);
        psd(count).dv_d_cc_g = psd(count).dv_d_cc_g(tmp);
        psd(count).dv_logd_cc_g = psd(count).dv_logd_cc_g(tmp);
    end
end


%-----------------------------------------------------------------------
% Plot original figures
disp('Filtering PSD data');
if doFilterPSDdata

    filterCoeff = 3;
    b = ones(1,(filterCoeff+1));
    a = (filterCoeff+1)*[1 zeros(1,filterCoeff)];

    for count = 1:length(fileNames)
        psd(count).dv_logd_filt_cc_g = filtfilt(b,a,psd(count).dv_logd_cc_g);
    end
end


%-----------------------------------------------------------------------
% Plot original figures
disp('Plotting original figures');
if doPlotOriginalData
    curr_outputDir = [outputDir,'\original']; mkdir(curr_outputDir);
    for count = 1:length(fileNames)
        plotIndividualFigure(psd(count).alpha, psd(count).diameter/10, ...
                 psd(count).dv_d_cc_g, [''], ['D (nm)'], ['dV_p / (d)'], ...
                 zoomsOrig_x, zoomsOrig_y, zoomsOrig_x_tick, curr_outputDir, ...
                 fontSize, lineWidth, markerSize);
    end
    plotSampleSummaryPlots(sample_plots, psd, 'orig', [''], ['d (nm)'], ...
            ['dV_p / (d)  [cm^3 g^{-1} nm^{-1}]'], shapes, zoomsOrigSummary_x, ...
            zoomsOrigSummary_y, zoomsOrigSummary_x_tick, curr_outputDir, fontSize, ...
            lineWidth, markerSize);
end

if doPlotLogData
    curr_outputDir = [outputDir,'\original_logd']; mkdir(curr_outputDir);
    for count = 1:length(fileNames)
        plotIndividualFigure(psd(count).alpha, psd(count).diameter/10,  ...
              psd(count).dv_logd_cc_g, [''], ['D (nm)'], ['dV_p / (logd)'], ...
              zoomsLog_x, zoomsLog_y, zoomsLog_x_tick, curr_outputDir, fontSize, lineWidth, markerSize);
    end
```

```
    plotSampleSummaryPlots(sample_plots, psd, 'log', [''], ['d (nm)'], ...
            ['dV_p / (logd)  [cm^3 g^{-1} log(nm^{-1})]'], shapes, ...
            zoomsLogSummary_x, zoomsLogSummary_y, zoomsLogSummary_x_tick, ...
            curr_outputDir, fontSize, lineWidth, markerSize);
end

if (doPlotLogFiltData)
    curr_outputDir = [outputDir,'\original_logd_filt']; mkdir(curr_outputDir);
    for count = 1:length(fileNames)
        plotIndividualFigure(psd(count).alpha, psd(count).diameter/10, ...
               psd(count).dv_logd_filt_cc_g, [''], ['D (nm)'], ['dV_p // (logD)'], ...
               zoomsLog_x, zoomsLog_y, zoomsLog_x_tick, curr_outputDir, fontSize, lineWidth, markerSize);
    end
    plotSampleSummaryPlots(sample_plots, psd, 'logfilt', [''], ['d (nm)'], ...
           ['dV_p / (logd)  [cm^3 g^{-1} log(nm^{-1})]'], shapes, ...
           zoomsLogSummary_x, zoomsLogSummary_y, zoomsLogSummary_x_tick, ...
           curr_outputDir, fontSize, lineWidth, markerSize);
end



%--------------------------------------------------------------------
% Fit curves to peaks
disp('Fitting curves to peaks');

curr_outputDir = [outputDir,'\fitting']; mkdir(curr_outputDir);

counter = 0;
for alpha = fit_alphas
    counter = counter + 1;

    [range_nm, add_points_d, add_points_dV] = get_fitParameters_subBack(material, alpha);
    fileNum = find(alphas == alpha);

    x = psd(fileNum).diameter/10;
    y = psd(fileNum).dv_logd_filt_cc_g;

    tmp = find(x >= range_nm(1) & x <= range_nm(2));
    x = x(tmp);
    y = y(tmp);

    x = [add_points_d x];
    y = [add_points_dV y];

    % Log normal
    p0 = [2, 1, 2];
    options = optimset('MaxFunEvals',3000,'MaxIter',2000,'TolFun',1e-7);
    f = @(p,xdata)p(3).*exp(-((log(xdata) - p(1))./(sqrt(2)*p(2))).^2) ./ (xdata .* p(2) * sqrt(2*pi));
    [p, resnorm, residual, exitflag] = lsqcurvefit(f, p0, x, y, [], [], options);
    fit_x = [1:0.1:10];
    fit_y = f(p, fit_x);

    [max_y, max_i] = max(fit_y);
    peakLoc(counter) = fit_x(max_i);
    peakVal(counter) = max_y;


    tmp = find(fit_y(1:max_i) >= peakVal(counter)/2);
    if (isempty(tmp))
        leftSpread(counter) = 0;
    else
        leftSpread(counter) = peakLoc(counter) - fit_x(tmp(1));
    end
    tmp = find(fit_y(max_i:end) >= peakVal(counter)/2);
    if (isempty(tmp))
        rightSpread(counter) = 0;
    else
        rightSpread(counter) = fit_x(max_i + tmp(end) -1) - peakLoc(counter);
    end

    figure('Visible','off');
    plot(psd(fileNum).diameter/10,psd(fileNum).dv_logd_filt_cc_g, 'ko', ...
             'LineWidth', lineWidth, 'MarkerSize', markerSize, ...
             'MarkerFaceColor', 'k', 'MarkerEdgeColor', 'k');
    hold on;
    tmp = find(fit_x*10 >= psd(fileNum).diameter(1) & fit_x*10 <= psd(fileNum).diameter(end));
    plot(fit_x(tmp),fit_y(tmp),'k-', 'LineWidth', lineWidth, ...
              'MarkerSize', markerSize, 'MarkerFaceColor', 'k', 'MarkerEdgeColor', 'k');
    title(['\alpha = ',num2str(alpha),', \mu = ',num2str(peakLoc(counter)), ...
           ' - ',num2str(leftSpread(counter)),' + ', ...
           num2str(rightSpread(counter))],'FontSize',fontSize,'FontWeight','bold');
```

```
    xlabel(['d [nm]'],'FontSize',fontSize,'FontWeight','bold');
    ylabel(['dV_p / (logd) [cm^3 g^{-1} log(nm^{-1})]'],'FontSize',fontSize,'FontWeight','bold');
    set(gca,'LineWidth',lineWidth,'FontSize',fontSize,'FontWeight','bold');
    print('-dtiff',[curr_outputDir,'\Fit_a_',num2str(alpha)]);

end


if (doSummaryPeakLocPlots)
    curr_outputDir = [outputDir,'\summary']; mkdir(curr_outputDir);
    plotSummaryPeaks(fit_alphas, peakLoc, peakVal, leftSpread, ...
                     rightSpread, curr_outputDir, fontSize, lineWidth, markerSize)
end

close all

disp('Done');
```

A.7 Code to estimate inter-column porosity by SEM image analysis

```
close all
clear all
clc

warning off all

fontSize = 20;
lineWidth = 2;
markerSize = 10;
colours = 'brmgckbrmgckbrmgckbrmgck';
shapes = 'svodx>+hsvodx>+hsvodx>';
lineStyles = {'-','--',':','-.','-','--',':','-.','-','--',':','-.','-','--',':','-.'};
white = 255;
black = 0;

homework = 1;

switch homework
    case 1
        beginDir = ['C:\My_Documents\'];
    case 2
        beginDir = ['H:\private\My Documents\'];
end


testCase = 'TiO2_SEMs';
testType = 'Black'; %'White';  % Black = find void distances,  White = find column diameters


switch testCase
    case 'TiO2_SEMs'

        inputDir = [beginDir,'Research\Research_Areas\Porosity\SurfaceAreaSimulation\SEM_Images'];
        outputDir = [beginDir,'Research\Research_Areas\Porosity\SurfaceAreaSimulation\', ...
                      testType,'Distances\TiO2_SEMs_OtsuThreshold'];mkdir(outputDir);


        material = 'TiO2';
        alpha = [65, 70, 75, 80, 85];

        % DepDates = {'March20_2008', 'March5_2008', 'March4_2008', 'Jan28_2008', 'March19_2008'}
        thickness = [0.95 0.999 1.09 2.18 1.43];

        image_filename = 'image_analyze.tiff'; %'image_thresh_best.tiff';
        data_filename = 'scale.txt';

        filtImgCoeff = [5 5 5 5 5];
        threshLevel = [0.32 0.45 0.40 0.48 0.34];   %Otsu's method

        targetDensity = [ 100 100 100   73.3004  61.9162   46.4520];

        stepSize = 1;
        switch testType
            case 'Black'
                binsMax = [50 200 200 300 600];
                plot_binsMax = [20 100 150 250 500];
```

```
                ymax = [2500, 2500, 1200, 800, 250]*stepSize;
            case 'White'
                binsMax = [300 500 500 500 500];
                plot_binsMax = [200 400 400 400 400];
                ymax = [200, 400, 400, 400, 200];
        end

        smallBins = 0;
        peakLocGrt = [ 5 20 20 20 50];
        peakLocLst = [ 50 150 150 150 200];
        filtHistNum = [15 31 15 15 21]; %[3 31 15 15 15];
        startFilt = [3 10 10 10 20]; %[2 10 10 10 20];


        % STEPS:
        % 1)  RUN with testType = 'White'
        %           Collect blackMeans
        % 2)  RUN with testType = 'Black'
        %           Collect whitePeaks
        % 3)  Fill in
        % 4)  Rerun to plot final density summary
        %
        % Results:
        whitePeaks = [12 89 85 83 128]; %[13 76 85 83 127];
        blackMeans = [3.4, 19.2, 34.1, 53.1, 114.3]; %[3.2, 20.7, 34.1, 53.1, 118.2];
        blackMeans_gr10nm = [12.5 24.6 39.1 59.8 123.4]; %[12.2, 25.8, 39.1, 59.8, 127.5];


end

outputFileName_orig = ['ORIGIMAGE'];
outputFileName_filt = ['FILTIMAGE'];
outputFileName_thresh = ['THRESHIMAGE'];
suffix = '.tiff';


for count = 1:length(alpha)

    curr_inputDir = [inputDir,'\',material,'_a',num2str(alpha(count))];
    directories{count} = {curr_inputDir};
end

numFiles = length(alpha);

disp('Loading image files');
samples = loadFiles(numFiles, directories, image_filename, data_filename); %, N);


% Finding peak
f = @(p,xdata)p(3).*exp(-(xdata - p(1)).^2 / (2*p(2)));


% Edge detection image
disp('Detecting edges for images and finding void distances');
for currSample = 1:length(samples)
    disp([' Alpha = ',num2str(alpha(currSample))]);

    origImage = squeeze(samples(currSample).image(:,:,1));


    % Filter image
    disp([' ... Filtering images']);

    b = ones(1,(filtImgCoeff(currSample)+1));
    a = (filtImgCoeff(currSample)+1)*[1 zeros(1,filtImgCoeff(currSample))];
    filtImage = filtfilt(b,a,double(origImage));
    filtImage = uint8((filtfilt(b,a,filtImage'))');


    % Threshold image
    disp([' ... Thresholding images']);
    level = graythresh(filtImage);
    threshImage = im2bw(filtImage, threshLevel(currSample)) * white;

    % Calculating image density
    whitePixels = sum(threshImage == white);
    blackPixels = sum(threshImage == black);
    image_density(currSample) = whitePixels/(whitePixels+blackPixels);
    disp([' ... image density = ',num2str(image_density(currSample))]);

    % Save the final image
```

```
disp([' ... Saving images']);
imwrite(uint8(origImage),[outputDir, '\', outputFileName_orig, ...
            '_', num2str(alpha(currSample)),suffix],'tiff');
imwrite(uint8(filtImage),[outputDir, '\', outputFileName_filt, ...
            '_', num2str(alpha(currSample)), suffix],'tiff');
imwrite(uint8(255*threshImage),[outputDir, '\', outputFileName_thresh, ...
            '_', num2str(alpha(currSample)), suffix],'tiff');
close all;


currImage = threshImage;
currImage(find(currImage == 0)) = 99;


% Try rotating sample image
rotatedImage = imrotate(currImage, 45);
rotatedImage(find(rotatedImage == 0)) = 125;
rotatedImage(find(rotatedImage == 99)) = 0;
currImage(find(currImage == 99)) = 0;


% Find edges of sample image
[edgeImage, edgeImage_x, edgeImage_y] = findEdgesImage(currImage == white);
[edgeImage, edgeImage_x45, edgeImage_y45] = findEdgesImage(rotatedImage == white);


% Find black distances
[blackDistances_x] = findBlackDistances(edgeImage_x, testType);
[blackDistances_y] = findBlackDistances(edgeImage_y', testType);
[blackDistances_x45] = findBlackDistances(edgeImage_x45, testType);
[blackDistances_y45] = findBlackDistances(edgeImage_y45', testType);

blackDistances_x_nm = blackDistances_x(find(blackDistances_x > smallBins)) ...
                    * samples(currSample).dx_nm;
blackDistances_y_nm = blackDistances_y(find(blackDistances_y > smallBins)) ...
                    * samples(currSample).dy_nm;
blackDistances_x45_nm = blackDistances_x45(find(blackDistances_x45 > smallBins)) ...
                    * samples(currSample).dx_nm;
blackDistances_y45_nm = blackDistances_y45(find(blackDistances_y45 > smallBins)) ...
                    * samples(currSample).dy_nm;


meanDistance_x_nm = mean(blackDistances_x_nm);
meanDistance_y_nm = mean(blackDistances_y_nm);
meanDistance_x45_nm = mean(blackDistances_x45_nm);
meanDistance_y45_nm = mean(blackDistances_y45_nm);
disp([' ... Mean ',testType,' Distances : x= ',num2str(meanDistance_x_nm), ...
       ', y= ',num2str(meanDistance_y_nm),', x45= ',num2str(meanDistance_x45_nm), ...
       ', y45= ',num2str(meanDistance_y45_nm),' nm']);


sumDistance_nm(currSample) = sum(blackDistances_x_nm) + sum(blackDistances_y_nm) + ...
                             sum(blackDistances_x45_nm) + sum(blackDistances_y45_nm);
disp([' ... Sum ',testType,' Distance = ',num2str(sumDistance_nm(currSample)),' nm']);


meanDistance_nm(currSample) = mean([meanDistance_x_nm, meanDistance_y_nm, ...
                                    meanDistance_x45_nm, meanDistance_y45_nm]);
disp([' ... Mean ',testType,' Distance = ',num2str(meanDistance_nm(currSample)),' nm']);

bins = [0:stepSize:binsMax(currSample)];
startingBin = 1;
useBins = bins(startingBin:end);
blackDistances_x_peaks = hist(blackDistances_x_nm,bins);
blackDistances_x_peaks = blackDistances_x_peaks(startingBin:end);
blackDistances_y_peaks = hist(blackDistances_y_nm,bins);
blackDistances_y_peaks = blackDistances_y_peaks(startingBin:end);
blackDistances_x45_peaks = hist(blackDistances_x45_nm,bins);
blackDistances_x45_peaks = blackDistances_x45_peaks(startingBin:end);
blackDistances_y45_peaks = hist(blackDistances_y45_nm,bins);
blackDistances_y45_peaks = blackDistances_y45_peaks(startingBin:end);

blackDistances_peaks = (blackDistances_x_peaks + blackDistances_y_peaks + ...
                        blackDistances_x45_peaks + blackDistances_y45_peaks)/4;

disp([' ... Sum counts ',num2str(sum(blackDistances_peaks))]);


switch testType
    case 'White'
        tmp = find(blackDistances_peaks > 0);
```

```
            fitPeaks = spline(useBins(tmp), blackDistances_peaks(tmp), useBins);

            fitPeaks(1:(startFilt(currSample)-1)) = fitPeaks(startFilt(currSample));

            if (filtHistNum(currSample) == 1)
                filtPeaks = fitPeaks;
            else
                b = ones(1,(filtHistNum(currSample)+1));
                a = (filtHistNum(currSample)+1)*[1 zeros(1,filtHistNum(currSample))];

                filtPeaks = filtfilt(b,a,double(fitPeaks));
            end


            [maxVal maxLoc] = max(filtPeaks(peakLocGrt(currSample):peakLocLst(currSample)));
            columnDiameter_nm(currSample) = useBins(maxLoc + peakLocGrt(currSample)-1);

            disp([' ... Diameter = ',num2str(columnDiameter_nm(currSample))]);

    end




    figure;
    tmp = find(blackDistances_peaks > 0);
    plot(useBins(tmp), blackDistances_peaks(tmp),'ko','LineWidth', lineWidth, ...
         'MarkerSize', markerSize, 'MarkerFaceColor', 'w', 'MarkerEdgeColor', 'k' );
    hold on;
    switch testType
        case 'White'
            plot(useBins, filtPeaks,'k-','LineWidth', lineWidth);
    end
    title([material, ' \alpha = ',num2str(alpha(currSample))], ...
                    'FontSize',fontSize,'FontWeight','bold');
    xlabel(['d_{void} (nm)'],'FontSize',fontSize,'FontWeight','bold');
    ylabel(['Count (-)'],'FontSize',fontSize,'FontWeight','bold');
    set(gca,'LineWidth',lineWidth,'FontSize',fontSize,'FontWeight','bold');
    axis([0,plot_binsMax(currSample),0,ymax(currSample)]);
    print('-dtiff',[outputDir,'\Histogram_a',num2str(alpha(currSample))]);
    close;

end

disp(['image_density = [',num2str(image_density*100),'];']);


switch testType
    case 'Black'

        disp(['macro_blackMeans = [',num2str(meanDistance_nm),'];']);
        disp(['blackSumDistance = [',num2str(sumDistance_nm),'];']);

        figure;
        plot(alpha , meanDistance_nm,'ko', 'LineWidth', lineWidth, 'MarkerSize', ...
             markerSize, 'MarkerFaceColor', 'k', 'MarkerEdgeColor', 'k');
        hold on;
        ylabel(['d_{void} (nm)'],'FontSize',fontSize,'FontWeight','bold');
        xlabel(['\alpha (^o)'],'FontSize',fontSize,'FontWeight','bold');
        set(gca,'LineWidth',lineWidth,'FontSize',fontSize,'FontWeight','bold');
        print('-dtiff',[outputDir,'\Mean_Void_Distance_vs_alpha']);
        axis([60 90 0 120]);
        close;
    case 'White'
        disp(['macro_whitePeaks = [',num2str(columnDiameter_nm),'];']);
        disp(['whiteSumDistance = [',num2str(sumDistance_nm),'];']);

        figure;
        plot(alpha , columnDiameter_nm,'ko', 'LineWidth', lineWidth, 'MarkerSize', ...
             markerSize, 'MarkerFaceColor', 'k', 'MarkerEdgeColor', 'k');
        hold on;
        ylabel(['d_{peak} (nm)'],'FontSize',fontSize,'FontWeight','bold');
        xlabel(['\alpha (^o)'],'FontSize',fontSize,'FontWeight','bold');
        set(gca,'LineWidth',lineWidth,'FontSize',fontSize,'FontWeight','bold');
        axis([60 90 0 140]);
        print('-dtiff',[outputDir,'\Peak_Diameter_vs_alpha']);
        close;

end

figure;
```

```
plot(alpha, image_density,'gd', 'MarkerSize', markerSize, ...
          'MarkerFaceColor', 'g', 'MarkerEdgeColor', 'k');
hold on;
% Tait's rule:  density_rel = 2*cos(alpha) / (1 + cos(alpha))
density_rel = 2 * cos(alpha*pi/180) ./ (1 + cos(alpha*pi/180));
plot(alpha, density_rel,'bo', 'MarkerSize', markerSize, ...
          'MarkerFaceColor', 'b', 'MarkerEdgeColor', 'k');
ylabel(['\delta (-)'],'FontSize',fontSize,'FontWeight','bold');
xlabel(['\alpha (^o)'],'FontSize',fontSize,'FontWeight','bold');
%legend('Image','D_{col}/(D_{col}+S_{void})','Tait',3);
legend('Image','Tait',3);
set(gca,'LineWidth',lineWidth,'FontSize',fontSize,'FontWeight','bold');
axis([60 90 0 1]);
print('-dtiff',[outputDir,'\Tait_Calc_Densities_vs_alpha']);
close;


disp('Done');



%=================================================================
function [blackDistances, meanDistance] = findBlackDistances(edgeImage, testType)

    switch testType
        case 'Black'
            whiteToBlack = 2;
            blackToWhite = 1;
        case 'White'
            whiteToBlack = 1;
            blackToWhite = 2;
    end


    numRows = size(edgeImage,1);
    numCols = size(edgeImage,2);

    % Track down regions of void (black) between columns (white)
    %blackDistance = zeros(numRows,1);
    blackDistances = [];

    for currRow = 1:numRows

        whiteToBlackTransition = find(edgeImage(currRow,:) == whiteToBlack);
        blackToWhiteTransition = find(edgeImage(currRow,:) == blackToWhite);

        if (isempty(whiteToBlackTransition) || isempty(blackToWhiteTransition))
        else
            legitBlackToWhiteTransition = find(whiteToBlackTransition(1) < blackToWhiteTransition);

            if (isempty(legitBlackToWhiteTransition))
                % There is no white to black transition and then back again
                whiteToBlackTransition = [];
                blackToWhiteTransition = [];
            else
                % There is at least one white to black transition
                blackToWhiteTransition = blackToWhiteTransition(legitBlackToWhiteTransition);

                legitWhiteToBlackTransition = find(blackToWhiteTransition(end) > whiteToBlackTransition);

                if (isempty(legitWhiteToBlackTransition))
                    % There is no return from black to white
                    whiteToBlackTransition = [];
                    blackToWhiteTransition = [];
                else
                    whiteToBlackTransition = whiteToBlackTransition(legitWhiteToBlackTransition);
                end
            end


            currBlackDistance = blackToWhiteTransition - whiteToBlackTransition;
            blackDistances = [blackDistances currBlackDistance];


            meanDistance(currRow) = sqrt(sum(currBlackDistance.^2)/length(currBlackDistance));
        end
    end


%=================================================================
function [edgeImage, edgeImage_x, edgeImage_y] = findEdgesImage(thresholdImage)
```

```
whiteToBlack = 1;
blackToWhite = 2;


% From First Row to Last

edgeImage_y = zeros(size(thresholdImage(2:end,:)));
tmp = thresholdImage(2:end,:) - thresholdImage(1:end-1,:);

whiteToBlack_y.index = find(tmp == 1);
blackToWhite_y.index = find(tmp == -1);

[whiteToBlack_y.row, whiteToBlack_y.col] = ind2sub(size(thresholdImage),whiteToBlack_y.index);
[blackToWhite_y.row, blackToWhite_y.col] = ind2sub(size(thresholdImage),blackToWhite_y.index);

edgeImage_y(whiteToBlack_y.index) = whiteToBlack;
edgeImage_y(blackToWhite_y.index) = blackToWhite;


% From First Col to Last

edgeImage_x = zeros(size(thresholdImage(:,2:end)));
tmp = thresholdImage(:,2:end) - thresholdImage(:,1:end-1);

whiteToBlack_x.index = find(tmp == 1);
blackToWhite_x.index = find(tmp == -1);

[whiteToBlack_x.row, whiteToBlack_x.col] = ind2sub(size(thresholdImage),whiteToBlack_x.index);
[blackToWhite_x.row, blackToWhite_x.col] = ind2sub(size(thresholdImage),blackToWhite_x.index);

edgeImage_x(whiteToBlack_x.index) = whiteToBlack;
edgeImage_x(blackToWhite_x.index) = blackToWhite;

