University of Alberta

The Spatial Statistics of Linear Features: An Application to Ecology

by

Brian C. Tucker

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

Doctor of Philosophy
in
Ecology

Department of Biological Sciences

© Brian C. Tucker
Spring 2012
Edmonton, Alberta

To my family.

Abstract

Spatial ecology is concerned with examining the spatial aspects of ecological systems, and it involves the integration of the spatial attributes of the study system into hypotheses, experimental design and analyses. Despite the work that has been undertaken and the diversity of analyses available to ecologists for examining spatial data, one area of analysis has seen little development in ecology: the examination of linear spatial structure in ecological systems. Although linear spatial structure can be found throughout ecological systems (e.g., animal movement paths, burrows, plant roots and shoots), the vegetative spread of clonal plants through stolons and rhizomes is of particular interest because of the important relationships between pattern (clonal spread) and process (physiological integration, foraging, dispersal, fitness, asexual reproduction). Fortunately there do exist tools that allow for the analysis of such data. Stochastic geometry and, more specifically, the theory of “fibre processes” and related theory provide methods capable of dealing rigorously with spatial structures composed of linear components. However, their application in ecology awaits. This thesis introduces methods of analysis applicable to plant ecology presented along with examples.

Preface

This thesis is composed of five integrated components that are the result of a persistent effort to investigate the feasibility of studying linear spatial data from ecological systems, in particular in plant ecology. Because the effort has been continuous, the sections follow a logical order and document a process of investigation toward the above mentioned goal. Chapter 1 provides an introduction to linear data, a discussion of its relevance in ecology, particularly plant ecology, and an overview of data analysis problems and potential solutions. Chapters 2 through 5 serve to introduce methods of data analysis relevant to the study of linear data in plant ecology. In each chapter the analyses are used alongside other more well-known analyses to investigate artificial, field or experimental datasets. Chapter 2 introduces a new method of examining the second order structure (clustering or inhibition) of linear data based on resampling using circular test lines. This analysis was used to examine the stolon arrangements of field and forest populations of *Fragaria virginiana*. Chapter 3 introduces a rotational analysis for examining the directional properties of linear data. The methods are applied to stolon mappings of *F. virginiana* along a forest-oldfield transition. Chapter 4 presents a genet randomization analysis and an analysis based on the analysis of random fibre points. Both of these analyses are used to describe the structure of stolon mappings from a *F. virginiana* transplanting experiment. Chapter 5 uses some of the previously developed tools and a local-global partitioning analysis to examine *F. virginiana* data from a nutrient heterogeneity transplant experiment. Chapter 6 provides closing thoughts, including a synopsis, further directions and unresolved issues.

Acknowledgements

I would like to acknowledge the assistance of those who have helped me in making this project a reality. If it were not for the patience of others I would not have been able to perform such an exploration as I have.

I would like to acknowledge my supervisor, Dr. M.R.T. Dale, for his guidance during the project. I appreciate the encouragement and the flexible research environment that allowed me to explore without restriction. I would like thank my advisory committee members, Dr. M. Lewis, Dr. J. Cahill, and Dr. P. Blenis, for their suggestions and support.

I would also like to acknowledge the agencies that have made this project a reality, either through funding to my supervisor, Dr. Dale, or to myself. These include NSERC, the Kilam Foundation, the University of Alberta, and the Thompson Foundation.

To my wife, Tracy, I extend my deepest love, respect and gratitude. You assisted me when I most needed it, provided helpful recommendations and supplied endless support. Without you this accomplishment would not be what it is. To my parents and my sister, thank you for the support you have provided throughout the years. Your thoughtfulness and encouragement has been a source of strength.

Table of Contents

List of Tables

List of Figures

Chapter 1

An introduction to the examination to linear structure in ecology

1.1 Introduction

1.1.1 The importance of "spatial ecology"

Spatial ecology is concerned with examining the spatial aspects of ecological systems, and it involves the integration of the spatial attributes of the study system into hypotheses, experimental design and analyses. The spatial aspect of ecological systems is studied widely throughout the field of ecology, and integration of spatial questions into ecological field experiments, theory, and natural experiments has been widespread (see Dale (1999) and references therein). Often in ecology, the spatial aspect of an ecological problem might be integral to the exploration of the questions at hand (e.g., arrangement of vegetation, ranges and behaviour of motile organisms, change of distributions over time and space, the distribution of species and species assemblages over the landscape). Such questions require the integration of the spatial aspect of the system into theory, experimental design, data collection, and analysis.

