UNIVERSITY OF ALBERTA

Expert Long Range Predictive Control

BY

Roman Walther

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES AND RESEARCH

IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR

THE DEGREE OF

Master of Science

IN

Process Control

Department of Chemical Engineering

EDMONTON, ALBERTA

Spring 1992

ISBN 0-315-73183-4

UNIVERSITY OF ALBERTA

RELEASE FORM

NAME OF AUTHOR:	Roman Walther
TITLE OF THESIS:	Expert Long Range Predictive Control
DEGREE:	Master of Science
YEAR THIS DEGREE IS GRANTED:	1992

(SIGNED)

Roman Walther
....

9318 - 81 Street

Fort Saskatchewan, AB

T8L 3L2

DATED: 21 April 1992

UNIVERSITY OF ALBERTA

FACULTY OF GRADUATE STUDIES AND RESEARCH

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research for acceptance, a thesis entitled *Expert Long Range Predictive Control* submitted by *Roman Walther* in partial fulfilment of the degree of *Master of Science* in *Process Control.*

[illegible]

Dr. S. L. Shah (supervisor)

[illegible]

Dr. K. E. Bollinger

[illegible]

Dr. D. G. Fisher

[illegible]

Dr. M. Rao

Dated 21 April 1992

to my parents and to Samar

Abstract

Long Range Predictive Control (LRPC) has been applied to multi-input multi-output processes for at least fifteen years. The fundamental problem of all LRPC algorithms (i.e. DMC of Cutler and Ramaker, 1980, MOCCA of Sripada and Fisher, 1985, and GPC of Clarke *et al.*, 1987), however, remains the degradation of control performance in the presence of model-plant mismatch and disturbances. This thesis addresses this issue through the concept of expert supervisory LRPC.

Expert supervisory LRPC integrates two academic fields of endeavour, namely process control and expert systems. The contributions of this thesis to the field of process control are twofold. Firstly, this work identifies LRPC tuning parameters suitable for adjustment in an on-line real-time environment for the purpose of performance tuning. And secondly, the commercial object-oriented real-time expert system development tool, G2 by Gensym, is used to implement a prototype LRPC performance supervisor. This prototype performance supervisor is called the Adaptive Long range predictive control Performance Supervisor, ALPS.

ALPS supervises the closed-loop control performance of several multivariable processes simultaneously. ALPS is interfaced to one member of the family of LRPC algorithms, namely constrained MGPC of Mutha (1990), in an on-line real time environment via the G2 Standard Interface (GSI). Both ALPS and the MGPC algorithm are generic and can be applied to multi-input multi-output processes of dimension n × m, where n is the number of process outputs and m is the number of process inputs. ALPS reflects the rules and procedures a seasoned process operator uses in adjusting LRPC tuning parameters in order to maintain user specified heuristic performance criteria.

ALPS is evaluated based upon simulation results of 2-input 2-output benchmark problems subjected to conditions of model-plant mismatch and disturbances. These investigations conclusively show that ALPS skilfully adjusts the LRPC tuning parameters. ALPS monitors the actual closed-loop control performance and proficiently manipulates the LRPC tuning parameters so as to achieve and maintain user specified closed-loop regulatory and servo control performance. The results also indicate that the success of ALPS depends upon the user supplied performance specifications. ALPS recognizes its limitations and achieves and maintains the desired closed-loop control performance only if the user supplies performance specifications that are realistic and achievable.

Acknowledgements

Although only one name appears as the author of this book, it is very much the product of many people. I thank my wife Samar for her patience, love, and support. I offer my sincere thanks to Dr. S.L. Shah and Dr. D.G. Fisher for their guidance and supervision throughout this research. I would also like to thank all the graduate students at the University of Alberta who have made the last two years enjoyable and productive. I owe much to discussions with Rajendra Mutha, Viral Maniar, David Shook, and my "office mate" Fengxi Zhou. Special thanks are also due to Henry Sit of the DACS centre.

The financial support from the Natural Sciences and Engineering Research Council of Canada, and the Department of Chemical Engineering is gratefully acknowledged.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

AI	Artificial Intelligence
ALPS	Adaptive Long range predictive control Performance Supervisor
DMC	Dynamic Matrix Control
ES	Expert System
GPC	Generalized Predictive Control
GSI	G2 Standard Interface
MGPC	Multivariable Generalized Predictive Control
MOCCA	Multivariable Optimal Constrained Control Algorithm
MPM	Model Plant Mismatch
KB	Knowledge Base
LRPC	Long Range Predictive Control

Chapter 1

Introduction

Long Range Predictive Control strategies are enjoying widespread acceptance and are routinely applied to full-scale industrial processes (Garcia *et al.*, 1988, Shah *et al.*, 1987). Of particular interest is the fact that these controllers are insensitive to process dead time and readily accommodate non-minimum phase plants. Dimensionally large multivariable implementations (see Cutler and Hawkins, 1988, for a 16×9 implementation, and Kelly *et al.*, 1988, for a 5×6 process) give LRPC techniques tremendous clout. The main thrust of this work is to address the fundamental problem of all LRPC algorithms which is the degradation of closed-loop control performance subsequent to commissioning. Both model-plant mismatch (MPM) and slow changing process dynamics cause closed-loop control performance to deteriorate. One goal of this work is to develop a performance supervisor whose role is to monitor the actual closed-loop control performance of the underlying LRPC algorithm and take appropriate actions so as to maintain user specified control performance.

The idea of using a closed-loop control performance supervisor is not new. Minter and Fisher (1988), through their work on commercially available expert adaptive controllers for the Proportional plus Integral plus Derivative action (PID) algorithm, conclude that the addition of a closed-loop performance supervisor to a control algorithm yields significantly better control performance over a broader operating region. A closed-loop performance supervisor is also considered by McIntosh (1988). He points out that even under ideal conditions of no MPM and no disturbances, the appropriate controller settings resulting in the desired closed-loop control performance may not always be known *a priori*. His implementation of a single-input single-output (SISO) unconstrained Generalized Predictive Control (GPC) performance supervisor illustrates that subsequent to commissioning, the GPC tuning parameters may be adjusted on-line in order to achieve the desired closed-loop control performance.

In recent years research efforts in the field of Artificial Intelligence (AI) have resulted in great advances in Expert Systems (ES) technology. ES are a subset of Knowledge Based Systems which themselves form a large subset of AI. ES allow qualitative rules-of-thumb or heuristics, as applied by a human expert to solve domain specific problems, to be captured and encoded into a computer program. The benefits of this technology can be tremendous. Firstly, the captured knowledge remains accessible should the human expert be transferred to another department or leave the company. Secondly, the knowledge base is easily expanded to consolidate the know-how of several human experts thereby increasing the complexity of the problems that can be solved. And lastly, given a particular problem the conclusions and recommendations made by an ES are more consistent than the recommendations made by several different human experts faced with the same problem (e.g. given a particular set of circumstances, the ES will always conclude with the same recommendations).

Although some ES implementations in the field of process control are disclosed in the literature (e.g. Åström *et al.* 1986, Sripada *et al.* 1987, Aynsley *et al.* 1989, Doraiswami *et al.* 1987, Haest *et al.* 1990, Dong *et al.* 1991), many more industrial applications go unreported due to the proprietary nature of these systems. As each application must reflect its own purpose, each implementation is either an off-line application, an on-line application, or an on-line application with real-time computing. Off-line applications operate without a direct communication link to external devices (i.e. the user manually enters all input data). Many of these applications are decision support systems taking the form of question and answer sessions. In these cases the ES application prompts the user to supply all the required information. Upon termination of the reasoning process it is the user who, at his discretion, acts upon all or some of the recommendations made. An application of this type is described by Dong *et al.* (1991). The authors report industrial success of their system which provides decision support to process operators during plant startup.

On-line applications address the circumstances where neither process data nor processing conditions remain constant for a long period of time. Process control applications in particular are time-varying due to the dynamic nature of the plant (i.e. temperatures, pressures, and flows change with respect to time). On-line implementations are interfaced to devices such as sensors in order to access plant data directly. Often only a small fraction of the information is entered by users or process operators. The ES performs its reasoning and inferencing exploiting the history and dynamic nature of plant data. This is usually referred to as temporal reasoning. Adams *et al.* (1987) present their on-line advisory system which performs process monitoring for the purpose of fault detection and diagnostics. This ES application, similar to most on-line advisory or decision support systems, does not directly take actions (e.g. changing process setpoints, or adjusting controller tuning parameters). Rather, these systems provide intelligent recommendations. Ultimately it is the process operator who - with the aid of the ES - decides upon the necessary actions, and implements these actions.

In contrast to both on-line and off-line ES applications, on-line real-time ES applications take direct action with minimal operator intervention. Process control applications utilizing on-line real-time ES technology range from controller tuning, adaptation, monitoring and diagnostics to supervision (e.g. Aynsley *et al.* 1989, Doraiswami and Jiang 1989, Årzén 1989, and Åström *et al.* 1986). This thesis is devoted exclusively to the latter of these issues, namely supervision. The primary contribution of this thesis is the development of the Adaptive Long range predictive control Performance Supervisor, ALPS. ALPS is implemented with the commercial real-time ES development tool G2 and performs closed-loop control performance supervision of several multivariable constrained LRPC algorithms. ALPS manipulates LRPC tuning parameters in a fashion similar to a very proficient human process operator.

1.1 Objective of Study

The development and implementation of an expert LRPC control performance supervisor integrates two academic fields of endeavour, namely process control and expert systems. The contributions of this thesis to the field of process control are two fold. Firstly, this work identifies LRPC tuning parameters suitable for adjustment in an on-line real-time environment. Secondly, a commercial object-oriented real-time expert system development tool is used to implement a prototype LRPC performance supervisor. This prototype performance supervisor is called the Adaptive Long range predictive control Performance Supervisor, ALPS. In brief, the objectives of this thesis are:

- implement one member of the family of LRPC algorithms as a multivariable controller having n process outputs and m process inputs, with constraints on input amplitude, incremental rate of change, and output amplitude
- quantify LRPC tuning parameters suitable for on-line adjustment for the purpose of performance tuning
- develop, implement, and demonstrate a LRPC performance supervisor utilizing the commercial real-time expert system development tool G2 by Gensym
- evaluate the prototype LRPC performance supervisor based on benchmark problems under conditions of model-plant mismatch and disturbances.

1.2 Structure of the Thesis

The structure of the thesis is outlined below with emphasis on the contributions of this work. A more detailed introduction as well as a summary is given at the beginning and end of each individual chapter.

Chapter 2 reviews popular multivariable LRPC algorithms (e.g. DMC, MOCCA, MGPC) utilizing a consistent nomenclature. It is shown that parametric and non-parametric process model predictive algorithms yield the identical control law. This result accentuates the intimate relationship among these long range predictive control algorithms. The chapter concludes by selecting MGPC for implementation due to its two degrees of freedom structure (i.e. MGPC allows for separate regulatory control and servo control performance specifications).

Chapter 3 quantifies both regulatory and servo control performance criteria, and selects suitable MGPC tuning parameters. The selection of MGPC tuning parameters is substantiated through the use of an illustrative example.

Chapter 4 focuses on the implemented LRPC performance supervisor, ALPS. It highlights the rules and procedures utilized by ALPS to perform regulatory and servo control performance tuning. In addition, this chapter provides the functional specifications of ALPS in comparison to a generic Expert Supervisor also developed in this chapter. The training facilities provided by ALPS to train personnel in the use of ALPS itself, as well as the underlying LRPC algorithms are also presented. A brief description of the commercial object-oriented real-time expert system development tool G2, used to implement ALPS, is also provided.

Chapter 5 demonstrates and evaluates the tuning strategies employed by ALPS. Three benchmark problems, subjected to conditions of MPM and disturbances, are used to investigate the performance of ALPS. These investigations show that ALPS adjusts LRPC tuning parameters in a competent manner. The results also exemplify the underlying assumptions that govern the success of ALPS.

Chapter 6 draws conclusions from the results presented in this thesis and provides suggestions for future work.

Chapter 2

Long Range Predictive Control - a unified framework

Long range predictive control algorithms represent a family of predictive controllers sharing several important features. This chapter highlights these similarities by considering non-parametric process model control algorithms as well as parametric process model control algorithms. Dynamic Matrix Control (DMC) (Cutler and Ramaker, 1979) and Multivariable Optimal Constrained Control Algorithm (MOCCA) (Sripada and Fisher, 1985) use step response models in their description of the process and are considered as examples of non-parametric LRPC algorithms. The Generalized Predictive Control (GPC) (Clarke *et al.*, 1987) uses an Auto-Regressive Integrated Moving Average with Exogenous Inputs (ARIMAX) process model and is considered as a representative of parametric LRPC algorithms. Parametric control algorithms lend themselves to adaptive implementations, however, for the purposes of this thesis GPC is considered as a fixed parameter non-adaptive control algorithm.

2.1 Introduction to LRPC

The control laws of different LRPC algorithms initially appear to be quite different. To a large degree these differences can be attributed to a non-standardized nomenclature. Upon closer inspection, the following key features can be extracted:

i) Process description

The mathematical description of the process can be parametric (e.g. GPC which uses the ARIMAX process model) or non-parametric (e.g. DMC and MOCCA which use step response coefficients).

ii) Objective function

The control action to be implemented is calculated based upon the minimization of an objective function. This objective function is expressed in terms of the minimization of the *predicted* discrepancy between the process outputs and their respective setpoints over a finite trajectory into the future. The objective function can be in the 1-norm, 2-norm, or ∞-norm (Garcia *et al.*, 1989). With the addition of input and/or output constraints the 1-norm and ∞-norm objective functions yield Linear Programming (LP) problems, and 2-norm formulations yield Quadratic Programming (QP) problems. Hence the form of the objective function has a significant impact on the control law.

iii) Process output prediction

Inherent to the use of an objective function is the need for output prediction. The minimization of the objective function is performed over a finite multi-step horizon requiring long range prediction. The prediction algorithm must be tailored to the specific process description selected.

iv) Finite control horizon and receding horizon control strategy

The assumption of a finite control horizon, after which all projected control increments are assumed to be zero, greatly reduces the computational effort. Furthermore, the use of a receding horizon control strategy, which implements only the first control action of the solution at every iteration interval, is common to all LRPC algorithms.

Figure 2.1 shows a simplified block diagram of the outlined LRPC strategy. The Predictor and Control Law blocks, as shown in this figure, are considered in detail in the following sections.

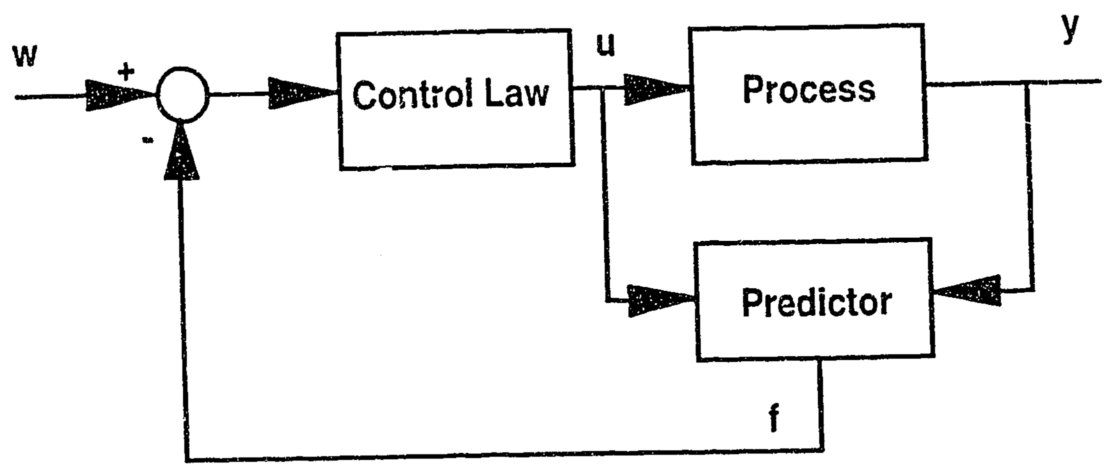


Figure 2.1 Simplified block diagram of LRPC strategy.

2.2 Multivariable non-parametric LRPC

This section presents multivariable non-parametric process model LRPC algorithms. In particular, non-parametric process models based on step response coefficients, such as DMC and MOCCA, are considered. Details specific to DMC and MOCCA can be found in the literature (see Cutler and Ramaker, 1980 for DMC, Sripada and Fisher, 1985 and Li *et al.*, 1989 for MOCCA). Although the nomenclature has been altered to become more intuitive, the derivations follow as per Li *et al.* (1989).

2.2.1 The step response process model

The step response process model for an n-output m-input system is represented by

$$y(t) = \sum_{i=1}^{N} G_{i-1} \Delta u(t-i) + G_N u(t-N-1) \tag{2.1}$$

where

y is the deviational output vector of dimension n × 1

u is the deviational input vector of dimension m × 1

Δ is the difference operator, $\Delta = 1 - q^{-1}$

G_i is the a matrix of dimension n × m, the elements of which are the step response coefficients of the individual input-output relationships at the ith sampling interval.

N is the truncation order and reflects the number of step response coefficients taken such that G_N is a reasonable approximation of the final or steady state value of the process.

2.2.2 The objective function

The objective function to be minimized is given by:

$$\min J = \sum_{j=1}^{P} [\hat{y}(t+j) - w(t+j)]^T \Gamma [\hat{y}(t+j) - w(t+j)] + \sum_{j=1}^{M} \Delta u(t+j-1)^T \Lambda \Delta u(t+j-1) \tag{2.2}$$

where

$\mathbf{w}(t+j)$ is a vector of setpoints of dimension $n \times 1$ such that

$\mathbf{w}(t+j) = [w_1(t+j) \ldots w_n(t+j)]^T$

$\hat{\mathbf{y}}(t+j)$ is a vector of predicted outputs of dimension $n \times 1$ such that

$\hat{\mathbf{y}}(t+j) = [\hat{y}_1(t+) \ldots \hat{y}_n(t+j)]^T$

$\Delta\mathbf{u}(t+j)$ is the input vector of dimension $m \times 1$ such that

$\Delta\mathbf{u}(t+j) = [\Delta u_1(t+j) \ldots \Delta u_m(t+j)]^T$

P is the output prediction horizon and $P < N$

M is the control horizon and $M \le P$

Λ is the diagonal control weighting matrix of dimension $m \times m$ allowing separate specification for each input given by:

$$\Lambda = \begin{bmatrix} \lambda_1 & & 0 \\ & \ddots & \\ 0 & & \lambda_m \end{bmatrix} \tag{2.3}$$

Γ is the diagonal output weighting matrix of dimension $n \times n$ allowing separate specification for each output given by:

$$\Gamma = \begin{bmatrix} \gamma_1 & & 0 \\ & \ddots & \\ 0 & & \gamma_n \end{bmatrix} \tag{2.4}$$

Equation 2.2 is not in its most general form. For example, output weighting and setpoint prefiltering can also be accommodated in this objective function. The weighting matrices Γ and Λ can also be made a function of the prediction horizon.

2.2.3 Process output prediction

The objective function given by equation 2.2 minimizes the predicted error between the process outputs and their respective setpoints over the prediction horizon P into the future. This requires the calculation of the predicted process outputs $\hat{\mathbf{y}}$ along the prediction horizon. Using the arguments of Garcia *et al.* (1989) this prediction can be split into three parts such that:

$$\hat{\boldsymbol{y}}(t+j) = G_{j-1}\Delta\boldsymbol{u}(t+j-1) + \boldsymbol{f}(t+j) + \boldsymbol{d}(t+j) \tag{2.5}$$

where

$\mathbf{f}(t+j)$ is the "free" response of the process at the jth sampling instant into the future and consists of the predicted process outputs assuming *no* future changes in the control action $\Delta\mathbf{u}$, and

$G_{j-1}\,\Delta\mathbf{u}(t+j-1)$ is the predicted process output based upon future control action yet to be calculated, and

$\mathbf{d}(t+j)$ is the vector defining the effect of future disturbances.

In order to predict the *future* disturbances **d**(t+j) consider the disturbance prediction technique employed with DMC where:

$$\mathbf{d}(t+j) = \mathbf{d}(t)$$
$$\text{for } j=1,2 \ldots P$$

and $\mathbf{d}(t)$ is estimated by the relationship:

$$\boldsymbol{d}(t) = \boldsymbol{y}_m(t) - \boldsymbol{y}(t) \tag{2.6}$$

where

$\mathbf{y}_m$ are the measured process outputs, and

$\mathbf{y}$ are the process outputs as calculated with equation 2.1

Using the above definitions equation 2.5 can be rewritten in its compact form:

$$\hat{\boldsymbol{y}} = \boldsymbol{G}_d \boldsymbol{\Delta u} + \boldsymbol{f} + \boldsymbol{d} \tag{2.7}$$

where

$\hat{\mathbf{y}}$ is the output vector along the entire prediction horizon of dimension $(n\ P) \times 1$, and includes the impact of future control actions, and

$\Delta\mathbf{u}$ is the input vector over the entire control horizon of dimension $(m\ M) \times 1$, and

$\mathbf{G}_d$ is the dynamic matrix of the process of dimension $(n\ P) \times (m\ M)$ and consists of

$$G_d = \begin{bmatrix} G_0 & 0 & 0 & \cdots & 0 \\ G_1 & G_0 & 0 & \cdots & 0 \\ G_2 & G_1 & G_0 & 0 & \vdots \\ & & & \ddots & 0 \\ \vdots & \vdots & \vdots & & G_0 \\ & & & & \vdots \\ G_{P-1} & G_{P-2} & G_{P-3} & \cdots & G_{P-M} \end{bmatrix} \tag{2.8}$$

Each G_i is a submatrix of dimension n × m, the elements of which are the step response coefficients of the ith sampling interval of each individual input-output relationship.

The free response of the process **f** is obtained from:

$$\begin{bmatrix} f(t+1) \\ f(t+2) \\ \vdots \\ f(t+P) \end{bmatrix} = G_N \begin{bmatrix} u(t-N) \\ u(t-N+1) \\ \vdots \\ u(t-N+P-1) \end{bmatrix} + \begin{bmatrix} G_{N-1} & G_{N-2} & & \cdots & G_1 \\ 0 & G_{N-1} & & \cdots & G_2 \\ \vdots & & \ddots & & \vdots \\ 0 & \cdots & 0 \quad G_{N-1} & \cdots & G_P \end{bmatrix} \begin{bmatrix} \Delta u(t-N+1) \\ \Delta u(t-N+2) \\ \vdots \\ \Delta u(t-1) \end{bmatrix} \tag{2.9}$$

2.2.4 The control law

The unconstrained solution to equation 2.2 is given by:

$$\Delta u(t) = \begin{bmatrix} I_m & 0 & \cdots & 0 \end{bmatrix} \left[G_d^T \Gamma G_d + \Delta \right]^{-1} G_d^T \Gamma (w - f) \tag{2.10}$$

where

$\mathbf{w}$ is the setpoint trajectory of dimension $(P\ n) \times 1$ given by

$$\mathbf{w} = [\mathbf{w}(t+1) \ldots \mathbf{w}(t+P)]^T$$

$\mathbf{f}$ is the free response of dimension $(P\ n) \times 1$ given by

$$\mathbf{f} = [\mathbf{f}(t+1) \ldots \mathbf{f}(t+P)]^T$$

$\underline{\Gamma}$ is a matrix of P block diagonal Γ matrices

$\underline{\Lambda}$ is a matrix of M block diagonal Λ matrices

$\mathbf{G_d}$ is the dynamic matrix of the process given by equation 2.8

2.2.5 Extensions to non-parametric LRPC algorithms

The preceding sections present non-parametric step-response LRPC algorithms, such as DMC and MOCCA, in their simplest form. Further research activity on these algorithms have resulted in more sophisticated step-response LRPC algorithms. The use of Kalman filtering techniques in disturbance prediction has been presented by Sripada and Fisher (1985) as well as Li *et al.* (1989). The solution to DMC subject to constraints (i.e. QDMC) is discussed by Garcia and Morshedi (1986).

2.3 Multivariable GPC

The multivariable generalized predictive control algorithm (MGPC) uses a parametric process model and is presented in this section. Details of this algorithm can be found in the literature (Mohtadi *et al.*, 1991, Shah *et al.*, 1987, and Mutha, 1990). The derivations follow as per Mohtadi *et al.* (1991).

2.3.1 The ARIMAX process model

The ARIMAX process model for an n-output m-input system is represented by

$$\boldsymbol{A}y(t) = \boldsymbol{B}u(t-1) + \frac{\boldsymbol{C}e(t)}{\Delta} \tag{2.11}$$

where

A and C are diagonal polynomial matrices of dimension n × n

B is a polynomial matrix of dimension n × m

y is the output vector of dimension n × 1

e is the noise vector of dimension n × 1 having zero-mean

u is the input vector of dimension m × 1

Δ is the difference operator, $\Delta = 1 - q^{-1}$

An arbitrary diagonal element ii of matrix A is given by:

$$A(q^{-1}) = 1 + a_1 q^{-1} + \cdots + a_{\delta A} q^{-\delta A} \tag{2.12}$$

and an arbitrary element ij of matrix **B** is given by:

$$B(q^{-1}) = b_0 + b_1 q^{-1} + \cdots + b_{\delta B} q^{-\delta B} \tag{2.13}$$

where the subscripts ii and ij have been dropped for the sake of brevity. Should any of the input-output channels of the process have a non-zero dead-time then the leading coefficients of the corresponding polynomial in matrix **B** are zero. For simplicity of the derivations set the coefficient matrix of the noise vector to identity (e.g. $\mathbf{C} = \mathbf{I}$); the implications of this are discussed in section 2.3.5.1 .

By setting e = 0 (its expected value) the ARIMAX model of equation 2.11 reduces to the Auto-Regressive Moving Average (ARMA) model:

$$y(t) = (\boldsymbol{A}\boldsymbol{\Delta})^{-1}\boldsymbol{B}\boldsymbol{\Delta}\boldsymbol{u}(t-1) \tag{2.14}$$

where $(\boldsymbol{A}\boldsymbol{\Delta})^{-1}\boldsymbol{B}$ are the step response coefficients of the process.

2.3.2 The objective function

The MGPC objective function to be minimized is given by:

$$\begin{aligned}\min J_{MGPC} = &\sum_{j=N_1}^{N_2} [\hat{y}(t+j) - w(t+j)]^T \Gamma [\hat{y}(t+j) - w(t+j)] \\ &+ \sum_{j=1}^{NU} \Delta u(t+j-1)^T \Lambda \Delta u(t+j-1)\end{aligned} \tag{2.15}$$

where

w(t+j) is a vector of setpoints of dimension n × 1 such that

$$w(t+j) = [w_1(t+j) \ \ldots \ w_n(t+j)]^T$$

$\hat{y}(t+j)$ is a vector of predicted outputs of dimension n × 1 such that

$$\hat{y}(t+j) = [\hat{y}_1(t+j) \ \ldots \ \hat{y}_n(t+j)]^T$$

$\Delta u(t+j)$ is the input vector of dimension m × 1 such that

$\Delta \mathbf{u}(t+j) = [\Delta u_1(t+j) \ldots \Delta u_m(t+j)]^T$

N_1 is the minimum prediction horizon

N_2 is the maximum prediction horizon

NU is the control horizon

Λ is the diagonal control weighting matrix of dimension m × m allowing separate specification for each input given by equation 2.3

Γ is the diagonal output weighting matrix of dimension n × n allowing separate specification for each output given by equation 2.4

2.3.3 Process output prediction

The MGPC objective function given by equation 2.15 minimizes the predicted error between the process outputs and their respective setpoints over the prediction horizon from N_1 to N_2. This requires the calculation of the predicted process outputs $\hat{\mathbf{y}}$ for the prediction horizon of interest. Using the standard arguments of Mohtadi (1987) this prediction can be split into two parts such that:

$$\hat{\boldsymbol{y}} = \boldsymbol{G_d} \boldsymbol{\Delta u} + \boldsymbol{f} \tag{2.16}$$

where

$\hat{\mathbf{y}}$ is the output vector along the entire prediction horizon of dimension $(N_2 - N_1 + 1)n \times 1$, and includes the impact of future control action(s) and

$\Delta \mathbf{u}$ is the input vector over the entire control horizon of dimension (m NU) × 1

$\mathbf{f}$ is the "free" response of the process and is of dimension $(N_2\text{-}N_1\text{+}1)n \times 1$ and consists of the predicted process outputs assuming *no* future changes in the control action, $\Delta \mathbf{u}$, and

$\mathbf{G_d}$ is the dynamic matrix of the process of dimension $(N_2 - N_1 + 1)n \times m$ NU and consists of

$$
G_d = \begin{bmatrix}
G_{N_1-1} & \cdots & G_0 & 0 & 0 & \cdots & 0 \\
G_{N_1} & \cdots & G_1 & G_0 & 0 & \cdots & 0 \\
 & & & & \ddots & & \vdots \\
 & & & & & \ddots & 0 \\
\vdots & & \vdots & \vdots & & & G_0 \\
 & & & & & & \vdots \\
G_{N_2-1} & & & & \cdots & & G_{N_2-NU}
\end{bmatrix} \tag{2.17}
$$

Each G_i is a submatrix of dimension n × m, the elements of which are the step response coefficients of the ith sampling interval of each individual input-output relationship.

There are two alternatives for obtaining the free response **f** of the process (Mutha, 1990):

- use of the matrix Diophantine identity

$$
\boldsymbol{I} = \boldsymbol{E}_j \boldsymbol{A} \Delta + q^{-j} \boldsymbol{F}_j \tag{2.18}
$$

where $\mathbf{E}_j$ and $\mathbf{F}_j$ are matrices of Diophantine coefficients, or by using a

- recursive implementation by setting subsequent $\Delta\mathbf{u}(t+j) = 0$ and iterating over the process model given by equation 2.14 for j=1,2 ... N_2.

On-line implementations of the former Diophantine identity method would require redimensioning of $\mathbf{E}_j$ and $\mathbf{F}_j$ whenever the prediction horizons N_1 or N_2 are changed. The implemented MGPC algorithm uses the latter recursive method in which there is no permanent storage requirement.

2.3.4 The MGPC control law

The unconstrained solution to equation 2.15 is given by:

$$\Delta u(t)=\begin{bmatrix} I_m & 0 & \cdots & 0 \end{bmatrix}\left[G_d^T \underline{\Gamma} G_d + \underline{\Lambda}\right]^{-1} G_d^T \underline{\Gamma}(w - f) \qquad (2.19)$$

where

w is the setpoint trajectory of dimension $(N_2 - N_1 + 1)n \times 1$ given by

$\mathbf{w} = [\mathbf{w}(t+N_1) \ldots \mathbf{w}(t+N_2)]^T$

f is the free response of dimension $(N_2 - N_1 + 1)n \times 1$ given by

$\mathbf{f} = [\mathbf{f}(t+N_1) \ldots \mathbf{f}(t+N_2)]^T$

$\underline{\Gamma}$ is a matrix of $(N_2 - N_1 + 1)$ block diagonal Γ matrices

$\underline{\Lambda}$ is a matrix of NU block diagonal Λ matrices

$\mathbf{G_d}$ is the dynamic matrix of the process given by equation 2.17

2.3.5 Extensions to the MGPC algorithm

The preceding sections presented the fundamental MGPC algorithm. As chemical processes are subjected to measurable and unmeasurable disturbances, and since MPM is always present to a certain degree, the MGPC algorithm is extended to include additional tuning parameters that make this algorithm more robust. In the following sections the disturbance tailoring matrix **T**, feedforward control, plus constraints on input and output amplitudes, as well as input rate constraints are discussed.

2.3.5.1 The disturbance rejection matrix T

The ARIMAX description of the process includes the noise term $\frac{C}{\Delta}e(t)$ where

the elements of the noise vector e(t) are white noise with zero-mean. By filtering e(t) with

$\frac{C}{\Delta}$ to obtain integrated (or non-stationary) colored noise, we obtain an approximation

of noise sources that behave like random steps at random times (e.g. Brownian noise). The problem, however, is the identification of C. It is advantageous to approximate the diagonal polynomial matrix C with a fixed estimate. The diagonal polynomial disturbance rejection matrix **T**, allowing separate $T(q^{-1})$ polynomial specifications for each output channel, is used for this purpose:

$$\boldsymbol{C} \approx \boldsymbol{T} = \begin{bmatrix} T_1(q^{-1}) & & 0 \\ & \ddots & \\ 0 & & T_n(q^{-1}) \end{bmatrix} \tag{2.20}$$

Thus the MGPC prediction equation 2.16 is filtered with $\mathbf{T}^{-1}$ and becomes:

$$\begin{aligned} \hat{\boldsymbol{y}}\,\boldsymbol{T}^{-1} &= \left(\boldsymbol{G}_d \Delta \boldsymbol{u} + \boldsymbol{f}\right)\boldsymbol{T}^{-1} \\ \hat{\boldsymbol{y}}_f &= \boldsymbol{G}_d \Delta \boldsymbol{u}_f + \boldsymbol{f}_f \end{aligned} \tag{2.21}$$

where the subscript $(\bullet)_f$ indicates quantities filtered with $\mathbf{T}^{-1}$.

The filtered free response, $\mathbf{f}_f$, is obtained by iterating the process model given by equation 2.14 as described in section 2.3.3 with the filtered values $\hat{\mathrm{y}}_f$, and $\Delta \mathrm{u}_f$ (with $\Delta \mathbf{u} = 0$).

The control law as given by equation 2.19, however, requires the unfiltered free response, **f**, of the process. This is obtained by *inverse* filtering the filtered free response, $\mathbf{f}_f$, with $\mathbf{T}^{-1}$ to obtain the unfiltered free response, **f**. This technique is given by the relationship:

$$f = f_f \, T \tag{2.22}$$

The interested reader may find further information on inverse filtering in the literature (see e.g. Clarke, 1991 for inverse filtering for the SISO GPC case).

As will be shown in the next chapter, the use of **T** significantly improves the MGPC algorithm's performance in the presence of MPM and allows for separate servo and regulatory performance specifications (McIntosh, 1988).

2.3.5.2 Feedforward Control

In the chemical process industry there are many processes that have measurable load disturbances. Examples of measurable disturbances are the variations in feedstock quality and flow rate. In these cases, feedforward control can significantly improve the performance of the control algorithm. For the inclusion of feedforward control with MGPC, the reader is directed to Mutha (1990). Feedforward control is not considered further in this thesis.

2.3.5.3 Prefiltered Setpoints

By filtering setpoint changes with the polynomial matrix, $\mathbf{P}(q^{-1})$, model-following is achieved. $\mathbf{P}(q^{-1})$ is usually taken as a diagonal polynomial matrix whose elements, $P_{ii}(q^{-1})$, define the inverse of the closed-loop response in the model-following context.

Prefiltered setpoints are not considered further in this thesis, for more information on $\mathbf{P}(q^{-1})$ the reader is directed to Mohtadi *et al.* (1991).

2.3.5.4 Constraints

The MGPC objective function, given by equation 2.15, and its analytical solution, given by equation 2.19, do not consider the physical limitations on the input vector or the output vector. The control input vector, $\mathbf{u}(t)$, must always remain within its physical operating range. A valve for example, can open only from 0% to 100%; the reflux ratio of a distillation column can not be negative. Constraints on the process output reflect the physical nature of the process. Take separation processes (e.g. distillation) for example, the mole fraction of one component within a fluid mixture ranges from 0 to 1; meaning that this fluid contains at least none of the component and at most all of the component (e.g. is pure).

Physical limitations can be incorporated into the solution of the objective function 2.15 by mapping these as constraints onto the plane of manipulated variables, $\Delta\mathbf{u}$. The constraints considered in this thesis are incremental input rate constraints, amplitude constraints on the input and output vectors. The procedure of mapping input amplitude constraints and output amplitude constraints onto input rate constraints, $\Delta\mathbf{u}$, is described by Mutha (1990).

Input rate constraints can be used directly without further modification and for a general MIMO system these become:

$$\begin{bmatrix} \Delta \boldsymbol{u}_{\min} \\ \Delta \boldsymbol{u}_{\min} \\ \vdots \\ \Delta \boldsymbol{u}_{\min} \end{bmatrix} \le \begin{bmatrix} \Delta \boldsymbol{u}(t) \\ \Delta \boldsymbol{u}(t+1) \\ \vdots \\ \Delta \boldsymbol{u}(t+NU-1) \end{bmatrix} \le \begin{bmatrix} \Delta \boldsymbol{u}_{\max} \\ \Delta \boldsymbol{u}_{\max} \\ \vdots \\ \Delta \boldsymbol{u}_{\max} \end{bmatrix} \tag{2.23}$$

The input amplitude constraints are:

$$\begin{bmatrix} \boldsymbol{u}_{\min} \\ \boldsymbol{u}_{\min} \\ \vdots \\ \boldsymbol{u}_{\min} \end{bmatrix} \le \begin{bmatrix} \boldsymbol{u}(t) \\ \boldsymbol{u}(t+1) \\ \vdots \\ \boldsymbol{u}(t+NU-1) \end{bmatrix} \le \begin{bmatrix} \boldsymbol{u}_{\max} \\ \boldsymbol{u}_{\max} \\ \vdots \\ \boldsymbol{u}_{\max} \end{bmatrix} \tag{2.24}$$

but $\Delta \boldsymbol{u}(t) = \boldsymbol{u}(t) - \boldsymbol{u}(t-1)$ thus equation 2.24 becomes

$$\begin{bmatrix} \boldsymbol{u}_{\min} - \boldsymbol{u}(t-1) \\ \boldsymbol{u}_{\min} - \boldsymbol{u}(t-1) \\ \vdots \\ \boldsymbol{u}_{\min} - \boldsymbol{u}(t-1) \end{bmatrix} \le \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 1 & 1 & \cdots & 0 \\ \vdots & & \ddots & \vdots \\ 1 & 1 & \cdots & 1 \end{bmatrix} \begin{bmatrix} \Delta \boldsymbol{u}(t) \\ \Delta \boldsymbol{u}(t+1) \\ \vdots \\ \Delta \boldsymbol{u}(t+NU-1) \end{bmatrix} \le \begin{bmatrix} \boldsymbol{u}_{\max} - \boldsymbol{u}(t-1) \\ \boldsymbol{u}_{\max} - \boldsymbol{u}(t-1) \\ \vdots \\ \boldsymbol{u}_{\max} - \boldsymbol{u}(t-1) \end{bmatrix} \tag{2.25}$$

where the dimension of the vectors are (m NU) $\times$ 1 and the dimension of the lower diagonal matrix is (m NU) $\times$ (m NU).

The output amplitude constraints are:

$$y_{\min} \leq \hat{y} \leq y_{\max} \tag{2.26}$$

substituting equation 2.16, $\hat{y} = G_d \Delta u + f$, into equation 2.26 gives:

$$y_{\min} - f \leq G_d \Delta u \leq y_{\max} - f \tag{2.27}$$

where the vectors and matrices are dimensioned for the appropriate prediction horizons N_1, and N_2, as well as the control horizon NU.

The solution to the constrained MGPC problem is obtained by minimizing the objective function, equation 2.15, subject to the constraints given by equations 2.23, 2.25, and 2.27 . This results in a quadratic programming (QP) problem. This QP is solved at every control interval. Mutha (1990) provides the analytical solution of constrained MGPC subject to output amplitude and input rate and amplitude constraints for NU = 1. For the general case of NU $\neq$ 1 the resulting QP does not have an analytical solution and must be solved numerically.

The commercial quadratic optimization software package QPSOL (Gill *et al.* 1984) is used to solve this quadratic programming problem on-line. QPSOL consists of about 6,000 lines of source code written in ANSI-66 FORTRAN, and provides the user with the *warm start* option. This option improves execution speed by using the previous solution and previous set of active constraints as initial conditions for the current sampling instant.

In summary, the MGPC objective function subject to constraints on output amplitude and input rate and amplitude results in a QP problem. This QP is solved on-line with the software package QPSOL at every sampling interval.

2.4 Summary

Parametric and non-parametric process model LRPC control algorithms have been presented. DMC and MOCCA use non-parametric step-response process model descriptions, whereas the parametric LRPC algorithm, MGPC, uses the ARIMAX process model. The intimate relationship among parametric and non-parametric LRPC algorithms is reflected in the objective functions and control laws by using a standard nomenclature. Table 2.1 summarizes the most important equations presented in this chapter. This table shows that although both LRPC algorithms have different process models, the resulting control laws are identical.

This chapter serves to highlight the similarities among various popular LRPC algorithms. There are, however, subtle differences among these algorithms. These differences make some algorithms superior to others for particular applications. In order to achieve the most effective expert system application the underlying LRPC algorithm must be flexible, robust, and capable of controlling a wide variety of chemical processes. The LRPC algorithm selected for this task is MGPC - it has all of these attributes. This algorithm can be thought of as a "general purpose" LRPC control algorithm as it will control the following types of processes (Mohtadi, 1987):

a) Open-loop stable

b) Open-loop unstable

c) Non-minimum phase.

In addition, this algorithm is attractive due to its use of a compact ARIMAX process model and hence is easily extended to become fully adaptive. Furthermore, as will be shown in the next chapter, MGPC allows for separate servo and regulatory control performance specifications (i.e. it has a two degrees of freedom structure).

The next chapter presents closed-loop performance specifications and provides insight into the use of the MGPC algorithm and its (many) tuning parameters.

	MOCCA/DMC	MGPC
Process Model	$y(t) = \sum_{i=1}^{N} G_{i-1}\Delta u(t-i) + G_N u(t-N-1)$	$Ay(t) = Bu(t-1) + \frac{Ce(t)}{\Delta}$
Objective Function	$\min J = \sum_{j=1}^{P}[\hat{y}(t+j)-w(t+j)]^T\Gamma[\hat{y}(t+j)-w(t+j)] + \sum_{j=1}^{M}\Delta u(t+j-1)^T\Lambda\Delta u(t+j-1)$	$\min J = \sum_{j=N_1}^{N_2}[\hat{y}(t+j)-w(t+j)]^T\Gamma[\hat{y}(t+j)-w(t+j)] + \sum_{j=1}^{NU}\Delta u(t+j-1)^T\Lambda\Delta u(t+j-1)$
Control Law	$\Delta u(t)=[I_m \ 0 - 0][G_d^T\Gamma G_d + \Lambda]^{-1}G_d^T\Gamma(w - f)$	$\Delta u(t)=[I_m \ 0 - 0][G_d^T\Gamma G_d + \Lambda]^{-1}G_d^T\Gamma(w - f)$
Min. Pred. Horizon	1	N_1
Max. Pred. Horizon	P	N_2
Control Horizon	M	NU
Output Weight	Γ	Γ
Control Weight	Λ	Λ
Filtering	various options (see Li *et al.*, 1989)	T

Table 2.1 Parametric and non-parametric LRPC objective functions and control laws

Chapter 3

Performance Tuning

The task of any control system is to meet its closed-loop control performance specifications. Performance specifications are an important component of the control system as they reflect the desired closed-loop response of the process in the presence of measurement noise, model-plant mismatch (MPM), and disturbances. Performance specifications can be expressed in terms of:

a) Performance indices which mathematically state the objective function to be minimized or maximized. Examples are the minimization of J_{MGPC} as given by equation 2.15, or the minimization of the integral squared error performance index

$$J_{ISE}=\int_0^\infty e^2(t)dt \tag{3.1}$$

where the error, e(t), is the difference between the desired process output or setpoint and the actual process output.

b) Frequency domain criteria such as bandwidth and damping could be used in the specification of the desired closed-loop performance. Furthermore, closed-loop performance can be expressed in terms of the gain margin and the phase margin.

c) Time domain criteria, examples of which are rise time, t_r, settling time, t_s, damping ratio, and overshoot.

The problem with using performance indices and frequency domain criteria for performance specifications is their inherent need for detailed analysis. It is difficult for process operators to develop heuristics or an engineering "feel" for these criteria. It is preferred to state the desired closed-loop response in terms of criteria that are easily observed on-line or easily computed.

Time domain performance criteria are most widely accepted and used in industry as they provide process control operators with friendly and intuitive specifications that are easily understood. These performance specifications directly relate to the desired shape of the process' output allowing process operators to judge - on-line - about the "goodness" of the control performance. Due to their wide acceptance and usage only time domain performance specifications are considered further in this thesis.

Section 3.1 presents the most important time domain performance specifications. Section 3.2 distinguishes amongst the different types of disturbances encountered in any process control application. Section 3.3 follows with an exhaustive list of tuning guidelines for SISO GPC and MIMO GPC, as found in the literature. Constrained MIMO GPC is also addressed. The selection of performance specifications in conjunction with the applicable MGPC tuning parameters for implementation in the expert system is the subject of section 3.4. Section 3.5 presents additional tuning parameters which may be useful during process startup and process shutdown. Section 3.6 concludes this chapter with a brief summary.

3.1 Performance Criteria

Detailed treatment of time domain performance specifications can be found in Seborg *et al.* (1989) and also in Stephanopoulos (1984). The following discussion proceeds by considering the underdamped normalized step response shown in Figure 3.1.

Time domain performance specifications are defined as follows:

1) Overshoot: This is the ratio of a/b, where b is the final or ultimate value of the response at steady state, and 'a' is the maximum amount of by which the response exceeds its ultimate value.

2) Decay Ratio: This is the ratio of c/a, where c is the amount by which the second peak exceeds the ultimate value.

3) Period of Oscillation: T is the time between two successive peaks.

4) Rise Time: t_r is the amount of time the process output takes to reach a specified output level for the first time. The specified output level is usually given in percent of the change in steady state values. Popular values are 80% as well as 100%. A special case is 63.3% for a first order process with no dead time; as this yields the value of its time constant τ. Alternate definitions exclude the process' dead time by defining t_r as the time required for the response to rise from 10% to 90% of its final value.

4) Time to First Peak: t_p is the amount of time the process output takes to reach its first maximum value.

5) Settling Time: t_s is the amount of time the process output takes to reach within ±5% of the final value and remain within this bound. A value of ±2% is

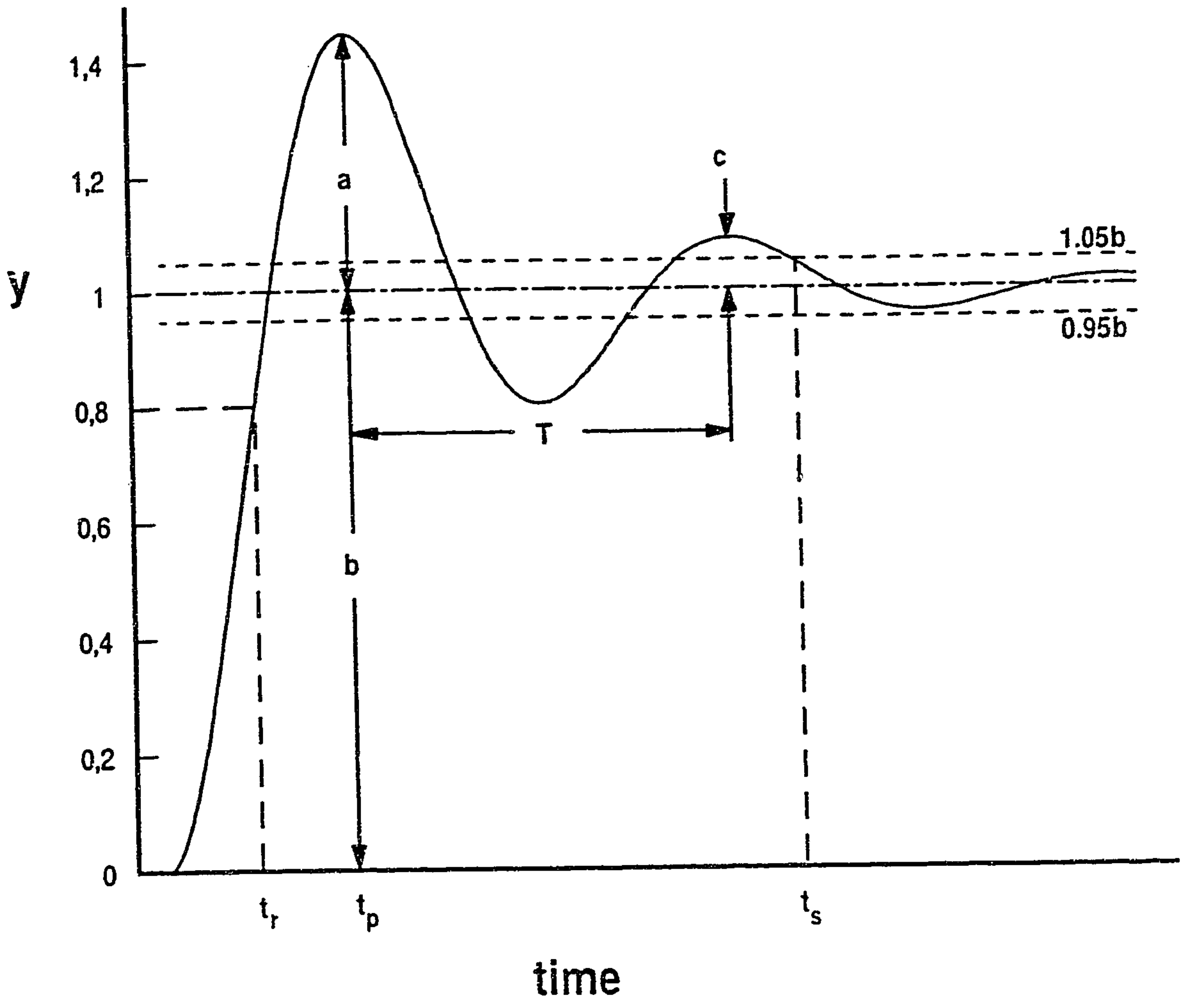


Figure 3.1 Characteristics of an underdamped unit step response.

also frequently used.