edgeImage = edgeImage_x(1:end-1,:) + edgeImage_y(:,1:end-1);
```

Appendix B

Optical relative humidity monitoring of GLAD films

The following images show selected LabVIEW interfaces and block diagrams for the code used to measure GLAD sample transmission spectra changes as a function of changing humidity. Not all code used in the humidity monitoring program is shown as portions (related to testing the humidity performance of GLAD-base capacitive sensors) were documented in [Steele2008].

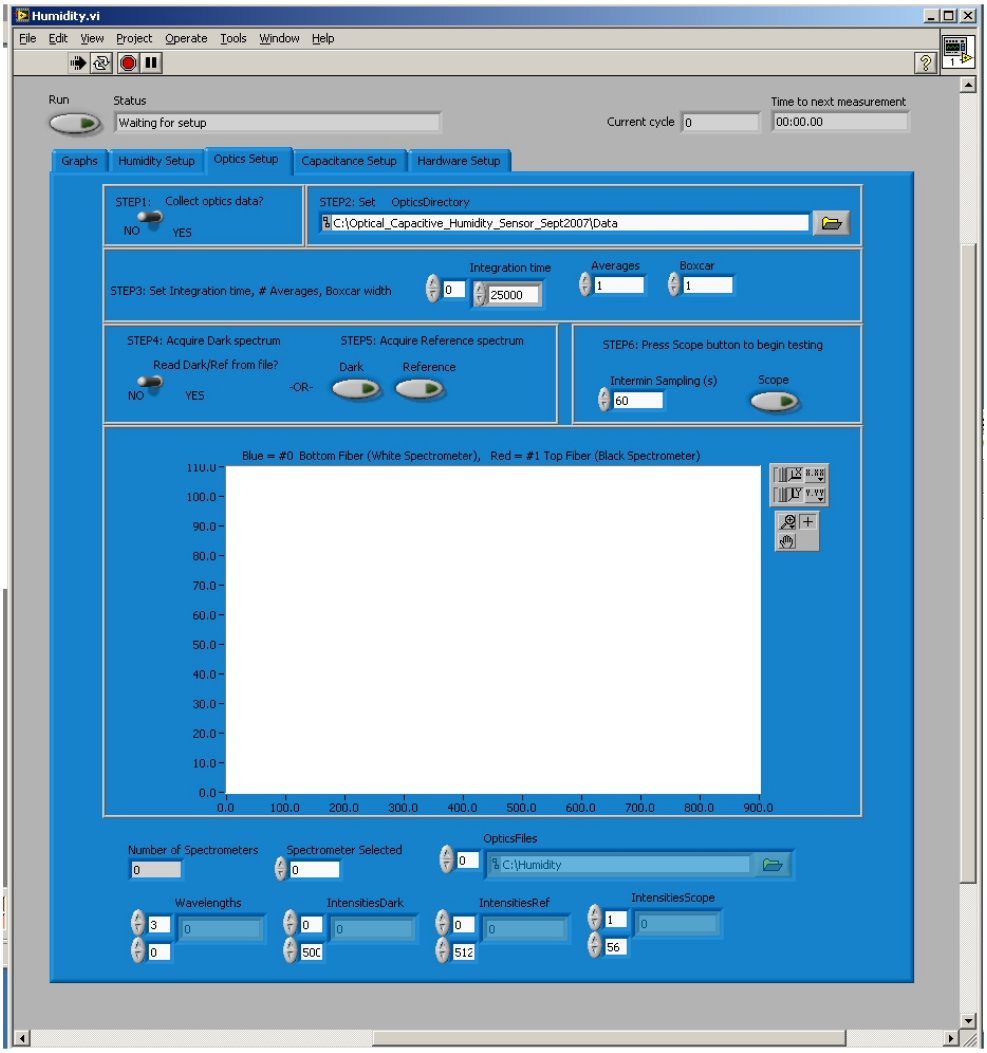


Figure B.1: Main user interface to LabVIEW program which monitors changes in GLAD film transmission spectra as a function of humidity.

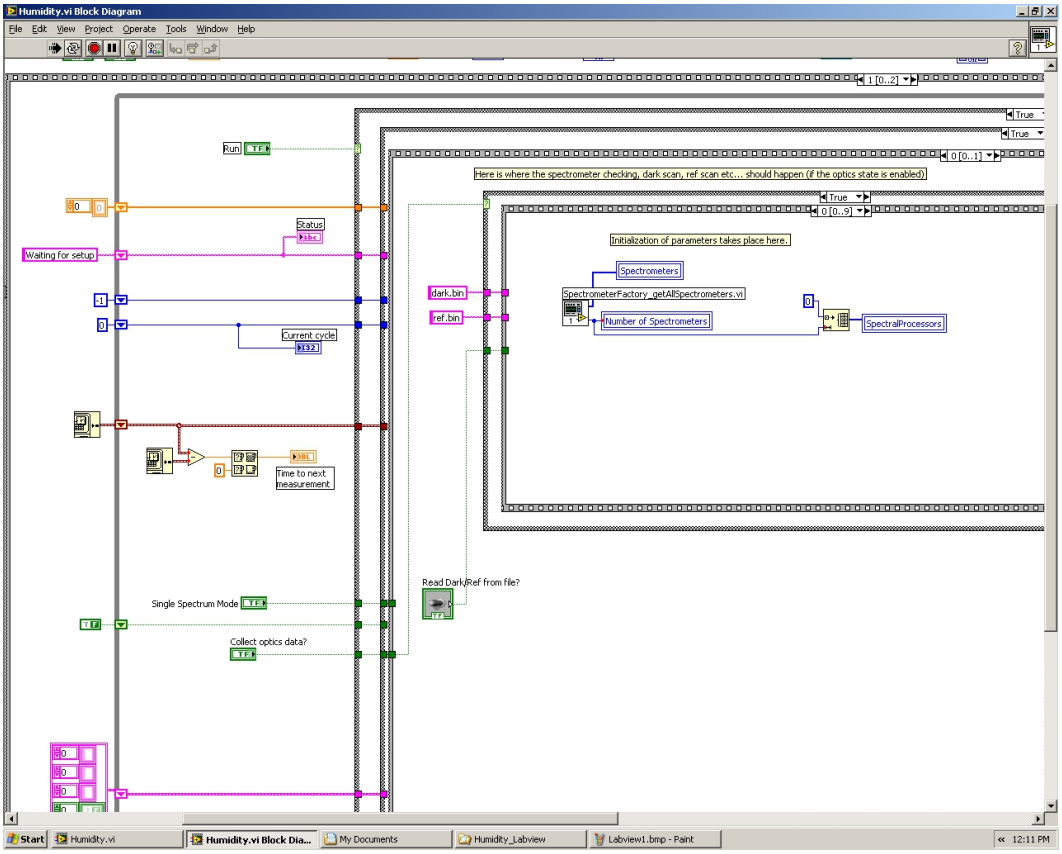


Figure B.2: Underlying LabVIEW code to user interface.

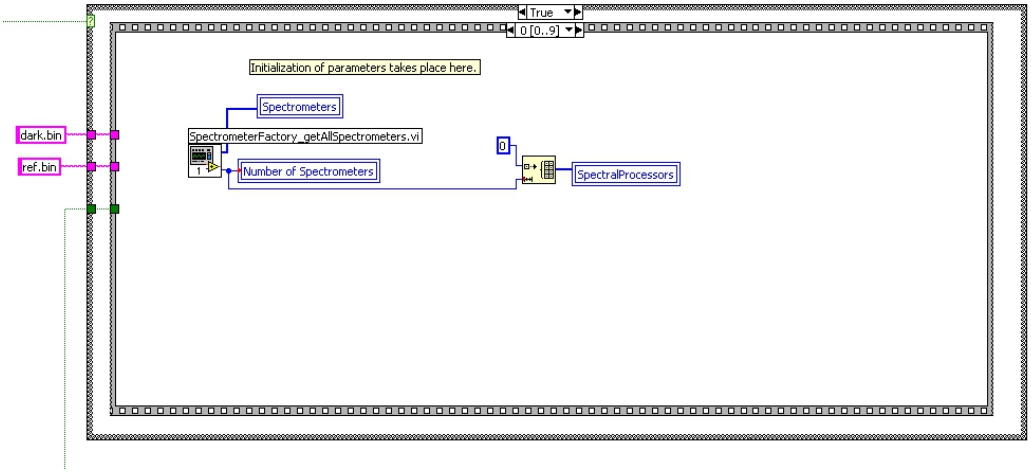


Figure B.3: Main LabVIEW code step 0 for extracting and analyzing transmission spectrum.

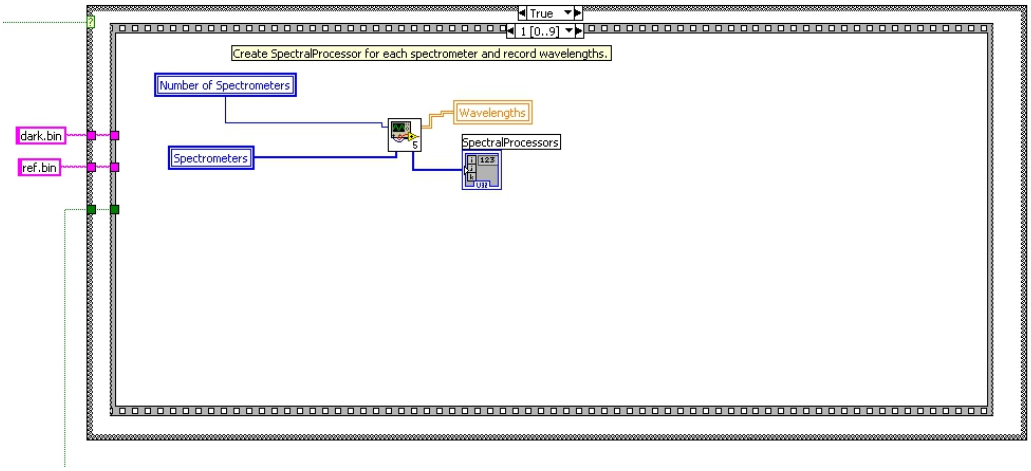


Figure B.4: Main LabVIEW code step 1 for extracting and analyzing transmission spectrum.

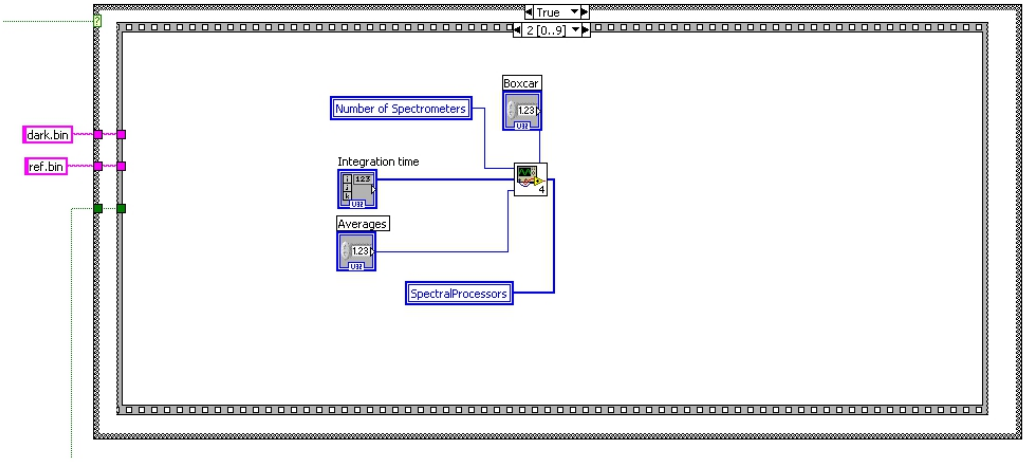


Figure B.5: Main LabVIEW code step 2 for extracting and analyzing transmission spectrum.

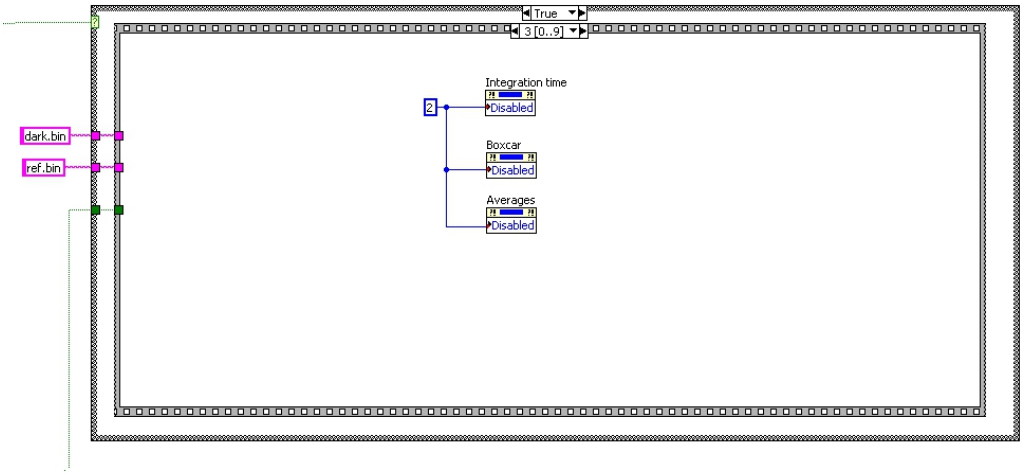


Figure B.6: Main LabVIEW code step 3 for extracting and analyzing transmission spectrum.

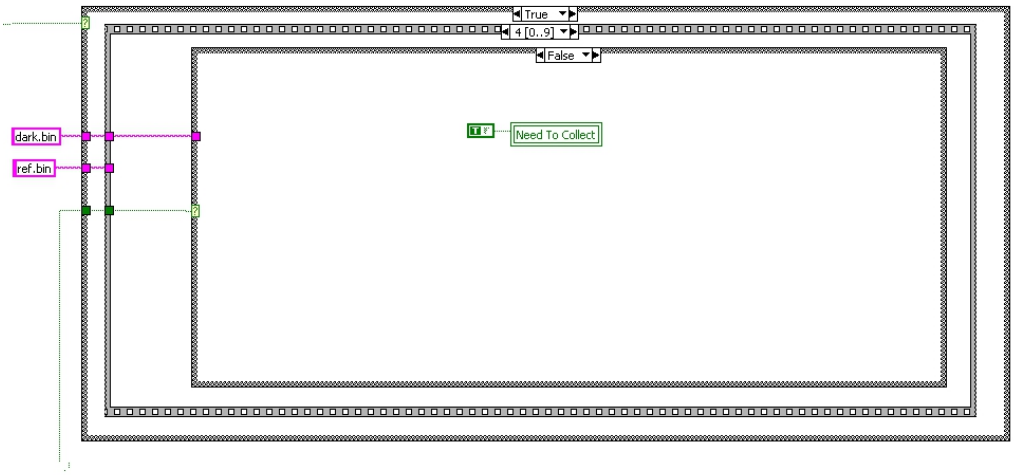


Figure B.7: Main LabVIEW code step 4 for extracting and analyzing transmission spectrum.

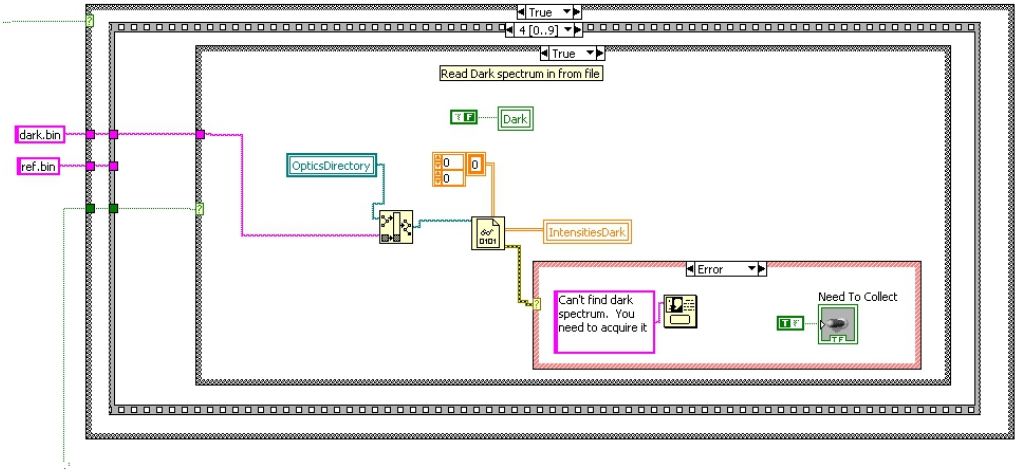


Figure B.8: Main LabVIEW code step 4b for extracting and analyzing transmission spectrum.

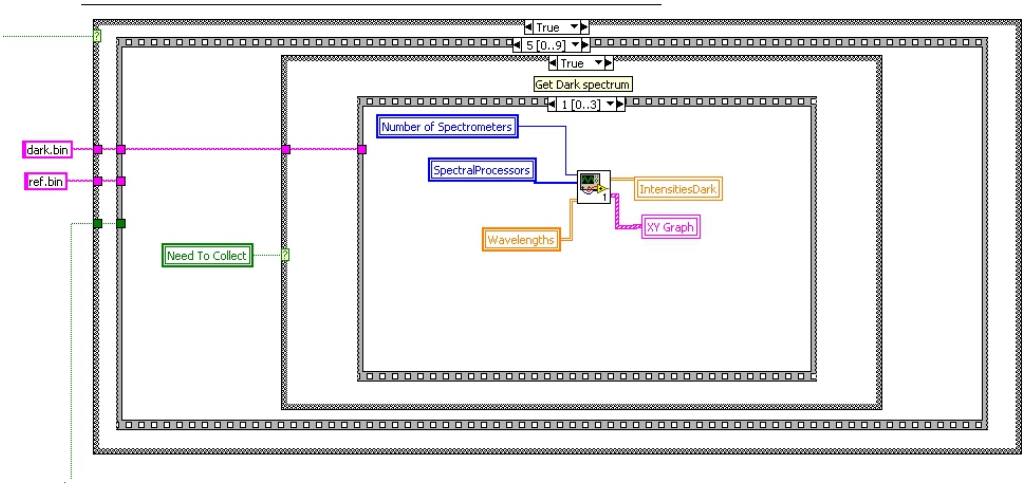


Figure B.9: Main LabVIEW code step 5 for extracting and analyzing transmission spectrum.

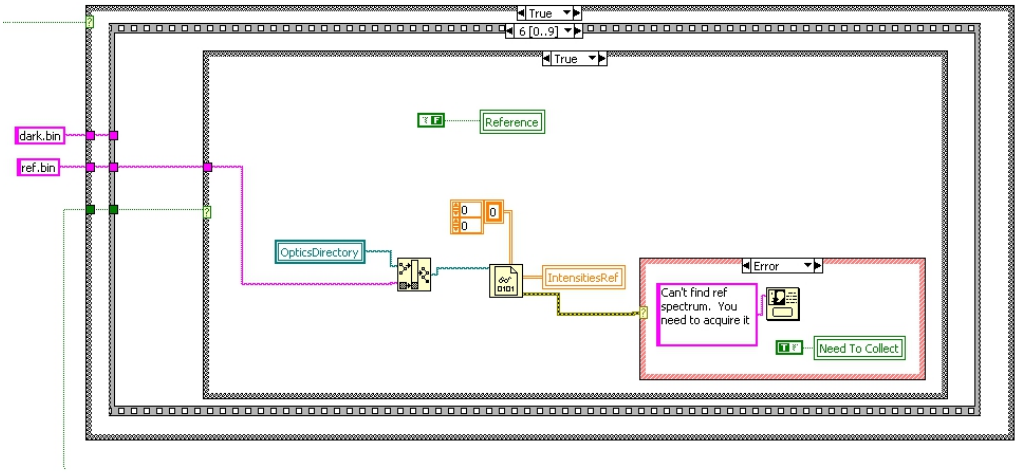


Figure B.10: Main LabVIEW code step 6 for extracting and analyzing transmission spectrum.

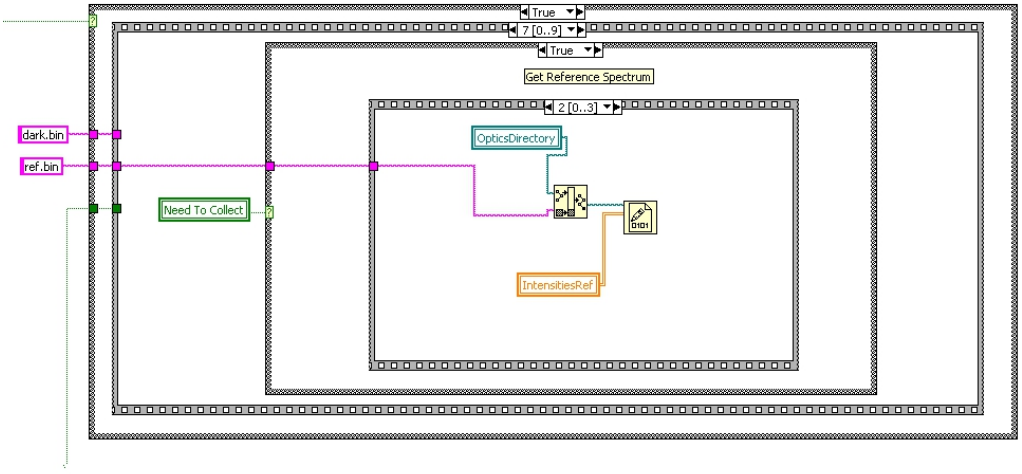


Figure B.11: Main LabVIEW code step 7 for extracting and analyzing transmission spectrum.

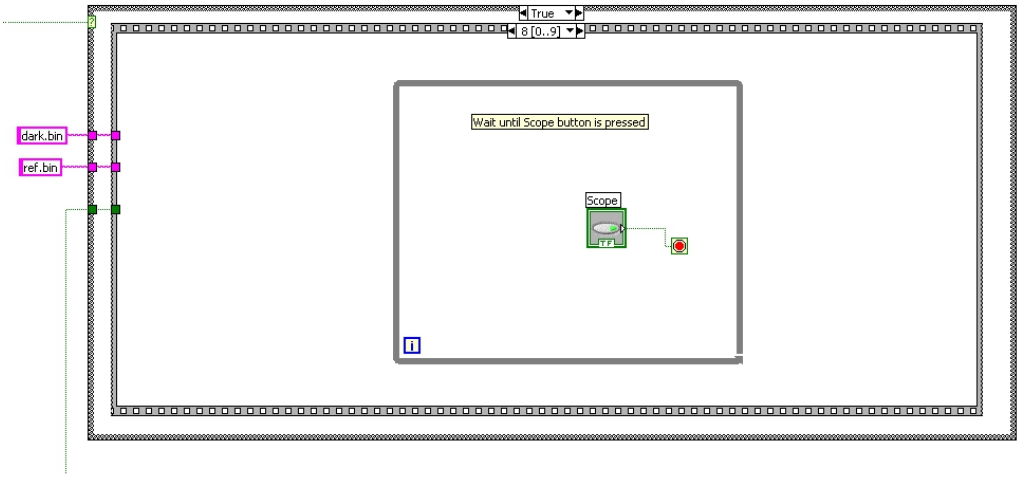


Figure B.12: Main LabVIEW code step 8 for extracting and analyzing transmission spectrum.

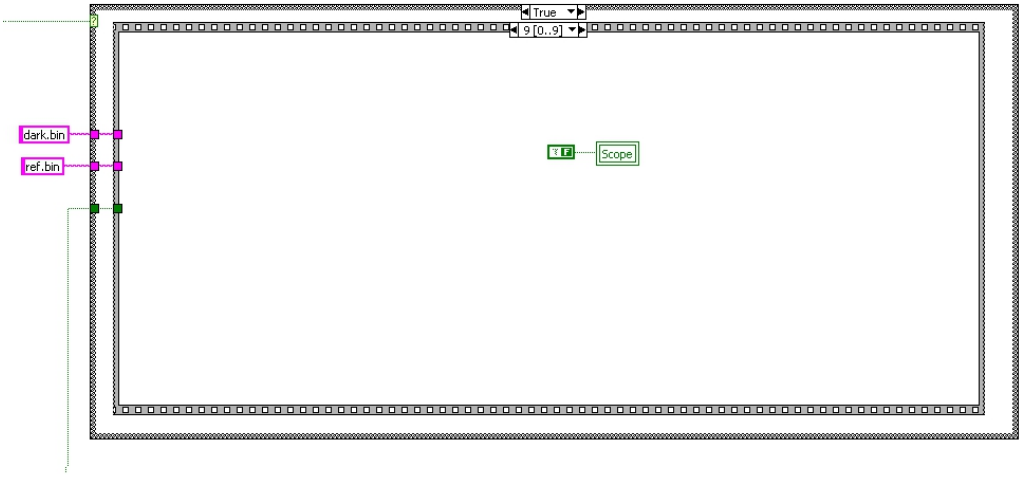


Figure B.13: Main LabVIEW code step 9 for extracting and analyzing transmission spectrum.

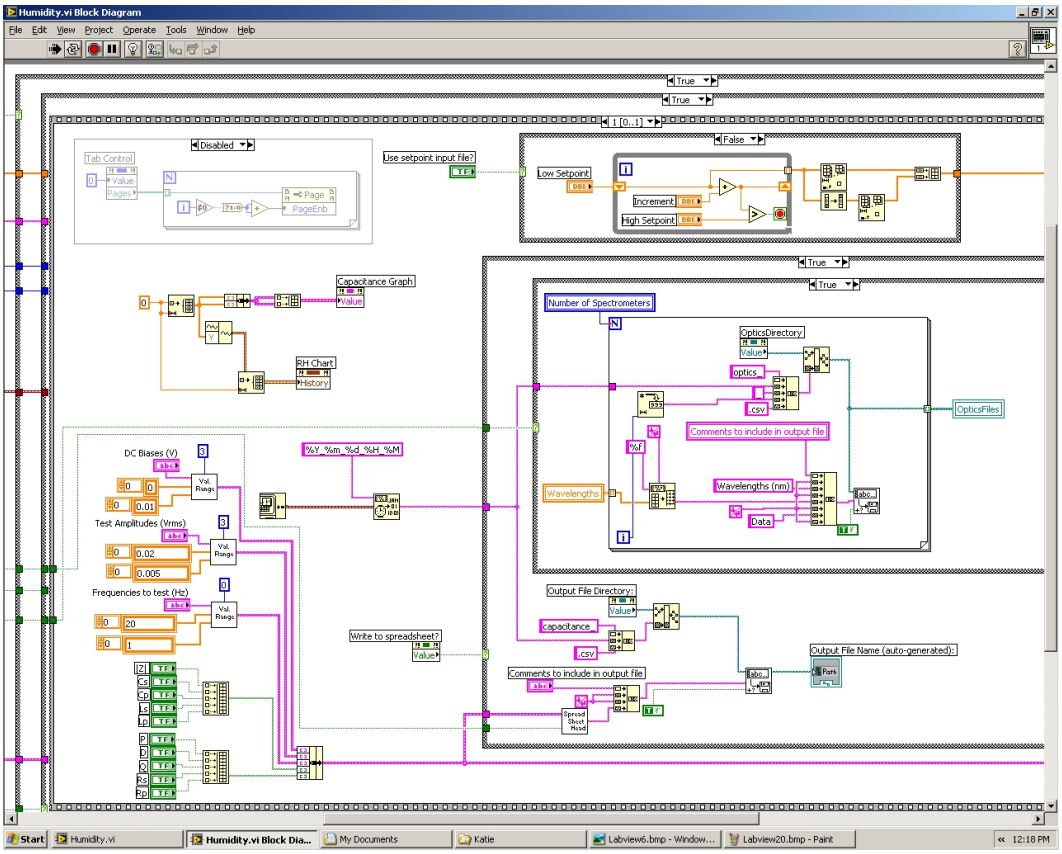


Figure B.14: Main LabVIEW code to display transmission spectrum along with measured humidity.

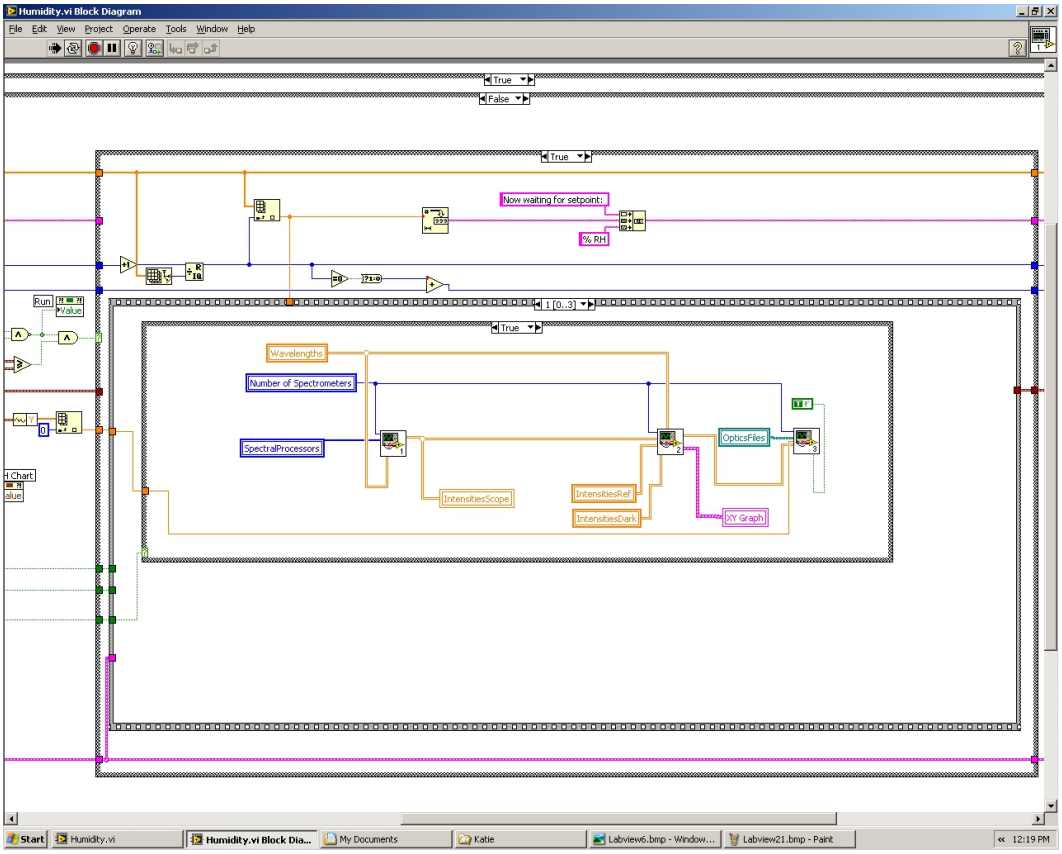


Figure B.15: Main LabVIEW code to calculate differences between sample and reference spectra.

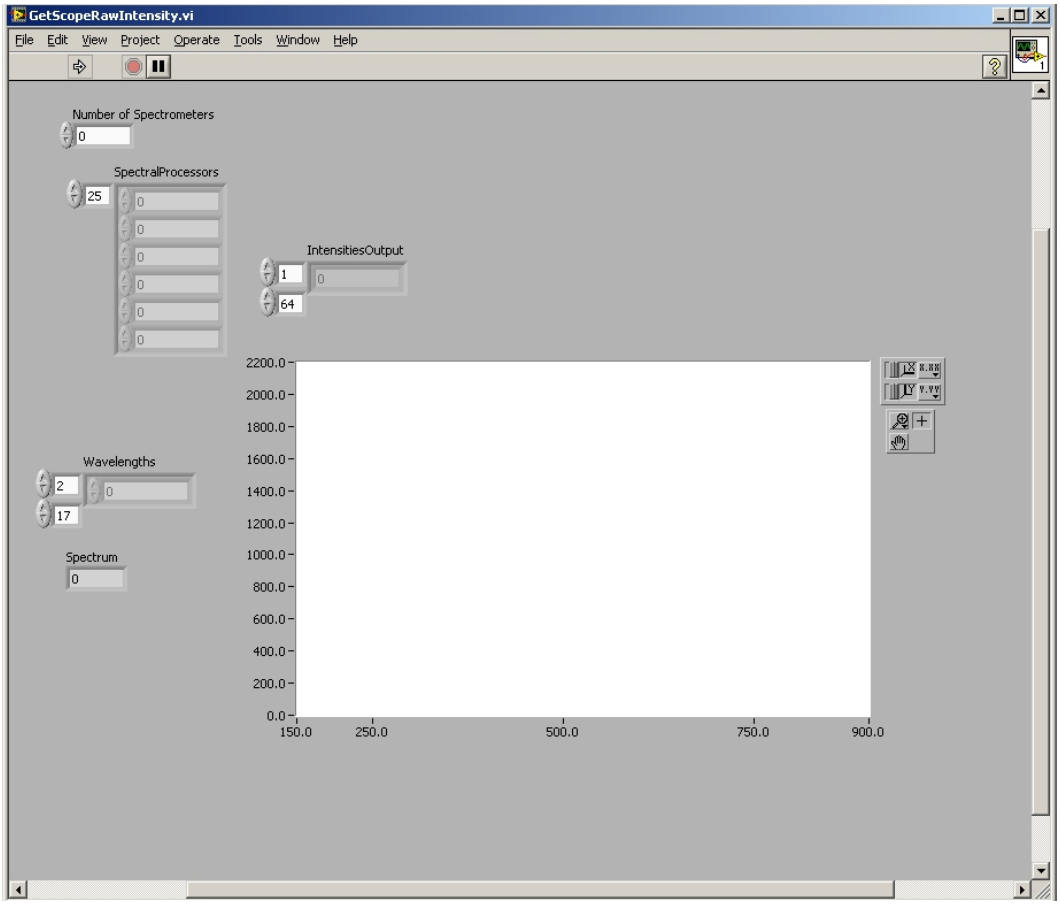


Figure B.16: LabVIEW interface for routine to get raw intensity data from spectrometer.

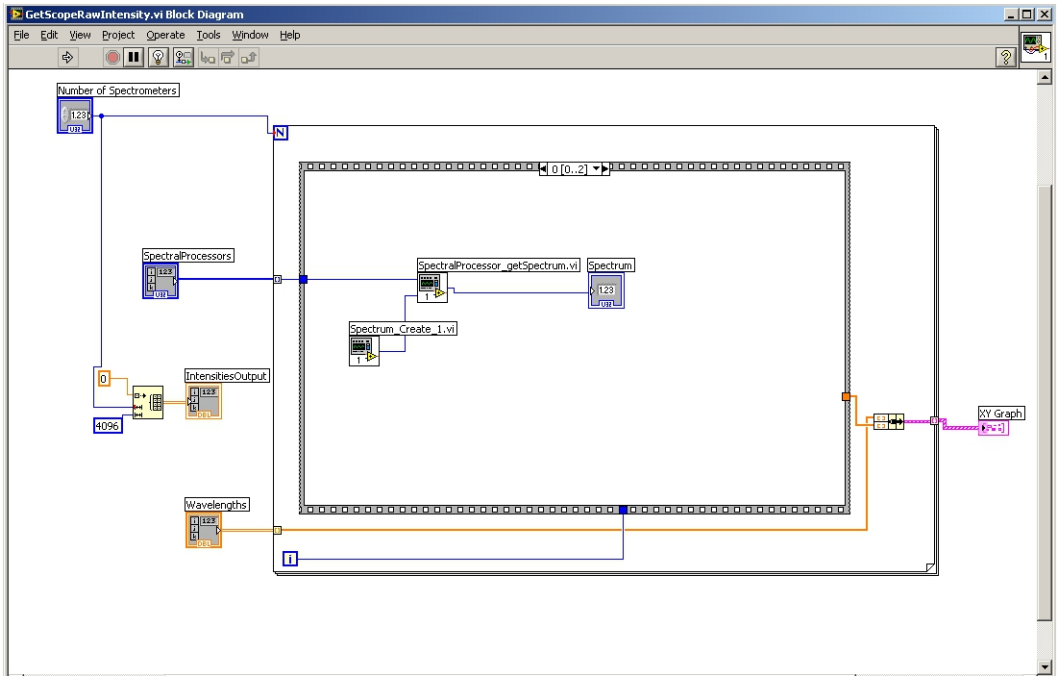


Figure B.17: LabVIEW code step 0 for routine to get raw intensity data from spectrometer.

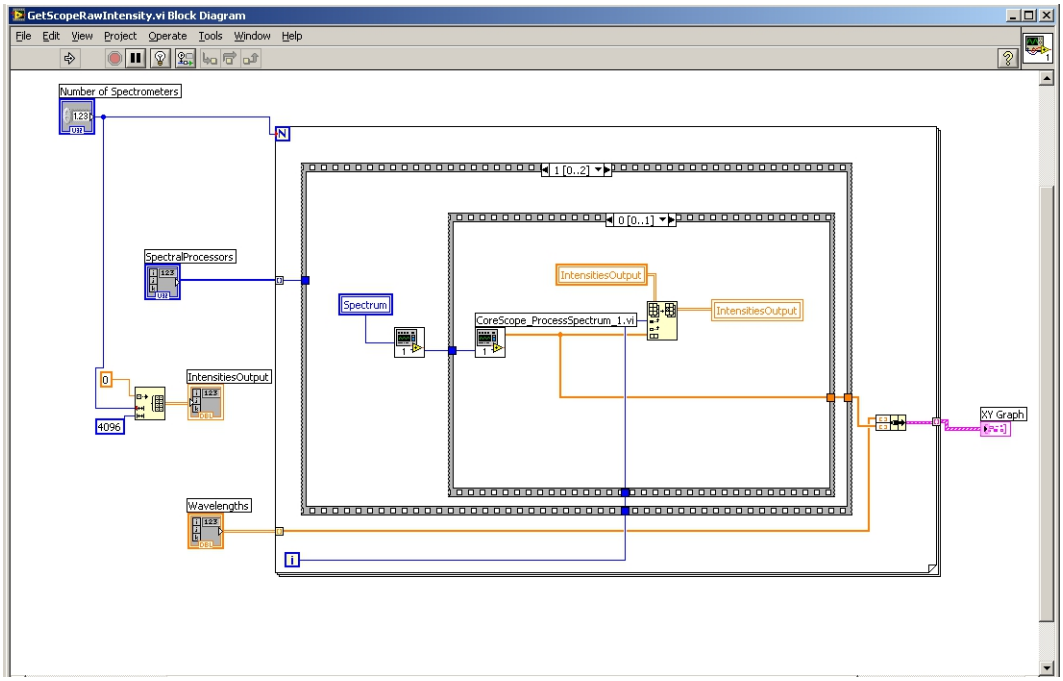


Figure B.18: LabVIEW code step 1 for routine to get raw intensity data from spectrometer.

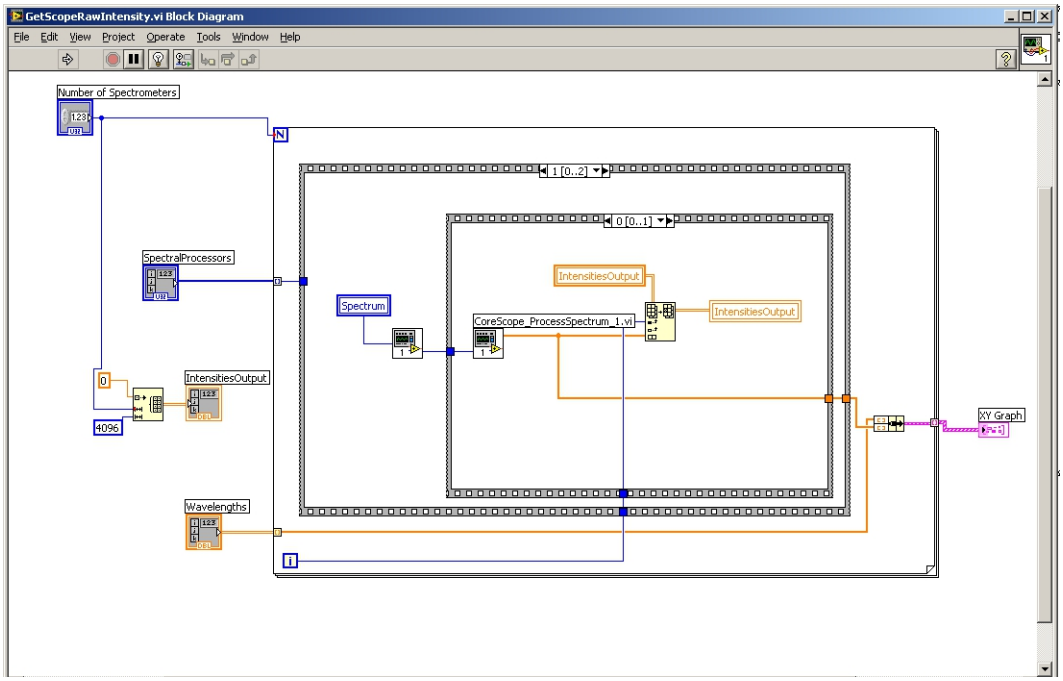


Figure B.19: LabVIEW code step 2 for routine to get raw intensity data from spectrometer.

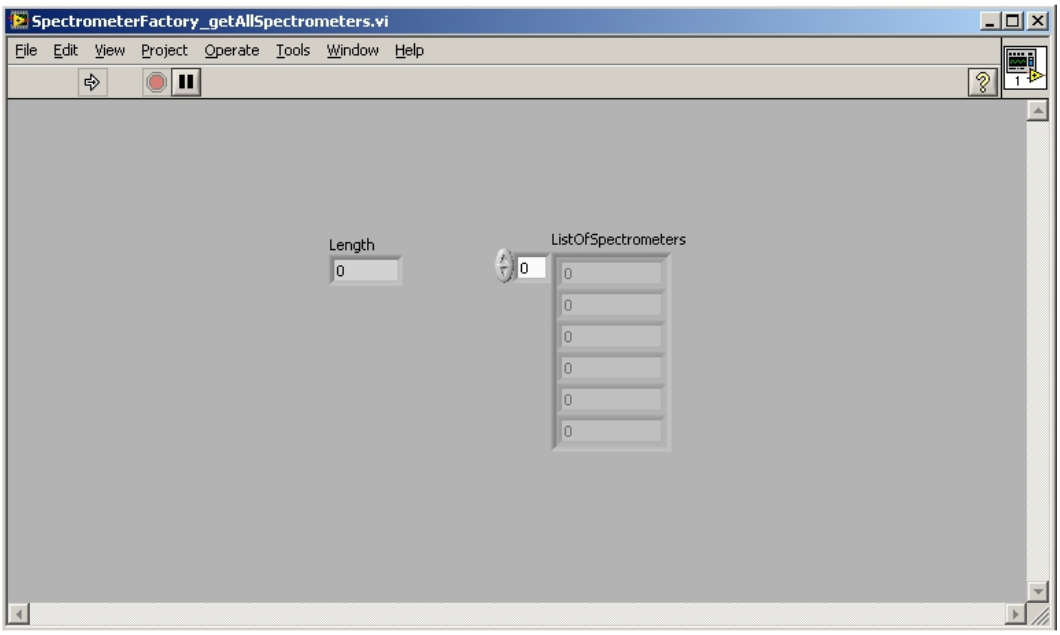


Figure B.20: LabVIEW interface for routine that identifies spectrometers.

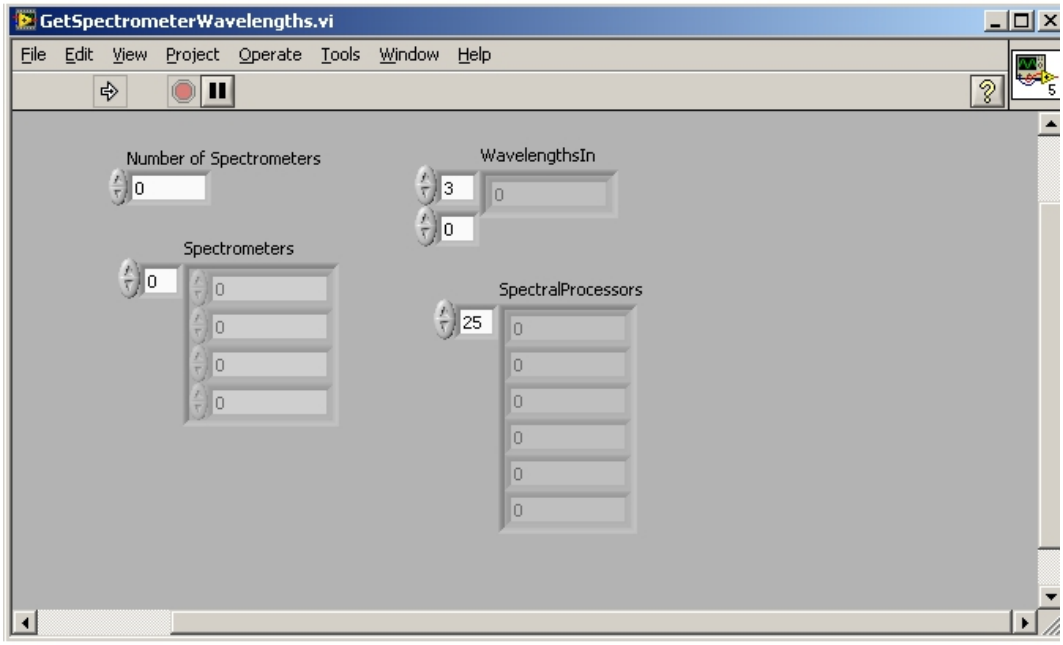


Figure B.21: LabVIEW interface for routine that gets wavelength array from spectrometer.

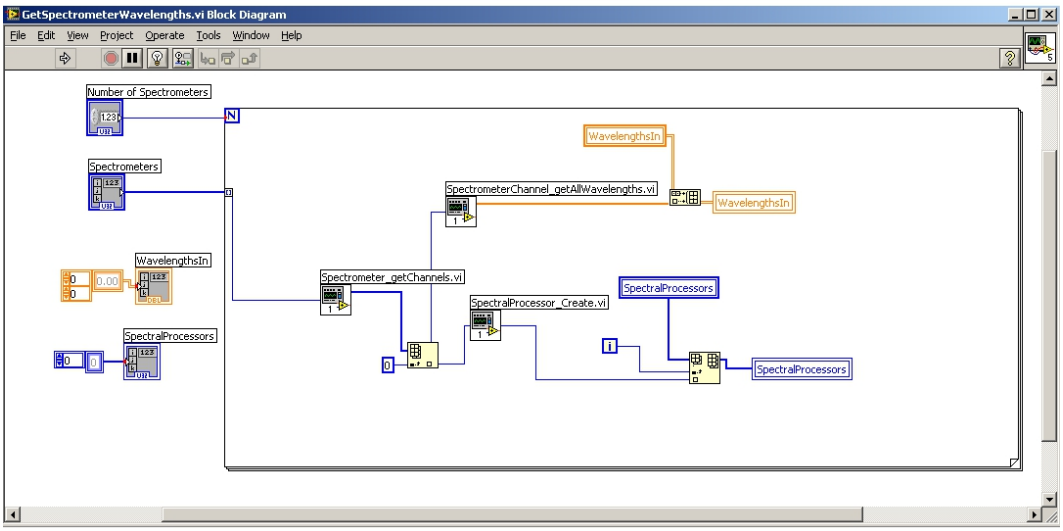


Figure B.22: LabVIEW code for routine that gets wavelength array from spectrometer.

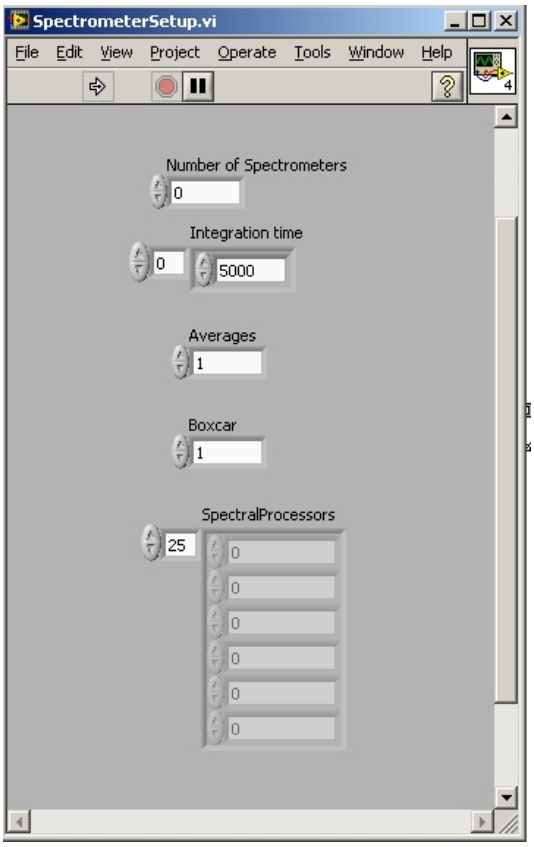


Figure B.23: LabVIEW interface for routine that sets spectrometer parameters.

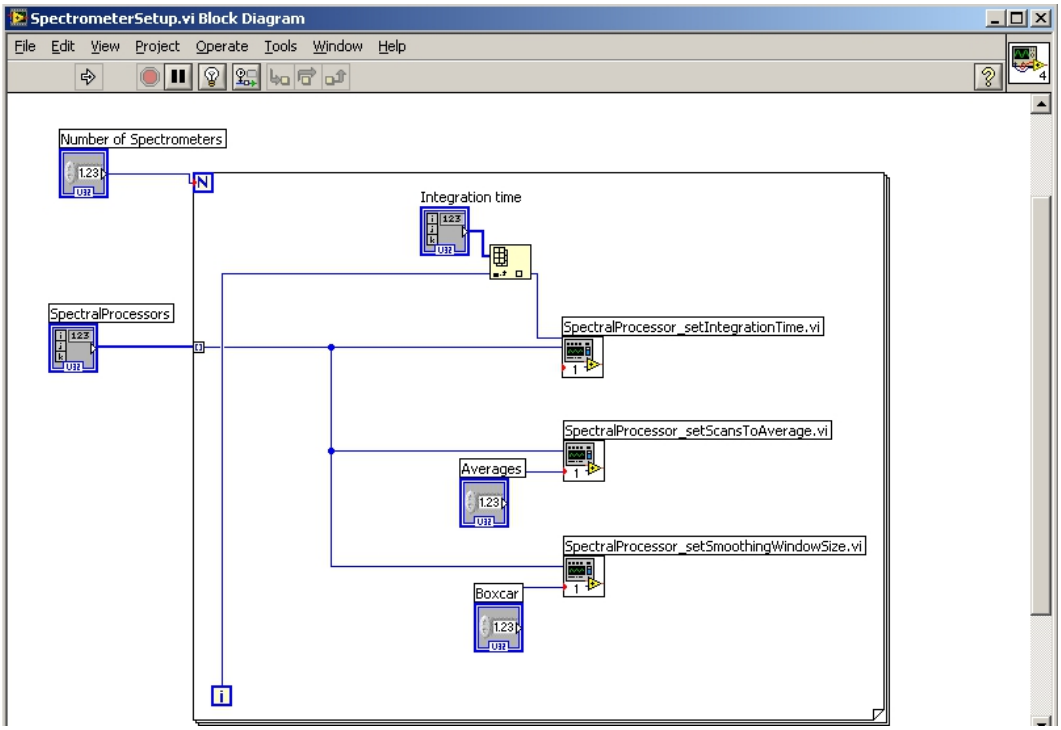


Figure B.24: LabVIEW code for routine that sets spectrometer parameters.

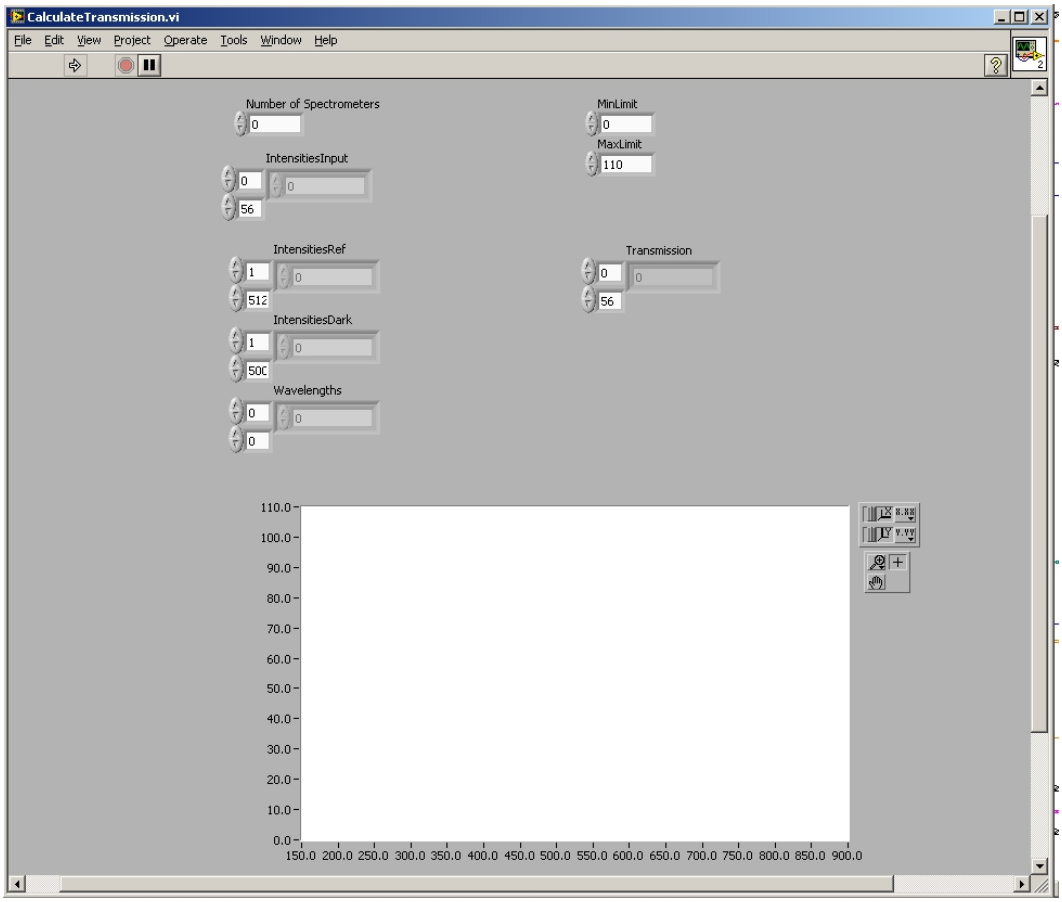


Figure B.25: LabVIEW interface for routine that calculates transmission spectrum.

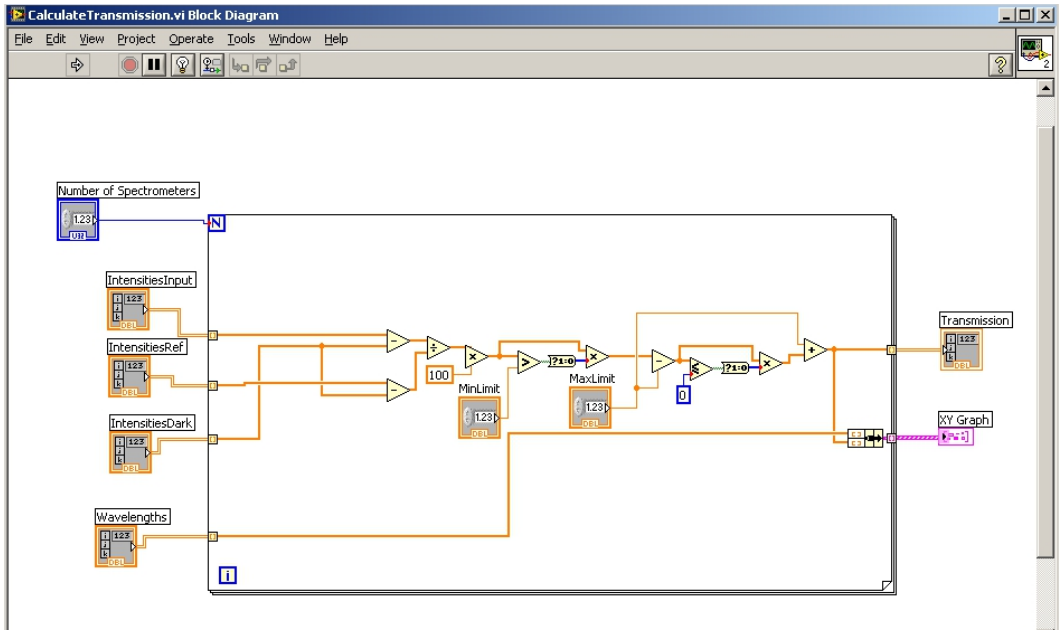


Figure B.26: LabVIEW code for routine that calculates transmission spectrum.

Appendix C

Image processing of FIB-SEM image slices

C.1 Code to track fiducial and image selection from image to image

```
% Code to track fiducial marks from image to image in each series
% Code to search for image selection for series stitching is essentially
% the same, with additional lines of code to help find the image selection
% region when the next series is reached

clear all
close all
clc

% Load up info on this particular set of FIB series
FIBscan_June09_TiO2VP5um

% Plot directory
plotDir = ['C:\My_Documents\Research\Research_Areas\ThreeDimGLAD_with_FIB\', ...
           'Processed_FIB_Scans\TopDownView\Processed_Scans_June09'];
doPlots = 1;
if(exist(plotDir,'dir') == 0)  mkdir(plotDir); end

% Coarse and Fine search parameters
searchParameters


%-------------------------------------------------------------------------
% Start of analysis

numSeries = length(scans.directories);

disp('Getting image file names ...');
[dataSeries] = getImageFileNames(numSeries, scans);

disp('Making plot directories ...');
[curr_plotDir, main_fiducial_plotDir] = ...
               makePlotDirectories(numSeries, scans, plotDir);


disp('Getting required image regions to do density analysis etc ...');
    for series = 1:numSeries
        output_plotDir = [curr_plotDir{series}];

        prev_fiducial = [];
        use_prev_scale = [];
        use_prev_shift = [];

        % Create output files
        [fid_fiducial_shifts, fid_select_shifts] = ...
                       createOutputFiles(plotDir, series, ...
                           scans.image_res_nm, maxColumnDiameter_pixel);

        for slice = (scans.firstImage(series)+1):(scans.lastImage(series)+1)
```

```
            disp([' ... series ',num2str(series),', slice ',num2str(slice)]);

            % Load current image
            if (scans.multiimage(series) == 1)
                currentImage = imread([char(scans.directories(series)),'\', ...
                                       char(scans.multiimage_fileName(series))], ...
                                       'tif',slice);
            else
                currentImage = loadImage([char(scans.directories(series)),'\', ...
                                          dataSeries(series).fileNames(slice).name]);
            end

            figure;image(currentImage/6);colormap('gray')

            % Determine best fit to previous image
            if (isempty(prev_select))
                disp('     ... Finding first fiducial image');
                [best_match_fiducial] = findFirstImageSelection(currentImage, ...
                                          scans.fiducial.size_pixel, ...
                                          scans.fiducial.centres_pixel, series, 0);

            else

                fiducial_plotDir = [char(main_fiducial_plotDir(series)), ...
                                    '\Fiducial_Slice',num2str(slice)];
                if ((exist(fiducial_plotDir,'dir') == 0) && (doPlots))
                    mkdir(fiducial_plotDir);
                end

                % Match current image to previous image, by moving
                % image selection around slightly
                % Use previous stretch to roughly adjust contrast of image
                disp('     ... Finding fiducial image');
                [best_match_fiducial] = findBestMatchTwoImages(currentImage, ...
                                                        scans.fiducial, ...
                                                        series, slice, ...
                                                        prev_fiducial, ...
                                                        'FIDUCIAL', doPlots, ...
                                                        fiducial_plotDir, 0);

            end

            best_match_select.y1 = round(best_match_select.y1 - scans.offset_y);
            best_match_select.y2 = round(best_match_select.y2 - scans.offset_y);
            best_match_select.x1 = round(best_match_select.x1 + scans.offset_x);
            best_match_select.x2 = round(best_match_select.x2 + scans.offset_x);

            % Image to analyze
            fiducial_Image = currentImage(best_match_fiducial.y1:best_match_fiducial.y2, ...
                                          best_match_fiducial.x1:best_match_fiducial.x2);

            disp('     ... Plotting fiducial image');
            plotSelectedImage(fiducial_Image, series, slice, ...
                              best_match_fiducial.x_shift, best_match_fiducial.y_shift, ...
                              slice, 'FIDUNFILT', char(main_fiducial_plotDir(series)));


            % Store current selected image to compare next image to
            disp('     ... Storing image info for next iteration');
            prev_fiducial = best_match_fiducial;
            prev_fiducial.image = fiducial_Image;


            fprintf(fid_fiducial_shifts, ...
                   '%6d, %6d, %6d, %6d, %6d, %6d, %6d \n', ...
                   slice, best_match_fiducial.x1, best_match_fiducial.x2, ...
                   best_match_fiducial.y1, best_match_fiducial.y2, ...
                   best_match_fiducial.x_shift, best_match_fiducial.y_shift);
        end
        fclose(fid_fiducial_shifts);
    end

disp('Done');
close all;

%================================================================================

% searchParameters

scans.fiducial.coarse.x_buffer = 40;
scans.fiducial.coarse.y_buffer = 40;
```

```
scans.fiducial.coarse.x_step = 5;
scans.fiducial.coarse.y_step = 5;
scans.fiducial.fine.x_buffer = 5;
scans.fiducial.fine.y_buffer = 5;
scans.fiducial.fine.x_step = 1;
scans.fiducial.fine.y_step = 1;

%================================================================================

% Image selection and fiducial mark properties
% Image resolution properties

% Series 1: FIB through top coating and top of photoresist.  Start of
%           slicing into GLAD film
% Series 2: Slicing begins from first image
% Series 3: Slicing begins from first image
% Series 4: no FIBing.  Image just drifts, but isn't sliced.  Not processed
% Series 5: Slices are very thin.  Very gradual slicing through film.
% Series 6: Need to read in multi-image TIF
% Series 7: Instrument error, use another scan (lower res, larger slice res)
% Series 8: High-res Images down to substrate

scans.directories = { ...
    'C:\My_Documents\Research\Research_Areas\ThreeDimGLAD_with_FIB\FIB_Scans\090628_DVick\series_1', ...
    'C:\My_Documents\Research\Research_Areas\ThreeDimGLAD_with_FIB\FIB_Scans\090629_DVick\series_2', ...
    'C:\My_Documents\Research\Research_Areas\ThreeDimGLAD_with_FIB\FIB_Scans\090630_DVick\series_3', ...
    'C:\My_Documents\Research\Research_Areas\ThreeDimGLAD_with_FIB\FIB_Scans\090702_DVick\series_4', ...
    'C:\My_Documents\Research\Research_Areas\ThreeDimGLAD_with_FIB\FIB_Scans\090702_DVick\series_5', ...
    'C:\My_Documents\Research\Research_Areas\ThreeDimGLAD_with_FIB\FIB_Scans\090703_DVick\series_6', ...
    'C:\My_Documents\Research\Research_Areas\ThreeDimGLAD_with_FIB\FIB_Scans\090226_KKrause', ...
    'C:\My_Documents\Research\Research_Areas\ThreeDimGLAD_with_FIB\FIB_Scans\090703_DVick\series_8'};

scans.prefix = 'FIBSLICE';
scans.extension = 'tif';
scans.multiimage = [0,0,0,0,0,1,0,0];  % Series 6 is a multi-image file!
scans.multiimage_fileName = {'','','','','','TiO2_GLAD_102.tif','',''};

scans.useScans = [1 2 3 5 6 7 8];

scans.film_thickness_nm = 5070;