The study of the spatial aspect of ecology is an important activity toward improved understanding of ecological systems. Characterization of the spatial pattern might provide insight into the processes likely involved in pattern formation and persistence (Watt, 1947). Pattern analysis can help identify or rule out processes that may or may not be involved in the establishment of observed patterns (Couteron and Kokou, 1997). Legendre (1993) argues that, in some cases, the consideration of spatial structure in ecological models, rather than the addition of more environmental variables, may offer a more robust explanation of ecological processes. In addition, biological systems can and do alter their surroundings; in essence, living systems become part of the landscape processes themselves, creating ample opportunity for feedback mechanisms. Further complicating the matter is the fact that the relationship between pattern and process likely varies from system to system and from place to place.

When considering pattern and process in a system, two fundamental questions are to what level can spatial pattern be used to infer the existence and nature of underlying ecological processes and to what level does process influence pattern (and vice versa). For example, Watt (1947), in his influential paper, discussed the relationship between pattern and process in ecological systems and argued for the existence of a link between pattern and underlying ecological processes. van der Maarel (1996) continued the discussion of pattern and process in the context of patch dynamics (natural disturbance and patch structure). More recently, a greater understanding of pattern and process in order to provide an improved understanding of species coexistence has been highlighted as a suggested area of research focus (Agrawal et al., 2007).

There are many examples from the literature discussing pattern and process in field research. For example, evidence for processes influencing spatial pattern has been detected in moss (Okland and Bakkestuen, 2004). Druckenbrod et al. (2005) have shown that competition, with light availability and soil moisture availability as principle drivers, is responsible for the spatial pattern of forest vegetation. Also, Odion and Davis (2000) describe a relatively clear link between fire, recruitment and spatial pattern of preburn vegetation communities. In some cases the pattern and processes can form a feedback system, in which case the question of how the relationship became established becomes critical to its understanding. An example of a cyclic reinforcement of pattern and process can be found in Bruce Tiger, where the pattern, once established, further reinforces the original process that was responsible for its formation (Dale, 1999). Similarly, a feedback system between wind and alpine treeline has been identified as the most significant factor in determining treeline pattern (Alftine and Malanson, 2004).

In addition, the integrated nature of ecological systems means that ecological components that would be considered separate under conventional nomenclature and classification approaches (e.g., species, genus, functional group) could have an intimate spatial relationship. The spatial structure of one ecological component could serve as the structure over which another component is distributed or a process which facilitates or inhibits the second component’s distribution. For example, seagrasses and their epiphytes are closely linked to the abundance of fauna, and the spatial structure of the vegetation components can alter the abundance of these species (Edgar and Robertson, 1992; Irlandi and Peterson, 1991). Further complicating the matter is that more than one process might be responsible for an observed pattern. For example, patterns of species invasions might be caused by a wide range of processes (Fridley et al., 2007). While ecological process and spatial pattern can have an integral relationship, in some cases the link between pattern and process can be elusive. The nature of the relationship also varies between ecological systems. For example, there has been some success in identifying processes from pattern in the case of savanna tree-grassland, but simulation and point pattern analyses showed that caution should be used when attempting to determine underlying processes from a single snapshot of the system (Jeltsch et al., 1999).

Although the importance of scale in ecological study is well-known (Levin, 1992), the study of process and pattern is further complicated by the problem of multiple scales (Levin, 2000). In addition, different processes can operate at different scales of the same system (Fridley et al., 2007). Gosz (1993) discusses the examination of pattern and process at many scales in the context of ecotones. Pyšek and Hulme (2005) have stressed the importance of data collected at multiple scales in order to understand plant invasions. Recently, the concept of scale has been discussed in the context of biological diversity (Beever et al., 2006).