The above performance criteria make little distinction between servo and regulatory control objectives. Indeed, for most control algorithms once servo control criteria are specified, the regulatory response is fixed as there is only one degree of freedom in the design of the controller. Thus performance tuning of one degree of freedom control algorithms is a juggling act or trade-off between regulatory and servo control performance. McIntosh (1988), however, shows that GPC is a two degrees of freedom controller allowing separate servo and regulatory control specifications. To clarify this distinction the following list provides possible servo and regulatory control criteria:

Servo control criteria:

a) Overshoot
b) Decay Ratio
c) Period of Oscillation
d) Rise Time
e) Time to First Peak
f) Settling Time

Regulatory control criteria:

a) Decay Ratio
b) Control Signal Variance
c) Settling Time

The final selection of performance criteria depends very much on the specific process to be controlled. In the petro-chemical industry the majority of processes display overdamped responses for which the decay ratio is not a suitable performance criteria. In addition, model-plant mismatch and stochastic disturbances (i.e. measurement noise) are invariably present in most control applications; and hence the calculation of the performance criteria must be based on several data points to reduce the effect of measurement noise. In addition, *a priori* information regarding the particular process may influence the final selection of performance criteria.

3.2 Types of Disturbances

Chemical processes are subject to various kinds of noise sources, ranging from stochastic zero-mean measurement noise to Brownian motion or non-stationary, random-walk type disturbances. This thesis considers only unmeasurable noise sources as feedforward control can be applied to measurable disturbances. It is advantageous to distinguish between input and output disturbances and relate these to the ARIMAX process model. Based on the ARIMAX model

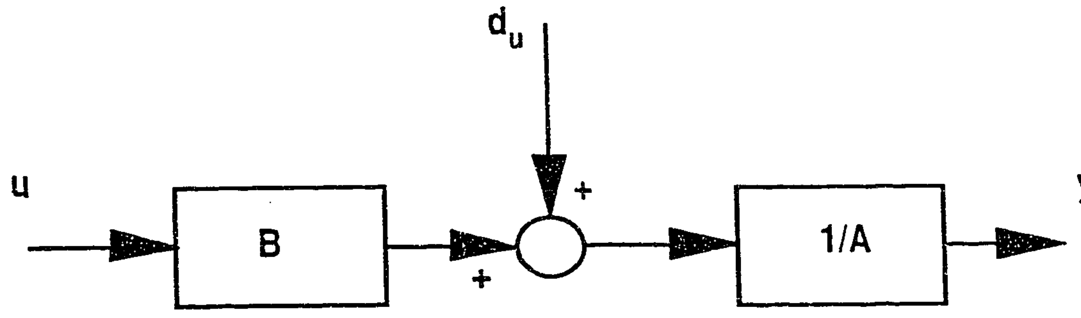
$$Ay(t) = Bu(t-1) + d_u(t) + Ad_y(t) \tag{3.2}$$

the following disturbance sources are considered in this thesis:

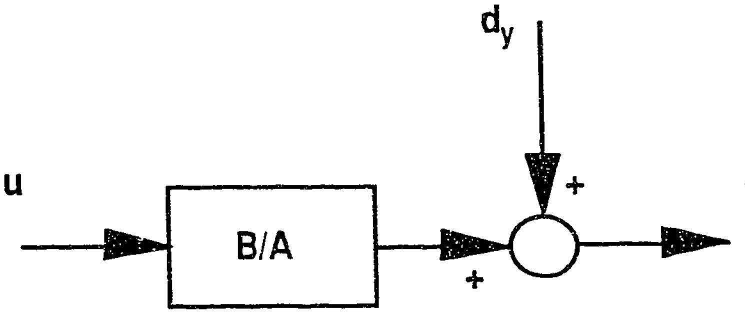
i) $d_u(t)$ represent non-zero-mean <u>input</u> step disturbances, and

ii) $d_y(t)$ represent stochastic <u>output</u> disturbances with zero-mean.

The above disturbance sources are illustrated in figure 3.2.



Input Type Disturbances



Output Type Disturbances

Figure 3.2 Input and Output disturbance sources

3.3 GPC Tuning Parameters

After the publication of the single-input single-output (SISO) GPC algorithm by Clarke *et al.* (1987a) research activity focused on the selection of the algorithm's tuning parameters. Mohtadi (1987) extended the basic GPC algorithm to include multi-input multi-output (MIMO) systems. It is useful to review the tuning guidelines provided in the literature for SISO GPC followed by the guidelines provided for MIMO GPC since there are subtle yet important differences between the two. This review is followed by observations and guidelines for constrained MIMO GPC.

3.3.1 SISO GPC Tuning Parameters

Guidelines for the selection of unconstrained SISO GPC tuning parameters have been provided by Clarke *et al.* (1987a,b), Mohtadi (1987), McIntosh (1988), and Scattolini *et al.* (1990). All authors agree that the maximum output prediction horizon, N_2, must be greater than the dead time of the process and ultimately relate it to the closed-loop rise-time, t_r. In the case of nonminimum-phase processes N_2 must also be selected sufficiently large as to "look beyond" the initial inverse response. If the dead time of the process is known then the minimum output prediction horizon, N_1, should be set to this value. In the case where the dead time is not known, or is variable, setting N_1 to 1 and setting the control weighting, λ, to a small value will result in no loss of stability. By filtering the setpoint with a user chosen $P(q^{-1})$ transfer function one can obtain an approximate pole placement controller or approximate model following (Mohtadi, 1987). Mohtadi (1987) and Clarke *et al.* (1987a,b) present the following stability criteria for limiting cases of GPC:

Closed Loop Stability

Clarke *et al.* (1987b) prove that the closed-loop system is stable if the system is observable and controllable and if

a) $N_2 \rightarrow \infty$, $NU = N_2$ and $\lambda > 0$ or

b) $N_2 \rightarrow \infty$, $NU \rightarrow \infty$, and $\lambda = 0$ where $NU \leq N_2 - n + 1$

where n is the number of states of the plant (i.e. $n=\max(\delta a+1,\delta b)$).

Mean Level Control

For open-loop stable processes Clarke *et al.* (1987b) show that the closed-loop poles will be placed at the same locations as the open-loop poles with

$$NU = N_1 = 1, \lambda = 0, P = 1 \text{ with } N_2 \rightarrow \infty .$$

Hence the GPC controller will provide a step change in control action following a step setpoint change which will drive the process output to the setpoint with the same dynamics as the open-loop system, but with no offset. For practical purposes $N_2 \rightarrow \infty$ can be replaced with a finite number equal to the settling time of the process in terms of sample times.

State-dead-beat Control

The closed-loop system is equivalent to a stable state-dead-beat controller if the system is observable and controllable and if

$$N_1 = n, \lambda = 0, NU = n, P = 1, \text{ and } N_2 \geq 2n - 1$$

where n is the number of states of the plant (Clarke *et al.*, 1987b). McIntosh (1991) indicates that relaxed specifications of

$$NU \geq \delta a + 1,\ N_1 \geq \delta b + 1,\ P = 1, \text{ and } N_2 \geq NU + N_1 - 1$$

still deliver dead-beat control.

Pole-placement Control

Mohtadi (1987) proves that the closed-loop system is equivalent to a pole-placement controller if the system is observable and detectable and if

$$N_1 = n,\ N_2 \to \infty, \text{ and } \lambda \to 0 \text{ and } NU = N_2 \text{ or}$$
$$\lambda = 0, \text{ and } NU = N_2 - n.$$

For finite horizons the above is reduced to

$$N_1 \geq n,\ N_2 - N_1 \geq n - 1,\ NU = N_1, \text{ and } \lambda \to 0.$$

where n is the number of states of the system. The closed-loop poles are placed at the zeros of $P(q^{-1})$.

McIntosh (1988) points out that exact pole placement can only be achieved if the process has no dead time. Furthermore, processes with nonminimum phase, and/or fractional delays, as well as , processes with pole over zero excess larger than two, will yield discrete systems with zeros outside the unit circle. Exact model-following will attempt to cancel these unstable zeros with unstable closed-loop poles. This is clearly not desirable, and hence McIntosh (1988, and 1991) develops the Detuned Model Following configuration (see below).

Subsequent to commissioning, the GPC tuning parameters may be adjusted on-line to achieve the desired closed-loop control performance. McIntosh (1988, 1991) presents three GPC control algorithm configurations aimed at fixing as many controller parameters as possible, and using but one remaining parameter to tune the process' response. The configurations are:

Output Horizon Configuration

For this configuration the fixed GPC parameters are

$$NU = 1,\ N_1 = 1,\ P = 1,\ \lambda = 0\ .$$

The active tuning parameter, N_2, is used to vary the speed of response over the range of

$$d + 1 \leq N_2 \leq \infty$$

where d is the delay of the process in sampling intervals. For practical implementation the above range can be reduced to

$$d_{max} + 1 < N_2 \leq t_s$$

where d_{max} is the maximum expected time delay and t_s is the settling time of the open-loop process in sampling intervals including the delay. If the process to be controlled is a nonminimum phase plant, then N_2 must be sufficiently large such that $N_2 \geq N_{min}$ where N_{min} satisfies the relation

$$sign\left(\sum_{j=1}^{N_{min}} g_{j-1}\right) = sign(K_p) \tag{3.3}$$

where g_i are the process step response coefficients, and K_p is the steady state gain of the process (i.e. $K_p = B(1)/A(1)$).

Lambda Weighting Configuration

This strategy fixes the following parameters:

$NU = \delta a + 1$, $N_1 = \delta b + 1$, $N_2 \geq \max(NU + N_1 - 1, t_r/T_s, N_{min})$, $P = 1$

where δa and δb are the orders of the model polynomials $A(q^{-1})$ and $B(q^{-1})$ respectively. The scalar control weighting, λ, is used as the active tuning parameter with

$$0 \leq \lambda \leq \infty .$$

A value of N_2 roughly equal to the rise time, t_r (here defined as the time required for the process output to reach 63% of the final steady state value including the delay), will also normally be larger than N_{min}.

Detuned Model Following

This configuration uses $P(q^{-1})$ as the active tuning parameter by choosing

$$P(q^{-1}) = 1/M(q^{-1})$$

where $M(q^{-1})$ is the desired closed-loop model. McIntosh (1988) provides guidelines for selecting $M(q^{-1})$ and fixes

$$NU = \delta a + 1,\ N_1 = 1,\ N_2 > d + NU, \text{ and } \lambda = 0.$$

Furthermore, it is recommended to set $N_2 \approx t_r$, where t_r is the rise time as defined above. Following these recommendations the controller is sufficiently detuned so as to avoid exact model following; hence circumventing the cancellation of any open-loop zeros.

In the case of open-loop unstable processes the Lambda Weighting configuration requires $\lambda < \lambda_{max}$, where λ_{max} is the upper bound on λ yielding closed-loop stability. This is conceptually misleading as large values of λ are generally considered to be conservative. Thus McIntosh (1988) recommends the Output Horizon Configuration and Detuned Model Following Configuration over the Lambda Weighting Configuration.

The robustness of the SISO GPC algorithm to model-plant mismatch as well as disturbances can be greatly enhanced by the use of the $T(q^{-1})$ polynomial without affecting servo response characteristics (McIntosh, 1988). Clarke *et al.* (1987b) as well as Mohtadi

(1987) recommend $1/T(q^{-1})$ equal to a fixed low-pass filter. Investigations by McIntosh (1988) show that

$$T(q^{-1}) = (1 - c_1)^{\delta A}$$

where $0.5 \leq c_1 \leq 0.95$, with a default value of $c_1 = 0.8$, yield improved performance. Clarke and Mohtadi (1989) suggest that the degree of $T(q^{-1})$ be equal to $\delta A + 1$. There is common agreement that any reasonable choice of $T(q^{-1})$ provides improved performance in the presence of MPM and also reduces control effort.

In addition to improved performance in the presence of MPM, the use of $T(q^{-1})$ provides GPC with its two degrees of freedom structure. McIntosh (1988) shows that servo and regulatory modes are completely decoupled if there is no MPM. Furthermore, the author points out that given a reasonable process model, $T(q^{-1})$ may be adjusted so as to modify the rejection of disturbances without affecting the response to setpoint changes significantly. This implies that both servo response and regulatory response criteria may be specified. Intuitively, by employing a low-pass filter, high frequency components due to unmodelled dynamics or high frequency unmeasurable load disturbances are removed prior to prediction (recall that prediction is performed with equation 2.21). This improves the robustness of GPC to MPM and stochastic disturbances with zero-mean (e.g. measurement noise or stochastic output disturbances with zero-mean). The use of $T(q^{-1})$, however, does adversely effect GPC's disturbance rejection performance of non-zero-mean noise sources. Specifically, regulatory response to non-zero-mean input step disturbances is degraded. This drawback, albeit noticeable, is greatly overshadowed by the advantages gained through the use of $T(q^{-1})$.

3.3.2 MIMO GPC Tuning Parameters

Subsequent to the extension of GPC to multi-input multi-output processes (e.g. MGPC), Shah *et al.* (1987) show that multivariable GPC is pairing or order invariant provided the output and control horizons are selected to be the same for each channel. Furthermore, the authors point out that settings of $N_1 = 1$, $NU = 1$, and as $N_2 \rightarrow \infty$ result in the mean level control law (analogous to the SISO GPC case discussed in section 3.3.1). In addition, the authors point out that knowledge of the process' interactor matrix is not necessary provided the following sufficient but not necessary condition is applied:

$$N_2 - NU \geq d - 1 \tag{3.4}$$

$$N_1 = 1,\ \mathbf{P} = \mathbf{I},\ \Lambda = 0$$

where d is the maximum forward shift in the interactor matrix. In other words, obeying the above guideline guarantees that $(\mathbf{G}_d^T \mathbf{G}_d)^{-1}$ exists (this term appears in the control law, equation 2.19).

Mohtadi *et al.* (1991) provide three guidelines for selecting MIMO GPC tuning parameters and they are listed below:

Guideline 1

There are two basic configurations for using MGPC designs:

- Set NU = 1 and adjust the properties of the controller with N_2. N_2 is typically set to the dominant time constant of the open-loop system (about 10 samples, if the sample rate is chosen according to usual rules of thumb). Different N_{2i} for different channels will only be necessary with systems which have widely varying dynamics.
- Set N_2 to a reasonable value ($\approx$ dominant time constant of the system), use large NU ($\approx$ number of oscillatory or unstable modes

of the system, say 4) and use the properties of Λ and **P** to achieve the desired results.

Guideline 2

With input/output based incremental models as opposed to impulse or step response models it is <u>always</u> necessary to use **T** to improve sensitivity and robustness. The order of **T** should be commensurate with **A**. A diagonal structure would only be necessary if there are different noise levels at different channels or the unmodelled dynamics is known to be highly structured. In most cases **T** proportional to identity will probably suffice. The position of the poles marginally faster than open-loop dynamics appears to yield satisfactory results. Highly resonant and slow poles should of course be excluded. With impulse response models **T** is effectively set to **A**, and therefore extra filtering is rarely necessary.

Guideline 3

Use γ_i and **P** to reduce high frequency coupling by penalizing the fast varying channels. Use Λ to improve the control sensitivity. Λ also appears to be the only tool to overcome problems associated with ill conditioned systems. Λ reduces the condition number of the controller and thus helps in reducing the sensitivity to directional perturbations with ill conditioned systems.

The reader may have noticed a great degree of similarity between MIMO GPC and SISO GPC tuning parameter guidelines. In fact, MIMO GPC Guideline 1 reflects the Output Horizon, Lambda Weighting, and Detuned Model Following configurations of SISO GPC. Similarly, the Mean Level Control property of MIMO GPC is analogous to that of SISO GPC. And lastly, the arguments presented in section 3.3.1 pertaining to the use of $T(q^{-1})$ for SISO GPC are also applicable to the use of **T** for MIMO GPC.

3.3.3 Constrained MIMO GPC Tuning Parameters

The addition of constraints for the multi-input, multi-output GPC control algorithm provides additional tuning parameters to meet closed-loop performance criteria. Mutha (1990) presents his MGPC algorithm which uses input rate and amplitude constraints as well as output amplitude constraints. Based on this algorithm he provides the following guidelines and observations:

a) The effect of input weighting (Λ) is masked in the presence of tight input rate constraints.

b) Input rate constraints handle unstable systems better than Λ weighting. However, if very large Λ do stabilize the system then the resulting control action is "smoother".

c) Tight input rate constraints can be used to achieve smooth decoupling among output channels.

d) Fixed parameter MGPC cannot achieve offset free control with large model-plant mismatch (i.e. mismatch > 50% of the steady state gain) in all output channels. A tuning guideline for selecting output weighting, Γ, for systems with model-plant mismatch is to select larger output weighting terms for output channels with higher model-plant mismatch.

e) If the amount of model-plant mismatch is unknown then γ_i should be selected proportional to the gain of the channel.

Although the constrained MIMO GPC algorithm provides the user with a multitude of tuning parameters, closed-loop stability analysis is not possible in the presence of active constraints. This shortcoming, although serious, is overshadowed by the algorithms' very apparent capabilities. Recent work by Zafiriou (1991) provides some insight into the difficulties of performing closed-loop stability analysis in the presence of active constraints.

3.4 Selection of Performance Criteria and GPC Tuning Parameters

The expert system implementation is applicable to a wide range of processes requiring the performance criteria to be equally applicable to a wide range of processes. Although most chemical processes display an overdamped response, if excited sufficiently hard these will also display an underdamped response. Hence the servo control criteria must accommodate both types of responses. This thesis assumes that the prediction horizons and control horizons (i.e. N_1, N_2, NU) are the same for each channel as different horizons for different channels are only necessary for processes with widely different dynamics in each channel (Mohtadi *et al.* 1990). This assumption implies that servo criteria for different output channels may not be met independently.

Disturbances in the form of measurement noise and Brownian motion are considered as these manifest themselves in any real control application. It is desirable to decouple output channels form each other as much as possible to reduce the effect of external disturbances. Indeed setpoint changes in one channel act as disturbances upon all remaining channels. The **T** filter implementation in this thesis is limited to $\mathbf{T} = \mathbf{I}\ T(q^{-1})$, hence one common $T(q^{-1})$ is applied to all channels. This implies that certain regulatory performance specifications for different input and output channels may not be met independently.

The selection of performance criteria is based upon the desired closed-loop response of the process as well as simplicity in detection. For the subsequent discussion consider the non-minimum phase system with dominant off-diagonal elements, and a non-diagonal delay matrix given by Shah *et al.* (1987):

$$(A^{-1}B)_{model} = \begin{bmatrix} \dfrac{0.4}{1-0.6z^{-1}} & \dfrac{z^{-2}}{1-0.6z^{-1}} \\ \dfrac{z^{-1}}{1-0.8z^{-1}} & \dfrac{0.2z^{-2}}{1-0.8z^{-1}} \end{bmatrix} \tag{3.5}$$

and for the purposes of model-plant mismatch consider the following:

$$(A^{-1}B)_{plant} = \begin{bmatrix} \dfrac{0.48}{1-0.66z^{-1}} & \dfrac{0.8z^{-2}}{1-0.54z^{-1}} \\ \dfrac{0.8z^{-1}}{1-0.72z^{-1}} & \dfrac{0.24z^{-2}}{1-0.88z^{-1}} \end{bmatrix} \tag{3.6}$$

Where the unit delay due to a zero order hold has been extracted from each individual transfer function (recall the definition of the ARIMAX model as per section 2.3.1). The amount of MPM consists of a 10% displacement in the pole location for every pole, and a minimum and maximum steady state gain mismatch of 30% and 100% respectively. A sampling time of $T_s = 2$ seconds is used.

3.4.1 Servo Performance Criteria

Servo control performance criteria are an important aspect of the control system as these reflect the desired closed-loop control performance during setpoint changes. In order to apply the expert system implementation to as many processes as possible, it is advantageous to accommodate both underdamped and overdamped processes. For this purpose, two servo response criteria have been selected. The primary and secondary servo response criteria, applicable to *each* output channel, are: rise time, and overshoot.

The primary servo response criteria, rise time, is defined as the time required by the process output to reach 80% of the final or steady state value the first time; and thus includes the dead time. For digital implementations t_r is measured from the first sampling interval after a setpoint change until the 80% level is reached. In order to reduce the effect of measurement noise, t_r is calculated on-line using three data points and the central difference derivative approximation. The on-line calculation of t_r is given by:

Let	T_s	=	the sampling interval, and
	k_{tr}	=	time in samples such that
			$k_{tr} T_s < t_r < (k_{tr} + 1)T_s$, and
	$y(k)$	=	the process output at time $t = k\ T_s$, and
	Δw	=	the change in setpoint of the process,

then the derivative approximation becomes

$$\left.\frac{dy}{dt}\right|_{t=k_{tr}} \approx \frac{y(k_{tr}+1) - y(k_{tr}-1)}{2T_s} = m \tag{3.7}$$

hence

$$t_r \approx T_s\, k_{tr} + \frac{0.8\,\Delta w - y(k_{tr})}{m} \tag{3.8}$$

For SISO systems t_r is calculated directly using equation 3.8. For MIMO systems the rise time of output channel i, namely $t_{r,i}$, is calculated in an analogous manner.

At this point is it useful to investigate the effect of the maximum output prediction horizon and control horizon (i.e. N_2, NU), on the rise time, t_r. For the process given by equation 3.5, consider figure 3.3 which depicts the rise times, $t_{r,1}$ and $t_{r,2}$, versus N_2 for given settings of NU. This figure indicates that within the calculation error given by equation 3.7 the rise time t_r is unaffected by the choice of output prediction horizon N_2, if $N_1 = 1$, and NU > 1, and the constraints are not active (i.e. unconstrained solution).

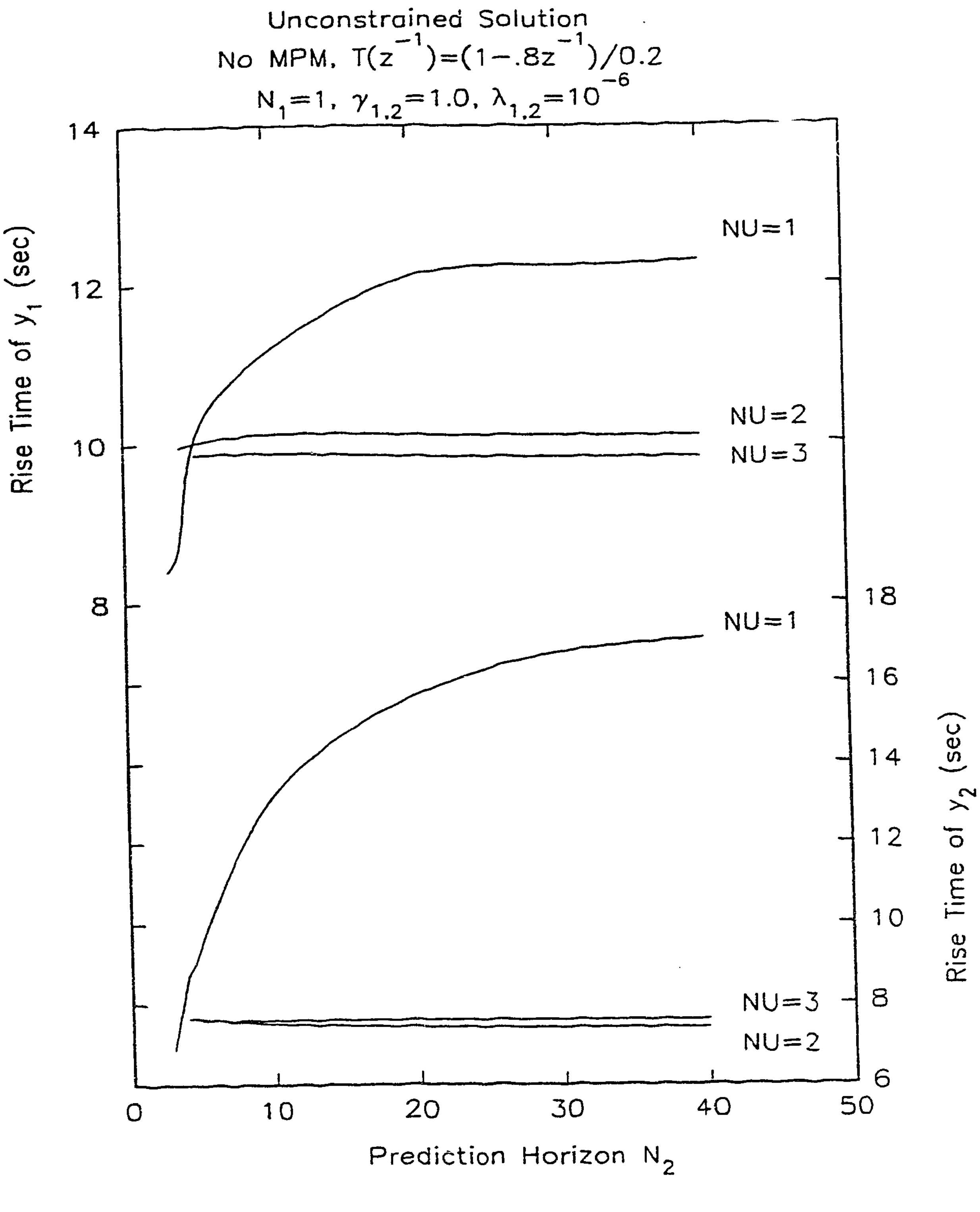


Figure 3.3 Rise time t_r versus prediction horizon N_2

Excluded from this figure are the results obtained with NU = 4 and NU = 5 as these yield approximately the same t_r as with NU = 3. Furthermore, this figure shows that by adjusting N_2 the duration of t_r varies from open-loop to values that exceed the responses with settings of $N_1 = 1$ and NU > 1. Although no figures are provided, similar results are obtained for the remaining two benchmark problems used in chapter 5. Based on these results, the default control and output prediction horizons, as used by the expert system, are:

Fix NU = 1, $N_1 = 1$, and

adjust N_2 to meet rise time specifications.

By fixing NU = 1, upper and lower bounds on N_2 can be established. The lower bound on N_2 is given by solving equation 3.4. This, however, requires explicit knowledge of the process' interactor matrix, d. Typically, $d < d_{max}$, where d_{max} is the maximum forward shift of the system's delay matrix (both d and d_{max} include the unit delay due the zero-order hold). Substituting d_{max} into equation 3.4, and using a non-zero value for Λ, yields the following conservative relationship which is used as the lower bound on N_2:

$$N_2 - NU \geq d_{max} - 1 \tag{3.9}$$

$N_1 < d_{max}$, $\mathbf{P} = \mathbf{I}$, $\Lambda = \mathbf{I}\ 10^{-6}$

It can be shown that for the case of $N_1 \geq d$ (but $N_1 < N_2$ of course) equation 3.4 is valid provided d is replaced with N_1. Hence in the event that $N_1 \geq d_{max}$ equation 3.10 is used to obtain the lower bound on N_2:

$$N_2 - NU \geq N_1 - 1 \tag{3.10}$$

$N_1 \geq d_{max}$, $\mathbf{P} = \mathbf{I}$, $\Lambda = \mathbf{I}\ 10^{-6}$

Equations 3.9 and 3.10 determine the lower bound of N_2. These relations do not, however, impose any limitations on its upper bound. As the computational load is directly proportional to N_2, an upper limit resulting in essentially open-loop dynamics is imposed.

This upper limit on N_2, as implemented, is 40. This choice is confirmed by figure 3.3, which shows that settings of $N_2 = 40$ and $NU = 1$ yield closed-loop responses with essentially open-loop dynamics.

The sampling time, T_s, impacts the initial setting of N_2 used during commissioning. For implementation with the expert system T_s is selected in accordance with the usual rules of thumb (i.e. $T_s \approx 1/10$ of dominant time constant) and an initial value of $N_2 = 10$ is used consistent with guideline 1 given in section 3.3.2 . Furthermore, the control weighting is set to a constant value of $\lambda = 10^{-6}$ for each input channel resulting in $\Lambda = \mathbf{I}\ 10^{-6}$, where $\mathbf{I}$ is the identity matrix. Hence the sole purpose of control weighting is to guard against ill conditioned systems.

The secondary servo response criteria, overshoot, consists of a user specified percentage applied to the change in setpoint. This criteria is implemented on-line using output constraints. Clearly t_r and overshoot are conflicting specifications. For overdamped processes which exhibit no overshoot as well as underdamped processes exhibiting little overshoot, N_2 is adjusted such that *only* t_r is met. For underdamped processes showing large overshoot, N_2 is adjusted such that the overshoot does not exceed its specified limit. Thus the overshoot specification represents the largest allowable limit if the rise time criteria for this channel cannot be met.

In summary, the user specifies both t_r and overshoot for each output channel. However, as N_2 affects the speed of response of all channels their respective specifications cannot be met independently. Hence the interpretation of the primary and secondary servo response specifications, t_r and overshoot, as applicable to MIMO processes is:

Fix $NU = 1$ and adjust N_2 until the output channel with the most stringent t_r specification is met subject to its overshoot specification.

This statement implies that there may be specifications whereby the most stringent t_r specification is met and t_r of all other output channels exceed their specifications (i.e. are faster than specified). The lower bound of N_2 is given by equations 3.9 and 3.10, and its upper bound, as implemented, is 40.

3.4.2 Regulatory Performance Criteria

Regulatory control performance criteria are an important aspect of the control system as these reflect the desired closed-loop control performance during steady state operation. Three regulatory performance criteria are selected for implementation with the expert system. The criteria are:

i) Control signal variance, σ^2_u, this is specified for each input channel and is calculated on-line omitting data during setpoint changes.

ii) Regulation band, r_b, which is specified for each output channel, and represents an upper and lower bound on the allowable drift from the desired setpoint.

iii) Maximum desirable standard deviation, σ_{ymax}, this is specified for the process and places an upper limit on the permitted standard deviation for output channels (this is equally applicable to all output channels).

The above criteria address both stochastic disturbances, typically in the form of measurement noise, as well as Brownian motion, which is often regarded as step like disturbances at random times. As will be shown shortly, σ^2_u is directly related to the amount of filtering performed with the disturbance rejection matrix, **T**. In contrast, the output weighting matrix, Γ, is used to "focus" the GPC algorithm thereby reducing the standard deviation of the most vigorous output channel. While r_b is readily implemented using output constraints, it places bounds on the allowable drift of output channels from their setpoints. At this point the reader may find it useful to briefly review sections 2.3.5.1 and 2.3.5.4 which deal exclusively with **T** and the use of constraints.

In order to minimize the computational load, this thesis limits the disturbance rejection matrix to **T** = T(q^{-1}) **I**, where **I** is the identity matrix. Hence one common T(q^{-1}) is applied to all channels. Of particular importance to MIMO processes is steady state gain normalization of T(q^{-1}) (i.e. T(q^{-1} = 1) = 1). McIntosh (1988) proves that this is not necessary for SISO processes, however, T(q^{-1}) **must** be normalized for MIMO processes. Through on-line simulations of MIMO processes it was observed that updating and failing to normalize T(q^{-1}) causes the GPC algorithm to erroneously detect and compensate for large sudden steady state gain mismatches resulting in great regulatory activity. Consequently consider T(q^{-1}) given by the following normalized configuration:

$$\frac{1}{T(q^{-1})} = \left(\frac{1-c_1}{1-c_1 q^{-1}}\right)^{\delta T} \tag{3.11}$$

$0.5 \le c_1 \le 0.95$, and $\delta T = 1$ or 2.

Based on equation 3.11, default first and second order T(q^{-1}) filters are defined:

Default first order filter

$$\frac{1}{T(q^{-1})} = \frac{1-c_1}{1-c_1 q^{-1}} \tag{3.12}$$

where c_1=0.8, and $\delta T = 1$.

Default second order filter

$$\frac{1}{T(q^{-1})} = \left(\frac{1-c_1}{1-c_1 q^{-1}}\right)^2 \tag{3.13}$$

where c_1=0.8, and $\delta T = 2$.

For processes whose input-output relationships are approximated by first order plus time delay continuous time transfer functions (i.e. $\delta A = 1$) the default $T(q^{-1})$ filters are of the recommended order (McIntosh 1988, Clarke and Mohtadi 1989, see also section 3.3.2).

Lower and upper bounds on $T(q^{-1})$ are obtained by considering the range of movement imposed on c_1 and δT. These bounds are obtained by substituting the appropriate values for c_1 and δT into equation 3.11. For convenience, the lower and upper bounds are listed below:

Lower bound on $T(q^{-1})$

$$\frac{1}{T(q^{-1})} = \frac{0.5}{1-0.5q^{-1}} \tag{3.14}$$

Upper bound on $T(q^{-1})$

$$\frac{1}{T(q^{-1})} = \frac{0.0025}{(1-0.95q^{-1})^2} \tag{3.15}$$

The effect of the default $T(q^{-1})$ filters on the closed-loop control performance in the presence of Gaussian noise is shown in figure 3.4. This figure illustrates the dramatic effect $T(q^{-1})$ has on σ^2_u. Also, this figure suggests that in the absence of any $T(q^{-1})$ filter the GPC algorithm vigorously attempts to cancel the effect of the Gaussian noise and, inadvertently, adds to the noise. This figure emphasizes the direct relationship among filtering with $T(q^{-1})$ and control signal activity.

Again consider the process given by equation 3.5 with MPM given by equation 3.6. Figure 3.5 clearly indicates that the default first order $T(q^{-1})$ filter easily stabilizes this

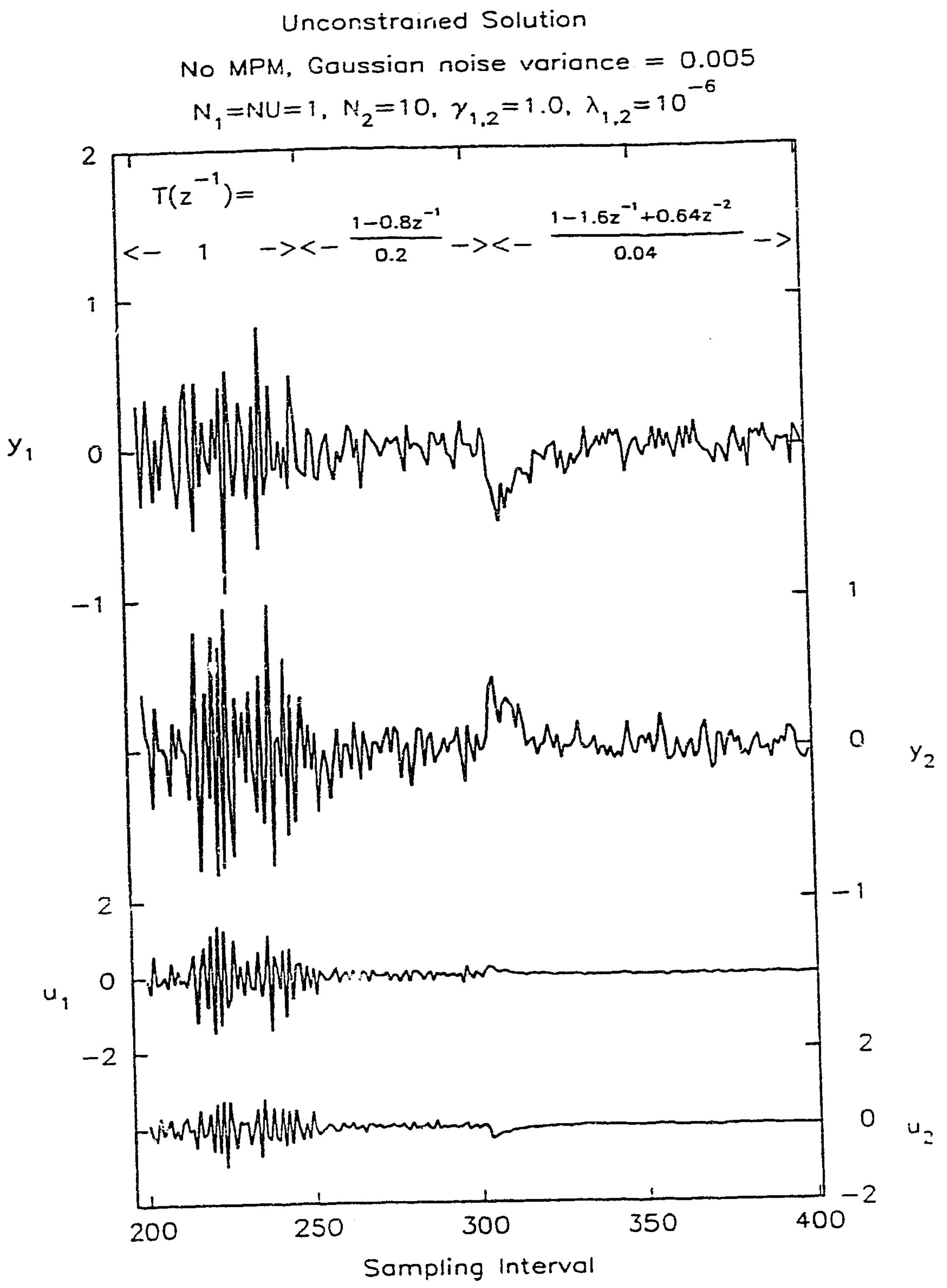


Figure 3.4 The effect of first and second order $T(q^{-1})$ filters on the non-minimum phase process subject to Gaussian noise.

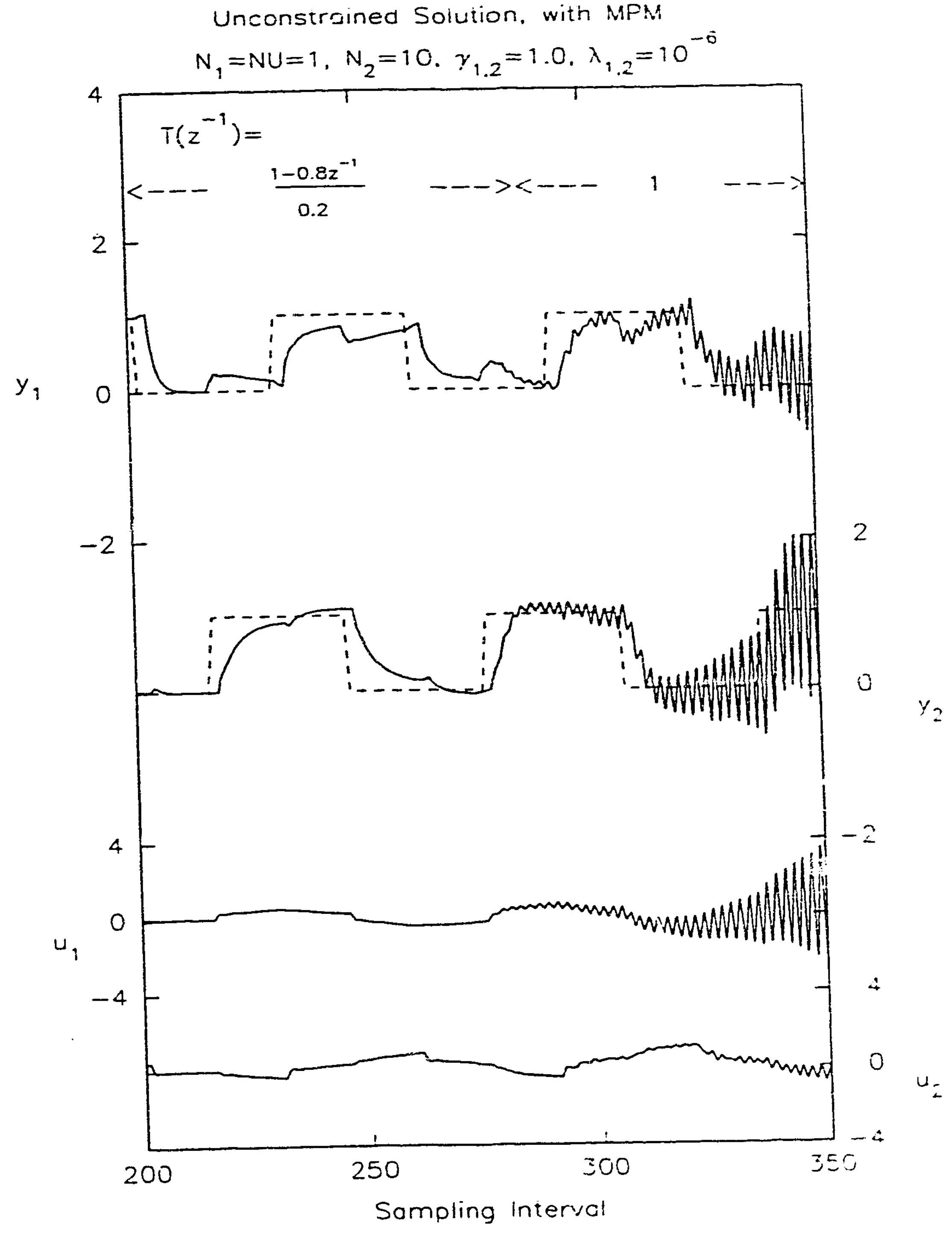


Figure 3.5 The non-minimum phase process under conditions of MPM, with and without first order default $T(q^{-1})$ filters.

system (recall that the amount of MPM is about 10% in the location of every pole, and 30% to 100% in the steady state gains !). This figure substantiates the use of $T(q^{-1})$ to improve GPC's robustness to MPM.

For practical implementations, c_1 is limited to $0.5 \leq c_1 \leq 0.95$. Where $c_1 = 0.5$ represents light filtering and $c_1 = 0.95$ represents heavy filtering. Also, whenever the value of c_1 is updated on-line estimates of filtered values for new y_f and new u_f are required (i.e. replacing the values of old y_f and old u_f at the end of the history stack requires previous values that have been discarded). It was found through simulations that provided the changes in c_1 are small a good estimate for the new y_f and new u_f are simply the values of the old y_f and old u_f. This is illustrated at the switching point from first order to second order default $T(q^{-1})$ filters in figure 3.4 (focus on iteration 305 of this figure). At this location the process' outputs appear to encounter disturbances; this however is due to the large switch from default first order $T(q^{-1})$ filter to default second order $T(q^{-1})$ filter. As will be shown in chapter 5, such "imaginary" disturbances are not encountered if adjustments in c_1 are gradual, or the switching occurs from second order to first order $T(q^{-1})$ filters.

By default, a $T(q^{-1})$ filter will always be implemented with the constrained MIMO GPC algorithm so as to reduce the effects of MPM. However, as $\mathbf{T} = T(q^{-1})\,\mathbf{I}$, it is apparent that the control signal variance specifications of each process input can not be met independently. Hence the tuning guideline for $T(q^{-1})$ is:

Tune $T(q^{-1})$ such that the process input with the most stringent control signal variance specification is met.

This implies that some process inputs will exceed their control signal variance specifications (i.e. better than specified).

The regulation band, r_b, serves to decouple process outputs during setpoint changes and reduces the effect of input step disturbances, $d_u(t)$. The regulation band is specified in percent such that the allowable bounds on $y_i(t)$ are given by:

$$(1-r_b) \leq \frac{y_i(t)}{b} \leq (1+r_b)\ ,\ b \neq 0 \tag{3.16}$$

where b = steady state value.

In the event that b = 0, the upper and lower bounds on $y_i(t)$ are given by $\pm r_b$, respectively.

The use of r_b to decouple output channels during setpoint changes is shown in figure 3.6. This figure illustrates the use of output constraints to implement r_b (i.e. the output constraints $y_{1,max}$ and $y_{1,min}$ are equal to $+r_b$ and $-r_b$, respectively). Output $y_1(t)$ is limited to within the regulation band during setpoint changes in $y_2(t)$. This figure also shows how output constraints on $y_2(t)$ are updated to reflect its overshoot specifications during setpoint changes; and, subsequently are set back to the values of the regulation band.

Figure 3.7 illustrates the use of r_b during input step disturbance rejection. Small input step disturbances which do not violate r_b of any output channel, are rejected in an analogous manner to the unconstrained solution (i.e. compare results of iteration 20 with iteration 100). On the other hand, the large input step disturbance in input channel 2 forces this output channel to drift beyond its r_b. Similarly, output channel 1 is forced outside of its r_b due to coupling among the channels. Comparing the results of iteration 60 with iteration 140, the use of r_b yields a marginal improvement in disturbance rejection for <u>both</u> output channels. In fact, the constrained GPC algorithm fully exploits the range of r_b to fight disturbances entering the system. Thus the use of r_b results in improved regulatory response to input step disturbances.

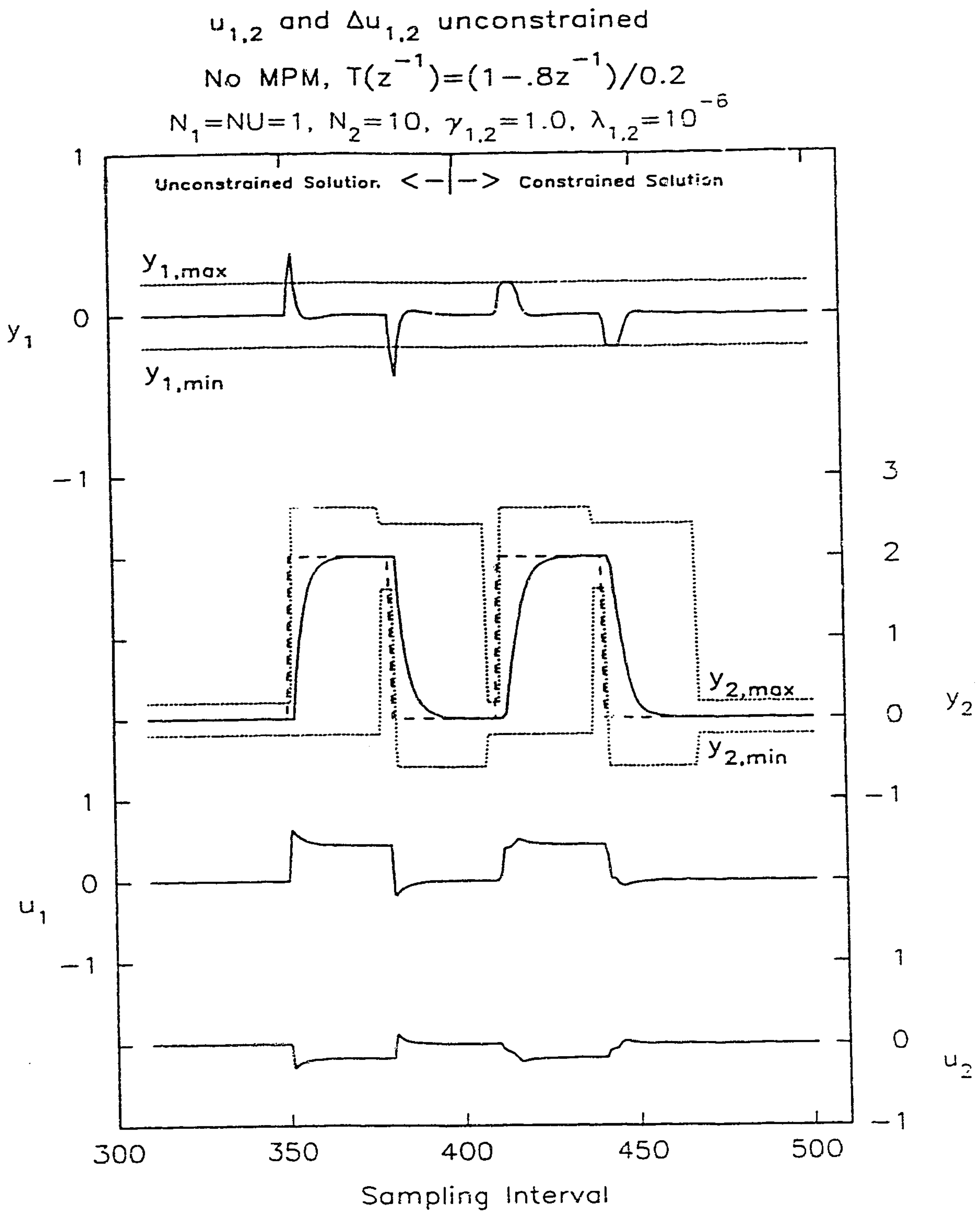


Figure 3.6 Use of r_b to decouple process outputs during setpoint changes.