% Resolution parameters
scans.magnification =            [ 13.99, 14.36, 19.01, 29.38, 29.08, 38.91,  27.6, 58.36];
scans.slice_res_pixel =          [   512,   512,   512,   512,   512,   512,   256,   512];
scans.x_size_nm =                [  8171,  8163,  6172,  3993,  4031,  3013,  4096,  2009];
scans.x_size_pixel =             [  2048,  2048,  2048,  2048,  2048,  2048,  1024,  2048];
scans.y_size_pixel =             [  1536,  1536,  1536,  1536,  1536,  1536,   768,  1536];


scans.image_res_nm =  scans.x_size_nm ./ scans.x_size_pixel;
scans.slice_res_nm =  scans.x_size_nm ./ scans.slice_res_pixel;


scans.firstImage =               [    45,     0,     0,     0,    0,     0,    225,   0];
scans.lastImage =                [    50,    92,   132,    83,   70,     63,   305,  33];
scans.thickness =  (scans.lastImage - scans.firstImage) .* scans.slice_res_nm;

scans.offset = [16, 16, 12, 8, 6, 700, 16, 0 ];
scans.additionalImages = [5, 0, 0, 0, 0, 0, 0, -3];

cumThickness = cumsum(scans.thickness(scans.useScans(end:-1:1)) + ...
                scans.offset(scans.useScans(end:-1:1)))  - 2*scans.slice_res_nm(1);
scans.cumThickness(scans.useScans) = cumThickness(end:-1:1);

% Fiducial mark in firstImage
scans.fiducial.x_start_pixel =   [  1750,    25,  1650,  1550,  1800,  1780,  3400,  350];
scans.fiducial.x_end_pixel =     [  2000,   150,  1950,  1800,  2000,  2000,  3760,  500];
scans.fiducial.y_start_pixel =   [   300,   150,   425,   800,   700,   160,  1200,  700];
scans.fiducial.y_end_pixel =     [  1250,  1000,  1250,  1150,  1300,   900,  2200, 1400];
scans.fiducial.size_pixel =    [[scans.fiducial.x_end_pixel - ...
                                 scans.fiducial.x_start_pixel]', ...
                                [scans.fiducial.y_end_pixel- ...
                                 scans.fiducial.y_start_pixel]'];
scans.fiducial.centres_pixel = [[round((scans.fiducial.x_end_pixel + ...
                                   scans.fiducial.x_start_pixel)/2)]', ...
                                [round((scans.fiducial.y_end_pixel + ...
                                   scans.fiducial.y_start_pixel)/2)]'];

% Image section in firstImage
scans.selection.x_start_pixel =  [   250,   650,   500,   800,  800,   800,  1900, 1000];
```

```
scans.selection.x_end_pixel =    [  1500,  1500,  1350,  1500, 1400,  1400,  2400, 1700];
scans.selection.y_start_pixel =  [   750,   750,   800,   750,  800,   800,  2000,  700];
scans.selection.y_end_pixel =    [  1500,  1500,  1500,  1500, 1500,  1400,  2500, 1500];
scans.selection.size_pixel =     [[scans.selection.x_end_pixel - ...
                                   scans.selection.x_start_pixel]', ...
                                  [scans.selection.y_end_pixel - ...
                                   scans.selection.y_start_pixel]'];
scans.selection.centres_pixel =  [[round((scans.selection.x_end_pixel + ...
                                   scans.selection.x_start_pixel)/2)]', ...
                                  [round((scans.selection.y_end_pixel + ...
                                   scans.selection.y_start_pixel)/2)]'];

% Contast adjustment
for numSeries = 1:length(scans.directories)
    scans.series(numSeries).contrastAdj = ones(1,(scans.lastImage(numSeries) + 1));
end

%=========================================================================

% Search for the fiducial image match in coarse steps and then fine steps

function [best_match_fine] = findBestMatchTwoImages(fullImage, image_selection, ...
                               series, slice, prev, prefix, doPlots, plotDir, verbose)

    % Coarse Steps
    disp('        ... Coarse Steps');
    [best_match_coarse] = findBestMatch(fullImage, ...
                                        image_selection.size_pixel, ...
                                        image_selection.centres_pixel, ...
                                        series, slice, prev,  ...
                                        [prev.x_shift-image_selection.coarse.x_buffer, ...
                                           prev.x_shift+image_selection.coarse.x_buffer], ...
                                        [prev.y_shift-image_selection.coarse.y_buffer, ...
                                           prev.y_shift+image_selection.coarse.y_buffer], ...
                                        image_selection.coarse.x_step, ...
                                        image_selection.coarse.y_step, ...
                                        [prefix,'Coarse'], doPlots, [plotDir], verbose);

    % Fine Steps
    disp('        ... Fine Steps');
    [best_match_fine] = findBestMatch(fullImage, ...
                                        image_selection.size_pixel, ...
                                        image_selection.centres_pixel, ...
                                        series, slice, prev,  ...
                                        [best_match_coarse.x_shift-image_selection.fine.x_buffer, ...
                                          best_match_coarse.x_shift+image_selection.fine.x_buffer], ...
                                        [best_match_coarse.y_shift-image_selection.fine.y_buffer, ...
                                          best_match_coarse.y_shift+image_selection.fine.y_buffer], ...
                                        image_selection.fine.x_step, ...
                                        image_selection.fine.y_step, ...
                                        [prefix,'Fine'], doPlots, [plotDir], verbose);

%=========================================================================

% Routine to find best match between a fiducial (or other subimage)
% image within the current image and the previous fiducial
% (from a previous slice or a previous series)

function [best_match] = findBestMatch(curr_image, selection_size, selection_centres, ...
                                      count, count2, prev,  x_step_max,y_step_max, ...
                                      x_step_size, y_step_size, prefix, doPlots, ...
                                      output_directory, verbose)

    fontSize = 20;
    lineWidth = 2;

    xsize = size(curr_image,2);
    ysize = size(curr_image,1);

    t1 = clock;
    min_diff = 999999999999;
    match_counter = 0;
    best_match = [];
    for x_step_curr = x_step_max(1):x_step_size:x_step_max(2)
        for y_step_curr = y_step_max(1):y_step_size:y_step_max(2)


            x1 = max(1, round(selection_centres(count,1) - ...
                              selection_size(count,1)/2 + x_step_curr));
            x2 = min(xsize, round(selection_centres(count,1) + ...
                              selection_size(count,1)/2 + x_step_curr));
```

```
y1 = max(1, round(selection_centres(count,2) - ...
                  selection_size(count,2)/2 + y_step_curr));
y2 = min(ysize, round(selection_centres(count,2) + ...
                  selection_size(count,2)/2 + y_step_curr));


curr.x_length = (x2-x1+1);
curr.y_length = (y2-y1+1);


% If the size of the current fiducial is getting too
% small (e.g. going off the side of the original image)
% the diregard this fiducial
x_diff_length = abs(curr.x_length - prev.x_length);
y_diff_length = abs(curr.y_length - prev.y_length);


max_diff_length = 10;

if  (( x_diff_length <= max_diff_length) && ...
     ( y_diff_length <= max_diff_length))

    curr.image = curr_image(y1:y2,x1:x2);


    if (curr.y_length > prev.y_length)
        curr.image = curr.image(1:prev.y_length,:);

        y2 = prev.y_length + y1 - 1;

        curr.y_length = prev.y_length;
        resize_prev.image = prev.image;
    elseif (curr.y_length < prev.y_length)
        resize_prev.image = prev.image(1:curr.y_length,:);
    else
        resize_prev.image = prev.image;
    end

    if (curr.x_length > prev.x_length)
        curr.image = curr.image(:,1:prev.x_length);

        x2 = prev.x_length + x1 - 1;

        curr.x_length = prev.x_length;
    elseif (curr.x_length < prev.x_length)
        resize_prev.image = resize_prev.image(:,1:curr.x_length);
    end


    % Find difference in image maps
    diff_fiducial_image = sum(sum(abs((curr.image - resize_prev.image))));

    if (diff_fiducial_image >= min_diff)
        if (verbose)
            disp(['          ... x_step: ',num2str(x_step_curr), ...
                ', y_step: ',num2str(y_step_curr), ...
                ', (x1,x2): (',num2str(x1),',',num2str(x2), ...
                '), (y1,y2): (',num2str(y1),',',num2str(y2), ...
                '), selection length (x,y): (',num2str(curr.x_length), ...
                ',',num2str(curr.y_length), ...
                '), diff_length (x,y): (',num2str(x_diff_length), ...
                ',',num2str(y_diff_length),'), image diff: ',num2str(diff_fiducial_image)]);
        end

    else
        if (verbose)
            disp(['          ... x_step: ',num2str(x_step_curr), ...
                ', y_step: ',num2str(y_step_curr), ...
                ', (x1,x2): (',num2str(x1),',',num2str(x2), ...
                '), (y1,y2): (',num2str(y1),',',num2str(y2), ...
                '), selection length (x,y): (',num2str(curr.x_length), ...
                ',',num2str(curr.y_length), ...
                '), diff_length (x,y): (',num2str(x_diff_length), ...
                ',',num2str(y_diff_length),'), image diff: ',num2str(diff_fiducial_image), ...
                ',  BEST MATCH!']);
        end
        match_counter = match_counter + 1;
        min_diff = diff_fiducial_image;
        best_match.x_shift = x_step_curr;
        best_match.y_shift = y_step_curr;
        best_match.image_delta = diff_fiducial_image;
```

```
                    best_match.image = curr.image;
                    best_match.x_length = curr.x_length;
                    best_match.y_length = curr.y_length;
                    best_match.x1 = x1;
                    best_match.x2 = x2;
                    best_match.y1 = y1;
                    best_match.y2 = y2;


                    if (doPlots)
                        % NOTE: Gray colormap: black = 1, white = 66
                        figure('Visible','Off');
                        subplot(2,1,1);image(resize_prev.image*66/255);
                        colormap('gray');hold on;
                        ylabel('Reference Selection','FontSize',fontSize, ...
                                'LineWidth',lineWidth,'FontWeight','bold');
                        title(['X offset: ',num2str(x_step_curr),', Y offset: ', ...
                                num2str(y_step_curr),', Diff: ',num2str(diff_fiducial_image)], ...
                                'FontSize',fontSize,'LineWidth',lineWidth,'FontWeight','bold');
                        set(gca,'LineWidth',lineWidth,'FontSize',fontSize,'FontWeight','bold');
                        subplot(2,1,2);image(curr.image*66/255);colormap('gray');
                        hold on;
                        ylabel('Current Selection','FontSize',fontSize, ...
                                'LineWidth',lineWidth,'FontWeight','bold');
                        set(gca,'LineWidth',lineWidth,'FontSize',fontSize,'FontWeight','bold');
                        print('-dtiff','-r300',[output_directory,'\',prefix, ...
                                '_MATCH_',num2str(match_counter),'_ser',num2str(count), ...
                                '_sli',num2str(count2),'_xshift',num2str(x_step_curr), ...
                                '_yshift',num2str(y_step_curr),'.tiff']);
                        close;
                    end
                end
            else
                if (verbose)
                    disp(['          ... x_step: ',num2str(x_step_curr), ...
                        ', y_step: ',num2str(y_step_curr), ...
                        ', (x1,x2): (',num2str(x1),',',num2str(x2), ...
                        '), (y1,y2): (',num2str(y1),',',num2str(y2), ...
                        '), selection length (x,y): (',num2str(curr.x_length), ...
                        ',',num2str(curr.y_length), ...
                        '), diff_length (x,y): (',num2str(x_diff_length), ...
                        ',',num2str(y_diff_length),')']);
                end
            end
        end
    end
    t2 = clock;

    disp(['          ... Analysis time = ',num2str(etime(t2,t1))]);
    disp(['          ... Required shift (x,y): (',num2str(best_match.x_shift),',', ...
                                        num2str(best_match.y_shift),')']);
```

C.2 Code to process and identify columns in each slice image selection

```
clear all;
close all;
clc
warning off all


% Directory stuff
homework = 1;   % 1 = home,  2 = work


% Adjust contrast of each image to achieve a uniform density
densityTarget = 0.55;

% Filter images slightly to remove noise
filterCoeffs =         [      3        3        3        3        3       3     3];

% Break each image into eqSquares x eqSquares tiles and adjust the contrast of each
eqSquares =            [      2        3        3        2        3       2      3];
```

```
% Fill holes in connected domains
% Useful for the very large, well-defined column slices towards the top of the film
fillShapes =           [      1        1        1        0        0       0     0];

% Whether to consider the inner boundary of a connectedDomain
% The inner boundaries of extremely large connected domains will always be considered
% This is useful for the slices which are near to the substrate, where the resolution
% starts to get poor at columns appear very clumped together
doInnards =            [      0        0        0        0        0       1     1];


% Only break up connected domains that larger than the
%  large = meanArea * 1.3
%  medium = meanArea
%  small = meanArea * 0.75
outlierTypes =         {'large', 'small', 'medium', 'medium',  'medium', 'small', 'small'};

% The distance between the points of a pair along the edge of a connected domain
% must be less than minBridgeWidths to consider this point
% pair as the possible delimiter of a bridge
minBridgeWidths =      [     40       55       40       30       30      15     45];

% Point 1 of a given pair must be farther than minPairDistances
% pixels away from point 1 of another pair
minPairDistances =     [     20       20       20       20       20       5     20];

% If distance between points is tiny then consider this to be a bridge
maxRequiredDistances = [      5        5        5        5        5      10     10];

% The change in normal vector at a given point along the boundary of a connected domain
% must be larger than peakDeltas_primary and the change in the normal vector at the pair
% to this point must be larger than peakDeltas_secondary to be considered as a delimiters
% of a bridge (in pixels/boundary point)
peakDeltas_primary =   [     15       15       15       20       20      15     15];
peakDeltas_secondary = [     10       10       10       15       15      10     10];

% Inflections in the change in normal vector must be within deltaDistances pixels
% to be considered close to a given point in a pair
deltaDistances =       [     25       30       40       40       30      40     30];

% Inflections in the change in normal vector must be within deltaDistances pixels
% to be considered close to a given point in a pair
peakDistances =        [      5       15       15       15       15       5     5];


% Select only certain slices for processing
seriesSelection(1).selection = [3 4 5];
seriesSelection(2).selection = [1 10:10:90 93 ];
seriesSelection(3).selection = [1 10:10:130 133];
seriesSelection(4).selection = [1 10:10:70 71];
seriesSelection(5).selection = [1 10:10:60 64];
seriesSelection(6).selection = [225:10:305];
seriesSelection(7).selection = [1:1:25];


% Useful for stitching - to put all slice series
% To the same resolution
overrideImageRes = 0;
refSeries = 5;
resizeRate = [8.2/4 8.2/4 6.2/4 999 4/4 3/4 4 1];


%  Use a flat density target for image contrast equalization
flatTargetDensities = [densityTarget * ones(1,7)];

% For the slices at the very top of the film don't match the density
% These slices just show the tips of the columns
noMatchDensity(1).slice = [48:50];
noMatchDensity(2).slice = [1:10];
noMatchDensity(3).slice = [];
noMatchDensity(4).slice = [];
noMatchDensity(5).slice = [];
noMatchDensity(6).slice = [];
noMatchDensity(7).slice = [];


for series = [1 2 3 4 5 6 7]

    contNumSquares = eqSquares(series);
    outlierType = char(outlierTypes(series));
    minBridgeWidth = minBridgeWidths(series);
```

```
    peakDelta = [peakDeltas_primary(series) peakDeltas_secondary(series)];
    peakDistance = peakDistances(series);
    deltaDistance = deltaDistances(series);
    minPairDistance = minPairDistances(series);
    maxRequiredDistance = maxRequiredDistances(series);
    doInnard = doInnards(series);
    selection = seriesSelection(series).selection;
    filterCoeff = filterCoeffs(series);
    flatTargetDensity = flatTargetDensities(series);
    fillShape = fillShapes(series);


    dataDir = ['C:\My_Documents\Research\Research_Areas\ThreeDimGLAD_with_FIB\', ...
               'Processed_FIB_Scans\TopDownView\Processed_Scans_June09\Series', ...
               num2str(series),'\Select\'];
    outputDir = ['C:\My_Documents\Research\Research_Areas\ThreeDimGLAD_with_FIB\', ...
               'Processed_FIB_Scans\RegionProcessing\Shape_Scans_June09', ...
               '\Series',num2str(series),'\'];
    parameterFile = ['C:\My_Documents\Research\Research_Areas\ThreeDimGLAD_with_FIB\', ...
               'Matlab\Process_FIB_Slices_TopDownView\FIBscan_June09_TiO2VP5um.m'];


    processRegion
end


% ====================================================================

% processRegion


mkdir(outputDir);


filePrefix = 'SELECTUNFILTIMAGE_';
origPrefix = '1_ORIG_';
filterPrefix = '2_FILTER_';
eqPrefix = '3_EQ_';
thresholdPrefix = '4_THRESHOLD_';
filledPrefix = '4_FILLED_';
shapePrefix = '5_SHAPE_';
selectedPrefix = '6_SELECT_';
fileSuffix = '.tif';


% Constrast adjustment
doContrastAdj = 1;


% Sub shape processing
breakDownSubShapes = 0;
padding = 5;

% Shape plotting
doPlots = 1;

% Get scan parameters
disp('  Getting scan parameters');
run(parameterFile);

% Rescale images for stitching together and for easier
% processing by eye
if (overrideImageRes)
    scans.image_res_nm = ones(1,7) * scans.image_res_nm(refSeries);
else
    scans.image_res_nm  = scans.image_res_nm ./ resizeRate / fudgeFactor;
end

scans.x_size_pixel = scans.x_size_pixel .* resizeRate;
scans.y_size_pixel = scans.y_size_pixel .* resizeRate;


% Find all file names for the given series
fileNames = dir([dataDir,filePrefix,'*',fileSuffix]);

% Iterate through all slices in the series
for currFile = selection
    % Get current file name
    fileName = fileNames(currFile).name;
    slice = str2num(fileNames(currFile).name( ...
                    strfind(fileNames(1).name,'_')+1: ...
```

```
               strfind(fileNames(currFile).name,fileSuffix)-1));
disp(['Series ',num2str(series),', Slice ',num2str(slice)]);

% File name info
subDir = ['Series',num2str(series),'_Slice',num2str(slice)];
plotDir = [outputDir,'\',subDir]; mkdir(plotDir);
[leadingZeros] = findLeadingZeros(slice);

% Calculate the current thickness (height)
disp(['  Calculating current thickness ']);
currThickness(slice) = scans.cumThickness(series) - ...
                      (slice - scans.firstImage(series)) * ...
                          scans.slice_res_nm(series);
disp(['      Thickness = ',num2str(currThickness(slice))]);


% Read in the input image
disp('  Reading in image file ...');
origImage = double(imread([dataDir,fileName]));
origImage = uint8(squeeze(origImage(:,:,1)));
imageSize_col = size(origImage,2);
imageSize_row = size(origImage,1);


% Enbiggen image and interpolate
disp(['  Resizing image']);
resizedImage = imresize(origImage,resizeRate(series));
origImage = resizedImage;  clear resizedImage;

% Calculate image size
imageSize_col = size(origImage,2);
imageSize_row = size(origImage,1);


% Equalize the original image
disp(['  Equalization in ',num2str(contNumSquares),'x', ...
       num2str(contNumSquares),' squares']);
[eqImage] = adjContrast(origImage, contNumSquares, 9);


% Filter image
disp('  Filtering image ...');
% Play with filter coefficients!!
if (filterCoeff > 1)
    b = ones(1,(filterCoeff+1));
    a = (filterCoeff+1)*[1 zeros(1,filterCoeff)];
    % Need to filter in x AND y directions
    filtImage = filtfilt(b,a,double(eqImage));
    filtImage = uint8(filtfilt(b,a,filtImage'))';
else
    filtImage = eqImage;
end


% Determine target density
   targetDensity = flatTargetDensity;


% Determine multiplier factor that will reach target density
disp(['  Iterating to matching density target of ',num2str(targetDensity)]);
densityError = 0.002;
metTarget = 0;
maxIterations = 40;
currIteration = 1;
if (find(noMatchDensity(series).slice == slice))
    currMultiplier = scans.series(series).ic_contrastAdj(slice - ...
                                    scans.firstImage(series));
    densityError = 1;
    currIteration = maxIterations - 1;
else
    currMultiplier = 1;
end
multiplierStep = 0.01;
while ((~metTarget) && (currIteration < maxIterations))
    disp(['        ... iteration ',num2str(currIteration)]);

    % Threshold the equalized image

    level = graythresh(filtImage) * currMultiplier;
    disp(['          ... Thresholding with factor ', ...
            num2str(currMultiplier),' to level ',num2str(level)]);
```

```
    threshImage = im2bw(filtImage,level);

    % Fill holes in image
    disp([’          ... Filling holes of thresholded image’]);
    if (fillShape)
        filledImage = imfill(threshImage, ’holes’);
    else
        filledImage = threshImage;
    end

    % Find density of thresholded images
    disp([’          ... Finding density of thresholded images’]);
    numWhites = sum(sum(filledImage == 1));
    numBlacks = sum(sum(filledImage == 0));
    density(slice) = numWhites ./ (numWhites + numBlacks);
    disp([’          ... Density = ’,num2str(density(slice))]);

    densityDiff = targetDensity - density(slice);
    if (abs(densityDiff) < densityError)
        metTarget = 1;
    elseif densityDiff > 0
        % Density too low, multiplier should go down
        currMultiplier = currMultiplier - multiplierStep;
    elseif densityDiff < 0
        % Density too high, multiplier should go up
        currMultiplier = currMultiplier + multiplierStep;
    end
    currIteration = currIteration + 1;
end
disp([’  Final Density = ’,num2str(density(slice))]);



% Find edges of thresholded image
disp([’  Finding edges of thresholded image’]);
[edgeImage, edgeImage_x, edgeImage_y] = findEdgesImage(filledImage);




% Identify shapes in image
disp(’  Identifying shapes ...’);
[foundShapes, numShapes] = bwlabel(filledImage,4);


% Plot processed figures
doPlots = 1;
if (doPlots)
    disp(’  Plotting figures ...’);
    imwrite(uint8(origImage),[outputDir, origPrefix, leadingZeros,  ...
                              num2str(slice), fileSuffix],’tiff’);
    imwrite(uint8(eqImage),[outputDir, eqPrefix, leadingZeros, ...
                              num2str(slice), fileSuffix],’tiff’);
    imwrite(uint8(filtImage),[outputDir, filterPrefix, leadingZeros, ...
                              num2str(slice), fileSuffix],’tiff’);
    imwrite(threshImage,[outputDir, thresholdPrefix, leadingZeros, ...
                              num2str(slice), fileSuffix],’tiff’);
    imwrite(filledImage,[outputDir, filledPrefix, leadingZeros, ...
                                 num2str(slice), fileSuffix],’tiff’);
    imwrite(uint8(label2rgb(foundShapes, @jet, ’k’)),[outputDir, shapePrefix, ...
                                leadingZeros, num2str(slice), fileSuffix],’tiff’);
    close all;
end

clear origImage;


% Isolate each sub-shape within the image
disp(’  Isolating sub-shapes ...’);
[shapes] = findSubShapes(eqImage, foundShapes, numShapes, padding);


% Plot shapes (takes a long time...)
if 0
    disp(’  Plotting shapes ...’);
    num_subShapes = length(shapes);

    % Should probably frame each shape so that there isn’t so much
    % black space around it
    for s = 1:num_subShapes
```

```
        figure('Visible','off');
        imshow(uint8(shapes(s).unfilledShape*255));
        hold on;title([num2str(s)]);
        print('-dtiff',[plotDir, '\', shapePrefix, leadingZeros, ...
                        num2str(slice), '_o', num2str(s),fileSuffix]);
        close;

        figure('Visible','off');
        imshow(uint8(shapes(s).unfilledImage));
        hold on;title([num2str(s)]);
        print('-dtiff',[plotDir, '\', eqPrefix, leadingZeros, ...
                        num2str(slice), '_o', num2str(s),fileSuffix]);
        close;
    end
end


% (Don't do) Reject edge shapes;
% disp('  Rejecting edge shapes ...');
% [shapes] = rejectEdgeShapes(shapes, imageSize_row, imageSize_col);


% Break up big shapes
disp('  Breaking down big shapes into subshapes ...');
if ((currFile == 1) || (currFile == length(fileNames)))
    plotShapes = 1;
else
    plotShapes = 0;
end

shapes = subdivideBigShapes(shapes, slice, outlierType, minBridgeWidth, ...
                            minPairDistance, maxRequiredDistance, peakDelta, ...
                            peakDistance, deltaDistance, doInnard, plotShapes, ...
                            plotDir, shapePrefix, leadingZeros, fileSuffix);

% Reject small shapes
disp('  Rejecting small shapes ...');
areaLimit = 2;  % percent
[shapes, rejects] = rejectSmallShapes(shapes, areaLimit, 'percent');


% Plot shapes
if (doPlots)
    disp('  Plotting shapes ...');
    num_subShapes = length(shapes);

    % Should probably frame each shape so that there isn't so much
    % black space around it
    for s = 1:num_subShapes
        figure('Visible','off');
        imshow(uint8(shapes(s).unfilledShape*255));
        hold on;title([num2str(s)]);
        print('-dtiff',[plotDir, '\', shapePrefix, leadingZeros, ...
                        num2str(slice), '_c', num2str(s),fileSuffix]);
        close;

        figure('Visible','off');
        imshow(uint8(shapes(s).filledImage));
        hold on;title([num2str(s)]);
        print('-dtiff',[plotDir, '\', eqPrefix, leadingZeros, ...
                          num2str(slice), '_c', num2str(s),fileSuffix]);
        close;
    end
end



% Save shape areas to a file
fid = fopen([outputDir,'\','Areas_',leadingZeros,num2str(slice),'.txt'],'w');
fprintf(fid,'Series \n');
fprintf(fid,'%d \n',series);
fprintf(fid,'Slice \n');
fprintf(fid,'%d \n', slice);
fprintf(fid,'Image Resolution (nm/pixel) \n');
fprintf(fid,'%6.2f \n',scans.image_res_nm(series));
fprintf(fid,'Slice Resolution (nm/slice) \n');
fprintf(fid,'%6.2f \n',scans.slice_res_nm(series));
fprintf(fid,'Thickness \n');
fprintf(fid,'%6.1f \n', currThickness(slice));
fprintf(fid,'Density \n');
fprintf(fid,'%6.2f \n',density(slice));
```

```
    fprintf(fid,'Shape, Area (pixels), Mean Radius (pixels), Row Radius (pixels), ', ...
                'Col Radius (pixels), Area (nm2), Mean Radius (nm), Row Radius (nm), ', ...
                'Col Radius (nm), Row top, Row bottom, Col left, Col right \n');
    for s = 1:length(shapes)
        disp(['  ',num2str(s),', ',num2str(shapes(s).area)]);
        fprintf(fid,'%d, %d, %d, %d, %d, %6.2f, %6.2f, %6.2f, %6.2f, %d, %d, %d, %d \n', ...
                        s,shapes(s).area,shapes(s).mean_radius, ...
                        shapes(s).row_radius,shapes(s).col_radius, ...
                        shapes(s).area * (scans.image_res_nm(series)).^2, ...
                        shapes(s).mean_radius * scans.image_res_nm(series) , ...
                        shapes(s).row_radius * scans.image_res_nm(series) , ...
                        shapes(s).col_radius * scans.image_res_nm(series) , ...
                        shapes(s).row1, shapes(s).row2, shapes(s).col1, shapes(s).col2);
    end
    fclose(fid);

end

% Done
disp('Done.');



% ================================================================================

% Code to equalize the contrast across a single image

function [contImage stretchMask filtMask] = adjContrast(origImage, nSquares, filterCoeff)
    row = 1;
    col = 2;


    binStep = 1;
    binMin = 0;
    binMax = 255;
    bins = [binMin:binStep:binMax];


    % Divide the image up into n x n tiles
    col_size = size(origImage,2);
    row_size = size(origImage,1);

    col_pixels = floor(col_size/nSquares)*ones(1,nSquares);
    row_pixels = floor(row_size/nSquares)*ones(1,nSquares);

    col_pixels(end) = col_pixels(end) + (col_size - sum(col_pixels));
    row_pixels(end) = row_pixels(end) + (row_size - sum(row_pixels));

    col_start = [1 (1+cumsum(col_pixels(1:end-1)))];
    row_start = [1 (1+cumsum(row_pixels(1:end-1)))];

    col_end = cumsum(col_pixels);
    row_end = cumsum(row_pixels);

    % Locate effective end of histogram
    percentDown = 0.5;
    [h_orig] = hist(origImage, bins);
    sum_h_orig = sum(h_orig,2);
    [max_h_orig, peakPixel] = max(sum_h_orig);
    tmp = find(sum_h_orig(peakPixel:end) < max_h_orig * percentDown/100);
    maxPixel = tmp(1) + peakPixel - 1;


    % Also stretch image to maximize dynamic range
    adjImage = (255 / maxPixel) * double(origImage) + 1;
    b = ones(1,(filterCoeff+1));
    a = (filterCoeff+1)*[1 zeros(1,filterCoeff)];
    filtImage = filtfilt(b,a,double(adjImage));
    filtImage = uint8((filtfilt(b,a,filtImage'))');
    adjImage = filtImage;


    % Determine black to white threshold of each tile
    % Rqualize tile contrast to average threshold
    if (nSquares < 2)
        contImage = uint8(adjImage);
    else
        for row_square = 1:nSquares
            for col_square = 1:nSquares
                squareImage = adjImage(row_start(row_square):row_end(row_square), ...
                                       col_start(col_square):col_end(col_square));
```

```
            level(row_square, col_square) = graythresh(uint8(squareImage));
            threshold(row_square, col_square) = (level(row_square, col_square) * 255);
        end
    end

    % Determine contrast stretch required for that square
    mean_threshold = sum(sum(threshold))/(nSquares^2);

    for row_square = 1:nSquares
        for col_square = 1:nSquares
            squareImage = adjImage(row_start(row_square):row_end(row_square), ...
                                   col_start(col_square):col_end(col_square));

            [h] = hist(squareImage, bins);
            curr_histFreq = sum(h,2);

            % This effectively stretches the two peaks to roughly the
            % locations in the mean image which preserves the density
            stretch = mean_threshold / threshold(row_square, col_square);

            % The tile is brought to the mean contrast level
            transImage = stretch * double(squareImage);
            contImage(row_start(row_square):row_end(row_square), ...
                      col_start(col_square):col_end(col_square)) = transImage;
        end
    end

    % Calculate the multiplier that eac pixel had to change by in order to equalize all tiles
    stretchMask = double(contImage) ./ double(adjImage);
    stretchMask(find(isinf(stretchMask))) = stretchMask(find(isinf(stretchMask))-1);
    stretchMask(find(isnan(stretchMask))) = stretchMask(find(isnan(stretchMask))-1);


    % Need to filter the stretch mask in x AND y directions
    divideBy = 2;
    defFilter = filterCoeff; %11;
    if (nSquares < 2)
        filterCoeff = defFilter;
    else
        filterCoeff = max(defFilter, min(min(col_pixels),min(row_pixels))/divideBy);
    end

    b = ones(1,(filterCoeff+1));
    a = (filterCoeff+1)*[1 zeros(1,filterCoeff)];
    filtMask = filtfilt(b,a,stretchMask);
    filtMask = (filtfilt(b,a,filtMask'))';

    % Apply filtered contrast mask to the original image
    stretchImage = double(adjImage) .* filtMask;

    % Stretch contrast of image
    contImage = uint8(255 * double((stretchImage - min(min(stretchImage))) / ...
                      (max(max(stretchImage)) - min(min(stretchImage)))));

end

% ================================================================================

function [edgeImage, edgeImage_x, edgeImage_y] = findEdgesImage(thresholdImage)
    whiteToBlack = 1;
    blackToWhite = 2;


    % From First Row to Last

    edgeImage_y = zeros(size(thresholdImage(2:end,:)));
    tmp = thresholdImage(2:end,:) - thresholdImage(1:end-1,:);

    whiteToBlack_y.index = find(tmp == -1); %1);
    blackToWhite_y.index = find(tmp == 1);  %-1);

    [whiteToBlack_y.row, whiteToBlack_y.col] = ...
             ind2sub(size(thresholdImage),whiteToBlack_y.index);
    [blackToWhite_y.row, blackToWhite_y.col] = ...
             ind2sub(size(thresholdImage),blackToWhite_y.index);

    edgeImage_y(whiteToBlack_y.index) = whiteToBlack;
    edgeImage_y(blackToWhite_y.index) = blackToWhite;


    % From First Col to Last
```

```
    edgeImage_x = zeros(size(thresholdImage(:,2:end)));
    tmp = thresholdImage(:,2:end) - thresholdImage(:,1:end-1);

    whiteToBlack_x.index = find(tmp == -1);
    blackToWhite_x.index = find(tmp == 1);

    [whiteToBlack_x.row, whiteToBlack_x.col] = ...
              ind2sub(size(thresholdImage),whiteToBlack_x.index);
    [blackToWhite_x.row, blackToWhite_x.col] = ...
              ind2sub(size(thresholdImage),blackToWhite_x.index);

    edgeImage_x(whiteToBlack_x.index) = whiteToBlack;
    edgeImage_x(blackToWhite_x.index) = blackToWhite;

    edgeImage = edgeImage_x(1:end-1,:) + edgeImage_y(:,1:end-1);


% ============================================================================================

% Routine to go through all connected domains and individual columns (i.e. shapes)
% that were identified for the given image (using the bwlabel routine)
% and break them off into their own small subimage matrix


function [sub_shapes] = findSubShapes(origImage, shapeImage, numShapes, padding)

    for s = 1:numShapes
        disp(['    ... shape ',num2str(s)]);

        % Current shape
        unfilledShape = zeros(size(shapeImage));

        unfilledShape(find(shapeImage == s)) = 1;

        % Fill holes in shape
        filledShape = imfill(unfilledShape, 'holes');

        % Isolated shape in original image
        orig_unfilledShape = double(uint8(unfilledShape) .* origImage);
        orig_filledShape = double(uint8(filledShape) .* origImage);


        % Adjust contrast of isolated eq image
        maxPixel = max(max(orig_filledShape));
        minPixel = min(min(orig_filledShape));
        adj_orig_unfilledShape = (255 * (orig_unfilledShape - minPixel) / (maxPixel - minPixel));
        adj_orig_filledShape = (255 * (orig_filledShape - minPixel) / (maxPixel - minPixel));


        % Isolate region around shape
        [pixel_row, pixel_col] = find(unfilledShape);
        minRow_val = max(1, min(pixel_row) - padding);
        maxRow_val = min(size(unfilledShape,1), max(pixel_row) + padding);
        minCol_val = max(1, min(pixel_col) - padding);
        maxCol_val = min(size(unfilledShape,2),max(pixel_col) + padding);

        sub_shapes(s).unfilledImage = ...
              uint8(adj_orig_unfilledShape(minRow_val:maxRow_val,minCol_val:maxCol_val));
        sub_shapes(s).filledImage = ...
              uint8(adj_orig_filledShape(minRow_val:maxRow_val,minCol_val:maxCol_val));


        sub_shapes(s).filledShape = ...
              uint8(filledShape(minRow_val:maxRow_val,minCol_val:maxCol_val));
        sub_shapes(s).unfilledShape = ...
              uint8(unfilledShape(minRow_val:maxRow_val,minCol_val:maxCol_val));


        sub_shapes(s).row1 = min(pixel_row);
        sub_shapes(s).row2 = max(pixel_row);
        sub_shapes(s).col1 = min(pixel_col);
        sub_shapes(s).col2 = max(pixel_col);

        sub_shapes(s).area = sum(sum(sub_shapes(s).filledShape));

    end


% ============================================================================================
```

```
function [new_shapes] = subdivideBigShapes(shapes, slice, outlierType, minBridgeWidth, ...
                         minPairDistance, maxRequiredDistance, peakDelta, peakDistance, ...
                         deltaDistance, doInnard, doPlots, plotDir, shapePrefix, ...
                         leadingZeros, fileSuffix)

    row = 1;
    col = 2;


    new_shapes = shapes(1);
    num_shapes = length(shapes);


    % Calculating areas of shapes
        disp('    Calculating shape areas ...');

        for count = 1:num_shapes
            shapeAreas(count) = shapes(count).area;
        end

        meanArea = mean(shapeAreas);
        stdArea = std(shapeAreas);
        disp(['     ... mean Area = ',num2str(meanArea),', stddev = ',num2str(stdArea)]);

    % Only analyze big shapes
    % (The bigness is set in the main calling function - and is selected
    %  by eye)
        if (isequal(outlierType,'all'))
            areaMax = 1;
        elseif (isequal(outlierType,'small'))
            areaMax = meanArea * 0.75;
        elseif (isequal(outlierType,'medium'))
            areaMax = meanArea;
        elseif (isequal(outlierType,'large'))
            areaMax = meanArea* 1.3;
        end

        useShapes = find(shapeAreas < (areaMax));


    % Don't analyze very small shapes
        rejectArea = mean(shapeAreas(useShapes)) * 0.1;


    % Go through each shape and break it into sub shapes
        disp('    Breaking down big shapes ...');
        sub_shapeCounter = 0;
        for s = 1:num_shapes
            disp(['      ... Shape ',num2str(s)]);


            if (shapeAreas(s) > areaMax)
                % Only attempt to subdivide large shapes
                disp(['         ... Outlier']);

                % If a connected domain is extremely large then
                % look for opportunities to break it up based
                % on its inner boundaries
                if (shapeAreas(s) > meanArea* 1.5)
                    doInnard = 1;
                end

                % Call on routine to break up the connected domain by
                % searching for possible bridges between connected columns
                [brokenShape] = subdivide_BreakShapeBridge(shapes(s).unfilledShape, ...
                                  minBridgeWidth, minPairDistance, maxRequiredDistance, ...
                                  peakDelta, peakDistance, deltaDistance, rejectArea, ...
                                  doInnard, 0);

                % Once the connected domain has been broken up, label each
                % individual column
                [foundSubShapes, num_subShapes] = bwlabel(brokenShape,4);

                disp(['         ... Plotting sub shapes']);
                if (doPlots)
                    figure('Visible','off');
                    imshow(uint8(label2rgb(foundSubShapes, @jet, 'k')));
                    print('-dtiff',[plotDir, '\', shapePrefix, leadingZeros, ...
                             num2str(slice),'_break',num2str(s), fileSuffix]);
                    close;
                end
```

```
            else
                disp(['         ... Not outlier']);
                brokenShape = shapes(s).unfilledShape;
                [foundSubShapes, num_subShapes] = bwlabel(brokenShape,4);
            end



            for s2 = 1:num_subShapes

                curr_subShape = zeros(size(foundSubShapes));
                curr_subShape(find(foundSubShapes == s2)) = 1;


                if ((sum(sum(curr_subShape(1,:))) + sum(sum(curr_subShape(end,:))) + ...
                     sum(sum(curr_subShape(:,1))) + sum(sum(curr_subShape(:,end)))) > 1)
                    % This shape is along an edge
                    % Reject it

                else
                    disp(['         ... Saving Sub Shape ',num2str(s2)]);

                    sub_shapeCounter = sub_shapeCounter + 1;

                    new_shapes(sub_shapeCounter).unfilledShape = uint8(curr_subShape);
                    new_shapes(sub_shapeCounter).filledShape = uint8(imfill(curr_subShape));

                    new_shapes(sub_shapeCounter).unfilledImage = ...
                             uint8(double(shapes(s).filledImage) .* curr_subShape);
                    new_shapes(sub_shapeCounter).filledImage = ...
                             uint8(double(shapes(s).filledImage) .* imfill(curr_subShape));

                    % Need to recalculate rows and cols for subshapes
                    [pixel_row, pixel_col] = find(curr_subShape);
                    minRow_val = min(pixel_row);
                    maxRow_val = max(pixel_row);
                    minCol_val = min(pixel_col);
                    maxCol_val = max(pixel_col);

                    new_shapes(sub_shapeCounter).row1 = shapes(s).row1 + minRow_val - 1;
                    new_shapes(sub_shapeCounter).row2 = shapes(s).row1 + maxRow_val - 1;
                    new_shapes(sub_shapeCounter).col1 = shapes(s).col1 + minCol_val - 1;
                    new_shapes(sub_shapeCounter).col2 = shapes(s).col1 + maxCol_val - 1;

                    new_shapes(sub_shapeCounter).area = sum(sum(imfill(curr_subShape)));
                    new_shapes(sub_shapeCounter).mean_radius = ...
                          round(sqrt(new_shapes(sub_shapeCounter).area / pi));
                    new_shapes(sub_shapeCounter).row_radius = ...
                          round(((maxRow_val - minRow_val))/2);
                    new_shapes(sub_shapeCounter).col_radius = ...
                          round(((maxCol_val - minCol_val))/2);
                end
            end

            close all

        end


% ============================================================================================

function [brokenShape] = subdivide_BreakShapeBridge(shape, minBridgeWidth, ...
                                minPairDistance, maxRequiredDistance, peakDelta, ...
                                peakDistance, deltaDistance, rejectArea, ...
                                doInnard, doPlots)

    row = 1;
    col = 2;

    offset = 10;
    flexibilityFactor = 0.4;   % Curve in second part of shape bridges can be
                               % a little less than the target


    % Don't process every point along the boundary of the connected domain
    % Just look at every fourth point
    decFactor = 4;


    % Default - pass back the same shape that was passed in
```

```
    brokenShape = shape;


    % Edge detection
    % Determine all boundaries (inner and outer) of a connected domain
    tmp = bwboundaries(shape);
    numBoundaries = length(tmp);


    % Travese boundaries looking for close points
    brokenShape = shape;
    for count = 1:numBoundaries
        currBoundary = tmp{count};

        % Determine direction of vector between adjacent points
        % From first point to last
        boundary(1).full_loc = currBoundary;
        boundary(1).loc = currBoundary(1:decFactor:end,:);

        % From last point to first
        boundary(2).full_loc = currBoundary(end:-1:1,:);
        boundary(2).loc = currBoundary(end:-decFactor:1,:);

        boundary(1).numBoundaryPoints = size(boundary(1).loc,1);
        boundary(2).numBoundaryPoints = size(boundary(2).loc,1);

        % A boundary must be sufficiently large to use it
        minBoundaryPoints = 10;


        if ((boundary(1).numBoundaryPoints < minBoundaryPoints) || ...
               (boundary(2).numBoundaryPoints < minBoundaryPoints))
            disp(['              ... Boundary ',num2str(count),' too small']);
        else
            [ sub_brokenShape ] = findShapeBridge(shape, boundary, ...
                                                  round(offset/decFactor), ...
                                                  minPairDistance, minBridgeWidth, ...
                                                  maxRequiredDistance, peakDelta, ...
                                                  peakDistance, deltaDistance, ...
                                                  rejectArea, doPlots);

            brokenShape = brokenShape & sub_brokenShape;
        end

        % Explore inner boundaries too
        if (doInnard)
            for count2 = (count+1):numBoundaries
                disp(['            ... Boundary ',num2str(count),' and ',num2str(count2)]);
                secondBoundary = tmp{count2};
                boundary(2).loc = secondBoundary(1:decFactor:end,:);
                boundary(2).full_loc = secondBoundary;
                boundary(2).numBoundaryPoints = size(boundary(2).loc,1);

                % Only look at boundary if it is large enough
                if (boundary(1).numBoundaryPoints < minBoundaryPoints) || ...
                      (boundary(2).numBoundaryPoints < minBoundaryPoints)
                    disp(['              ... Boundary ',num2str(count), ...
                             ' or ',num2str(count2),' too small']);
                else
                    [ sub_brokenShape ] = findShapeBridge(shape, boundary,  0,  ...
                                                minPairDistance, minBridgeWidth, ...
                                                maxRequiredDistance, peakDelta, ...
                                                peakDistance, deltaDistance, ...
                                                rejectArea, doPlots);

                    brokenShape = brokenShape & sub_brokenShape;
                end
            end
        end
    end