Despite these challenges, there is an extensive body of literature covering a wide range of techniques available to the ecologist wishing to characterize the spatial structure of ecological data (e.g., see Dale (1999); Fortin and Dale (2002); Legendre and Fortin (1989); Ripley (1981); Dale et al. (2002); Perry et al. (2002); Goodall and West (1979)). Examples of such methods include point pattern analyses (Dale and Powell, 2001;

Mugglestone, 1996; Ripley, 1981), quadrat methods (Dale and MacIsaac, 1989), contact sampling (Dale et al., 1991), and lacunarity analysis (Dale, 2000). It's also important to note that many of the various spatial analyses are related in various ways (Dale et al., 2002).

Despite recent advances and active research, dealing with the spatial component of ecological systems is potentially daunting. Space can be dealt with in one, two or three dimensions. As dimensionality increases, both computational and data collection requirements become increasingly problematic and labour intensive.

The examination of space in ecology is also fraught with peril. If not considered in the experimental design of field experiments, existing spatial structure can influence tests of significance (Legendre et al., 2004). For example, spatial autocorrelation has been found important in the spatial analysis of plant phenological variables in *Chromolaena odorata* (Almeida-Neto and Lewinsohn, 2004). A number of statistical tests might be required for the control of spatial autocorrelation (Fortin and Payette, 2002). However, spatial autocorrelation might be best dealt with by modelling the spatial data and the autocorrelation structure and using this model in a Monte Carlo simulation to make inferences using a test statistic (Dale and Fortin, 2002). A further complication is that spatial autocorrelation may in itself be heterogeneous, and local measures may be required to investigate its properties (Boots, 2002).

In many cases, particularly in the case of ecological data, the spatial structure of the process under study changes with location (heterogeneous) and/or changes with direction (anisotropic): the process is non-stationary. This is problematic because stationarity is an assumption of many statistical analyses, and in order to make inferences using many methods, the properties of the process must be assumed to be constant. Ripley (1981) defines the stationarity of a stochastic process in terms of two geometric transformations, translations and rotations. A stochastic process is stationary under translations (homogeneous) if the distribution is unchanged when the origin of the index set is translated. A stochastic process that is stationary under rotations about the origin is called isotropic. Also, see Cressie (1991) for an in depth discussion of stationarity. Ripley (1981) comments on non-stationarity by stating, “Note that they [homogeneity and isotropy] can, at most, be partially checked by, for example, splitting the study region into disjoint parts and checking their similarity.” This statement alludes to the need to examine the global pattern in the context of small, local analyses.

Despite the range of work that has been undertaken and the diversity of analyses available to ecologists for examining spatial data, one area of analysis has seen little development in ecology: the examination of linear data. Linear spatial structure can be found throughout ecological systems. It becomes particularly evident in the study of vegetation, where a wide range of structures and potentially associated processes might be considered (e.g., roots, shoots, rhizomes, stolons, blowdown, strings & flarks). Linear spatial structure is also important in the study of other organisms (e.g., micorrhizal fungi (Crites and Dale, 1998; Boddy, 1993), earthworms (Benes et al., 1997), and animal movement paths (Benhamou, 2004, 2006; Schick et al., 2008)). The research direction highlighted in this thesis introduces a series of promising approaches to examining the linear spatial structure (e.g., lines and segments) of ecological systems. This discussion will put into context the application of a specific subset of computational tools that provide for enhanced investigation of the linear spatial structure found in some ecological systems. Although the methods will be discussed primarily in the context of clonal plants, the potential application of these methods and approaches extends beyond that of clonal vegetation or of vegetation in general.

1.1.2 Vegetation and linear structure

Linear spatial structure can be found in vegetation at a wide range of scales. At scales smaller than the plant, transport systems (e.g., in stalks and leaves) can be described by collections of lines. At the scale of the plant, the repetition of linear modules produces plant architecture that also can be described using lines. Such structures include above-ground morphology (stalks, trunks, limbs, stolons) and below-ground morphology (roots and rhizomes). At scales larger than the plant, linear structure can arise from the arrangement of the plant community (e.g., riparian vegetation, vegetation waves, and the boundary between vegetation types). By extension, fallen vegetative material, such as blowdown, twigs and needles, can also be expected to produce linear structure and possibly provide insight into the processes that resulted in its distribution (e.g., a disturbance event, water flow, prevailing wind direction).