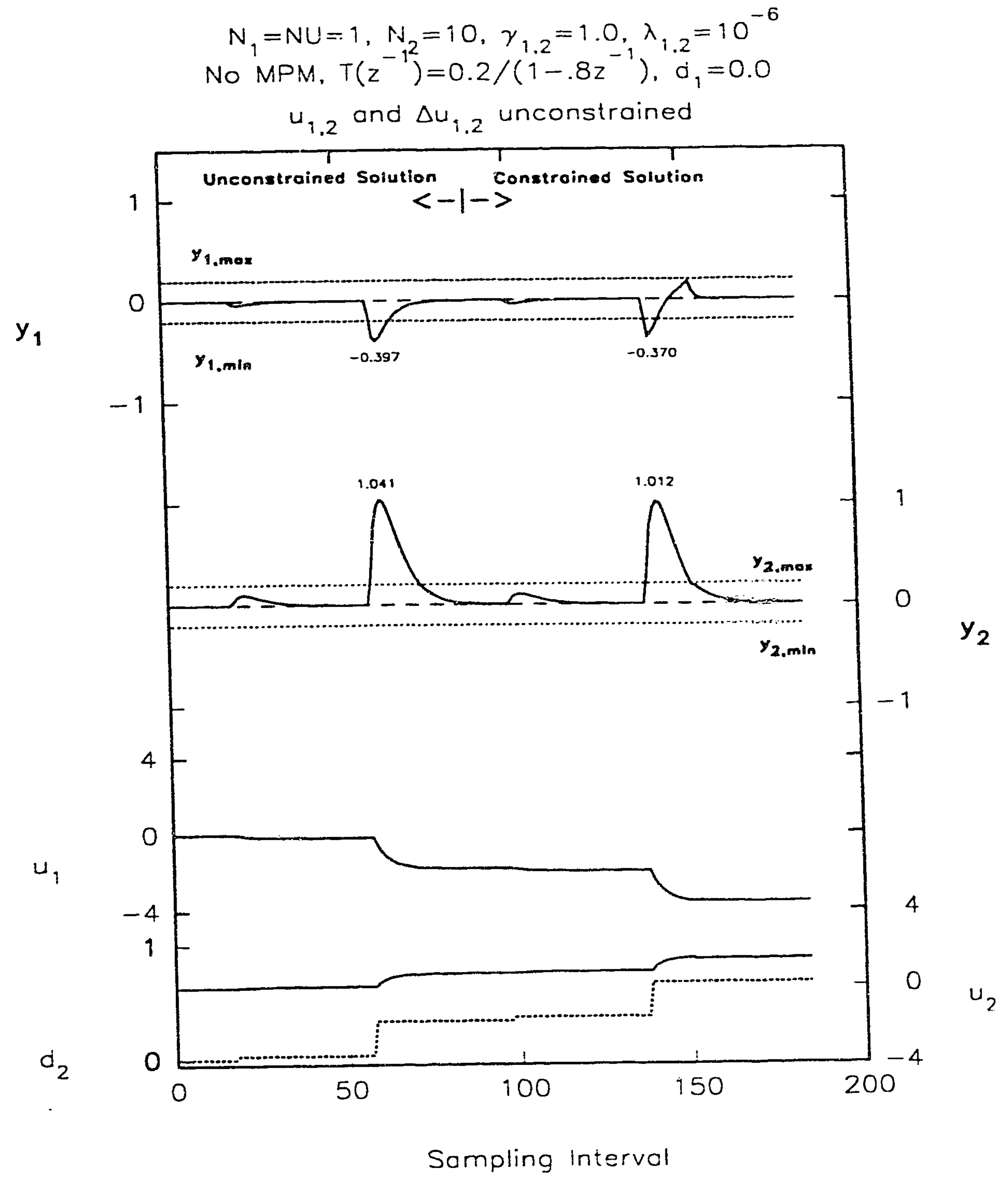


Figure 3.7 Use of r_b to improve regulatory control to input step disturbances.

Output weighting, Γ, affects the standard deviation of process outputs, σ_y, and is used to focus the GPC algorithm. This tuning parameter is used to balance process outputs by reducing σ_y of the most vigorous channel at a cost of increased control signal activity. Consider figure 3.8 which was generated with the process given by equation 3.5 and simulated using Gaussian noise. While maintaining $\gamma_2 = 1.0$ the value of γ_1 was reduced causing output $y_1(t)$ to drift from its setpoint. Throughout this run output $y_2(t)$ remains relatively constant (i.e. its mean and standard deviation do not change much). As illustrated Γ is an important tuning parameter in balancing process outputs. Recall that the scalar quadratic cost function J_{MGPC} directly addresses setpoint tracking, however, does not explicitly handle decoupling. Γ can be exploited to balance the control performance among individual process outputs by scaling such that under typical operating conditions the values used within the control calculation after analog to digital conversion as well as normalization are of similar magnitude (Mutha, 1990). This thesis does not address process outputs of greatly dissimilar magnitudes.

In summary, three regulatory performance criteria are selected: a control signal variance, σ^2_u, is specified for each input channel; a regulation band, r_b, is specified for each output channel; and a maximum output standard deviation, σ_{ymax}, is specified for the process. These specifications are met by adjusting appropriate GPC tuning parameters. As shown earlier, σ^2_u is directly related to the amount of **T** filtering performed. In contrast, r_b is readily implemented using output amplitude constraints and effectively decouples process outputs during setpoint changes. In addition, the use of r_b yields improved regulatory response to input step disturbances. The output weighting matrix, Γ, is used to "focus" the GPC algorithm thereby reducing the standard deviation of the most vigorous output channel. Scaling is performed whenever the standard deviation of one or more output channels approaches or exceed the σ_{ymax} specification.

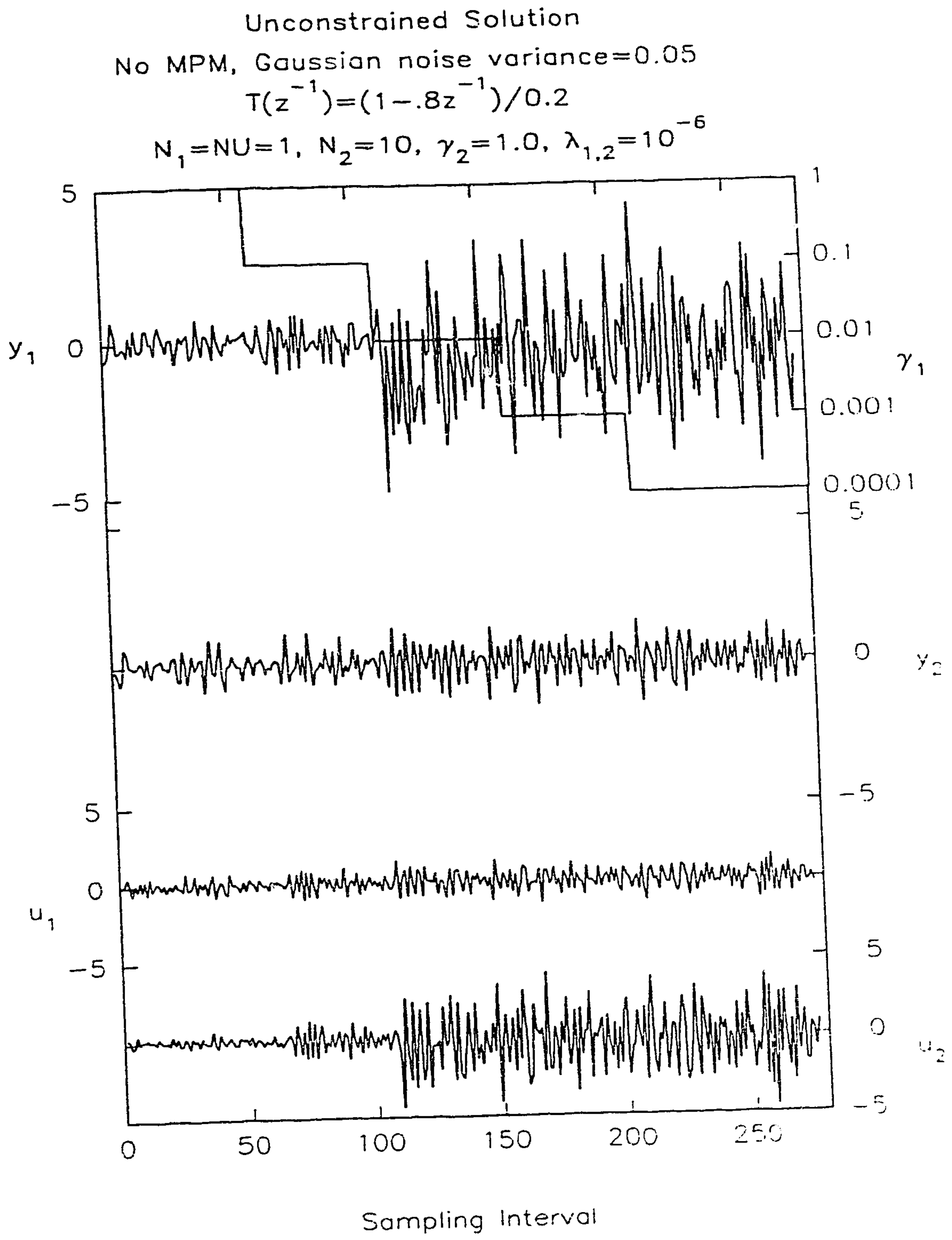


Figure 3.8 Effect of output weighting Γ on output drift.

3.5 Additional GPC Tuning Parameters

The constrained MIMO GPC algorithm provides tuning parameters that may be useful during process startup or process shutdown. Figure 3.9 illustrates three alternatives for implementing setpoint changes. Iterations 0 to 30 depict the unconstrained response to step setpoint changes in $y_1(t)$ and $y_2(t)$. Iterations 30 to 80 show similar step setpoint changes, however, implemented with tight input rate constraints. Tight rate constraints prevent large sudden control actions; hence, $y_1(t)$ and $y_2(t)$ track their setpoints slowly but steadily. Iterations 80 to 150 shows the process' response to setpoint scheduling. Setpoint scheduling causes the process outputs to approximately follow the shape of the setpoints delayed by the dead time. Setpoint scheduling is implemented by gradually adjusting the setpoint as opposed to making one large step like setpoint change.

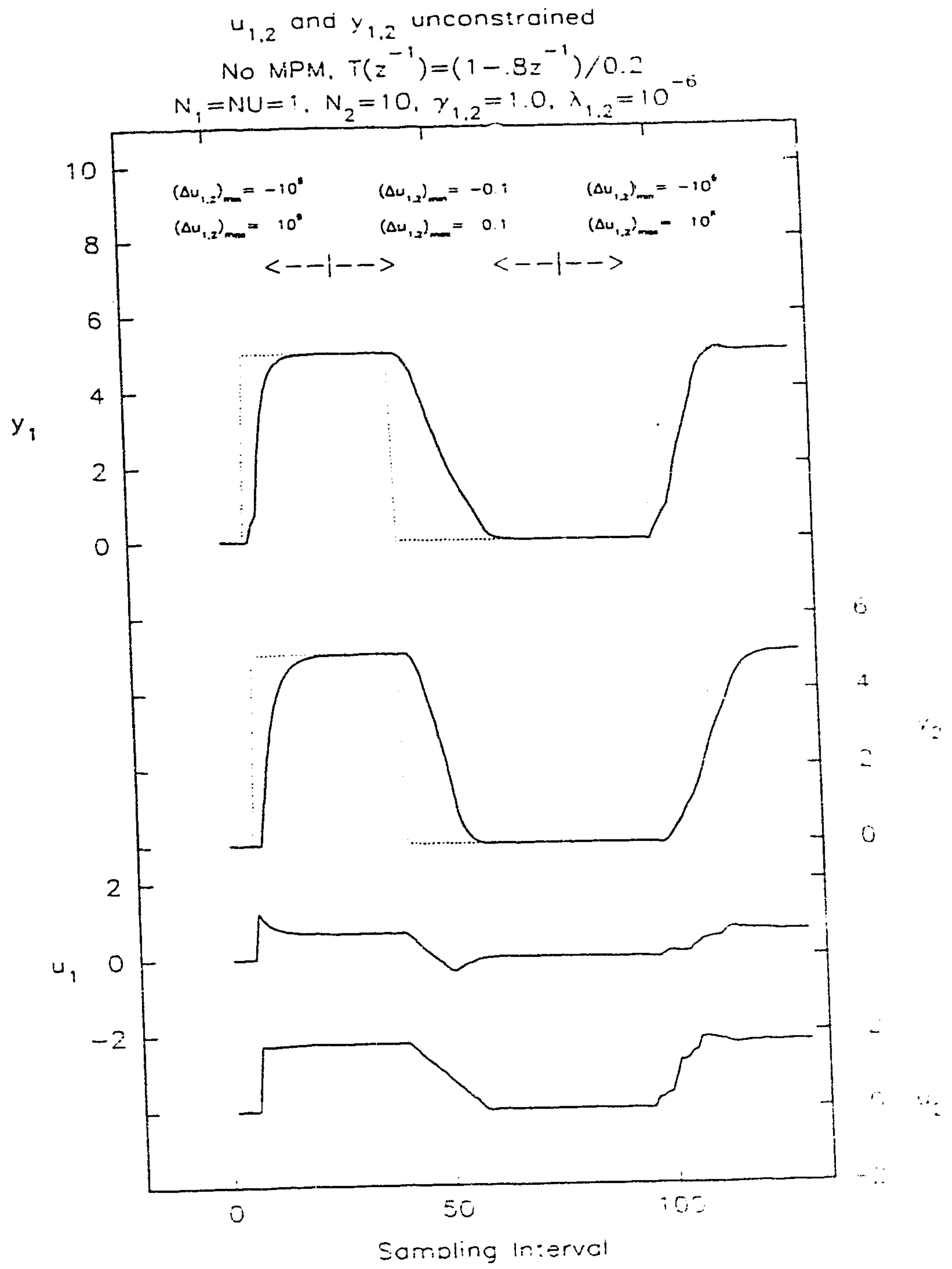


Figure 3.9 Use of rate constraints and setpoint scheduling to slow down process' response.

3.6 Summary

Performance specifications are of utmost importance to any control system as these reflect the desired closed-loop response. GPC's unique two degrees of freedom structure allows for separate servo and regulatory control performance specifications. Time domain criteria are selected for implementation as these are intuitive, easily observed on-line, and easily computed. Furthermore, in order to apply the expert system implementation to as many processes as possible both underdamped and overdamped processes are accommodated. The servo control performance criteria consist of t_r and overshoot for each output channel (t_r uses the 80% level and includes the process' dead time). The regulatory control performance criteria consist of the regulation band, r_b, the control signal variance, σ^2_u, and the maximum desirable output standard deviation, σ_{ymax}. The criteria σ^2_u and r_b are specified for each input and output channel respectively. A single value for σ_{ymax} applies equally to all process outputs.

In order to maintain the closed-loop control performance, the expert system supervisor adjusts pertinent GPC tuning parameters in an on-line real-time environment. Hence the appropriate performance-criteria-GPC-tuning-parameter pairings are also identified in this chapter. Based on the studies conducted, each performance criteria is paired with the most effective GPC tuning parameter. Upper and lower bounds are also developed for each tuning parameter, as well as default settings for all remaining parameters. Table 3.1 summarizes the servo and regulatory performance criteria along with the applicable GPC tuning parameters.

The expert system supervisor as well as the precise tuning strategies for adjusting GPC parameters is the topic of chapter 4.

Performance Criteria	**Performance Specification**	**GPC Tuning Parameter**
Servo Control	t_r	N_2, (with NU = 1)
Servo Control	overshoot	N_2, (with NU = 1)
Regulatory Control	σ^2_u	$T(z^{-1})$, (adjust c_1 and δT)
Regulatory Control	r_b	y_{min}, y_{max}
Regulatory Control	σ_{ymax}	Γ

Table 3.1 Performance specifications with GPC tuning parameters

Chapter 4

Supervisory LRPC

This chapter focuses on the performance supervisor whose role is to monitor the closed-loop control performance of the underlying control algorithm and take appropriate action so as to maintain user specified performance. The idea of using a supervisory shell to monitor the actual closed-loop performance is not new. Indeed, Turnbull Control Systems market their Auto-Tuning Controller which, based on the actual closed-loop performance, calculates recommended settings for the Proportional plus Integral plus Derivative action (PID) control algorithm. Foxboro's Expert Adaptive Controller (EXACT) goes one step further; it automatically updates PID controller settings once per setpoint or disturbance transient in order to meet user specified closed-loop performance specifications. Minter and Fisher (1988) review both of these controllers and conclude that the addition of a closed-loop performance supervisor to a control algorithm yields significantly better control performance over a broader operating region.

The need for a closed-loop performance supervisor is substantiated by McIntosh (1988). He points out that model-plant mismatch causes control performance to deteriorate. Even under ideal conditions of no MPM the controller settings required to produce the desired performance may not always be known *a priori*. McIntosh's implementation of a performance supervisor for SISO unconstrained GPC illustrates that subsequent to commissioning, the GPC tuning parameters need to be adjusted on-line in order to achieve the desired closed-loop control performance.

In recent years research activity has resulted in great advances in Expert Systems (ES) technology. ES are a subset of Knowledge Based Systems which themselves are a subset of AI. Specifically, ES are computer programs that can solve domain specific problems. Implementations of ES technology for process control applications, however, require on-line communication and real-time computing. These stringent requirements were a major obstacle and kept ES technology inside control research labs and well away from "real-life" process control implementations. Major advances in computer technology in conjunction with ES technology have paid off. Several companies now offer expert system development tools for real-time applications (e.g. Personal Consultant Plus by Texas Instruments, G2 by Gensym).

The application of ES technology to process control problems has resulted in applications ranging from controller tuning, adaptation, monitoring and diagnostics, to supervision. Åström (1988) presents his views on the evolution of control technology and suggests that expert systems, when incorporated into the process control environment, represent the highest plateau of evolution. Integrating ES technology with real-time control systems is discussed by Årzén (1989a,b), Oyen *et al.* (1990), and by Beck and Lauber (1990). The topic of process monitoring and advisory systems via the use of an expert system in a real-time environment is addressed by Adams *et al.* (1987), Doraiswami and Jiang (1989), and Tzouanas (1988). Betta *et al.* (1990) and Haest *et al.* (1990) address ES for system identification, so far, however, these are limited to off-line applications. The use of ES to automate and implement heuristic control algorithms is discussed by Zhongyuan (1990) and by Sripada *et al.* (1987). Aynsley *et al.* (1989) present an ES application for fermentation control which performs supervision, monitoring, fault detection and diagnostics. The use of a meta system to coordinate the activity among several ES and numerical processing routines is discussed by Rao *et al.* (1989).

Section 4.1 presents the generic architecture for an ES supervisor. The real-time object oriented expert system development tool G2, which was utilized to implement a subset of the proposed ES supervisor, is the topic of section 4.2. Section 4.3 details the performance tuning strategies and the developed software that forms the Adaptive Long range predictive control Performance Supervisor (ALPS).

4.1 Expert Supervision - an architecture

This section outlines the architecture and tasks of an ES supervisor. As seen in the introduction to this chapter, current ES technology reaches well beyond the tuning of LRPC algorithms. The routine task of monitoring the closed-loop performance and subsequent fine tuning of LRPC parameters is but one component of an ES supervisor. The architecture presented below, at first glace, may appear far reaching; but is achievable with the current state of ES technology.

The ES supervisor hierarchy is graphically illustrated in figure 4.1. This figure shows that the ES supervisor resides above the control algorithms. Furthermore, the control algorithms are external to the ES but may or may not reside within the same physical computer. By separating the control algorithms from the ES supervisor, an extra degree of fault tolerance or safety is achieved as both systems operate independently. Also, this allows for the best utilization of ES capabilities which is symbolic processing, linked to external routines for numerical processing.

Due to the interaction between sequential process units, where the end-product of one unit is the feedstock to the next unit, the supervisor must interact with and monitor the performance of several unit operations simultaneously. This strategy allows for the

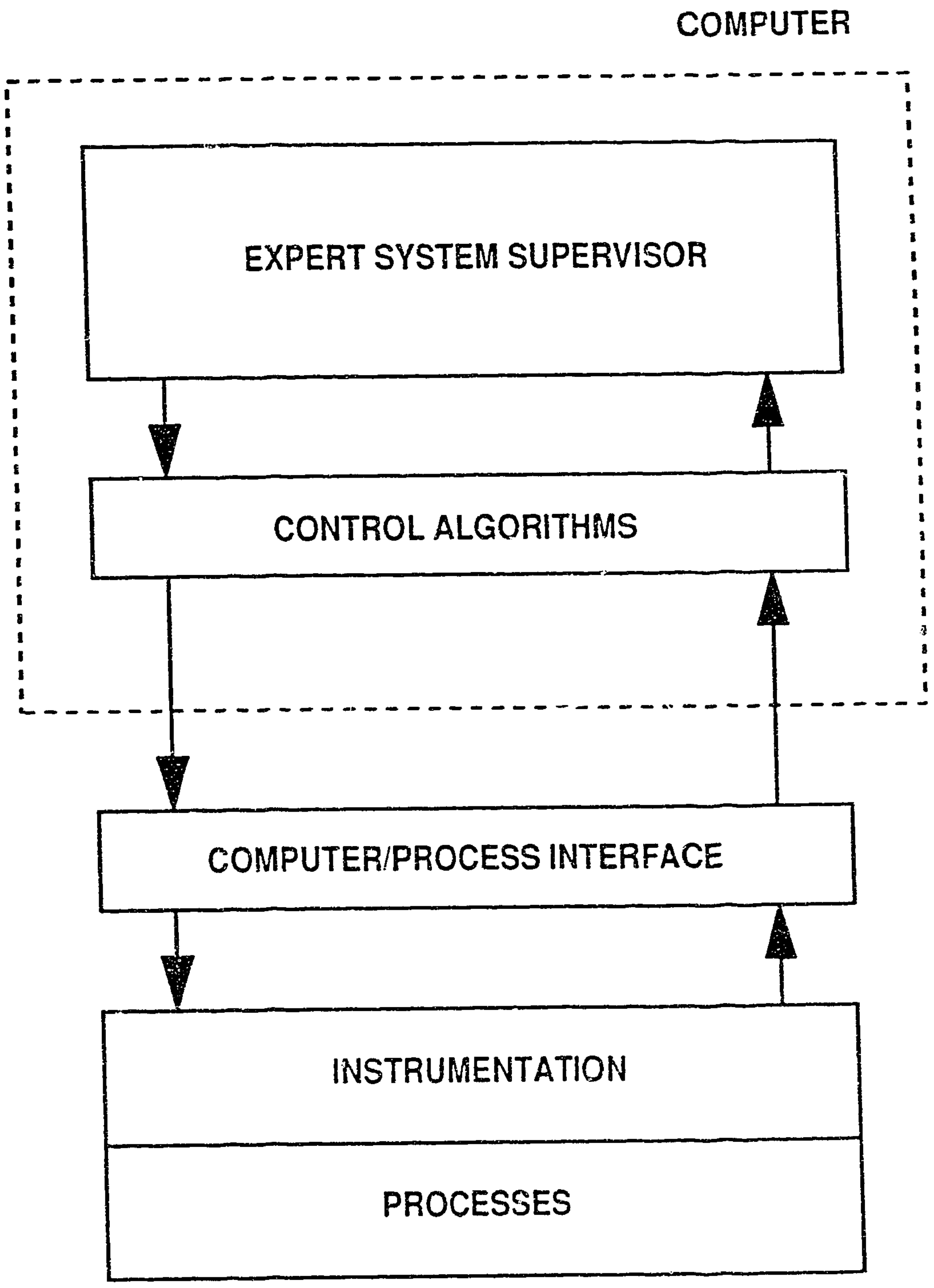


Figure 4.1 The expert system supervisor hierarchy.

development of meaningful contingency operations in the event of upsets which could affect a series of down-stream process units. Each individual process may be of dimension n × m (i.e. MIMO processes).

The ES supervisor consists of several components:

a) Closed-loop performance supervision and LRPC tuning

b) Fault and disturbance detection and diagnostics

c) Contingency operations

d) Operator interface

These issues are discussed in greater detail in the following sections.

4.1.1 Closed-loop performance supervision and LRPC tuning

The supervisor monitors the *actual* closed-loop control performance and makes the necessary adjustments in the LRPC tuning parameters so as to achieve the desired closed-loop control performance. Figure 4.2 illustrates this procedure. Notice the similarity of this strategy with that of a conventional feedback loop. While the LRPC algorithm calculates process inputs based upon the error between the setpoints and the actual process outputs, the LRPC supervisor adjusts the tuning parameters based upon the error between the desired and actual closed-loop control performance.

The supervisor must take advantage of the two degrees of freedom structure which the LRPC algorithm permits. Performance criteria are to consist of both servo performance criteria and regulatory performance criteria.

The supervisor must be able to recognize the limitations of the LRPC algorithms. In the event that the desired closed-loop performance is not achievable, either due to unrealistic performance specifications or due to closed-loop stability considerations limiting further tuning, the supervisor should alert the process operator.

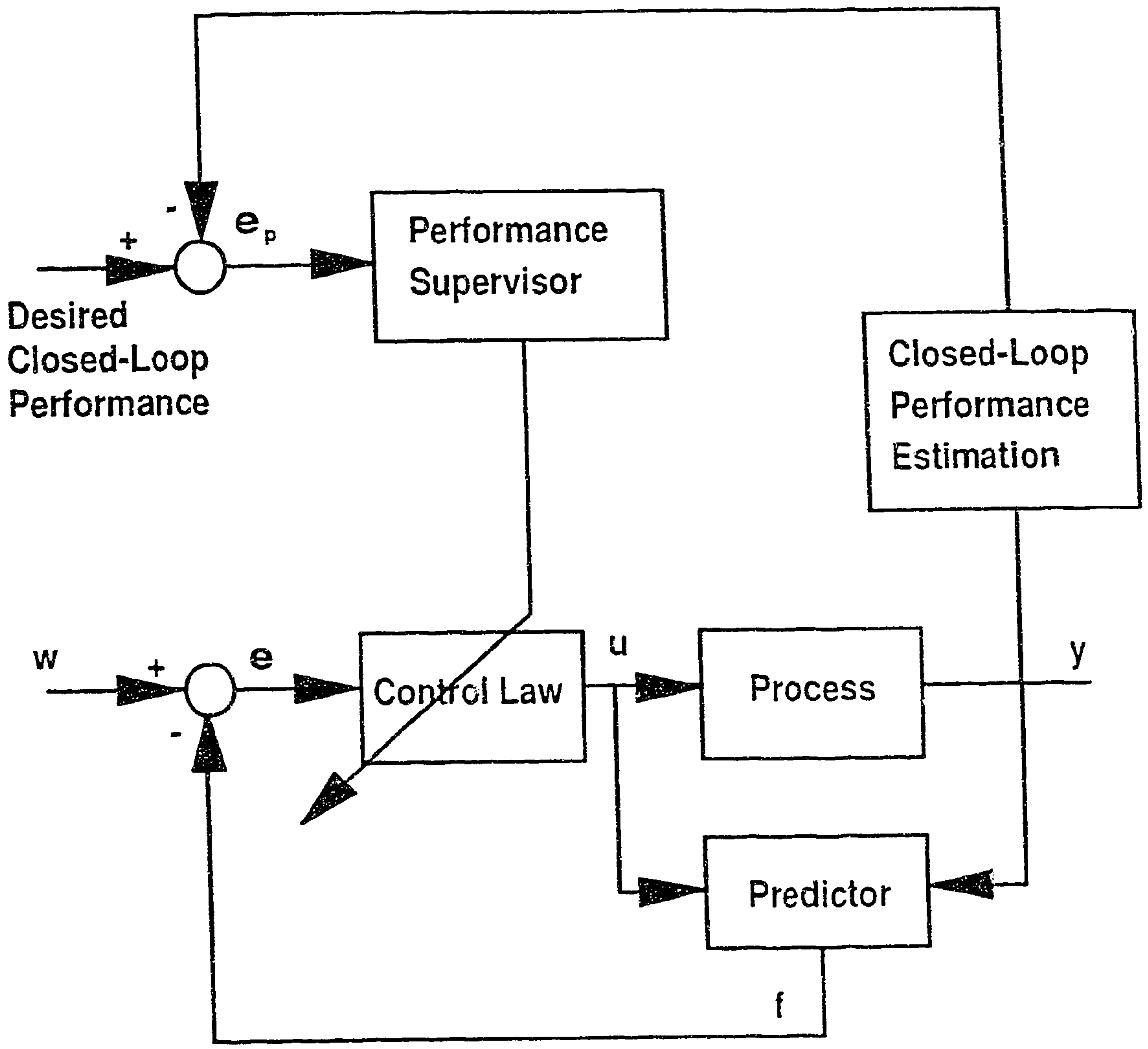

+
-
e_p
Performance Supervisor
Desired Closed-Loop Performance
Closed-Loop Performance Estimation
w
+
e
-
Control Law
u
Process
y
Predictor
f

Figure 4.2 Closed-loop performance feedback for LRPC supervision

4.1.2 Fault and disturbance detection and diagnostics

Both faults and disturbances adversely affect the closed-loop control performance. The supervisor must be able to detect faults and disturbances and initiate diagnostics. Once a fault or disturbance has been detected it must be verified. Subsequent to positive verification the operator is alerted and contingency or backup mode operations are initiated.

The intelligent representation of information is important. Operator warning messages and alarms must be concise and reflect the cause of a condition as opposed to its symptoms. For example, consider the failure of a pump which is signified by abnormally low pressures and flows. In this case diagnostics must reflect pump failure as opposed to issuing redundant alarms indicating abnormal pressures and flows.

4.1.3 Contingency operations

Subsequent to the detection of faults or disturbances, contingency operations may be initiated. Contingency operations are alternate strategies of closed-loop control. In the event of equipment failures, large disturbances, startup or shutdown, it is the task of the supervisor to select default or backup settings for the LRPC algorithms. Furthermore, the contingency operations reflect the best possible control strategy of the affected and interacting processes.

4.1.4 Operator interface

The process operator represents the final or highest authority within the process control hierarchy. Although the ES supervisor is perceived to be fully automated in an on-line real-time environment, ultimately it is the process operator who may overrule the ES supervisor. Depending upon the circumstances the operator may find the need to turn off the operation of one or more ES supervisor components (e.g. LRPC tuning, diagnostics etc. may be turned off).

The man-machine interface is of utmost importance. Messages and recommendations must be clear, concise, and revealing to process operators. In the area of fault detection and diagnostics for example, the use of color-coded on-screen schematics indicating the extent of the fault and/or diagnosis would be be very useful. The ES supervisor must display intelligent information and, upon request, provide additional data.

4.2 G2

The real-time expert system development tool - G2 - is utilized to implement a subset of the ES supervisor. This section provides a brief overview of the facilities and development environment provided by G2. Although the illustrative examples in the following sections relate directly to LRPC, G2 is **not** specific to process control applications. The interested reader can find general discussions on G2 in the literature (Moore *et al.* 1990, Wolfe 1987) and detailed information in the reference manuals (Gensym 1990,1991). Current uses of G2 include:

- Process Control
- Computer Integrated Manufacturing
- Financial Market Trading
- Automatic Testing
- Network Monitoring
- Autonomous Robots
- Simulation
- Satellite Monitoring
- Defense Systems
- Environmental Systems
- Office Decision Systems

4.2.1 Hardware Platform

G2 is available for a number of hardware platforms (e.g. Symbolics 3600, Texas Instruments Explorers and Microexplorers, MicroVAX VMS, Sun-3 and Sun-4, SPARC stations, DEC stations, HP-9000 300, 700 and 800 series, Macintosh II) and is installed at the Department of Chemical Engineering's research facility on a HP-9000 model 370 workstation which uses the HP-UX disk operating system. Two other HP-9000 series 300 workstations, namely a model 340 and a model 320, are linked to the model 370 forming a cluster. The model 370 workstation uses the Motorola 68030 microprocessor and operates at a clock speed of 33 MHz. The HP-UX disk operating system offers a real-time multi-tasking environment. The versions of the disk operating system and G2 are 7.0 and 2.1 respectively.

4.2.2 General Environment

G2 is a real-time object oriented rule based expert system development tool and is written in the programming language Common Lisp. Figure 4.3 illustrates the basic building blocks of a rule based ES:

- rule base
- data base
- inference engine
- user interface.

The inference engine uses the rules and procedures to infer how to respond to the conditions of variables and parameters contained in the data base. In addition to these components, the G2 Standard Interface (GSI) allows the developer to link his application to external routines. GSI is a general link to the "outside world" and is written in the programming language 'C'. External routines can range from simple data input/output (I/O) drivers, to complex real-time applications. External applications can communicate with G2 through GSI provided their host machines support Gensym's Intelligent Communication Protocol (ICP) which provides the handshaking. In addition, G2 provides many built-in expressions and functions and has the facility to import simple functions written in FORTRAN or 'C' directly into the application.

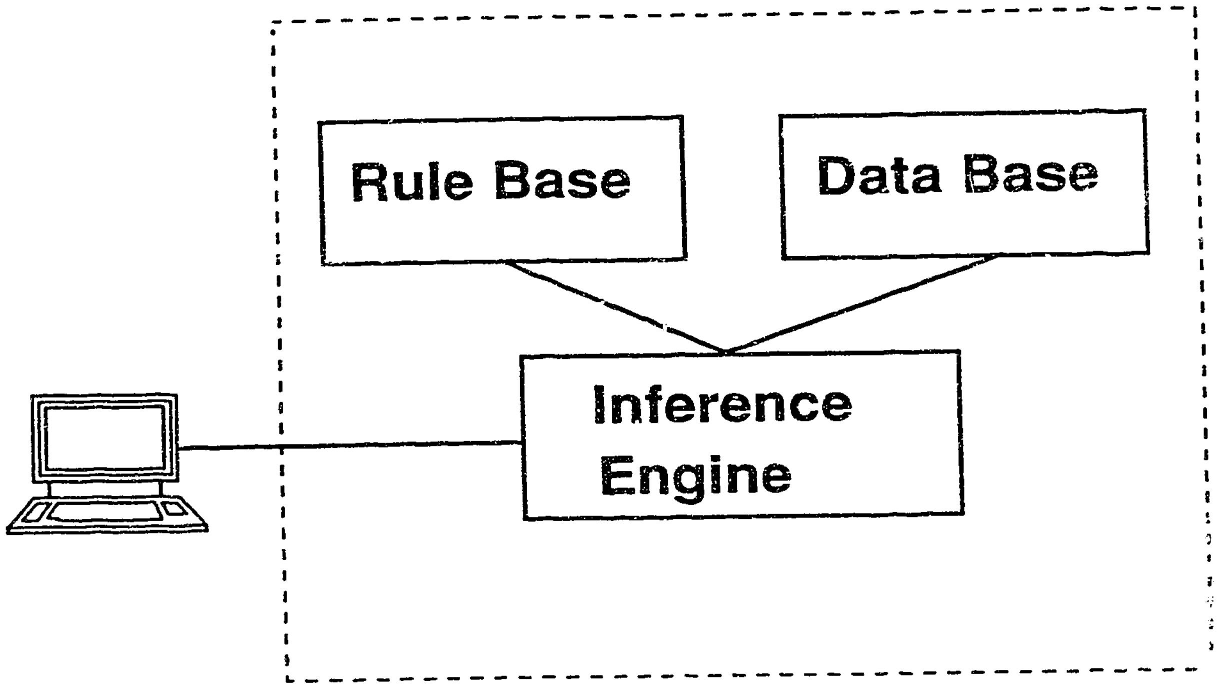


Figure 4.3 Components of a rule based expert system

An application of G2 to a specific domain requires the development of the knowledge base (KB). As G2 is object oriented, development begins with the definition of objects called classes. A class hierarchy is established by defining sub-classes which inherit the attributes (in the simplest case attributes are variables and parameters) from their superior-classes. Instances of classes are interconnected in a manner reflecting the schematic of the physical problem under investigation. The inference engine infers and reasons using the interconnections among objects.

Knowledge about objects and their interconnections is encoded with rules, formulas, and procedures. Deep knowledge is represented in terms of formulas and procedures. Formulas can range from simple expressions to differential equations which are solved numerically with either the Euler method or the fourth-order Runge-Kutta method. Procedures are lists of actions to be performed sequentially. Heuristics or shallow knowledge is encoded in rules. Rules can be generic (i.e. applicable to an entire class of objects) or specific to one object. The general form of a generic rule is:

```
for any <class>
if <antecedent>
then <consequent>
```

A specific rule, on the other hand, does not contain the "for any <class>" statement. In addition, rules can be forward chaining or backward chaining, and are written in an expressive English-like syntax. The search space can be considerably narrowed via the use of the focus and invoke commands. A built-in application simulator is also provided and may be used to test all or portions of a KB; or as is frequently done in diagnostics, run a simulation in parallel with a working process to detect and diagnose faults.

The concept of real-time is represented within G2 in terms of update intervals, validity intervals, and scan intervals. Update intervals tell G2 how frequently to collect data for variables. Validity intervals represent the length of time during which the current value of a variable remains relevant. Analysis of past data is performed with built-in statistical functions. A scan interval is a rule attribute which directs the inference engine to regularly invoke a particular rule. The shortest time interval is 1 second.

G2 employs a menu-driven windowing environment allowing the end-user or developer to view many items at once. Menus are made to appear by depressing and then releasing a mouse button (called clicking). A selection is made by clicking on the appropriate menu choice. A particular G2 application, the KB, consists of rules, schematic diagrams, graphs, tables, dials, object definitions, etc. which are placed on workspaces. Workspaces dynamically adjust to the required size and can also be manually hidden, moved, and scaled as required. End-user controls, which consist of action buttons, radio buttons, check boxes, sliders, and type-in boxes, allow the end-user and developer to enter data or commence the execution of procedures or invocation of rules merely by clicking on the appropriate choice. G2 provides an icon editor with which the shape and color of objects is defined. The color of objects can be changed and flashed by rules (indeed objects can be moved on their workspaces by rules !) and can be used to capture and focus the attention of the end-user. A context sensitive rule editor facilitates <u>easy</u> rule entry and modification. A KB can be tested and debugged using various built-in features. This friendly and interactive environment is powerful and enjoyable to work with.

4.3 ALPS

The Adaptive Long range predictive control Performance Supervisor - ALPS - is implemented with the real-time expert system development tool G2 and performs performance tuning of several constrained MGPC algorithms simultaneously. The tuning mechanisms of ALPS as well as the constrained MGPC algorithms are generic and can be applied to MIMO processes of any dimension n × m. To illustrate the use of ALPS, three 2 × 2 processes have been selected and linked to ALPS using the GSI interface. This configuration is illustrated in figure 4.4. This figure also shows the commercial quadratic optimisation software package QPSOL which is linked to the MGPC algorithm (recall that the general solution to the quadratic objective function subject to rate and amplitude constraints on the inputs and amplitude constraints on the outputs yield a quadratic programming problem, see also section 2.3.5.4).

Section 4.3.1 lists the specifications for ALPS which is a subset of the generic ES Supervisor presented in section 4.1. Although ALPS provides many features, its emphasis is on performance tuning of the constrained MGPC algorithms so as to maintain user specified regulatory and servo control performance. Section 4.3.2 presents options that have been incorporated into ALPS for the purpose of training plant personnel in utilizing ALPS as well as the MGPC algorithm(s). Sections 4.3.3 details the ALPS.kb knowledge base and the tuning strategies employed. To give the reader an appreciation of the magnitude of the developed software, table 4.1 lists the number of source code lines for the major modules and their programming languages. The MGPC module is the generic (i.e. of general dimension n × m) control algorithm allowing on-line updating of all MGPC tuning parameters considered in this thesis. This module uses a mixed language interface (FORTRAN/C) to QPSOL, and provides both the constrained and unconstrained solutions. The GSI-link module interfaces to actual processes or a process simulator. The process dimension of the GSI-link is currently limited to a maximum of 2 × 2 (i.e. processes can be of dimension 1 × 1, 1 × 2, 2 × 1, or 2 × 2). Table 4.2 shows the relative size of the developed ALPS.kb knowledge base in terms of its components. The tuning

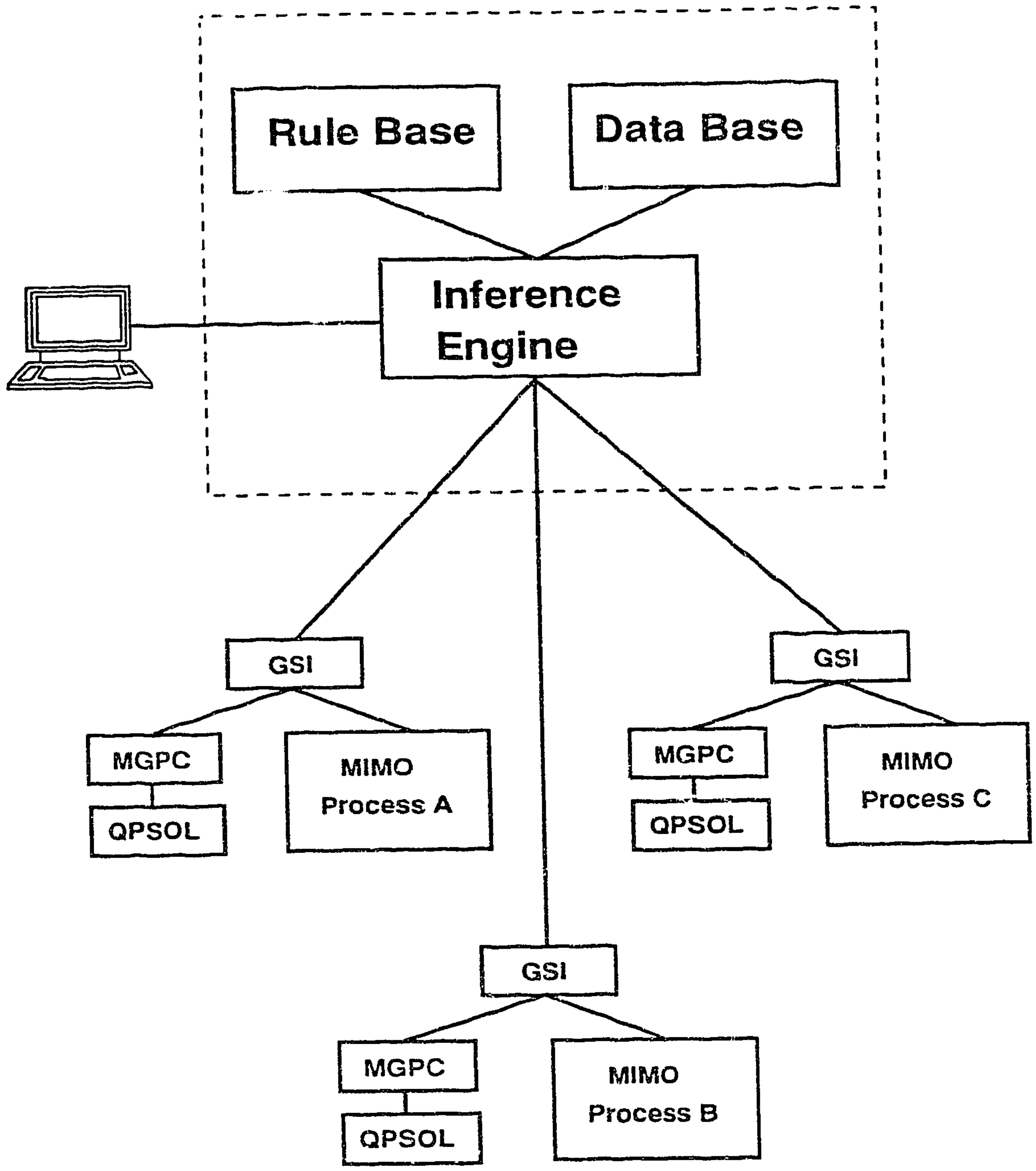


Figure 4.4 Block diagram of ALPS linked to three external MIMO processes through GSI.

strategies employed by ALPS are generic to accommodate processes of dimension n × m. The detection of the rise time is currently limited to processes with up to 2 outputs, but is easily extended.

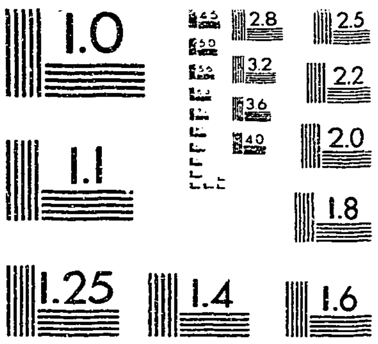
Module	Language	No. lines of code
MGPC	C	3500
GSI-Link	C	1200
QPSOL*	FORTRAN	6000

Table 4.1 Programming language and size of developed source code.
(* QPSOL is a commercial software package)

Item	Quantity
Rules	326
Procedures	9
Formulas	7
Readouts	42

Table 4.2 Size of ALPS.kb in terms of its components.

2



4.3.1 Specifications of ALPS

This section points out the criteria implemented as part of ALPS in comparison to the generic specification of the ES supervisor architecture presented in section 4.1.

4.3.1.1 Closed-loop performance supervision and LRPC tuning

The closed-loop performance specifications utilize the MGPC algorithm's two degree of freedom structure. The user specifies both servo and regulatory control performance criteria. These specifications along with the appropriate LRPC tuning parameters are listed in table 3.1. Note that each of the three MGPC algorithms, as shown in figure 4.4, is tuned independently.

ALPS recognizes the limitations of the MGPC algorithms. Warning messages are issued to the process operator questioning the validity of the appropriate performance specification whenever this specification is not met and,

- the upper bound of $N_2 = 40$ was used during the last two consecutive tuning periods
- the lower bound of N_2 (given by equations 3.9 and 3.10) was reached during the last two consecutive tuning periods
- the lower limit of the first order $T(q^{-1})$ filter, given by equation 3.15, was reached during the last two consecutive tuning periods
- the upper limit of the second order $T(q^{-1})$ filter, given by equation 3.16, was used during the last two consecutive tuning periods.

4.3.1.2 Fault and disturbance detection and diagnostics

ALPS detects communication link failures between GSI and the external MGPC algorithms. In such an event ALPS closes the communication link, informs the process operator, and continues operation with the remaining processes unaffected. No further fault and disturbance detection is performed.

4.3.1.3 Contingency operations

Contingency operations illustrating the use of setpoint scheduling and "tight" input rate constraints, which may be useful during process startup and shutdown, are provided. Also, in the event of a GSI-MGPC communication link failure the switch to manual/backup mode for the affected process is indicated with a warning message to the process operator.

4.3.1.4 Operator interface

ALPS uses an integrated, color coded, windowing environment. This environment, shown only in black and white due to the limitations of the screen dump facility, is shown in figure 4.5. This figure was generated after sliding apart overlapping workspaces. Notice that there are three workspaces for each set of functions: manual setpoint changes by operator, manual adjustment of output horizons and control horizons, selection of options, selection of noise and disturbance options, and graphs. When ALPS is in use, only the very top workspaces are in view. The operator may "leaf through" overlapping workspaces by clicking with the mouse on action buttons (shown as oval icons). For example, by clicking on the action button labelled Plots_2 the workspace showing the graphs for the second process (named stripper_2, it also has the text SHELL to its right) is pulled onto the top. Also, by clicking on the action button labelled Hide, the particular workspace completely disappears from view.

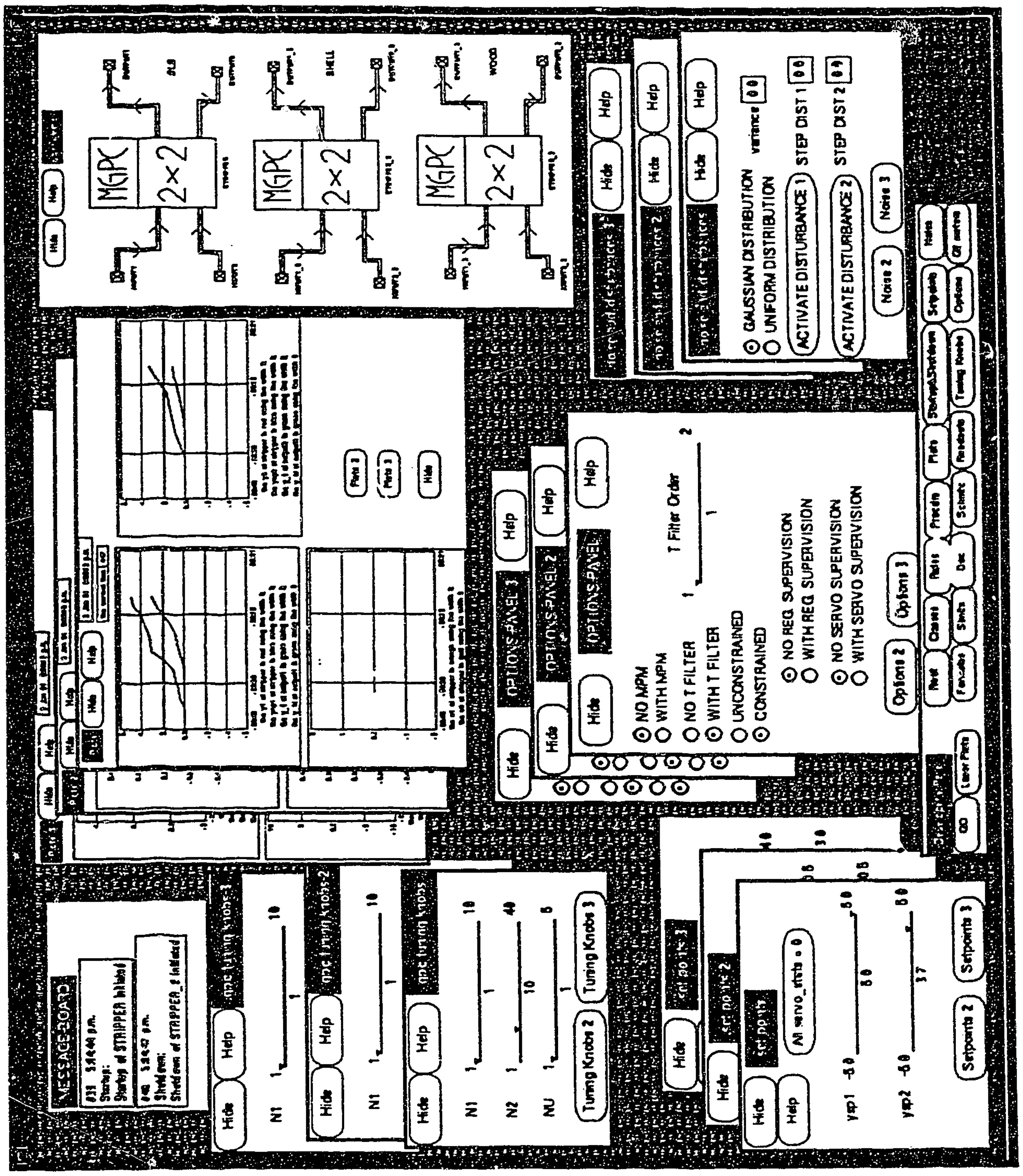


Figure 4.5 Integrated windowing environment provided by ALPS.