% ============================================================================================

% The main routine used to break up a connected domain (i.e. shape)
% into individual columns
function [ brokenShape ] = findShapeBridge(shape, boundary, offset, ...
                                   minPairDistance, minBridgeWidth, ...
                                   maxRequiredDistance, peakDelta, ...
                                   peakDistance, deltaDistance, ...
                                   rejectArea, doPlots)
```

```
row = 1;
col = 2;

peakDelta_primary = peakDelta(1);
peakDelta_secondary = peakDelta(2);

% Default - pass back the same shape that was passed in
brokenShape = shape;

% Matrix of boundary vectors
    boundary(1).rowMatrix = repmat(boundary(1).loc(1:(end-offset),row), ...
                                   [1, boundary(2).numBoundaryPoints-offset]);
    boundary(1).colMatrix = repmat(boundary(1).loc(1:(end-offset),col), ...
                                   [1, boundary(2).numBoundaryPoints-offset]);
    boundary(2).rowMatrix = (repmat(boundary(2).loc(1:(end-offset),row), ...
                                   [1, boundary(1).numBoundaryPoints-offset]))';
    boundary(2).colMatrix = (repmat(boundary(2).loc(1:(end-offset),col), ...
                                   [1, boundary(1).numBoundaryPoints-offset]))';

for d = 1:2
    if (d == 1)
        flip = 1;
    else
        flip = 0;
    end

    % Smooth boundary slightly so that not every single noisy dip is
    % interpreted as a possible bridge
    filterCoeff = 3;

    % Find normal vectors to every point along each of the two boundaries
    % Fast changes in the direction of the normal vector indicate a possible
    % Concavity along the boundary which could be one half of a shape bridge
    [normvect, d_normvect] = findNormalVector(boundary(d).loc, 1, filterCoeff, flip);
    boundary(d).normvect = normvect;
    boundary(d).d_normvect = d_normvect;
end


% Distance between points along the two boundaries
pointDist = sqrt((boundary(1).rowMatrix - boundary(2).rowMatrix).^2 + ...
                 (boundary(1).colMatrix - boundary(2).colMatrix).^2);


% Find local minima from distances between points
[zmax,imax,zmin,imin] = extrema2(pointDist);

% Ignore local minima that are tiny (noise)
%imin = imin(find(zmin > 1));
%zmin = zmin(find(zmin > 1));

% Ignore local minima that are too big
imin = imin(find(zmin < minBridgeWidth));
zmin = zmin(find(zmin < minBridgeWidth));


% From these local minima, identify boundary point pairs
% that could be bridges between shapes (i.e. columns)
% We'll want to break these shape bridges
[boundary(1).pairLoc,boundary(2).pairLoc] = ind2sub(size(pointDist),imin);
boundary(1).pairCol = boundary(1).loc(boundary(1).pairLoc,col);
boundary(1).pairRow = boundary(1).loc(boundary(1).pairLoc,row);
boundary(2).pairCol = boundary(2).loc(boundary(2).pairLoc,col);
boundary(2).pairRow = boundary(2).loc(boundary(2).pairLoc,row);
boundary(1).distance = ((boundary(2).pairRow - ...
                         boundary(1).pairRow).^2 + ...
                        (boundary(2).pairCol - ...
                         boundary(1).pairCol).^2).^(1/2);
boundary(1).valid = ones(size(boundary(1).pairLoc));
boundary(2).valid = ones(size(boundary(2).pairLoc));

% Plot shape, boundary, normal vectors, and bridge points
if (doPlots)
    plotShapeBridges(shape, boundary(1).loc, boundary(2).loc, ...
                     boundary(1).normvect, boundary(2).normvect, ...
                     boundary(1).valid, boundary(2).valid, boundary(1).pairCol, ...
                     boundary(1).pairRow, boundary(2).pairCol, boundary(2).pairRow);
end
```

```
% Need to draw a line between the two points
% If there is any black space then reject these points
for p = 1:length(boundary(1).pairCol)

    rowLength = abs(boundary(2).pairRow(p) - boundary(1).pairRow(p))+1;
    colLength = abs(boundary(2).pairCol(p) - boundary(1).pairCol(p))+1;
    pairLineRow = round(linspace(boundary(1).pairRow(p), ...
                        boundary(2).pairRow(p),max(rowLength,colLength)));
    pairLineCol = round(linspace(boundary(1).pairCol(p), ...
                        boundary(2).pairCol(p),max(rowLength,colLength)));
    pixelsLine = shape(sub2ind(size(shape),pairLineRow, pairLineCol));

    % Find black holes between the start and end of the pair points
    % (black at either end is acceptable)
    holes = find(pixelsLine(2:(end-1)) == 0);

    if (~isempty(diff(holes)))
        % There are black pixels between the two points
        % Reject
        boundary(1).valid(p) = 0;
        boundary(2).valid(p) = 0;
    elseif (~isempty(holes))
        if (sum(holes) == length(holes))
            % Reject
            boundary(1).valid(p) = 0;
            boundary(2).valid(p) = 0;
        else
            % Allow
        end
    else
        % Allow
    end
end

if (doPlots)
    plotShapeBridges(shape, boundary(1).loc, boundary(2).loc, ...
                        boundary(1).normvect, boundary(2).normvect, ...
                        boundary(1).valid, boundary(2).valid, ...
                        boundary(1).pairCol, boundary(1).pairRow, ...
                        boundary(2).pairCol,boundary(2).pairRow);
end


% Determine which points are right near each other
% Reject any point pairs that are actually neighbors
validPoints = find(boundary(1).valid);
closeDistance = 3;
for p = 1:length(boundary(1).pairRow(validPoints))

    tmp = find(boundary(1).full_loc(:,col) == boundary(1).pairCol(validPoints(p)));
    tmp2 = find(boundary(1).full_loc(:,row) == boundary(1).pairRow(validPoints(p)));

    pointLoc = intersect(tmp, tmp2);

    % Decrement of boundary points may make this difficult
    tmp = find(boundary(1).full_loc(:,col) == boundary(2).pairCol(validPoints(p)));
    tmp2 = find(boundary(1).full_loc(:,row) == boundary(2).pairRow(validPoints(p)));


    pairLoc = intersect(tmp, tmp2);

    if ((isempty(pairLoc)) || (isempty(pointLoc)))
        % It is possible that this could happen if the two boundaries
        % are different (not the same boundary in reverse)
        error('Cannot find pair location.');
    else
        if (abs(pointLoc - pairLoc) <= closeDistance)
            boundary(1).valid(validPoints(p)) = 0;
            boundary(2).valid(validPoints(p)) = 0;
        end
    end
end


if (doPlots)
    plotShapeBridges(shape, boundary(1).loc, boundary(2).loc, ...
                        boundary(1).normvect, boundary(2).normvect, ...
                        boundary(1).valid, boundary(2).valid, ...
                        boundary(1).pairCol, boundary(1).pairRow, ...
                        boundary(2).pairCol,boundary(2).pairRow);
```

```
end


% Filter d_normvect
% To remove noise
% We will use d_normvect to see about inflection points in the direction
% vectors around the boundaries
% If there is a big inflection then this might be a bridge
filterCoeff = 3;
b = ones(1,(filterCoeff+1));
a = (filterCoeff+1)*[1 zeros(1,filterCoeff)];
boundary(1).filt_d_normvect = filtfilt(b,a,boundary(1).d_normvect);
boundary(2).filt_d_normvect = filtfilt(b,a,boundary(2).d_normvect);



% Need to check whether minima corresponds to a significant
% inflection point on at least one side of the shape minima
% Find local minima and maxima in the derivative of the normal vector
[dmax,lmax,dmin,lmin] = extrema(boundary(1).filt_d_normvect);


% Neglect max/minima at ends of data
tmin = intersect(find(lmin > 1), ...
                 find(lmin < length(boundary(1).filt_d_normvect)));
tmax = intersect(find(lmax > 1), ...
                 find(lmax < length(boundary(1).filt_d_normvect)));
lmin = lmin(tmin);
lmax = lmax(tmax);
dmin = dmin(tmin);
dmax = dmax(tmax);

[lmin i] = sort(lmin);
dmin = dmin(i);
[lmax i] = sort(lmax);
dmax = dmax(i);

boundary(1).d_norm_minloc = lmin;
boundary(1).d_norm_maxloc = lmax;
boundary(1).d_norm_minval = dmin;
boundary(1).d_norm_maxval = dmax;


% Boundary 2
[dmax,lmax,dmin,lmin] = extrema(boundary(2).filt_d_normvect);

% Neglect max/minima at ends of data
tmin = intersect(find(lmin > 1), find(lmin < length(boundary(1).filt_d_normvect)));
tmax = intersect(find(lmax > 1), find(lmax < length(boundary(1).filt_d_normvect)));
lmin = lmin(tmin);
lmax = lmax(tmax);
dmin = dmin(tmin);
dmax = dmax(tmax);

[lmin i] = sort(lmin);
dmin = dmin(i);
[lmax i] = sort(lmax);
dmax = dmax(i);

boundary(2).d_norm_minloc = lmin;
boundary(2).d_norm_maxloc = lmax;
boundary(2).d_norm_minval = dmin;
boundary(2).d_norm_maxval = dmax;

if (doPlots)
    figure;subplot(2,1,1);
           plot(boundary(1).filt_d_normvect); hold on;
           plot(boundary(1).d_norm_maxloc, boundary(1).d_norm_maxval,'ro');
           plot(boundary(1).d_norm_minloc, boundary(1).d_norm_minval,'gs');
           title('boundary 1');
           subplot(2,1,2);
           plot(boundary(2).filt_d_normvect); hold on;
           plot(boundary(2).d_norm_minloc, boundary(2).d_norm_minval,'ro');
           plot(boundary(2).d_norm_maxloc, boundary(2).d_norm_maxval,'gs');
           title('boundary 2');
end


validPoints = find(boundary(1).valid);

for p = 1:length(boundary(1).pairRow(validPoints))
```

```
% Conditions:
%  Inflection of boundary point 1 must be positive
%  Inflection of boundary point 2 must be negative
if ((boundary(1).d_normvect(boundary(1).pairLoc(validPoints(p))) > 0) && ...
    (boundary(2).d_normvect(boundary(2).pairLoc(validPoints(p))) < 0))


    % Look for maxima in the normal vector change (d_norm)
    % Is the nearest maxima for boundary1 point
    % very close to the given point (i.e. we have a bit of flexibility)?
    % If so, then this boundary point might be part of a bridge
    tmp = find(abs(boundary(1).d_norm_maxloc - boundary(1).pairLoc(validPoints(p))) < peakDistance);
    if (isempty(tmp))
        found_max_peak = 0;
    else
        found_max_peak = 1;
    end

    % Look for minima in the normal vector change for boundary 2
    % Is the nearest minima for the boundary2 point close to the given point?
    % If so, then this boundary point might be part of a bridge
    tmp = find(abs(boundary(2).d_norm_minloc - boundary(2).pairLoc(validPoints(p))) < peakDistance);
    if (isempty(tmp))
        found_min_peak = 0;
    else
        found_min_peak = 1;
    end

    % Where is the nearest minima to the left for this boundary 1 point
    neg_minval = find(boundary(1).d_norm_minval < 0);
    tmp = find(boundary(1).d_norm_minloc(neg_minval) < boundary(1).pairLoc(validPoints(p)));
    if (isempty(tmp))
        minloc_left = -99;
    else
        minloc_left = boundary(1).d_norm_minloc(neg_minval(tmp(end)));
    end

    % Where is the nearest minima to the right for this boundary 1 point
    tmp = find(boundary(1).d_norm_minloc(neg_minval) > boundary(1).pairLoc(validPoints(p)));
    if (isempty(tmp))
        minloc_right = -99;
    else
        minloc_right = boundary(1).d_norm_minloc(neg_minval(tmp(1)));
    end

    % How deep of an inflection is there for the left minima?
    % Is it large?
    if (minloc_left == -99)
        delta_left = 0;
    elseif (abs(boundary(1).pairLoc(validPoints(p)) - minloc_left) <= deltaDistance)
        delta_left = boundary(1).d_normvect(boundary(1).pairLoc(validPoints(p))) - ...
                                            boundary(1).d_normvect(minloc_left);
    else
        delta_left = 0;
    end

    % How deep of an inflection is there for the right minima?
    if (minloc_right == -99)
        delta_right = 0;
    elseif (abs(boundary(1).pairLoc(validPoints(p)) - minloc_right) <= deltaDistance)
        delta_right = boundary(1).d_normvect(boundary(1).pairLoc(validPoints(p))) - ...
                                             boundary(1).d_normvect(minloc_right);
    else
        delta_right = 0;
    end
    delta_max_1 = max(delta_left, delta_right);

    % For boundary point 2...
    % Nearest maximas for boundary2 point
    pos_maxval = find(boundary(2).d_norm_maxval > 0);
    tmp = find(boundary(2).d_norm_maxloc(pos_maxval) < boundary(2).pairLoc(validPoints(p)));
    if (isempty(tmp))
        maxloc_left = -99;
    else
        maxloc_left = boundary(2).d_norm_maxloc(pos_maxval(tmp(end)));
    end
    tmp = find(boundary(2).d_norm_maxloc(pos_maxval) > boundary(2).pairLoc(validPoints(p)));
    if (isempty(tmp))
        maxloc_right = -99;
    else
```

```
            maxloc_right = boundary(2).d_norm_maxloc(pos_maxval(tmp(1)));
        end

        % Determine how deep the inflection is for boundary point 2
        if (maxloc_left == -99)
            delta_left = 0;
        elseif (abs(boundary(2).pairLoc(validPoints(p)) - maxloc_left) <= deltaDistance)
            delta_left = boundary(2).d_normvect(  boundary(2).pairLoc(validPoints(p))) - ...
                                                  boundary(2).d_normvect(maxloc_left);
        else
            delta_left = 0;
        end
        if (maxloc_right == -99)
            delta_right = 0;
        elseif (abs(boundary(2).pairLoc(validPoints(p)) - maxloc_right) <= deltaDistance)
            delta_right = boundary(2).d_normvect(boundary(2).pairLoc(validPoints(p))) - ...
                                                 boundary(2).d_normvect(maxloc_right);
        else
            delta_right = 0;
        end
        delta_max_2 = min(delta_left, delta_right);


        % Now look at the inflections for boundary points 1 and 2.
        % Are they large enough
        % One must be larger than peakDelta_primary and the other
        % must be larger than peakDelta_secondary
        % (i.e. the bridge must have at least one fairly large concavity)
        % If the inflection at one point pair is large and
        if ((((delta_max_1 > peakDelta_primary) && (delta_max_2 < -peakDelta_secondary)) || ...
            ((delta_max_1 > peakDelta_secondary) && (delta_max_2 < -peakDelta_primary))) && ...
            found_max_peak  && found_min_peak )
            % Accept

        elseif (boundary(1).distance(validPoints(p)) <= maxRequiredDistance)
            % Allow if the distance between point pairs is tiny
            % (i.e. even if the inflections aren't strong, this could be a bridge
            %  if the distance across is very small)

        else
            % Reject
            boundary(1).valid(validPoints(p)) = 0;
            boundary(2).valid(validPoints(p)) = 0;
        end

    elseif (boundary(1).distance(validPoints(p)) <= maxRequiredDistance)
        % Allow if the distance between point pairs is tiny

    else
        % Reject
        boundary(1).valid(validPoints(p)) = 0;
        boundary(2).valid(validPoints(p)) = 0;
    end
end


if (doPlots)
    plotShapeBridges(shape, boundary(1).loc, boundary(2).loc, ...
                     boundary(1).normvect, boundary(2).normvect, ...
                     boundary(1).valid, boundary(2).valid, ...
                     boundary(1).pairCol, boundary(1).pairRow, ...
                     boundary(2).pairCol,boundary(2).pairRow);
end



% Need to reject pairs that are similar
% Search each distance individually first
validPoints = find(boundary(1).valid);

pair_rowMatrix = repmat(boundary(1).pairRow(validPoints), 1, length(boundary(1).pairCol(validPoints)));
pair_colMatrix = repmat(boundary(1).pairCol(validPoints), 1, length(boundary(1).pairRow(validPoints)));
pair_dist_1 = sqrt((pair_rowMatrix - pair_rowMatrix').^2 + (pair_colMatrix - pair_colMatrix').^2);
pair_rowMatrix = repmat(boundary(2).pairRow(validPoints), 1, length(boundary(2).pairCol(validPoints)));
pair_colMatrix = repmat(boundary(2).pairCol(validPoints), 1, length(boundary(2).pairRow(validPoints)));
pair_dist_2 = sqrt((pair_rowMatrix - pair_rowMatrix').^2 + (pair_colMatrix - pair_colMatrix').^2);


for p = 1:length(boundary(d).pairRow(validPoints))
     % Find pairs that are far apart
     far_pairs_1 = find(pair_dist_1(p,:) >= minPairDistance);
```

```
     far_pairs_2 = find(pair_dist_2(p,:) >= minPairDistance);
     % Find pairs that are close together
     close_pairs_1 = find(pair_dist_1(p,:) < minPairDistance);
     close_pairs_2 = find(pair_dist_2(p,:) < minPairDistance);

     pairMatch = intersect(close_pairs_1, close_pairs_2);

     % Which has lowest distance?
     [min_zmin, min_loc] = min(boundary(1).distance(validPoints(pairMatch)));
     % Keep pair with lowest zmin
     keep_pair = pairMatch(min_loc);
     % Remove pairs which were close to this pair
     remove_pairs = pairMatch(find(pairMatch ~= keep_pair));

     % Remove these pairs from rest of grid
     pair_dist_1(remove_pairs,:) = 55555;
     pair_dist_2(remove_pairs,:) = 55555;

     pair_dist_1(:,remove_pairs) = 55555;
     pair_dist_2(:,remove_pairs) = 55555;

end

% Which pairs to reject
pair_ind = find(pair_dist_1 >= 55555);

[good_pair_i, good_pair_j] = ind2sub(size(pair_dist_1),pair_ind);

boundary(1).valid(validPoints(good_pair_i(find(good_pair_j == 1)))) = 0;
boundary(2).valid(validPoints(good_pair_i(find(good_pair_j == 1)))) = 0;


if (doPlots)
    plotShapeBridges(shape, boundary(1).loc, boundary(2).loc, ...
                        boundary(1).normvect, boundary(2).normvect, ...
                        boundary(1).valid, boundary(2).valid, ...
                        boundary(1).pairCol, boundary(1).pairRow, ...
                        boundary(2).pairCol,boundary(2).pairRow);
end


% Reject pairs that are the same as those found in the opposite direction
validPoints = find(boundary(1).valid);

pair_rowMatrix_1 = repmat(boundary(1).pairRow(validPoints), 1, ...
                            length(boundary(1).pairCol(validPoints)));
pair_colMatrix_1 = repmat(boundary(1).pairCol(validPoints), 1, ...
                            length(boundary(1).pairRow(validPoints)));
pair_rowMatrix_2 = repmat(boundary(2).pairRow(validPoints), 1, ...
                            length(boundary(2).pairCol(validPoints)));
pair_colMatrix_2 = repmat(boundary(2).pairCol(validPoints), 1, ...
                            length(boundary(2).pairRow(validPoints)));
pair_dist_1 = sqrt((pair_rowMatrix_1 - pair_rowMatrix_2').^2 + ...
                    (pair_colMatrix_1 - pair_colMatrix_2').^2);
pair_dist_2 = sqrt((pair_rowMatrix_2 - pair_rowMatrix_1').^2 + ...
                    (pair_colMatrix_2 - pair_colMatrix_1').^2);


for p = 1:length(boundary(d).pairRow(validPoints))
     % Find pairs that are far apart
     far_pairs_1 = find(pair_dist_1(p,:) >= minPairDistance);
     far_pairs_2 = find(pair_dist_2(p,:) >= minPairDistance);
     % Find pairs that are close together
     close_pairs_1 = find(pair_dist_1(p,:) < minPairDistance);
     close_pairs_2 = find(pair_dist_2(p,:) < minPairDistance);

     pairMatch = sort([p intersect(close_pairs_1, close_pairs_2)]);


     % Which has lowest distance?
     [min_zmin, min_loc] = min(boundary(1).distance(validPoints(pairMatch)));
     % Keep pair with lowest zmin
     keep_pair = pairMatch(min_loc);
     % Remove pairs which were close to this pair
     remove_pairs = pairMatch(find(pairMatch ~= keep_pair));

     % Remove these pairs from rest of grid
     pair_dist_1(remove_pairs,:) = 55555;
     pair_dist_2(remove_pairs,:) = 55555;

     pair_dist_1(:,remove_pairs) = 55555;
```

```
     pair_dist_2(:,remove_pairs) = 55555;

end

% Which pairs to reject
pair_ind = find(pair_dist_1 >= 55555);

[good_pair_i, good_pair_j] = ind2sub(size(pair_dist_1),pair_ind);

boundary(1).valid(validPoints(good_pair_i(find(good_pair_j == 1)))) = 0;
boundary(2).valid(validPoints(good_pair_i(find(good_pair_j == 1)))) = 0;




if (doPlots)
    plotShapeBridges(shape, boundary(1).loc, boundary(2).loc, ...
                        boundary(1).normvect, boundary(2).normvect, ...
                        boundary(1).valid, boundary(2).valid, boundary(1).pairCol, ...
                        boundary(1).pairRow, boundary(2).pairCol,boundary(2).pairRow);
end


% Need to check whether the line between a given point pair crosses
% the line between a smaller point pair
% Reject the point pair with the longer path distance

% Draw a black line between each pair on the shape
validPoints = find(boundary(d).valid);
brokenShape = shape;
for p = 1:length(boundary(1).pairRow(validPoints))

    rowLength = abs(boundary(2).pairRow(validPoints(p)) - ...
                      boundary(1).pairRow(validPoints(p))) + 1;
    colLength = abs(boundary(2).pairCol(validPoints(p)) - ...
                      boundary(1).pairCol(validPoints(p))) + 1;


    pairLineRow = round(linspace(boundary(1).pairRow(validPoints(p)), ...
                                 boundary(2).pairRow(validPoints(p)), ...
                                 max(rowLength,colLength)));
    pairLineCol = round(linspace(boundary(1).pairCol(validPoints(p)), ...
                                 boundary(2).pairCol(validPoints(p)), ...
                                 max(rowLength,colLength)));

    brokenShape(sub2ind(size(shape),pairLineRow, pairLineCol)) = zeros(1,length(pairLineRow));

end


% Are any of the subshapes tiny?
[foundSubShapes, num_subShapes] = bwlabel(brokenShape,4);

for s = 1:num_subShapes
    subShape(s).shape = size(foundSubShapes);
    subShape(s).shape(find(foundSubShapes == s)) = 1;
    subShape(s).area = sum(sum(foundSubShapes == s));
end

[new_shapes, reject] = rejectSmallShapes(subShape, rejectArea, 'abs');


% Remove any pairs that result in a tiny shape
validPoints = find(boundary(d).valid);
for s = 1:num_subShapes
    if (reject(s))
        for p = 1:length(boundary(1).pairRow(validPoints))
            tmp = bwboundaries(subShape(s).shape);
            boundaryShape = tmp{1,1};

            tmp = find(boundaryShape(:,row) >= ...
                        (boundary(1).pairRow(validPoints(p)) - 1));
            tmp2 = find(boundaryShape(:,row) <= ...
                        (boundary(1).pairRow(validPoints(p)) + 1));
            tmp3 = find(boundaryShape(:,col) >= ...
                        (boundary(1).pairCol(validPoints(p)) - 1));
            tmp4 = find(boundaryShape(:,col) <= ...
                        (boundary(1).pairCol(validPoints(p)) + 1));

            boundary1ok = intersect(intersect(tmp,tmp2), intersect(tmp3,tmp4));
```

```
                tmp = find(boundaryShape(:,row) >= ...
                            (boundary(2).pairRow(validPoints(p)) - 1));
                tmp2 = find(boundaryShape(:,row) <= ...
                            (boundary(2).pairRow(validPoints(p)) + 1));
                tmp3 = find(boundaryShape(:,col) >= ...
                            (boundary(2).pairCol(validPoints(p)) - 1));
                tmp4 = find(boundaryShape(:,col) <= ...
                            (boundary(2).pairCol(validPoints(p)) + 1));

                boundary2ok = intersect(intersect(tmp,tmp2), intersect(tmp3,tmp4));

                if ((~isempty(boundary1ok)) && (~isempty(boundary2ok)))

                    % Remove this pair
                    boundary(1).valid(validPoints(p)) = 0;
                    boundary(2).valid(validPoints(p)) = 0;

                end
            end
        end

    end
    clear subShape;

    if (sum(reject) >= 1)
        % Redraw brokenShape with small areas removed
        validPoints = find(boundary(d).valid);
        brokenShape = shape;
        for p = 1:length(boundary(1).pairRow(validPoints))

            rowLength = abs(boundary(2).pairRow(validPoints(p)) - ...
                            boundary(1).pairRow(validPoints(p))) + 1;
            colLength = abs(boundary(2).pairCol(validPoints(p)) - ...
                            boundary(1).pairCol(validPoints(p))) + 1;


            pairLineRow = round(linspace(boundary(1).pairRow(validPoints(p)), ...
                          boundary(2).pairRow(validPoints(p)),max(rowLength,colLength)));
            pairLineCol = round(linspace(boundary(1).pairCol(validPoints(p)), ...
                          boundary(2).pairCol(validPoints(p)),max(rowLength,colLength)));

            brokenShape(sub2ind(size(shape),pairLineRow, pairLineCol)) = zeros(1,length(pairLineRow));
        end

    end

    close all;

% ================================================================================================

function [ normvect, d_normvect ] = findNormalVector(boundary, ...
                                    boundaryFilterCoeff, vectorFilterCoeff, flip)
    row = 1;
    col = 2;

    % Determine distance between adjacent points (x2 - x1)
        dist = diff(boundary);

    % Filter difference
        if (boundaryFilterCoeff == 1)
            filtdist = dist;
        else
            b = ones(1,(boundaryFilterCoeff+1));
            a = (boundaryFilterCoeff+1)*[1 zeros(1,boundaryFilterCoeff)];
            filtdist = filtfilt(b,a,dist);
        end

    % Determine direction between adjacent points
        vect = atan(abs(filtdist(:,col))./abs(filtdist(:,row))) * 180 / pi;

        signRow = sign(dist(:,row));
        signCol = sign(dist(:,col));

    % Determine correct quadrant for direction
        vect((signRow ==  1) & (signCol == -1)) = 360 - vect((signRow ==  1) & (signCol == -1));
        vect((signRow == -1) & (signCol == -1)) = 180 + vect((signRow == -1) & (signCol == -1));
        vect((signRow == -1) & (signCol ==  1)) = 180 - vect((signRow == -1) & (signCol ==  1));
        vect((signRow ==  0) & (signCol ==  1)) =  90;
        vect((signRow == -1) & (signCol ==  0)) = 180;
        vect((signRow ==  0) & (signCol == -1)) = -90;
```

```
        tmp = find(isnan(vect));
        if (~isempty(tmp))
            if (tmp(1) == 1)
                findNotNan = 1;
                nextCounter = 1;
                while findNotNan
                    if (~isnan(vect(tmp(1) + nextCounter)))
                        vect(tmp) = vect(tmp + nextCounter);
                        findNotNan = 0;
                    else
                        nextCounter = nextCounter + 1;
                    end
                end
            else
                vect(tmp) = vect(tmp-1);
            end
        end

        if (vect > 360)
            vect = vect - 360;
        end



    % Find normal vector
        normvect = vect + 90;

        if (flip)
            normvect = normvect + 180;
        end

    % Unwrap (smooth out) normvect angle jumps
        normvect = unwrap(normvect/180*pi)*180/pi;

    % Filter direction vector
        if ((vectorFilterCoeff == 1) || (vectorFilterCoeff * 4 > length(normvect)))
        else
            b = ones(1,(vectorFilterCoeff+1));
            a = (vectorFilterCoeff+1)*[1 zeros(1,vectorFilterCoeff)];
            normvect = filtfilt(b,a,normvect);
        end

    % Add last point to normvect to make the vector length the same as that
    % of boundary
        normvect = [normvect;normvect(end)];

    % Find derivative of normvect
        d_normvect = normvect(2:end) - normvect(1:end-1);
        d_normvect = [d_normvect;d_normvect(end)];

% ================================================================================================

function [] = plotShapeBridges(filledShape, firstBoundary, secondBoundary, ...
                    normvect_1, normvect_2, valid1, valid2, direction1_col, ...
                    direction1_row, direction2_col, direction2_row)

    row = 1;
    col = 2;

    vect_length = 1;

    %Shape
    figure;
    image(filledShape*255);colormap('gray'); %imshow(filledShape);
    hold on;

    % Boundary
    plot(firstBoundary(:,2),firstBoundary(:,1),'g','LineWidth',2);
    plot(secondBoundary(:,2),secondBoundary(:,1),'g','LineWidth',2);
    plot(firstBoundary(1,2),firstBoundary(1,1),'cs','MarkerSize',5);
    plot(secondBoundary(1,2),secondBoundary(1,1),'cs','MarkerSize',5);


    % Plot vectors
    plotVectors = 1;
    if (plotVectors)
        colours = ['r'];
        if (~isempty(normvect_1))
            for p = 1:length(normvect_1)
                p0 = firstBoundary(p,:);
```

```
                p1 = [firstBoundary(p,row) + vect_length * cos(normvect_1(p)*pi/180),  ...
                        firstBoundary(p,col) + vect_length * sin(normvect_1(p)*pi/180)];
                vectarrow(p0,p1,colours(mod(p,length(colours))+1),5,0.5,0.8)
            end
        end
        colours = ['b'];
        if (~isempty(normvect_2))
            for p = 1:length(normvect_2)
                p0 = secondBoundary(p,:);
                p1 = [secondBoundary(p,row) + vect_length * cos(normvect_2(p)*pi/180), ...
                            secondBoundary(p,col) + vect_length * sin(normvect_2(p)*pi/180)];
                vectarrow(p0,p1,colours(mod(p,length(colours))+1),5,0.5,0.8)
            end
        end
    end


    % Min distance pairs
    colours = 'rmcb';
    shapes = 'osdh>pv';

    for count = 1:length(direction1_col)

        if (valid1(count))
            currcolour = mod(count-1,length(colours)) + 1;
            currshape = mod(count-1,length(shapes)) + 1;

            plot(direction1_col(count),direction1_row(count), ...
                      [colours(currcolour),shapes(currshape)],'MarkerSize',16)
            plot(direction2_col(count),direction2_row(count), ...
                      [colours(currcolour),shapes(currshape)],'MarkerSize',16)
        end
    end


% ================================================================================
```

C.3 Code to Plot Column Diameters and to Find Nearest Neighbors

```
clear all;
close all;
clc
warning off all

fontSize = 20;
lineWidth = 2;
markerSize = 10;

seriesNums =  [1 2 3 4 5 6 7];
resampleFactor = [1 1 1 1 1 1 1];


baseDir = ['C:\My_Documents\Research\Research_Areas\ThreeDimGLAD_with_FIB\', ...
               'Processed_FIB_Scans\RegionProcessing\Shape_Scans_June_Feb09_Combo\'];


outputDir = [baseDir,'NearestNeighbors_circles_nearest_secondNearest_neighbors\'];
mkdir(outputDir);

binsMax = 1000;
binsStep = 50;


dataDir = ['\'];
filePrefix = 'Areas_';
fileSuffix = '.txt';
imagePrefix = '1_ORIG_';
imageSuffix = '.tif';


plotHistograms = 1;
```

```
disp('Processing data ...');
lastSliceNum = 0;
for series = seriesNums
    disp(['  Series: ',num2str(series)]);


    % Find all file names for the given series
    disp('    Getting file names');
    fileNames = dir([baseDir,'Series',num2str(series),dataDir, ...
                        filePrefix,'*',fileSuffix]);

    imageFileNames = dir([baseDir,'Series',num2str(series),dataDir, ...
                            imagePrefix,'*',imageSuffix]);

    data(series).numSlices = length(fileNames);

    disp('    Reading in file data');
    for sliceCount = 1:length(fileNames)
        [data(series).slice(sliceCount)] = readFile([baseDir,'Series', ...
                                                num2str(series),dataDir, ...
                                                 fileNames(sliceCount).name]);
    end

    for sliceCount = 1:length(imageFileNames)
        data(series).iminfo(sliceCount) = imfinfo([baseDir,'Series',num2str(series), ...
                                                dataDir,imageFileNames(sliceCount).name]);
    end

    for sliceCount = 1:length(fileNames)
        data(series).meanArea_nm2(sliceCount) = ...
                        mean(data(series).slice(sliceCount).area_nm2);
        data(series).stdArea_nm2(sliceCount) = ...
                        std(data(series).slice(sliceCount).area_nm2);
        data(series).meanEstRadius_nm(sliceCount) = ...
                        mean(data(series).slice(sliceCount).mean_radius_nm);
        data(series).std_meanEstRadius_nm(sliceCount) = ...
                         std(data(series).slice(sliceCount).mean_radius_nm);
        data(series).medianEstRadius_nm(sliceCount) = ...
                         median(data(series).slice(sliceCount).mean_radius_nm);
        data(series).meanRowRadius_nm(sliceCount) = ...
                         mean(data(series).slice(sliceCount).row_radius_nm);
        data(series).meanColRadius_nm(sliceCount) = ...
                         mean(data(series).slice(sliceCount).col_radius_nm);
        data(series).thickness(sliceCount) = data(series).slice(sliceCount).thickness;
        data(series).density(sliceCount) = data(series).slice(sliceCount).density;
        data(series).numShapes(sliceCount) = data(series).slice(sliceCount).numShapes;
        data(series).imageArea(sliceCount) = data(series).iminfo(sliceCount).Width * ...
                                                data(series).iminfo(sliceCount).Height;
        data(series).imageArea_nm2(sliceCount) = data(series).imageArea(sliceCount) * ...
                                                   (data(series).slice(sliceCount).imgres)^2;
        data(series).imageArea_um2(sliceCount) = data(series).imageArea_nm2(sliceCount)/ ...
                                                      (1000^2);
        data(series).numShapesPerArea(sliceCount) = data(series).numShapes(sliceCount) / ...
                                                         data(series).imageArea(sliceCount);
        data(series).numShapesPerArea_um2(sliceCount) = data(series).numShapes(sliceCount) / ...
                                                            data(series).imageArea_um2(sliceCount);

    end
end



colours = 'brmgckbrmgck';
shapes = 'svodx>+hsvodx>+h';

bins = [0:binsStep:binsMax];

% Fit curve to distributions of column diameters for each slice
disp('Fitting histograms');
fitType = 0;   % 1 = poisson, 0 = gaussian
if plotHistograms
    if fitType == 0
        p = [100, 100, 15];
    else
        p = [10, 150];
    end

    options = optimset('MaxFunEvals',3000,'MaxIter',2000,'TolFun',1e-7);
```

```
    for series = seriesNums
        for slice =  1:length(data(series).slice)
            if fitType == 0

                % Gaussian
                f = @(p,xdata)p(3).*exp(-(xdata - p(1)).^2 / (2*p(2)));

                x = bins;
                y = hist(data(series).slice(slice).mean_radius_nm*2,bins);
                [p, resnorm, residual, exitflag] = lsqcurvefit(f, p, x, y, [], [], options);

                peak_loc = p(1);
                peak_val = p(3);
                sigma = sqrt(p(2));

                data(series).peakDiameter_nm(slice) = peak_loc;
                data(series).sigmaDiameter_nm(slice) = sigma;
                data(series).peakVal(slice) = peak_val;

            else

                % Poisson
                f = @(p,xdata)p(2).*p(1).^xdata.*exp(-p(1))./factorial(xdata);

                t = [1:1:length(bins)];
                y = hist(data(series).slice(slice).mean_radius_nm*2,bins);
                [p, resnorm, residual, exitflag] = lsqcurvefit(f, p, t, y, [], [], options);

                %figure;plot(bins,y,'bo');hold on;plot(bins,f(p,t),'r-');

                lambda = p(1);
                multFactor = p(2);

                data(series).peakLambda(slice) = lambda;
                data(series).peakLambda_true(slice) = lambda * bins(end)/length(bins);
                data(series).multFactor(slice) = multFactor;

            end
        end
    end
end

disp('Plotting figures');
if plotHistograms
    for series = seriesNums
        for slice = 1:length(data(series).slice)
            x = bins;
            y = hist(data(series).slice(slice).mean_radius_nm*2,bins);

            if fitType == 0
                t = [1:1:x(end)];
                fit_y = f([data(series).peakDiameter_nm(slice), ...
                                data(series).sigmaDiameter_nm(slice).^2, ...
                                 data(series).peakVal(slice)], t);
            else
                t = [1:1:length(bins)];
                fit_y = f([data(series).peakLambda(slice), data(series).multFactor(slice)], t);
            end

            figure('Visible','off');
            plot(x, y,'ko', 'LineWidth', lineWidth, 'MarkerSize', markerSize, ...
                          'MarkerFaceColor', 'w', 'MarkerEdgeColor', 'k');
            hold on;
            plot(t, fit_y,'k--', 'LineWidth', lineWidth);
            plot(ones(1,2)*data(series).meanEstRadius_nm(slice)*2,[0 max(y)],'kd-', ...
                       'LineWidth', lineWidth, 'MarkerSize', markerSize, ...
                       'MarkerFaceColor', 'w', 'MarkerEdgeColor', 'k');
            xlabel('w (nm)','FontSize',fontSize,'FontWeight','bold');
            ylabel('f (-)','FontSize',fontSize,'FontWeight','bold');

            if fitType == 0
                title(['\mu = ',num2str(data(series).peakDiameter_nm(slice)), ...
                        ', p = ',num2str(data(series).peakVal(slice)), ...
                       ', d = ',num2str(data(series).meanEstRadius_nm(slice)*2)]);
            else
                title(['\lambda = ',num2str(data(series).peakLambda_true(slice)), ...
                        ', m = ',num2str(data(series).multFactor(slice)), ...
                       ', d = ',num2str(data(series).meanEstRadius_nm(slice)*2)]);
            end
            set(gca,'LineWidth',lineWidth,'FontSize',fontSize,'FontWeight','bold');
```

```
            set(gca,'Xtick',0:100:500);

            %axis([0 500 0 15]);
            print('-dtiff',[baseDir,'\Series',num2str(series),'_', ...
                                num2str(data(series).slice(slice).slice), ...
                                '_meanDiameter_histogram_bin',num2str(binsStep)]);
            close;
        end
    end
end

if plotHistograms

    if fitType == 0
        figure('Visible','off');
        for series = seriesNums
            plot(data(series).thickness, data(series).peakDiameter_nm,'ko', ...
                      'LineWidth', lineWidth, 'MarkerSize', markerSize, ...
                      'MarkerFaceColor', 'w', 'MarkerEdgeColor', 'k');
            hold on;
        end
        xlabel('t (nm)','FontSize',fontSize,'FontWeight','bold');
        ylabel('w (nm)','FontSize',fontSize,'FontWeight','bold');
        set(gca,'LineWidth',lineWidth,'FontSize',fontSize,'FontWeight','bold');
        axis([0 5000 0 300]);
        print('-dtiff',[baseDir,'\Gaussian_mean_diameter']);
        close;

    else

        figure('Visible','off');
        for series = seriesNums
            plot(data(series).thickness, data(series).peakLambda_true,'ko', ...
                      'LineWidth', lineWidth, 'MarkerSize', markerSize, ...
                      'MarkerFaceColor', 'w', 'MarkerEdgeColor', 'k');
            hold on;
        end
        xlabel('t (nm)','FontSize',fontSize,'FontWeight','bold');
        ylabel('w (nm)','FontSize',fontSize,'FontWeight','bold');
        set(gca,'LineWidth',lineWidth,'FontSize',fontSize,'FontWeight','bold');
        axis([0 5000 0 300]);
        print('-dtiff',[baseDir,'\Poisson_mean_diameter']);
        close;
    end
end


% ------------------------------------------------------------------------
%
% Code to fit curves to column density
% Also find fit to column diameter and standard deviation of column diameter
% distribution curves


    % ---------------------------------------------------------------
    % Fit curve to column density curve
    disp('Fitting column density curve');
    first_point_nm = 100;
    last_point_nm = 4400;
    x_fit = 1:1:5000;

    f = @(p,xdata)p(1)./(xdata.^p(2));

    all_thickness = [];
    all_densities = [];
    for series = seriesNums
        all_thickness = [all_thickness data(series).thickness];
        all_densities = [all_densities data(series).numShapesPerArea_um2];
    end

    [x,i] = sort(all_thickness);
    y = all_densities(i);
    tmp = find(x > first_point_nm);
    x = x(tmp);
    y = y(tmp);
    tmp = find(x < last_point_nm);
    x = x(tmp);
    y = y(tmp);

    k_o = [500];
    e_o = [1];
```

```
[p, resnorm, residual, exitflag] = lsqcurvefit(f, [k_o, e_o], x, y);
k_o = p(1);
e_o = p(2);

disp([' ... k_o = ',num2str(k_o),', exp = ',num2str(e_o)]);

figure;
leg = [];
for series = seriesNums
    legEntry = [' ',num2str(series)];
    leg = [leg; legEntry];
    plot(data(series).thickness, data(series).numShapesPerArea_um2, ...
           ['k',shapes(series)], 'LineWidth', lineWidth, 'MarkerSize', ...
           markerSize, 'MarkerFaceColor', 'w', 'MarkerEdgeColor', 'k');
    hold on;
end
plot(x_fit, f([k_o, e_o], x_fit), 'k--', 'LineWidth', lineWidth);
xlabel('Thickness (nm)','FontSize',fontSize,'FontWeight','bold');
ylabel('Columns Density [column \mum^{-2}]','FontSize',fontSize,'FontWeight','bold');
set(gca,'LineWidth',lineWidth,'FontSize',fontSize,'FontWeight','bold');
legend(leg, -1);
axis([0 5000 0 400]);
print('-dtiff',[baseDir,'\NumColumnsPerArea_bw']);
close;



% ------------------------------------------------------------------
% Fit p-scaling curve to column diameter curve
disp('Fitting diameter curve');
first_point_nm = 0;
last_point_nm = 4400;
x_fit = 1:1:5000;
w_o = 10;
h_p = 0.35;
f = @(p,xdata)p(1)*xdata.^p(2);

all_thickness = [];
all_diameters = [];
for series = seriesNums
    all_thickness = [all_thickness data(series).thickness];
    all_diameters = [all_diameters data(series).meanEstRadius_nm*2];
end

[x,i] = sort(all_thickness);
y = all_diameters(i);
tmp = find(x > first_point_nm);
x = x(tmp);
y = y(tmp);
tmp = find(x < last_point_nm);
x = x(tmp);
y = y(tmp);


[p, resnorm, residual, exitflag] = lsqcurvefit(f, [w_o, h_p], x, y);
w_o = p(1);
h_p = p(2);

disp([' ... w_o = ',num2str(w_o),', p = ',num2str(h_p)]);



% ------------------------------------------------------------------
% Fit p-scaling curve to standard deviation of diameter curve

disp('Fitting diameter std dev curve');
first_point_nm = 0;
last_point_nm = 4400;
x_fit = 1:1:5000;
w_s = 10;
h_s = 0.35;
f = @(p,xdata)p(1)*xdata.^p(2);

all_thickness = [];
all_stddev = [];
for series = seriesNums
    all_thickness = [all_thickness data(series).thickness];
    all_stddev = [all_stddev data(series).std_meanEstRadius_nm*2];
end
```

```
    [x,i] = sort(all_thickness);
    y = all_stddev(i);
    tmp = find(x > first_point_nm);
    x = x(tmp);
    y = y(tmp);
    tmp = find(x < last_point_nm);
    x = x(tmp);
    y = y(tmp);