The spatial structures produced from linear systems can be complex and daunting. For example, see Brisson and Reynolds (1994) for an example of root spatial structure. Below-ground plant components (e.g., rhizomes) require excavation prior to mapping, and both above- and below-ground data collection requires some form of mapping technique to capture accurately the linear data in two dimensions. As a result, the collection of such data can be expensive in terms of time and effort.

While the linear structure of vegetation offers many promising avenues of investigation, the vegetative spread of clonal plants through stolons is the primary focus of this research. Clonal plants can produce a range of spatial structures (e.g., see Bell and Tomlinson, 1980; Kenkel, 1993; Maddox et al., 1989). Often clonal species have a high degree of plasticity, and in some cases they can also exhibit plasticity of components other than stolons and rhizomes. For example, Semchenko et al. (2007) found that root placement patterns of *G. hederacea* and *F. vesca* grew differently when in the presence of neighbours whereby *F. vesca* roots were stimulated by contact with *G. hederacea*. This can have important ecological implications. For example, Semchenko et al. (2010) have also found that these contrasting rooting behaviours in *F. vesca* and *G. hederacea* can allow for coexistence. Some spatial structures (clonal growth forms) are likely more suited to specific habitats. For example, Sosnová et al. (2010) found disturbance regimes and water gradients in aquatic and wetland habitats to be important predictors of the type of clonal growth strategies employed by species composing the communities at these sites. Regardless of neighbour or site conditions, directly mapping stolon or rhizome connections allows for a number of processes to be considered in the spatial context (e.g., asexual reproduction, resource acquisition, territoriality), and examining the spatial attributes of the mapped spatial structure (e.g., the length of stolon in a unit area) might lead to an improved understanding of the underlying processes.

Often a clonal plant remains an interconnected collection of ramets joined together by stolons or rhizomes. Thus, their spread through rhizomes and stolons can result in the development of collections of genet networks (e.g., Edwards (1984); Maddox et al. (1989)). This can lead to physiological integration of interconnected ramets, which in turn offers advantages to clonal plants not available to non-clonal species. For example, resource transfer along stolons or rhizomes between separately rooted ramets can lead to performance increases in heterogeneous environments (Alpert et al., 2003). Some clonal plants can preferentially allocate plant parts (e.g., rhizomes and/or stolons and ramets) to take advantage of heterogeneous soil conditions and thus increase productivity in the form of biomass (Birch and Hutchings, 1994). This highlights the need to consider processes such as foraging, habitat selection and habitat preference in the context of clonal spatial patterns. For this thesis, foraging will be defined as "the processes whereby an organism searches, or ramifies within its habitat, which enhance its acquisition of essential resources" (Hutchings and de Kroon, 1994). Habitat selection can be defined as a process whereby an organism makes a series of decisions with respect to its habitat (Hutto, 1985), and habitat preference can be defined as the result of the process of habitat selection (Hall et al., 1997). There is also evidence that clonal plants may regulate physiological integration processes. Alpert et al. (2002) found that resource sharing between ramets of *F. chiloensis* can be modified by hormones (auxin) suggesting a potential role for these compounds in regulating clonal integration processes. Xiao et al. (2010) found clonal integration of *Spartina alterniflora* to enhance tolerance to perturbation (flooding). The clonal growth form is also important in the exploration and maintenance of territory. For example, *Clintonia borealis* has been found to invade territory in a wedge-shaped fashion due to a combination of internode length and branching angle (Angevine and Handel, 1986).