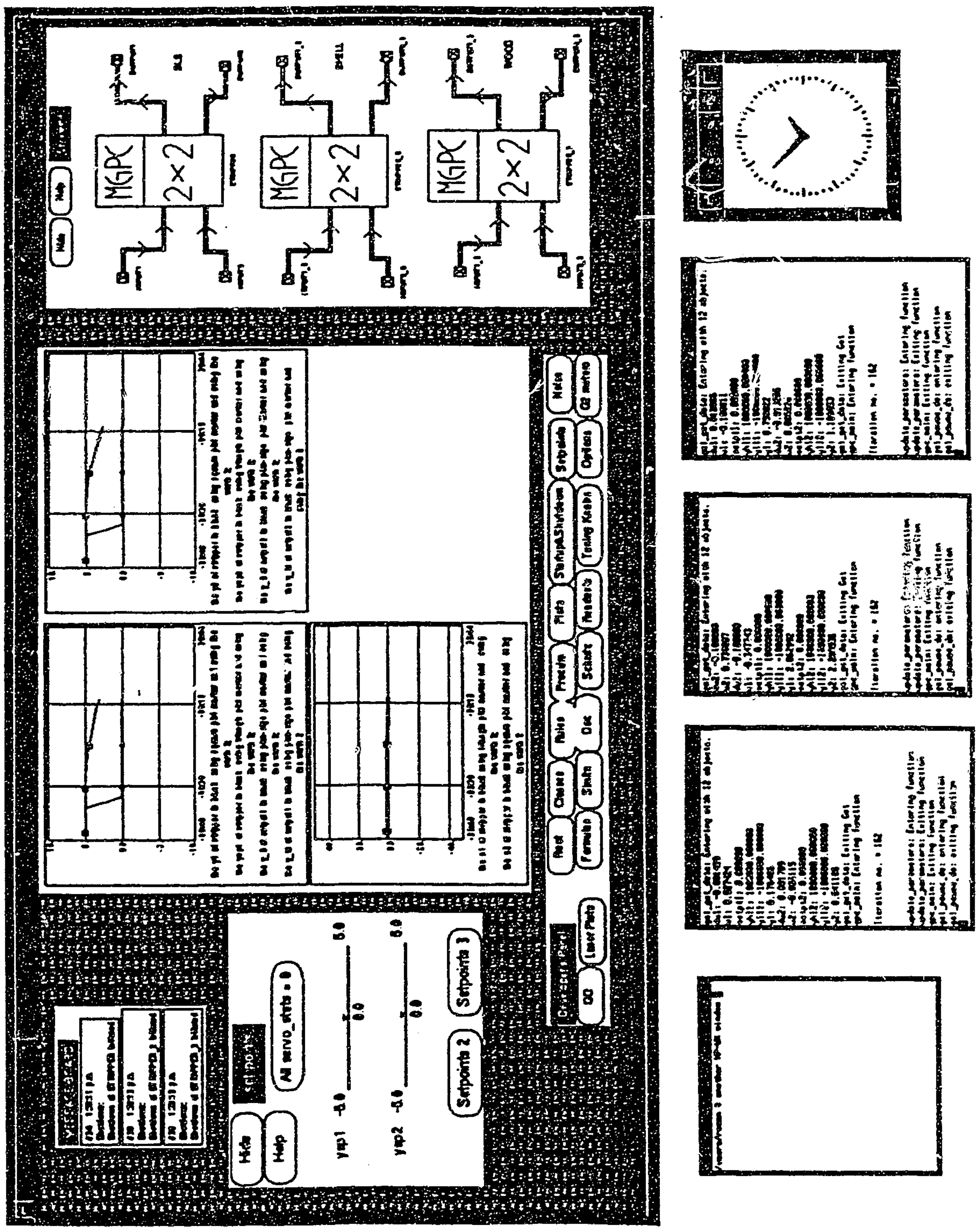


Figure 4.6 Typical workspace layout as seen by the process operator.

The workspace called Developer's Panel, at the bottom of figure 4.5, serves as the command panel from which all workspaces can be brought into view by clicking with the mouse. Specific information about any object is obtained by clicking on its icon. By clicking on the icon of INPUT1 for example, the operator can access its attribute table which lists all of its information in detail (this is discussed further in section 4.3.3). Figure 4.6 is a snap-shot in time of a more typical workspace layout as seen by the process operator. This figure also shows additional HP-UX windows. Three of these windows are used by the MGPC algorithms (one MGPC algorithm per window, external to G2) to display the data I/O transferred over the GSI-MGPC links. An extra HP-UX window, which the operator can use for any additional tasks, and the real-time HP-UX clock are also shown. This environment is efficient, allows quick access to workspaces and their contents, and shows only the requested information.

ALPS is fully automated and keeps the process operator informed of actions performed with appropriate messages (see the MESSAGE-BOARD workspace in figure 4.6). Also, the operator can - at any time - change setpoints and manually adjust MGPC tuning parameters by using the provided end-user controls. Process outputs, setpoints, regulation band r_b, and inputs are displayed on graphs versus time. Readout-tables showing the variance of process inputs, standard deviation of process outputs, etc. are also provided. Process inputs and outputs are graphically interconnected forming a schematic (see the Schematic workspace in figure 4.6).

The operator can disable regulatory and/or servo response tuning activities. With both servo and regulatory response supervision disabled, the role of ALPS is reduced to providing an interactive real-time interface between the process operator, the MGPC algorithms, and the processes. ALPS merely schedules data I/O and the execution of the MGPC algorithms at every sampling interval. This means that the block diagram of figure 4.1 is valid, provided the EXPERT SYSTEM SUPERVISOR block is removed.

4.3.2 Training facilities provided by ALPS

This section presents the options that have been built into ALPS for the purpose of training plant personnel. The options can be divided into two categories to facilitate the training of:

a) ALPS, and
b) constrained MGPC.

4.3.2.1 Training in the use of ALPS

Training in the use of ALPS by plant personnel, such as engineers and process operators, is facilitated. Personnel new to ALPS may initially disable the regulatory and servo response supervision modes and concentrate on becoming familiar with the interactive windowing environment. It is important to access the proper workspace for the particular process of interest. By utilizing an external process simulator (built into the MGPC algorithm) ALPS can be run in either real-time or simulated-time (change to simulated-time using G2's Timing Parameters options menu). Engineers and process operators can now become familiar with using sliders to change setpoints, output prediction horizons, and control horizons.

Having mastered the windowing environment, the regulatory and servo response supervision modes can be enabled and studied. ALPS now tunes the MGPC parameters so as to achieve or maintain the desired closed-loop performance specifications. This can be verified by comparing the actual performance with the specified performance. By clicking on the icon named Rules (found on the Developer's Panel) the workspaces containing the rules responsible for tuning the MGPC algorithms come into view. Selecting the option "Highlight Invoked Rules" from G2's Run Options menu causes G2 to flash invoked rules. Thus, whenever the inference engine updates the MGPC tuning parameters the invoked rules are flashed and can be studied in detail.

4.3.2.2 Training in the use of constrained MGPC

ALPS has several facilities useful for learning the properties of MGPC. By using a process simulator (see section 4.3.2.1) the following options can be utilized for any given process, independent of other processes:

- MPM or no MPM
- $T(q^{-1})$ filter or no $T(q^{-1})$ filter
- switch from first to second order $T(q^{-1})$ filter, and vice versa
- constrained solution using QPSOL or unconstrained solution using equation 2.19

The following disturbance sources have been implemented in order to make the process simulations realistic:

- Input step disturbances, $d_u(t)$, of a user specified amplitude can be implemented at any input channel
- Stochastic zero-mean output disturbances with a Gaussian or Uniform distribution, with a user specified variance, can be implemented for each process independently.

The user may make his selection at any time using the mouse.

The power spectral densities of both the Gaussian and the Uniform noise distributions are shown in figure 4.7. This figure was generated with MATLAB's spectrum command using 2048 data points and shows the $\log_{10}$(power spectral density) versus normalized frequency. The power spectral density of Gaussian noise is relatively constant throughout its entire frequency range. In contrast, the power spectral density of

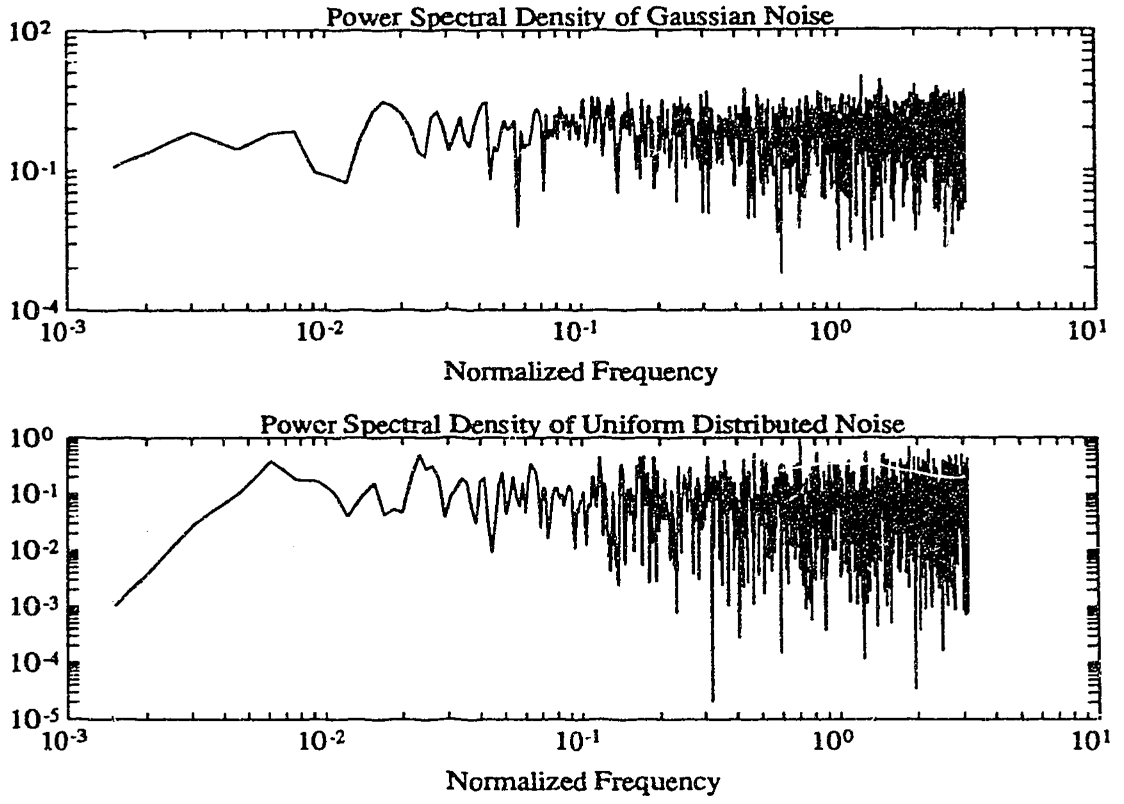


Figure 4.7 Power spectrum of the Gaussian and Uniform noise distributions.

Uniform noise drops off by about 2 orders of magnitude in its low frequency range. Thus the use of Gaussian noise is recommended as it consistently excites all frequencies throughout its spectrum.

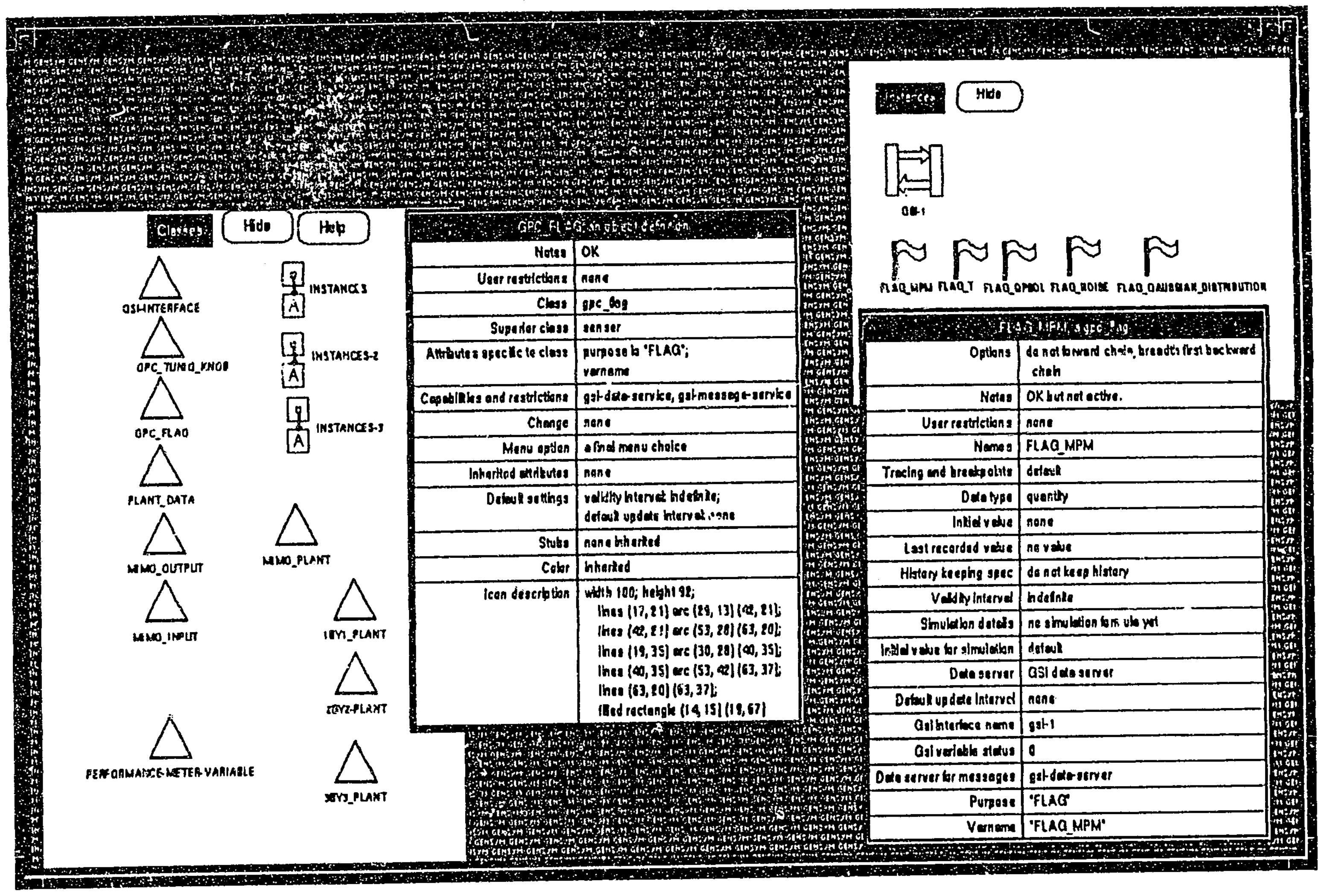
4.3.3 The ALPS.kb knowledg')ase

This section details the developed knowledge base ALPS.kb. The G2 inference engine uses the knowledge encoded in ALPS.kb in terms of its rules and procedures, to conduct performance tuning of constrained MGPC subject to conditions of MPM and disturbances. Should both the servo response supervision and regulatory response supervision modes be disabled by the operator, then the role of ALPS is reduced to providing a real-time on-line interface by scheduling activities among the external MGPC algorithms and data I/O.

The ALPS.kb knowledge base does not reflect the most elegant or efficient use of G2 built-in functions. Instead it utilizes as many G2 features as possible. The sections to follow provide an overview with emphasis on the regulatory and servo control tuning strategies employed. All rules, procedures, and formulas as used by ALPS for the purpose of tuning are listed in appendix A. Rules, procedures, and formulas which provide the user interface are numerous and mundane and have been omitted from the appendix for the sake of brevity.

4.3.3.1 Introduction to ALPS.kb

As G2 provides an object-oriented programming environment, development of ALPS starts with the definition of classes. Consider figure 4.8. This figure shows various classes which are represented by triangles on the workspace named Classes. Upon clicking on the icon of GPC_FLAG the table named "GPC_FLAG, an object definition"



GPC_FLAG, an object definition

Notes	OK
User restrictions	none
Class	gpc_flag
Superior class	sensor
Attributes specific to class	purpose is 'FLAG'; varname
Capabilities and restrictions	gsi-data-service, gsi-message-service
Change	none
Menu option	a final menu choice
Inherited attributes	none
Default settings	validity interval: indefinite; default update interval: none
Stubs	none inherited
Color	inherited

Icon description	width 100; height 92;
	lines (17, 21) arc (29, 13) (42, 21);
	lines (42, 21) arc (53, 28) (63, 20);
	lines (19, 35) arc (30, 28) (40, 35);
	lines (40, 35) arc (53, 42) (63, 37);
	lines (63, 20) (63, 37);
	filled rectangle (14, 15) (19, 67)

FLAG_MPM, a gpc_flag

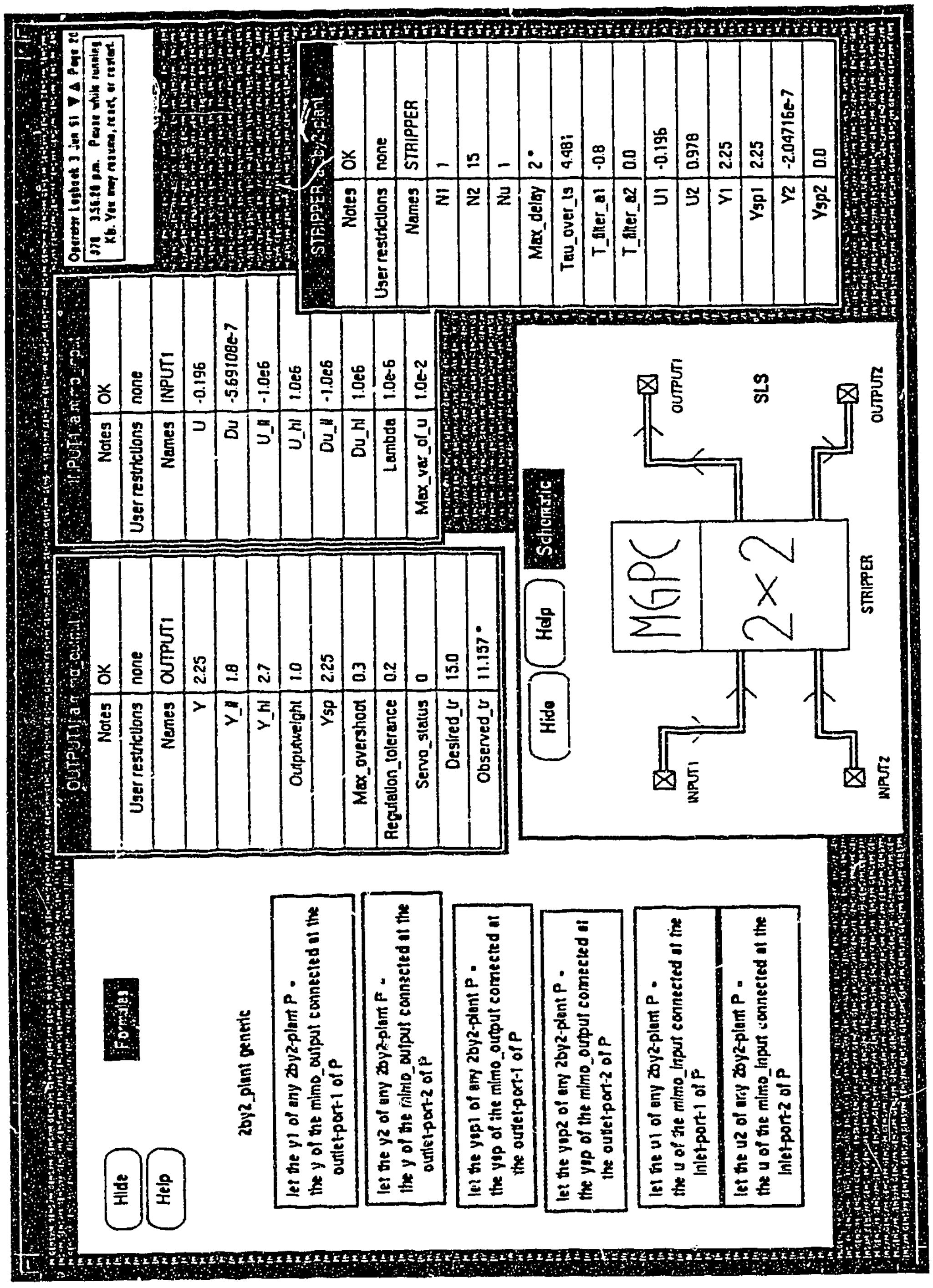
Options	do not forward chain, breadth first backward chain
Notes	OK but not active.
User restrictions	none
Names	FLAG_MPM
Tracing and breakpoints	default
Data type	quantity
Initial value	none
Last recorded value	no value
History keeping spec	do not keep history
Validity interval	indefinite
Simulation details	no simulation formula yet
Initial value for simulation	default
Data server	GSI data server
Default update interval	none
Gsi interface name	gsi-1
Gsi variable status	0
Data server for messages	gsi-data-server
Purpose	'FLAG'
Varname	'FLAG_MPM'

Figure 4.8 Class definitions of ALPS.kb

appears. This table shows the attributes for this class such as: name, superior class, attributes, color, icon definition etc. . The workspace named Instances shows five different instances of the class GPC_FLAG, they are: FLAG_MPM, FLAG_T, FLAG_QPSOL, FLAG_NOISE, and FLAG_GAUSSIAN_DISTRIBUTION. By clicking on the icon of FLAG_MPM its table named "FLAG_MPM, a gpc_flag" appears. This table lists all information on FLAG_MPM, such as its data type, last recorded value, history keeping specification, default update interval, data server, as well as its inherited attributes, Purpose and Varname. Notice that instances of GPC_FLAG are GSI variables and they are used to set particular options within the (external to G2) MGPC algorithm through the GSI-INTERFACE object GSI-1. FLAG_MPM, for example, if set to '1' tells the MGPC algorithm to use MPM in its control calculations, and when set to '0' no MPM is used in control calculations. The purpose of the remaining GPC_FLAG's is easily deduced by reconsidering section 4.3.2.2.

Through the use of superior-classes and sub-classes a hierarchy can be established. Sub-classes inherit the attributes from their superior classes. The class definition MIMO_PLANT in figure 4.8 is the superior-class to 1BY1_PLANT, 2BY2_PLANT, and 3BY3_PLANT. MIMO_PLANT is used to define attributes that are common to its sub-classes and inherited throughout the class hierarchy. These common attributes are the output prediction horizons and the control horizon (i.e. N_1, N_2, NU); as well as the maximum delay of the MIMO process' delay matrix, Max_delay, and the $T(q^{-1})$ filter parameters: Tau_over_Ts, T_filter_a1, and T_filter_a2. The sub-classes 1BY1_PLANT, 2BY2_PLANT, and 3BY3_PLANT are used to define their icons as well as stubs (i.e. connections) for 1 input and 1 output connections, 2 input and 2 output connections, and 3 input and 3 output connections respectively. Thus the naming convention adopted refers to MIMO processes (of dimension $n \times m$) as nBYm_PLANT.

The inference engine infers and reasons using the interconnections among objects. Figure 4.9 shows an instance of a 2BY2_PLANT named STRIPPER connected to



Operator Logbook 3 Jan 91 Page 2
378 3:56:28 p.m. Pause while running Kb. You may resume, reset, or restart.

2by2_plant generic

let the y1 of any 2by2-plant P = the y of the mimo_output connected at the outlet-port-1 of P

let the y2 of any 2by2-plant P = the y of the mimo_output connected at the outlet-port-2 of P

let the ysp1 of any 2by2-plant P = the ysp of the mimo_output connected at the outlet-port-1 of P

let the ysp2 of any 2by2-plant P = the ysp of the mimo_output connected at the outlet-port-2 of P

let the u1 of any 2by2-plant P = the u of the mimo_input connected at the inlet-port-1 of P

let the u2 of any 2by2-plant P = the u of the mimo_input connected at the inlet-port-2 of P

OUTPUT1, a mimo_output

Notes	OK
User restrictions	none
Names	OUTPUT1
Y	2.25
Y_ll	1.8
Y_hl	2.7
Outputweight	1.0
Ysp	2.25
Max_overshoot	0.3
Regulation_tolerance	0.2
Servo_status	0
Desired_tr	15.0
Observed_tr	11.157 *

INPUT1, a mimo_input

Notes	OK
User restrictions	none
Names	INPUT1
U	-0.196
Du	-5.69108e-7
U_ll	-1.0e6
U_hl	1.0e6
Du_ll	-1.0e6
Du_hl	1.0e6
Lambda	1.0e-6
Max_var_of_u	1.0e-2

STRIPPER, a 2by2-plant

Notes	OK
User restrictions	none
Names	STRIPPER
N1	1
N2	15
Nu	1
Max_delay	2 *
Tau_over_ts	4.481
T_filter_a1	-0.8
T_filter_a2	0.0
U1	-0.196
U2	0.978
Y1	2.25
Ysp1	2.25
Y2	-2.04716e-7
Ysp2	0.0

Figure 4.9 The schematic of the 2 × 2 process named STRIPPER

Quantity	Meaning
Y	$y_1(t)$
Y_ll	$y_{1,min}$
Y_hl	$y_{1,max}$
Output weight	$(\gamma_1)^{½}$
Ysp	$w_1(t)$
Max_overshoot	overshoot specification
Regulation_ tolerance	r_b specification
servo_status	if > 0 implies setpoint change in progress
Desired_tr	t_r specification
Observed_tr	measured t_r

Table 4.3 Attribute table OUTPUT1, a MIMO_OUTPUT

instances of MIMO_INPUT's and MIMO_OUTPUT's. The inference engine uses the generic formulas, located on the workspace FORMULAS of this figure, to update the values of U1 through Ysp2 of STRIPPER. This causes the inference engine to seek current data from the appropriate (instance of) MIMO_INPUT or MIMO_OUTPUT. Furthermore, this figure shows the attribute tables of OUTPUT1, INPUT1, and STRIPPER which are obtained by clicking on the icon of the respective objects. The attribute table "OUTPUT1, a mimo_output" shows all information pertaining to output channel 1 of STRIPPER. These quantities and their meaning are listed in table 4.3. Similarly, the attribute table "INPUT1, a mimo_input" lists all information pertaining to input channel 1 of STRIPPER. These quantities and their meaning are listed in table 4.4. Quantities listed in attribute tables of figure 4.9 with a star (*) indicate that they have expired.

Quantity	Meaning
U	$u_1(t)$
Du	$\Delta u_1(t-1)$
U_ll	$u_{1,min}$
U_hl	$u_{1,max}$
Du_ll	$\Delta u_{1,min}$
Du_hl	$\Delta u_{1,max}$
Lambda	λ_1
Max_var_of_u	σ^2_u specification

Table 4.4 Attribute table INPUT1, a MIMO_INPUT

Timing is critical in real-time applications. A sampling time of $T_s = 2$ seconds is used for simulation purposes. All variable update intervals reflect this sampling time. The MGPC algorithms are executed by scanning a generic rule. This causes the inference engine to invoke this rule for each instance of a MIMO_PLANT. This rule unconditionally executes the MGPC algorithms as external procedure calls.

Tuning to meet servo performance criteria is initiated whenever a setpoint change occurs. Should the process operator, or a procedure executed by the operator, perform a setpoint change in between sampling intervals, then the servo_status parameter of this (instance of) MIMO_OUTPUT is incremented by 1. Monitoring of t_r for the particular (instance of) MIMO_OUTPUT is initiated if its servo_status changed from 0 to 1 and the servo_status of all <u>other</u> (instances of) MIMO_OUTPUTs connected to the (instance of) MIMO_PLANT are 0. This strategy circumvents erroneous measurements of t_r obtained due to coupling from other channels. The value of servo_status is decremented by 1 after a fixed time interval. The output response is monitored during this "window" and t_r is calculated. For simulation purposes a "window" of 30 sampling instances is used.

Thereafter adjustment of N_2 is performed if NU = 1, and all (instances of) MIMO_OUTPUTs connected to the (instance of) MIMO_PLANT have current values for their Observed_tr variables. Since slowly changing plant dynamics may render Observed_tr obsolete, an expiration period is utilized. For simulation purposes an expiration period of 10 minutes is used for all Observed_tr variables.

The ALPS.kb knowledge base has been structured in a manner reflecting its purpose. Figure 4.10 shows the workspace hierarchy consisting of:

a) Global Supervisor
b) Regulatory Response Supervisor
c) Servo Response Supervisor

Each of these workspaces contains the knowledge, as used by the inference engine, to perform the global, regulatory, and servo supervisory functions. The tuning strategies used by ALPS to adjust the MGPC parameters and the G2 features used to perform these tasks is the subject of the following sections.

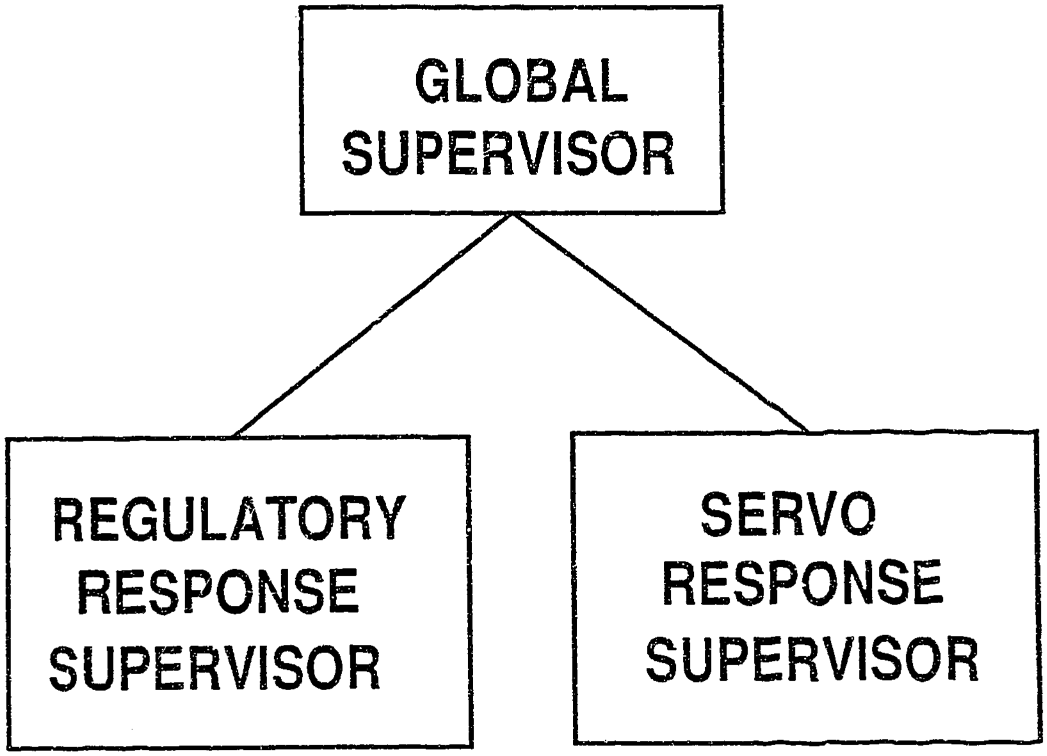


Figure 4.10 The workspace hierarchy of ALPS.kb

4.3.3.2 The Global Supervisor workspace

This workspace contains rules causing the inference engine to:

- monitor GSI-MGPC communication failure,
- set default values for MGPC parameters,
- detect changes in N_1, N_2, and NU,
- additional rules that respond to selected options (see section 4.3.2.2)

A simplified example of a forward chaining rule alerting the operator and deactivating a subworkspace in the event of a communication failure is:

```
if(the value of the gsi-interface-status of gsi-1 < 0)
then inform the operator for the next 30 seconds that"
*****************************************************

ERROR: Interface GSI-1 has lost
its communication link.

Process Control switched to MANUAL.

*****************************************************"
and deactivate the subworkspace of <name>
```

Note that rules detecting GSI failures can not be generic.

The generic rule which concludes that all observed rise times of all (instances of) MIMO_OUTPUTs connected to an (instance of) MIMO_PLANT have become invalid because its output horizons or control horizons have changed either manually by the process operator or due to tuning of N_2 by the servo supervisor is:

```
for any MIMO_PLANT PLANT
for any MIMO_OUTPUT OUTP connected to PLANT
whenever the N2 of PLANT receives a value or the N1 of PLANT receives a
value or the NU of PLANT receives a value and
when ( the N2 of PLANT /= the N2 of PLANT as of 2 seconds ago or the N1 of
PLANT /= the N1 of PLANT as of 2 seconds ago or    the NU of PLANT /= the
NU of PLANT as of 2 seconds ago )
then
conclude that the Observed_tr of OUTP has no current value
```

Where the value of 2 seconds reflects the sampling time $T_s = 2$ seconds. This rule is invoked by the inference engine, for example, with PLANT set to STRIPPER and OUTP set to OUTPUT1 and OUTPUT2 (see figure 4.9).

In conclusion, the rules invoked by the inference engine located on the Global Supervisor workspace are generic and specific forward chaining rules. In addition, this workspace contains generic forward chaining rules utilizing the rule attributes, focal classes and categories to respond to user selected options (see section 4.3.2.2). Generic "initially rules" are used to set defaults.

4.3.3.3 The Regulatory Response Supervisor workspace

This workspace contains the generic rules developed for tuning Γ and $T(q^{-1})$. Tuning to meet regulatory performance criteria is initiated periodically at fixed intervals. Tuning is suspended unless the attribute servo_status of all (instances of) MIMO_OUTPUT connected to the particular (instance of) MIMO_PLANT are 0. A servo_status > 0 implies that a setpoint change is in progress for that particular output channel. This strategy circumvents erroneous tuning of $T(q^{-1})$ and Γ while a setpoint change is in progress. The tuning strategies for Γ and $T(q^{-1})$, along with the adopted G2 features used to implement these strategies, are the topic of the following two sections:

4.3.3.3.1 Tuning of Γ

Tuning of Γ is initiated periodically by scanning two mutually exclusive rules at fixed intervals. The first rule fires (i.e. it is invoked by the inference engine and its antecedent evaluates to true causing the rule's consequence to be executed) if a setpoint change is in progress and sends a message to the operator. Keeping an eye on the not, this rule is:

```
for any MIMO_PLANT PLANT
if(not(for every MIMO_OUTPUT OUTP connected to PLANT)
              (the servo_status of OUTP = 0)))
then
inform the operator for the next 10 seconds that"
Regulation Supervisor:
Setpoint change in progress for [the name of PLANT];
hence suspending the updating of
output weighting for [the name of PLANT]"
```

The second rule fires if there are no active setpoint changes and initiates tuning of Γ:

```
for any MIMO_PLANT PLANT
if(for every MIMO_OUTPUT OUTP connected to PLANT
        (the servo_status of OUTP = 0))
then
start OUTPUT_WEIGHTING(PLANT)
and
inform the operator for the next 10 seconds that"
Regulation Supervisor:
Updating output weighting for [the name of PLANT]"
```

The above rule starts the procedure OUTPUT_WEIGHTING and sends a message to the operator indicating that the output weighting matrix, Γ, is being updated. For simulation purposes both rules are scanned every 2 minutes. The procedure OUTPUT_WEIGHTING is listed in appendix A and it adjusts Γ based on the following:

Let	σ_{yi}	=	the standard deviation of the ith output connected to the process during the last 20 seconds, and
	max_dev	=	the maximum over all σ_{yi} values i=1,2...n and
	σ_{ymax}	=	user specified constant,

then set γ_i to

$$\gamma_i = \left(\frac{\sigma_{yi} + \sigma_{ymax}}{\text{max\_}dev + \sigma_{ymax}} \right)^2 \tag{4.1}$$

where i = 1,2, ... n

In the case where a lot of noise is present (i.e. $\sigma_{yi} >> \sigma_{ymax}$ for i=1,2...n), equation 4.1 reduces to the simple relationship

$$\gamma_i \approx (\sigma_{y_i}/\text{max\_dev})^2$$

In the other extreme, when little noise is present (i.e. $\sigma_{y_i} << \sigma_{y\max}$ for i=1,2...n) the relationship becomes

$$\gamma_i = 1$$

This procedure penalizes the noisiest channel in relationship to the remaining channels. The reasoning behind this scaling strategy is that each γ_i is scaled in direct relation to the standard deviation of y_i in comparison to the noisiest channel. The weighting of the noisiest channel will always be set to unity. The square, $(\bullet)^2$, is due to the quadratic objective function which minimizes the sum of the squares. The point beyond which scaling occurs is adjusted with $\sigma_{y\max}$.

In summary, tuning of Γ is accomplished by two (mutually exclusive) generic forward chaining rules. These rules fire at specified scan intervals and invoke a procedure to perform the adjustment of Γ.

4.3.3.3.2 Tuning of $T(q^{-1})$

The tuning of $T(q^{-1})$ is initiated periodically at fixed intervals. Actual tuning takes place only if the servo_status of all (instances of) MIMO_OUTPUT's connected to the particular (instance of) MIMO_PLANT are 0 (i.e. no setpoint change is in progress). By setting the default update interval of Tau_over_ts to 1 minute the inference engine will seek to update Tau_over_ts at this rate. Tau_over_ts is the inverse of the normalized break frequency of $T(q^{-1})$ and is an attribute of each instance of a MIMO_PLANT. Tau_over_ts is tuned according to the recommendation by McIntosh (1988). A generic formula performs the following:

Let $\sigma^2_{u,max}(t)$ = the maximum over all input variances σ^2_{ui} i = 1,2...m, during the last 10 values, and

$\sigma^2_{u,min}$ = the minimum over all input variance specifications $\sigma^2_{u,i}$ i = 1,2...m.

then

$$\frac{\frac{\tau}{T_s}(t) - \frac{\tau}{T_s}(t-1)}{\frac{\tau}{T_s}(t-1)} = w\left(\frac{\sigma^2_{u,max}(t) - \sigma^2_{u,min}}{\sigma^2_{u,min}}\right) \tag{4.2}$$

with

$$1.442 \le \frac{\tau}{T_s} \le 19.496 \tag{4.3}$$

where the relaxation factor w = 0.2, and

Tau_over_ts = τ/T_s

The time dependency (•)(t-1) and (•)(t) are the settings for the previous and current tuning durations respectively. Tau_over_ts is related to the discrete time pole c_1 of the first and second order $T(q^{-1})$ filters by:

$$c_1 = e^{-\frac{T_s}{\tau}} \tag{4.4}$$

Thus the bounds of equation 4.3 are obtained from the limits placed on c_1:

$$0.5 \le c_1 \le 0.95 \tag{4.5}$$

Equation 4.2 relates the tuning parameter setting, Tau_over_ts, to the measured and specified performance criteria $\sigma^2_{u,max}(t)$, and $\sigma^2_{u,min}$. Notice that equation 4.2 adjusts the break frequency of $T(q^{-1})$ linearly based on 1/5th of the percentage difference between the input channel with the highest measured variance and the input channel with the smallest input variance specification. The bounds on c_1 by equation 4.5 are consistent with the implementation considerations given in section 3.4.2.

Once the inference engine updates Tau_over_ts, forward chaining rules are invoked to set new $T(q^{-1})$ filter parameters. The GSI variables T_filter_a1 and T_filter_a2 are related to the second order $T(q^{-1})$ filter by:

$$\frac{1}{T(q^{-1})} = \left(\frac{1-c_1}{1-c_1q^{-1}}\right)^2 = \frac{1-a_1-a_2}{1-a_1q^{-1}-a_2q^{-2}} \tag{4.6}$$

where T_filter_a1 = a_1 and
T_filter_a2 = a_2 .

In the case of a first order $T(q^{-1})$ filter equation 4.6 requires $a_2 = 0$ and the square term $(\bullet)^2$ is replaced with unity $(\bullet)^1$. The generic formula and rules performing this are listed in appendix A.

Forward chaining rules are used to detect whether the order of the $T(q^{-1})$ filter should be switched from first order to second order and vice versa. A rule that detects that a first order $T(q^{-1})$ filter was tuned to its upper bound (i.e. $c_1 = 0.95$) twice during the last two consecutive tuning periods and hence causes a switch to the default second order $T(q^{-1})$ filter given by equation 3.13 is:

```
whenever the Tau_over_ts of STRIPPER receives a value
and when(the Tau_over_ts of STRIPPER = 19.496 and
        t-filter-order = 1 and
        the standard deviation of Tau_over_ts of STRIPPER
        during the last 2 minutes = 0.)
then
conclude that the Tau_over_ts of STRIPPER = 4.481
and conclude that t-filter order = 2
and inform the operator for the next 10 seconds that"
Regulation Supervisor:
Switching to second order T filter for STRIPPER"
```

Note that this rule is specific to the MIMO_PLANT STRIPPER. The more complicated generic rule is listed in the appendix. The rule detecting that a second order $T(q^{-1})$ filter was tuned to its lower bound (i.e. $c_1 = 0.5$) during the last two consecutive tuning periods and hence switches to the default first order $T(q^{-1})$ filter given by equation 3.12 is analogous to the above rule and is also listed in appendix A.

In addition to the $T(q^{-1})$ filter tuning rules, there are two generic rules which detect whether maximum or minimum filtering was performed over the last two consecutive tuning periods. The purpose of these rules is to inform the operator questioning the validity of the σ^2_u specifications. The assumption is that in order to meet user σ^2_u specifications the amount of filtering should fall between the maximum and minimum filtering configurations given by equations 3.14 and 3.15 respectively.

In summary, tuning of $T(q^{-1})$ is initiated by placing a default update interval on the attribute Tau_over_ts. The inference engine uses a generic formula that adjusts the values of Tau_over_ts, which in turn causes generic forward chaining rules to be invoked. These forward chaining rules are responsible for setting the $T(q^{-1})$ filter parameters of the applicable MGPC algorithm, switching the $T(q^{-1})$ filter orders from first to second and vice versa, and alert the operator when maximum and minimum filtering is applied over two consecutive tuning periods.

4.3.3.4 The Servo Response Supervisor workspace

The servo response supervisor is responsible for adjusting the output constraints y_{min} and y_{max} as well as the output prediction horizons N_2 for each process. The developed tuning strategies are discussed in detail below.

4.3.3.4.1 Tuning strategies for r_b and overshoot

The regulatory performance specification $r_{b,i}$ represents the allowable drift in $y_i(t)$ from its setpoint $w_i(t)$ while output channel 'i' is at steady state (i.e. there is no setpoint change). The servo performance specification overshoot$_i$ reflects the maximum allowable overshoot during setpoint changes in $w_i(t)$. Both of these specifications are implemented using output constraints. It is apparent that unless a setpoint change is in progress output constraints remain fixed at the following values:

$$\begin{aligned} y_{\max,i} &= w_i(t)\left(1+r_b\right) \\ y_{\min,i} &= w_i(t)\left(1-r_b\right) \end{aligned} \tag{4.7}$$

with $w_i(t) > 0$

or

$$\begin{aligned} y_{\min,i} &= w_i(t)\left(1+r_b\right) \\ y_{\max,i} &= w_i(t)\left(1-r_b\right) \end{aligned} \tag{4.8}$$

with $w_i(t) < 0$

and

$$\begin{aligned} y_{\max,i} &= +r_b \\ y_{\min,i} &= -r_b \end{aligned} \tag{4.9}$$

when $w_i(t) = 0$

Three generic rules are used to invoke the adjustment of output constraints for steady state operation. Further rules having attributes focal class, and category are used to perform the adjustments as per equations 4.7 to 4.9. For example, one generic whenever rule fires should the servo_status attribute of any (instance of) MIMO_OUTPUT be set to '0' indicating that this output channel has reached its steady state following setpoint changes. Keeping an eye on the key word "invoke", the rule is:

```
for any MIMO_OUTPUT OUT
whenever the servo_status of OUT receives a value
and when(the servo_status of OUT = 0)
then invoke apply-output-constraints rules for OUT
```

When the above rule fires, the inference engine will invoke all rules having MIMO_OUTPUT as their focal class attribute and apply-output-constraint as their categories attribute. There are two generic rules that have these attributes. One is to set the output constraints in accordance to equation 4.9 and the remaining rule sets the output constraints in accordance to equations 4.7 and 4.8 . The simplest of these two rules is:

```
for any MIMO_OUTPUT OUT
if ((the ysp of OUT = 0.0)
and (the servo_status of OUT = 0))
then set the y_ll of OUT to (-1*the regulation_tolerance of OUT)
and set the y_hl of OUT to (the regulation_tolerance of OUT)
and inform the operator for the next 10 seconds that"
Servo Supervisor:
Setting the y_ll and y_hl of [the name of OUT]
connected to [the name of the MIMO_PLANT connected to OUT] for normal
operation"
```

Focal Classes	MIMO_OUTPUT
Categories	apply-output-constraints

If the process operator disables the servo supervision mode, all output constraints are set to very large values (i.e. $\pm 10^6$). Upon startup of ALPS, and whenever the operator enables the servo supervision mode, the following generic rule is invoked:

```
initially
for any MIMO_PLANT PLANT
for any MIMO_OUTPUT OUTP connected to PLANT
unconditionally invoke apply-output-constraints rules for OUTP
```

Initially rules are useful to set defaults.

Two generic whenever rules are used to update the output constraints for setpoint changes. One rule serves to set the y_{max} for positive setpoint changes, while the remaining rule serves to set y_{min} for negative setpoint changes. The former of these rules is:

```
for any MIMO_OUTPUT OUT
whenever the ysp of OUT receives a value
and when( the ysp of OUT > the value of the ysp of OUT as of 1 second ago)
then set the y_hl of OUT to ( the ysp of OUT +
        (the ysp of OUT - the value of the ysp of OUT
        as of 2 seconds ago)*(the max_overshoot of OUT))
and inform the operator for the next 10 seconds that"
Servo Supervisor:
Updating the y_hl of [the name of OUT]
connected to [the name of the MIMO_PLANT connected to OUT]"
```

In summary, the knowledge on how to update output constraints during setpoint changes as well as for steady state operation is encoded into generic forward chaining rules and generic whenever rules. The rule attributes focal classes, and categories are also used. Generic initially rules are used to set defaults.

4.3.3.4.2 Output horizon tuning strategy

The tuning strategy developed to update the output prediction horizon N_2 is the topic of this section. As to what N_2 adjustment mechanism represents the best answer to meeting and maintaining user specified servo performance criteria is open to discussion. The ground work of chapter 3, however, indicates that there is a strong correlation between rise time t_r and N_2, provided NU = 1. The strategy implemented by ALPS is a trade-off in terms of the accuracy of the measured rise times - corrupted by noise - and the adjustment made in N_2.

Recall N_2 affects the speed of response of all output channels. Thus the rise time of the ith output channel, $t_{r,i}$, is affected by the setting of N_2. The guideline employed adjusts N_2 such that the measured rise time of every output channel (i.e. $t_{rm,i}$ for $i = 1,2...n$) meets or exceeds its specification. The implicit assumption that no output channel violates its overshoot constraint has been made, and can be supported. In the chemical process industry very few applications require servo control specifications displaying any amount of overshoot. Thus by providing an overshoot specification for each channel, overshoot$_i$, ample leeway is provided in meeting rise time criteria. This, of course, assumes that all $t_{r,i}$ specifications are realistic and attainable such that there is sufficient "room" in adjusting N_2 (recall that the upper limit of N_2 is 40 and its lower limit is given by equations 3.9 and 3.10).