    [p, resnorm, residual, exitflag] = lsqcurvefit(f, [w_s, h_s], x, y);
    w_s = p(1);
    h_s = p(2);

    disp([' ... w_o = ',num2str(w_s),', p = ',num2str(h_s)]);


    figure;
    leg = [];
    counter = 0;
    for series = seriesNums
        counter = counter + 1;
        legEntry = [' ',num2str(counter)];
        leg = [leg; legEntry];
        plot(data(series).thickness, data(series).meanEstRadius_nm*2, ['k',shapes(series)], ...
                'LineWidth', lineWidth, 'MarkerSize', markerSize, 'MarkerFaceColor', ...
                'w', 'MarkerEdgeColor', 'k');
        hold on;
    end
    plot(x_fit, f([w_o, h_p], x_fit), 'k--', 'LineWidth', lineWidth);
    plot(x_fit, f([w_o, h_p] , x_fit) + f([w_s, h_s], x_fit), 'k-.', 'LineWidth', lineWidth);
    plot(x_fit, f([w_o, h_p] , x_fit) - f([w_s, h_s], x_fit), 'k-.', 'LineWidth', lineWidth);
    xlabel('Thickness (nm)','FontSize',fontSize,'FontWeight','bold');
    ylabel('Mean Diameter (nm)','FontSize',fontSize,'FontWeight','bold');
    set(gca,'LineWidth',lineWidth,'FontSize',fontSize,'FontWeight','bold');
    legend(leg, -1);
    axis([0 5000 0 300]);
    print('-dtiff',[baseDir,'\MeanDiameter_plus_error_bw']);
    close;

end

% ================================================================================

% Code to calculate nearest neighbors

    disp('Calculating nearest neighbors ...');
    nn_counter = 0;
    for series = [1 2 3 4 5 6 7]
        disp(['  Series: ',num2str(series)]);
        for sliceCount = 1:length(data(series).slice)
            nn_counter = nn_counter + 1;

            disp(['    Slice: ',num2str(data(series).slice(sliceCount).slice)]);

            % All shapes for the given image
            uniqueShape = 1:length(data(series).slice(sliceCount).row_bottom_nm);

            % Radius along row and column vector of each shape
            currRowRadiusVector = round((data(series).slice(sliceCount).row_bottom_nm(uniqueShape) - ...
                                    data(series).slice(sliceCount).row_top_nm(uniqueShape))/2);
            currColRadiusVector = round((data(series).slice(sliceCount).col_right_nm(uniqueShape) - ...
                                    data(series).slice(sliceCount).col_left_nm(uniqueShape))/2);

            % Center Position (in row & col) of shape
            currRowVector = round((data(series).slice(sliceCount).row_bottom_nm(uniqueShape) + ...
                                  data(series).slice(sliceCount).row_top_nm(uniqueShape))/2);
            currColVector = round((data(series).slice(sliceCount).col_right_nm(uniqueShape) + ...
                                    data(series).slice(sliceCount).col_left_nm(uniqueShape))/2);

            % Calculated mean radius of shape (from previous region processing
            % analysis)
            currRadiusVector = data(series).slice(sliceCount).mean_radius_nm(uniqueShape);


            currRowMatrix = repmat(currRowVector, length(currRowVector), 1);
            currColMatrix = repmat(currColVector, length(currColVector), 1);
            currRadiusMatrix = repmat(currRadiusVector, length(currRadiusVector), 1);

            % Determine the distances between center positions of all shapes
```

```
distanceMatrix = tril(9999*ones(size(currRowMatrix))) + ...
                 triu(ones(size(currRowMatrix)),1) .* ...
                    sqrt( (currRowMatrix - currRowMatrix').^2 + ...
                          (currColMatrix - currColMatrix').^2 );

% Determine the spacing in between shapes
spacingMatrix = distanceMatrix - (currRadiusMatrix + currRadiusMatrix');

% Create an image with all shapes blocked off
disp('     Creating column image ...');
currImage = zeros(max(data(series).slice(sliceCount).row_bottom_nm(uniqueShape)), ...
                  max(data(series).slice(sliceCount).col_right_nm(uniqueShape)));
counter = 0;
for currShape = uniqueShape
    disp(['      Column ',num2str(currShape)]);
    counter = counter + 1;

    curr_rad = round(currRadiusVector(currShape));
    curr_row_center = round(currRowVector(currShape));
    curr_col_center = round(currColVector(currShape));

    curr_row_vector = 0:1:curr_rad;
    curr_col_vector = round(sqrt(curr_rad^2 - curr_row_vector.^2));


    curr_cols_matrix = (linspaceNDim(zeros(size(curr_col_vector)), ...
                                     curr_col_vector, (curr_rad+1)))';
    curr_rows_matrix = repmat(curr_row_vector, (curr_rad+1), 1);


    % First quadrant
    first_rows_matrix = round(min(size(currImage,1), ...
                                  max(1, curr_rows_matrix + curr_row_center)));
    first_cols_matrix = round(min(size(currImage,2), ...
                                  max(1, curr_cols_matrix + curr_col_center)));

    ind = sub2ind(size(currImage), first_rows_matrix, first_cols_matrix);
    currImage(ind) = counter;


    % Second quadrant
    first_rows_matrix = round(min(size(currImage,1), ...
                                   max(1, curr_rows_matrix + curr_row_center)));
    first_cols_matrix = round(min(size(currImage,2), ...
                                   max(1, curr_col_center - curr_cols_matrix)));

    ind = sub2ind(size(currImage), first_rows_matrix, first_cols_matrix);
    currImage(ind) = counter;

    % Third quadrant
    first_rows_matrix = round(min(size(currImage,1), ...
                                  max(1, curr_row_center - curr_rows_matrix)));
    first_cols_matrix = round(min(size(currImage,2), ...
                                  max(1, curr_col_center - curr_cols_matrix)));

    ind = sub2ind(size(currImage), first_rows_matrix, first_cols_matrix);
    currImage(ind) = counter;

    % Fourth quadrant
    first_rows_matrix = round(min(size(currImage,1), ...
                                  max(1, curr_row_center - curr_rows_matrix)));
    first_cols_matrix = round(min(size(currImage,2), ...
                                  max(1, curr_cols_matrix + curr_col_center)));

    ind = sub2ind(size(currImage), first_rows_matrix, first_cols_matrix);
    currImage(ind) = counter;

    clear curr_cols_matrix
    clear curr_rows_matrix
    clear first_rows_matrix
    clear first_cols_matrix
    clear ind
end

% Find nearest neighbors along certain trajectories
% E.g. up, left,right, down, left-down, left-right etc...
% Find mean distance and spacing of these
% Find nearest neighbor
% Second nearest neighbor etc...
disp('     Finding nearest neighbor shapes');
angles = [0:30:330]; % [0:45:315];
```

```
doPlots = 0;
currDirection = 0;
nearestShape = [];
nearestDistance = [];
nearestSpacing = [];
for currAngle = angles
    disp([’       Angle ’,num2str(currAngle)]);
    currDirection = currDirection + 1;

    disp(’          Creating mask’);
    [maskImage] = provideMaskImageAlongVector(currImage, ...
                                              data(series).slice(sliceCount), ...
                                              uniqueShape, currRowVector, ...
                                              currColVector, currRadiusVector, ...
                                              currAngle, doPlots);


    disp(’          Applying mask’);
    [nearestShape(currDirection,:), nearestDistance(currDirection,:), ...
       nearestSpacing(currDirection,:)] = ...
                  findNearestShapeAlongVector(uniqueShape, currImage, maskImage, ...
                                               distanceMatrix, spacingMatrix, currAngle, ...
                                               currRowVector, currColVector, doPlots, ...
                                              [outputDir,’\Series’,num2str(series),’_Slice’, ...
                                              num2str(data(series).slice(sliceCount).slice), ...
                                              ’_maskImage_’,num2str(currAngle)]);
end


for angleCounter = 1:length(angles)
    disp([’         ’,num2str(angles(angleCounter)), ...
          ’, mean = ’,num2str(mean(nearestDistance(angleCounter, ...
                              find(nearestDistance(angleCounter,:) < 9999))))]);
end


% Need to ignore duplicate shapes
numShapes = size(nearestShape,2);
use_nearestDistance = [];
use_nearestSpacing = [];
for shapeCounter = 1:numShapes
    [uniqueShapes, uniqueIndex, j] = unique(nearestShape(:,shapeCounter));
    use_nearestDistance = [use_nearestDistance; nearestDistance(uniqueIndex,shapeCounter)];
    use_nearestSpacing = [use_nearestSpacing; nearestSpacing(uniqueIndex,shapeCounter)];

end


disp(’     Calculating histograms of shape to shape distances’);
binsMax_distance = 1000;
binsMax_spacing = 500;
binsStep = 5;
bins_distance = [0:binsStep:binsMax_distance];
bins_spacing = [0:binsStep:binsMax_spacing];
histDistance = hist(use_nearestDistance(find(use_nearestDistance < binsMax_distance & ...
                                             use_nearestDistance > 0)),bins_distance);
histSpacing = hist(use_nearestSpacing(find(use_nearestSpacing < binsMax_spacing & ...
                                           use_nearestSpacing > 0)),bins_spacing);
meanDistance = mean(use_nearestDistance(find(use_nearestDistance < binsMax_distance)));
stdDistance = std(use_nearestDistance(find(use_nearestDistance < binsMax_distance)));
meanSpacing = mean(use_nearestSpacing(find(use_nearestSpacing < binsMax_spacing)));
meanDistance_gr0 = mean(use_nearestDistance(find(use_nearestDistance > 0 & ...
                                                 use_nearestDistance < binsMax_distance)));
stdDistance_gr0 = std(use_nearestDistance(find(use_nearestDistance > 0 & ...
                                               use_nearestDistance < binsMax_distance)));
meanSpacing_gr0 = mean(use_nearestSpacing(find(use_nearestSpacing > 0 & ...
                                               use_nearestSpacing < binsMax_spacing)));

nn.thickness(nn_counter) = data(series).thickness(sliceCount);
nn.meanDistance(nn_counter) = meanDistance;
nn.meanSpacing(nn_counter) = meanSpacing;
nn.meanDistance_gr0(nn_counter) = meanDistance_gr0;
nn.meanSpacing_gr0(nn_counter) = meanSpacing_gr0;


fid = fopen([outputDir,’\Series’,num2str(series),’_Slice’, ...
                        num2str(data(series).slice(sliceCount).slice), ...
                         ’_nearestHistograms.txt’],’w’);
fprintf(fid,’Series, %d, Slice, %d, \n’,series,data(series).slice(sliceCount).slice);
fprintf(fid,’Thickness, %d \n’,data(series).thickness(sliceCount));
fprintf(fid,’Mean Distance, %6.4f \n’,meanDistance);
```

```
            fprintf(fid,'StdDev Distance, %6.4f \n',stdDistance);
            fprintf(fid,'Mean Spacing, %6.4f \n',meanSpacing);
            fprintf(fid,'Mean Distance > 0, %6.4f \n',meanDistance_gr0);
            fprintf(fid,'StdDev Distance > 0, %6.4f \n',stdDistance_gr0);
            fprintf(fid,'Mean Spacing > 0, %6.4f \n',meanSpacing_gr0);
            fprintf(fid,'Bin, Distance \n');
            for count = 1:length(bins_distance)
                fprintf(fid,'%d, %d \n', bins_distance(count), histDistance(count));
            end
            fprintf(fid,'Bin, Spacing \n');
            for count = 1:length(bins_spacing)
                fprintf(fid,'%d, %d \n', bins_spacing(count), histSpacing(count));
            end
            fclose(fid);

        end
    end

    save('nn_circles_all','nn');


% ============================================================================

% Code to plot nearest neighbors


clear all;
close all;
clc
warning off all
colours = 'brmgckbrmgck';
shapes = 'svodx>+hsvodx>+h';


fontSize = 20;
lineWidth = 2;
markerSize = 10;


seriesNums =  [1 2 3 4 5 6 7];
resampleFactor = [1 1 1 1 1 1 1];


baseDir = ['C:\My_Documents\Research\Research_Areas\ThreeDimGLAD_with_FIB\', ...
              'Processed_FIB_Scans\RegionProcessing\Shape_Scans_June_Feb09_Combo\', ...
               'NearestNeighbors_circles_nearest_secondNearest_neighbors\'];


seriesName = 'Series';
sliceName = 'Slice';

fileNames = dir([baseDir,'Series*_Slice*_nearestHistograms.txt']);
fileCount = length(fileNames);

seriesCounter = zeros(size(seriesNums));
all_thickness = [];
all_dist = [];
all_stddev = [];
all_series = [];


for fileNum = 1:fileCount
    currFileName = fileNames(fileNum).name;
    [data(fileNum)] = readFileNN([baseDir,currFileName]);
    seriesCounter(data(fileNum).series) = seriesCounter(data(fileNum).series) + 1;

    all_thickness = [all_thickness data(fileNum).thickness];
    all_dist = [all_dist data(fileNum).meanDist];
    all_stddev = [all_stddev data(fileNum).stddevDist];
    all_series = [all_series data(fileNum).series];
end


% Fit p-scaling curve to column spacing curve
disp('Fitting column spacing curve');
first_point_nm = 0;
last_point_nm = 4400;
x_fit = 1:1:5000;
w_o = 10;
h_p = 0.35;
f = @(p,xdata)p(1)*xdata.^p(2);
```

```
x = all_thickness;
y = all_dist;
tmp = find(x > first_point_nm);
x = x(tmp);
y = y(tmp);
tmp = find(x < last_point_nm);
x = x(tmp);
y = y(tmp);


[p, resnorm, residual, exitflag] = lsqcurvefit(f, [w_o, h_p], x, y);
w_o = p(1);
h_p = p(2);

disp([' ... w_o = ',num2str(w_o),', p = ',num2str(h_p)]);

% w_o = 25.5463, p = 0.2902 - for 45 degree neighbors
% w_o = 33.1718, p = 0.27997 - for 30 nearest and second nearest neighbors
%   / 1.37 to get nearest neighbor estimate =>  w_o = 24.2


% Fit p-scaling curve to stddev column spacing curve
disp('Fitting column spacing curve');
first_point_nm = 0;
last_point_nm = 4400;
x_fit = 1:1:5000;
w_s = 10;
h_s = 0.35;
f = @(p,xdata)p(1)*xdata.^p(2);

x = all_thickness;
y = all_stddev;
tmp = find(x > first_point_nm);
x = x(tmp);
y = y(tmp);
tmp = find(x < last_point_nm);
x = x(tmp);
y = y(tmp);


[p, resnorm, residual, exitflag] = lsqcurvefit(f, [w_s, h_s], x, y);
w_s = p(1);
h_s = p(2);

disp([' ... w_o = ',num2str(w_s),', p = ',num2str(h_s)]);

% w_o = 10.0533, p = 0.33305
%

disp('Plotting ...');

figure('Visible','off');
counter = 0;
leg = [];
for currSeries = seriesNums
    counter = counter + 1;
    legEntry = [' ',num2str(counter)];
    leg = [leg; legEntry];

    curr_thickness = all_thickness(find(all_series == currSeries));
    curr_meanDistance = all_dist(find(all_series == currSeries));

    plot(curr_thickness, curr_meanDistance,['k',shapes(currSeries)], ...
                'LineWidth', lineWidth, 'MarkerSize', markerSize, ...
                'MarkerFaceColor', 'w', 'MarkerEdgeColor', 'k');
    hold on;
end
plot(x_fit, f([w_o, h_p], x_fit), 'k--', 'LineWidth', lineWidth);
plot(x_fit, f([w_o, h_p], x_fit) + f([w_s, h_s], x_fit), 'k-.', 'LineWidth', lineWidth);
plot(x_fit, f([w_o, h_p], x_fit) - f([w_s, h_s], x_fit), 'k-.', 'LineWidth', lineWidth);
xlabel('Thickness (nm)','FontSize',fontSize,'FontWeight','bold');
ylabel('Column center to center distance (nm)','FontSize',fontSize,'FontWeight','bold');
legend(leg,-1);
set(gca,'LineWidth',lineWidth,'FontSize',fontSize,'FontWeight','bold');
axis([0 5000 0 500]);
print('-dtiff',[baseDir,'\meanColumnDistance']);
close


% Plot histograms out again with reduced bin size (i.e. combine some bins)
```

```
for dataFile = 1:length(data)

    origBinSpacing = data(dataFile).bin(2)-data(dataFile).bin(1);

    % Gaussian fit to histogram
    f = @(p,xdata)p(3).*exp(-(xdata - p(1)).^2 / (2*p(2)));
    p = [100, 100, 15];
    options = optimset('MaxFunEvals',3000,'MaxIter',2000,'TolFun',1e-7);


    x = data(dataFile).bin;
    y = data(dataFile).freq;
    [p, resnorm, residual, exitflag] = lsqcurvefit(f, p, x, y, [], [], options);

    t = [1:1:x(end)];
    fit_y = f(p, t);


    figure('Visible','off');
    plot(data(dataFile).bin, data(dataFile).freq,'ko', 'LineWidth', lineWidth, ...
             'MarkerSize', markerSize, 'MarkerFaceColor', 'w', 'MarkerEdgeColor', 'k');
    hold on;
    plot(t, fit_y, 'k--', 'LineWidth', lineWidth);
    xlabel('c (nm)','FontSize',fontSize,'FontWeight','bold');
    ylabel('f (-)','FontSize',fontSize,'FontWeight','bold');
    set(gca,'LineWidth',lineWidth,'FontSize',fontSize,'FontWeight','bold');
    print('-dtiff',[baseDir,'\Histogram_',num2str(data(dataFile).series), ...
                '_',num2str(data(dataFile).slice),'_space',num2str(origBinSpacing)]);
    close

    % Resample the original histograms to a coarser bin spacing
    sampleRates = [2 4 8];

    for sampleRate = sampleRates
        new_sample_count = floor(length(data(dataFile).bin)/sampleRate);
        new_bin = data(dataFile).bin(1:sampleRate:new_sample_count*sampleRate);
        new_freq = data(dataFile).freq(1:sampleRate:new_sample_count*sampleRate);
        for counter = 2:sampleRate
            new_freq = new_freq +  data(dataFile).freq(1:sampleRate:new_sample_count*sampleRate);
        end

        x = new_bin;
        y = new_freq;
        [p, resnorm, residual, exitflag] = lsqcurvefit(f, p, x, y, [], [], options);

        t = [1:1:x(end)];
        fit_y = f(p, t);


        figure('Visible','off');
        plot(new_bin, new_freq,'ko', 'LineWidth', lineWidth, 'MarkerSize', ...
                  markerSize, 'MarkerFaceColor', 'w', 'MarkerEdgeColor', 'k');
        hold on;
        plot(t, fit_y, 'k--', 'LineWidth', lineWidth);
        xlabel('c (nm)','FontSize',fontSize,'FontWeight','bold');
        ylabel('f (-)','FontSize',fontSize,'FontWeight','bold');
        set(gca,'LineWidth',lineWidth,'FontSize',fontSize,'FontWeight','bold');
        print('-dtiff',[baseDir,'\Histogram_',num2str(data(dataFile).series), ...
                 '_',num2str(data(dataFile).slice),'_space',num2str(origBinSpacing*sampleRate)]);
        close
    end
end
```

C.4 Code to Make Movies of FIB slices and to save binary file for analysis by JEOL TEMography software

```
close all;
clear all;
clc;
warning off all

homework = 2;

switch homework
```

```
    case 1
        beginDir = ['C:\My_Documents\'];
    case 2
        beginDir = ['H:\private\My Documents\'];
end


series = [6 5 3];
refSeries = 5;
inputDir = [beginDir,'Research\Research_Areas\ThreeDimGLAD_with_FIB\Processed_FIB_Scans\', ...
                     'RegionProcessing\Stitched_Shape_Scans_June09_v1\Series'];
parameterFile = [beginDir,'Research\Research_Areas\ThreeDimGLAD_with_FIB\Matlab\', ...
                     'Process_FIB_Slices_TopDownView\FIBscan_June09_TiO2VP5um_stitch.m'];


subInputDir = ['\Threshold\'];
outputDir = [beginDir,'Research\Research_Areas\ThreeDimGLAD_with_FIB\', ...
              'Processed_FIB_Scans\Tomography\Series'];
for currSeries = series
    outputDir = [outputDir,num2str(currSeries),'_'];
end
outputDir = [outputDir,'\'];


mkdir(outputDir);

% Get scan parameters
disp('Getting scan parameters');
run(parameterFile);


filePrefixCounter = '1_ORIG_';
filePrefix = '3_EQ_';
sliceNumPos = 2;
fileSuffix = '.tif';
movieName = 'BottomUp';
tomoName = 'GLAD';
extension = '.bin';

framesPerSecond = 15;

minmaxPair = [ 1 30000];

mov = avifile([outputDir,'\',movieName,'.avi'],'compression','None','fps',framesPerSecond);
fid = fopen([outputDir,'\',tomoName,extension],'w');

refImageRes = scans.image_res_nm(refSeries);

% Determine total number of z-direction files
disp('Determining total number of files ...');
numImages = 0;
for currSeries = series
    disp([' ... Series ',num2str(currSeries)]);
    % Find all file names for the given series
    fileNames = dir([inputDir,num2str(currSeries),subInputDir,filePrefix,'*',fileSuffix]);
    fileNamesCounter = dir([inputDir,num2str(currSeries),subInputDir,filePrefixCounter,'*',fileSuffix]);
    numDupSlices = round(scans.slice_res_nm(currSeries) / refImageRes );

    numImages = numImages + length(fileNamesCounter) * numDupSlices;

end

disp('Writing out data ...');
for currSeries = series
    disp([' ... Series ',num2str(currSeries)]);
    % Find all file names for the given series
    fileNames = dir([inputDir,num2str(currSeries),subInputDir,filePrefix,'*',fileSuffix]);
    fileNamesCounter = dir([inputDir,num2str(currSeries), ...
                            subInputDir,filePrefixCounter,'*',fileSuffix]);



    disp(['   ... Reading in image files and creating movie ...']);
    sliceCounter = 0;
    for currFile = [length(fileNames):-1:1]
        % Get current file name
        fileName = fileNames(currFile).name;
        tmp = strfind(fileNames(currFile).name,'_');
        if (length(tmp) == sliceNumPos)
            sliceCounter = sliceCounter + 1;
```

```
            slice = str2num(fileNames(currFile).name(tmp(2)+1: ...
                            strfind(fileNames(currFile).name,fileSuffix)-1));
            disp([’      ... Slice ’,num2str(slice)]);

            % Read in the input image
            currImage = double(imread([inputDir,num2str(currSeries),subInputDir,fileName]));

            F = im2frame(currImage/255);
            mov = addframe(mov,F);


            minNum = minmaxPair(1,1);
            maxNum = minmaxPair(1,2);
            imageFile = squeeze(currImage(:,:,1)) / 255 * maxNum;


            byte1 = floor(imageFile/16^3);
            byte2 = floor((imageFile - byte1*16^3)/16^2);
            byte3 = floor((imageFile - byte1*16^3 - byte2*16^2)/16^1);
            byte4 = floor((imageFile - byte1*16^3 - byte2*16^2 - byte3*16^1)/16^0);
            imageFile_inc = byte1*16^3 + byte2*16^2 + byte3*16^1 + byte4*16^0;


            if (sliceCounter == 1)
                % Option 1:
                fwrite(fid,[size(imageFile_inc,1); size(imageFile_inc,2); numImages],’uint16’);
            end

            % There are 4 slices for every pixel

            numDupSlices = round(scans.slice_res_nm(currSeries) / refImageRes );

            for dupSlice = 1:numDupSlices
                fwrite(fid, reshape(imageFile_inc, 1, ...
                                    size(imageFile_inc,1) * size(imageFile_inc,2)),’uint16’);
            end

        end
    end
end

mov = close(mov);
fclose(fid);

clear all

disp(’Done’);
```

C.5 Code to Determine Surface Area from Milled Images

```
clear all;
close all;
clc;

warning off all

fontSize = 20;
lineWidth = 2;
markerSize = 10;


homework = 1;

switch homework
    case 1
        beginDir = [’C:\My_Documents\’];
    case 2
        beginDir = [’H:\private\My Documents\’];
end

series = [1,2,3,4,5,6,7];
inputDir = [beginDir,’Research\Research_Areas\ThreeDimGLAD_with_FIB\’, ...
                    ’Processed_FIB_Scans\RegionProcessing\Shape_Scans_June_Feb09_Combo’];
outputDir = [beginDir,’Research\Research_Areas\ThreeDimGLAD_with_FIB\’, ...
                    ’Processed_FIB_Scans\EdgeDetection’];
```

```
infoFilePrefix = 'Areas_';
infoFileSuffix = '.txt';
filePrefix = '4_THRESHOLD_';
fileSuffix = '.tif';

mkdir(outputDir);


numSeries = length(series);

disp('Reading in image files');
counter = 0;
for count = 1:numSeries
    disp(['  ... Series ',num2str(series(count))]);
    curr_outputDir = [outputDir,'\','Series',num2str(series(count))];
    mkdir(curr_outputDir);
    fileNames = dir([inputDir,'\','Series',num2str(series(count)),'\',filePrefix,'*',fileSuffix]);

    for count2 = 1:length(fileNames)
        counter = counter + 1;
        currFile = fileNames(count2).name;
        tmp = findstr(currFile,'_');
        tmp2 = findstr(currFile,fileSuffix);
        sliceText = currFile(tmp(2)+1:tmp2-1);
        slice = str2num(currFile(tmp(2)+1:tmp2-1));
        %data(count).slice(counter) = slice;
        disp(['    ... Count ',num2str(count2),', Slice ',num2str(slice)]);

        % Read in the input image
        currImage = double(imread([inputDir,'\','Series',num2str(series(count)),'\',currFile]));
        imageSize = size(currImage,1) * size(currImage,2);

        %figure;image(currImage*255);colormap('gray');

        [edgeImage, edgeImage_x, edgeImage_y] = findEdgesImage(currImage); %./255);

        %figure;image(edgeImage*255);colormap('gray');

        % Read in the info file
        [thicknessFile] = readFile([inputDir,'\','Series',num2str(series(count)), ...
                           '\',infoFilePrefix,sliceText,infoFileSuffix]);

        data.thickness_nm(counter) = thicknessFile.thickness;
        imageSize_nm2 = imageSize * (thicknessFile.imgres)^2;

        data.surfaceArea(counter) = sum(sum(edgeImage));
        data.imgres_nm(counter) = thicknessFile.imgres;
        data.imsize_nm2(counter) = imageSize_nm2;
        data.surfaceArea_nm(counter) = data.surfaceArea(counter) * data.imgres_nm(counter);
        data.surfaceArea_rel(counter) = data.surfaceArea_nm(counter) / data.imsize_nm2(counter);

    end
end

% Store thickness and surface area from 0 nm thickness
data.rev_surfaceArea_rel = data.surfaceArea_rel(end:-1:1);
data.rev_thickness_nm = data.thickness_nm(end:-1:1);

% Find thickness steps
data.delta_thickness_nm = data.rev_thickness_nm(2:end) - data.rev_thickness_nm(1:end-1);
data.delta_thickness_nm = [data.delta_thickness_nm data.delta_thickness_nm(end)];

% Calculate surface area as a function of thickness
data.surfaceArea_nm2 = data.rev_surfaceArea_rel * 1000^2.* data.delta_thickness_nm;

% Calculate cumulative surface area
data.cum_surfaceArea_m2_m2 = cumsum(data.surfaceArea_nm2) / 1000^2;


% Spline data
plot_thickness_nm = 0:100:5000;
plot_cum_surfaceArea_m2_m2 = ...
     spline(data.rev_thickness_nm, data.cum_surfaceArea_m2_m2, plot_thickness_nm);
plot_gasAdsorb_surfaceArea_m2_m2 = (plot_thickness_nm/1000 * 235 - 9);


x_fit = 1:1:5000;
w_o = 0.4;
h_p = 0.7;
f = @(p,xdata)p(1)*xdata.^p(2);
```

```
x = data.rev_thickness_nm(2:2:end);
y = data.cum_surfaceArea_m2_m2(2:2:end);

[p, resnorm, residual, exitflag] = lsqcurvefit(f, [w_o, h_p], x, y);
w_o = p(1);
h_p = p(2);

disp([' ... w_o = ',num2str(w_o),', p = ',num2str(h_p)]);

% w_o = 0.24863, p = 0.74446

colours = 'rbgmckrbgmck';
shapes = 'os>hvp+d<os>hv';


figure;
plot(data.rev_thickness_nm(2:2:end), data.cum_surfaceArea_m2_m2(2:2:end), 'ko', ...
     'LineWidth', lineWidth, 'MarkerSize', markerSize, 'MarkerFaceColor', ...
     'k', 'MarkerEdgeColor', 'k');
hold on;
xlabel('t (nm)','FontSize',fontSize,'FontWeight','bold');
ylabel(['SA (m^2 m^{-2})'],'FontSize',fontSize,'FontWeight','bold');
set(gca,'LineWidth',lineWidth,'FontSize',fontSize,'FontWeight','bold');
axis([0, 5000, 0, 140]);
print('-dtiff',[outputDir,'\FIB_Cumulative_Surface_Area']);
close



figure;
plot(plot_thickness_nm(2:2:end), plot_cum_surfaceArea_m2_m2(2:2:end), 'ko', ...
     'LineWidth', lineWidth, 'MarkerSize', markerSize, 'MarkerFaceColor', ...
     'k', 'MarkerEdgeColor', 'k');
hold on;
plot(plot_thickness_nm(2:2:end), ...
     plot_gasAdsorb_surfaceArea_m2_m2(2:2:end), 'ks', ...
     'LineWidth', lineWidth, 'MarkerSize', markerSize, 'MarkerFaceColor', ...
     'w', 'MarkerEdgeColor', 'k');
xlabel('t (nm)','FontSize',fontSize,'FontWeight','bold');
ylabel(['SAE (m^2 m^{-2})'],'FontSize',fontSize,'FontWeight','bold');
set(gca,'LineWidth',lineWidth,'FontSize',fontSize,'FontWeight','bold');
legend('FIB  ','GasAd',-1);
axis([0, 5000, 0, 1200]);
print('-dtiff',[outputDir,'\FIB_vs_GasAd_Surface_Area']);
close

figure;
plot(plot_thickness_nm(2:2:end), ...
      plot_gasAdsorb_surfaceArea_m2_m2(2:2:end)./plot_cum_surfaceArea_m2_m2(2:2:end), ...
     'ko', 'LineWidth', lineWidth, 'MarkerSize', markerSize, 'MarkerFaceColor', ...
     'k', 'MarkerEdgeColor', 'k');
hold on;
xlabel('t (nm)','FontSize',fontSize,'FontWeight','bold');
ylabel(['SAE_{GasAd} / SAE_{FIB}'],'FontSize',fontSize,'FontWeight','bold');
set(gca,'LineWidth',lineWidth,'FontSize',fontSize,'FontWeight','bold');
print('-dtiff',[outputDir,'\FIB_div_GasAd_Surface_Area']);
axis([0, 5000, 0, 10]);
close


disp(['Total Surface Area = ',num2str(data.cum_surfaceArea_m2_m2(end)), ...
       ' m^2/m^2 for a ', ...
       num2str(data.rev_thickness_nm(end)),'nm thick film.  SA = ', ...
       num2str(data.cum_surfaceArea_m2_m2(end) / (data.rev_thickness_nm(end)/1000)), ...
       ' m^2/m^2/um']);

disp('Done');
```

C.6 Code to Determine Inter-Column Spacing

```
clear all;
close all;
clc
warning off all
```

```
fontSize = 20;
lineWidth = 2;
markerSize = 10;
colours = 'brmgckbrmgckbrmgckbrmgck';
shapes = 'svodx>+hsvodx>+hsvodx>';
lineStyles = {'-','--',':','-.','-','--',':','-.','-','--',':','-.','-','--',':','-.'};
white = 255;
black = 0;


baseDir = ['C:\My_Documents\Research\Research_Areas\ThreeDimGLAD_with_FIB\'];
inputDir = [baseDir, 'Processed_FIB_Scans\TopDownView\Processed_Scans_June09\'];
outputDir = [baseDir, 'Processed_FIB_Scans\InterColumnSpacing\June09\'];
mkdir(outputDir);
parameterFile = [baseDir, 'Matlab\Process_FIB_Slices_TopDownView\FIBscan_June09_TiO2VP5um.m'];


seriesSelection(1).selection = [3 4 5];
seriesSelection(2).selection = [1:1:93];
seriesSelection(3).selection = [1:1:133];
seriesSelection(4).selection = [1:1:71];
seriesSelection(5).selection = [1:1:64];
seriesSelection(6).selection = [250:305];
seriesSelection(7).selection = [1:1:25];

resizeRate = [8.2/4 8.2/4 6.2/4 999 4/4 3/4 4 1 ];

% Equalize images in tiles
eqSquares =            [      2        3         4       4        4       3    4];
% Filter images to remove noise
filterCoeff =          [      3        3        3      3       3     3       3];

useSeries = [1 2 3 4 5 6 7];


% Bring all images to a common density for contrast equalization
% Leave the slices nearest to the top of the film since they are just
% the peaks of the columns
flatTargetDensity = [0.55 0.55 0.55 0.55 0.55 0.55 0.55];
noMatchDensity(1).slice = [48:50];
noMatchDensity(2).slice = [1:10];
noMatchDensity(3).slice = [];
noMatchDensity(4).slice = [];
noMatchDensity(5).slice = [];
noMatchDensity(6).slice = [];
noMatchDensity(7).slice = [];

% Some bounds for plotting
blackBinMax = [200 400 400 400 400 400 100];
black_peakFilterCoeff = [9 24 25 99 24 24 11];
black_peakBounds = [[10 50];[40 100];[10 200]; [10 200]; [10 200];[10 200];[10 80]];



subDir = ['Select\'];
inputFileName = ['SELECTUNFILTIMAGE'];
outputFileName_orig = ['ORIGIMAGE'];
outputFileName_eq = ['EQIMAGE'];
outputFileName_filt = ['FILTIMAGE'];
outputFileName_thresh = ['THRESHIMAGE'];
outputFileName_filled = ['FILLEDIMAGE'];
outputFileName_edge = ['EDGEIMAGE'];
outputFileName_void = ['VOIDDIST'];
outputFileName_diameters = ['COLDIAM'];


% Get scan parameters
disp('Getting scan parameters');
run(parameterFile);

% Resample images to common sampling scheme
scans.image_res_nm  = scans.image_res_nm ./ resizeRate;
scans.x_size_pixel = scans.x_size_pixel .* resizeRate;
scans.y_size_pixel = scans.y_size_pixel .* resizeRate;

[profile_thickness, profile_density] = openDensityPlot(baseDir);

% Open image files
disp('Opening and processing image files');
```

```
for series = useSeries
    disp([’  Series ’,num2str(series)]);

    for slice = (seriesSelection(series).selection + scans.firstImage(series))
        % Set up the file info
        currInputDir = [inputDir, ’Series’, num2str(series), ’\’, subDir];
        currOutputDir = [outputDir, ’Series’, num2str(series),’\’]; mkdir(currOutputDir);
        if (slice < 10)
            prefix = ’000’;
        elseif (slice < 100)
            prefix = ’00’;
        elseif (slice < 1000)
            prefix = ’0’;
        end
        currFileName = [inputFileName, ’_’, prefix, num2str(slice), suffix];
        disp([’    Slice ’,num2str(slice),’: ’,currFileName]);


        % Calculate the current thickness (height)
        disp([’      Calculating current thickness ’]);
        saveData(series).currThickness(slice) = scans.cumThickness(series) - ...
                            (slice - scans.firstImage(series)) * scans.slice_res_nm(series);
        disp([’         Thickness = ’,num2str(saveData(series).currThickness(slice))]);


        % Read in the input file
        origImage = uint8(imread([currInputDir,currFileName]));
        origImage = squeeze(origImage(:,:,1));
        imageSize_col = size(origImage,2);
        imageSize_row = size(origImage,1);


        % Enbiggen image and interpolate
        disp([’      Resizing image’]);
        resizedImage = imresize(origImage,resizeRate(series));
        origImage = resizedImage;

        % Calculate image size
        imageSize_col = size(origImage,2);
        imageSize_row = size(origImage,1);


        % Equalize the original image
        disp([’      Equalization in ’,num2str(eqSquares(series)), ...
                ’x’,num2str(eqSquares(series)),’ squares’]);
        [eqImage stretchMask filtMask] = adjContrast(origImage, eqSquares(series), 1);


        % Filter image
        disp([’      Filtering image with filterCoeff ’,num2str(filterCoeff(series))]);
        if (filterCoeff(series) > 1)
            b = ones(1,(filterCoeff(series)+1));
            a = (filterCoeff(series)+1)*[1 zeros(1,filterCoeff(series))];
            filtImage = filtfilt(b,a,double(eqImage));
            filtImage = uint8((filtfilt(b,a,filtImage’))’);
        else
            filtImage = eqImage;
        end



        % Determine target density
        targetDensity = flatTargetDensity(series);


        % Determine multiplier factor that will reach target density
        disp([’      Iterating to matching density target of ’,num2str(targetDensity)]);
        densityError = 0.01;
        maxIterations = 40;
        metTarget = 0;
        currIteration = 1;
        if (find(noMatchDensity(series).slice == slice))
            currMultiplier = 1;
            densityError = 1;
            currIteration = maxIterations - 1;
        else
            currMultiplier = 1;
        end
        multiplierStep = 0.01;
        while ((~metTarget) && (currIteration < maxIterations))
            disp([’        ... iteration ’,num2str(currIteration)]);
```

```
    % Threshold the equalized image
    level = graythresh(filtImage) * currMultiplier;
    disp([’          ... Thresholding with factor ’,num2str(currMultiplier), ...
          ’ to level ’,num2str(level)]);
    threshImage = im2bw(filtImage,level);

    % Fill holes in image
    disp([’          ... Filling holes of thresholded image’]);
    filledImage = threshImage; %imfill(threshImage, ’holes’);

    % Find density of thresholded images
    disp([’          ... Finding density of thresholded images’]);
    numWhites = sum(sum(filledImage == 1));
    numBlacks = sum(sum(filledImage == 0));
    saveData(series).density(slice) = numWhites ./ (numWhites + numBlacks);
    disp([’          ... Density = ’,num2str(saveData(series).density(slice))]);

    densityDiff = targetDensity - saveData(series).density(slice);
    if (abs(densityDiff) < densityError)
        metTarget = 1;
    elseif densityDiff > 0
        % Density too low, multiplier should go down
        currMultiplier = currMultiplier - multiplierStep;
    elseif densityDiff < 0
        % Density too high, multiplier should go up
        currMultiplier = currMultiplier + multiplierStep;
    end
    currIteration = currIteration + 1;
end
disp([’      Final Density = ’,num2str(saveData(series).density(slice))]);



% Find edges of thresholded image
disp([’      Finding edges of thresholded image’]);
[edgeImage, edgeImage_x, edgeImage_y] = findEdgesImage(filledImage);


% Save the final image
disp([’      Saving images’]);
imwrite(uint8(origImage),[currOutputDir, outputFileName_orig, ’_’, ...
          prefix, num2str(slice), suffix],’tiff’);
imwrite(uint8(eqImage),[currOutputDir, outputFileName_eq, ’_’, ...
           prefix, num2str(slice), suffix],’tiff’);
imwrite(uint8(stretchMask*100),[currOutputDir, outputFileName_eq, ’_stretchMask_’, ...
           prefix, num2str(slice), suffix],’tiff’);
imwrite(uint8(filtMask*100),[currOutputDir, outputFileName_eq, ’_filtMask_’, ...
           prefix, num2str(slice), suffix],’tiff’);
imwrite(uint8(filtImage),[currOutputDir, outputFileName_filt, ’_’, ...
           prefix, num2str(slice), suffix],’tiff’);
imwrite(uint8(255*threshImage),[currOutputDir, outputFileName_thresh, ’_’, ...
           prefix, num2str(slice), suffix],’tiff’);
imwrite(uint8(255*filledImage),[currOutputDir, outputFileName_filled, ’_’, ...
           prefix, num2str(slice), suffix],’tiff’);
imwrite(uint8(120*edgeImage_x),[currOutputDir, outputFileName_edge, ’_’, ...
           prefix, num2str(slice), suffix],’tiff’);
close all;


% Find distances between white columns
disp([’      Finding distances between white columns’]);
[blackDistances_x, meanDistances_x] = findBlackDistances(edgeImage_x);
[blackDistances_y, meanDistances_y] = findBlackDistances(edgeImage_y’);

smallBins = 0;

blackDistances_x_nm = blackDistances_x(find(blackDistances_x > smallBins)) * ...
                           scans.image_res_nm(series);
blackDistances_y_nm = blackDistances_y(find(blackDistances_y > smallBins)) * ...
                           scans.image_res_nm(series);


binSteps = 1;
black_bins = [0:binSteps:blackBinMax(series)];


% Find the histogram of void distances
disp([’      Finding the histogram of void distances’]);
blackDistances_x_peaks = hist(blackDistances_x_nm, black_bins);
blackDistances_y_peaks = hist(blackDistances_y_nm, black_bins);
```

```
blackDistances_peaks = (blackDistances_x_peaks + blackDistances_y_peaks)/2;
blackDistances_locations = black_bins;


% Filter the histogram
% Mean of voids important

disp(['      Filtering the histograms']);
filtHistNum = black_peakFilterCoeff(series);
b = ones(1,(filtHistNum+1));
a = (filtHistNum+1)*[1 zeros(1,filtHistNum)];

tmp = find(blackDistances_x_peaks > 0);
filt_peaks = spline(blackDistances_locations(tmp), blackDistances_x_peaks(tmp), ...
                    blackDistances_locations);
blackDistances_x_peaks_filt = filtfilt(b,a,double(filt_peaks));

tmp = find(blackDistances_y_peaks > 0);
filt_peaks = spline(blackDistances_locations(tmp), blackDistances_y_peaks(tmp), ...
                     blackDistances_locations);
blackDistances_y_peaks_filt = filtfilt(b,a,double(filt_peaks));

tmp = find(blackDistances_peaks > 0);
filt_peaks = spline(blackDistances_locations(tmp), blackDistances_peaks(tmp), ...
                     blackDistances_locations);
blackDistances_peaks_filt = filtfilt(b,a,double(filt_peaks));

[maxVal maxLoc] = ...
    max(blackDistances_x_peaks_filt(black_peakBounds(series,1):black_peakBounds(series,2)));
saveData(series).peakLoc_x(slice) = ...
    blackDistances_locations(maxLoc + black_peakBounds(series,1) - 1);
[maxVal maxLoc] = ...
    max(blackDistances_y_peaks_filt(black_peakBounds(series,1):black_peakBounds(series,2)));
saveData(series).peakLoc_y(slice) = ...
    blackDistances_locations(maxLoc + black_peakBounds(series,1) - 1);
[maxVal maxLoc] = ...
    max(blackDistances_peaks_filt(black_peakBounds(series,1):black_peakBounds(series,2)));
saveData(series).peakLoc(slice) = ...
    blackDistances_locations(maxLoc + black_peakBounds(series,1) - 1);
saveData(series).peakVal(slice) = maxVal;