The clonal growth of plants has received considerable recent attention (Stuefer et al., 2002; Price and Marshall, 1999; Tolvanen et al., 2004; Sammul et al., 2008), particularly in the context of environmental heterogeneity. However, only first steps have been taken to examine the explicit spatial structure of clonal plants, and few studies have tried to examine the explicit spatial properties in the context of the clonal response to heterogeneous environments. Maddox et al. (1989) investigated the spatial structure of *Solidago altissima* rhizomes by examining rhizome length and angular distribution. Sammul et al. (2004) found that clonal mobility increased and branching intensity of rhizomes decreased when moving from open meadows to forests. Sampaio et al. (2004) examined the directional growth of a clonal plant and found evidence for habitat selection in vegetation islands. These studies focus on first-order properties (e.g., directional distribution) but do not use second-order spatial analyses (e.g., at a range of scales), nor do they take into account the explicit spatial properties of ramets, nodes, and stolon or rhizome internodes. It is possible to consider linear features in terms of branching pattern architecture (Bell and Tomlinson, 1980), but this approach doesn't allow for quantitative analysis of structure. Fractal methods have seen application in ecology (Halley et al., 2004) and allow the treatment of linear features, but they do not provide easy to interpret quantitative measures of shape. It is tempting to treat clonal plants as networks (Strogatz, 2001), but such analyses would emphasize connectivity rather than spatial properties.

The lack of explicit spatial treatment of linear spatial structures is likely due to the fact that the examination of linear data presents a number of logistical and theoretical problems. The data, being composed of points and lines, both of which can have additional attributes, is effort- and computationally-intensive to collect and analyze, respectively. This requires the development and testing of efficient field sampling procedures.

It also requires the utilization and development of computational and statistical procedures for data processing and analysis after collection. A further complication is that linear structure can be sampled and stored using two fundamentally different formats (raster and vector data) between which conversion is not always practical nor warranted. Additionally, spatial heterogeneity and anisotropy are issues that, although not specific to linear data, can render statistical tests invalid. These problems must be considered before the analysis of linear features can be effectively adopted in ecological studies.

1.2 Statistical Considerations

The field of stochastic geometry provides tools that allow for the analysis of spatial structure in a number of dimensions (D = 0 (points), D = 1 (lines), D = 2 (polygons), & D = 3 (volumes)) (e.g., see Weibel (1980); Stoyan et al. (1995)). Some of these methods are well-known in ecology. For example, point pattern analysis has been a common method for examining the point locations of individual plants.

1.2.1 Spatial analysis: structures composed of points

Point pattern analyses examine the locations of points in two dimensions (Ripley, 1981). The theory is relatively well-known in ecology and has had a long history of application in plant ecology. The relative ease of measuring plant location has led to the wide use of point pattern analysis to examine the spatial properties of vegetation, and a wide range of methods exist to treat point pattern data (e.g., see Dale, 1999; Ripley, 1976, 1981). The utility of these analyses in ecology is underscored by the continued interest in their application (Wiegand and Moloney, 2004; Perry et al., 2006; Picard et al., 2004).

1.2.2 Spatial analysis: structures composed of regions

The use of regions, or polygons, is also relatively well-established in ecology. Polygons often represent regions containing a particular species of interest, particularly in the case of plant spatial pattern. Polygons of plant community patterns are the result of detailed mappings of the plant's location represented as a 2-dimensional shape. Quantitative methods of examining polygon change have received some attention (Sadahiro, 2001). Another family of polygon analyses include tessellation methods, which have a relationship to point patterns and hold at their core the idea of dividing the study area into adjacent polygons having specific properties (Mugglestone, 1996; Dale, 1999). Polygons have also been used to examine the effect of neighbours on root distribution in *Larrea tridentata* (Brisson and Reynolds, 1994). Stochastic geometry also provides methods to examine regions (Stoyan, 1990; Stoyan et al., 1995; Weibel, 1980).

1.2.3 Spatial analysis: structures composed of lines

The collection of data in the form of lines, contrary to points and polygons, occurs much less frequently in ecology, despite the ubiquity of linear spatial structures in ecological and biological systems. In plant ecology, detailed analyses of linear spatial data are relatively limited. There does exist, however, a suite of statistical tools that allows for the examination of such data. Stochastic geometry and, more specifically, the theory of "fibre processes" (Mecke and Stoyan, 1980; Stoyan et al., 1995) and related theory provide methods capable of characterizing spatial structures composed of linear components. To date, however, there has been little application of these procedures to ecological data.