Tuning of N_2 is performed as follows:

Let	$t_{r,i}$	= rise time specification of ith output channel
	$t_{rm,i}$	= measured rise time of ith output channel
	Δt_r	= minimum of $(t_{r,i} - t_{rm,i})$ $i = 1,2...n$
	$N_2(t)$	= new setting of N_2
	$N_2(t-1)$	= current setting of N_2
	$N_{2,min}$	= smallest setting of N_2 consistent with equations 3.9 and 3.10 .

where Δt_r is correlated with an adjustment ΔN_2 yielding:

$$N_2(t) = N_2(t-1) + \Delta N_2 \tag{4.10}$$

$$\text{with } N_{2,min} \leq N_2 \leq 40$$

The correlation among Δt_r and ΔN_2 was found through simulation studies conducted on the benchmark problems (see chapter 5 for the benchmark problems). This correlation is given by table 4.5 and represents a trade-off between speed of convergence and sensitivity (i.e. making ΔN_2 too big causes N_2 to "overshoot"). Also, table 4.5 assumes a sampling time of $T_s = 2$ seconds (for $T_s \neq 2$ seconds, ΔN_2 may be scaled accordingly).

Δt_r	ΔN_2	
	$\Delta t_r < 0$	$\Delta t_r > 0$
$\|\Delta t_r\| < 1.0$	0	0
$1.0 \leq \|\Delta t_r\| < 2.0$	-1	1
$2.0 \leq \|\Delta t_r\| < 4.0$	-2	2
$\|\Delta t_r\| \geq 4.0$	-5	5

Table 4.5 Correlation among Δt_r and ΔN_2

The above scheme is implemented with a combination of generic forward and backward chaining rules. The following generic whenever rule starts the tuning process:

```
for any MIMO_OUTPUT OUTP
for any MIMO_PLANT connected to OUTP
whenever the observed_tr of OUTP receives a value and
when ( for every MIMO_OUTPUT OUT connected to the MIMO_PLANT
        (the observed_tr of OUT has a current value))
then
invoke n2-tuning-rules for the MIMO_PLANT
```

The above rule fires when a process output channel receives a new value for its measured rise time and all output channels have current values of measured rise times. As the validity interval of each measured rise time (i.e. validity interval of observed_tr) is set to $300*T_s$ data obtained earlier is deemed no-longer-applicable as slow changing plant dynamics make it obsolete. This implies that N_2 tuning occurs only when the operator performs independent setpoint changes for each channel (i.e. one setpoint change per process output channel to avoid interaction due to coupling) within a $300*T_s$ time frame.

Several rules with the focal class attribute MIMO_PLANT and the categories attribute n2-tuning-rules are used. If $NU \neq 1$ only one of these rules fires. This particular rule alerts the operator with a message stating that tuning of N_2 is not possible. The rule is:

```
for any MIMO_PLANT PLANT
if (the NU of PLANT /= 1)
then
inform the operator for the next 15 seconds that"
Servo Supervisor:
The rise time of [the name of PLANT] can
NOT be tuned as its NU ≠ 1 rather it
is currently set to [the NU of PLANT]"

Focal Classes        MIMO_PLANT
Categories           n2-tuning
```

In the event that NU = 1 two generic rules are invoked by the inference engine. One of these rules fires if the lower limit of N_2 is given by equation 3.9; and the remaining rule fires if the lower limit of N_2 is given by equation 3.10 . The former of these rules is:

```
for any MIMO_PLANT PLANT
if the NU of PLANT = 1 and
the N1 of PLANT < the max_delay of PLANT + 1 and
delta_N2 /= 0
then in order
set the N2 of PLANT to (
        min( 40, max(the N2 of PLANT + delta_N2, the max_delay of PLANT +
        1 )))
and
inform the operator for the next 15 seconds that"
Servo Supervisor:
Updating the N2 of [the name of PLANT] and
note that N1 < max_delay + 1"
and
conclude that delta_tr has no current value and
conclude that delta_N2 has no current value

Focal Classes        MIMO_PLANT
Categories           n2-tuning
```

When the inference engine invokes the above rule it will seek all the required data to evaluate its antecedent. Thus a current value for max_delay is obtained through the GSI interface, and the current value of delta_N2 is obtained through backward chaining. Five generic rules, relating delta_N2 with delta_tr as per table 4.5, will then be invoked. One of these rules is:

```
for any MIMO_PLANT PLANT
if abs( delta_tr ) ≥ 1.0 and abs( delta_tr ) < 2.0
then in order
conclude that delta_N2 = ( if delta_tr ≥ 0.0 then 1 else -1 )
and inform the operator for the next 15 seconds that"
Servo Supervisor:
Rise times for [the name of PLANT]
are slightly OFF SPEC updating
N2 by [the value of delta_N2]"
```

For the antecedent to be evaluated, the inference engine must seek a value for delta_tr, starting one additional level of backward chaining. Both levels of backward chaining are evaluated by the inference engine when it obtains a value for delta_tr from:

```
for any MIMO_PLANT PLANT
unconditionally conclude that
delta_tr = (the minimum over each MIMO_OUTPUT OUTP connected to Plant
of (the desired_tr of OUTP - the observed_tr of OUTP))
```

Thus two levels of backward chaining are used to adjust N_2.

Once the inference engine updates N_2, three generic whenever rules will be invoked. These rules perform upper and lower limit checks on N_2. The upper limit of N_2 is 40, and the lower limit is given by equations 3.9 or 3.10 . If a limit was reached twice during the last two tuning periods a message is issued to the operator indicating that the rise time specifications may not be met. The implicit assumption is that in order to meet the closed loop rise time specifications the value of N_2 must fall between its upper and lower bounds.

In summary, the rise time tuning scheme is based upon a heuristic correlation between the adjustment in N_2 and the error between the measured and specified rise times among all output channels. This method is implemented using generic forward and backward chaining rules. The rule attributes focal class, and categories are also used. In addition, generic forward chaining rules alert the process operator should N_2 be set to its upper or lower limit during two consecutive tuning periods indicating that the closed-loop performance specification may not be met.

4.4 Summary

This chapter presented the Adaptive Long range predictive control Performance Supervisor, ALPS. ALPS is implemented with the real-time expert system development tool G2. The focus of ALPS is to provide closed-loop performance supervision and tuning of several LRPC algorithms. ALPS adjusts LRPC parameters based on both closed-loop regulatory and servo control performance. ALPS uses Gensym's GSI interface to link to several constrained MGPC algorithms. Both the MGPC algorithms and ALPS are generic, and hence provide closed-loop control and supervision for processes of any dimension $n \times m$. Additional features for the purpose of training personnel in the use of ALPS and the MGPC algorithms is also provided. For demonstrative purposes ALPS has been linked to three processes each of dimension 2×2.

The regulatory tuning strategies used by ALPS periodically update the output weighting matrix Γ and the disturbance rejection filter $T(q^{-1})$. In addition, a regulatory band, reflecting the maximum allowable drift in process outputs, is implemented.

Servo response tuning strategies consist of updating the maximum output prediction horizon N_2 (provided $NU = 1$), and updating output constraints in accordance to overshoot specifications. Adjustment of N_2 occurs only if all process outputs have recent values of measured rise times. ALPS measures the closed-loop rise time of a process output channel if all remaining process outputs channels remain at steady state throughout the setpoint change. This method eliminates erroneous measurements of t_r by eliminating interactions due to coupling. This scheme is useful provided the dimension of the MIMO processes is small (say $n \leq 4$).

The next chapter introduces three 2×2 benchmark processes used to evaluate the tuning strategies employed by ALPS. Both closed-loop servo and regulatory control tuning strategies are evaluated with and without disturbances and MPM.

Chapter 5

Evaluation of ALPS

Chapter 4 introduced the Adaptive Long range predictive control Performance Supervisor, ALPS. ALPS provides closed-loop performance supervision of several LRPC algorithms. ALPS adjusts LRPC tuning parameters so as to achieve and maintain user-specified closed-loop regulatory and servo control performance in the presence of MPM and disturbances. This chapter demonstrates and evaluates the tuning strategies employed by ALPS.

Three 2×2 benchmark problems are used to evaluate the tuning strategies. Both closed-loop servo control tuning strategies and regulatory control tuning strategies are investigated with and without disturbances and MPM.

Section 5.1 presents the three 2×2 benchmark problems used to evaluate ALPS' tuning strategies. Section 5.2 presents the results of simulation studies for each benchmark problem. Section 5.3 summarizes this chapter and recaps the servo and regulatory control tuning strategies.

5.1 Selection of benchmark problems

Three 2 × 2 benchmark problems, along with their performance specifications, are presented in this section. For convenience, the processes are labelled process A, process B, and process C.

5.1.1 Process A

Process A consists of the non-minimum phase system with dominant off-diagonal elements and a non-diagonal delay matrix given by Shah *et al.* (1987):

$$(A^{-1}B)_{model} = \begin{bmatrix} \dfrac{0.4}{1-0.6z^{-1}} & \dfrac{z^{-2}}{1-0.6z^{-1}} \\ \dfrac{z^{-1}}{1-0.8z^{-1}} & \dfrac{0.2z^{-2}}{1-0.8z^{-1}} \end{bmatrix} \tag{5.1}$$

For the purposes of model-plant mismatch two additional equations are considered. Investigations involving servo response supervision use:

$$(A^{-1}B)_{plant} = \begin{bmatrix} \dfrac{0.44}{1-0.62z^{-1}} & \dfrac{0.92z^{-2}}{1-0.58z^{-1}} \\ \dfrac{0.92z^{-1}}{1-0.78z^{-1}} & \dfrac{0.22z^{-2}}{1-0.82z^{-1}} \end{bmatrix} \tag{5.2}$$

Where the unit delay due to a zero order hold has been extracted from each individual transfer function (recall the definition of the ARIMAX model in section 2.3.1). The amount of model-plant mismatch among equations 5.1 and 5.2 consist of 3% displacements in the pole location for every pole, and a minimum and maximum steady state gain mismatch of 12% and 22% respectively. Investigations involving regulatory supervision use:

$$(A^{-1}B)_{plant} = \begin{bmatrix} \dfrac{0.48}{1-0.66z^{-1}} & \dfrac{0.8z^{-2}}{1-0.54z^{-1}} \\ \dfrac{0.8z^{-1}}{1-0.72z^{-1}} & \dfrac{0.24z^{-2}}{1-0.88z^{-1}} \end{bmatrix} \tag{5.3}$$

Thus the amount of model-plant mismatch among equations 5.1 and 5.3 consists of 10% displacements in the pole location for every pole, and a minimum and maximum steady state gain mismatch of 30% and 100% respectively.

The closed-loop performance specifications and input rate and amplitude constraints for the servo supervision investigations are listed in table 5.1. Table 5.2 lists the closed-loop performance specifications and input rate and amplitude constraints for the regulatory supervision investigations. Unless stated otherwise, "tight" input rate constraints imply settings for Δu_{min} and Δu_{max} of 1/10 the values listed in tables 5.1 and 5.2. For illustrative purposes, a sampling interval of $T_s = 2$ seconds is used.

Quantity	Value
$\Delta u_{min\ 1,2}$	-1.0
$\Delta u_{max\ 1,2}$	1.0
$u_{min\ 1,2}$	-4.0
$u_{max\ 1,2}$	4.0
$overshoot_{1,2}$	30%
$r_{b\ 1,2}$	20%
$t_{r\ 1,2}$	13.0 sec
$\sigma^2_{u\ 1,2}$	10^{-4}
σ_{ymax}	$5.\times10^{-3}$

Table 5.1 Process A, performance specifications (servo investigations)

Quantity	Value
$\Delta u_{min\ 1,2}$	-1.0
$\Delta u_{max\ 1,2}$	1.0
$u_{min\ 1,2}$	-4.0
$u_{max\ 1,2}$	4.0
$overshoot_{1,2}$	30%
$r_{b\ 1,2}$	20%
$t_{r\ 1,2}$	13.0 sec
$\sigma^2_{u\ 1,2}$	10^{-2}
σ_{ymax}	0.1

Table 5.2 Process A, performance specifications (regulatory investigations)

5.1.2 Process B

Process B is the linear model of a binary distillation column as given by Wood and Berry (1973) (see also Berry 1973). This particular process consists of an eight tray distillation column which separates a binary mixture of methanol and water. The process model is given by:

$$\begin{bmatrix} y_1(s) \\ y_2(s) \end{bmatrix} = \begin{bmatrix} \dfrac{12.8\, e^{-s}}{16.7s+1} & \dfrac{-18.9\, e^{-3s}}{21s+1} \\ \dfrac{6.6\, e^{-7s}}{10.9s+1} & \dfrac{-19.4\, e^{-3s}}{14.4s+1} \end{bmatrix}_{model} \begin{bmatrix} u_1(s) \\ u_2(s) \end{bmatrix} \tag{5.4}$$

and for the purposes of model-plant mismatch consider:

$$\begin{bmatrix} y_1(s) \\ y_2(s) \end{bmatrix} = \begin{bmatrix} \dfrac{10.1\, e^{-s}}{15s+1} & \dfrac{-24.5\, e^{-3s}}{23s+1} \\ \dfrac{5\, e^{-7s}}{9s+1} & \dfrac{-17.1\, e^{-3s}}{13s+1} \end{bmatrix}_{plant} \begin{bmatrix} u_1(s) \\ u_2(s) \end{bmatrix} \tag{5.5}$$

where

$y_1(s)$	is the deviation of the distillate concentration in units of %weight from its nominal operating condition of 96%, and
$y_2(s)$	is the deviation of the bottoms concentration in units of %weight from its nominal operating point of 0.5%, and
$u_1(s)$	is the deviation of the reflux flow from its nominal operating condition of 1.95 lbs/min, and
$u_2(s)$	is the deviation of the reboiler steam flow from its nominal operating condition of 1.71 lbs/min

The units of equations 5.4 and 5.5 are (%weight)/(lbs/min) for the process gains, and the time constants and delays are given in minutes.

The amount of model-plant mismatch among equations 5.4 and 5.5 consist of 9% to 17% in the displacement of the pole locations, and a minimum and maximum steady state gain mismatch of 12% and 30% respectively.

The closed-loop performance specifications and input rate and amplitude constraints are listed in table 5.3. Note that for the purpose of closed-loop simulations the input amplitude constraints have been rounded up to the nearest integers. Unless stated otherwise, "tight" input rate constraints imply settings for Δu_{min} and Δu_{max} of 1/10 the values listed in table 5.3. A sampling interval of T_s = 1 minute is used.

Quantity	Value
$\Delta u_{min\ 1,2}$	-1.0
$\Delta u_{max\ 1,2}$	1.0
$u_{min\ 1,2}$	-2.0
$u_{max\ 1,2}$	2.0
$\text{overshoot}_{1,2}$	40%
$r_{b\ 1,2}$	30%
$t_{r\ 1,2}$	7.5 min
$\sigma^2_{u\ 1,2}$	10^{-2}
σ_{ymax}	$5.\times10^{-3}$

Table 5.3 Process B, performance specifications

5.1.3 Process C

Process C consists of the Shell heavy oil fractionator documented in the Shell Process Control Workshop (see Prett and Morari 1986). The original Shell control problem describes a 7-output, 5-input process, however, for evaluation purposes the following 2 × 2 subset is selected:

$$\begin{bmatrix} y_1(s) \\ y_2(s) \end{bmatrix} = \begin{bmatrix} \dfrac{4.05\, e^{-27s}}{50s+1} & \dfrac{5.88\, e^{-27s}}{50s+1} \\ \dfrac{4.38\, e^{-20s}}{33s+1} & \dfrac{7.20}{19s+1} \end{bmatrix}_{model} \begin{bmatrix} u_1(s) \\ u_2(s) \end{bmatrix} \qquad (5.6)$$

and for the purposes of model-plant mismatch consider:

$$\begin{bmatrix} y_1(s) \\ y_2(s) \end{bmatrix} = \begin{bmatrix} \dfrac{4.68\, e^{-27s}}{50s+1} & \dfrac{5.585\, e^{-27s}}{50s+1} \\ \dfrac{5.31\, e^{-20s}}{33s+1} & \dfrac{6.535}{19s+1} \end{bmatrix}_{plant} \begin{bmatrix} u_1(s) \\ u_2(s) \end{bmatrix} \qquad (5.7)$$

where

$y_1(s)$	is the deviation of the distillate concentration or top end point, and
$y_2(s)$	is the deviation of the bottoms reflux temperature, and
$u_1(s)$	is the deviation of the top draw, and
$u_2(s)$	is the deviation of the bottoms reflux heat duty.

The time constants and delays are given in minutes.

The amount of model-plant mismatch among equations 5.6 and 5.7 consists of a minimum and maximum steady state gain mismatch of 5% and 21% respectively. In addition, the Shell control problem has the following control objectives and constraints:

Control objectives

1) The original problem (Prett and Morari 1986) specifies regulatory control of the top draw product end point (i.e. $y_1(s) = 0.0 \pm 0.005$, and $-0.5 \leq y_2(s) \leq \infty$ at steady state). But to illustrate the use of output constraints more clearly this is increased to 0.0 ± 0.05 for both $y_1(s)$ and $y_2(s)$.

2) Keep the closed-loop speed of response between 0.8 and 1.25 of the open-loop process bandwidth.

Constraints

1) All draws must be within hard maximum and minimum bounds of 0.5 and -0.5.

2) The bottom reflux heat duty is constrained within the hard bounds of 0.5 and -0.5. In addition to this original specification, the top draw is also constrained to within the hard bounds of 0.5 and -0.5.

3) All manipulated variables have maximum move size limitations of magnitude 0.05 per minute.

4) Fastest sampling time is 1 minute.

5) The bottom reflux draw temperature has a minimum value of -0.5.

6) The top end point must be maintained within the maximum and minimum values of 0.5 and -0.5.

The closed-loop performance specifications and input rate and amplitude constraints, as derived from the above control objectives and control constraints, are listed in table 5.4. A sampling interval of T_s = 8 minutes is used to reduce the number of discrete delay sample periods.

Quantity	Value
$\Delta u_{min\ 1,2}$	-0.4
$\Delta u_{max\ 1,2}$	0.4
$u_{min\ 1,2}$	-0.5
$u_{max\ 1,2}$	0.5
$\text{overshoot}_{1,2}$	20%
$r_{b\ 1,2}$	5%
$t_{r\ 1}$	110 min
$t_{r\ 2}$	35 min
$\bar{\sigma}^2_{u\ 1,2}$	2.5×10^{-4}
σ_{ymax}	$5.\times10^{-3}$

Table 5.4 Process C, performance specifications

5.2 Evaluation of performance tuning strategies

This section presents the results of simulation studies conducted on the 2 × 2 benchmark problems presented in section 5.1. In order to highlight each tuning strategy and isolate its effect on the closed-loop control performance, the results are presented in terms of the following two categories:

1) regulatory control tuning strategies, and
2) servo control tuning strategies.

Recall that ALPS performs closed-loop servo control performance tuning whenever the process operator initiates setpoint changes. On the other hand, ALPS performs closed-loop regulatory performance tuning continually.

The simulation results for each benchmark problem are discussed in the following subsections.

5.2.1 Process A

The simulation results obtained with process A are presented in this subsection. Tests revealing the execution speed of the MGPC algorithms were also conducted. For all 2 × 2 problems, the MGPC execution speed ranges between 100 to 200 ms per iteration (includes process simulation and QPSOL).

5.2.1.1 Regulatory Performance Tuning

ALPS' regulatory performance tuning strategies adjust both $T(q^{-1})$ and Γ. The interaction of tuning $T(q^{-1})$ and Γ simultaneously is avoided by investigating each strategy separately. This is achieved by keeping one tuning parameter fixed at its default value while tuning the remaining parameter. In addition, to clearly illustrate the procedure of switching from a first order $T(q^{-1})$ filter to a second order default $T(q^{-1})$ filter, a rather large Gaussian noise variance is employed. To avoid the interaction of this large Gaussian noise with output amplitude constraints, the use of r_b is omitted throughout this section.

Consider figure 5.1 which illustrates the $T(q^{-1})$ filter tuning strategy while maintaining $\Gamma = \mathbf{I}$, under conditions of Gaussian noise and no MPM. Regulatory supervision is enabled at iteration 40. Due to the large Gaussian noise variance, initially set to 0.15, ALPS responds by increasing the amount of filtering from the default first order $T(q^{-1})$ filter, having $T\_a_1 = -0.8$, to its new maximum possible setting of $T\_a_1 = -0.95$ at iteration 40. Notice also the large drift in output channel 2 caused by this large jump in the $T(q^{-1})$ filter parameter T_a_1 (recall section 3.4.2). As the regulatory performance specifications cannot be met with this first order $T(q^{-1})$ filter, ALPS switches to the default second order $T(q^{-1})$ filter at iteration 100. This second order $T(q^{-1})$ filter is fine tuned to meet the closed-loop regulatory performance specifications. The closed-loop performance specifications are met subsequent to 2 or 3 tuning periods as is indicated by the small adjustments made at iterations 160 and 190. At iteration 200 the Gaussian noise variance is reduced to 0.002. ALPS responds by decreasing the amount of filtering and ultimately switches to the default first order $T(q^{-1})$ filter at iteration 460. ALPS fine tunes this first order $T(q^{-1})$ filter until the regulatory performance specifications are met at iteration 550.

Figure 5.2 presents the $T(q^{-1})$ filter tuning strategy while maintaining $\Gamma = \mathbf{I}$, under conditions of Gaussian noise and MPM. At iteration 40 the regulatory supervision mode of ALPS is activated. The large Gaussian noise variance of 0.15 causes ALPS to increase the amount of filtering, and at iteration 90 ALPS switches to the second order default $T(q^{-1})$ filter. ALPS fine tunes the second order $T(q^{-1})$ filter until the closed-loop regulatory performance specifications are satisfied. At iteration 180 the Gaussian noise variance is reduced to 0.002. ALPS responds to this new condition by reducing the amount of filtering, and at iteration 400 switches to the default first order $T(q^{-1})$ filter. Again, ALPS fine tunes the first order $T(q^{-1})$ filter until the closed-loop regulatory performance specifications are met.

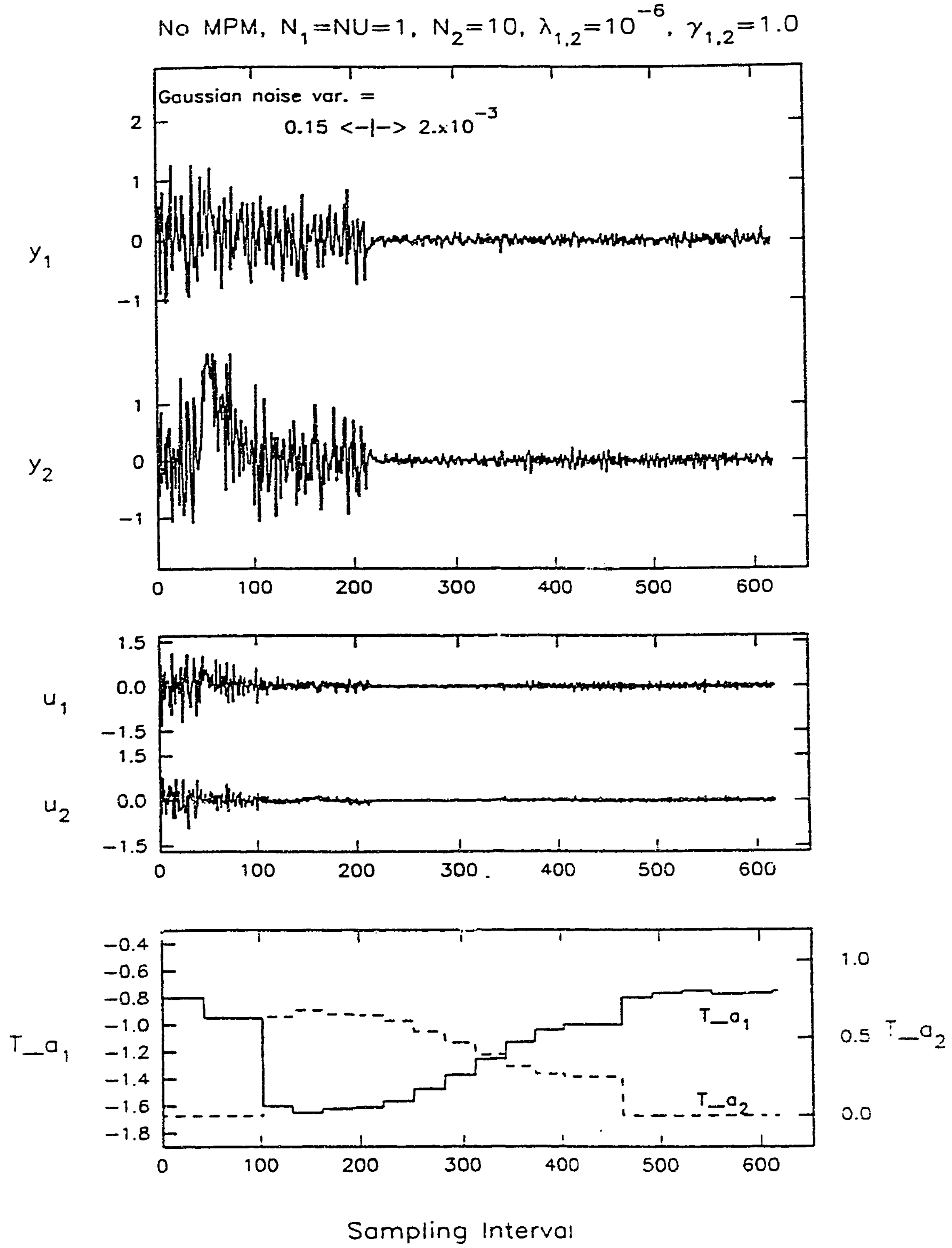


Figure 5.1 Process A, T(q^{-1}) filter tuning, no MPM

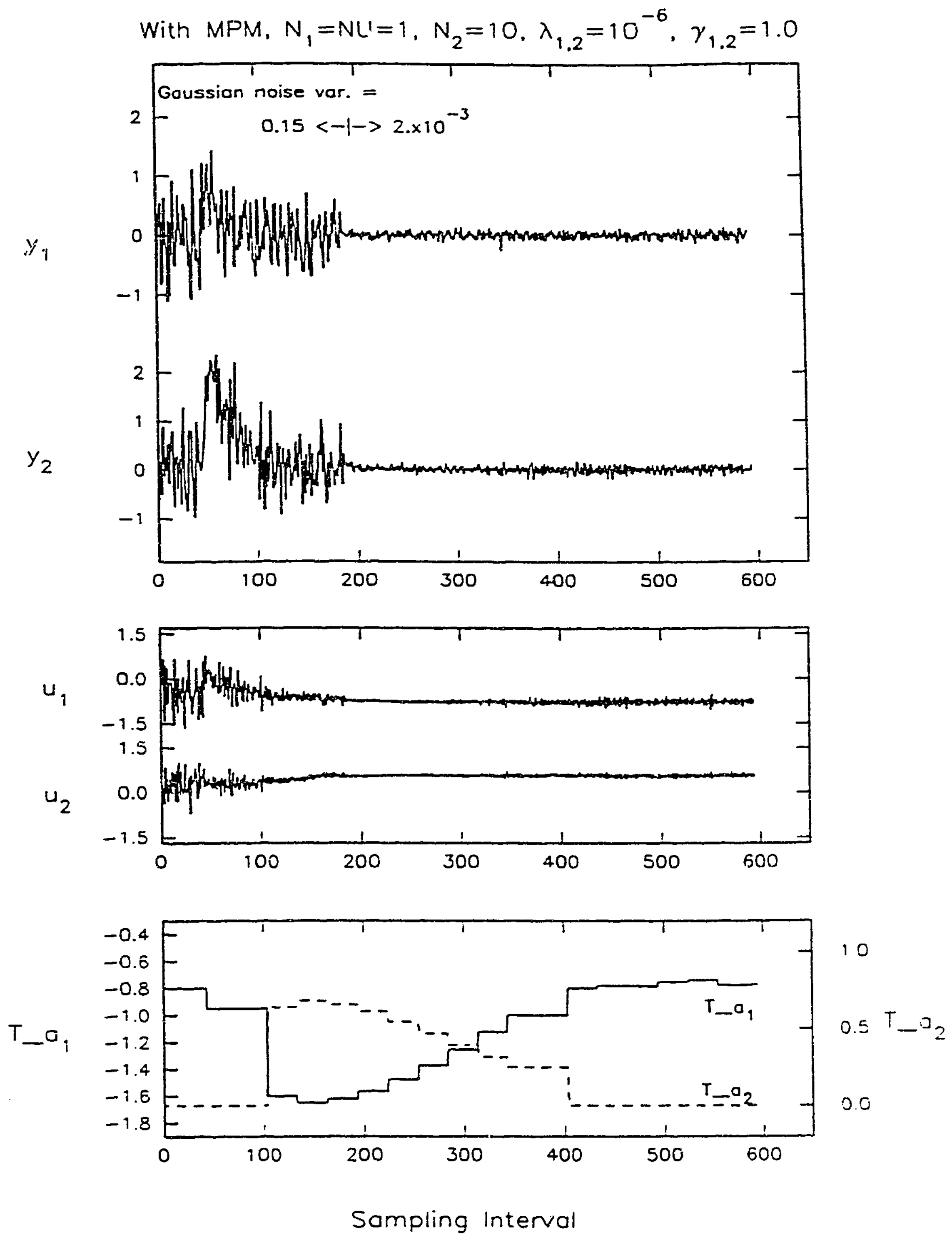


Figure 5.2 Process A, $T(q^{-1})$ filter tuning, with MPM

Figure 5.3 shows the Γ tuning strategy while maintaining a first order default $T(q^{-1})$ filter under conditions of Gaussian noise and no MPM. Consider iterations 0 to 330 where a large Gaussian noise variance is used (a Gaussian noise variance of 0.15 causes $\sigma_{y1,2} >> \sigma_{ymax}$). This portion of the figure indicates that at iteration 80 output channel 2 is weighted heavier by reducing γ_1. The effect of these settings is shown by the reduced variance in y_2 between iterations 110 and 200. At iteration 200 ALPS sets γ_1 to 0.954, which allows the standard deviation of output channel 2 to grow. At iteration 260 ALPS weights output channel 2 heavier again by reducing γ_1. At iteration 340 the Gaussian noise variance is reduced to 0.002 (a Gaussian noise variance of 0.002 causes $\sigma_{y1,2} \approx \sigma_{ymax}$). Throughout iterations 340 to 490 ALPS consistently weights output channel 2 heavier. At iteration 490 the Gaussian noise variance is reduced to a mere 10^{-4}. ALPS responds to this new condition by weighting each output channel equally (i.e. setting $\Gamma = \mathbf{I}$ as $\sigma_{y1,2} << \sigma_{ymax}$).

Figure 5.4 depicts the Γ tuning strategy while maintaining a first order default $T(q^{-1})$ filter under conditions of Gaussian noise and MPM. Consider the period where a large Gaussian noise variance is used (i.e. iterations 0 to 350). During this period ALPS is undecided as to which output channel should receive the heavier output weighting (specifically see iterations 80 to 250). Iterations 350 to 490 use a Gaussian noise variance of 0.002. During this period ALPS again is undecided as to which output channel should consistently receive the heaviest output weighting; albeit switching at a more gradual pace. At iteration 490 the Gaussian noise variance is reduced to 10^{-4} and ALPS responds by setting $\Gamma = \mathbf{I}$.

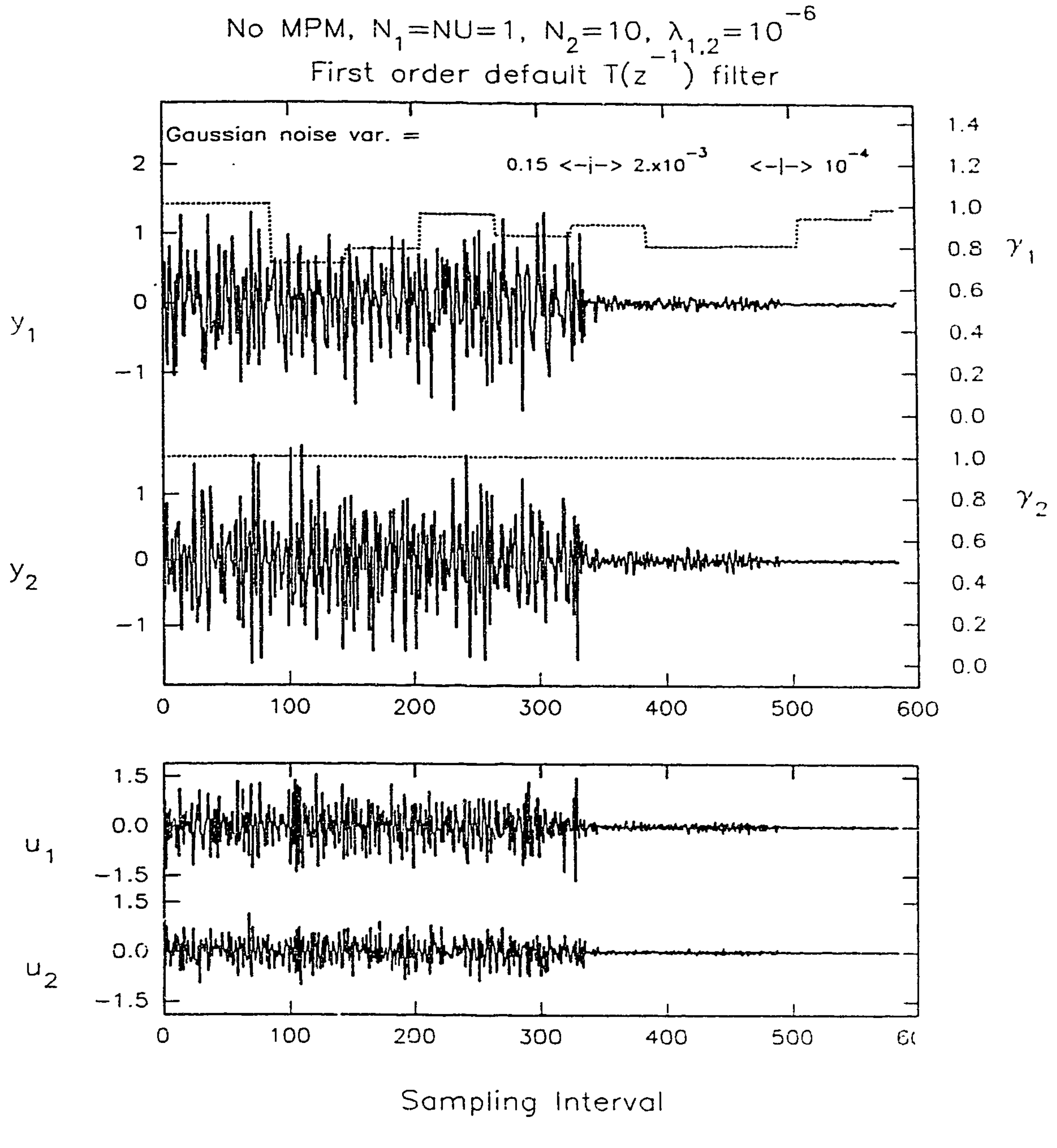


Figure 5.3 Process A, Γ tuning, no MPM

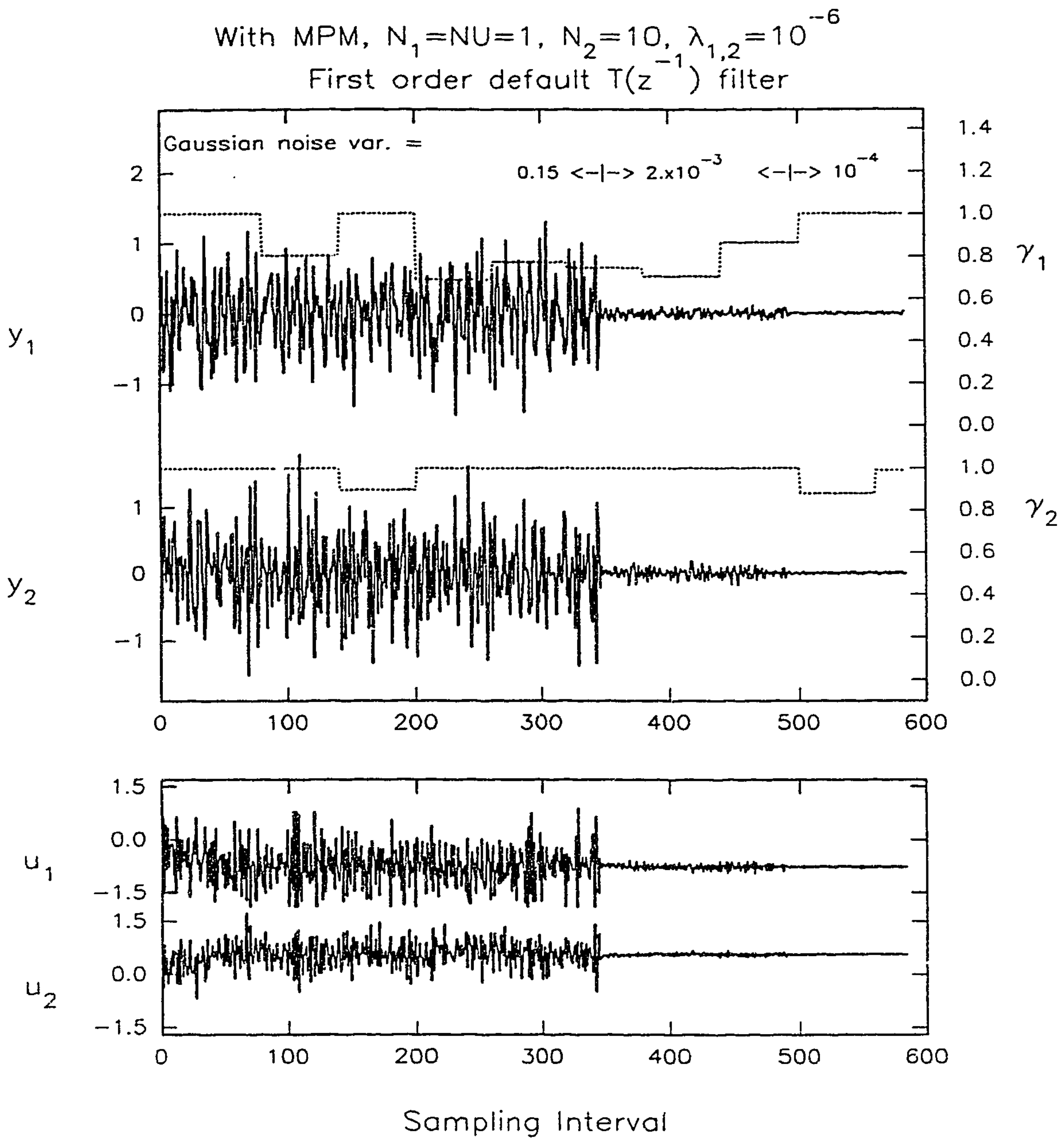


Figure 5.4 Process A, Γ tuning, with MPM

Figures 5.1 and 5.2 show the $T(q^{-1})$ filter tuning strategy with $\Gamma = \mathbf{I}$ under conditions of no MPM and MPM, respectively. On the other hand, figures 5.3 and 5.4 illustrate the Γ tuning strategy using a first order default $T(q^{-1})$ filter, again under conditions of no MPM and MPM. The "complete" regulatory tuning strategy employed by ALPS, however, tunes both Γ and $T(q^{-1})$. This is shown in figure 5.5 for the case of Gaussian noise and no MPM. Initially the Gaussian noise variance is set to 0.15 and regulatory supervision is enabled at iteration 30. ALPS increases the amount of filtering and ultimately switches to a second order $T(q^{-1})$ filter at iteration 110. This second order $T(q^{-1})$ filter is tuned until the performance specifications are met (tuning is completed in one to two successive $T(q^{-1})$ filter tuning periods). Similarly, ALPS adjusts the output weighting Γ albeit in an undecided manner (i.e. during successive Γ tuning periods the output weighting of each output channel is set to unity). At iteration 190 the amount of Gaussian noise is reduced to a variance of 0.002. ALPS responds to this new condition by reducing the amount of filtering and switches to a default first order $T(q^{-1})$ filter at iteration 430. This first order $T(q^{-1})$ filter is tuned until the performance specifications are met (one to two tuning periods are required after the $T(q^{-1})$ filter order switch occurred). ALPS also adjusts Γ during this time frame. Throughout iterations 190 to 590 ALPS adjusts Γ in a gradual manner weighting output channel 2 the heaviest. The latter portion of this figure (i.e. iterations 400 to 600), however, shows that ALPS ultimately resets $\Gamma = \mathbf{I}$. Although not shown in this figure, iterations beyond 600 show that ALPS fixes $\gamma_2 = 1.0$ while gradually cycling γ_1 between the limits of $0.6 < \gamma_1 \leq 1.0$.

Figure 5.6 illustrates the regulatory tuning strategy of adjusting both Γ and $T(q^{-1})$ under conditions of Gaussian noise and MPM. Note the similarity of this figure to figure 5.5 . Indeed, the entire discussion presented for figure 5.5 is valid for this figure albeit the cyclical manner of Γ is more pronounced. In comparison to figure 5.5, the $T(q^{-1})$ filter parameters satisfying the regulatory performance specifications are tuned to values resulting in slightly heavier filtering.

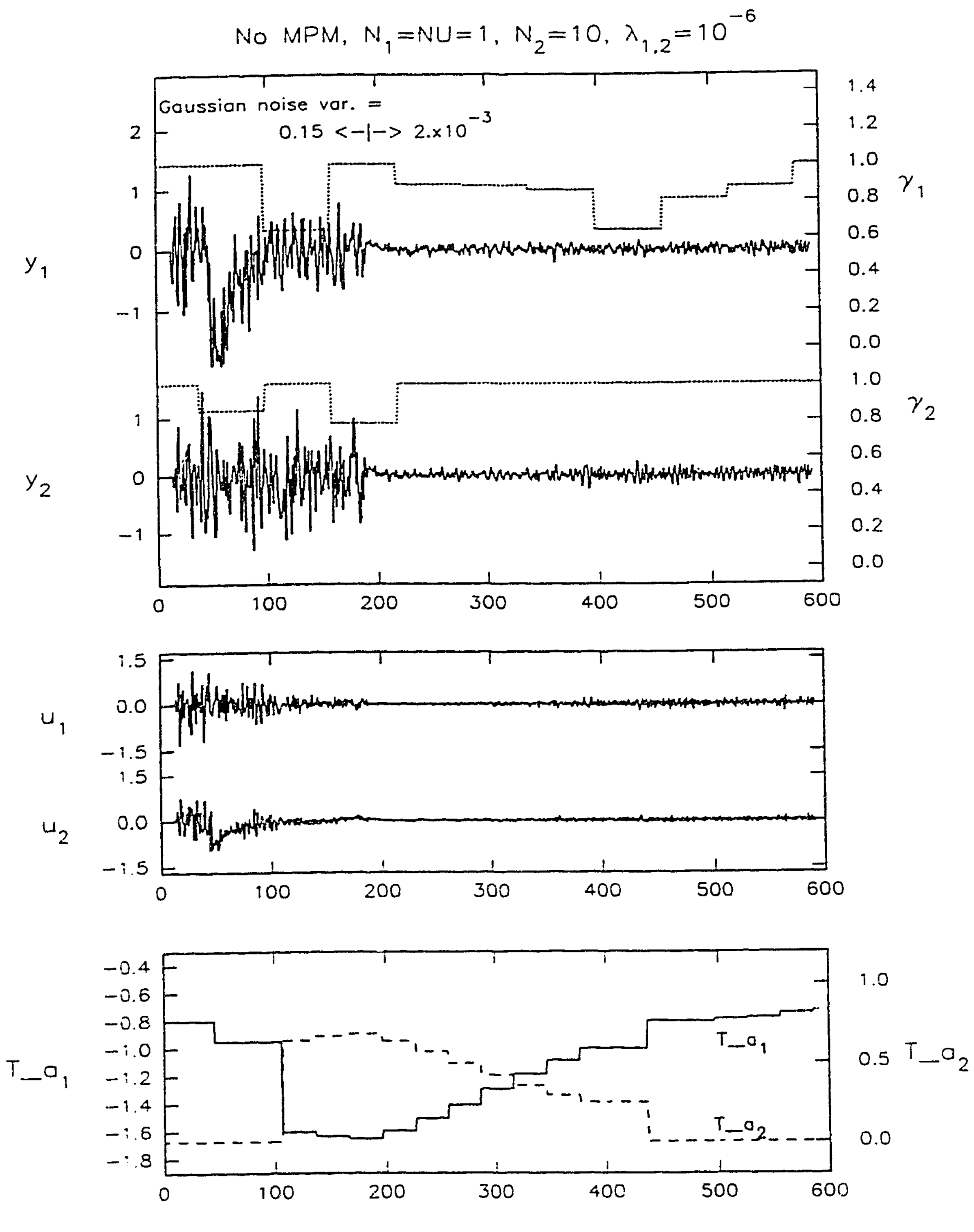


Figure 5.5 Process A, tuning of Γ and $T(q^{-1})$, no MPM

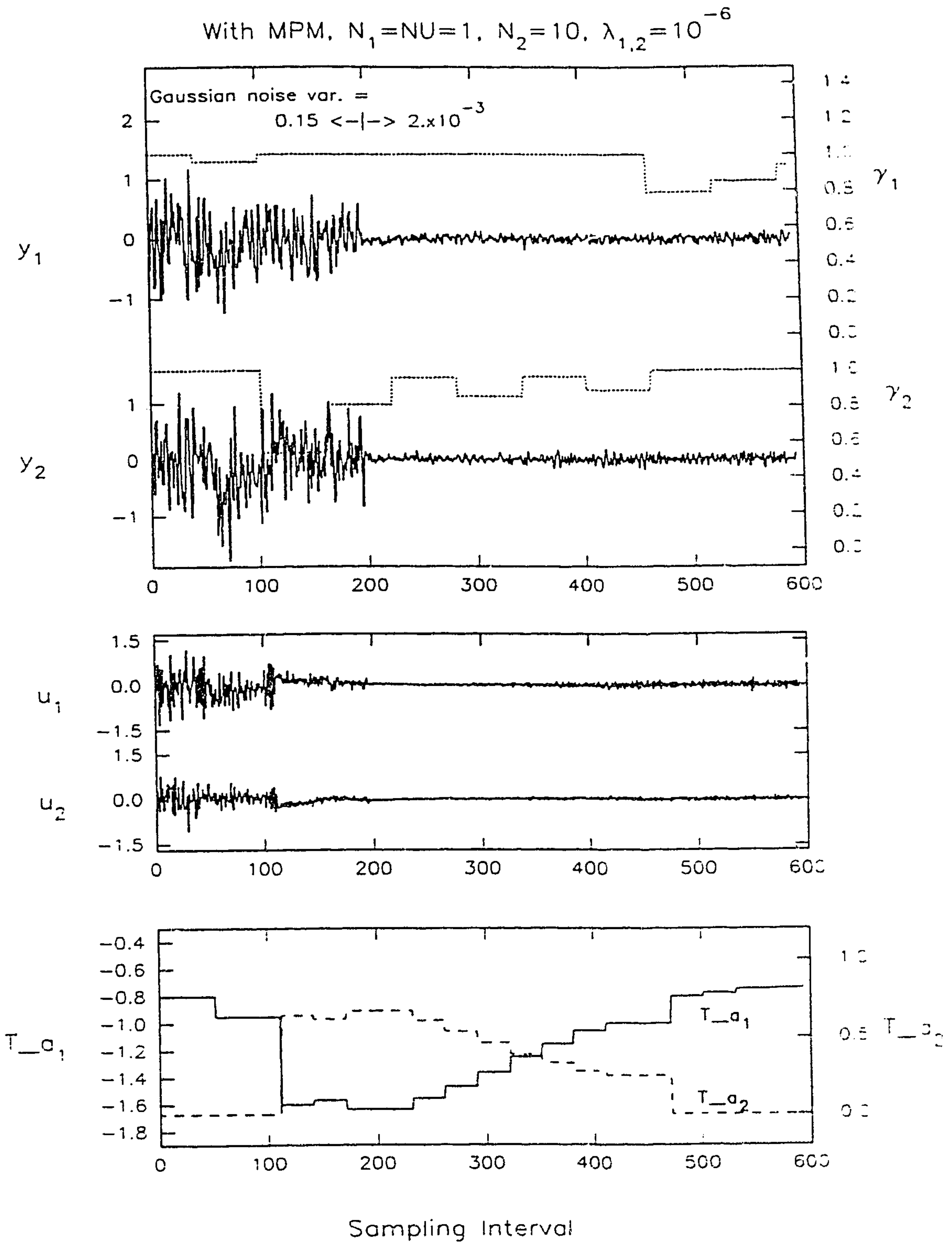


Figure 5.6 Process A, tuning of Γ and $T(q^{-1})$, with MPM

Important conclusions can be drawn from figures 5.1 through 5.6 . These figures show that ALPS successfully adjusts the $T(q^{-1})$ filter order and break frequency to achieve and maintain the closed-loop regulatory performance criteria, σ^2_u, under conditions of Gaussian noise, with and without MPM, with and without adjusting the output weighting Γ. Unfortunately, these figures also reveal a shortcoming in the Γ tuning strategy. Figures 5.3 through 5.6 indicate that adjusting Γ as per equation 4.1 initially does have the desired effect (i.e. balancing the outputs such that $\sigma_{y,1} \approx \sigma_{y,2}$). Subsequent adjustments in Γ, according to equation 4.1 with $\Gamma \neq \mathbf{I}$, do not achieve the desired effect as the current value of Γ is not taken into account.