% Locate the peaks in the histogram
disp(['      Finding peak (average) location of void distances']);
saveData(series).rmsLoc_x(slice) = sqrt( sum(blackDistances_x_nm.^2)/ ...
                                          length(blackDistances_x_nm)  );
saveData(series).meanLoc_x(slice) = mean(blackDistances_x_nm);
saveData(series).stdDev_x(slice) = std(blackDistances_x_nm);

saveData(series).rmsLoc_y(slice) = sqrt( sum(blackDistances_y_nm.^2)/ ...
                                          length(blackDistances_y_nm)  );
saveData(series).meanLoc_y(slice) = mean(blackDistances_y_nm);
saveData(series).stdDev_y(slice) = std(blackDistances_y_nm);

saveData(series).rmsLoc(slice) = sqrt( ( sum(blackDistances_x_nm.^2)/ ...
                                           length(blackDistances_x_nm) + ...
                                     sum(blackDistances_y_nm.^2)/ ...
                                           length(blackDistances_y_nm) )/2 );
saveData(series).meanLoc(slice) = mean([mean(blackDistances_x_nm), ...
                                    mean(blackDistances_y_nm)]);
saveData(series).stdDev(slice) = std([blackDistances_x_nm, blackDistances_y_nm]);

disp(['         Void Distance = ',num2str(saveData(series).rmsLoc(slice)),' rms, ', ...
       num2str(saveData(series).meanLoc(slice)),' mean, ', ...
       num2str(saveData(series).peakLoc(slice)),' peak']);


% Plot void distances
disp(['      Plotting peak (average) location of void distances']);
figure('Visible','off');
tmp = find(blackDistances_peaks > 0);
x = blackDistances_locations(tmp);
y = blackDistances_peaks(tmp);
plot(x, y,'ko', 'LineWidth', lineWidth, 'MarkerSize', markerSize, ...
           'MarkerFaceColor', 'w', 'MarkerEdgeColor', 'k');
hold on;

if (series == 1)
    tmp = find(x > 1);
    x = x(tmp);
    y = y(tmp);
```

```
    x = x(1:end-2);
    y = y(1:end-2);

elseif (series == 2)
    if (slice <= 15)
        tmp = find(x > 2);
    elseif (slice <= 15)
        tmp = find(x > 5);
    elseif (slice <= 30)
        tmp = find(x > 8);
    elseif (slice <= 40)
        tmp = find(x > 8);
    elseif (slice <= 50)
        tmp = find(x > 12);
    elseif (slice <= 60)
        tmp = find(x > 14);
    elseif (slice <= 80)
        tmp = find(x > 22);
    else
        tmp = find(x > 20);
    end

    x = x(tmp);
    y = y(tmp);
end

x = x(1:end-2);
y = y(1:end-2);

fitType = 2;  % = 0 Gaussian,  = 1 Poisson,  = 2 Log Normal

if fitType == 0
    % Gaussian
    p0 = [100, 100, 250];
    f = @(p,xdata)p(3).*exp(-(xdata - p(1)).^2 / (2*p(2)));
    [p, resnorm, residual, exitflag] = lsqcurvefit(f, p0, x, y);
    fit_y = f(p, x);

    saveData(series).fitLoc(slice) = p(1);

elseif fitType == 1
    % Poisson
    p0 = [10, 150];

    options = optimset('MaxFunEvals',3000,'MaxIter',2000,'TolFun',1e-7);

    f = @(p,xdata)p(2).*p(1).^xdata.*exp(-p(1))./factorial(xdata);

    decimateRate = 10;
    q = decimate(y,decimateRate); q(find(q < 1)) = 1;
    t = [1:1:length(q)];
    [p, resnorm, residual, exitflag] = lsqcurvefit(f, p0, t, q, [], [], options);

    fit_y = f(p,t);

    figure;plot(t,q,'bo');hold on;plot(t,fit_y,'r-');

    saveData(series).fitLoc(slice) = p(1);

elseif fitType == 2
    % Log normal
    p0 = [4, 0.8, 10000];
    options = optimset('MaxFunEvals',3000,'MaxIter',2000,'TolFun',1e-7);
    f = @(p,xdata)p(3).*exp(-((log(xdata) - p(1))./(sqrt(2)*p(2))).^2) ./ ...
             (xdata .* p(2) * sqrt(2*pi));
    [p, resnorm, residual, exitflag] = lsqcurvefit(f, p0, x, y, [], [], options);
    fit_y = f(p, x);

    [max_y, max_i] = max(fit_y);
    saveData(series).fitLoc(slice) = x(max_i);

    tmp = find(fit_y(max_i:end) <= max_y/2);
    saveData(series).fitSpread(slice) = x(tmp(1) + max_i - 1) - x(max_i);

end


plot(x, fit_y,'k-', 'LineWidth', lineWidth);
title(['Series ',num2str(series),', Slice ',num2str(slice),', \mu = ', ...
       num2str(saveData(series).fitLoc(slice))],'FontSize',fontSize,'FontWeight','bold');
xlabel(['Void Distance (nm)'],'FontSize',fontSize,'FontWeight','bold');
```

```
        ylabel(['Count (-)'],'FontSize',fontSize,'FontWeight','bold');
        set(gca,'LineWidth',lineWidth,'FontSize',fontSize,'FontWeight','bold');
        if (series == 2)
            ymax = 500;
        elseif (series == 1)
            ymax = 200;
        else
            ymax = ceil(saveData(series).peakVal(slice)/50)*50;
        end
        axis([0,blackBinMax(series),0,ymax]);
        %legend('x','y','avg','filt',-1);
        print('-dtiff',[currOutputDir, outputFileName_void, '_', ...
                  prefix, num2str(slice), suffix]);
        close


        fid = fopen([currOutputDir,'\Series',num2str(series),'_Slice', ...
                        num2str(slice),'_voidSpacingHistograms.txt'],'w');
        fprintf(fid,'Series, %d, Slice, %d, \n',series,slice);
        fprintf(fid,'Thickness, %6.4f \n',saveData(series).currThickness(slice) );
        fprintf(fid,'Peak Void Distance, %6.4f \n',saveData(series).fitLoc(slice) );
        fprintf(fid,'Bin, Void Distance, Fit Distance \n');
        for count = 1:length(x)
            fprintf(fid,'%d, %d, %d \n', x(count), round(y(count)), round(fit_y(count)));
        end
        fclose(fid);

    end
end



fid = fopen([outputDir, 'SummaryData.txt'],'w');
fprintf(fid,'Series, Slice, Thickness (nm), Void RMS x, Void Mean x, Void Peak x,', ...
            'Void RMS y, Void Mean y, Void Peak y, Void RMS, Void Mean, Void Peak, Fit Void Peak \n');
for series = useSeries
    for slice = (seriesSelection(series).selection + scans.firstImage(series))

        disp(['Series ',num2str(series),', Slice ',num2str(slice), ...
               ', Void Distance = ',num2str(saveData(series).rmsLoc(slice)),' rms, ', ...
               num2str(saveData(series).meanLoc(slice)),' mean, ', ...
               num2str(saveData(series).peakLoc(slice)),' peak']);

        fprintf(fid, '%d, %d, %6.2f,   %6.2f, %6.2f, %6.2f, %6.2f, %6.2f, %6.2f, %6.2f,', ...
                      '%6.2f, %6.2f, %6.2f, %6.2f, %6.2f, %6.2f, \n', ...
                      series, slice, saveData(series).currThickness(slice), ...
                      saveData(series).rmsLoc_x(slice), saveData(series).meanLoc_x(slice), ...
                      saveData(series).peakLoc_x(slice), saveData(series).stdDev_x(slice), ...
                      saveData(series).rmsLoc_y(slice), saveData(series).meanLoc_y(slice), ...
                      saveData(series).peakLoc_y(slice), saveData(series).stdDev_y(slice), ...
                      saveData(series).rmsLoc(slice), saveData(series).meanLoc(slice), ...
                      saveData(series).peakLoc(slice), saveData(series).stdDev(slice), ...
                      saveData(series).fitLoc(slice), saveData(series).fitSpread(slice),...
                      );
    end
end
fclose(fid);

disp('Done');

% ================================================================================

function [blackDistances, meanDistance] = findBlackDistances(edgeImage)
    whiteToBlack = 1;
    blackToWhite = 2;

    numRows = size(edgeImage,1);
    numCols = size(edgeImage,2);

    % Track down regions of void (black) between columns (white)
    blackDistances = [];

    for currRow = 1:numRows

        whiteToBlackTransition = find(edgeImage(currRow,:) == whiteToBlack);
        blackToWhiteTransition = find(edgeImage(currRow,:) == blackToWhite);

        if (isempty(whiteToBlackTransition) || isempty(blackToWhiteTransition))
        else
            legitBlackToWhiteTransition = find(whiteToBlackTransition(1) < blackToWhiteTransition);
```

```
            if (isempty(legitBlackToWhiteTransition))
                % There is no white to black transition and then back again
                whiteToBlackTransition = [];
                blackToWhiteTransition = [];
            else
                % There is at least one white to black transition
                blackToWhiteTransition = blackToWhiteTransition(legitBlackToWhiteTransition);

                legitWhiteToBlackTransition = find(blackToWhiteTransition(end) > whiteToBlackTransition);

                if (isempty(legitWhiteToBlackTransition))
                    % There is no return from black to white
                    whiteToBlackTransition = [];
                    blackToWhiteTransition = [];
                else
                    whiteToBlackTransition = whiteToBlackTransition(legitWhiteToBlackTransition);
                end
            end

            currBlackDistance = blackToWhiteTransition - whiteToBlackTransition;
            blackDistances = [blackDistances currBlackDistance];

            meanDistance(currRow) = sqrt(sum(currBlackDistance.^2)/length(currBlackDistance));
        end
    end


% ============================================================================================

% Code to equalize the contrast across a single image

function [contImage stretchMask filtMask] = adjContrast(origImage, nSquares, filterCoeff)
    row = 1;
    col = 2;


    binStep = 1;
    binMin = 0;
    binMax = 255;
    bins = [binMin:binStep:binMax];


    % Divide the image up into n x n tiles
    col_size = size(origImage,2);
    row_size = size(origImage,1);

    col_pixels = floor(col_size/nSquares)*ones(1,nSquares);
    row_pixels = floor(row_size/nSquares)*ones(1,nSquares);

    col_pixels(end) = col_pixels(end) + (col_size - sum(col_pixels));
    row_pixels(end) = row_pixels(end) + (row_size - sum(row_pixels));

    col_start = [1 (1+cumsum(col_pixels(1:end-1)))];
    row_start = [1 (1+cumsum(row_pixels(1:end-1)))];

    col_end = cumsum(col_pixels);
    row_end = cumsum(row_pixels);

    % Locate effective end of histogram
    percentDown = 0.5;
    [h_orig] = hist(origImage, bins);
    sum_h_orig = sum(h_orig,2);
    [max_h_orig, peakPixel] = max(sum_h_orig);
    tmp = find(sum_h_orig(peakPixel:end) < max_h_orig * percentDown/100);
    maxPixel = tmp(1) + peakPixel - 1;


    % Also stretch image to maximize dynamic range
    adjImage = (255 / maxPixel) * double(origImage) + 1;
    b = ones(1,(filterCoeff+1));
    a = (filterCoeff+1)*[1 zeros(1,filterCoeff)];
    filtImage = filtfilt(b,a,double(adjImage));
    filtImage = uint8((filtfilt(b,a,filtImage')'));
    adjImage = filtImage;


    % Determine black to white threshold of each tile
    % Rqualize tile contrast to average threshold
    if (nSquares < 2)
        contImage = uint8(adjImage);
    else
```

```
        for row_square = 1:nSquares
            for col_square = 1:nSquares
                squareImage = adjImage(row_start(row_square):row_end(row_square), ...
                                       col_start(col_square):col_end(col_square));
                level(row_square, col_square) = graythresh(uint8(squareImage));
                threshold(row_square, col_square) = (level(row_square, col_square) * 255);
            end
        end

        % Determine contrast stretch required for that square
        mean_threshold = sum(sum(threshold))/(nSquares^2);

        for row_square = 1:nSquares
            for col_square = 1:nSquares
                squareImage = adjImage(row_start(row_square):row_end(row_square), ...
                                       col_start(col_square):col_end(col_square));

                [h] = hist(squareImage, bins);
                curr_histFreq = sum(h,2);

                % This effectively stretches the two peaks to roughly the
                % locations in the mean image which preserves the density
                stretch = mean_threshold / threshold(row_square, col_square);

                % The tile is brought to the mean contrast level
                transImage = stretch * double(squareImage);
                contImage(row_start(row_square):row_end(row_square), ...
                          col_start(col_square):col_end(col_square)) = transImage;
            end
        end

        % Calculate the multiplier that eac pixel had to change by in order to equalize all tiles
        stretchMask = double(contImage) ./ double(adjImage);
        stretchMask(find(isinf(stretchMask))) = stretchMask(find(isinf(stretchMask))-1);
        stretchMask(find(isnan(stretchMask))) = stretchMask(find(isnan(stretchMask))-1);


        % Need to filter the stretch mask in x AND y directions
        divideBy = 2;
        defFilter = filterCoeff; %11;
        if (nSquares < 2)
            filterCoeff = defFilter;
        else
            filterCoeff = max(defFilter, min(min(col_pixels),min(row_pixels))/divideBy);
        end

        b = ones(1,(filterCoeff+1));
        a = (filterCoeff+1)*[1 zeros(1,filterCoeff)];
        filtMask = filtfilt(b,a,stretchMask);
        filtMask = (filtfilt(b,a,filtMask'))';

        % Apply filtered contrast mask to the original image
        stretchImage = double(adjImage) .* filtMask;

        % Stretch contrast of image
        contImage = uint8(255 * double((stretchImage - min(min(stretchImage))) / ...
                          (max(max(stretchImage)) - min(min(stretchImage)))));

    end


% ================================================================================================

% Code to plot summary plots of inter-column void distances

close all
clear all
clc

warning off all

fontSize = 20;
lineWidth = 2;
markerSize = 10;
colours = 'brmgckbrmgckbrmgckbrmgck';
shapes = 'svodx>+hsvodx>+hsvodx>';
lineStyles = {'-','--',':','-.','-','--',':','-.','-','--',':','-.','-','--',':','-.'};


baseDir = ['C:\My_Documents\Research\Research_Areas\ThreeDimGLAD_with_FIB\'];
directories = { [baseDir,'Processed_FIB_Scans\InterColumnSpacing\June09']};
```

```
outputDir = [baseDir,'Processed_FIB_Scans\InterColumnSpacing\'];
fileName = 'SummaryData.txt';



disp('Reading in data');
for count = 1:length(directories)
    [summary(count)] = readSummaryFile([char(directories{count}),'\',fileName]);
end

% Just use thicknesses as is, compensate for tilt later
for count = 1:length(directories)
    summary(count).upright_thickness = summary(count).thickness ;
end

% Fit curve to mean void distance
all_thicknesses = [];
all_voidDistances = [];
all_voidStdDev = [];
for count = 1:length(directories)
    all_thicknesses = [all_thicknesses summary(count).upright_thickness];
    all_voidDistances = [all_voidDistances summary(count).voidDist_mean];
    all_voidStdDev = [all_voidStdDev summary(count).voidDist_stddev];
end




% Fit p-scaling curve to void spacing curve
disp('Fitting void spacing curve');
first_point_nm = 0;
last_point_nm = 4400;
x_fit = 1:1:5000;
w_o = 10;
h_p = 0.35;
f = @(p,xdata)p(1)*xdata.^p(2);

[x,i] = sort(all_thicknesses);
y = all_voidDistances(i);
tmp = find(x > first_point_nm);
x = x(tmp);
y = y(tmp);
tmp = find(x < last_point_nm);
x = x(tmp);
y = y(tmp);

[p, resnorm, residual, exitflag] = lsqcurvefit(f, [w_o, h_p], x, y);
w_o = p(1);
h_p = p(2);

disp([' ... w_o = ',num2str(w_o),', p = ',num2str(h_p)]);


% Fit p-scaling curve to void spacing std dev curve
disp('Fitting void spacing std dev curve');
first_point_nm = 0;
last_point_nm = 4400;
x_fit = 1:1:5000;
w_s = 10;
h_s = 0.35;
f = @(p,xdata)p(1)*xdata.^p(2);

[x,i] = sort(all_thicknesses);
y = all_voidStdDev(i);
tmp = find(x > first_point_nm);
x = x(tmp);
y = y(tmp);
tmp = find(x < last_point_nm);
x = x(tmp);
y = y(tmp);

[p, resnorm, residual, exitflag] = lsqcurvefit(f, [w_s, h_s], x, y);
w_s = p(1);
h_s = p(2);

disp([' ... w_s = ',num2str(w_s),', p = ',num2str(h_s)]);


for count = 1:length(directories)
    for count2 = 1:length(summary(count).series)
```

```
        currSeries = summary(count).series(count2);
        currSlice = summary(count).slice(count2);
        curr_thickness = summary(count).upright_thickness(count2);
        curr_voidDist = summary(count).voidDist_mean(count2);
        curr_stdDev = summary(count).voidDist_stddev(count2);

        seriesData(currSeries).slice(currSlice) = currSlice;
        seriesData(currSeries).thickness(currSlice) = curr_thickness;
        seriesData(currSeries).voidDist(currSlice) = curr_voidDist;
        seriesData(currSeries).voidStdDev(currSlice) = curr_stdDev;

    end
end
numSeries = length(seriesData);


figure;
leg = [];
counter = 0;
stepSize = 1;
for count = 1:numSeries
    legEntry = [' ',num2str(count)];
    leg = [leg; legEntry];
    plot(seriesData(count).thickness(1:stepSize:end), seriesData(count).voidDist(1:stepSize:end), ...
            ['k', shapes(count)], 'LineWidth', lineWidth, 'MarkerSize', markerSize, ...
            'MarkerFaceColor', 'w', 'MarkerEdgeColor', 'k');
    hold on;
end
plot(seriesData(2).thickness(1:10), seriesData(2).voidDist(1:10), ['k', shapes(2)], ...
         'LineWidth', lineWidth, 'MarkerSize', markerSize, 'MarkerFaceColor', 'w', ...
         'MarkerEdgeColor', 'k');
plot(x_fit, f([w_o, h_p], x_fit), 'k--', 'LineWidth', lineWidth);
plot(x_fit, f([w_o, h_p], x_fit)+f([w_s, h_s], x_fit), 'k-.', 'LineWidth', lineWidth);
plot(x_fit, f([w_o, h_p], x_fit)-f([w_s, h_s], x_fit), 'k-.', 'LineWidth', lineWidth);
xlabel('Thickness (nm)','FontSize',fontSize,'FontWeight','bold');
ylabel('Mean Void Distance (nm)','FontSize',fontSize,'FontWeight','bold');
set(gca,'LineWidth',lineWidth,'FontSize',fontSize,'FontWeight','bold');
legend(leg, -1);
maxDistance1 = 300;
axis([0 5000 0 maxDistance1]);
print('-dtiff',[outputDir, 'SummaryVoidMeanLoc_bw_with_stddev.tiff']);


close all;
disp('Done');
```

C.7 Code to Analyze Bifurcation of Individual Columns

```
% Stage 1: take user-selected first image and find it in the first slice image that
%          the user selects.  Then work backwards (i.e. towards the top of the film)
%          finding the corresponding column

clear all;
close all;
clc;

warning off all

fontSize = 20;
lineWidth = 2;
markerSize = 8;


homework = 1;

switch homework
    case 1
        beginDir = ['C:\My_Documents\'];
    case 2
        beginDir = ['H:\private\My Documents\'];
end


inputDir = [beginDir,'Research\Research_Areas\ThreeDimGLAD_with_FIB\Processed_FIB_Scans\', ...
                        'Bifurcation\Bifurcation_June09_stitch_6_5_3\MultiColumnEdittedImages'];
```

```
parameterFile = [beginDir,'Research\Research_Areas\ThreeDimGLAD_with_FIB\Matlab\', ...
                         'Process_FIB_Slices_TopDownView\FIBscan_June09_TiO2VP5um_stitch.m'];


outputDir = [beginDir,'Research\Research_Areas\ThreeDimGLAD_with_FIB\Processed_FIB_Scans\', ...
                      'Bifurcation\SelectedColumn1'];
firstShapeName = 'shape1.tiff';
selected.images = [1:241];
firstRegion.row = [548, 697];
firstRegion.col = [491, 610];

% outputDir = [beginDir,'Research\Research_Areas\ThreeDimGLAD_with_FIB\Processed_FIB_Scans\', ...
                        'Bifurcation\SelectedColumn2_test'];
% firstShapeName = 'shape2.tiff';
% selected.images = [1:241];
% firstRegion.row = [433, 542];
% firstRegion.col = [743, 842];

%outputDir = [beginDir,'Research\Research_Areas\ThreeDimGLAD_with_FIB\Processed_FIB_Scans\', ...
                        'Bifurcation\SelectedColumn3'];
%firstShapeName = 'shape3.tiff';
%selected.images = [1:241];
%firstRegion.row = [340, 459];
%firstRegion.col = [604, 743];

% outputDir = [beginDir,'Research\Research_Areas\ThreeDimGLAD_with_FIB\Processed_FIB_Scans\', ...
                        'Bifurcation\SelectedColumn4'];
% firstShapeName = 'shape4.tiff';
% selected.images = [1:241];
% firstRegion.row = [460, 609 ];
% firstRegion.col = [576, 725];

filePrefix = 'currImage_';
fileSuffix = '.tiff';


shiftMax = 1;
thresholdMultiplier = 0.4;
fillFactor = 3;

verbose = 0;

mkdir(outputDir);

% Get scan parameters
disp('Getting scan parameters');
run(parameterFile);

disp('Reading in image files');
counter = 0;

fileNames = dir([inputDir,'\',filePrefix,'*',fileSuffix]);

for count2 = length(fileNames):-1:1
    currFile = fileNames(count2).name;
    tmp = findstr(currFile,'_');

    tmp2 = findstr(currFile,fileSuffix);
    slice = str2num(currFile(tmp(1)+1:tmp2-1));

    if (find(selected.images == slice))
        counter = counter + 1;
        disp(['    ... Counter ',num2str(counter),', Slice ',num2str(slice)]);

        % Read in the input image
        currImage = double(imread([inputDir,'\',currFile]));

        if (counter == 1)
            currCoords = firstRegion;
            matchShape = double(imread([inputDir,'\',firstShapeName]));
            matchShape = squeeze(matchShape(:,:,1));

            level = graythresh(uint8(matchShape));
            threshImage = im2bw(uint8(matchShape),level);
            [foundShapes, numShapes] = bwlabel(threshImage,4);
            for shapeCount = 1:numShapes
                numWhites(shapeCount) = sum(sum(foundShapes == shapeCount));
            end
            [maxVal, maxIndex] = max(numWhites);
            currMask = imfill(foundShapes == maxIndex,'holes');
            matchImage = zeros(size(matchShape));
```

```
            matchImage(find(currMask)) = matchShape(find(currMask));

            imwrite(uint8(matchImage),[outputDir,'\selectedImage_', ...
                                       getSuffix(counter),'.tiff'],'tiff');

        else
            [newMatchImage, newMask, newCoords, reqShift, diffImage] = ...
                        findMatchImage(currImage, matchImage, currMask, ...
                                       currCoords, shiftMax, thresholdMultiplier, ...
                                       fillFactor, verbose);
            imwrite(uint8(newMatchImage),[outputDir,'\selectedImage_', ...
                                          getSuffix(counter),'.tiff'],'tiff');
            imwrite(uint8(diffImage),[outputDir,'\diffImage_', ...
                                          getSuffix(counter),'.tiff'],'tiff');
            currCoords = newCoords;
            matchImage = newMatchImage;
            currMask = newMask;
            clear newMatchImage;
            disp(['       ... reqShift = (',num2str(reqShift.row),',',...
                      num2str(reqShift.col),')']);
        end
    end
end

disp('Done');

% ================================================================================

function [newMatchImage, newMask, newCoords, reqShift, diffPixels] = ...
    findMatchImage(currImage, matchImage, currMask, currCoords, ...
                   shiftMax, multiplier, fillFactor, verbose)

    size_currImage.row = size(currImage,1);
    size_currImage.col = size(currImage,2);
    size_matchImage.row = size(matchImage,1);
    size_matchImage.col = size(matchImage,2);

    newMatchImage = zeros(size(matchImage));
    newCoords.row = [0,0];
    newCoords.col = [0,0];
    currDiffImage = 9999999999;

    if (verbose)
        disp(['       ... currCoords = (', ...
           num2str(currCoords.row(1)),':',num2str(currCoords.row(2)),',', ...
           num2str(currCoords.col(1)),':',num2str(currCoords.col(2)),')']);
    end

    for shift_row = -shiftMax:1:shiftMax
        for shift_col = -shiftMax:1:shiftMax

            padding = 10;

            start_row = max(1, (currCoords.row(1) + shift_row - padding));
            end_row = min(size_currImage.row, ...
                          (currCoords.row(2) + shift_row + padding));
            start_col = max(1, (currCoords.col(1) + shift_col - padding));
            end_col = min(size_currImage.col, ...
                           (currCoords.col(2) + shift_col + padding));
            selectImage = currImage(start_row:end_row, start_col:end_col);


            level = graythresh(uint8(selectImage))* multiplier;
            threshImage = im2bw(uint8(selectImage),level);
            [foundShapes, numShapes] = bwlabel(threshImage,8);

            selectMatch = matchImage;
            lessRows = size_matchImage.row - (end_row - start_row + 1);
            lessCols = size_matchImage.col - (end_col - start_col + 1);

            if (lessRows > 0)
                % Need to shrink match image
                selectMatch = matchImage(1:(end-lessRows),:);
                selectMask = currMask(1:(end-lessRows),:);
            elseif (lessRows < 0)
                % Need to pad match image
                leftSide = round(-lessRows/2);
                rightSide = -lessRows - leftSide;
                selectMatch = [zeros(leftSide,size(matchImage,2)); ...
                               matchImage; zeros(rightSide,size(matchImage,2))];
                selectMask = [zeros(leftSide,size(matchImage,2));  ...
```

```
                    currMask; zeros(rightSide,size(matchImage,2))];
else
    selectMatch = matchImage;
    selectMask = currMask;
end
if (lessCols > 0)
    % Need to shrink match image
    selectMatch = selectMatch(:,1:(end-lessCols));
    selectMask = selectMask(:,1:(end-lessCols));
elseif (lessCols < 0)
    % Need to pad match image
    leftSide = round(-lessCols/2);
    rightSide = -lessCols - leftSide;
    selectMatch = [zeros(size(selectMatch,1),leftSide), ...
                      selectMatch, zeros(size(selectMatch,1),rightSide)];
    selectMask = [zeros(size(selectMask,1),leftSide),  ...
                      selectMask, zeros(size(selectMask,1),rightSide)];
end



numWhites_mask = sum(sum(selectMask));

percentClose = 50;%30;

for count = 1:numShapes

    useShape = logical(imfill(imdilate(foundShapes == ...
                               count,strel('disk',fillFactor,0)),'holes'));
    numWhites = sum(sum(useShape));

    if (((numWhites* (1 - percentClose/100)) < numWhites_mask) && ...
              ((numWhites* (1 + percentClose/100)) > numWhites_mask))

        % Shapes are similiar in size
        commonMask = useShape;  %imfill(imdilate(uint8(useShape) , se), 'holes');
        useImage = zeros(size(selectImage));
        useImage(find(commonMask)) = selectImage(find(commonMask));

        diffPixels = useImage - selectMatch;

        diffImage = sum(sum(abs(diffPixels)));

        if (verbose)
            disp(['       ... shift (',num2str(shift_row),',', ...
                        num2str(shift_col),'), shape = ',num2str(count)]);
            disp(['           ... lessRows = ',num2str(lessRows), ...
                        ', lessCols = ',num2str(lessCols), ...
                            ', diffImage = ',num2str(diffImage)]);
            disp(['            ... currCoords = (',num2str(start_row), ...
                        ':',num2str(end_row),',', ...
                        num2str(start_col),':',num2str(end_col),')']);
        end


        if (diffImage < currDiffImage)
            if (verbose)
                disp(['           ****']);
            end
            currDiffImage = diffImage;

            %figure;image(diffPixels);colormap('gray')

            % Center new shape
            [pixel_row, pixel_col] = find(useImage);
            minRow_val = min(pixel_row);
            maxRow_val = max(pixel_row);
            minCol_val = min(pixel_col);
            maxCol_val = max(pixel_col);

            row_offset = round(size_matchImage.row/2 - (maxRow_val + minRow_val)/2);
            col_offset = round(size_matchImage.col/2 - (maxCol_val + minCol_val)/2);


            newMatchImage = useImage(minRow_val:maxRow_val, minCol_val:maxCol_val);
            newMask = commonMask(minRow_val:maxRow_val, minCol_val:maxCol_val);

            reqShift.row = shift_row;
            reqShift.col = shift_col;

            padding = 2;
```

```
                        newCoords.row = [min(size_currImage.row,  ...
                                                max(1,(start_row + minRow_val - 1 - padding))),  ...
                                         min(size_currImage.row, ...
                                                max(1,(start_row + maxRow_val - 1 + padding)))];
                        newCoords.col = [min(size_currImage.col,  ...
                                                max(1,(start_col + minCol_val - 1 - padding))),  ...
                                         min(size_currImage.col, ...
                                                max(1,(start_col + maxCol_val - 1 + padding)))];
                    end
                end
            end
        end
    end


% ============================================================================

% Stage 2: Look at isolated columns and determine whether there are any
% branches/arms for the column in a given image slice.  Calculate the diameter
% of each branch and store this information

clear all;
close all;
clc;

warning off all

fontSize = 20;
lineWidth = 2;
markerSize = 8;


homework = 1;
shape = 1;

switch homework
    case 1
        beginDir = ['C:\My_Documents\'];
    case 2
        beginDir = ['H:\private\My Documents\'];
end


switch shape
    case 1

        origDir = [beginDir,'Research\Research_Areas\ThreeDimGLAD_with_FIB\Processed_FIB_Scans\', ...
                           'Bifurcation\Bifurcation_June09_stitch_6_5_3\SelectedColumn1'];
        inputDir = [beginDir,'Research\Research_Areas\ThreeDimGLAD_with_FIB\Processed_FIB_Scans\', ...
                            'Bifurcation\Bifurcation_June09_stitch_6_5_3\Arms1'];
        outputDir = [beginDir,'Research\Research_Areas\ThreeDimGLAD_with_FIB\Processed_FIB_Scans\', ...
                             'Bifurcation\Bifurcation_June09_stitch_6_5_3\ProcessedColumn1'];
        firstLevel = 0.247;
        pixel_resolution_nm = 0.86;   % Image resolution from ImageJ comparison of
                                      % first image with TiO2_GLAD_062 of SliceSeries 3
        percentLimit = 3;      % Limit for recording area of branch

        % Slice resolutions for Series 3,5,6
        slice_resolution_nm = [ones(1, 113)*12.06, ones(1,71)*7.87, ones(1,57)*5.88];
        % Need to bring threshold back to optimal level (used 0.4 to 0.6 in stage1)
        shapeMultiplierNominal = 1.3;
        multiplierOffset = 0.3;
        shapeMultiplier = [ones(1, 113)*(shapeMultiplierNominal-multiplierOffset), ...
                           ones(1,71)*shapeMultiplierNominal, ones(1,57)*shapeMultiplierNominal];
        thickness_nm = cumsum(slice_resolution_nm);
        % Series 6 is 796nm above substrate + 6 slices offset * 5.88nm resolution
        thickness_nm = thickness_nm(end) - thickness_nm + 831;

    case 2

        origDir = [beginDir,'Research\Research_Areas\ThreeDimGLAD_with_FIB\Processed_FIB_Scans\', ...
                           'Bifurcation\Bifurcation_June09_stitch_6_5_3\SelectedColumn2'];
        inputDir = [beginDir,'Research\Research_Areas\ThreeDimGLAD_with_FIB\Processed_FIB_Scans\', ...
                            'Bifurcation\Bifurcation_June09_stitch_6_5_3\Arms2'];
        outputDir = [beginDir,'Research\Research_Areas\ThreeDimGLAD_with_FIB\Processed_FIB_Scans\', ...
                             'Bifurcation\Bifurcation_June09_stitch_6_5_3\ProcessedColumn2'];
        firstLevel = 0.52;
        pixel_resolution_nm = 0.86;   % Image resolution for slice series 6
        percentLimit = 3;      % Limit for recording area of branch
```

```
        slice_resolution_nm = [ones(1,57)*5.88, ones(1,71)*7.87, ones(1,113)*12.06];
        shapeMultiplierNominal = 1.6;              % Need to bring threshold back to
                                                   % optimal level (used 0.4 to 0.6 in stage1)
        multiplierOffset = 0.3;
        shapeMultiplier = [ones(1,57)*shapeMultiplierNominal, ...
                           ones(1,71)*shapeMultiplierNominal, ...
                           ones(1, 113)*(shapeMultiplierNominal-multiplierOffset)];
        % Series 6 is 796nm above substrate + 6 slices offset * 5.88nm resolution
        thickness_nm = cumsum(slice_resolution_nm) + 831;


    case 3

        origDir = [beginDir,'Research\Research_Areas\ThreeDimGLAD_with_FIB\Processed_FIB_Scans\', ...
                           'Bifurcation\Bifurcation_June09_stitch_6_5_3\SelectedColumn3'];
        inputDir = [beginDir,'Research\Research_Areas\ThreeDimGLAD_with_FIB\', ...
                           'Processed_FIB_Scans\Bifurcation\Bifurcation_June09_stitch_6_5_3\Arms3'];
        outputDir = [beginDir,'Research\Research_Areas\ThreeDimGLAD_with_FIB\Processed_FIB_Scans\', ...
                           'Bifurcation\Bifurcation_June09_stitch_6_5_3\ProcessedColumn3'];
        shapeMultiplierNominal = 1.3;              % Need to bring threshold back to
                                                   % optimal level (used 0.4 to 0.6 in stage1)
        multiplierOffset = 0.3;
        shapeMultiplier = [ones(1,57)*shapeMultiplierNominal, ...
                           ones(1,71)*shapeMultiplierNominal, ...
                           ones(1, 113)*(shapeMultiplierNominal-multiplierOffset)];
        firstLevel = 0.41;
        pixel_resolution_nm = 0.86;   % Image resolution for slice series 6
        percentLimit = 3;      % Limit for recording area of branch
        slice_resolution_nm = [ones(1,57)*5.88, ones(1,71)*7.87, ones(1,113)*12.06];
        % Series 6 is 796nm above substrate + 6 slices offset * 5.88nm resolution
        thickness_nm = cumsum(slice_resolution_nm) + 831;

    case 4

        origDir = [beginDir,'Research\Research_Areas\ThreeDimGLAD_with_FIB\Processed_FIB_Scans\', ...
                           'Bifurcation\Bifurcation_June09_stitch_6_5_3\SelectedColumn4'];
        inputDir = [beginDir,'Research\Research_Areas\ThreeDimGLAD_with_FIB\Processed_FIB_Scans\', ...
                           'Bifurcation\Bifurcation_June09_stitch_6_5_3\Arms4'];
        outputDir = [beginDir,'Research\Research_Areas\ThreeDimGLAD_with_FIB\Processed_FIB_Scans\', ...
                           'Bifurcation\Bifurcation_June09_stitch_6_5_3\ProcessedColumn4'];
        shapeMultiplierNominal = 1.3;              % Need to bring threshold back to
                                                   % optimal level (used 0.4 to 0.6 in stage1)
        multiplierOffset = 0.3;
        shapeMultiplier = [ones(1,57)*shapeMultiplierNominal, ...
                                   ones(1,71)*shapeMultiplierNominal, ...
                                   ones(1, 113)*(shapeMultiplierNominal-multiplierOffset)];
        firstLevel = 0.41;
        pixel_resolution_nm = 0.86;   % Image resolution for slice series 6
        percentLimit = 3;      % Limit for recording area of branch
        slice_resolution_nm = [ones(1,57)*5.88, ones(1,71)*7.87, ones(1,113)*12.06];
        thickness_nm = cumsum(slice_resolution_nm) + 831;   % Series 6 is 796nm above substrate
                                                            % + 6 slices offset * 5.88nm resolution

end

filePrefix = 'selectedImage_';
fileSuffix = '.tiff';



verbose = 0;

mkdir(outputDir);

fileNames = dir([inputDir,'\',filePrefix,'*',fileSuffix]);

fid = fopen([outputDir,'\geometry.txt'],'w');
fprintf(fid, 'File, Thickness (nm), Column Area (pixels), Column Area (nm^2), ', ...
             'Column Diameter (nm), Surface Area (pixels), Surface Area (nm), ', ...
             'Shape Count, Quadrant, Area (pixels), Area (nm^2), Diameter (nm), ', ...
             'Percent Area, Center Row, Center Col \n');


disp('Reading in input files ...');
for count = 1:1:length(fileNames)
    currFile = fileNames(count).name;
    tmp = findstr(currFile,'_');
    tmp2 = findstr(currFile,fileSuffix);
    slice = str2num(currFile(tmp(1)+1:tmp2(1)-1));

    disp([' ... Slice: ',num2str(slice)]);
```

```
% Shape un-subdivided
origImage = double(imread([origDir,'\',currFile]));
origImage = squeeze(origImage(:,:,1));

% Threshold un-subdivided image
if (count == 1)
    level = firstLevel;
else
    level = graythresh(uint8(origImage)) * shapeMultiplier(count);
end

disp(['    ... level = ',num2str(level)]);
threshImage = im2bw(uint8(origImage),level);

% Edge detection
disp(['    ... edge detection']);
[edgeImage, edgeImage_x, edgeImage_y] = findEdgesImage(threshImage);

surfaceArea = sum(sum(edgeImage));
surfaceArea_abs_nm = surfaceArea * pixel_resolution_nm;


% Shape subdivided
currImage = double(imread([inputDir,'\',currFile]));
currImage = squeeze(currImage(:,:,1));


% Threshold subdivided image
if (count == 1)
    level = firstLevel;
else
    level = graythresh(uint8(currImage)) * shapeMultiplier(count);
end

threshImage = im2bw(uint8(currImage),level);
[foundShapes, numShapes] = bwlabel(threshImage,4);

maskedImage = currImage .* threshImage;

%figure;imshow(uint8(maskedImage));

disp(['    ... cross-sectional area']);
areaImage = sum(sum(threshImage));
areaImage_nm = round(areaImage * pixel_resolution_nm ^2);


[pixel_row, pixel_col] = find(threshImage);
minRow_val = min(pixel_row);
maxRow_val = max(pixel_row);
minCol_val = min(pixel_col);
maxCol_val = max(pixel_col);
centerImage.row = (maxRow_val + minRow_val)/2;
centerImage.col = (maxCol_val + minCol_val)/2;

meanDiameter = ((maxRow_val - minRow_val) + (maxCol_val - minCol_val))/2;
meanDiameter_nm = round(meanDiameter * pixel_resolution_nm);

fprintf(fid,'%d, %d, %d, %d, %d, %d, %6.3f, ', ...
               count, round(thickness_nm(count)), areaImage, ...
               areaImage_nm, meanDiameter_nm, surfaceArea, surfaceArea_abs_nm);

disp(['    ... evaluating arms']);
counter = 0;
for currShape = 1:numShapes
    currArea = sum(sum(foundShapes == currShape));
    percentArea = (currArea ./ areaImage * 100);
    if ( percentArea > percentLimit)
        counter = counter + 1;
        [pixel_row, pixel_col] = find((foundShapes == currShape));
        minRow_val = min(pixel_row);
        maxRow_val = max(pixel_row);
        minCol_val = min(pixel_col);
        maxCol_val = max(pixel_col);
        centerShape.row = round((maxRow_val + minRow_val)/2);
        centerShape.col = round((maxCol_val + minCol_val)/2);
        delta.row = centerImage.row - centerShape.row;
        delta.col = centerImage.col - centerShape.col;
        if ((delta.row > 1) && (delta.col > 1))
            quadrant(counter) = 2;
        elseif ((delta.row > 1) && (delta.col < 1))
```

```
            quadrant(counter) = 1;
        elseif ((delta.row < 1) && (delta.col > 1))
            quadrant(counter) = 3;
        else
            quadrant(counter) = 4;
        end

        areaShape(counter) = currArea;
        areaPercent(counter) = round(percentArea);
        areaShape_nm(counter) = round(currArea * pixel_resolution_nm^2);

        meanDiameterShape(counter) = sqrt(areaShape)/pi;
        meanDiameterShape_nm(counter) = round(meanDiameterShape(counter) * ...
                                              pixel_resolution_nm);

        fprintf(fid,'%d, %d, %d, %d, %d, %d, %d, %d, ', ...
                    counter, quadrant(counter), areaShape(counter), ...
                    areaShape_nm(counter), meanDiameterShape_nm(counter), ...
                    areaPercent(counter), centerShape.row, centerShape.col);
    end
end

fprintf(fid,'\n');

disp(['    ... plotting figures']);

bins = 5:5:255;
histImage = sum(hist(currImage, bins),2);
figure('Visible','off');bar(bins(2:end), histImage(2:end));
print('-dtiff',[outputDir, '\hist_', getSuffix(count), fileSuffix]);
figure('Visible','off');
imwrite(uint8(currImage), [outputDir, '\orig_', getSuffix(count), fileSuffix], 'tiff');
figure('Visible','off');
imwrite(uint8(threshImage*255), [outputDir, '\thresh_', getSuffix(count), fileSuffix], 'tiff');
figure('Visible','off');
imwrite(uint8(edgeImage*255), [outputDir, '\edge_', getSuffix(count), fileSuffix], 'tiff');
figure('Visible','off');
imwrite(uint8(maskedImage), [outputDir, '\masked_', getSuffix(count), fileSuffix], 'tiff');
figure('Visible','off');
imwrite(uint8(label2rgb(foundShapes, @jet, 'k')), ...
                        [outputDir, '\shapes_', getSuffix(count), fileSuffix], 'tiff');

close all;

end

fclose(fid);

disp('Done');


% ================================================================================================

% Stage 3: track arms of each column and plot summary figures

clear all;
close all;
clc;


% Point where shape 1 becomes bigger and outshadows shape 4:
%  at thickness = 1800 nm
%  file = 105
% Shape 4 breaks into three separated arms which then fade away


warning off all

fontSize = 20;
lineWidth = 2;
markerSize = 12;

colours = [ 'brmgckbrmgckbrmgckbrmgckbrmgckbrmgckbrmg', ...
            'ckbrmgckbrmgckbrmgckbrmgckbrmgckbrmgckbrm', ...
            'gckbrmgckbrmgckbrmgckbrmgckbrmgckbrmgckb', ...
            'rmgckbrmgckbrmgckbrmgck'];
shapes = ['svodx>+hsvodx>+hsvodx>+hsvodx>+hsvodx>+hsvodx>+', ...
          'hsvodx>+hsvodx>+hsvodx>+hsvodx>+hsvodx>+hsvodx>+', ...
          'hsvodx>+hsvodx>+hsvodx>+hsvodx>+h'];
```