1.2.3.1 Fibres, fibre systems, and fibre processes

The theory of “fibre processes” (Mecke and Stoyan, 1980; Mecke, 1981; Stoyan et al., 1980), and the broader fields of stochastic geometry (Stoyan et al., 1995) and stereology (Stoyan and Gerlach, 1987; Weibel, 1980) provide methods for the characterization of patterns produced by linear components. There has been some limited application of these and related methods to ecological data. Benes et al. (1997) illustrated the use of length estimation of fibre processes by examining the pattern of earthworm burrows, and Stoyan and Kuschka (2001) examined pitfall trap data to estimate animal abundance. In addition, the straightness of linear spatial structure has been discussed in the context of animal movement paths (Benhamou, 2004, 2006). Although these methods show promise in examining the linear patterns produced by ecological systems, they largely await application in plant ecology.

Fibre processes are a promising framework with which to examine linear features in ecology due to their foundation in stochastic geometry, and thus their provision for null model hypothesis testing. The general theory of fibre processes was introduced by Mecke and Stoyan (1980). A *fibre* in the two-dimensional case is a smooth differentiable curve occurring in the plane (Figure 1.1A). Mecke and Stoyan (1980) formally define a fibre as a subset of $\mathbb{R}^2$ which can be represented as the image $\gamma([0,1])$ of the segment $[0,1]$, where $\gamma : (a,b) \to \mathbb{R}^2$ is a mapping of the segment onto $\mathbb{R}^2$.

It is possible to think of a *fibre system* as the union of a finite number of individual fibres (Figure 1.1B). Formally, a fibre system ϕ is a closed subset of $\mathbb{R}^2$ which can be represented as a union of at most countably many fibres, with the property that any compact set is intersected by only a finite number of the fibres and such that distinct fibres have only end-points in common (Mecke and Stoyan, 1980). However, ϕ can consist of intersecting curves because fibres can end at intersection points (Stoyan et al., 1995).

A *fibre process* Φ is a process which "distributes" fibres in $\mathbb{R}^2$ or $\mathbb{R}^3$ according to some distribution (Figure 1.1C). Mecke and Stoyan (1980) define a fibre process Φ as a random variable that is a measurable mapping from an underlying probability space onto the family of all planar fibre systems. This is directly analogous to the term *point process* referring not to the points themselves but to the random process that distributed the points.

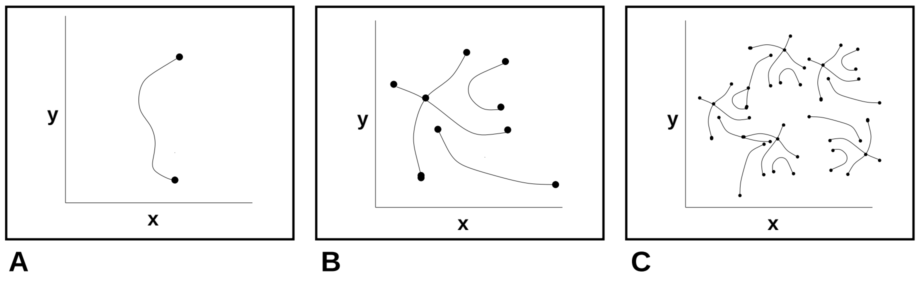


Figure 1.1: A fibre (A), fibre system (B), and fibre process (C) represented in $\mathbb{R}^2$.

In a second paper, Stoyan et al. (1980) apply the theory introduced by Mecke and Stoyan (1980) to partially oriented fibre systems. They develop formulae for the determination of the mean length of lines per unit area via measurement of the mean number of intersections per unit test line. In a third paper, Mecke (1981) extends the work done by Stoyan et al. (1980) by considering the intersection of stationary fibre processes with arbitrary curved lines of finite total length.

Stoyan (1981) discusses the second-order analysis of planar fibre processes and provides three example analyses arising from different spatial models. Stoyan and Stoyan (1986) introduce models that can be used to analyze dislocation distributions, irregular spatial structures of interest in the materials sciences. The analysis of orientation of fibre systems has been explored (Karkkainen et al., 2002), and Stoyan and Gerlach (1987) provide methods for examining curvature distributions for fibre systems. Overlapping, thick fibres have been examined, and a Boolean model having rectangular grains has been suggested as a suitable mathematical model (Molchanov et al., 1993).