Regulatory performance supervision consists of tuning the $T(q^{-1})$ filter and Γ. The role of Γ is to balance the LRPC algorithm such that the weighted standard deviation of all output channels are of the same order of magnitude. Adjustments made in Γ, however, also affect the control signal variance of the input channels (recall section 3.4.2). Figures 5.5 and 5.6 clearly demonstrate that any adjustments made in Γ do not affect ALPS' ability to meet the $\sigma^2_{u\ 1,2}$ regulatory performance specifications. In other words, any adverse effects on σ^2_u through tuning Γ are removed by adjusting the $T(q^{-1})$ filter. Hence, the evaluation of ALPS' regulatory performance tuning strategies for the remaining benchmark problems (processes B and C) consist of simultaneous tuning of $T(q^{-1})$ and Γ.

5.2.1.2 Servo Performance Tuning

This section focuses on the servo performance tuning strategies employed by ALPS. Unless stated otherwise, regulatory performance supervision is disabled in order to highlight the servo performance tuning strategy.

Consider figure 5.7 which shows the trajectories of the output prediction horizon N_2, and the output constraints y_{min} and y_{max}, under ideal conditions (i.e. no MPM and no measurement noise). The setpoints of y (i.e. w_1 and w_2) are square-waves of amplitude 0.1 and sufficiently offset in time to enable each output channel to reach steady state during a ¼ cycle. Notice how the output constraints are updated whenever a setpoint change occurs. For a positive setpoint change in output channel 'i', the maximum output constraint $y_{i,max}$ is updated to reflect its overshoot specification. After a fixed duration of 30 sample intervals the output constraints $y_{i,max}$ and $y_{i,min}$ are updated to reflect the regulation band specification $r_{b,i}$. This figure also shows that due to the small setpoint changes (i.e. ± 0.1) the overshoot specifications result in "tighter" bounds during setpoint changes than do the regulation band specifications. Servo supervision is enabled at iteration 0 with the output prediction horizon initially set to $N_2 = 30$. ALPS updates N_2 whenever all output channels have current values for their rise times $t_{r,1}$ and $t_{r,2}$ (recall that $t_{r,i}$ is valid for a particular value of N_2 up to a duration of 10 minutes after detection). The values of $t_{r,1}$ and $t_{r,2}$ are included in figure 5.7 . The final value of N_2, satisfying the rise time specifications of both output channels, is $N_2 = 11$.

Figure 5.8 depicts the trajectories of the output prediction horizon N_2, and output constraints y_{min} and y_{max}, under conditions of MPM. Servo supervision is enabled at iteration 0 with the output prediction horizon initially set to $N_2 = 20$. ALPS continually decreases N_2 to satisfy the servo performance specifications $t_{r,1}$ and $t_{r,2}$. Observe, however, that the rise time calculation scheme, given by equation 3.8, is not without error. Iterations 300 to 400 reveal an erroneous value of $t_{r,1} = 13.687$ seconds (compare this value with the data to its left and right). This value misleads ALPS to assume that both output channels meet their specifications with $N_2 = 9$. ALPS continues tuning in the subsequent cycle and ultimately meets the servo performance specifications with $N_2 = 8$.

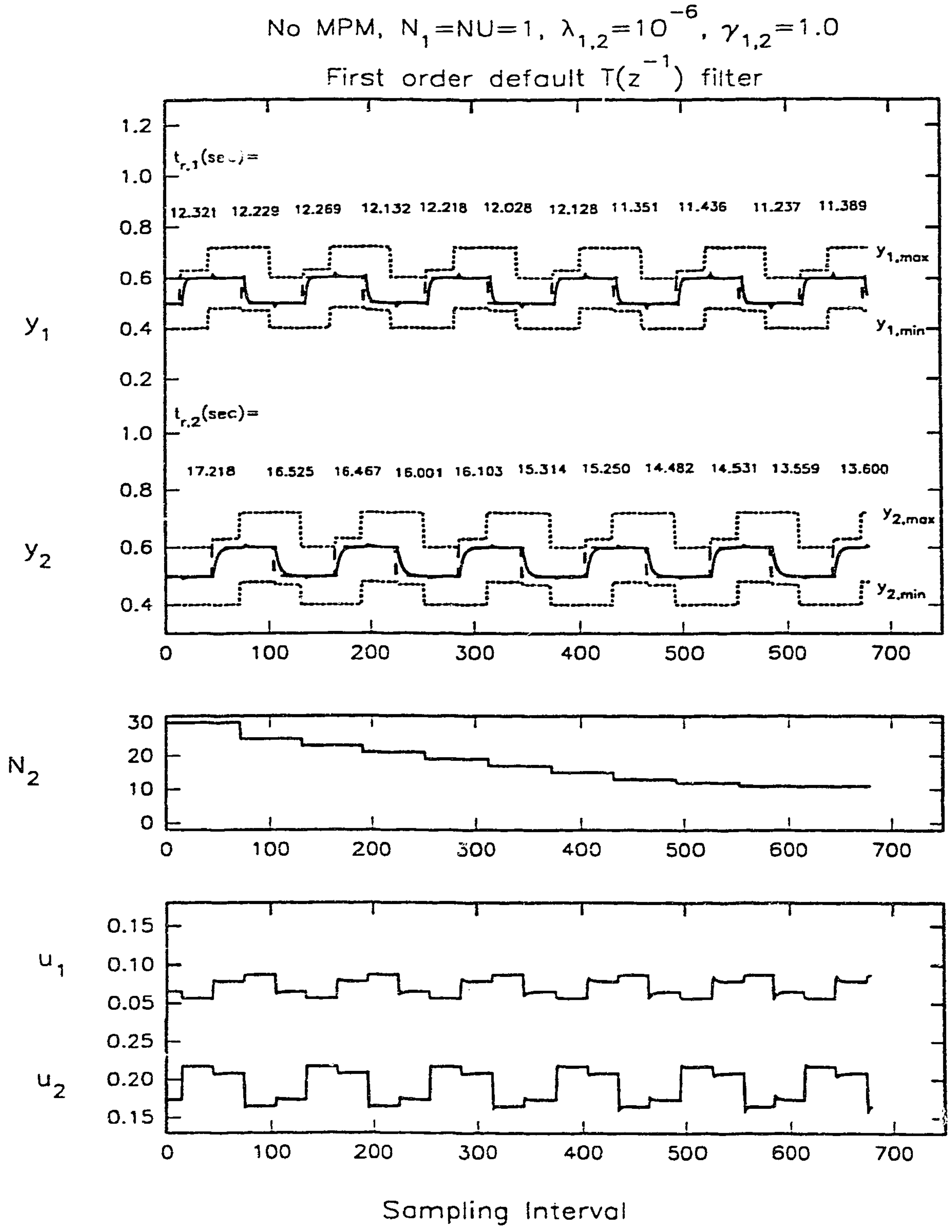


Figure 5.7 Process A, tuning of N_2, no MPM

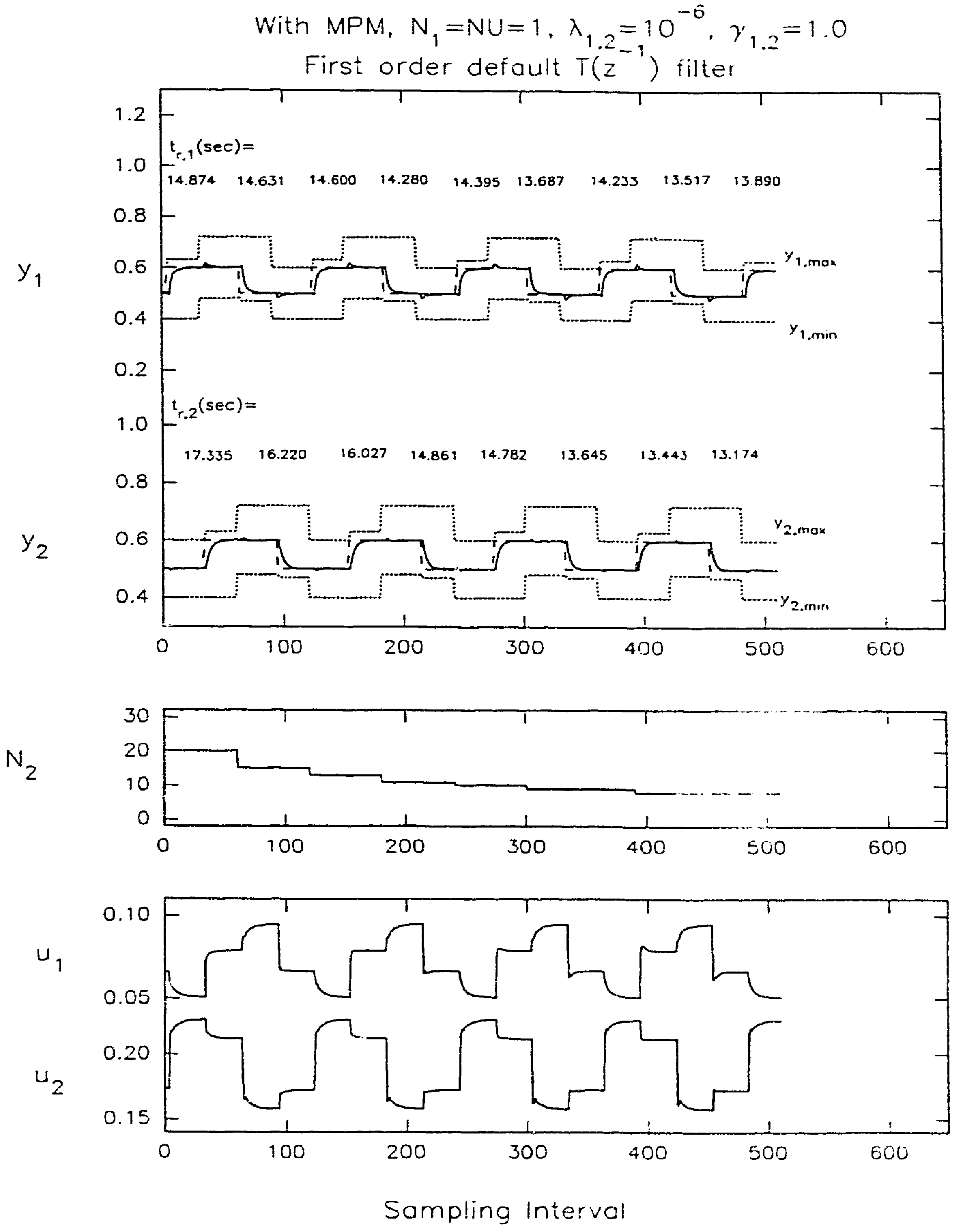


Figure 5.8 Process A, tuning of N_2, with MPM

Consider figure 5.9 which illustrates the servo performance tuning strategy applied to process A under conditions of Gaussian noise and no MPM. Servo supervision is enabled at iteration 0 with the output prediction horizon initially set to $N_2 = 30$. ALPS decreases N_2 in order to meet the closed-loop servo performance specifications. However, the rise time detection scheme, given by equation 3.8, is at times seriously affected by the Gaussian noise. During iterations 150 to 250, for example, $t_{r,2}$ is calculated to be significantly smaller than observed. This causes ALPS to assume that the performance specifications have been met, and hence does not update N_2. This situation is encountered again between iterations 430 to 520. In addition, consider the interval between iterations 500 to 550. During this period both calculated values of $t_{r,1}$ and $t_{r,2}$ are smaller than observed, causing ALPS to increase N_2 by 1. Ultimately ALPS meets the servo performance criteria with a setting of $N_2 = 11$.

Figure 5.10 depicts the servo performance tuning strategy applied to process A under conditions of Gaussian noise and MPM. Servo supervision is enabled at iteration 0 with the maximum output prediction horizon initially set to $N_2 = 30$. ALPS continually decreases N_2 in order to meet the closed-loop servo performance specifications. ALPS tunes the output prediction horizon to its final value of $N_2 = 8$; which satisfies all servo performance specifications. Careful examination of the calculated rise times $t_{r,1}$ and $t_{r,2}$ reveals that, from time to time, these values are affected by the Gaussian noise. ALPS - at least during this run - consistently updates N_2 without any apparent error.

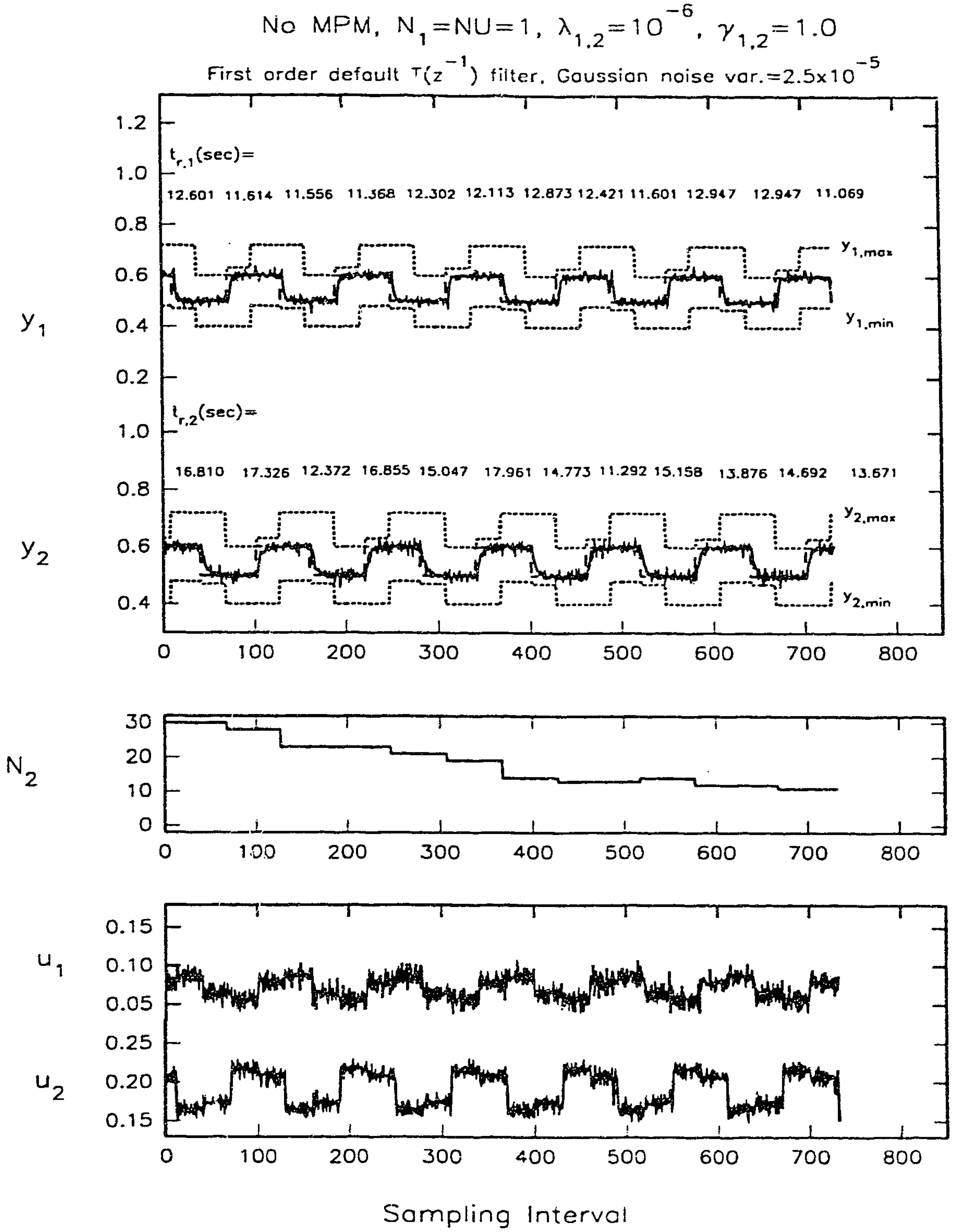


Figure 5.9 Process A, tuning of N_2, no MPM, with Gaussian noise

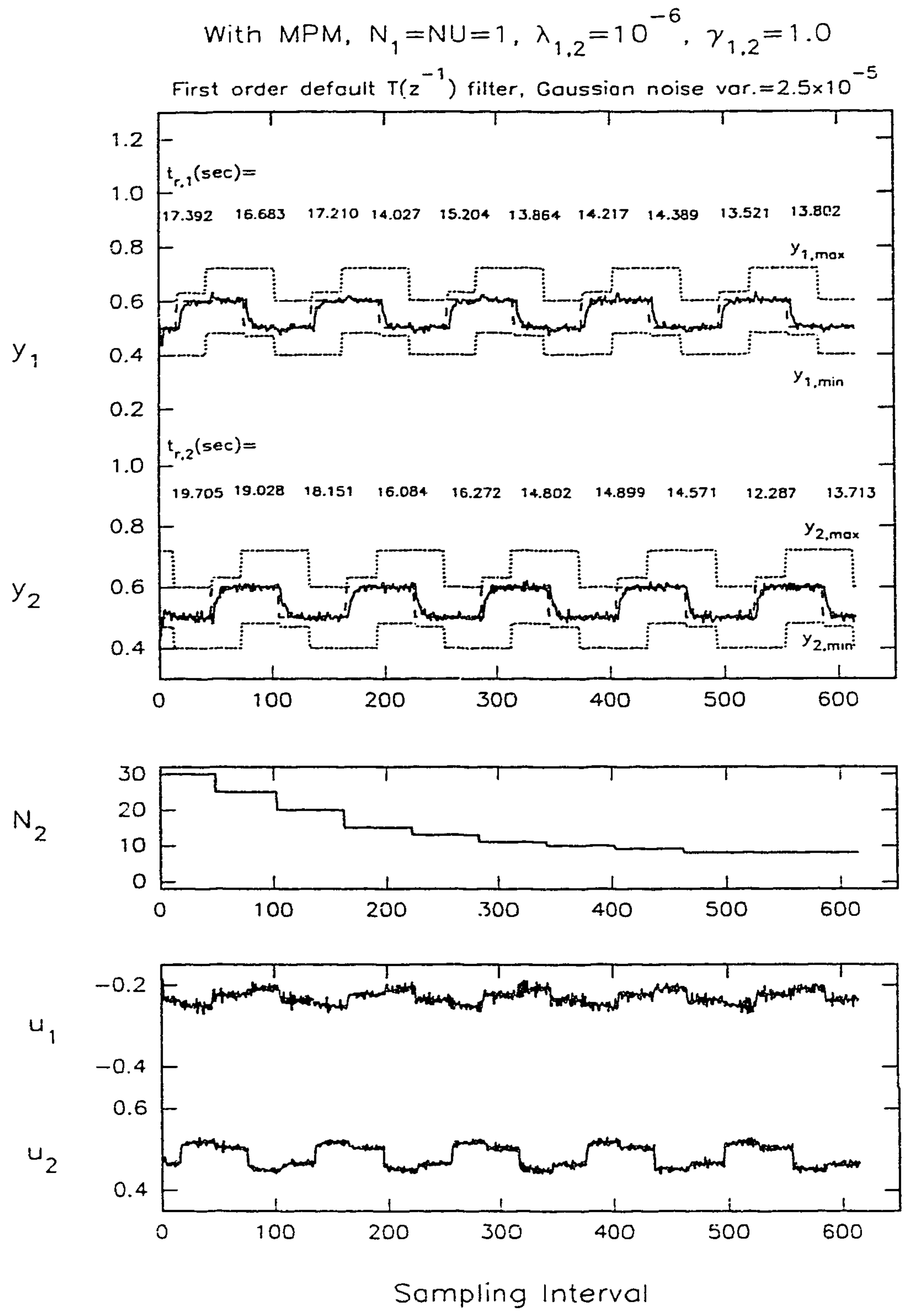


Figure 5.10 Process A, tuning of N_2, with MPM, with Gaussian noise

Important conclusions can be drawn from figures 5.7 through 5.10 . These figures show that ALPS successfully adjusts N_2 to achieve and maintain the closed-loop servo performance criteria, t_r, under conditions of Gaussian noise, with and without MPM. In addition, ALPS updates the output constraints y_{min} and y_{max} to reflect overshoot and regulatory band specifications. The figures also reveal that the rise time calculation (equation 3.8) is affected by Gaussian noise. Although the net effect of Gaussian noise on the final value of N_2 is nil, ALPS will respond to incorrect t_r values and adjust N_2 causing N_2 to slowly "bounce around". A simple analog low-pass pre-filter would almost certainly eliminate this problem.

Section 5.2.1.1 evaluates ALPS' regulatory response supervision schemes whereas section 5.2.1.2 focuses on ALPS' servo response supervision procedures as applied to process A. Typical applications of LRPC algorithms are multivariable continuous processes such as distillation columns. Continuous processes require the LRPC algorithm to operate throughout three distinct modes of control: plant startup, prolonged normal operation, and ultimately plant shutdown for the purpose of maintenance. Figure 5.11 illustrates these control modes. This figure depicts the trajectories of the maximum output prediction horizon N_2, output constraints y_{min} and y_{max}, output weighting Γ, and $T(q^{-1})$ filter, under conditions of MPM and Gaussian noise. The setpoints of y (i.e. w_1 and w_2) are square-waves of amplitude 0.1 with a d.c. component of 0.5. Also, a ½ cycle square-wave is sufficiently long in time to allow ALPS to tune Γ and $T(q^{-1})$. The simulation begins with a default first order $T(q^{-1})$ filter and $N_2 = 15$. Between iterations 0 to 30 setpoint scheduling is used to gradually increase the setpoints to values of 0.5. Notice how y_{max} is updated to reflect the overshoot specifications throughout this interval. Once both output channels reach steady state (i.e. within $0.5*(1 \pm r_b))$ ALPS updates y_{min} and

y_{max} to reflect the r_b specifications. Following iteration 60 both setpoints are updated according to the square-waves discussed above. ALPS adjusts N_2, Γ, and $T(q^{-1})$ in order to meet the closed-loop servo and regulatory performance specifications. ALPS meets the all performance specifications with $N_2 = 8$, and a first order $T(q^{-1})$ filter with $T\_a_1 = -0.78$. Notice that Γ gradually cycles (similar to figures 5.5 and 5.6). ALPS fixes $\gamma_2 = 1.0$ while gradually cycling γ_1 between the limits of $0.65 < \gamma_1 \leq 1.0$. At iteration 740 both setpoints are set to 0 using large step setpoint changes (as opposed to setpoint scheduling) while the values of Γ and $T(q^{-1})$ are "frozen" and "tight" input rate constraints are applied. The maximum output prediction horizon is reset to a more conservative value of $N_2 = 10$.

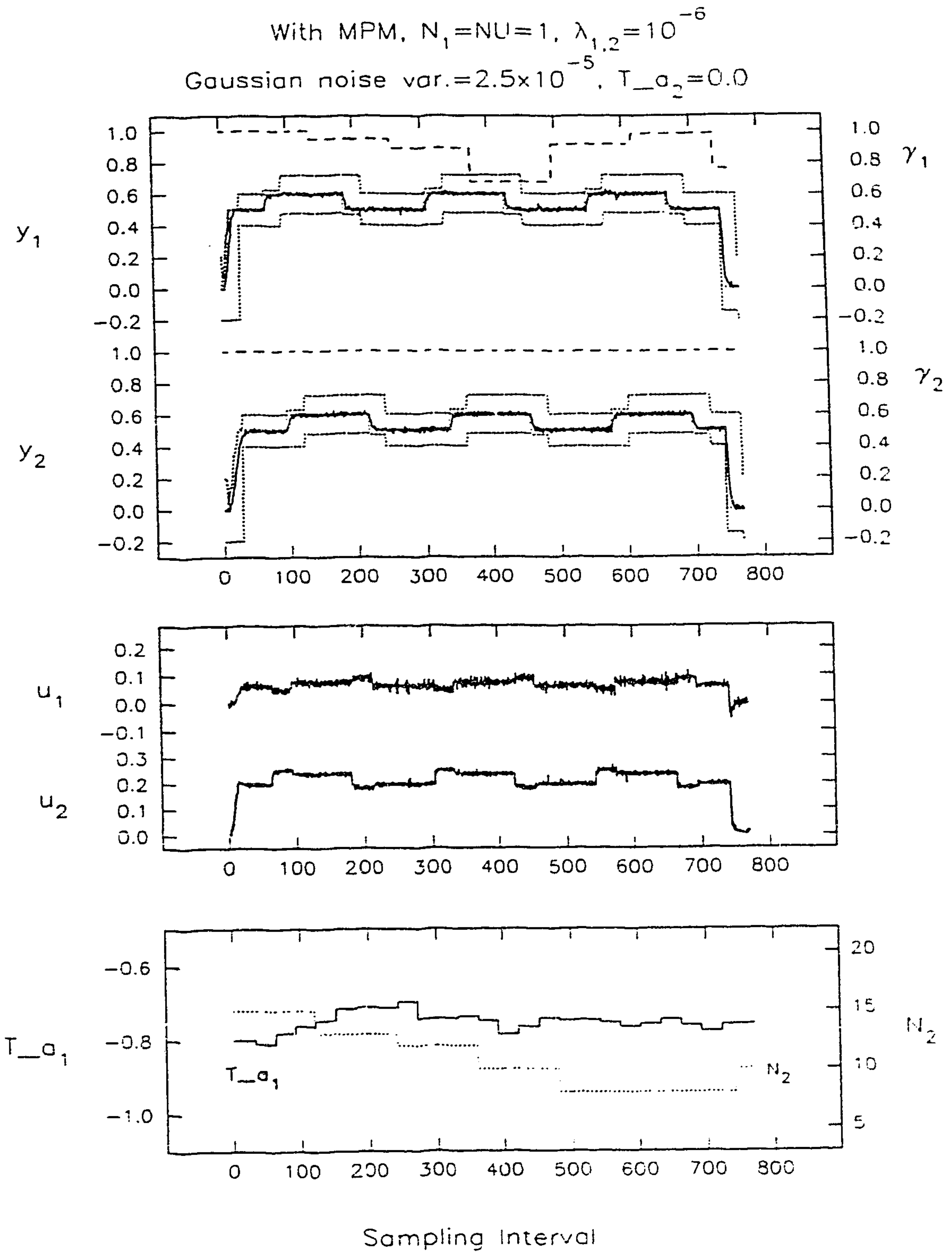


Figure 5.11 Process A, tuning of Γ, $T(q^{-1})$, N_2, with MPM, and Gaussian noise

In summary, both servo and regulatory supervision schemes worked very well when applied to process A. The conclusions drawn from figures 5.1 through 5.11 clearly show that, at least for process A, ALPS successfully adjusts the $T(q^{-1})$ filter order and break frequency, as well as the output prediction horizon N_2 to achieve and maintain the closed-loop regulatory and servo performance specifications under conditions of Gaussian noise, with and without MPM. Unfortunately, these figures also reveal a shortcoming in the Γ tuning strategy. Adjusting Γ as per equation 4.1 does have the desired effect (i.e. balancing the outputs such that $\sigma_{y,1} \approx \sigma_{y,2}$) provided, however, the adjustment is made under conditions of $\Gamma = \mathbf{I}$. Subsequent adjustments in Γ, with $\Gamma \neq \mathbf{I}$, do not achieve the desired effect as the <u>current</u> value of Γ is not taken into account. Thus equation 4.1 should be altered to include the setting of Γ for which $\sigma_{y,i}$ $i = 1...n$, are measured.

In addition, a simple analog low-pass filter should be used to filter "raw" data (i.e. filter y_i $i = 1...n$) and reduce the effect of Gaussian noise (i.e. measurement noise) on the calculated rise times. This will almost certainly eliminate the gradual "bouncing around" of N_2.

The next section evaluates the regulatory and servo performance tuning strategies as applied to process B.

5.2.2 Process B

The performance and tuning of the distillation column (process B) was completed following essentially the same procedure as for process A. Table 5.3 lists the closed-loop regulatory and servo control performance specifications used for the investigations.

5.2.2.1 Regulatory Performance Tuning

Simultaneous tuning of Γ and $T(q^{-1})$, under conditions of Gaussian noise and no MPM, is illustrated in figure 5.12. Initially the Gaussian noise variance is set to 1.7×10^{-3} and regulatory supervision is enabled at iteration 50. ALPS increases the amount of filtering and adjusts $T(q^{-1})$ until the performance specifications are met (tuning is completed in 3 to 4 successive $T(q^{-1})$ filter tuning periods, at iteration 190). Similarly, ALPS adjusts the output weighting Γ albeit in an undecided manner (i.e. during successive Γ tuning periods the output weighting of each output channel is set to unity). At iteration 260 the amount of Gaussian noise is reduced. ALPS responds to this new condition by reducing the amount of filtering until the performance specifications are met, at iteration 500. ALPS also adjusts Γ during this time frame. Throughout iterations 260 to 570 ALPS adjusts Γ, however, in an undecided manner. During consecutive Γ tuning periods ALPS weights alternate output channels heaviest (i.e. $\gamma_1 = 1.0$ and $\gamma_2 < 1.0$ followed by $\gamma_1 < 1.0$ and $\gamma_2 = 1.0$ etc.). Adjustments in Γ are performed according to equation 4.1, which does not take into account the current setting of Γ. Iteration 360 highlights this point. Here ALPS sets $\gamma_1 = 1.0$ and $\gamma_2 = 0.55$, and at iteration 420 scaling results in $\gamma_1 = 0.98$ and $\gamma_2 = 1.0$. This indicates that the previous settings (i.e. $\gamma_1 = 1.0$ and $\gamma_2 = 0.55$) should have been maintained since the desired result of $\sigma_{y,1} \approx \sigma_{y,2}$ was indeed achieved.

Figure 5.13 shows the effect of MPM on the regulatory tuning strategy under conditions of Gaussian noise. Initially the Gaussian noise variance is set to 1.7×10^{-3} and regulatory supervision is enabled at iteration 60. ALPS increases the amount of filtering

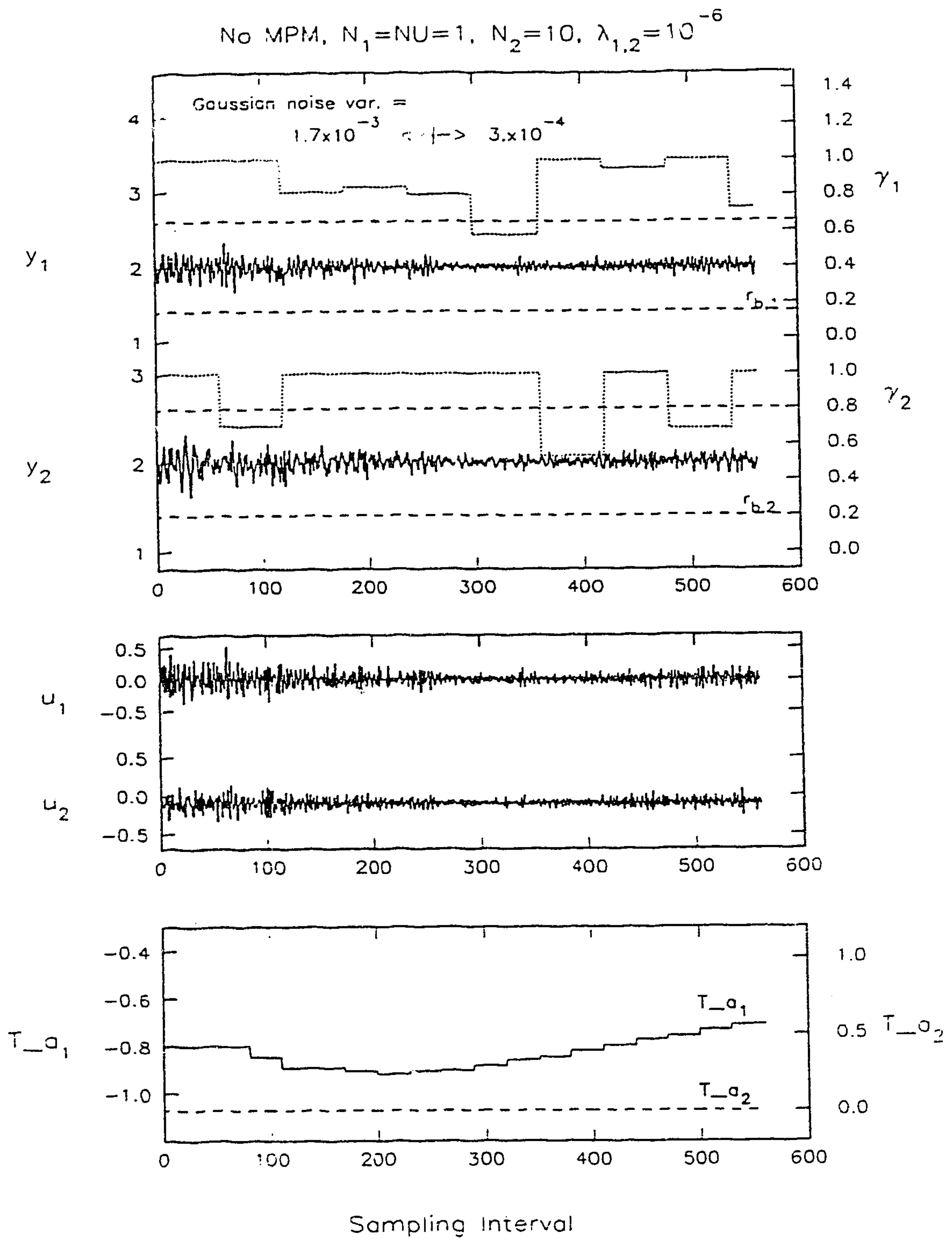


Figure 5.12 Process B, tuning of Γ and $T(q^{-1})$, no MPM

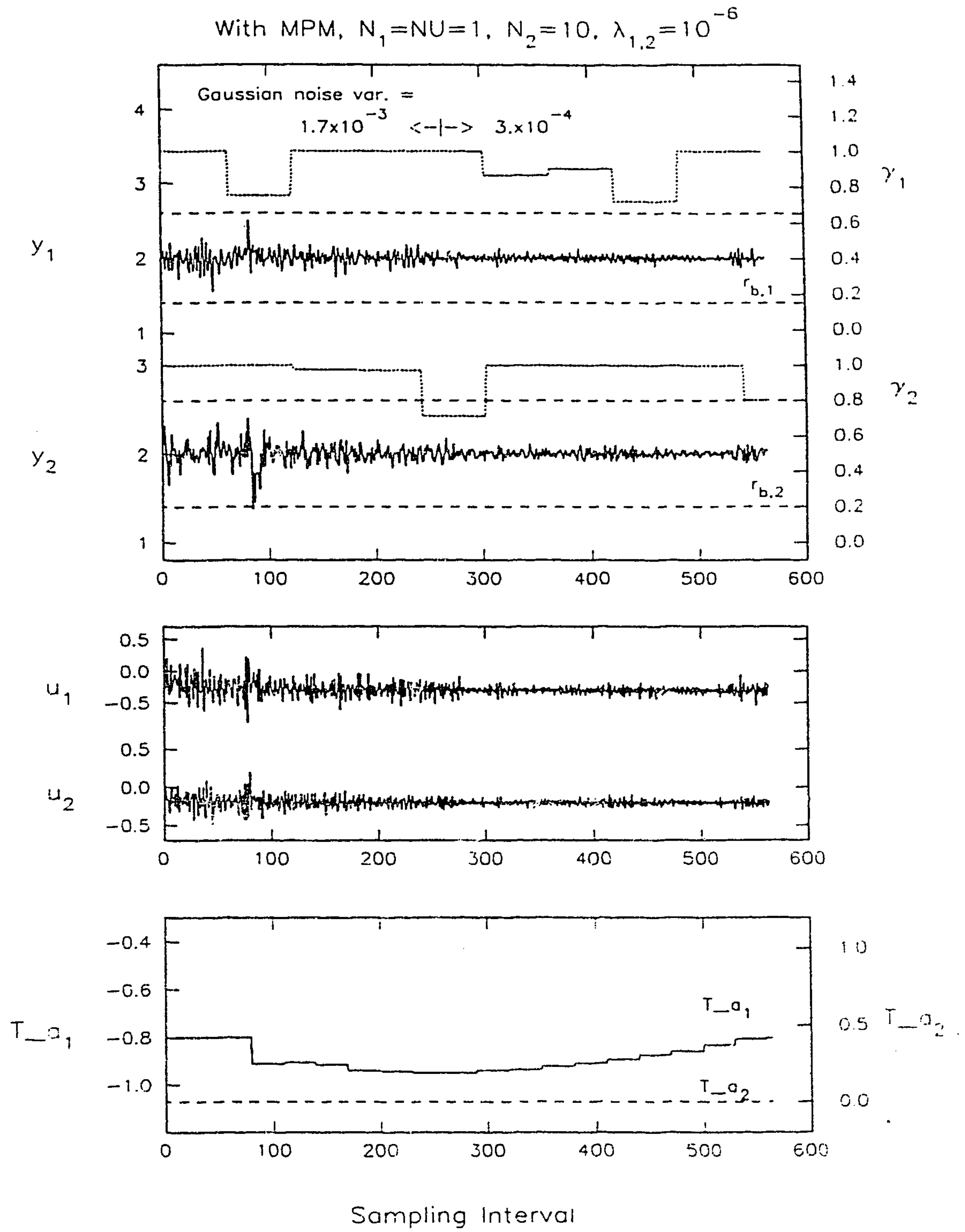


Figure 5.13 Process B, tuning of Γ and $T(q^{-1})$, with MPM

and meets the performance specifications with $T\_a_1 = -0.946$. At iteration 270 the Gaussian noise variance is reduced to 3.0×10^{-4}. ALPS responds by reducing the amount of filtering and meets the performance specifications with $T\_a_1 = -0.807$, at iteration 550. ALPS also adjusts the output weighting Γ throughout this simulation. The results obtained for the case of MPM are very similar to the results obtained without MPM. ALPS initially adjusts Γ such that the desired result of $\sigma_{y,1} \approx \sigma_{y,2}$ is met, only to reset $\Gamma = I$ in the subsequent tuning period. This of course allows $\sigma_{y,1}$ and $\sigma_{y,2}$ to drift apart. This indicates that the Γ tuning strategy, equation 4.1, must be updated to include the current value of Γ.

In summary, figures 5.12 and 5.13 show that ALPS successfully adjusts the $T(q^{-1})$ filter break frequency to achieve and maintain the closed-loop regulatory performance criteria, σ^2_u, under conditions of Gaussian noise, with and without MPM. In addition, these figures reveal that the current Γ tuning strategy (i.e. equation 4.1) should be updated such that the current setting of Γ is taken into account.

5.2.2.2 Servo Performance Tuning

Figure 5.14 illustrates the servo response tuning strategies of ALPS as applied to process B under ideal conditions (i.e. no MPM and no measurement noise). This figure depicts the trajectories of the output prediction horizon N_2, and output constraints y_{min} and y_{max}. The setpoints of y are square-waves of amplitude 0.5 and a d.c. component of 2.0 . Servo response supervision is enabled at iteration 0 with an initial value of $N_2 = 30$. The figure shows that ALPS adjusts N_2 progressively until the servo performance specifications are met with $N_2 = 9$.

Figure 5.15 illustrates the servo response tuning strategies under conditions of Gaussian noise and MPM. Servo supervision is enabled at iteration 0, with an initial value of $N_2 = 30$. This figure shows that ALPS consistently reduces N_2 until the servo performance specifications are satisfied with $N_2 = 8$.

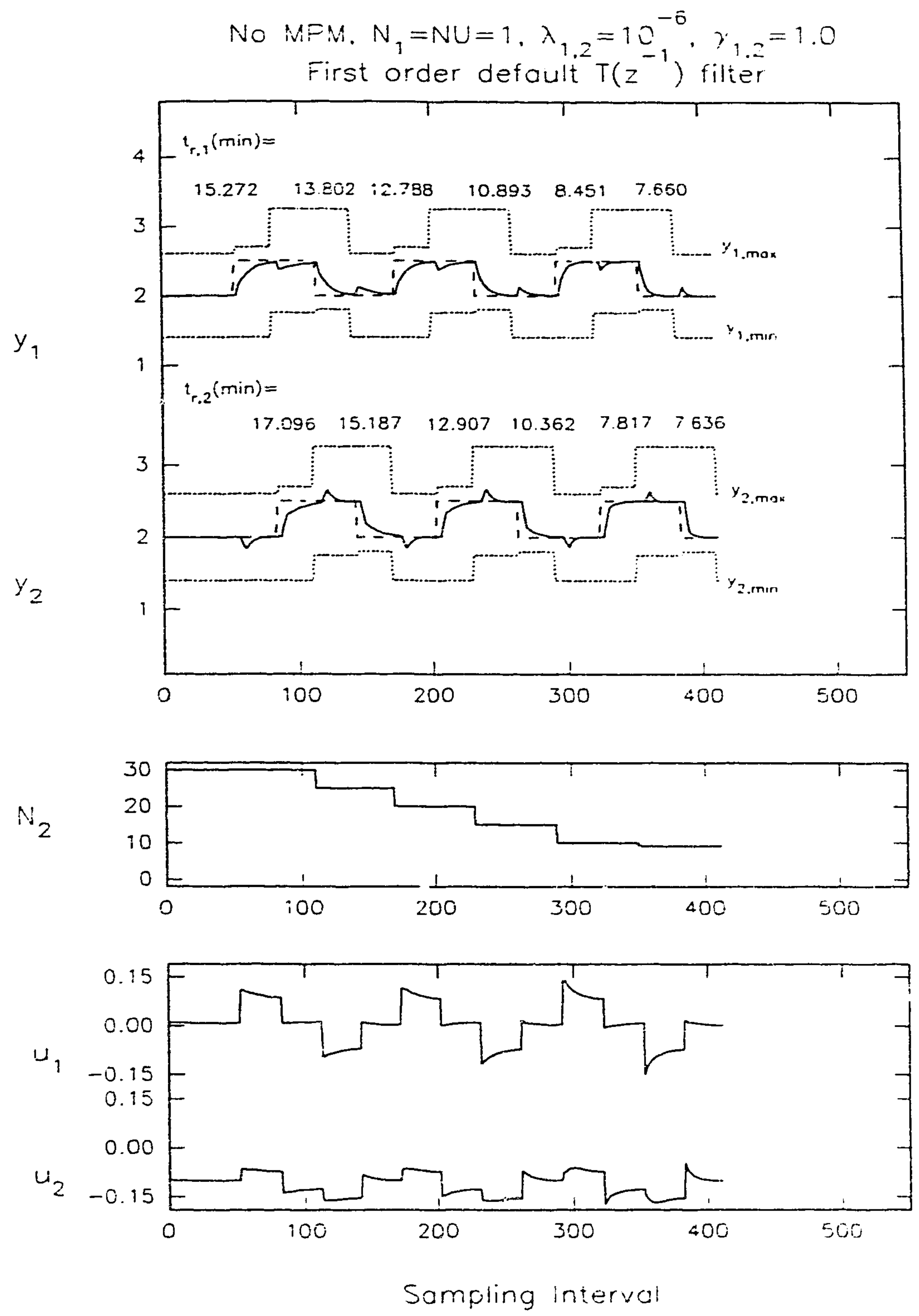


Figure 5.14 Process B, tuning of N_2, no MPM

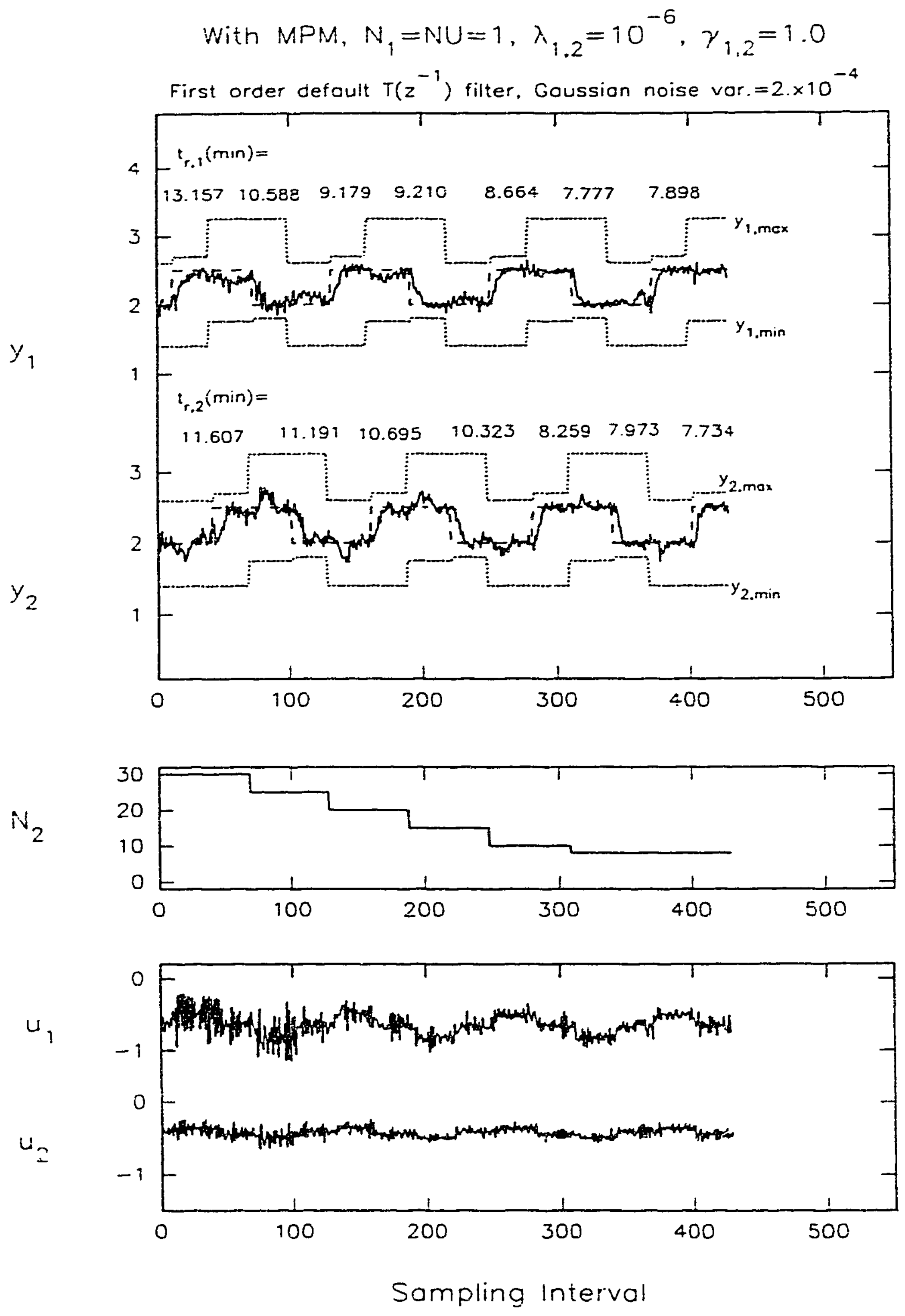


Figure 5.15 Process B, tuning of N_2, with MPM, with Gaussian noise

Figures 5.14 and 5.15 clearly show that ALPS successfully adjusts N_2 so as to achieve and maintain the closed-loop servo performance criteria, t_r, under conditions of Gaussian noise, with and without MPM. In addition, ALPS updates the output constraints y_{min} and y_{max} to reflect overshoot and regulatory band specifications. These figures also reveal that the calculation of the rise time (equation 3.8) is only marginally affected by Gaussian noise since the gradual "bouncing around" of N_2 is not observed.

Sections 5.2.2.1 and 5.2.2.2 evaluate ALPS' regulatory and servo response supervision procedures respectively. A more typical application of ALPS, namely simultaneous regulatory and servo response supervision, is shown in figure 5.16. The setpoints of y (i.e. w_1 and w_2) are square-waves of amplitude 0.5 with a d.c. component of 2.0. In addition, a ½ cycle square-wave is sufficiently long in duration for ALPS to tune Γ and $T(q^{-1})$. The simulation begins with a default first order $T(q^{-1})$ filter and $N_2 = 15$. Between iterations 0 to 40 setpoint scheduling is used to gradually increase the setpoints to values of 2.0. Throughout this period y_{max} is updated to reflect the overshoot specifications. Once both output channels reach steady state (i.e. within $0.5*(1 \pm r_b))$ ALPS updates y_{min} and y_{max} to reflect the r_b specifications. Following iteration 60 both setpoints are updated according to the square-waves discussed above. ALPS adjusts N_2, Γ, and $T(q^{-1})$ in order to meet the closed-loop servo and regulatory performance specifications. ALPS tunes these parameters and meets the performance specifications with $N_2 = 8$, and a first order $T(q^{-1})$ filter with $T\_a_1 = -0.72$. This figure also displays the gradual cycling of Γ (similar to figures 5.11, 5.12, and 5.13). At iteration 750 both setpoints are set to 0 using large step setpoint changes (as opposed to setpoint scheduling); while the values of Γ and $T(q^{-1})$ are "frozen", and "tight" input rate constraints are applied. The output prediction horizon is reset to a more conservative value of $N_2 = 15$.