```
homework = 1;

switch homework
    case 1
        beginDir = ['C:\My_Documents\'];
    case 2
        beginDir = ['H:\private\My Documents\'];
end

inputDir = [beginDir,'Research\Research_Areas\ThreeDimGLAD_with_FIB\Processed_FIB_Scans\', ...
                    'Bifurcation\Bifurcation_June09_stitch_6_5_3'];
outputDir = [beginDir,'Research\Research_Areas\ThreeDimGLAD_with_FIB\Processed_FIB_Scans\', ...
                     'Bifurcation\Bifurcation_June09_stitch_6_5_3'];

columns = [1 4 3 2];
filePrefix = 'geometry';
fileExtension = '.csv';

disp('Reading in data files ...');
for currCol = columns
    disp([' ... File ',num2str(currCol)]);
    fileName = [inputDir,'\',filePrefix,num2str(currCol),fileExtension];
    [data] = readFile(fileName);
    columnData(currCol) = data;
    clear data;
end


disp('Tracking arms ...');
dummyVal = 99999;
for currCol = columns
    for currQuad = 1:4
        columnData(currCol).quad(currQuad).arm_diameter_nm = ...
               dummyVal * ones(1,length(columnData(currCol).numArms));
    end
    for currSlice = 1:length(columnData(currCol).numArms)
        for currArm = 1:columnData(currCol).numArms(currSlice)
            if (columnData(currCol).slice(currSlice).shapes(currArm).diameter_nm < ...
                    columnData(currCol).quad(columnData(currCol).slice(currSlice).
                                shapes(currArm).quadrant).arm_diameter_nm(currSlice))
                columnData(currCol).quad(columnData(currCol).slice(currSlice).
                                shapes(currArm).quadrant).arm_diameter_nm(currSlice) = ...
                   columnData(currCol).slice(currSlice).shapes(currArm).diameter_nm;
            end
        end
    end
end

percentClose = 20;
smallDiameter = 100;
smallDiameterDelta = 20;
distanceClose = 20;
for currCol = columns
    disp([' ... column ',num2str(currCol)]);
    activeArms = [];
    newArm = 0;
    for currSlice = 1:length(columnData(currCol).numArms)
        disp(['     ... slice ',num2str(currSlice),', thickness: ', ...
               num2str(columnData(currCol).thickness_nm(currSlice))]);

        nextActiveArms = [];

        for currArm = 1:columnData(currCol).numArms(currSlice)
            disp(['       ... arm ',num2str(currArm), ...
                  ', diameter: ', ...
                  num2str(columnData(currCol).slice(currSlice).shapes(currArm).diameter_nm), ...
                  ', quadrant: ', ...
                  num2str(columnData(currCol).slice(currSlice).shapes(currArm).quadrant), ...
                  ', center: (', ...
                  num2str(columnData(currCol).slice(currSlice).shapes(currArm).center_row), ...
                  ',',num2str(columnData(currCol).slice(currSlice).shapes(currArm).center_col),')']);

            foundMatch = 0;
            matchedArms = [];
            matchedDiameters = [];
            matchedQuadrants = [];
            matchedPercents = [];
            matchedQuadrantDiff = [];
            matchedDist = [];
            for activeArm = activeArms
                numSteps = columnData(currCol).activeArm(activeArm).numSteps
```

```
    pastArmDiameter = columnData(currCol).activeArm(activeArm).arm_diameter_nm(numSteps);
    pastQuadrant = columnData(currCol).activeArm(activeArm).quadrant(numSteps);
    pastCenter_row = columnData(currCol).activeArm(activeArm).center_row(numSteps);
    pastCenter_col = columnData(currCol).activeArm(activeArm).center_col(numSteps);

    currArmDiameter = columnData(currCol).slice(currSlice).shapes(currArm).diameter_nm;
    currQuadrant = columnData(currCol).slice(currSlice).shapes(currArm).quadrant;
    currCenter_row = columnData(currCol).slice(currSlice).shapes(currArm).center_row;
    currCenter_col = columnData(currCol).slice(currSlice).shapes(currArm).center_col;

    percentDiff = abs((currArmDiameter - pastArmDiameter)/currArmDiameter*100);
    distance = sqrt((currCenter_row - pastCenter_row)^2 + ...
                    (currCenter_col - pastCenter_col)^2);
    percentDist = distance / columnData(currCol).columnDiameter_nm(currSlice) * 100;

    if (((percentDiff < percentClose) || ...
         ((currArmDiameter < smallDiameter) && ...
         (abs(currArmDiameter - pastArmDiameter) < smallDiameterDelta))) && ...
        (percentDist < distanceClose))
        foundMatch = 1;
        matchedArms = [matchedArms activeArm];
        matchedDiameters = [matchedDiameters pastArmDiameter];
        matchedQuadrants = [matchedQuadrants pastQuadrant];
        matchedPercents = [matchedPercents percentDiff];
        matchedQuadrantDiff = [matchedQuadrantDiff (currQuadrant == pastQuadrant)];
        matchedDist = [matchedDist percentDist];
    end
end

if (foundMatch)
    tmp = find(matchedQuadrantDiff == 1);
    if (isempty(tmp))
        [minPercentDiff, bestMatch] = min(matchedPercents);
    else
        [minPercentDiff, bestMatch] = min(matchedPercents(tmp));
    end
    columnData(currCol).activeArm(matchedArms(bestMatch)).numSteps = ...
           columnData(currCol).activeArm(matchedArms(bestMatch)).numSteps + 1;
    numSteps = columnData(currCol).activeArm(matchedArms(bestMatch)).numSteps;
    columnData(currCol).activeArm(matchedArms(bestMatch)).thickness_nm(numSteps) = ...
           columnData(currCol).thickness_nm(currSlice);
    columnData(currCol).activeArm(matchedArms(bestMatch)).arm_diameter_nm(numSteps) = ...
           columnData(currCol).slice(currSlice).shapes(currArm).diameter_nm;
    columnData(currCol).activeArm(matchedArms(bestMatch)).quadrant(numSteps) = ...
           columnData(currCol).slice(currSlice).shapes(currArm).quadrant;
    columnData(currCol).activeArm(matchedArms(bestMatch)).center_row(numSteps) = ...
           columnData(currCol).slice(currSlice).shapes(currArm).center_row;
    columnData(currCol).activeArm(matchedArms(bestMatch)).center_col(numSteps) = ...
           columnData(currCol).slice(currSlice).shapes(currArm).center_col;

    % Keep track of active arms
    nextActiveArms = [nextActiveArms matchedArms(bestMatch)];

    disp(['          ... found match ',num2str(matchedArms(bestMatch)), ...
          ', percentDiff: ',num2str(minPercentDiff), ...
          ', numSteps: ', ...
          num2str(columnData(currCol).activeArm(matchedArms(bestMatch)).numSteps), ...
          ', thickness: ', ...
          num2str(columnData(currCol).activeArm(matchedArms(bestMatch)).
                                    thickness_nm(numSteps)), ...
          ', diameter: ', ...
          num2str(columnData(currCol).activeArm(matchedArms(bestMatch)).
                                     arm_diameter_nm(numSteps)), ...
          ', quadrant: ', ...
          num2str(columnData(currCol).activeArm(matchedArms(bestMatch)).
                                     quadrant(numSteps)), ...
          ', center: (', ...
          num2str(columnData(currCol).activeArm(matchedArms(bestMatch)).
                                     center_row(numSteps)),',', ...
          num2str(columnData(currCol).activeArm(matchedArms(bestMatch)).
                                     center_col(numSteps)), ...
          ]);

else
    newArm = newArm + 1;
    nextActiveArms = [nextActiveArms newArm];
    columnData(currCol).activeArm(newArm).numSteps = 1;
    numSteps = columnData(currCol).activeArm(newArm).numSteps;
    columnData(currCol).activeArm(newArm).thickness_nm(numSteps) = ...
                columnData(currCol).thickness_nm(currSlice);
    columnData(currCol).activeArm(newArm).arm_diameter_nm(numSteps) = ...
```

```
                        columnData(currCol).slice(currSlice).shapes(currArm).diameter_nm;
                columnData(currCol).activeArm(newArm).quadrant(numSteps) = ...
                        columnData(currCol).slice(currSlice).shapes(currArm).quadrant;
                columnData(currCol).activeArm(newArm).center_row(numSteps) = ...
                        columnData(currCol).slice(currSlice).shapes(currArm).center_row;
                columnData(currCol).activeArm(newArm).center_col(numSteps) = ...
                        columnData(currCol).slice(currSlice).shapes(currArm).center_col;

                disp(['          ... new arm ',num2str(newArm)]);

            end
        end

        activeArms = nextActiveArms; %(tmp);
    end
end



disp('Plotting data ...');

downsample = 5;
figure;
leg = [];
for currCol = columns
    legEntry = [' ',num2str(currCol)];
    leg = [leg; legEntry];

    plot(columnData(currCol).thickness_nm(1:downsample:end) , ...
           columnData(currCol).columnDiameter_nm(1:downsample:end), ...
           [colours(currCol),shapes(currCol)], 'LineWidth', 2, 'MarkerSize', ...
           markerSize, 'MarkerFaceColor', 'w', 'MarkerEdgeColor', 'k');
    hold on;
end
xlabel('t (nm)','FontSize',fontSize,'FontWeight','bold');
ylabel('w (nm)','FontSize',fontSize,'FontWeight','bold');
set(gca,'LineWidth',lineWidth,'FontSize',fontSize,'FontWeight','bold');
legend(leg, -1);
axis([500 3500 0 350]);
print('-dtiff',[outputDir,'\ColDiameter']);
close;



downsample = 5;
figure;
leg = [];
for currCol = columns
    legEntry = [' ',num2str(currCol)];
    leg = [leg; legEntry];

    plot(columnData(currCol).thickness_nm(1:downsample:end) , ...
           columnData(currCol).file(1:downsample:end), [colours(currCol),shapes(currCol)], ...
           'LineWidth', 2, 'MarkerSize', markerSize, 'MarkerFaceColor', 'w', ...
           'MarkerEdgeColor', 'k');
    hold on;
end
xlabel('t (nm)','FontSize',fontSize,'FontWeight','bold');
ylabel('File','FontSize',fontSize,'FontWeight','bold');
set(gca,'LineWidth',lineWidth,'FontSize',fontSize,'FontWeight','bold');
legend(leg, -1);
print('-dtiff',[outputDir,'\File']);
close;



downsample = 10;
figure;
leg = [];
for currCol = columns
    legEntry = [' ',num2str(currCol)];
    leg = [leg; legEntry];

    plot(columnData(currCol).thickness_nm(1:downsample:end) , ...
           columnData(currCol).cum_surfaceArea_um2(1:downsample:end), ...
           [colours(currCol),shapes(currCol)], 'LineWidth', 2, 'MarkerSize', ...
           markerSize, 'MarkerFaceColor', 'w', 'MarkerEdgeColor', 'k');
    hold on;
end
xlabel('t (nm)','FontSize',fontSize,'FontWeight','bold');
```

```
ylabel(['SA (\mum^2)'],'FontSize',fontSize,'FontWeight','bold');
set(gca,'LineWidth',lineWidth,'FontSize',fontSize,'FontWeight','bold');
legend(leg, -1);
print('-dtiff',[outputDir,'\CumSA']);
close;

downsample = 10;
footprint_area_um = 0.09;
figure;
leg = [];
for currCol = columns
    legEntry = [' ',num2str(currCol)];
    leg = [leg; legEntry];

    plot(columnData(currCol).thickness_nm(1:downsample:end) , ...
            columnData(currCol).cum_surfaceArea_um2(1:downsample:end) / footprint_area_um, ...
            [colours(currCol),shapes(currCol)], 'LineWidth', 2, 'MarkerSize', ...
            markerSize, 'MarkerFaceColor', 'w', 'MarkerEdgeColor', 'k');
    hold on;
end
xlabel('t (nm)','FontSize',fontSize,'FontWeight','bold');
ylabel(['A (\mum^2 \mum^{-2})'],'FontSize',fontSize,'FontWeight','bold');
set(gca,'LineWidth',lineWidth,'FontSize',fontSize,'FontWeight','bold');
legend(leg, -1);
axis([500 3500 0 50]);
print('-dtiff',[outputDir,'\CumSA_rel']);
close;



for currCol = columns

    numArms = columnData(currCol).numArms - 1;
    filtNumArms = numArms; %filtfilt(b,a,numArms);

    figure;
    plot(columnData(currCol).thickness_nm , numArms, ['k',shapes(currCol)], ...
             'LineWidth', 2 , 'MarkerSize', markerSize, 'MarkerFaceColor', ...
             'w', 'MarkerEdgeColor', 'k');
    hold on;
    xlabel('t (nm)','FontSize',fontSize,'FontWeight','bold');
    ylabel('B (# column^{-1})','FontSize',fontSize,'FontWeight','bold');
    set(gca,'LineWidth',lineWidth,'FontSize',fontSize,'FontWeight','bold');
    axis([500 3500 0 7]);
    print('-dtiff',[outputDir,'\Branches_',num2str(currCol)]);
    close;
end

figure;
downsample = 10;
leg = [];
for currCol = columns
    filtCoeff = 3;
    b = ones(1,(filtCoeff+1));
    a = (filtCoeff+1)*[1 zeros(1,filtCoeff)];

    numArms = columnData(currCol).numArms - 1;
    filtNumArms = filtfilt(b,a,numArms);
    plot(columnData(currCol).thickness_nm(1:downsample:end) , ...
            filtNumArms(1:downsample:end), ['k',shapes(currCol)], 'LineWidth', 2, ...
            'MarkerSize', markerSize, 'MarkerFaceColor', 'w', 'MarkerEdgeColor', 'k');
    hold on;
    legEntry = [' ',num2str(currCol)];
    leg = [leg; legEntry];
end
for currCol = columns
    filtCoeff = 3;
    b = ones(1,(filtCoeff+1));
    a = (filtCoeff+1)*[1 zeros(1,filtCoeff)];

    numArms = columnData(currCol).numArms - 1;
    filtNumArms = filtfilt(b,a,numArms);
    plot(columnData(currCol).thickness_nm , filtNumArms, ...
            [colours(currCol), '-'], 'LineWidth', 2);
end
xlabel('t (nm)','FontSize',fontSize,'FontWeight','bold');
ylabel('B (# column^{-1})','FontSize',fontSize,'FontWeight','bold');
set(gca,'LineWidth',lineWidth,'FontSize',fontSize,'FontWeight','bold');
legend(leg, -1);
print('-dtiff',[outputDir,'\Branches']);
close;
```

```
for currCol = columns
    figure;
    for currArm = 1:length(columnData(currCol).activeArm)
        plot_thickness = columnData(currCol).activeArm(currArm).thickness_nm;
        plot_diameter = columnData(currCol).activeArm(currArm).arm_diameter_nm;
        plot(plot_thickness , plot_diameter, ['k',shapes(currArm),'-'], ...
                  'LineWidth', 2, 'MarkerSize', markerSize, ...
                  'MarkerFaceColor', 'w', 'MarkerEdgeColor', 'k');
        hold on;
    end
    xlabel('t (nm)','FontSize',fontSize,'FontWeight','bold');
    ylabel('w_B (nm)','FontSize',fontSize,'FontWeight','bold');
    set(gca,'LineWidth',lineWidth,'FontSize',fontSize,'FontWeight','bold');
    print('-dtiff',[outputDir,'\Arm_Diameter_v2_',num2str(currCol)]);
    close;
end


disp('Done');
```

Appendix D

Spatially graded GLAD modeling

D.1 Code to plot shadowed slanted post calibration film thicknesses

```
% Dep from June 7' 07
% Slanted Posts
% Shadowed
% Calibration run
% alpha = 30
% Nominally a lum film
clear all;
close all;
clc;

alpha = 30;
titleHead = ['Shadowed Slanted Posts, alpha=',num2str(alpha)];
dataDir = ['C:\My_Documents\Research\Research_Areas\Graded\Calibration\', ...
           'Meas\Shadowed_Slanted_Post_Calibration_Data\'];
plotDir = [dataDir,'\Alpha_',num2str(alpha)];mkdir(plotDir);
shadow_r = 25;
safety_r = 7;   % 25 mm + safety_r mm

axisParam = [0,50,0,1.2];

% Shadow profile closer to the chuck
% 0.1cm from shadow block edge
chuck.shad(1).avg_y = 1.0 + shadow_r;

d = [ 50, 48,  45,   40, 35,  32,   31, 30, 28,   25,  22, ...
       20, 18, 15,   10,  5,   0];
h = [1.02,1.02,1.02,1.02,1.01,0.98,0.95,0.9,0.8,0.25,0.1, ...
       0.08,0.06,0.04,0.025,0.02,0.01];
[d,i] = sort(d);
h = h(i);

chuck.shad(1).x_orig = d;
chuck.shad(1).y_orig = ones(length(d),1) * chuck.shad(1).avg_y;
chuck.shad(1).h_orig = h;
chuck.shad(1).avg_h_orig = mean( h(find(d > (shadow_r + safety_r))) );


chuck.shad(1).x = d;
chuck.shad(1).y = ones(length(d),1) * chuck.shad(1).avg_y;
chuck.shad(1).h = h;
chuck.shad(1).avg_h = mean( h(find(d > (shadow_r + safety_r))) );

figure;hold on;
plot(d,h,'kd');
xlabel('Distance from Radial axis (mm)');
ylabel('Height (um)');


% Shadowed piece #1
% 0.65cm from shadow block edge
chuck.shad(2).avg_y = 6.5 + shadow_r;
```

```
x = [44.042,44.074,44.107,44.123,44.146,44.166,44.181,44.207, ...
       44.214,44.259,44.293,44.323,44.329, ...
     44.367,44.398,44.425,44.442,44.457,44.477,44.497,44.525, ...
       44.558,44.592];
y = [0.298,2.565,4.884,5.918,7.382,8.591,9.566,11.293,13.545, ...
       14.788,16.543,18.284, 19.282,21.471,22.806,24.859, ...
       26.243,27.340,28.753,30.312,32.682,35.361,38.126];
x_end = [44.6482];
y_end = [43.150];
h = [0.99,0.99,0.99,0.99,0.99,0.99,0.99,0.99,0.99,0.985,0.98,0.97, ...
     0.95,0.92,0.88,0.73,0.31,0.21,0.15,0.11,0.08,0.06,0.06];

d = sqrt( (x-x_end).^2 + (y-y_end).^2 );

chuck.shad(2).x_orig = d;
chuck.shad(2).y_orig = ones(length(d),1) * chuck.shad(1).avg_y;
chuck.shad(2).h_orig = h;
chuck.shad(2).avg_h_orig = mean( h(find(d > (shadow_r + safety_r))) );

% Add extra points to smooth out spline
d = [d,45,50,1,0];
h = [h,0.99,0.99,0.06,0.06];
[d,i] = sort(d);
h = h(i);

chuck.shad(2).x = d;
chuck.shad(2).y = ones(length(d),1) * chuck.shad(1).avg_y;
chuck.shad(2).h = h;
chuck.shad(2).avg_h = mean( h(find(d > (shadow_r + safety_r))) );

plot(d,h,'bo');


% Shadowed piece #2
% 1.4cm from shadow block edge
chuck.shad(3).avg_y = 14 + shadow_r;

x = [4.454,4.576,4.667,4.811,4.954,5.141,5.295,5.446,5.699, ...
       5.755,5.868,5.977,6.267,6.454,6.688,6.748];
y = [45.829,43.424,41.708,39.902,36.215,32.378,29.384,26.460, ...
       21.727,20.715,18.660,16.685,11.361,7.975,3.774,2.718];
x_end = [6.747];
y_end = [2.709];
h = [0.97,0.97,0.97,0.97,0.97,0.968,0.96,0.95,0.925,0.92,0.91, ...
      0.90,0.87,0.81,0.70,0.70 ];


d = sqrt( (x-x_end).^2 + (y-y_end).^2 );

chuck.shad(3).x_orig = d;
chuck.shad(3).y_orig = ones(length(d),1) * chuck.shad(1).avg_y;
chuck.shad(3).h_orig = h;
chuck.shad(3).avg_h_orig = mean( h(find(d > (shadow_r + safety_r))) );

% Add extra points to smooth out spline
d = [d,4,10,28,45,50];
h = [h,0.75,0.88,0.964,0.97,0.97];
[d,i] = sort(d);
h = h(i);

chuck.shad(3).x = d;
chuck.shad(3).y = ones(length(d),1) * chuck.shad(2).avg_y;
chuck.shad(3).h = h;
chuck.shad(3).avg_h = mean( h(find(d > (shadow_r + safety_r))) );

plot(d,h,'ro');

% Shadowed piece #3
% 1.95cm from shadow block edge
chuck.shad(4).avg_y = 19.5 + shadow_r;

x = [45.777,45.457,45.174,44.807,44.524,44.220, ...
       44.052,43.852,43.586,43.406,43.245,43.118 ];
y = [4.239,9.403,14.047,20.099,24.581,29.400,32.077, ...
       35.268,39.310,42.118,44.668,46.685];
x_end = [43.109];
y_end = [46.721];
h = [0.93,0.93,0.93,0.93,0.93,0.928,0.925,0.92,0.91, ...
       0.905,0.90,0.90];
```

```
d = sqrt( (x-x_end).^2 + (y-y_end).^2 );

chuck.shad(4).x_orig = d;
chuck.shad(4).y_orig = ones(length(d),1) * chuck.shad(1).avg_y;
chuck.shad(4).h_orig = h;
chuck.shad(4).avg_h_orig = mean( h(find(d > (shadow_r + safety_r))) );

% Add extra points to smooth out spline
d = [d,45,50];
h = [h,0.93,0.93];
[d,i] = sort(d);
h = h(i);

chuck.shad(4).x = d;
chuck.shad(4).y = ones(length(d),1) * chuck.shad(3).avg_y;
chuck.shad(4).h = h;
chuck.shad(4).avg_h = mean( h(find(d > (shadow_r + safety_r))) );

plot(d,h,'go');


% Shadowed piece #4
% 2.60cm from shadow block edge
chuck.shad(5).avg_y = 26 + shadow_r;

x =  [7.114,6.926,6.675,6.359,6.016,5.149,4.843,4.356];
y = [45.572,43.019,39.714,35.556,30.987,19.220,14.826,7.989];
x_end = [4.016];
y_end = [4.219];
h = [0.90,0.90,0.90,0.90,0.90,0.90,0.90,0.90];

d = sqrt( (x-x_end).^2 + (y-y_end).^2 );

chuck.shad(5).x_orig = d;
chuck.shad(5).y_orig = ones(length(d),1) * chuck.shad(1).avg_y;
chuck.shad(5).h_orig = h;
chuck.shad(5).avg_h_orig = mean( h(find(d > (shadow_r + safety_r))) );

% Add extra points to smooth out spline
d = [0,d];
h = [0.889,h];
[d,i] = sort(d);
h = h(i);

chuck.shad(5).x = d;
chuck.shad(5).y = ones(length(d),1) * chuck.shad(4).avg_y;
chuck.shad(5).h = h;
chuck.shad(5).avg_h = mean( h(find(d > (shadow_r + safety_r))) );

plot(d,h,'mo');



% Shadowed piece #5
% 3.70cm from shadow block edge
chuck.shad(6).avg_y = 37 + shadow_r;

x = [43.097,43.457,44.058,44.800,45.847];
y = [4.174,8.839,16.645,26.320,40.210];
x_end = [45.847];
y_end = [40.210];
h = [0.86,0.86,0.86,0.86,0.86];

d = sqrt( (x-x_end).^2 + (y-y_end).^2 );

chuck.shad(6).x_orig = d;
chuck.shad(6).y_orig = ones(length(d),1) * chuck.shad(1).avg_y;
chuck.shad(6).h_orig = h;
chuck.shad(6).avg_h_orig = mean( h(find(d > (shadow_r + safety_r))) );

% Add extra points to smooth out spline
d = [40,50,d];
h = [0.86,0.86, h];
[d,i] = sort(d);
h = h(i);

chuck.shad(6).x = d;
chuck.shad(6).y = ones(length(d),1) * chuck.shad(5).avg_y;
chuck.shad(6).h = h;
chuck.shad(6).avg_h = mean(h); %mean( h(find(d > (shadow_r + safety_r))) );
```

```
plot(d,h,'ko');


% Shadowed piece #6
% 5.10cm from shadow block edge
chuck.shad(7).avg_y = 51 + shadow_r;

x = [5.913,5.314,4.940];
y = [28.385,19.422,13.696];
x_end = [4.841];
y_end = [12.083];
h = [0.80,0.80,0.80];

d = sqrt( (x-x_end).^2 + (y-y_end).^2 );

chuck.shad(7).x_orig = d;
chuck.shad(7).y_orig = ones(length(d),1) * chuck.shad(1).avg_y;
chuck.shad(7).h_orig = h;
chuck.shad(7).avg_h_orig = mean( h(find(d > (shadow_r + safety_r))) );

% Add extra points to smooth out spline
d = [30,40,50,d];
h = [0.80, 0.80, 0.80, h];
[d,i] = sort(d);
h = h(i);

chuck.shad(7).x = d;
chuck.shad(7).y = ones(length(d),1) * chuck.shad(6).avg_y;
chuck.shad(7).h = h;
chuck.shad(7).avg_h = mean(h); %mean( h(find(d > (shadow_r + safety_r))) );

plot(d,h,'co');



% Radial piece #1
chuck.rad(1).avg_y = -10;

x = [42.730,43.226,44.104,44.710,45.375,45.901];
y = [0.172,6.025,16.297,23.284,31.182,37.397 ];
x_end = [42.730];
y_end = [0.172];
h = [1.111,1.115,1.148,1.141,1.141,1.120];

d = sqrt( (x-x_end).^2 + (y-y_end).^2 );

chuck.rad(1).x = d + shadow_r;
chuck.rad(1).y = ones(length(d),1) * chuck.rad(1).avg_y;
chuck.rad(1).h = h;
chuck.rad(1).avg_h = mean(h);

plot(d,h,'bs');

% Radial piece #2
chuck.rad(2).avg_y = 0;

x = [7.290,6.930,6.505,5.849,4.936];
y = [49.964,44.858,38.953,29.693,16.944];
x_end = [7.290];
y_end = [49.964];
h = [1.061,1.072,1.085,1.1071,1.098];

d = sqrt( (x-x_end).^2 + (y-y_end).^2 );

chuck.rad(2).x = d + shadow_r;
chuck.rad(2).y = ones(length(d),1) * chuck.rad(2).avg_y;
chuck.rad(2).h = h;
chuck.rad(2).avg_h = mean(h);

plot(d,h,'rs');

% Radial piece #3
chuck.rad(3).avg_y = 10;

x = [42.694,42.969,43.366,44.221,45.078,45.717];
y = [1.458,4.619,9.344,19.519,29.711,37.697];
x_end = [42.603];
y_end = [1.494];
h = [1.030,1.032,1.045,1.047,1.057,0.9922];

d = sqrt( (x-x_end).^2 + (y-y_end).^2 );
```

```
chuck.rad(3).x = d + shadow_r;
chuck.rad(3).y = ones(length(d),1) * chuck.rad(3).avg_y;
chuck.rad(3).h = h;
chuck.rad(3).avg_h = mean(h);

plot(d,h,'rs');


avg_h = chuck.rad(2).avg_h;


% Unshadowed piece #1
% 1.3cm from edge
chuck.unshad(1).avg_y = -13 - shadow_r;

offset = 20;

x = [7.019,6.935,6.624,5.937,5.580,4.976];
y = [47.000,45.755,41.142,30.575,25.211,15.856];
x_end = [6.955];
y_end = [47.089];
h = [1.226,1.232,1.246,1.261,1.266,1.277];

d = sqrt( (x-x_end).^2 + (y-y_end).^2 )  - offset;

chuck.unshad(1).x = d;
chuck.unshad(1).y = ones(length(d),1) * chuck.unshad(1).avg_y;
chuck.unshad(1).h = h;
chuck.unshad(1).avg_h = mean(h);

plot(d,h,'bd');


% Unshadowed piece #2
% 2.4cm from edge
% Strange data - disregarding


% Unshadowed piece #3
% 3.7cm from edge
chuck.unshad(2).avg_y = -37 - shadow_r;
x = [6.761,6.363,5.504];
y = [43.994,37.681,24.081];
x_end = [6.737];
y_end = [44.685];
h = [1.299,1.300,1.300 ];

d = sqrt( (x-x_end).^2 + (y-y_end).^2 )  - offset;

chuck.unshad(2).x = d;
chuck.unshad(2).y = ones(length(d),1) * chuck.unshad(2).avg_y;
chuck.unshad(2).h = h;
chuck.unshad(2).avg_h = mean(h);

plot(d,h,'gd');


% Unshadowed piece #4
% 5.2cm from edge
chuck.unshad(3).avg_y = -52 - shadow_r;
x = [43.850,44.257];
y = [14.478,19.557];
x_end = [43.651];
y_end = [12.922];
h = [1.367,1.354];

d = sqrt( (x-x_end).^2 + (y-y_end).^2 )  - offset;

chuck.unshad(3).x = d;
chuck.unshad(3).y = ones(length(d),1) * chuck.unshad(3).avg_y;
chuck.unshad(3).h = h;
chuck.unshad(3).avg_h = mean(h);

plot(d,h,'md');


Calcs_Plots


%===============================================================================
```

```
% Dep from Sept 26 2007
% Slanted Posts
% Shadowed
% Calibration run
% alpha = 60
% Nominally a 1um film
clear all;
close all;
clc;

alpha = 60;
titleHead = ['Shadowed Slanted Posts, alpha=',num2str(alpha)];
dataDir = ['C:\My_Documents\Research\Research_Areas\Graded\Calibration\', ...
           'Meas\Shadowed_Slanted_Post_Calibration_Data\'];
plotDir = [dataDir,'\Alpha_',num2str(alpha)];mkdir(plotDir);
shadow_r = 25;
safety_r = 10;   % 25 mm + 10 mm safety

axisParam = [0,50,0,0.8];

% Shadowed piece #1
% 1cm from shadow block edge
chuck.shad(1).avg_y = 9 + shadow_r;

x = [5.68 5.857 5.848 5.844 5.826 5.810 5.793 5.727 5.636];
y = [40.428 39.892 39.415 39.215 37.885 36.884 35.421 31.515 26.515];
x_end = [5.636];
y_end = [11];
h = [0.5596 0.4643 0.3010 0.2652 0.1657 0.1119 0.07739 0.03225 0.01873];

d = sqrt( (x-x_end).^2 + (y-y_end).^2 );

chuck.shad(1).x_orig = d;
chuck.shad(1).y_orig = ones(length(d),1) * chuck.shad(1).avg_y;
chuck.shad(1).h_orig = h;
chuck.shad(1).avg_h_orig = mean( h(find(d > (shadow_r + safety_r))) );

% Add extra points to smooth out spline
d = [0,    5,      10,  d,  30,   31,   35,   40,   45,   48,   49,  50 ];
h = [0.01, 0.01, 0.015, h, 0.61, 0.65, 0.69, 0.72, 0.73, 0.735, 0.735 0.735];
[d,i] = sort(d);
h = h(i);

chuck.shad(1).x = d;
chuck.shad(1).y = ones(length(d),1) * chuck.shad(1).avg_y;
chuck.shad(1).h = h;
chuck.shad(1).avg_h = 0.74; %mean( h(find(d > (shadow_r + safety_r))) );

figure;hold on;
plot(d,h,'bo');
xlabel('Distance from Radial axis (mm)');
ylabel('Height (um)');

% Shadowed piece #2
% 2cm from shadow block edge
chuck.shad(2).avg_y = 20 + shadow_r;

x = [ 44.465 44.466 44.467 44.467 44.471 44.475 44.480 44.495 44.511 44.536];
y = [ 11.680 11.701 12.063 12.343 12.976 13.590 14.958 16.238 19.055 24.233];

x_end = [44.536];
y_end = [40];
h = [ 0.5655 0.5646 0.5209 0.4485 0.2927 0.2342 0.1720 0.1508 0.1010 0.049 ];

d = sqrt( (x-x_end).^2 + (y-y_end).^2 );

chuck.shad(2).x_orig = d;
chuck.shad(2).y_orig = ones(length(d),1) * chuck.shad(2).avg_y;
chuck.shad(2).h_orig = h;
chuck.shad(2).avg_h_orig = 0.69; %mean( h(find(d > (shadow_r + safety_r))) );

% Add extra points to smooth out spline
d = [0,    5,      10,  d,   30,   35,   40,    45, 50];
h = [0.03, 0.035, 0.04,  h, 0.63, 0.68, 0.69, 0.70, 0.70];
[d,i] = sort(d);
h = h(i);

chuck.shad(2).x = d;
chuck.shad(2).y = ones(length(d),1) * chuck.shad(2).avg_y;
chuck.shad(2).h = h;
```

```
chuck.shad(2).avg_h = 0.70; %mean( h(find(d > (shadow_r + safety_r))) );

plot(d,h,'ro');

% Shadowed piece #3
% 3cm from shadow block edge
chuck.shad(3).avg_y = 31 + shadow_r;

x = [ 3.613  3.718  3.878  4.061  4.251  4.286  4.303  4.489  4.834  ...
        4.988  5.143  5.306  5.631   5.931   6.239  6.463 6.891 7.265 ];
y = [36.207 35.477 34.294 32.937 31.540 31.296 31.174 29.802 27.410 ...
        26.283 25.146 23.978 21.598 19.437 17.169  15.540  12.503  9.725 ];

x_end = [7.776];
y_end = [6.212];
h = [0.63 0.61 0.58 0.5429 0.4167 0.3827 0.3614 0.2466 0.1734 0.1573 ...
       0.1394 0.1278 0.1136 0.1037 0.09558 0.094 0.091 0.090];

d = sqrt( (x-x_end).^2 + (y-y_end).^2 );

chuck.shad(3).x_orig = d;
chuck.shad(3).y_orig = ones(length(d),1) * chuck.shad(3).avg_y;
chuck.shad(3).h_orig = h;
chuck.shad(3).avg_h_orig = mean( h(find(d > (shadow_r + safety_r))) );

% Add extra points to smooth out spline
d = [0,     d,    32,    35,   38,     40,  45,   48,     50];
h = [0.089, h,   0.64, 0.655, 0.66,   0.67, 0.67, 0.67,  0.67];
[d,i] = sort(d);
h = h(i);

chuck.shad(3).x = d;
chuck.shad(3).y = ones(length(d),1) * chuck.shad(3).avg_y;
chuck.shad(3).h = h;
chuck.shad(3).avg_h = 0.67; %mean( h(find(d > (shadow_r + safety_r))) );

plot(d,h,'go');


% Shadowed piece #4
% 4cm from shadow block edge
chuck.shad(4).avg_y = 40 + shadow_r;

x =  [26.717 27.127 28.190 29.242 31.168 32.327 33.957 37.752 ...
        41.298 42.848 44.800 47.785];
y = [1.786 2.114 2.894 3.704 5.150 6.009 7.233 10.100 12.777 ...
        13.943 15.425 17.699];
x_end = [50];
y_end = [20.556];
h = [0.635 0.63 0.625 0.621 0.6019 0.5926 0.5655 0.4799 0.33 ...
         0.30 0.295 0.29];

d = sqrt( (x-x_end).^2 + (y-y_end).^2 );

chuck.shad(4).x_orig = d;
chuck.shad(4).y_orig = ones(length(d),1) * chuck.shad(4).avg_y;
chuck.shad(4).h_orig = h;
chuck.shad(4).avg_h_orig = mean( h(find(d > (shadow_r + safety_r))) );

% Add extra points to smooth out spline
d = [0,   d,    35,    40,   45,  50];
h = [0.28,h, 0.645, 0.649, 0.65, 0.65];
[d,i] = sort(d);
h = h(i);

chuck.shad(4).x = d;
chuck.shad(4).y = ones(length(d),1) * chuck.shad(4).avg_y;
chuck.shad(4).h = h;
chuck.shad(4).avg_h = 0.65; %mean( h(find(d > (shadow_r + safety_r))) );

plot(d,h,'mo');



% Shadowed piece #5
% 5cm from shadow block edge
chuck.shad(5).avg_y = 52 + shadow_r;

x = [36.805 32.519 17.03];
y = [40.913 42.691 49.041];
x_end = [17.03];
```

```
y_end = [49.041];
h = [0.60 0.59 0.58];

d = sqrt( (x-x_end).^2 + (y-y_end).^2 );

chuck.shad(5).x_orig = d;
chuck.shad(5).y_orig = ones(length(d),1) * chuck.shad(5).avg_y;
chuck.shad(5).h_orig = h;
chuck.shad(5).avg_h_orig = mean( h(find(d > (shadow_r + safety_r))) );

% Add extra points to smooth out spline
d = [ 5,     10,  d,  30,   40,    50];
h = [0.58, 0.585, h, 0.60, 0.60,  0.60];
[d,i] = sort(d);
h = h(i);

chuck.shad(5).x = d;
chuck.shad(5).y = ones(length(d),1) * chuck.shad(5).avg_y;
chuck.shad(5).h = h;
chuck.shad(5).avg_h = 0.60; %mean( h(find(d > (shadow_r + safety_r))) );

plot(d,h,'ko');



% Radial piece #1
%chuck.rad(1).avg_y = xx;
%x = [44.862 44.795 44.660 44.421 44.266 ];
%y = [10.673 15.219 23.193 36.603 44.884 ];
%x_end = [44.258];
%y_end = [45.011];
%h = [1.0051 0.944 0.9012 0.845 0.8004];
%d = sqrt( (x-x_end).^2 + (y-y_end).^2 );
%chuck.rad(1).x = d + shadow_r;
%chuck.rad(1).y = ones(length(d),1) * chuck.rad(1).avg_y;
%chuck.rad(1).h = h;
%chuck.rad(1).avg_h = mean(h);
%plot(d,h,'bs');

chuck.rad(1).avg_y = -20;
chuck.rad(1).avg_h = 1.00;
chuck.rad(2).avg_y = -15;
chuck.rad(2).avg_h = 0.944;
chuck.rad(3).avg_y = -8;
chuck.rad(3).avg_h = 0.901;
chuck.rad(4).avg_y = 6.5;
chuck.rad(4).avg_h = 0.845;
chuck.rad(5).avg_y = 14.8;
chuck.rad(5).avg_h = 0.800;

avg_h = 0.9;

% Unshadowed piece #1
% xxcm from edge
%chuck.unshad(1).avg_y = -xx - shadow_r;
%offset = xx;
%x = [5.690 5.686 5.688 5.694 5.706 5.715 ];
%y = [30.723 24.713 19.497 13.717 7.621 1.543];
%x_end = [5.448];
%y_end = [0.000];
%h = [1.161 1.151 1.121 1.093 1.058 1.005];
%d = sqrt( (x-x_end).^2 + (y-y_end).^2 )  - offset;
%chuck.unshad(1).x = d;
%chuck.unshad(1).y = ones(length(d),1) * chuck.unshad(1).avg_y;
%chuck.unshad(1).h = h;
%chuck.unshad(1).avg_h = mean(h);
%plot(d,h,'bd');

chuck.unshad(1).avg_y = -2 - shadow_r;
chuck.unshad(1).avg_h = 1.02;
chuck.unshad(2).avg_y = -8 - shadow_r;
chuck.unshad(2).avg_h = 1.06;
chuck.unshad(3).avg_y = -14.2 - shadow_r;
chuck.unshad(3).avg_h = 1.09;
chuck.unshad(4).avg_y = -20 - shadow_r;
chuck.unshad(4).avg_h = 1.12;
chuck.unshad(5).avg_y = -25.2 - shadow_r;
chuck.unshad(5).avg_h = 1.15;
chuck.unshad(6).avg_y = -31.2 - shadow_r;
chuck.unshad(6).avg_h = 1.16;
```

Calcs_Plots

```
%================================================================================


% Dep from June 7' 07
% Slanted Posts
% Shadowed
% Calibration run
% alpha = 70
% Nominally a 1um film
clear all;
close all;
clc;

alpha = 70;
titleHead = ['Shadowed Slanted Posts, alpha=',num2str(alpha)];
dataDir =  ['C:\My_Documents\Research\Research_Areas\Graded\Calibration\', ...
            'Meas\Shadowed_Slanted_Post_Calibration_Data\'];
plotDir = [dataDir,'\Alpha_',num2str(alpha)];mkdir(plotDir);
shadow_r = 25;
safety_r = 10;   % 25 mm + 10 mm safety

axisParam = [0,50,0,0.8];

% Shadowed piece #1
% 0.9cm from shadow block edge
chuck.shad(1).avg_y = 9 + shadow_r;

x = [45.561, 45.538, 45.467, 45.354, 45.103, 45.010, 44.958, 44.901, ...
        44.884, 44.777, 44.724];
y = [ 0.023,  0.338,  2.237,  5.089, 11.575, 14.001, 15.607, 16.843, ...
        17.204, 20.016, 21.416];
x_end = [43.779];
y_end = [43.466];
h = [0.77,  0.77, 0.77, 0.77, 0.73, 0.68, 0.62, 0.36, 0.26, 0.099, 0.074 ];

d = sqrt( (x-x_end).^2 + (y-y_end).^2 );

chuck.shad(1).x_orig = d;
chuck.shad(1).y_orig = ones(length(d),1) * chuck.shad(1).avg_y;
chuck.shad(1).h_orig = h;
chuck.shad(1).avg_h_orig = mean( h(find(d > (shadow_r + safety_r))) );

% Add extra points to smooth out spline
d = [d,50,45, 28.5,30,25,20,15,5,0];
h = [h,0.77,0.77,0.65,0.695,0.14,0.05,0.024,0.01,0.005];
[d,i] = sort(d);
h = h(i);

chuck.shad(1).x = d;
chuck.shad(1).y = ones(length(d),1) * chuck.shad(1).avg_y;
chuck.shad(1).h = h;
chuck.shad(1).avg_h = mean( h(find(d > (shadow_r + safety_r))) );

figure;hold on;
plot(d,h,'bo');
xlabel('Distance from Radial axis (mm)');
ylabel('Height (um)');

% Shadowed piece #2
% 2cm from shadow block edge
chuck.shad(2).avg_y = 20 + shadow_r;

x = [ 1.733,  2.482,  3.110,  3.916,  4.259,  4.534,  4.720,  ...
         4.783,  4.850,  4.982,  5.267];
y = [49.997, 44.812, 40.546, 35.053, 32.675, 30.810, 29.370, ...
         28.971, 28.548, 27.626, 25.268];
x_end = [8.2];
y_end = [5.3];
h = [ 0.7177, 0.71, 0.70, 0.6405, 0.5304, 0.2072, 0.13, 0.12, ...
         0.11, 0.099, 0.07541];

d = sqrt( (x-x_end).^2 + (y-y_end).^2 );

chuck.shad(2).x_orig = d;
chuck.shad(2).y_orig = ones(length(d),1) * chuck.shad(1).avg_y;
chuck.shad(2).h_orig = h;
chuck.shad(2).avg_h_orig = mean( h(find(d > (shadow_r + safety_r))) );
```

```
d = [50, d];
h = [h(1),h];

% Add extra points to smooth out spline
d = [d,28,15,5,0];
h = [h,0.56,0.036,0.021,0.017];
[d,i] = sort(d);
h = h(i);

chuck.shad(2).x = d;
chuck.shad(2).y = ones(length(d),1) * chuck.shad(2).avg_y;
chuck.shad(2).h = h;
chuck.shad(2).avg_h = mean( h(find(d > (shadow_r + safety_r))) );

plot(d,h,'ro');

% Shadowed piece #3
% 3.2cm from shadow block edge
chuck.shad(3).avg_y = 32 + shadow_r;

x = [47.023, 46.434, 45.740, 45.382, 45.221, 45.110 ];
y = [10.362, 14.434, 19.317, 21.766, 22.847, 23.668 ];
x_end = [41.65];
y_end = [47.70];
h = [0.65, 0.63, 0.47, 0.18, 0.15, 0.14];

d = sqrt( (x-x_end).^2 + (y-y_end).^2 );

chuck.shad(3).x_orig = d;
chuck.shad(3).y_orig = ones(length(d),1) * chuck.shad(1).avg_y;
chuck.shad(3).h_orig = h;
chuck.shad(3).avg_h_orig = mean( h(find(d > (shadow_r + safety_r))) );

% Add extra points to smooth out spline
d = [50, 45, d];
h = [0.66,0.66, h];
d = [d,30,20,15,10,5,0];
h = [h,0.55,0.09,0.065,0.047,0.040,0.034];
[d,i] = sort(d);
h = h(i);

chuck.shad(3).x = d;
chuck.shad(3).y = ones(length(d),1) * chuck.shad(3).avg_y;
chuck.shad(3).h = h;
chuck.shad(3).avg_h = mean( h(find(d > (shadow_r + safety_r))) );

plot(d,h,'go');


% Shadowed piece #4
% 4.35cm from shadow block edge
chuck.shad(4).avg_y = 43.5 + shadow_r;

x =  [7.367,  7.130,  6.823,  6.560,  6.367,  6.135,  5.946,  ...
       5.653,  5.283,  4.971,  4.624,  4.276];
y = [49.642, 46.844, 42.924, 39.710, 37.288, 34.319, 31.814, ...
       28.213, 23.387, 19.258, 14.680, 9.999];
x_end = [4.276];
y_end = [9.999];
h = [0.61, 0.60, 0.58, 0.48, 0.24, 0.15, 0.13, 0.11, 0.09, ...
        0.08, 0.07, 0.06];

d = sqrt( (x-x_end).^2 + (y-y_end).^2 );

chuck.shad(4).x_orig = d;
chuck.shad(4).y_orig = ones(length(d),1) * chuck.shad(1).avg_y;
chuck.shad(4).h_orig = h;
chuck.shad(4).avg_h_orig = mean( h(find(d > (shadow_r + safety_r))) );

% Add extra points to smooth out spline
d = [50,45,31,d];
h = [0.62,0.62,0.54,h];
[d,i] = sort(d);
h = h(i);

chuck.shad(4).x = d;
chuck.shad(4).y = ones(length(d),1) * chuck.shad(4).avg_y;
chuck.shad(4).h = h;
chuck.shad(4).avg_h = mean( h(find(d > (shadow_r + safety_r))) );

plot(d,h,'mo');
```

```
% Shadowed piece #5
% 5.45cm from shadow block edge
chuck.shad(5).avg_y = 54.5 + shadow_r;

x = [43.590, 43.839, 44.112, 44.460, 44.863, 45.286];
y = [10.655, 14.588, 19.282, 25.173, 31.987, 39.604];
x_end = [45.4];
y_end = [41.33];

h = [0.49, 0.26, 0.16, 0.13, 0.12, 0.11];

d = sqrt( (x-x_end).^2 + (y-y_end).^2 );

chuck.shad(5).x_orig = d;
chuck.shad(5).y_orig = ones(length(d),1) * chuck.shad(1).avg_y;
chuck.shad(5).h_orig = h;
chuck.shad(5).avg_h_orig = mean( h(find(d > (shadow_r + safety_r))) );

% Add extra points to smooth out spline
d = [50, 45,40,35,32,0,d];
h = [0.58, 0.58, 0.57, 0.55,0.52,0.109 h];
[d,i] = sort(d);
h = h(i);

chuck.shad(5).x = d;
chuck.shad(5).y = ones(length(d),1) * chuck.shad(5).avg_y;
chuck.shad(5).h = h;
chuck.shad(5).avg_h = mean( h(find(d > (shadow_r + safety_r))) );

plot(d,h,'ko');



% Radial piece #1
chuck.rad(1).avg_y = 5;

x = [ 4.851,  5.496,  6.368,  6.969 ];
y = [32.638, 23.306, 11.007, 2.536 ];
x_end = [7.197];
y_end = [0.0404];
h = [0.8811, 0.8632, 0.8910, 0.8890];

d = sqrt( (x-x_end).^2 + (y-y_end).^2 );

chuck.rad(1).x = d + shadow_r;
chuck.rad(1).y = ones(length(d),1) * chuck.rad(1).avg_y;
chuck.rad(1).h = h;
chuck.rad(1).avg_h = mean(h);

plot(d,h,'bs');

% Radial piece #2
chuck.rad(2).avg_y = 20;

x = [46.034, 45.311, 44.635, 43.582, 42.729];
y = [ 5.346, 13.809, 21.592, 33.694, 43.368];
x_end = [42.394];
y_end = [48.257];
h = [0.8051, 0.8354, 0.8363, 0.8592, 0.8394];

d = sqrt( (x-x_end).^2 + (y-y_end).^2 );

chuck.rad(2).x = d + shadow_r;
chuck.rad(2).y = ones(length(d),1) * chuck.rad(1).avg_y;
chuck.rad(2).h = h;
chuck.rad(2).avg_h = mean(h);

plot(d,h,'rs');

avg_h = chuck.rad(1).avg_h;


% Unshadowed piece #1
% 0.8cm from edge
chuck.unshad(1).avg_y = -8 - shadow_r;

% Estimate radial line along wafer at
offset = 15;  %mm
```

```
x = [ 6.001,  5.834,  5.683,  5.488,  5.304,  5.174,  4.990,  4.589];
y = [44.091, 38.989, 34.228, 28.240, 22.643, 18.550, 12.719, 0.218];
x_end = [4.589];
y_end = [0.218];
h = [1.159, 1.184, 1.175, 1.184, 1.195, 1.183, 1.175, 1.153];

d = sqrt( (x-x_end).^2 + (y-y_end).^2 ) - offset;

chuck.unshad(1).x = d;
chuck.unshad(1).y = ones(length(d),1) * chuck.unshad(1).avg_y;
chuck.unshad(1).h = h;
chuck.unshad(1).avg_h = mean(h);

plot(d,h,'bd');


% Unshadowed piece #2
% 1.8cm from edge
chuck.unshad(2).avg_y = -18 - shadow_r;
x = [43.814, 44.018, 44.224, 44.491, 44.995];
y = [ 5.664, 13.369, 21.006, 31.150, 49.945 ];
x_end = [43.858];
y_end = [2.713];
h = [1.212, 1.1915, 1.212, 1.201, 1.177];

d = sqrt( (x-x_end).^2 + (y-y_end).^2 ) - offset;

chuck.unshad(2).x = d;
chuck.unshad(2).y = ones(length(d),1) * chuck.unshad(2).avg_y;
chuck.unshad(2).h = h;
chuck.unshad(2).avg_h = mean(h);

plot(d,h,'rd');


% Unshadowed piece #3
% 2.8cm from edge
chuck.unshad(3).avg_y = -28 - shadow_r;
x = [5.938, 5.633, 5.419, 4.900];
y = [42.016, 33.679, 27.743, 13.578];
x_end = [6.201];
y_end = [47.587];
h = [1.290, 1.314, 1.300, 1.304 ];

d = sqrt( (x-x_end).^2 + (y-y_end).^2 ) - offset;

chuck.unshad(3).x = d;
chuck.unshad(3).y = ones(length(d),1) * chuck.unshad(3).avg_y;
chuck.unshad(3).h = h;
chuck.unshad(3).avg_h = mean(h);

plot(d,h,'gd');


% Unshadowed piece #4
% 3.8cm from edge
chuck.unshad(4).avg_y = -38 - shadow_r;
x = [ 43.860, 44.015, 44.328, 44.667];
y = [  4.212, 10.295, 22.589, 36.356 ];
x_end = [43.860];
y_end = [4.212];
h = [ 1.349, 1.377, 1.361, 1.363];

d = sqrt( (x-x_end).^2 + (y-y_end).^2 )  - offset;

chuck.unshad(4).x = d;
chuck.unshad(4).y = ones(length(d),1) * chuck.unshad(4).avg_y;
chuck.unshad(4).h = h;
chuck.unshad(4).avg_h = mean(h);

plot(d,h,'md');


% Unshadowed piece #5
% 4.9cm from edge
chuck.unshad(5).avg_y = -49 - shadow_r;
x = [ 5.707,  5.647,  5.575,  5.489,  5.436,  5.354];
y = [39.608, 34.590, 28.674, 21.194, 17.109, 10.034];
x_end = [5.780];
y_end = [43.664];
```

```
h = [1.525, 1.538, 1.541, 1.511, 1.525, 1.531];

d = sqrt( (x-x_end).^2 + (y-y_end).^2 ) - offset;

chuck.unshad(5).x = d;
chuck.unshad(5).y = ones(length(d),1) * chuck.unshad(5).avg_y;
chuck.unshad(5).h = h;
chuck.unshad(5).avg_h = mean(h);

plot(d,h,'cd');


Calcs_Plots

%================================================================================

% Dep from June 6' 07
% Slanted Posts
% Shadowed
% Calibration run
% alpha = 85
% Nominally a 1um film
clear all;
close all;
clc;

alpha = 85;
titleHead = ['Shadowed Slanted Posts, alpha=',num2str(alpha)];
dataDir = ['C:\My_Documents\Research\Research_Areas\Graded\', ...
           'Calibration\Meas\Shadowed_Slanted_Post_Calibration_Data\]';
plotDir = [dataDir,'\Alpha_',num2str(alpha)];mkdir(plotDir);
shadow_r = 25;   % 25 mm + 10 mm safety
safety_r = 8;

axisParam = [0,50,0,1.2];

% Shadowed piece #1
% 0.75cm from shadow block edge
chuck.shad(1).avg_y = 7.5 + shadow_r;

x = [ 5.519, 5.535, 5.544, 5.557, 5.579,   5.596,  5.618, 5.629,  5.641,   5.667 ];
y = [ 0.671, 2.824, 4.053, 5.682, 8.849,  11.398, 14.389, 16.106, 17.647, 21.220 ];
x_end = [5.670];
y_end = [44.151];
h = [ 1.11, 1.11, 1.105, 1.10, 1.09, 1.08, 1.05, 0.91,0.30, 0.11];

d = sqrt( (x-x_end).^2 + (y-y_end).^2 );

chuck.shad(1).x_orig = d;
chuck.shad(1).y_orig = ones(length(d),1) * chuck.shad(1).avg_y;
chuck.shad(1).h_orig = h;
chuck.shad(1).avg_h_orig = mean( h(find(d > (shadow_r + safety_r))) );


% Add extra points to smooth out spline
d = [d,50,48,47,28.5,25,20,15,5,0];
h = [h,1.11,1.11,1.11,0.99,0.143,0.08,0.05,0.02,0.01];
[d,i] = sort(d);
h = h(i);


chuck.shad(1).x = d;
chuck.shad(1).y = ones(length(d),1) * chuck.shad(1).avg_y;
chuck.shad(1).h = h;
chuck.shad(1).avg_h = mean( h(find(d > (shadow_r + safety_r))) );

figure;hold on;
plot(d,h,'bo');
xlabel('Distance from Radial axis (mm)');
ylabel('Height (um)');

% Shadowed piece #2
% 2cm from shadow block edge
chuck.shad(2).avg_y = 20 + shadow_r;

x = [44.706, 44.697, 44.686, 44.677, 44.666, 44.657, 44.650];
y = [49.667, 45.782, 41.503, 37.634, 33.276, 30.210, 27.833];
x_end = [44.408];
y_end = [4.227];
h = [0.99, 0.99, 0.97, 0.95, 0.77, 0.19, 0.11];
```

```
d = sqrt( (x-x_end).^2 + (y-y_end).^2 );

chuck.shad(2).x_orig = d;
chuck.shad(2).y_orig = ones(length(d),1) * chuck.shad(1).avg_y;
chuck.shad(2).h_orig = h;
chuck.shad(2).avg_h_orig = mean( h(find(d > (shadow_r + safety_r))) );


% Add extra points to smooth out spline
d = [d,50,30,25,22,20,15,5,0];
h = [h,0.99,0.85,0.14,0.095,0.084,0.05,0.025,0.015];
[d,i] = sort(d);
h = h(i);


chuck.shad(2).x = d;
chuck.shad(2).y = ones(length(d),1) * chuck.shad(2).avg_y;
chuck.shad(2).h = h;
chuck.shad(2).avg_h = mean( h(find(d > (shadow_r + safety_r))) );

plot(d,h,'ro');

% Shadowed piece #3
% 3.2cm from shadow block edge
chuck.shad(3).avg_y = 32 + shadow_r;

x = [ 5.724, 5.723,  5.726,  5.727,  5.731,  5.731,  5.733,  5.735,  5.737,  5.739];
y = [ 0.017, 4.945, 10.405, 15.072, 19.857, 20.502, 22.746, 25.419, 27.606, 30.660];
x_end = [5.595];
y_end = [46.919];
h = [0.92, 0.92, 0.90, 0.82, 0.20, 0.16, 0.12, 0.086, 0.069, 0.053];

d = sqrt( (x-x_end).^2 + (y-y_end).^2 );

chuck.shad(3).x_orig = d;
chuck.shad(3).y_orig = ones(length(d),1) * chuck.shad(1).avg_y;
chuck.shad(3).h_orig = h;
chuck.shad(3).avg_h_orig = mean( h(find(d > (shadow_r + safety_r))) );


% Add extra points to smooth out spline
d = [d,50,33,28,5,0];
h = [h,0.92,0.86,0.40,0.03,0.02];
[d,i] = sort(d);
h = h(i);

chuck.shad(3).x = d;
chuck.shad(3).y = ones(length(d),1) * chuck.shad(3).avg_y;
chuck.shad(3).h = h;
chuck.shad(3).avg_h = mean( h(find(d > (shadow_r + safety_r))) );


plot(d,h,'go');



% Shadowed piece #4
% 4.1cm from shadow block edge
chuck.shad(4).avg_y = 41 + shadow_r;

x = [44.841, 44.822, 44.790, 44.752, 44.721, 44.679, 44.621, 44.595, 44.548];
y = [46.109, 42.967, 38.759, 34.137, 29.111, 24.120, 19.117, 16.122, 11.011];
x_end = [44.261];
y_end = [3.714];
h = [0.81, 0.80, 0.78, 0.64, 0.15, 0.090, 0.058, 0.050, 0.043];

d = sqrt( (x-x_end).^2 + (y-y_end).^2 );

chuck.shad(4).x_orig = d;
chuck.shad(4).y_orig = ones(length(d),1) * chuck.shad(1).avg_y;
chuck.shad(4).h_orig = h;
chuck.shad(4).avg_h_orig = mean( h(find(d > (shadow_r + safety_r))) );

% Add extra points to smooth out spline
d = [d,50,23,0];
h = [h,0.81,0.108,0.032];
[d,i] = sort(d);
h = h(i);

chuck.shad(4).x = d;
chuck.shad(4).y = ones(length(d),1) * chuck.shad(4).avg_y;
```

```
chuck.shad(4).h = h;
chuck.shad(4).avg_h = mean( h(find(d > (shadow_r + safety_r))) );

plot(d,h,'mo');



% Shadowed piece #5
% 5.2cm from shadow block edge
chuck.shad(5).avg_y = 52 + shadow_r;

x = [5.626, 5.653, 5.683, 5.712, 5.750];
y = [11.466, 15.766, 19.876, 24.656, 30.319];
x_end = [5.654];
y_end = [42.913];
h = [0.62, 0.19, 0.11, 0.079, 0.068];

d = sqrt( (x-x_end).^2 + (y-y_end).^2 );

chuck.shad(5).x_orig = d;
chuck.shad(5).y_orig = ones(length(d),1) * chuck.shad(1).avg_y;
chuck.shad(5).h_orig = h;
chuck.shad(5).avg_h_orig = mean( h(find(d > (shadow_r + safety_r))) );

% Add extra points to smooth out spline
d = [25,34,   40,   45, 50,5,0, d];
h = [0.13,0.703, 0.725, 0.73,0.73, 0.06, 0.06, h];
[d,i] = sort(d);
h = h(i);

chuck.shad(5).x = d;
chuck.shad(5).y = ones(length(d),1) * chuck.shad(5).avg_y;
chuck.shad(5).h = h;
chuck.shad(5).avg_h = mean( h(find(d > (shadow_r + safety_r))) );

plot(d,h,'ko');



% Radial piece
chuck.rad(1).avg_y = 5;

x = [44.471, 44.488, 44.527, 44.583, 44.650, 44.701, 44.749, 44.806, 44.851];
y = [6.765,   8.488, 12.649, 19.101, 26.432, 32.103, 37.498, 44.036, 49.999];
x_end = [44.263];
y_end = [3.671];
h = [1.218, 1.226, 1.282, 1.290, 1.278, 1.262, 1.239, 1.238, 1.201];

d = sqrt( (x-x_end).^2 + (y-y_end).^2 );

chuck.rad(1).x = d + shadow_r;
chuck.rad(1).y = ones(length(d),1) * chuck.rad(1).avg_y;
chuck.rad(1).h = h;
chuck.rad(1).avg_h = mean(h);

plot(d,h,'bs');

avg_h = chuck.rad(1).avg_h;



% Unshadowed piece #1
% 0.8cm from edge
chuck.unshad(1).avg_y = -8 - shadow_r;

% Estimate radial line along wafer at
offset = 15;  %mm

x = [ 2.806,  3.563,  4.274,  4.730,  5.300,   6.326,  6.489, 6.807,  7.267];
y = [49.861, 42.557, 35.745, 31.432, 25.970,  16.124, 14.507, 11.596, 7.053];
x_end = [7.749];
y_end = [2.686];
h = [1.552,   1.572,  1.568,  1.591,  1.568,   1.540,  1.576,  1.572, 1.528];

d = sqrt( (x-x_end).^2 + (y-y_end).^2 ) - offset;

chuck.unshad(1).x = d;
chuck.unshad(1).y = ones(length(d),1) * chuck.unshad(1).avg_y;
chuck.unshad(1).h = h;
chuck.unshad(1).avg_h = mean(h);
```

```
plot(d,h,'bd');


% Unshadowed piece #2
% 1.95cm from edge
chuck.unshad(2).avg_y = -19.5 - shadow_r;
x = [47.424, 47.167, 46.714, 46.216, 45.903, 45.550, 45.122, 44.914, ...
       44.600, 44.156, 43.666, 43.185, 42.728, 42.358];
y = [ 0.725,  3.006,  6.907, 11.184, 13.899, 16.864, 20.562, 22.390, ...
       24.987, 28.826, 33.011, 37.129, 41.055, 44.162];
x_end = [41.919];
y_end = [48.168];
h = [ 1.627,  1.621,  1.699,  1.647,  1.679,  1.675,  1.667,  1.643,  ...
       1.651,  1.651,  1.675,  1.660,  1.651,  1.631];

d = sqrt( (x-x_end).^2 + (y-y_end).^2 ) - offset;

chuck.unshad(2).x = d;
chuck.unshad(2).y = ones(length(d),1) * chuck.unshad(2).avg_y;
chuck.unshad(2).h = h;
chuck.unshad(2).avg_h = mean(h);

plot(d,h,'rd');


% Unshadowed piece #3
% 3.05cm from edge
chuck.unshad(3).avg_y = -30.5 - shadow_r;
x = [ 3.137,  3.279,  3.781,  4.228,  4.737,  5.344,  ...
       6.014,  6.422,  6.975,  7.351];
y = [48.627, 47.254, 42.499, 38.284, 33.553, 27.845, ...
       21.620, 17.767, 12.549, 8.971];
x_end = [7.856];
y_end = [4.289];
h = [ 1.754,  1.730,  1.746,  1.806,  1.758,  1.750,  ...
       1.786,  1.802,  1.778,  1.778];

d = sqrt( (x-x_end).^2 + (y-y_end).^2 ) - offset;

chuck.unshad(3).x = d;
chuck.unshad(3).y = ones(length(d),1) * chuck.unshad(3).avg_y;
chuck.unshad(3).h = h;
chuck.unshad(3).avg_h = mean(h);

plot(d,h,'gd');


% Unshadowed piece #4
% 3.9cm from edge
chuck.unshad(4).avg_y = -39 - shadow_r;
x = [46.883, 45.988, 45.326, 44.578, 43.748, 42.904];
y = [ 4.262, 11.918, 17.636, 24.07,  31.063, 38.288];
x_end = [41.860];
y_end = [47.049];
h = [1.806, 1.834, 1.905, 1.873, 1.885, 1.861];

d = sqrt( (x-x_end).^2 + (y-y_end).^2 )  - offset;

chuck.unshad(4).x = d;
chuck.unshad(4).y = ones(length(d),1) * chuck.unshad(4).avg_y;
chuck.unshad(4).h = h;
chuck.unshad(4).avg_h = mean(h);

plot(d,h,'md');


% Unshadowed piece #5
% 5.3cm from edge
chuck.unshad(5).avg_y = -53 - shadow_r;
x = [5.042, 5.292, 5.810, 6.332, 6.679];
y = [38.912, 33.900, 24.364, 14.674, 8.258];
x_end = [6.860];
y_end = [5.553];
h = [1.984, 1.965, 2.012, 2.099, 2.056];

d = sqrt( (x-x_end).^2 + (y-y_end).^2 ) - offset;

chuck.unshad(5).x = d;
chuck.unshad(5).y = ones(length(d),1) * chuck.unshad(5).avg_y;
chuck.unshad(5).h = h;
chuck.unshad(5).avg_h = mean(h);
```

```
plot(d,h,'cd');


Calcs_Plots

%================================================================================


% Calcs_Plots code
%   Now put together the shadowed and unshadowed thickness profiles


% A few constants
    chuck.radius = 84;  %mm
    chuck.small_radius = 24; %mm
    chuck.small_safety = 10; %mm
    chuck.dr = 1;
    chuck.shadStop = 50; %mm

% Put together a consise set of vectors
% describing how the thickness profile changes with
% y direction (along path of flux)
    counter = 0;
    for count = 1:length(chuck.shad)
        counter = counter + 1;
        summary.distance(counter) = chuck.shad(count).avg_y;
        summary.thickness(counter) = chuck.shad(count).avg_h;
    end
    for count = 1:length(chuck.rad)
        counter = counter + 1;
        summary.distance(counter) = chuck.rad(count).avg_y;
        summary.thickness(counter) = chuck.rad(count).avg_h;
    end
    for count = 1:length(chuck.unshad)
        counter = counter + 1;
        summary.distance(counter) = chuck.unshad(count).avg_y;
        summary.thickness(counter) = chuck.unshad(count).avg_h;
    end
    numpoints = counter;


% Fit a line to the thickness values found at various y (radial) distances
% along the chuck
    tmp = polyfit(summary.distance,summary.thickness,1); m = tmp(1); b = tmp(2);
    summary.fit_thickness = summary.distance * m + b;
    summary.mid_thickness = b;
    summary.thickness_change = (summary.distance * m )/b*100;
    summary.thickness_change_gradient = ...
              summary.thickness_change(1) / summary.distance(1);


    figure;
    plot(summary.distance,summary.thickness,'ko','MarkerSize',12,'LineWidth',3);
    hold on;
    set(gca,'LineWidth',2,'FontWeight','bold','FontSize',20);
    plot(summary.distance,summary.fit_thickness,'k-','LineWidth',3);
    xlabel(['d_r (mm)'],'LineWidth',2,'FontWeight','bold','FontSize',18);
    ylabel(['t ( \mu m)'],'LineWidth',2,'FontWeight','bold','FontSize',18);
    print(gcf,'-djpeg',[plotDir,'\','Thickness']);



% Now put together a 3D thickness map using the unshadowed
% and shadowed data obtained from the SEM

    % First put together a grid of thickness versus radial and normal
    % distance from the chuck - for the case where no shadowing
    % block is considered
    chuck.x_grid = -chuck.radius:chuck.dr:chuck.radius;
    chuck.y_grid = -chuck.radius:chuck.dr:chuck.radius;
    chuck.thickness_change = 1 + chuck.y_grid * ...
                               summary.thickness_change_gradient/100;
    chuck.thickness_map = repmat(chuck.thickness_change,length(chuck.x_grid),1);

    % Now put together the shadow block effect
    chuck.shadow.normal_grid = 0:chuck.dr:chuck.shadStop;
    for count = 1:length(chuck.shad)
        chuck.shadow.spl_heights(count,:) =  ...
              spline(chuck.shad(count).x,chuck.shad(count).h,chuck.shadow.normal_grid);
        chuck.shadow.relHeights(count,:) = ...
```

```
          1-(avg_h-chuck.shadow.spl_heights(count,:))/ avg_h;
    chuck.shadow.radial_grid(count) = chuck.shad(count).avg_y;
end

% Plot the shadowing block profiles
figure;
colours = ['b','r','g','m','k','c','y'];
shapes = ['o','s','d','p','v','x','>','h','+'];
leg = [];
for count = 1:length(chuck.shad)
    plot(chuck.shad(count).x_orig,chuck.shad(count).h_orig, ...
            [colours(count),shapes(count)],'LineWidth',2,'MarkerSize',8);
    hold on;
    radial_grid(count) = chuck.shad(count).avg_y;
    distFromSB = radial_grid(count) - chuck.small_radius;
    if(distFromSB < 10)
         legentry = [' ',int2str(floor(distFromSB)),'.', ...
                 int2str(round(10*(distFromSB-floor(distFromSB)))/10),'mm'];
    else
         legentry = [int2str(floor(distFromSB)),'.', ...
                int2str(round(10*(distFromSB-floor(distFromSB)))),'mm'];
    end
    leg = [leg;legentry];
end
legend(leg,-1);
xlabel(['d_n (mm)'],'LineWidth',2,'FontSize',18,'FontWeight','bold');
ylabel('t ({\mu}m)','LineWidth',2,'FontSize',18,'FontWeight','bold');
for count = 1:length(chuck.shad)
    plot(chuck.shadow.normal_grid,chuck.shadow.spl_heights(count,:), ...
                 [colours(count),'-'],'LineWidth',2);
end
set(gca,'LineWidth',2,'FontWeight','bold','FontSize',18);
axis(axisParam);
print(gcf,'-djpeg',[plotDir,'\','Thicknesses_2d_frmSB']);



close all;


% We want the shadowing block thickness profile to be on the same
% grid as the "unshadowed" thickness gradient
% We therefore need to interpolate along y to get shadowing effect
chuck.shadow.spl_radial_grid = ...
      round(chuck.shadow.radial_grid):chuck.dr:chuck.radius;
for count = 1:length(chuck.shadow.normal_grid)
    polyfit_N = 4;
    x = chuck.shadow.radial_grid;
    y = chuck.shadow.relHeights(:,count)';
    for count2 = 1:10
        y = [y(1) y y(end)];
        x = [(x(1)-1) x  (x(end)+1)];
    end

    pp = polyfit(x,y,polyfit_N);
    xfit = chuck.shadow.spl_radial_grid;
    yfit = 0;
    for count2 = 1:(polyfit_N+1)
        yfit = yfit + pp(count2) * xfit.^(polyfit_N+1-count2);
    end

    chuck.shadow.spl_relHeights(:,count) = yfit;
end

% For the region where there was no SEM data, just
% use the same shadowing profile (scaled)



chuck.shadow.extrapolate_radial = 0:chuck.dr:(chuck.shadow.spl_radial_grid(1)-chuck.dr);

deltaHeights = (chuck.shadow.spl_relHeights(1,2:end) - ...
                chuck.shadow.spl_relHeights(1,1:(end-1))) ./ ...
         (chuck.shadow.normal_grid(2:end) - chuck.shadow.normal_grid(1:(end-1)));
[maxVal,maxInd] = max(deltaHeights);
maxPos = chuck.shadow.normal_grid(maxInd);
rate = (maxPos - shadow_r) / (chuck.shadow.spl_radial_grid(1)-chuck.dr - 0);
for count = 1:length(chuck.shadow.extrapolate_radial)
    position = rate * (count - 0) + shadow_r;
    shrink = round(100*maxPos/position);
    interp_vector = interp(chuck.shadow.spl_relHeights(1,:),100);
```

```
        shrunk_vector = interp_vector(1:shrink:end);
        diffL = length(chuck.shadow.spl_relHeights(1,:)) - length(shrunk_vector);
        chuck.shadow.extr_relHeights(count,:) = [shrunk_vector ones(1,diffL) * ...
                                                chuck.shadow.spl_relHeights(1,end)];
    end

    chuck.shadow.thickness_map = [chuck.shadow.extr_relHeights; ...
                                  chuck.shadow.spl_relHeights];
    chuck.shadow.comb_radial_grid = [chuck.shadow.extrapolate_radial, ...
                                     chuck.shadow.spl_radial_grid];
    chuck.shadow.thickness_map = [chuck.shadow.thickness_map(:,end:-1:2) ...
                                  chuck.shadow.thickness_map];
    chuck.shadow.comb_normal_grid = [-chuck.shadow.normal_grid(end:-1:2) ...
                                     chuck.shadow.normal_grid];

    % Combine the "unshadowed" and "shadowed" thickness maps
    tmp = find(chuck.shadow.comb_normal_grid(1) <= chuck.x_grid);
    x_index1 = tmp(1);
    tmp = find(chuck.shadow.comb_normal_grid(end) <= chuck.x_grid);
    x_index2 = tmp(1);
    tmp = find(chuck.shadow.comb_radial_grid(1) <= chuck.y_grid);
    y_index1 = tmp(1);
    tmp = find(chuck.shadow.comb_radial_grid(end) <= chuck.y_grid);
    y_index2 = tmp(1);

    scale1 = 1.12;
    scale2 = 1.5;
    N = size(chuck.shadow.thickness_map,1);
    scale_factor = linspace(scale1, scale2, N);
    scale_grid = repmat(scale_factor,size(chuck.shadow.thickness_map,2),1);

    chuck.thickness_map(x_index1:x_index2,y_index1:y_index2) = ...
              (scale_grid .* chuck.shadow.thickness_map') .* ...
              chuck.thickness_map(x_index1:x_index2,y_index1:y_index2);



    figure;
    set(gca,'LineWidth',2,'FontWeight','bold','FontSize',20);
    hold on;grid on;
    xlabel(['d_r (mm)'],'LineWidth',2,'FontWeight','bold','FontSize',20);
    ylabel(['d_n (mm)'],'LineWidth',2,'FontWeight','bold','FontSize',20);
    zlabel(['t_{rel}'],'LineWidth',2,'FontWeight','bold','FontSize',20);
    mesh(chuck.x_grid,chuck.y_grid,chuck.thickness_map,'LineWidth',2);
    view([0,90]);
    print(gcf,'-djpeg',[plotDir,'\','Rel_Thicknesses_3d_down']);
    for xangle = 0:15:90 %0:10:90
        for yangle = 90:-15:0 %90:-10:0
            view([xangle,yangle]);
            print(gcf,'-djpeg',[plotDir,'\','Rel_Thicknesses_3d_angle_', ...
                    num2str(xangle),'_',num2str(yangle)]);
        end
    end


    % Save final data to a file
    disp('Saving data to a file');
    fid = fopen([plotDir,'\Thickness_Profile.txt'],'w');

    fprintf(fid,'0 ,');
    for count = 1:length(chuck.x_grid)
        fprintf(fid,'%6.4f, ',chuck.x_grid(count));
    end
    fprintf(fid,'\n');

    for count = 1:length(chuck.y_grid)
        fprintf(fid,'%6.4f, ',chuck.y_grid(count));
        for count2 = 1:length(chuck.x_grid)
            fprintf(fid,'%6.4f, ',chuck.thickness_map(count2,count));
        end
        fprintf(fid,'\n');
    end
    fclose(fid);


    close all;

%=========================================================================================
```

D.2 Code to simulated shadowed chiral film thicknesses

```
clear all;
close all;
clc;

% ------------------------------------------------------------------------
% Declarations and Initialization
% ------------------------------------------------------------------------
% Shadown block casts a shadow
% Break this shadow into divisions
shadowBlk.numDivisions = 20;
shadowBlk.divisions = [(1/shadowBlk.numDivisions):(1/shadowBlk.numDivisions):1];


dataDir = ['C:\My_Documents\Research\Research_Areas\Graded\', ...
           'Shadowing_Simulations\8turn_20mmHigh_25mmRCylinder_Chiral60'];
fileName = 'Chiral-8per-a60.txt';
alpha_dep = 60;
thickness_fileName = 'Thickness_Profile.txt';
alpha_correction = 0;
h_small = 20;                          % Shadowing height (mm)
z_factor = 0.95;
SEM_x = [0    2.8740    5.9479    9.1517   13.9671   17.5604   22.4545 ...
         25.8072   30.1686   35.5707   41.4000   43.1000   46.9751   ...
         55.2292   58.7700];
SEM_y = 1000 * [2.0010    2.3020    2.4900    2.5830    2.7050    2.7780  ...
                2.8730    2.9090    3.0080    3.1080    3.1950    3.2590  ...
                  3.2590    3.2840    3.2690];
dist_min = 0;
dist_max = 60;
thick_min = 2000;
thick_max = 3500;



% dataDir = ['C:\Measurements\Shadowing\Shadowing_Simulations\', ...
%            '8turn_20mmHigh_25mmRCylinder_Chiral65'];
% fileName = 'Chiral-8per-a65.txt';
% alpha_dep = 65;
% shadowShape = 'square'; %'square';
% parPower = 20;
% correction = 1;                        % Correction to height of shadowing block
% alpha_correction = 3;                   % Correction to alpha
% h_small = 20;                            % Shadowing height (mm)
% thickness_correction = 0.88;
% shadowBlk.gradient = [1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0.9 0.7 0.5 0.3 0.2];
% prefix = '';
% SEM_x = [0,0.1920,0.8759,1.9982,3.8589,6.3014,8.7806,12.0406,15.3109, ...
%       18.1849,21.3934,23.8679,27.2417,31.2740,31.7680,33.9680,36.4156,38.3986, ...
%       43.0058,44.8977,45.7748,46.7235,47.9517,49.1203,51.0910,52.6879, ...
%       53.3732,53.8120];
% SEM_y = [1746,1913,2028,2111,2226,2334,2340,2469,2556,2596,2659,2699,2774,2854, ...
%          2830,2897,2937,2984,3068,3096,3108,3123,3139,3151,3175,3167,3171,3167];
% dist_min = 0;
% dist_max = 60;
% thick_min = 1500;
% thick_max = 3500;



% dataDir = ['J:\My_Documents\Research\Research_Areas\Graded\', ...
%            'Shadowing_Simulations\8turn_20mmHigh_25mmRCylinder_Chiral70'];
% fileName = 'Chiral-8per-a70.txt';
% alpha_dep = 70;
% thickness_fileName = 'Thickness_Profile.txt';
% alpha_correction = 0;
% h_small = 20;                          % Shadowing height (mm)
% z_factor = 0.95;
% SEM_x = [0,       0.1000,   0.1871,  0.3225,  3.7231,  9.1649,  ...
%          9.2063,11.8068,13.4840,14.3088,17.1593, ...
%          21.9209, 22.3011, 25.9871, 30.2502, 34.4698, 35.4698, 35.7702, ...
%          37.4494, 40.7190, 44.1196, 49.4541, 54.3146, 57.9930, 61.4423];
% SEM_y = [2163,2148,2202,2222,2520,2743,2773,2838,2862,2937,2967,3036,3081,3106, ...
%          3170,3220,3250,3260,3260,3290,3314,3344,3395,3443,3463];
% dist_min = 0;
% dist_max = 60;
% thick_min = 2000;
% thick_max = 3500;
```

```
% dataDir = ['C:\Measurements\Shadowing\Shadowing_Simulations\', ...
%             '8turn_20mmHigh_25mmRCylinder_Chiral75'];
% fileName = 'Chiral-8per-a75.txt';
% alpha_dep = 75;
% alpha_correction = 0;
% h_small = 20;                          % Shadowing height (mm)
% SEM_x = [0,1.3357,2.6175,5.1099,7.2599,10.2619,14.7116,16.6297,18.2926, ...
%        20.4025,23.0408,26.4087,27.5939,30.1060,31.6276,33.8276,33.8997,36.4659, ...
%        38.3353,40.3390,42.4809,44.7658,47.1679,48.4983,49.4991,50.6032,51.4560, ...
%        52.1045,54.6798,56.3939];
% SEM_y = [1695,1893,2028,2179,2302,2405,2520,2584,2600,2627,2671,2703,2719,2742, ...
%       2723,2758,2768,2798,2810,2806,2814,2850,2881,2889,2893,2881,2881,2909,2921,2941];
% dist_min = 0;
% dist_max = 60;
% thick_min = 1500;
% thick_max = 3000;


% dataDir = ['J:\My_Documents\Research\Research_Areas\Graded\', ...
%             'Shadowing_Simulations\8turn_20mmHigh_25mmRCylinder_Chiral85'];
% fileName = 'Chiral-8per-a85.txt';
% alpha_dep = 85;
% thickness_fileName = 'Thickness_Profile.txt';
% h_small = 20;                          % Shadowing height (mm)
% alpha_correction = 0;
% z_factor = 1.07;
% SEM_x = [1.2438, 1.9246, 3.4979, 6.3579, 12.6759, 16.7161, 20.1830, 23.9511, ...
%          28.2918, 32.7063, 36.6696, 39.0616, 40.8657, 44.8166, 48.9894, 52.7173, ...
%          56.3953, 59.4431, 63.4589];
% SEM_y = [2.3460, 2.4290, 2.6000, 2.7860, 3.0000, 3.1110, 3.1870, 3.2980, ...
%          3.3650, 3.4330, 3.4590, 3.4950, 3.5630, 3.5800, 3.5940, 3.5900, ...
%          3.6620, 3.6710, 3.6670]*1000;
% dist_min = 0;
% dist_max = 60;
% thick_min = 2000;
% thick_max = 4000;



% Substrate chuck and shadowing block dimensions
dr = 0.1;
r_big = 84;                         % Big substrate chuck radius (mm)
r_small = 24;                       % Small shadow block radius (mm)
r_remain = r_big - r_small;         % Distance on chuck for substrates (mm)

% Growth assumptions
diameter_vs_alpha = [1:-1/90:0];    % When growing rugate films alpha is changed
                                    % As alpha is changed the diameter of
                                    % the pillars change
                                    % Assume a linear relationship between
                                    % column diameter and alpha (0 --> 90)
                                    % in order to predict the structure in
                                    % this simulation



testName = ['CalModel'];


% ------------------------------------------------------------------------
% Program
% ------------------------------------------------------------------------

tmp = csvread([dataDir,'/',thickness_fileName]);

calibration.profile = tmp(2:end,2:(end-1));
calibration.x_axis = tmp(1,2:(end-1))';
calibration.y_axis = tmp(2:end,1);
%figure;mesh(calibration.x_axis,calibration.y_axis,calibration.profile);


fid=fopen([dataDir,'/',fileName]);
th_index = 0;
while 1
    tline = fgetl(fid);
    if ~ischar(tline), break, end
    th_index = th_index + 1;
    tmp = str2num(tline);
```

```
    % Thickness in nm -- convert to mm
    orig_thickness(th_index) = tmp(1);
    orig_alpha(th_index) =  tmp(2) + alpha_correction;
    orig_phi(th_index) =   tmp(3);
end
fclose(fid);


% Use the motion file thickness/alpha/phi data
% However, only simulate a subset of points in order to speed up the
% simulation
steps = [1:1:length(orig_thickness)];
interp_points = 200;
spline_steps = [1/interp_points:1/interp_points:length(orig_thickness)];

thickness = spline([1:1:length(orig_thickness)],orig_thickness, ...
                   [1:(1/interp_points):length(orig_thickness)]);
alpha = spline(orig_thickness,orig_alpha,thickness);
phi = spline(orig_thickness,orig_phi,thickness);

% One "GLAD" column is simulated for each mm between the
% shadow block and edge of the main substrate chuck
column_loc.radial = [1:1:r_remain];

% Now simulate substrate motion and the effect of shadowing
for step = 2:length(thickness)
    if (step/100 == round(step/100))
        disp(['Step: ',num2str(step),', Thickness: ', ...
                num2str(thickness(step)),', Alpha: ', ...
                 num2str(alpha(step)),', Phi: ',num2str(phi(step))]);
    end


    % The location of the column in the x and y plane in calculated for the
    % given phi
    column_loc.x = sin(rem(phi(step) ,360) * pi / 180) * (column_loc.radial + r_small);
    column_loc.y = cos(rem(phi(step) ,360) * pi / 180) * (column_loc.radial + r_small);

    % Now you need to figure out where each of the above GLAD columns fall
    % in terms of the substrate.
    for count = 1:length(column_loc.radial)
        [index] = find(calibration.x_axis >= column_loc.x(count));
        if (isempty(index))
            % Column is outside of chuck!
            x_index_lower = length(calibration.x_axis);
            x_index_upper = x_index_lower;
        else
            x_index_upper = index(1);
            if (index(1) == 1)
                x_index_lower = x_index_upper;
            else
                x_index_lower = index(1)-1;
            end
        end
        [index] = find(calibration.y_axis >= column_loc.y(count));
        if (isempty(index))
            % Column is outside of chuck!
            y_index_lower = length(calibration.y_axis);
            y_index_upper = y_index_lower;
        else
            y_index_upper = index(1);
            if (index(1) == 1)
                y_index_lower = y_index_upper;
            else
                y_index_lower = index(1)-1;
            end
        end

        if (x_index_lower == x_index_upper)
            if (y_index_lower == y_index_upper)
                thickness_val(count) = calibration.profile(x_index_lower,y_index_lower);
            else
                val_l = calibration.profile(x_index_lower,y_index_lower);
                val_u = calibration.profile(x_index_lower,y_index_upper);
                m = (val_u - val_l) / (calibration.y_axis(y_index_upper) - ...
                                       calibration.y_axis(y_index_lower));
                thickness_val(count) = m * (column_loc.y(count) - ...
                                            calibration.y_axis(y_index_lower)) + val_l;
            end
        else
```

```
            if (y_index_lower == y_index_upper)
                val_l = calibration.profile(x_index_lower,y_index_lower);
                val_u = calibration.profile(x_index_upper,y_index_lower);
                m = (val_u - val_l) / (calibration.x_axis(x_index_upper) - ...
                                       calibration.x_axis(x_index_lower));
                thickness_val(count) = m * (column_loc.x(count) - ...
                                            calibration.x_axis(x_index_lower)) + val_l;
            else
                val_ll = calibration.profile(x_index_lower,y_index_lower);
                val_lu = calibration.profile(x_index_lower,y_index_upper);
                val_ul = calibration.profile(x_index_upper,y_index_lower);
                val_uu = calibration.profile(x_index_upper,y_index_upper);

                ratio_x = (calibration.x_axis(x_index_upper) - column_loc.x(count)) / ...
                          (calibration.x_axis(x_index_upper) - ...
                                calibration.x_axis(x_index_lower));
                ratio_y = (calibration.y_axis(y_index_upper) - column_loc.y(count)) / ...
                          (calibration.y_axis(y_index_upper) - ...
                                calibration.y_axis(y_index_lower));
                thickness_val(count) = ratio_x * ratio_y * val_ll + ...
                                ratio_x * (1-ratio_y) * val_lu + ...
                                (1-ratio_x) * ratio_y * val_ul + ...
                                (1-ratio_x) * (1-ratio_y) * val_uu;
            end
        end

    end

    columns.delta_thickness(step,:) = thickness_val * z_factor *...
                                       (thickness(step)-thickness(step-1));
    % Predict the GLAD column diameters
    % Nothing fancy done here and no effect of shadowing is modelled
    columns.diameter(step,:) = diameter_vs_alpha(round(alpha(step)))* ...
                                       length(column_loc.radial);

end

% Find the total thickness of each GLAD column
columns.thickness = cumsum(columns.delta_thickness);
columns.total_thickness = columns.thickness(end,:);

% Plot the GLAD column thickness as a function of distance from shadow
% block -- radially outward from shadow block
figure;
plot(SEM_x,SEM_y,'ko','MarkerSize',12,'LineWidth',2);
hold on;
plot(column_loc.radial(1:2:end),columns.total_thickness(1:2:end),'k--', ...
          'MarkerSize',11,'LineWidth',2);
set(gca,'LineWidth',1.5,'FontName','Arial','FontSize',24) %,'FontWeight','bold'
xlabel('Distance from Shadow Block Edge [mm]','FontName','Arial','FontSize',24);
ylabel('Thickness [nm]','FontName','Arial','FontSize',24);
print('-dtiff','-r800',[dataDir,'\GLAD_column_thickness_',testName,'_simLine']);

close all;

fid = fopen([dataDir,'\Exp_Sim_',num2str(alpha_dep),'.txt'],'w');
fprintf(fid,'Distance (mm), Thickness Experiment (mm) \n');
for count = 1:length(SEM_x)
    fprintf(fid,'%6.2f, %6.2f \n',SEM_x(count),SEM_y(count));
end
fprintf(fid,'Distance (mm), Thickness Simulation (mm) \n');
for count = 1:length(column_loc.radial)
    fprintf(fid,'%6.2f, %6.2f \n',column_loc.radial(count),columns.total_thickness(count));
end
fclose(fid);


%================================================================================
```