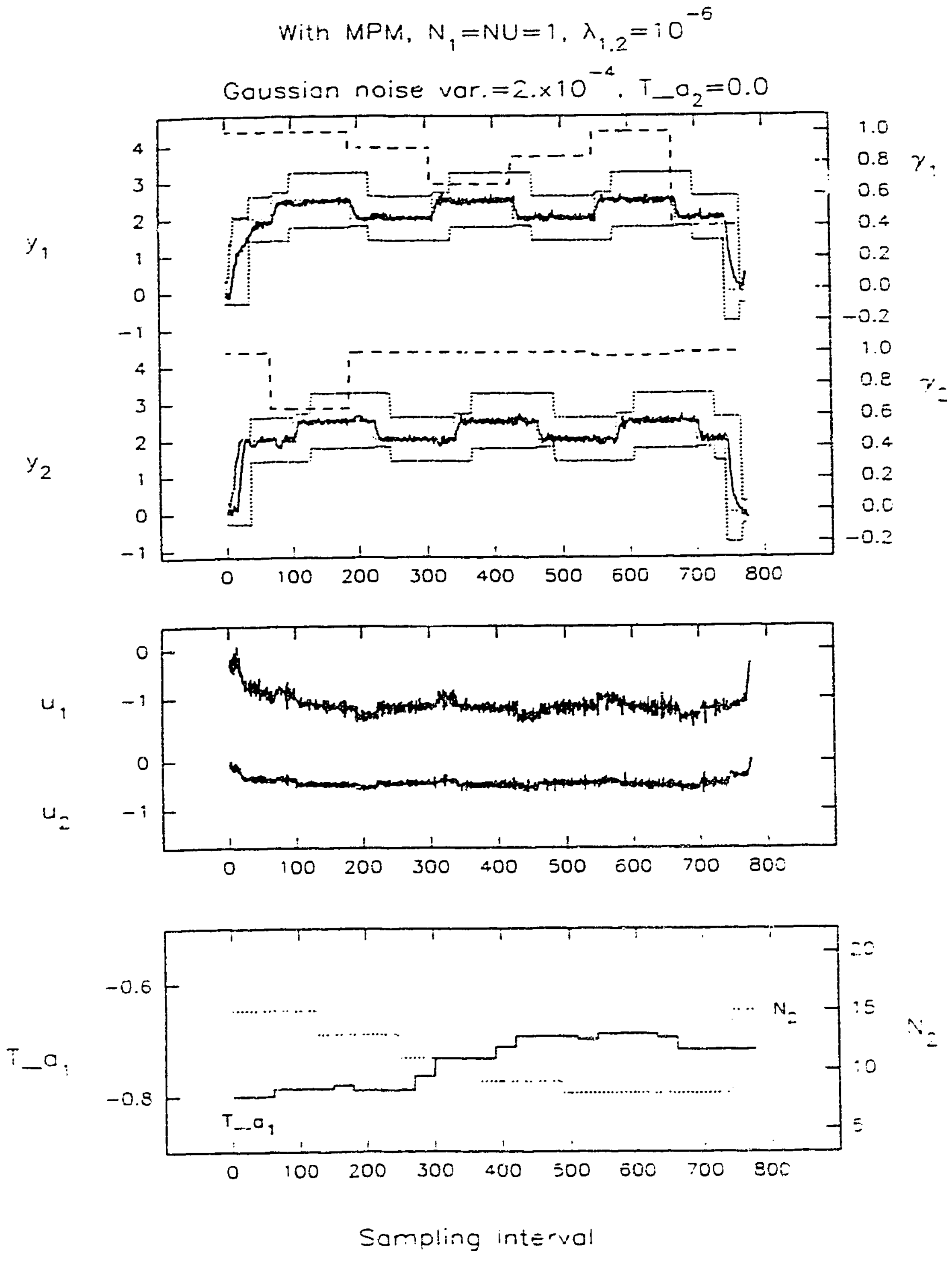


Figure 5.16 Process B, tuning of Γ, $T(q^{-1})$, N_2, with MPM and Gaussian noise

The regulatory and servo performance tuning strategies worked successfully when applied to process B. Figures 5.12 through 5.16 clearly prove that ALPS successfully adjusts the $T(q^{-1})$ filter and output prediction horizon, N_2, to achieve and maintain the closed-loop regulatory and servo performance specifications under conditions of Gaussian noise, with and without MPM. These figures also indicate the same shortcoming of the Γ tuning strategy as was observed with process A. In addition, these figures reveal that the calculation of the rise time (equation 3.8) is only marginally affected by Gaussian noise since the gradual "bouncing around" of N_2 is not observed during servo supervision.

The next section evaluates the regulatory and servo performance tuning strategies as applied to process C.

5.2.3 Process C

The simulation results conducted with process C are presented in this subsection. Table 5.4 lists the closed-loop performance specifications and input constraints used for the investigations.

5.2.3.1 Regulatory Performance Tuning

Consider figure 5.17 which illustrates the regulatory tuning strategy of adjusting both Γ and $T(q^{-1})$ under conditions of Gaussian noise without MPM. Initially the Gaussian noise variance is set to 10^{-4} and regulatory supervision is enabled at iteration 80. ALPS increases the amount of filtering and adjusts $T(q^{-1})$ until the performance specifications are met at iteration 210. ALPS also tunes the output weighting Γ which proved particularly effective for this process. During iterations 0 to 270 for example, ALPS adjusts Γ (i.e. $\gamma_2 = 1.0$ and $\gamma_1 = 0.55$) to reduce $\sigma_{y,2}$ relative to $\sigma_{y,1}$. At iteration 290 the amount of Gaussian noise is reduced to a variance of 2.0×10^{-5}. ALPS responds to this new condition by reducing the amount of filtering until the performance specifications are met. During this time ALPS adjusts Γ in slowly time varying cyclical manner similar to the results obtained for processes A and B.

Figure 5.18 depicts the regulatory tuning strategy under conditions of Gaussian noise and MPM. Initially the Gaussian noise variance is set to 10^{-4} and regulatory supervision is enabled at iteration 40. ALPS increases the amount of filtering and meets the performance specifications at iteration 130. At iteration 250 the Gaussian noise variance is reduced to 2.0×10^{-5}. ALPS responds by reducing the amount of filtering until the performance specifications are met (at iteration 330). ALPS adjusts the output weighting Γ throughout this simulation in a manner analogous to the discussion presented for figure 5.17.

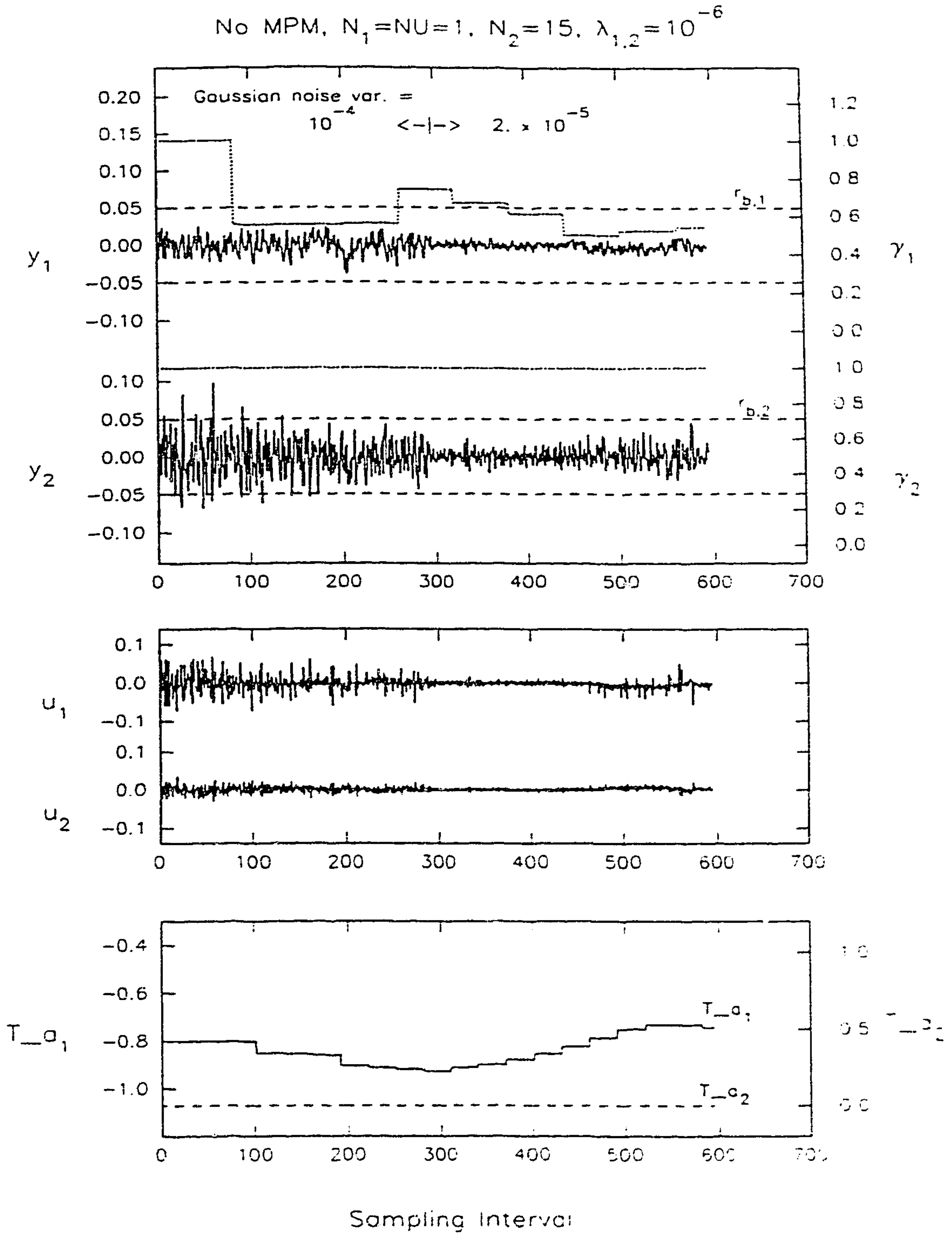


Figure 5.17 Process C, tuning of Γ and $T(q^{-1})$, no MPM

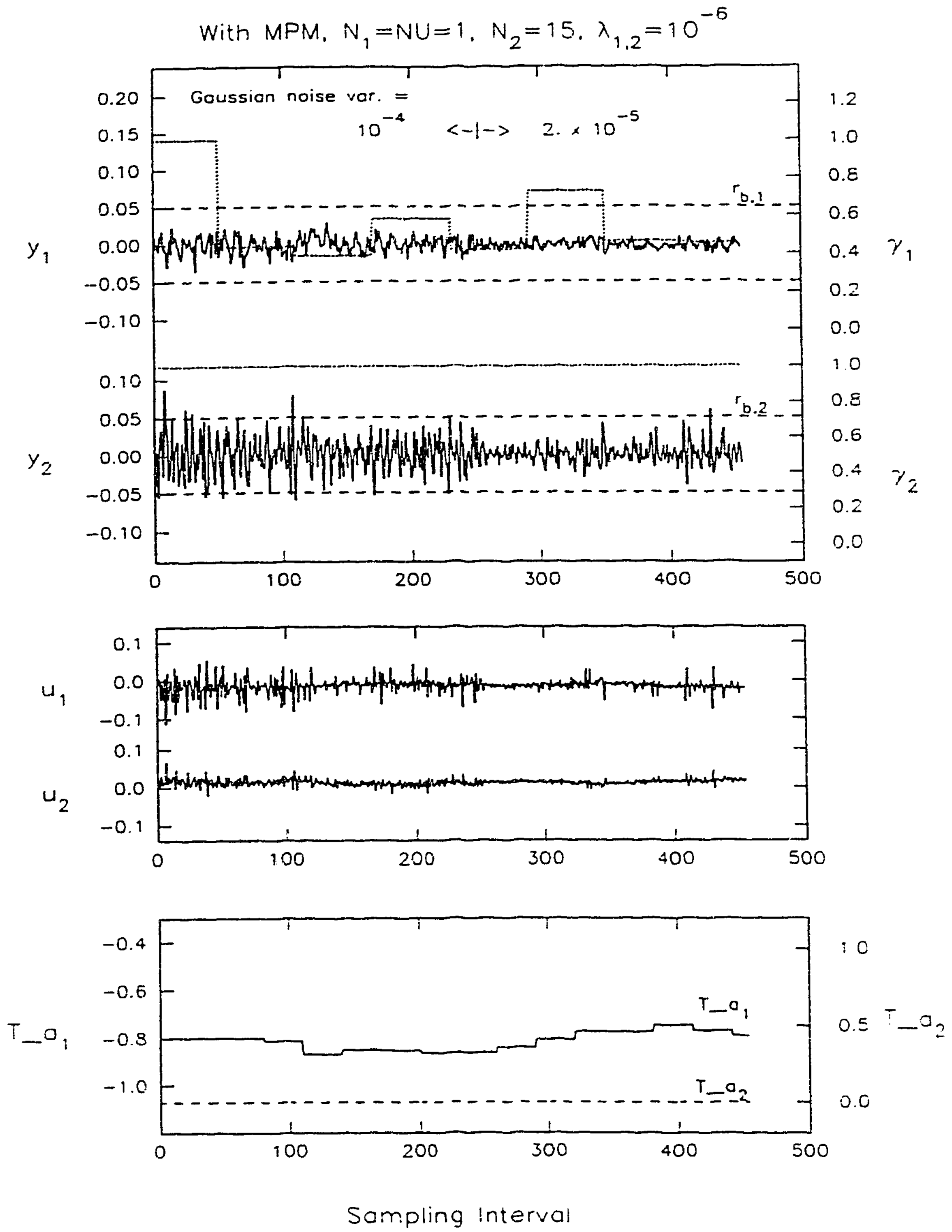


Figure 5.18 Process C, tuning of Γ and T(q^{-1}), with MPM

In summary, ALPS' regulatory supervision schemes worked successfully when applied to process C. ALPS skilfully adjusts the $T(q^{-1})$ filter break frequency to achieve and maintain the closed-loop regulatory performance specifications, σ^2_u, under conditions of Gaussian noise, with and without MPM. In addition, these figures revealed the same flaw in the Γ tuning strategy that was observed with processes A and B. Regardless of its deficient tuning strategy, Γ proved to be particularly effective in reducing $\sigma_{y,2}$ relative to $\sigma_{y,1}$.

5.2.3.2 Servo Performance Tuning

Numerous simulations using process C were performed. Even under ideal conditions of no MPM and no Gaussian noise ALPS was unable to adjust N_2 to meet the closed-loop servo control specifications listed in table 5.4. Clues as to the cause of ALPS' difficulties are revealed when considering the shape of the process' response to a setpoint change. The implemented LRPC algorithm readily controls this process, in fact, in response to a setpoint change, the LRPC algorithm is able to force output channel 2 to its new steady state in a single sample interval after the dead time (upon discretization output channel 2 has only a 1 sample delay due to the zero order hold). This means that $t_{r,2}$ is essentially invariant to N_2.

In contrast to output channel 2, output channel 1 has a very large dead time and very slow dynamics. The implications of this are that $t_{r,1}$ takes longer in duration than the allowed servo response "window" (recall section 4.3.3.1, servo response window is 30 sampling instances in duration). Thus for this particular process, $t_{r,1}$ could not be detected.

Simulations under conditions of MPM and Gaussian noise were also performed. ALPS encountered the same difficulties as discussed for the case without MPM, and hence was unable to satisfy the closed-loop servo response specifications.

In summary, ALPS' servo response supervision knowledge was unable to tune N_2 to meet the specified closed-loop servo response performance specifications. After several simulations it was observed that, subsequent to a step setpoint change, the implemented LRPC algorithm forced output channel 2 to its new steady state in a single sample interval after the dead time. Thus ALPS' heuristic N_2 tuning knowledge, which assumes a strong cause-and-effect relationship among t_r and N_2, proved to be not applicable to this process. Also, since output channel 1 displays a very large dead time and has very slow dynamics, $t_{r,1}$ could in fact not be measured.

This section evaluated the regulatory and servo performance tuning strategies used by ALPS as applied to process C. Figures (5.17) and (5.18) prove that ALPS successfully adjusts the $T(q^{-1})$ filter to achieve and maintain the closed-loop regulatory performance criteria under conditions of Gaussian noise, with and without MPM. These figures also show that Γ is a particularly useful tuning parameter for this process. Γ proved very effective in reducing $\sigma_{y,2}$ relative to $\sigma_{y,1}$. ALPS' servo response supervision knowledge, on the other hand, is inappropriate for process C. ALPS inherently assumes a strict cause-and-effect relationship between N_2 and t_r. This process illustrates that under certain conditions N_2 may be an ineffective servo response tuning parameter.

5.3 Summary

The studies conducted on three 2 × 2 benchmark problems included stochastic output disturbances (Gaussian noise) and MPM. The results conclusively show that ALPS effectively tunes the $T(q^{-1})$ filter order and break frequency to achieve and maintain user specified regulatory performance criteria.

In addition, the investigations illustrate that Γ is a useful tuning parameter in adjusting the relative standard deviations among output channels (i.e. reducing $\sigma_{y,2}$ relative to $\sigma_{y,1}$ or vice versa). Unfortunately, the results also reveal a shortcoming in the Γ tuning strategy. Adjusting Γ as per equation 4.1 does have the desired effect (i.e. balancing the outputs such that $\sigma_{y,1} \approx \sigma_{y,2}$) provided, however, the adjustment is made under conditions of $\Gamma = \mathbf{I}$. Subsequent adjustments in Γ, with $\Gamma \neq \mathbf{I}$, do not achieve the desired effect as the current value of Γ is not taken into account. Thus equation 4.1 should be altered to include the setting of Γ for which $\sigma_{y,i}$ $i = 1...n$, are measured.

ALPS' servo performance tuning strategy of adjusting N_2 with $NU = 1$ is also evaluated. The investigations undoubtedly show that ALPS maintains user specified servo performance criteria in the presence of MPM and Gaussian noise. Of particular interest are the results obtained from process C. This process highlights the underlying assumptions that govern the success of ALPS:

1) The user must supply performance specifications that are realistic and achievable given the adjustment ranges of the LRPC tuning parameters.

2) The current body of servo performance tuning knowledge, encoded within ALPS in terms of its generic rules and procedures, may prove inadequate for processes displaying very fast and/or very slow process dynamics.

In addition, ALPS' servo performance tuning strategy depends upon accurate detection of t_r. The current rise time detection scheme as given by equation 3.8 is susceptible to measurement noise. A simple analog low-pass filter should be used to filter "raw" data (i.e. y_i $i = 1...n$) and thus reduce the effect of measurement noise on the calculated rise times. This will almost certainly eliminate the gradual "bouncing around" of N_2 occasionally observed during servo performance tuning.

Chapter 6

Conclusions and Recommendations

This thesis focused on Supervisory LRPC for the purpose of performance tuning in an on-line real-time environment. The major contribution of this work is the development of the Adaptive Long range predictive control Performance Supervisor, ALPS. ALPS is implemented with existing ES technology and monitors the closed-loop control performance of several LRPC algorithms simultaneously. ALPS adjusts LRPC tuning parameters based upon the actual closed-loop control performance visavis user specified control performance. The tuning strategies employed by ALPS were evaluated on MIMO benchmark processes. These investigations showed that ALPS skilfully adjusts the controller tuning parameters. ALPS' supervisory roles became apparent in two distinct ways. Firstly, even under ideal conditions (i.e. no MPM and no disturbances) the appropriate controller settings resulting in the desired closed-loop control performance may not always be known *a priori*. Thus ALPS allows commissioning of LRPC algorithms with conservative controller settings. Subsequent to commissioning, ALPS manipulates the appropriate LRPC tuning parameters so as to achieve the user specified control performance. And secondly, ALPS continually monitors the actual closed-loop control performance and fine-tunes controller settings in order to maintain the user specified control performance in the presence of MPM and disturbances. Overall ALPS proved to be very successful in achieving and maintaining the user specified closed-loop control performance. There is, of course, room for improvement. Suggestions for future work are given in section 6.1. The main contributions of this thesis are listed below:

1. Development of the ALPS.kb knowledge base

The Adaptive Long range predictive control Performance Supervisor was developed with Gensym's object oriented real-time expert system development tool G2. G2 facilitates rapid prototyping and offers an exceptional development environment with a rich set of built-in functions. This tool enables the developer to incrementally test and expand his applications (not possible with conventional programming languages such as FORTRAN or 'C'). First time knowledge base developers though, will find G2 to be rather complex due to its (almost) countless number of options. G2 has a very steep learning curve. Once mastered, however, the developer quickly appreciates its tremendous capabilities. End-users, on the other hand, find G2 easy to use; they merely point and click with the mouse. It is safe to say that ALPS could not have been developed and implemented in as short a time frame (about 8 months) without an ES development tool.

For educational purposes ALPS was not optimized for execution speed rather ALPS was implemented in a fashion illustrating the following features that are provided with G2:

- forward chaining and backward chaining
- "whenever" rules and "initially" rules
- procedures
- external procedure calls
- object oriented programming
- formulas
- focus
- invoke
- tabular functions
- GSI
- generic and specific knowledge

2. Development of on-line LRPC tuning guidelines

Time domain performance criteria were selected for implementation with ALPS due to their heuristic and intuitive nature. Based on these time domain performance criteria, regulatory and servo control tuning strategies suitable for on-line implementation were developed. Chapter 3 lays the ground work by defining the performance criteria and selecting appropriate LRPC tuning parameters. The very heart of ALPS - its tuning strategies - are detailed in chapter 4. Overall, based on the investigations conducted on the MIMO benchmark problems, ALPS' tuning strategies worked very well. ALPS does have its limitations, though, and the following assumptions should provide some guidelines:

i) The user must supply performance specifications that are realistic and achievable given the adjustment ranges of the LRPC tuning parameters.

ii) The current body of servo performance tuning knowledge, encoded within ALPS in terms of its generic rules and procedures, may prove inadequate for processes displaying very fast and/or very slow process dynamics. In addition, ALPS' servo performance tuning strategies are useful provided the dimension of the MIMO process is small (say $n \leq 4$, where n is the number of output channels).

iii) The recommended improvements listed in section 6.1 should be implemented.

3. Training facilities provided with ALPS.kb

ALPS' supervision modes may be disabled to allow the user to manually adjust all LRPC tuning parameters in addition to performing setpoint changes. With both servo and regulatory response supervision disabled, the role of ALPS is reduced to providing an interactive real-time interface between the user, the LRPC algorithms, and the processes. ALPS merely schedules data I/O and the execution of the LRPC algorithms at every sampling interval. The user may manually change any of the LRPC tuning parameters on-line using the mouse. The visual impact of the this process-LRPC-operator interface is very gripping. Data is displayed continuously and in real-time in the form of graphs and tables; providing immediate feedback to the user. Also, note that this manual mode of operation is useful for establishing realistic and achievable performance specifications (see item 'i' above).

Optional features useful for investigating the performance of ALPS itself as well as the underlying LRPC algorithms have been incorporated into the its knowledge base. The following features may be selected on-line using the mouse:

- use of $T(q^{-1})$ filter or no $T(q^{-1})$ filter
- switch from first to second order $T(q^{-1})$ filter, and vice versa
- constrained solution using QPSOL or unconstrained solution using equation 2.19

In addition to the above, when using a process simulator (see section 4.3.2.1) the user may select among the following options at any time with the mouse:

- MPM or no MPM
- input step disturbances, $d_u(t)$, of a user specified amplitude can be implemented at any input channel
- stochastic zero-mean output disturbances, $d_y(t)$, with a Gaussian or Normal distribution, at a user specified variance

4. MGPC for the HP-9000 hardware platform

Software was developed to implement the constrained MGPC algorithm of Mutha (1990) for the HP-9000 hardware platform. This algorithm was chosen since it has a unique two degree of freedom structure allowing for independent regulatory and servo control performance specifications. Furthermore, both amplitude and rate constraints can be imposed on process inputs, and amplitude constraints can be imposed on process outputs. This algorithm uses the commercially available quadratic optimization package QPSOL (Gill *et al.* 1990), and demonstrates mixed language programming techniques between FORTRAN and 'C' for the HP-UX operating system.

6.1 Recommended Improvements for ALPS

1. ALPS' servo performance tuning strategy depends upon accurate measurements of t_r. The current rise time detection scheme, given by equation 3.8, is susceptible to measurement noise. A simple analog low-pass filter should be used to filter "raw" data (i.e. filter y_i $i = 1...n$) and thus reduce the effect of measurement noise on the calculated rise times. This will almost certainly eliminate the gradual "bouncing around" of N_2 occasionally observed during servo performance tuning.

2. ALPS' Γ tuning strategy can also be improved. Regulatory performance tuning consists of tuning the $T(q^{-1})$ filter as well as Γ. The role of Γ is to balance the LRPC algorithm such that the weighted standard deviation of all output channels are of the same order of magnitude. The current Γ tuning strategy, equation 4.1, should be altered to include the setting of Γ for which the standard deviations of all output channels are measured.

6.2 Suggestions for Future Work

1. A more thorough investigation of the effect of N_1, N_2, and NU, on the rise times for MIMO processes is needed. Chapter 3 showed that the rise times are unaffected by the choice of N_2 if $NU > 1$ and $N_1 = 1$; and the constraints are not active. This unexpected result warrants further investigation.

2. The performance of ALPS, based on simulations conducted with MIMO benchmark problems, was confirmed. Experimental studies on a pilot-scale process are in progress (Dhaliwal, 1992) and should result in further improvements to ALPS' performance tuning strategies.

3. High level diagnostics strategies should be developed to exploit ALPS' generic architecture. Since ALPS is capable of supervising several LRPC algorithms simultaneously, intelligent diagnostics schemes should be developed to detect and diagnose faults among sequential process units.

4. The MGPC algorithm has been implemented on the HP-9000 as a non-adaptive control strategy. If a multivariable identification package is available, this algorithm can be readily implemented as a multivariable adaptive control algorithm. In addition, feedforward control should be added (see e.g. Mutha, 1990). These additions, however, may require the modification of ALPS' performance tuning knowledge.

References

J. Adams, E. Rachlin, and E. Sullivan, 1987. "COMA", presented at the 1987 annual AIChE meeting, New York, New York.

J. L. Alty and G. Johannsen, 1989. "Knowledge-based Dialogue for Dynamic Systems", *Automatica*, **25** (6), 829-840.

K. E. Årzén, 1989a. "An Architecture for Expert System Based Feedback Control", *Automatica*, **25** (6), 813-827.

K. E. Årzén, 1989b. " Knowledge-based control systems - aspects on the unification of conventional control systems and knowledge-based systems", *Proc. 1989 American Control Conference*, Pittsburgh, **3** 2233-2238.

K. J. Åström, 1989. "Toward Intelligent Control", keynote speech to the 1988 American Control Conference, *IEEE Control Systems Magazine*, April 1989, 60-64.

K. J. Åström, J. J. Anton, and K. E. Årzén, 1986. "Expert Control", *Automatica*, **22** (3), 277-286.

M. Aynsley, D. Peel, and A. J. Morris, 1989. "A Real-Time Knowledge-Based System for Fermentation Control", *Proc. 1989 American Control Conference*, Pittsburgh, **3** 2239-2244.

Th. Beck and R. J. Lauber, 1990. "Integration of an Expert System into a Real-Time Software System", *11th IFAC World Congress*, Tallinn, Estonia, **7**, 158-161.

M. W. Berry, 1973. *Binary Composition Control of a Binary Distillation Column*. M.Sc. thesis, Dept. of Chem. Eng., University of Alberta.

A. Betta and D. A. Linkens, 1990. "Intelligent knowledge-based system for dynamic system identification", *IEE Proc. D*, **137** (1) 1-12.

D. W. Clarke, 1991. "Adaptive Generalized Predictive Control", presented at 4th International Conference on Chemical Process Control, Feb 17-22, 1991, South Padre Island, Texas.

D. W. Clarke, C. Mohtadi, and P. S. Tuffs, 1987. "Generalised Predictive Control - Part I. The Basic Algorithm", *Automatica*, **23** (2), 137-148.

D. W. Clarke, C. Mohtadi, and P. S. Tuffs, 1987. "Generalised Predictive Control - Part II. Extensions and Interpretations", *Automatica*, **23** (2), 149-160.

D. W. Clarke and C. Mohtadi, 1989. "Properties of Generalised Predictive Control", *Automatica*, **25** (6), 859-875.

C. R. Cutler and R. B. Hawkins, 1988. "Application of a large predictive multivariable controller to a hydrocracker second stage reactor", *Proc. 1988 American Control Conference*, Atlanta, GA.

C. R. Cutler and B. L. Ramaker, 1980. "Dynamic Matrix Control - A Computer Control Algorithm", *Proc. 1980 American Control Conference*, San Francisco, paper WP5-B, (also presented at 83rd National AIChE Meeting, Houston, 1979).

[illegible] S. Dhaliwal, 1992. *Supervisory Long Range Predictive Control*. M.Sc. thesis in progress, Dept. of Chem. Eng., University of Alberta.

R. Dong, H. Bacon, V. Mahalec, and M. Rao, 1991. "An Expert Advisory System for Safe Plant Startup", *4th International Symposium on Process Systems Engineering*, Montebello, Quebec, August 5-9, IV.4.1-IV.4.15.

R. Doraiswami and J. Jiang, 1989. "Performance Monitoring in Expert Control Systems", *Automaitca*, **25** (6), 799-811.

G2 Reference Manual, Version 2.0, Gensym Corporation, Cambridge Park Drive, Cambridge, MA 02140.

C. E. Garcia and A. M. Morshedi, 1986. "Quadratic programming solution of Dynamic Matrix Control (QDMC)", *Chem. Engng. Commun.*, **46**, 73-87.

C. E. Garcia, D. M. Prett, and M. Morari, 1989. "Model Predictive Control: Theory and Practice - a Survey", *Automatica*, **25**, (3), 335-348.

P. E. Gill, W. Murray, M. A. Saunders, and M. H. Wright, 1984. "User's Guide for QPSOL: A Fortran Package for Quadratic Programming", *Technical Report SOL 84-6*, Dept. of Operations Research, Stanford University, Stanford, U.S.A.

GSI User's Manual, Version 2.0, Gensym Corporation, Cambridge Park Drive, Cambridge, MA 02140.

M. Haest, G. Bastin, M. Gevers, and V. Wertz, 1990. "ESPION: an Expert System for System Identification", *Automatica*, **26** (1), 85-95.

S. J. Kelly, M. D. Rogers, and D. W. Hoffman, 1988. "Quadratic Dynamic Matrix Control of Hydrocracking Reactors", *Proc. 1988 American Control Conference*, Atlanta, GA.

S. Li, K. Y. Lim, and D. G. Fisher, 1989. "A State Space Formulation for Model Predictive Control", *AIChE Journal*, **35** (2), 241-249.

R. S. H. Mah, K. D. Schnelle, and A. N. Patel, 1991. "A plant-wide quality expert system for Steel Mills", *Comput. Chem. Eng.*, **15** (6), 445-450.

A. R. McIntosh, 1988. *Performance and Tuning of Adaptive Generalized Predictive Control*. M.Sc. thesis, Dept. of Chem. Eng., University of Alberta.

A. R. McIntosh, S. L. Shah, and D. G. Fisher, 1989. "Analysis and Tuning of Adaptive Generalized Predictive Control", *Can. J. Chem. Eng.*, **69**, 97-110.

B. J. Minter and D. G. Fisher, 1988. "A Comparison of Adaptive Controllers: Academic vs Industrial", *Proc. 1988 American Control Conference*, Atlanta, 1653-1658.

C. Mohtadi, S. L. Shah, and D. G. Fisher, 1991. "Frequency Response Characteristics of MIMO GPC", *Proc. 1991 European Control Conference*, Grenoble, France, 2 1845-1850. Also accepted for publication in *International J. of Control* (date of acceptance January 1991).

C. Mohtadi, 1987. *Advanced Self-Tuning Algorithms*. D. Phil thesis, University of Oxford.

R. Moore, H. Rosenof, and G. Stanley, 1990. "Process control using a real-time expert system", *11th IFAC World Congress*, Tallinn, Estonia, 7, 234-239.

R. K. Mutha, 1990. *Constrained Long Range Predictive Control*. M.Sc. thesis, Dept. of Chem. Eng., University of Alberta.

K. Ogata, 1970. *Modern Control Engineering*, Prentice-Hall, Englewood Cliffs, N.J.

R. A. Oyen, M. P. Lukas, and M. G. O'Callaghan, 1990. "Incorporation of AI and Expert Systems in Distributed Computer Control Systems", *11th IFAC World Congress*, Tallinn, Estonia, **7**, 94-99.

D. M. Prett and M. Morari, 1986. *Shell Process Control Workshop*, Butterworth Publishers, Stoneham, MA.

M. Rao, T. S. Jiang, and J. J. P. Tsai, 1989. "Combining symbolic and numerical processing for real-time intelligent control", *Eng. Appli. of AI*, **2**, March 1989, 19-27.

R. Scattolini and S. Bittanti, 1990. "On the Choice of the Horizon in Long-range Predictive Control - Some Simple Criteria", *Automatica*, **26** (5), 915-917.

D. E. Seborg, T. F. Edgar, and D. A. Mellichamp, 1989. *Process Dynamics and Control*, John Wiley & Sons, Toronto, Ont.

S. L. Shah, C. Mohtadi, and D. W. Clarke, 1987. "Multivariable adaptive control without a prior knowledge of the delay matrix", *Systems & Control Letters*, **9**, 295-306.

N. R. Sripada, D. G. Fisher, and A. J. Morris, 1987. "AI application for process regulation and servo control", *IEE Proc. D*, **134** (4) 251-259.

N. R. Sripada and D. G. Fisher, 1985. "Multivariable Optimal Constrained Control Algorihtm (MOCCA): Part 1. Formulation and Application", *Proc. Int. Conf. on Industrial Process Modeling and Control*, **1**, Hangzhou, China (June 1985).

G. Stephanopoulos, 1984. *Chemical Process Control*, Prentice-Hall, Englewood Cliffs, N.J.

V. K. Tzouanas, C. Georgakis, W. L. Luyben, and L. H. Ungar, 1988. "Expert Multivariable Control", *Comput. Chem. Eng.*, **12**, 1065-1074.

A. Wolfe, 1987. "An easier way to build a real-time Expert System", *Electronics* March 1987, 71-73.

R. K. Wood and M. W. Berry, 1973. "Terminal Compensation Control of a Binary Distillation Column", *Chem. Eng. Science*, **28** 1707-1717.

E. Zafiriou, 1991. "On the robustness of Model Predictive Controllers", presented at 4th International Conference on Chemical Process Control, Feb 17-22, 1991, South Padre Island, Texas.

L. Zhongyuan, 1990. "Expert Control Strategy used for CFB Boiler Control Systems", *11th IFAC World Congress*, Tallinn, Estonia, 7, 50-53.

Appendix A

This appendix lists the classes, rules, procedures, and formulas used in the ALPS.kb knowledge base. The listing is divided into the following sections:

1) Classes
2) Procedures
3) Global Sup[illegible]vision rules
4) Regulatory Supervision rules
5) Servo Supervision rules for adjusting y_{min}, y_{max}
6) Servo Supervision rules for adjusting N_2

1. Classes

```
                         3BY3_PLANT, an object-definition
Notes                          OK
User restrictions              none
Class                          3by3_plant
Superior class                 mimo_plant
Attributes specific to class   u1 is given by a quantitative-variable;
                               u2 is given by a quantitative-variable;
                               u3 is given by a quantitative-variable;
                               y1 is given by a quantitative-variable;
                               ysp1 is given by a quantitative-variable;
                               y2 is given by a quantitative-variable;
                               ysp2 is given by a quantitative-variable;
                               y3 is given by a quantitative-variable;
                               ysp3 is given by a quantitative-variable
Capabilities and restrictions  none
Change                         none
Menu option                    a final menu choice
Inherited attributes           n1 is an instance of a gpc_tunig_knob;
                               n2 is an instance of a gpc_tunig_knob;
                               nu is an instance of a gpc_tunig_knob;
                               max_delay is an instance of a plant_data;
                               tau_over_ts   is   given   by   a
                               quantitative-variable;
                               t_filter_a1   is   an   instance   of   a
                               gpc_tunig_knob;
                               t_filter_a2   is   an   instance   of   a
                               gpc_tunig_knob
Default settings               none
Stubs                          an input flow-pipe inlet-port-1 located at
                               left 130;
                               an input flow-pipe inlet-port-2 located at
                               left 200;
                               an input flow-pipe inlet-port-3 located at
                               left 270;
                               an output flow-pipe outlet-port-1 located
                               at right 130;
                               an output flow-pipe outlet-port-2 located
                               at right 200;
                               an output flow-pipe outlet-port-3 located
                               at right 270
Color                          inherited
Icon description               width 200; height 290;
                               black:
                                    outline (0, 90) (0, 290) (200, 290)
                                    (200, 90);
                                    lines (30, 170) arc (48, 197) (26,
                                    200);
                                    lines (28, 137) arc (50, 152) (30,
                                    170);
                                    lines (135, 136) arc (160, 152) (134,
                                    171);
                                    lines (136, 171) arc (158, 194) (130,
                                    206);
                                    lines (77, 158) (102, 176);
                                    lines (77, 176) (102, 158);
                               black:
                                    outline (0, 0) (0, 90) (200, 90)
                                    (200, 0);
                               black:
```

lines (54, 73) (54, 18) (43, 45) (29, 17) (29, 73);
lines (90, 18) arc (65, 43) (88, 73);
lines (96, 73) (96, 45) (76, 45);
lines (111, 18) (111, 73);
lines (111, 18) arc (130, 35) (111, 47);
lines (162, 17) arc (140, 47) (166, 73)

1BY1_PLANT, an object-definition

Notes	OK
User restrictions	none
Class	1by1_plant
Superior class	mimo_plant
Attributes specific to class	u is given by a quantitative-variable; y is given by a quantitative-variable; ysp is given by a quantitative-variable
Capabilities and restrictions	none
Change	none
Menu option	a final menu choice
Inherited attributes	n1 is an instance of a gpc_tunig_knob; n2 is an instance of a gpc_tunig_knob; nu is an instance of a gpc_tunig_knob; max_delay is an instance of a plant_data; tau_over_ts is given by a quantitative-variable; t_filter_a1 is an instance of a gpc_tunig_knob; t_filter_a2 is an instance of a gpc_tunig_knob
Default settings	none
Stubs	an input flow-pipe located at left 120; an output flow-pipe located at right 120
Color	inherited
Icon description	width 101; height 165;

```
black:
     filled rectangle (22, 90) (23, 137);
     filled rectangle (73, 90) (74, 138);
     lines (40, 122) (59, 106);
     lines (40, 107) (57, 122);
     outline (0, 70) (0, 165) (101, 165)
     (101, 70);
     outline (0, 0) (0, 70) (101, 70)
     (101, 0);
black:
     lines (26, 48) (26, 18) (22, 34) (16,
     18) (16, 48);
     lines (42, 19) arc (32, 34) (42, 48);
     lines (45, 48) (45, 32) (36, 32);
     lines (52, 18) (52, 48);
     lines (52, 18) arc (62, 27) (52, 34);
     lines (79, 18) arc (68, 34) (80, 48)
```

```
                         2BY2-PLANT, an object-definition
Notes                          OK
User restrictions              none
Class                          2by2-plant
Superior class                 mimo_plant
Attributes specific to class   u1 is given by a quantitative-variable;
                               u2 is given by a quantitative-variable;
                               y1 is given by a quantitative-variable;
                               ysp1 is given by a quantitative-variable;
                               y2 is given by a quantitative-variable;
                               ysp2 is given by a quantitative-variable
Capabilities and restrictions  none
Change                         none
Menu option                    a final menu choice
Inherited attributes           n1 is an instance of a gpc_tunig_knob;
                               n2 is an instance of a gpc_tunig_knob;
                               nu is an instance of a gpc_tunig_knob;
                               max_delay is an instance of a plant_data;
                               tau_over_ts     is     given     by     a
                               quantitative-variable;
                               t_filter_a1    is    an    instance    of    a
                               gpc_tunig_knob;
                               t_filter_a2    is    an    instance    of    a
                               gpc_tunig_knob
Default settings               none
Stubs                          an input flow-pipe inlet-port-1 located at
                               left 130;
                               an input flow-pipe inlet-port-2 located at
                               left 210;
                               an output flow-pipe outlet-port-1 located
                               at right 130;
                               an output flow-pipe outlet-port-2 located
                               at right 210
Color                          inherited
Icon description               width 150; height 240;
                               black:
                                    outline (0, 91) (0, 240) (150, 240)
                                    (150, 91);
                                    lines (12, 142) arc (28, 133) (36,
                                    156);
                                    lines (12, 182) (36, 155);
                                    lines (12, 182) (42, 182);
                                    lines (104, 140) arc (126, 134) (126,
                                    153);
                                    lines (102, 184) (126, 154);
                                    lines (102, 184) (129, 184);
                                    lines (53, 148) (85, 167);
                                    lines (53, 166) (88, 146);
                               black:
                                    outline (0, 0) (0, 91) (150, 91)
                                    (150, 0);
                               black:
                                    lines (40, 68) (40, 18) (31, 40) (19,
                                    18) (19, 68);
                                    lines (65, 18) arc (49, 42) (65, 68);
                                    lines (71, 68) (71, 42) (56, 42);
                                    lines (84, 18) (84, 68);
                                    lines (84, 18) arc (103, 32) (84,
                                    44);
                                    lines (129, 18) arc (109, 42) (129,
                                    68)
```

```
                    MIMO_PLANT, an object-definition
Notes                           OK
User restrictions               none
Class                           mimo_plant
Superior class                  object
Attributes specific to class    n1 is an instance of a gpc_tunig_knob;
                                n2 is an instance of a gpc_tunig_knob;
                                nu is an instance of a gpc_tunig_knob;
                                max_delay is an instance of a plant_data;
                                tau_over_ts   is   given   by   a
                                quantitative-variable;
                                t_filter_a1   is   an   instance   of   a
                                gpc_tunig_knob;
                                t_filter_a2   is   an   instance   of   a
                                gpc_tunig_knob
Capabilities and restrictions   none
Change                          none
Menu option                     not a final menu choice
Inherited attributes            none
Default settings                none
Stubs                           none inherited
Color                           inherited
Icon description                inherited


                    MIMO_OUTPUT, an object-definition
Notes                           OK
User restrictions               none
Class                           mimo_output
Superior class                  object
Attributes specific to class    y is given by a plant_data;
                                y_ll is given by a gpc_tunig_knob;
                                y_hl is given by a gpc_tunig_knob;
                                outputweight is given by a gpc_tunig_knob;
                                ysp is given by a gpc_tunig_knob;
                                max_overshoot is given by a quantitative-
                                parameter;
                                regulation_tolerance   is   given   by   a
                                quantitative-parameter;
                                servo_status   is   given   by   a
                                quantitative-parameter;
                                desired_tr   is   given   by   a
                                quantitative-parameter;
                                observed_tr   is   given   by   a
                                quantitative-variable
Capabilities and restrictions   none
Change                          none
Menu option                     a final menu choice
Inherited attributes            none
Default settings                none
Stubs                           an input flow-pipe located at left 10
Color                           inherited
Icon description                width 20; height 20;
                                red:
                                     outline (0, 0) (0, 20) (20, 20) (20,
                                     0);
                                     lines (1, 0) (19, 20);
                                     lines (1, 20) (19, 0)
```

MIMO_INPUT, an object-definition

Notes	OK
User restrictions	none
Class	mimo_input
Superior class	object
Attributes specific to class	u is given by a plant_data;
	du is given by a plant_data;
	u_ll is given by a gpc_tunig_knob;
	u_hl is given by a gpc_tunig_knob;
	du_ll is given by a gpc_tunig_knob;
	du_hl is given by a gpc_tunig_knob;
	lambda is given by a gpc_tunig_knob;
	max_var_of_u is given by a quantitative-parameter
Capabilities and restrictions	none
Change	none
Menu option	a final menu choice
Inherited attributes	none
Default settings	none
Stubs	an output flow-pipe located at right 10
Color	inherited
Icon description	width 20; height 20;
	blue:
	outline (0, 0) (0, 20) (20, 20) (20, 0);
	lines (1, 0) (20, 20);
	lines (1, 20) (20, 0)

PLANT_DATA, an object-definition

Notes	OK
User restrictions	none
Class	plant_data
Superior class	sensor
Attributes specific to class	purpose is "SENSOR";
	varname
Capabilities and restrictions	gsi-data-service, gsi-message-service
Change	none
Menu option	a final menu choice
Inherited attributes	none
Default settings	validity interval: 2 seconds;
	default update interval: 2 seconds;
	history keeping spec: keep history with maximum number of data points = 50
Stubs	none inherited
Color	inherited
Icon description	width 74; height 62;
	dark-gray:
	lines (57, 42) (47, 29) (23, 29) (5, 40);
	brown:
	filled polygon (10, 38) (10, 58) (54, 58) (54, 38);
	red:
	filled polygon (30, 18) (30, 29) (27, 29) (27, 18);
	dark-gray:
	filled circle (26, 14) (29, 11) (32, 14);

filled circle (30, 12) (35, 7) (40, 12);
filled circle (38, 12) (48, 2) (58, 12)

GPC_FLAG, an object-definition

Notes	OK
User restrictions	none
Class	gpc_flag
Superior class	sensor
Attributes specific to class	purpose is "FLAG"; varname
Capabilities and restrictions	gsi-data-service, gsi-message-service
Change	none
Menu option	a final menu choice
Inherited attributes	none
Default settings	validity interval: indefinite; default update interval: none
Stubs	none inherited
Color	inherited

Icon description	width 100; height 92;
	lines (17, 21) arc (29, 13) (42, 21);
	lines (42, 21) arc (53, 28) (63, 20);
	lines (19, 35) arc (30, 28) (40, 35);
	lines (40, 35) arc (53, 42) (63, 37);
	lines (63, 20) (63, 37);
	filled rectangle (14, 15) (19, 67)

GPC_TUNIG_KNOB, an object-definition

Notes	OK
User restrictions	none
Class	gpc_tunig_knob
Superior class	sensor
Attributes specific to class	purpose is "KNOB"; varname
Capabilities and restrictions	gsi-data-service, gsi-message-service
Change	none
Menu option	a final menu choice
Inherited attributes	none

Default settings	validity interval: indefinite;
	default update interval: none;
	history keeping spec: keep history with maximum number of data points = 50

Stubs	none inherited
Color	inherited

Icon description	width 65; height 56;
	filled circle (14, 33) (37, 10) (60, 33);
	filled polygon (19, 24) (26, 15) (13, 4) (6, 13)

```
                          GSI-INTERFACE, an object-definition
Notes                               OK
User restrictions                   none
Class                               gsi-interface
Superior class                      object
Attributes specific to class        none
Capabilities and restrictions       gsi-interface-configuration
Change                              none
Menu option                         a final menu choice
Inherited attributes                none
Default settings                    none
Stubs                               none inherited
Color                               inherited
Icon description                    width 112; height 98;
                                        outline (82, 12) (82, 82) (100, 82)
                                        (100, 12);
                                        outline (14, 10) (14, 81) (32, 81)
                                        (32, 10);
                                        lines (32, 30) (69, 30);
                                        lines (32, 20) (69, 20);
                                        lines (69, 20) (69, 14);
                                        lines (69, 30) (69, 37);
                                        lines (69, 14) (82, 25);
                                        lines (69, 37) (82, 27);
                                        lines (44, 60) (82, 60);
                                        lines (82, 72) (43, 72);
                                        lines (44, 60) (44, 53);
                                        lines (44, 72) (44, 80);
                                        lines (32, 66) (44, 52);
                                        lines (32, 69) (44, 80)
```

```
                          DELTA_TR, a quantitative-variable
Options                             do  not  forward  chain,  breadth  first
                                    backward chain
Notes                               OK
User restrictions                   none
Names                               DELTA_TR
Tracing and breakpoints             default
Data type                           quantity
Initial value                       none
Last recorded value                 no value
History keeping spec                do not keep history
Validity interval                   supplied
Formula
Simulation details                  no simulation formula yet
Initial value for simulation        default
Data server                         inference engine
Default update interval             none
```

DELTA_N2, a quantitative-variable

Options	do not forward chain, breadth first backward chain
Notes	OK
User restrictions	none
Names	DELTA_N2
Tracing and breakpoints	default
Data type	quantity
Initial value	none
Last recorded value	no value
History keeping spec	do not keep history
Validity interval	supplied
Formula	
Simulation details	no simulation formula yet
Initial value for simulation	default
Data server	inference engine
Default update interval	none

2. Procedures

```
                GO_GPC, a procedure
Notes                           OK
User restrictions               none
Tracing and breakpoints         default
Default procedure priority      1
go_gpc (PLANT:class mimo_plant )
stripper:class mimo_plant;
begin
if (the name of PLANT is stripper ) then
       call MIMOGPC() across GSI-1;
if (the name of PLANT is stripper_2 ) then
       call MIMOGPC() across GSI-2;
 if (the name of PLANT is stripper_3 ) then
       call MIMOGPC() across GSI-3;

end


                STARTUP, a procedure
Notes                           OK
User restrictions               none
Tracing and breakpoints         default
Default procedure priority      1
startup (PLANT:class mimo_plant , TYPE: text)
stripper,stripper_2,stripper_3:class mimo_plant;
INP:class mimo_input;
OUT: class mimo_output;
COUNTER: quantity;
begin
inform the operator for the next 50 seconds that "
Startup:
Startup of [the name of PLANT] initiated";
if(the name of PLANT is stripper or
   the name of PLANT is stripper_2 or
   the name of PLANT is stripper_3) then
begin
   for OUT=each mimo_output connected to PLANT
      do
      conclude that the servo_status of OUT = 1
         with collection time the current time - 3
         seconds
      end;
   set the nu of PLANT to 1;
   set the n1 of PLANT to 1;
   set the n2 of PLANT to 10;
   case (TYPE) of
   "SLS": begin
            for INP=each mimo_input connected to
               PLANT do
            set the u_hl of INP to 4.0;
            set the u_ll of INP to -4.0;
            set the du_hl of INP to 1.0;
            set the du_ll of INP to -1.0;
            end;
            end;
     "SHELL":begin
            for INP=each mimo_input connected to
               PLANT do
            set the u_hl of INP to 0.5;
            set the u_ll of INP to -0.5;
            set the du_hl of INP to 0.2;
            set the du_ll of INP to -0.2;
            end;
            end;
    "WOOD":begin
            for INP=each mimo_input connected to
               PLANT do
            set the u_hl of INP to 10.0;
            set the u_ll of INP to -10.0;
            set the du_hl of INP to 1.0;
            set the du_ll of INP to -1.0;
            end;
            end;
     end;
     for COUNTER=0. to 50. by 2.
     do
 if(the name of PLANT is stripper) then
 begin
     do in parallel
       conclude that ysp1prime = startup_output1
          (COUNTER);
       conclude that ysp1prime_old =
          startup_output1 (COUNTER);
       conclude that ysp2prime = startup_output2
          (COUNTER);
       conclude that ysp2prime_old =
          startup_output2 (COUNTER);
       set the ysp of the mimo_output connected at
          the outlet-port-1 of PLANT to
          startup_output1 (COUNTER);
       set the ysp of the mimo_output connected at
          the outlet-port-2 of PLANT to
          startup_output2 (COUNTER);
      end;
 end;
 if(the name of PLANT is stripper_2) then
 begin
     do in parallel
       conclude that ysp1prime_2 = startup_output1
          (COUNTER)/25.0;
       conclude that ysp1prime_old_2 =
          startup_output1 (COUNTER)/25.0;
       conclude that ysp2prime_2 = startup_output2
          (COUNTER)/25.0;
       conclude that ysp2prime_old_2 =
          startup_output2 (COUNTER)/25.0;
       set the ysp of the mimo_output connected at
          the outlet-port-1 of PLANT to
          startup_output1 (COUNTER)/25.0;
       set the ysp of the mimo_output connected at
          the outlet-port-2 of PLANT to
          startup_output2 (COUNTER)/25.0;
      end;
 end;
 if(the name of PLANT is stripper_3) then
 begin
     do in parallel
       conclude that ysp1prime_3 = startup_output1
          (COUNTER)/2.5;
       conclude that ysp1prime_old_3 =
          startup_output1 (COUNTER)/2.5;
       conclude that ysp2prime_3 = startup_output2
```

```
          (COUNTER)/2.5;
       conclude that ysp2prime_old_3 =
          startup_output2 (COUNTER)/2.5;
       set the ysp of the mimo_output connected at
          the outlet-port-1 of PLANT to
          startup_output1 (COUNTER)/2.5;
       set the ysp of the mimo_output connected at
          the outlet-port-2 of PLANT to
          startup_output2 (COUNTER)/2.5;
     end;
end;
           wait for 2 seconds;
     end;
    inform the operator for the next 15 seconds
       that"
Startup:
Waiting until all channels reach their
regulation tolerances for [the name of PLANT]";
     wait until for every mimo_output OUT
        connected to PLANT (
     abs(the y of OUT) <=
  (1.+the regulation_tolerance of OUT)*
    abs(the ysp of OUT) and
     abs(the y of OUT) >=
  (1. - the regulation_tolerance of OUT)*
     abs(the ysp of OUT)
   ) checking every 2 seconds;
     for OUT=each mimo_output connected to PLANT
     do
     conclude that the servo_status of OUT = 0
        with collection time the current time - 3
        seconds
     end;
     inform the operator for the next 10 seconds
        that"
Startup:
Startup of [the name of PLANT] completed";
end
else inform the operator for the next 15 seconds
   that "
WARNING: startup of [the name of PLANT]
unknown; EXITING procedure."  ;
end


          OUTPUT_WEIGHTING, a procedure
Notes                        OK
User restrictions            none
Tracing and breakpoints      default
Default procedure priority   1
output_weighting (PLANT: class mimo_plant)
max_dev: quantity;
OUT: class mimo_output;
stripper,stripper_2,stripper_3:class mimo_plant;
begin
max_dev = (the maximum over each mimo_output OUTP
   connected to PLANT of (the standard deviation
   of the y of OUTP during the last 20 seconds)
      ) ;
if(the name of PLANT is stripper or
the name of PLANT is stripper_3) then
begin
```

```
      for OUT=each mimo_output connected to PLANT
      do
      set the outputweight of OUT to (
      (the standard deviation of the y of OUT
         during the last 20 seconds  + 0.1)
         /(max_dev+0.1)
      )
      end;
      return;
end;
if(the name of PLANT is stripper_2) then
begin
      for OUT=each mimo_output connected to PLANT
      do
      set the outputweight of OUT to (
      (the standard deviation of the y of OUT
         during the last 20 seconds  + 0.005)
         /(max_dev+0.005)
      )
      end;
      return;
end;
inform the operator for the next 15 seconds that "
WARNING: output_weighting( [the name of PLANT]  )
unknown; EXITING procedure."  ;
end



           OBSERVE_2X2_PLANT, a procedure
Notes                           OK
User restrictions               none
Tracing and breakpoints         default
Default procedure priority      1
observe_2x2_plant(PLANT:class 2by2-plant)
OUTP:class mimo_output;
estimated_tr:quantity;
OUTP1:class mimo_output;
OUTP2:class mimo_output;
begin
allow other processing;
if( for every mimo_output OUTP connected to PLANT (
     the servo_status of OUTP = 0))
then return;

if( for every mimo_output OUTP connected to PLANT (
     the servo_status of OUTP - the value of the
        servo_status of OUTP  as of 2 seconds ago =
        0))
then return;
OUTP1 =(the mimo_output
        connected at the outlet-port-1 of PLANT);
OUTP2 =(the mimo_output
        connected at the outlet-port-2 of PLANT);
if( the servo_status of OUTP1 =0 and
     the servo_status of OUTP2 =1 and (
    (the servo_status of OUTP2 - the value of the
       servo_status of OUTP2 as of 2 seconds ago )
       =1))
then begin
       start find_tr(OUTP2,OUTP1);
      return;
      end;

if( the servo_status of OUTP1 =1 and
    the servo_status of OUTP2 =0 and (
   (the servo_status of OUTP1 - the value of the
      servo_status of OUTP1 as of 2 seconds ago )
      =1))
then begin
      start find_tr(OUTP1,OUTP2);
      return;
      end;

if ((the servo_status of OUTP1 - the value of the
   servo_status of OUTP1 as of 2 seconds ago ) =1)
then start servo_wait_cycle(OUTP1);

if ((the servo_status of OUTP2 - the value of the
   servo_status of OUTP2 as of 2 seconds ago ) =1)
then start servo_wait_cycle(OUTP2);

end
```

```
          SERVO_WAIT_CYCLE, a procedure
Notes                         OK
User restrictions             none
Tracing and breakpoints       default
Default procedure priority    1
servo_wait_cycle(OUTP:class mimo_output )
begin
allow other processing;
   inform the operator for the next 50 seconds
      that "
servo_wait_cycle:
Entering with [the name of OUTP]
connected to [the name of the
mimo_plant connected to OUTP]";

   wait for 50 seconds;
   conclude that the servo_status of
   OUTP = (the servo_status of OUTP - 1);

   inform the operator for the next 10 seconds
      that "
servo_wait_cycle:
Exiting with [the name of OUTP]
 connected to [the name of the
mimo_plant connected to OUTP]";

end
```

```
                  FIND_TR, a procedure
Notes                           OK
User restrictions               none
Tracing and breakpoints         default
Default procedure priority      1
find_tr(OUTP1:class mimo_output,
         OUTP0:class mimo_output)
old_ysp,new_ysp:quantity;
watch_for_y,loop_counter,tr:quantity;
current_y,current_y_zm1:quantity;
current_y_zm2:quantity;
delta_ysp,slope:quantity;
begin
   allow other processing;
   inform the operator for the next 50 seconds
      that "
   find_tr:
   Entering with [the name of OUTP1]
   connected to [the name of the mimo_plant
   connected to OUTP1]";
repeat
   collect data
   new_ysp= the ysp of OUTP1;
   old_ysp =the value of the ysp of OUTP1
                as of 2 second ago;  end;
   delta_ysp=new_ysp - old_ysp;
   exit if(delta_ysp /= 0.0);
    allow other processing;
end;
   watch_for_y=old_ysp+0.8*delta_ysp;
collect data
current_y=the y of OUTP1;
current_y_zm1=the value of the y of OUTP1 as of 2
   seconds ago;
current_y_zm2=the value of the y of OUTP1 as of 4
   seconds ago;
end;
   for loop_counter=0 to 50 by 2
   do
     wait for 2 seconds;
     current_y_zm2=current_y_zm1;
     current_y_zm1=current_y;
     collect data
     current_y= the y of OUTP1;
     end;
     slope=(current_y - current_y_zm2)/4.0;
 inform the operator for the next 6 seconds that "
 loop_counter= [loop_counter]
 current_y=[current_y]
 current_y_zm1=[current_y_zm1]
 current_y_zm2=[current_y_zm2]
 watch_for_y=[watch_for_y]
 slope=[slope]";
      exit if(( if (delta_ysp > 0.0) then
         (current_y > watch_for_y) else
          (current_y < watch_for_y)));
    end;
    tr=loop_counter + (watch_for_y  -
       current_y_zm1)/slope;
    wait for (50 - loop_counter);
   if(loop_counter >= 50) then
    begin
   inform the operator for the next 10 seconds
      that "
   find_tr:
   Exiting with [the name of OUTP1]
   connected to [the name of the mimo_plant
   connected to OUTP1]
   UNABLE to find tr, window expired";
   conclude that the servo_status of OUTP1 = (the
      servo_status of OUTP1 - 1);
  return;
  end;
   if(the servo_status of OUTP1 =1 and
      the servo_status of OUTP0=0) then
      conclude that the observed_tr of OUTP1 = tr;


   conclude that the servo_status of OUTP1 = (the
      servo_status of OUTP1 - 1);
   inform the operator for the next 10 seconds
      that "
   find_tr:
   Exiting with [the name of OUTP1]
   connected to [the name of the mimo_plant
   connected to OUTP1] ";
end
```

```
                 SHUTDOWN, a procedure
Notes                             OK
User restrictions                 none
Tracing and breakpoints           default
Default procedure priority        1
shutdown (PLANT:class mimo_plant , TYPE: text)
stripper,stripper_2,stripper_3:class mimo_plant;
INP:class mimo_input;
OUT: class mimo_output;
COUNTER: quantity;
begin
inform the operator for the next 50 seconds that "
Shutdown:
Shutdown of [the name of PLANT] initiated";
if(the name of PLANT is stripper or
the name of PLANT is stripper_2 or
the name of PLANT is stripper_3) then
begin
   for OUT=each mimo_output connected to PLANT
       do
  conclude that the servo_status of OUT = 1  with
     collection time the current time - 3 seconds
       end;
   set the nu of PLANT to 1;
   set the n1 of PLANT to 1;
   set the n2 of PLANT to 10;
   case (TYPE) of
   "SLS": begin
            for INP=each mimo_input connected to
               PLANT do
            set the u_hl of INP to 4.0;
            set the u_ll of INP to -4.0;
            set the du_hl of INP to 0.1;
            set the du_ll of INP to -0.1;
```

```
             end;
             end;
     "SHELL":begin
             for INP=each mimo_input connected to
                PLANT do
             set the u_hl of INP to 0.5;
             set the u_ll of INP to -0.5;
             set the du_hl of INP to 0.02;
             set the du_ll of INP to -0.02;
             end;
             end;
     "WOOD":begin
             for INP=each mimo_input connected to
                PLANT do
             set the u_hl of INP to 10.0;
             set the u_ll of INP to -10.0;
             set the du_hl of INP to 0.1;
             set the du_ll of INP to -0.1;
             end;
             end;
      end;
if(the name of PLANT is stripper) then
begin
do in parallel
       conclude that ysp1prime =0.0;
       conclude that ysp1prime_old =0.0;
       conclude that ysp2prime =0.0;
       conclude that ysp2prime_old =0.0;
       set the ysp of the mimo_output connected at
          the outlet-port-1 of PLANT to 0.0;
       set the ysp of the mimo_output connected at
          the outlet-port-2 of PLANT to 0.0;
end;
end;
if(the name of PLANT is stripper_2) then
begin
do in parallel
       conclude that ysp1prime_2 =0.0;
       conclude that ysp1prime_old_2 =0.0;
       conclude that ysp2prime_2 =0.0;
       conclude that ysp2prime_old_2 =0.0;
       set the ysp of the mimo_output connected at
          the outlet-port-1 of PLANT to 0.0;
       set the ysp of the mimo_output connected at
          the outlet-port-2 of PLANT to 0.0;
end;
end;
if(the name of PLANT is stripper_3) then
begin
do in parallel
       conclude that ysp1prime_3 =0.0;
       conclude that ysp1prime_old_3 =0.0;
       conclude that ysp2prime_3 =0.0;
       conclude that ysp2prime_old_3 =0.0;
       set the ysp of the mimo_output connected at
          the outlet-port-1 of PLANT to 0.0;
       set the ysp of the mimo_output connected at
          the outlet-port-2 of PLANT to 0.0;
end;
end;
     wait for 50 seconds;
inform the operator for the next 15 seconds that"
```

```
Shutdown:
Waiting until all channels reach below
abs(regulation tolerances) for [the name of
   PLANT]";
     wait until for every mimo_output OUT
        connected to PLANT (
     the y of OUT <=
  the ysp of OUT + the regulation_tolerance of OUT
     and
  the y of OUT >=
  the ysp of OUT - the regulation_tolerance of OUT
   ) checking every 2 seconds;
     for OUT=each mimo_output connected to PLANT
     do
 conclude that the servo_status of OUT = 0 with
    collection time the current time - 3 seconds
     end;
     inform the operator for the next 10 seconds
        that "
Shutdown of [the name of PLANT] completed";
end
else inform the operator for the next 15 seconds
   that "
WARNING: shutdown of [the name of PLANT]
unknown; EXITING procedure."  ;
end
```

```
                SWITCH_T, a procedure
Notes                           OK
User restrictions               none
Tracing and breakpoints         default
Default procedure priority      1
switch_T (PLANT: class mimo_plant,ORDER: quantity
   )
stripper,stripper_2,stripper_3:class mimo_plant;
begin
case (the name of PLANT) of
stripper:  begin
  conclude that t-filter-order = ORDER;
            end;
stripper_2: begin
  conclude that t-filter-order_2 = ORDER;
            end;
stripper_3: begin
  conclude that t-filter-order_3 = ORDER;
            end;
end
end
```

3. Global Supervision Rules

```
Options                                       a rule
                                              invocable via backward
                                              chaining, invocable via forward
                                              chaining, may cause data
                                              seeking, may cause forward
                                              chaining
Notes                                         OK
User restrictions                             none
Names                                         none
Tracing and breakpoints                       default
for any mimo_plant PLANT
for any mimo_output OUT connected to PLANT
for any mimo_input INP connected to PLANT

unconditionally
 set the y_hl of OUT to 1.e+06 and
 set the y_ll of OUT to -1.e+06 and
 set the u_hl of INP to 1.e+06 and
 set the u_ll of INP to -1.e+06 and
 set the du_hl  of INP to 1.e+06 and
 set the du_ll of INP to -1.e+06
Scan interval                                 none
Focal classes                                 mimo_plant
Focal objects                                 none
Categories                                    relax-all-constraints
Rule priority                                 6
Depth first backward chaining precedence      1
Timeout for rule completion                   use default


Options                                       a rule
                                              invocable via backward
                                              chaining, invocable via forward
                                              chaining, may cause data
                                              seeking, may
                                              cause forward chaining
Notes                                         OK
User restrictions                             none
Names                                         none
Tracing and breakpoints                       default
initially
for any mimo_plant PLANT
for any mimo_output OUTP connected to PLANT
for any mimo_input INP connected to PLANT
unconditionally

set the outputweight of OUTP to 1.0
and set the y_hl of OUTP to 1.e+06
and set the y_ll of OUTP to -1.e+06
and conclude that  the servo_status of OUTP=0
and set the lambda of INP to 1.e-06
and set the u_hl of INP to 1.e+06
and set the u_ll of INP to -1.e+06
and set the du_hl of INP to 1.e+06
and set the du_ll of INP to -1.e+06
Scan interval                                 none
Focal classes                                 none
Focal objects                                 none
Categories                                    none
Rule priority                                 6
```

```
Depth first backward chaining precedence  1
Timeout for rule completion               use default


                                           a rule
Options                                   not  invocable  via  backward
                                          chaining,  not  invocable  via
                                          forward chaining, may not cause
                                          data  seeking,  may  not  cause
                                          forward chaining
Notes                                     OK
User restrictions                         none
Names                                     none
Tracing and breakpoints                   default
for any mimo_plant PLANT
whenever the tau_over_ts of PLANT fails to receive a value then conclude
that the
   tau_over_ts of PLANT  = the value of the tau_over_ts of PLANT as of 1
second ago
Scan interval                             none
Focal classes                             none
Focal objects                             none
Categories                                none
Rule priority                             6
Depth first backward chaining precedence  1
Timeout for rule completion               use default


                                           a rule
Options                                   not  invocable  via  backward
                                          chaining, invocable via forward
                                          chaining, may cause data
                                          seeking,  may  cause  forward
                                          chaining
Notes                                     OK
User restrictions                         none
Names                                     none
Tracing and breakpoints                   default
if(the value of the gsi-interface-status
 of gsi-1 < 0)
then inform the operator for the next 30 seconds that"
*************************************************

ERROR: Interface GSI-1 has
lost its communication link.

Process Control switched to MANUAL.

(You must kill and restart the external
application and then click on ''GO'')

**************************************************"
and deactivate the subworkspace of instances
Scan interval                             none
Focal classes                             none
Focal objects                             none
Categories                                none
Rule priority                             1
Depth first backward chaining precedence  1
Timeout for rule completion               use default
```

```
                                           a rule
Options                                   not  invocable  via  backward
                                          chaining,  not  invocable  via
                                          forward  chaining,  may  cause
                                          data seeking,
                                          may cause forward chaining
Notes                                     OK
User restrictions                         none
Names                                     none
Tracing and breakpoints                   default
for any mimo_plant PLANT
for any mimo_output OUTP connected to PLANT
whenever the n2 of PLANT receives a value or the nl of PLANT receives a
value
or the nu of PLANT receives a value
and when (the n2 of PLANT /= the value of the n2 of PLANT as of 2 seconds
ago or
the nl of PLANT /= the value of the nl of PLANT as of 2 seconds ago or
 the NU of PLANT /= the value of the NU of PLANT as of 2 seconds ago)
then
conclude that the observed_tr of OUTP has no current value
Scan interval                             none
Focal classes                             none
Focal objects                             none
Categories                                none
Rule priority                             6
Depth first backward chaining precedence  1
Timeout for rule completion               use default



                                           a rule
Options                                   not  invocable  via  backward
                                          chaining, invocable via forward
                                          chaining,   may   cause   data
                                          seeking,  may  cause  forward
                                          chaining
Notes                                     OK
User restrictions                         none
Names                                     none
Tracing and breakpoints                   default
if(the value of the gsi-interface-status
 of gsi-2 < 0)
then inform the operator for the next 30 seconds that"
***********************************************

ERROR: Interface GSI-2 has
lost its communication link.

Process Control switched to MANUAL.

(You must kill and restart the external
application and then click on ''GO'')

***********************************************"
and deactivate the subworkspace of instances-2
Scan interval                             none
Focal classes                             none
Focal objects                             none
Categories                                none
Rule priority                             1
```

```
Depth first backward chaining precedence  1
Timeout for rule completion               use default


                                             a rule
Options                                   not  invocable  via  backward
                                          chaining, invocable via forward
                                          chaining,   may   cause   data
                                          seeking,  may  cause  forward
                                          chaining
Notes                                     OK
User restrictions                         none
Names                                     none
Tracing and breakpoints                   default
if(the value of the gsi-interface-status
 of gsi-3 < 0)
then inform the operator for the next 30 seconds that"
***********************************************

ERROR: Interface GSI-3 has
lost its communication link.

Process Control switched to MANUAL.

(You must kill and restart the external
application and then click on ''GO'')

***********************************************"
and deactivate the subworkspace of instances-3
Scan interval                             none
Focal classes                             none
Focal objects                             none
Categories                                none
Rule priority                             1
Depth first backward chaining precedence  1
Timeout for rule completion               use default
```

4. Regulatory Supervision rules

```
                                               a rule
Options                                        not invocable via backward
                                               chaining, not invocable via
                                               forward chaining, may not cause
                                               data seeking, may not cause
                                               forward chaining
Notes                                          OK
User restrictions                              none
Names                                          none
Tracing and breakpoints                        default
for any mimo_plant PLANT
whenever the tau_over_ts of PLANT receives a value
and when (the current value of the tau_over_ts of PLANT /= the value of
the tau_over_ts of
   PLANT as of 1 second ago
and
(the name of PLANT is stripper and flagtprime =1 and t-filter-order = 1
or the name of PLANT is stripper_2 and flagtprime_2 =1 and
t-filter-order_2 = 1
or the name of PLANT is stripper_3 and flagtprime_3 =1 and
t-filter-order_3 = 1))

 then set the t_filter_a1 of PLANT to - (exp(-(the tau_over_ts of
PLANT)^-1)) and set the
    t_filter_a2 of PLANT to 0.0
and inform the operator for the next 10 seconds that "
Regulation Supervisor:
Updating the first order T filter
 for [the name of PLANT]"
Scan interval                                  none
Focal classes                                  none
Focal objects                                  none
Categories                                     none
Rule priority                                  6
Depth first backward chaining precedence       1
Timeout for rule completion                    use default
```

```
                                               a rule
Options                                        not invocable via backward
                                               chaining, not invocable via
                                               forward chaining, may not cause
                                               data seeking, may not cause
                                               forward chaining
Notes                                          OK
User restrictions                              none
Names                                          none
Tracing and breakpoints                        default
for any mimo_plant PLANT
whenever the tau_over_ts of PLANT receives a value
 and when (the current value of the tau_over_ts of PLANT /= the value of
the tau_over_ts of
    PLANT as of 1 second ago
and
((the name of PLANT is stripper and flagtprime =1 and t-filter-order = 2)
or (the name of PLANT is stripper_2 and flagtprime_2 =1 and
t-filter-order_2 = 2)
or  (the  name  of  PLANT  is  stripper_3  and  flagtprime_3  =1  and
t-filter-order_3 = 2)))
 then set the t_filter_a1 of PLANT to -2.*(exp(-(the tau_over_ts of
PLANT)^-1)) and set the
   t_filter_a2 of PLANT to (exp(-(the tau_over_ts of PLANT)^-1))^2
and inform the operator for the next  10 seconds that "
Regulation Supervisor:
 Updating the second order T filter
 for [the name of PLANT]"
Scan interval                              none
Focal classes                              none
Focal objects                              none
Categories                                 none
Rule priority                              6
Depth first backward chaining precedence   1
Timeout for rule completion                use default



                                            a rule
Options                                    not  invocable  via  backward
                                           chaining,  not  invocable  via
                                           forward chaining, may not cause
                                           data  seeking,  may  not  cause
                                           forward chaining
Notes                                      OK
User restrictions                          none
Names                                      none
Tracing and breakpoints                    default
for any mimo_plant PLANT
whenever the tau_over_ts of PLANT receives a value
and when  (((the  name  of  PLANT  is  stripper  and  flagtprime  =1  and
t-filter-order = 1)or( the
   name of PLANT is stripper_2 and flagtprime_2 =1 and t-filter-order_2 =
1)or (the name of
   PLANT is stripper_3 and flagtprime_3 =1 and t-filter-order_3 = 1))and
(the tau_over_ts of
   PLANT =19.496) and (the standard deviation of the tau_over_ts of PLANT
during the last 2
   minutes <= 2.e-3))
 then conclude that the tau_over_ts of PLANT=4.481
and start switch_T (PLANT,2)
and inform the operator for the next  10 seconds that "
Regulation Supervisor:
Switching to second order T filter for [the name of PLANT]"
Scan interval                              none
Focal classes                              none
Focal objects                              none
Categories                                 none
Rule priority                              6
Depth first backward chaining precedence   1
Timeout for rule completion                use default
```

```
                                              a rule
Options                                       not invocable via backward
                                              chaining, not invocable via
                                              forward chaining, may not cause
                                              data seeking, may not cause
                                              forward chaining
Notes                                         OK
User restrictions                             none
Names                                         none
Tracing and breakpoints                       default
for any mimo_plant PLANT
whenever the tau_over_ts of PLANT receives a value
and when (((the name of PLANT is stripper and flagtprime =1 and
t-filter-order = 2)or( the
   name of PLANT is stripper_2 and flagtprime_2 =1 and t-filter-order_2 =
2)or (the name of
   PLANT is stripper_3 and flagtprime_3 =1 and t-filter-order_3 = 2))
and (the tau_over_ts of PLANT =1.442) and (the standard deviation of the
tau_over_ts of
   PLANT during the last 2 minutes <= 2.e-3)
)
 then conclude that the tau_over_ts of PLANT=4.481
and start switch_T (PLANT,1)
and inform the operator for the next  10 seconds that "
Regulation Supervisor:
Switching to first order T filter for [the name of PLANT]"
Scan interval                                 none
Focal classes                                 none
Focal objects                                 none
Categories                                    none
Rule priority                                 6
Depth first backward chaining precedence      1
Timeout for rule completion                   use default
```

```
                                              a rule
Options                                       invocable via backward
                                              chaining, invocable via
                                              forward chaining, may cause
                                              data seeking, may
                                              cause forward chaining
Notes                                         OK
User restrictions                             none
Names                                         none
Tracing and breakpoints                       default
for any mimo_plant PLANT
if(for every mimo_output OUTP connected to PLANT
  (the servo_status of OUTP = 0))
then
start OUTPUT_WEIGHTING(PLANT) and inform the operator for the next 10
seconds that "
Regulation Supervisor:
Updating output weighting for [the name of PLANT]"
Scan interval                                 2 minutes
Focal classes                                 none
Focal objects                                 none
Categories                                    none
Rule priority                                 6
Depth first backward chaining precedence      1
Timeout for rule completion                   use default
```

```
                                              a rule
Options                                       not   invocable   via   backward
                                              chaining,  not  invocable  via
                                              forward chaining, may not cause
                                              data  seeking,  may  not  cause
                                              forward chaining
Notes                                         OK
User restrictions                             none
Names                                         none
Tracing and breakpoints                       default
for any mimo_plant PLANT
whenever the tau_over_ts of PLANT receives a value
and
when ((the name of PLANT is stripper and flagtprime = 0)or(
 the name of PLANT is stripper_2 and flagtprime_2 = 0)or(
the name of PLANT is stripper_3 and flagtprime_3 = 0))
 then inform the operator for the next  10 seconds that "
Regulation Supervisor:
Operator selected MGPC algorithm
WITHOUT T filter; hence, the
updating the T filter for [the name of PLANT]
has been cancelled"
Scan interval                                 none
Focal classes                                 none
Focal objects                                 none
Categories                                    none
Rule priority                                 6
Depth first backward chaining precedence      1
Timeout for rule completion                   use default


                                              a rule
Options                                       invocable    via    backward
                                              chaining,invocable via forward
                                              chaining,   may   cause   data
                                              seeking,  may  cause  forward
                                              chaining
Notes                                         OK
User restrictions                             none
Names                                         none
Tracing and breakpoints                       default
for any mimo_plant PLANT
if(not(for every mimo_output OUTP connected to PLANT
  (the servo_status of OUTP = 0)))
then
 inform the operator for the next 10 seconds that "
Regulation Supervisor:
Setpoint change in progress for [the name of PLANT];
hence, suspending the updating of
 output weighting for [the name of PLANT]"
Scan interval                                 2 minutes
Focal classes                                 none
Focal objects                                 none
Categories                                    none
Rule priority                                 6
Depth first backward chaining precedence      1
Timeout for rule completion                   use default
```

```
                                              a rule
Options                                      not  invocable  via  backward
                                             chaining,  not  invocable  via
                                             forward chaining, may not cause
                                             data  seeking,  may  not  cause
                                             forward chaining
Notes                                        OK
User restrictions                            none
Names                                        none
Tracing and breakpoints                      default
for any mimo_plant PLANT
whenever the tau_over_ts of PLANT receives a value
and when (((the name of PLANT is stripper and flagtprime = 1)or (
 the name of PLANT is stripper_2 and flagtprime_2 = 1)or (
the name of PLANT is stripper_3 and flagtprime_3 = 1))and not(for every
mimo_output OUTP
   connected to PLANT
  (the servo_status of OUTP = 0)))
 then inform the operator for the next  10 seconds that "
Regulation Supervisor:
Setpoint change in progress for [the name of PLANT];
hence, suspending the updating of
T filter for [the name of PLANT]"
Scan interval                                none
Focal classes                                none
Focal objects                                none
Categories                                   none
Rule priority                                6
Depth first backward chaining precedence     1
Timeout for rule completion                  use default
```

```
                                                 a rule
Options                                         not   invocable   via   backward
                                                chaining,
                                                not invocable
                                                via forward chaining, may not
                                                cause  data
                                                seeking, may not cause forward
                                                chaining
Notes                                           OK
User restrictions                               none
Names                                           none
Tracing and breakpoints                         default
for any mimo_plant PLANT
whenever the tau_over_ts of PLANT receives a value
and when  (((the  name  of  PLANT  is  stripper  and  flagtprime  =1  and
t-filter-order = 1)or( the
   name of PLANT is stripper_2 and flagtprime_2 =1 and t-filter-order_2 =
1)or (the name of
   PLANT is stripper_3 and flagtprime_3 =1 and t-filter-order_3 = 1))
and (the tau_over_ts of PLANT =1.442) and (the standard deviation of the
tau_over_ts of
   PLANT during the last 2 minutes <= 2.e-3)
)
 then inform the operator for the next  10 seconds that "
Regulation Supervisor:
WARNING:  Applying lightest filtering
to [the name of PLANT], the specs on
the max_var on U may be too large"
Scan interval                                    none
Focal classes                                    none
Focal objects                                    none
Categories                                       none
Rule priority                                    6
Depth first backward chaining precedence         1
Timeout for rule completion                      use default
```

```
                                             a rule
Options                                     not  invocable  via  backward
                                            chaining,
                                            not invocable
                                            via forward chaining, may not
                                            cause data
                                            seeking, may not cause forward
                                            chaining
Notes                                       OK
User restrictions                           none
Names                                       none
Tracing and breakpoints                     default
for any mimo_plant PLANT
whenever the tau_over_ts of PLANT receives a value
and when  (((the  name  of  PLANT  is  stripper  and  flagtprime  =1  and
t-filter-order = 2)or( the
   name of PLANT is stripper_2 and flagtprime_2 =1 and t-filter-order_2 =
2)or (the name of
   PLANT is stripper_3 and flagtprime_3 =1 and t-filter-order_3 = 2))and
(the tau_over_ts of
   PLANT =19.496) and (the standard deviation of the tau_over_ts of PLANT
during the last 2
   minutes <= 2.e-3))
 then inform the operator for the next  10 seconds that "
Regulation Supervisor:
WARNING:  Applying heaviest filtering
to [the name of PLANT], the specs on
the max_var on U may be too small"
Scan interval                               none
Focal classes                               none
Focal objects                               none
Categories                                  none
Rule priority                               6
Depth first backward chaining precedence   1
Timeout for rule completion                 use default
```

```
                  a generic formula
Notes                OK
User restrictions    none
Names                none
let the tau_over_ts of any mimo_plant PLANT =
(
if (for every mimo_output OUTP connected to PLANT
  (the servo_status of OUTP = 0))
then (
min(19.496,max(1.442,
(the value of the tau_over_ts of PLANT as of 1
   second ago)*
(1. + 0.2*(
(the maximum over each mimo_input INP connected to
   PLANT of ((the standard deviation of the u of
   INP during the last 20 seconds)^2) -
the minimum over each mimo_input INPT connected to
   PLANT of (the max_var_of_u of INPT)
)/ the minimum over each mimo_input INT connected
   to PLANT of (the max_var_of_u of INT)
))))
) else (
the value of the tau_over_ts of PLANT as of 1
   second ago)
)
```

5. Servo Supervision rules for adjusting y_{min}, y_{max}

```
                                              a rule
Options                                     invocable via backward chaining,
                                              invocable via
                                             forward chaining, may cause data
                                              seeking, may
                                              cause forward chaining
Notes                                         OK
User restrictions                             none
Names                                         none
Tracing and breakpoints                       default
initially
for any mimo_plant PLANT
for any mimo_output OUTP connected to PLANT
unconditionally invoke apply-output-constraints rules for OUTP
Scan interval                                 none
Focal classes                                 none
Focal objects                                 none
Categories                                    none
Rule priority                                 6
Depth first backward chaining precedence      1
Timeout for rule completion                   use default


                                              a rule
Options                                       not  invocable  via  backward
                                              chaining,
                                              not invocable
                                              via forward chaining, may cause
                                              data seeking,
                                              may cause forward chaining
Notes                                         OK
User restrictions                             none
Names                                         none
Tracing and breakpoints                       default
for any mimo_output  OUT
if ((the ysp of OUT = 0.0)
and (the servo_status of OUT=0))
then set the y_ll of OUT to ( -1.*the regulation_tolerance of OUT
)
and set the y_hl of OUT to (the regulation_tolerance of OUT
)
and inform the operator for the next 10 seconds that "
Servo Supervisor:
Setting the y_ll and y_hl of [the name of OUT]
connected to [the name of the mimo_plant connected to OUT] for
normal operation"
Scan interval                                 none
Focal classes                                 mimo_output
Focal objects                                 none
Categories                                    apply-output-constraints
Rule priority                                 6
Depth first backward chaining precedence      1
Timeout for rule completion                   use default
```

```
                                            a rule
Options                                    not   invocable   via   backward
                                           chaining,
                                           not invocable
                                           via forward chaining, may cause
                                           data seeking,
                                           may cause forward chaining
Notes                                      OK
User restrictions                          none
Names                                      none
Tracing and breakpoints                    default
for any mimo_output OUT
whenever the servo_status of OUT receives a value
and when (the servo_status of OUT = 0)
then invoke apply-output-constraints rules for OUT
Scan interval                              none
Focal classes                              none
Focal objects                              none
Categories                                 none
Rule priority                              6
Depth first backward chaining precedence   1
Timeout for rule completion                use default


                                            a rule
Options                                    not   invocable   via   backward
                                           chaining,
                                           not invocable
                                           via forward chaining, may cause
                                           data seeking,
                                           may cause forward chaining
Notes                                      OK
User restrictions                          none
Names                                      none
Tracing and breakpoints                    default
for any mimo_output OUT
if( (the ysp of OUT /= 0.0) and (the servo_status of OUT=0))
then set the y_ll of OUT to (if(the ysp of OUT > 0.0) then (
the ysp of OUT*(1. - the regulation_tolerance of OUT)) else(
the ysp of OUT*(1. + the regulation_tolerance of OUT))
)
and set the y_hl of OUT to (if(the ysp of OUT > 0.0) then (
the ysp of OUT*(1. + the regulation_tolerance of OUT)) else(
the ysp of OUT*(1. - the regulation_tolerance of OUT))
)
and inform the operator for the next 10 seconds that "
Servo Supervisor:
Setting the y_ll and y_hl of [the name of OUT]
connected to [the name of  the mimo_plant connected to OUT] for
normal operation"
Scan interval                              none
Focal classes                              mimo_output
Focal objects                              none
Categories                                 apply-output-constraints
Rule priority                              6
Depth first backward chaining precedence   1
Timeout for rule completion                use default
```

```
                                            a rule
Options                                    not  invocable  via  backward
                                           chaining,
                                           not invocable
                                           via forward chaining, may cause
                                           data seeking,
                                           may cause forward chaining
Notes                                      OK
User restrictions                          none
Names                                      none
Tracing and breakpoints                    default
for any mimo_output OUT
whenever the ysp of OUT receives a value
and when (the ysp of OUT < the value of the ysp of OUT as of 1 second ago)
then set the y_ll of OUT to (the ysp of OUT + (the ysp of OUT - the value
of the ysp of OUT
   as of 2 second ago)*
(the max_overshoot of OUT))
and
inform the operator for the next 10 seconds that "
Servo Supervisor:
Updating the y_ll of [the name of OUT]
connected to [the name of the mimo_plant connected to OUT]"
Scan interval                              none
Focal classes                              none
Focal objects                              none
Categories                                 none
Rule priority                              6
Depth first backward chaining precedence   1
Timeout for rule completion                use default
```

```
                                                a rule
Options                                        not  invocable  via  backward
                                               chaining,
                                               not invocable
                                               via forward chaining, may cause
                                               data seeking,
                                               may cause forward chaining
Notes                                          OK
User restrictions                              none
Names                                          none
Tracing and breakpoints                        default
for any mimo_output OUT
whenever the ysp of OUT receives a value
and when (the ysp of OUT > the value of the ysp of OUT as of 1 second ago)
then set the y_hl of OUT to (the ysp of OUT + (the ysp of OUT - the value
of the ysp of OUT
   as of 2 second ago)*
(the max_overshoot of OUT))
and
inform the operator for the next 10 seconds that "
Servo Supervisor:
Updating the y_hl of [the name of OUT]
connected to [the name of the mimo_plant connected to OUT]"
Scan interval                                  none
Focal classes                                  none
Focal objects                                  none
Categories                                     none
Rule priority                                  6
Depth first backward chaining precedence       1
Timeout for rule completion                    use default
```

6. **Servo Supervision rules for adjusting N_2**

```
                                                a rule
Options                                         not  invocable  via  backward
                                                chaining,
                                                not invocable
                                                via forward chaining, may cause
                                                data seeking,
                                                may cause forward chaining
Notes                                           OK
User restrictions                               none
Names                                           none
Tracing and breakpoints                         default
for any mimo_output OUTP
for any mimo_plant connected to OUTP
whenever the observed_tr of OUTP receives a value and when (
for every mimo_output OUT connected to the mimo_plant (the observed_tr of
OUT has a current
   value))
then
invoke n2-tuning rules for the mimo_plant
Scan interval                                   none
Focal classes                                   none
Focal objects                                   none
Categories                                      none
Rule priority                                   6
Depth first backward chaining precedence        1
Timeout for rule completion                     use default
```

```
                                                a rule
Options                                        invocable via backward chaining,
                                                invocable via
                                                forward chaining, may cause data
                                                seeking, may
                                                cause forward chaining
Notes                                           OK
User restrictions                               none
Names                                           none
Tracing and breakpoints                         default
for any mimo_plant PLANT
if (the nu of PLANT /=1)
then
inform the operator for the next 15 seconds that "
servo supervisor:
The rise time of [the name of PLANT] can
NOT be tuned as its NU /= 1 rather
it is currently set to [the nu of PLANT]"
Scan interval                                   none
Focal classes                                   mimo_plant
Focal objects                                   none
Categories                                      n2-tuning
Rule priority                                   6
Depth first backward chaining precedence        1
Timeout for rule completion                     use default
```

```
Options                                      a rule
                                            invocable via backward chaining,
                                              invocable via
                                              forward chaining, may cause data
                                              seeking, may
                                              cause forward chaining
Notes                                         OK
User restrictions                             none
Names                                         none
Tracing and breakpoints                       tracing message level 3 (trace
messages at every
                                                 step)
for any mimo_plant PLANT

if( the nu of PLANT =1 and
the n1 of PLANT <= the max_delay of PLANT + 1  and
delta_n2 /= 0)
then in order
set the n2 of PLANT to (
min(40,max(the n2 of PLANT + delta_n2,
               the max_delay of PLANT + 1)))
and
inform the operator for the next 15 seconds that "
servo supervisor:
Updating N2 of [the name of PLANT] and
note that N1 <= max_delay + 1"
and
conclude that delta_tr has no current value and
 conclude that delta_n2 has no current value
Scan interval                                none
Focal classes                                mimo_plant
Focal objects                                none
Categories                                   n2-tuning
Rule priority                                6
Depth first backward chaining precedence     1
Timeout for rule completion                  20 seconds


                                             a rule
Options                                     invocable via backward chaining,
                                              invocable via
                                              forward chaining, may cause data
                                              seeking, may
                                              cause forward chaining
Notes                                         OK
User restrictions                             none
Names                                         none
Tracing and breakpoints                       default
for any mimo_plant PLANT

if( the nu of PLANT =1 and
the n1 of PLANT > the max_delay of PLANT + 1  and
delta_n2 /= 0)
then in order
set the n2 of PLANT to (
min(40,max(the n2 of PLANT + delta_n2,
               the n1 of PLANT)))
and
 inform the operator for the next 15 seconds that "
servo supervisor:
Updating N2 of [the name of PLANT] and
```

```
note that N1 > max_delay + 1"
and
conclude that delta_tr has no current value and
 conclude that delta_n2 has no current value
Scan interval                                 none
Focal classes                                 mimo_plant
Focal objects                                 none
Categories                                    n2-tuning
Rule priority                                 6
Depth first backward chaining precedence      1
Timeout for rule completion                   use default


                                               a rule
Options                                       invocable    via    backward
                                              chaining, not
                                              invocable via
                                              forward  chaining,  may  cause
                                              data seeking, may
                                              cause forward chaining
Notes                                         OK
User restrictions                             none
Names                                         none
Tracing and breakpoints                       default
for any mimo_plant PLANT
if abs (delta_tr) < 1.0
then
conclude that delta_n2 = 0 with expiration the current time + 1 second
and conclude that delta_tr has no current value
and inform the operator for the next 15 seconds that"
servo supervisor:
Rise times for [the name of PLANT]
are ON SPEC"
Scan interval                                 none
Focal classes                                 none
Focal objects                                 none
Categories                                    none
Rule priority                                 6
Depth first backward chaining precedence      1
Timeout for rule completion                   use default


                                               a rule
Options                                       invocable    via    backward
                                              chaining, not
                                              invocable via
                                              forward
                                              chaining, may
                                              cause data
                                              seeking, may
                                              cause forward
                                              chaining
Notes                                         OK
User restrictions                             none
Names                                         none
Tracing and breakpoints                       default
for any mimo_plant PLANT
unconditionally conclude that
delta_tr = (the minimum over each mimo_output OUTP connected to PLANT of
(
    the desired_tr of OUTP - the observed_tr of OUTP))
```

```
Scan interval                              none
Focal classes                              none
Focal objects                              none
Categories                                 none
Rule priority                              6
Depth first backward chaining precedence   1
Timeout for rule completion                use default


                                            a rule
Options                                    invocable    via    backward
                                           chaining, not
                                           invocable via
                                           forward
                                           chaining, may
                                           cause data
                                           seeking, may
                                           cause forward
                                           chaining
Notes                                      OK
User restrictions                          none
Names                                      none
Tracing and breakpoints                    default
for any mimo_plant PLANT
if abs (delta_tr) >= 1.0
and abs (delta_tr) < 2.0
then in order conclude that
delta_n2 = ( if delta_tr >= 0.0 then 1 else -1)
and inform the operator for the next 15 seconds that"
servo supervisor:
Rise times for [the name of PLANT]
are slightly OFF SPEC updating
N2 by [the value of delta_n2]"
Scan interval                              none
Focal classes                              none
Focal objects                              none
Categories                                 none
Rule priority                              6
Depth first backward chaining precedence   1
Timeout for rule completion                use default



                                            a rule
Options                                    invocable via
                                           backward
                                           chaining, not
                                           invocable via
                                           forward
                                           chaining, may
                                           cause data
                                           seeking, may
                                           cause forward
                                           chaining
Notes                                      OK
User restrictions                          none
Names                                      none
Tracing and breakpoints                    default
for any mimo_plant PLANT
if abs (delta_tr) >= 2.0
and abs (delta_tr) < 4.0
```

```
then in order conclude that
delta_n2 = ( if delta_tr >= 0.0 then 2 else -2)
and inform the operator for the next 15 seconds that"
servo supervisor:
Rise times for [the name of PLANT]
are OFF SPEC updating
N2 by [the value of delta_n2]"
Scan interval                              none
Focal classes                              none
Focal objects                              none
Categories                                 none
Rule priority                              6
Depth first backward chaining precedence   1
Timeout for rule completion                use default

                                            a rule
Options                                    invocable    via    backward
                                           chaining, not
                                           invocable via
                                           forward
                                           chaining, may
                                           cause data
                                           seeking, may
                                           cause forward
                                           chaining
Notes                                      OK
User restrictions                          none
Names                                      none
Tracing and breakpoints                    default
for any mimo_plant PLANT
if abs (delta_tr) >= 4.0
then in order conclude that
delta_n2 = ( if delta_tr >= 0.0 then 5 else -5)
and inform the operator for the next 15 seconds that"
servo supervisor:
Rise times for [the name of PLANT]
are badly OFF SPEC updating
N2 by [the value of delta_n2]"
Scan interval                              none
Focal classes                              none
Focal objects                              none
Categories                                 none
Rule priority                              6
Depth first backward chaining precedence   1
Timeout for rule completion                use default

                                            a rule
Options                                    not  invocable  via  backward
                                           chaining, not invocable
                                           via forward
                                           chaining, may
                                           cause data
                                           seeking, may
                                           cause forward
                                           chaining
Notes                                      OK
User restrictions                          none
Names                                      none
Tracing and breakpoints                    default
for any mimo_plant
whenever the n2 of the mimo_plant receives a value and when (
 the n2 of the mimo_plant = the max_delay of the mimo_plant +1
and
the n2 of the mimo_plant =the value of the n2 of the mimo_plant as of 2
seconds ago
and the n1 of the mimo_plant <= the max_delay of the mimo_plant + 1
and
the nu of the mimo_plant =1 )
then inform the operator for the next 20 seconds that "
servo supervisor:
N2 for [the name of the mimo_plant] has
been set to [the value of the n2 of the mimo_plant] twice,
this is the lower limit hence check
for tight rate constraints on U or
for large setpoint changes causing
interaction among channels - the
desired rise times may not be met"

Scan interval                              none
Focal classes                              none
Focal objects                              none
Categories                                 none
Rule priority                              6
Depth first backward chaining precedence   1
Timeout for rule completion                use default

                                                 a rule
Options                                    not  invocable  via  backward
                                           chaining,
                                           not invocable
                                           via forward chaining, may cause
                                           data seeking,
                                           may cause
                                           forward
                                           chaining
Notes                                      OK
User restrictions                          none
Names                                      none
Tracing and breakpoints                    default
for any mimo_plant
whenever the n2 of the mimo_plant receives a value
and when(
 the n2 of the mimo_plant =the value of the n2 of the mimo_plant as of 2
seconds ago and the
    n2 of the mimo_plant = 40)
then inform the operator for the next 15 seconds that "
servo supervisor:
N2 for [the name of the mimo_plant] has
been set to 40 twice, this is the upper
limit of the MGPC algorithm (!) NO larger
values are possible"
Scan interval                              none
Focal classes                              none
Focal objects                              none
Categories                                 none
Rule priority                              6
Depth first backward chaining precedence   1
Timeout for rule completion                use default
```

```
                                                  a rule
Options                                          not  invocable  via  backward
                                                 chaining,
                                                 not invocable
                                                 via forward chaining, may cause
                                                 data seeking,
                                                 may cause
                                                 forward
                                                 chaining
Notes                                            OK
User restrictions                                none
Names                                            none
Tracing and breakpoints                          default
for any mimo_plant
whenever the n2 of the mimo_plant receives a value and when (
 the n2 of the mimo_plant = the n1 of the mimo_plant
and the n2 of the mimo_plant =the value of the n2 of the mimo_plant as of
2 seconds ago
and the n1 of the mimo_plant > the max_delay of the mimo_plant + 1
and the nu of the mimo_plant =1 )
then inform the operator for the next 15 seconds that "
servo supervisor:
N2 for [the name of the mimo_plant] has
been set to [the value of the n2 of the mimo_plant]  twice,
this is the lower limit hence check
for tight rate constraints on U or
for large setpoint changes causing
interaction among channels - the
desired rise times may not be met"
Scan interval                                    none
Focal classes                                    none
Focal objects                                    none
Categories                                       none
Rule priority                                    6
Depth first backward chaining precedence         1
Timeout for rule completion                      use default
```