The basic theory of fibre processes shares its roots with that of point processes (Stoyan et al., 1995), and often work discussing first and second order measures on fibre processes discusses point processes as well (e.g., Hanisch et al. (1985)). In fact, several of the measures of fibre systems are close relatives to those of point processes.

Point processes have a rich history of application in plant science and other fields, and the theory allows for flexible application and extension. For example, in the case of stationary and isotropic spatial fibre system, the second-order pair-correlation function and K-function can be used to examine its spatial properties (Krasnoperov and Stoyan, 2004; Stoyan et al., 1995). Also, the Boolean model (Molchanov, 1995) has been applied to the analysis of linear systems (Molchanov et al., 1993; Molchanov and Stoyan, 1994). In addition, Penttinen and Stoyan (1989) have used point patterns to examine the spatial properties of segment processes.

Stereology can be considered a branch of spatial statistics (Stoyan, 1990), and Stoyan (1985) provides an introduction to using stereological methods to determine the orientation, second-order quantities and correlations in fibre processes. Stoyan (1998) provides an introduction to random sets and discusses a wide range of models.

Fibre processes, however, are subject to the same assumptions of the theory of point patterns, that of the underlying stochastic process being homogeneous and isotropic. Problems of anisotropy have been explored by Benes et al. (1997) and Benes et al. (1994). Cruz-Orive et al. (1985) provide parametric methods for analysis of anisotropic linear structures, and Stoyan et al. (1980) further the discussion by providing methods for analyzing partially oriented fibre systems.

First order measures of fibre processes

First order measures give mean values of characteristics of the fibre process under consideration. They provide a description of the general behaviour of the stochastic process as a whole.

These include:

a. Intensity or fibre density (*L*): This is the average length of fibre segments in a given area in $\mathbb{R}^2$ or volume in $\mathbb{R}^3$.

   To find B_A, the boundary length per unit area, Weibel (1980) provides

$$B_A = \frac{\pi}{2} \times I_L \tag{1.1}$$

   where,

   B_A = the estimate of the length of line per unit area

   I_L = the intensity of intersection points (# / unit length of test line)

Stoyan et al. (1995) provides equivalent formulae:

If the process is isotropic:

$$\hat{L}_A^i = \frac{\pi \times \#\{T \bigcap \Phi\}}{2 \times h_l(T)} \tag{1.2}$$

where,

$\hat{L}_A^i$ = the estimate of the length of line per unit area

$\#\{T \bigcap \Phi\}$ = the number of intersection points between a test line set (T) and the fibre process Φ

$h_l(T)$ = the length of test line

If the process is not isotropic:

$$\hat{L}_A^0 = \frac{\#\{C \bigcap \Phi\}}{4 \times n \times R} \tag{1.3}$$

where,

$\hat{L}_A^0$ = the estimate of the length of line per unit area

$\#\{C \bigcap \Phi\}$ = the number of intersection points between a circular test line set (C) and the fibre process Φ

R = fixed radius of circular test lines

b. Rose of directions or directional distribution: The average directional distribution of tangents to the fibres occurring in the space. It is a summary of the orientation of the fibres.

Second order measures of fibre processes

Hanisch et al. (1985) point out that, in order to describe clustering, repulsion, or other forms of internal correlation, mean values (first order measures) are not suitable. For these types of questions (i.e., the description of fluctuations in the internal structure of a random process) the second reduced moment function and the pair correlation function can be used. The second order measures allow for the examination of the fibre process properties at a range of radii.

a. Second reduced moment function (*K*): If x is a typical fibre point in Φ, the *LK*(*B*) is the mean total length of all pieces of Φ in the ball or circle *B* (Stoyan et al., 1995).

b. Pair correlation function *g*(*r*): This is related to the *K* function but provides a measure of fibre density of Φ at a particular distance (Stoyan et al., 1995).

1.2.3.2 Weighted fibres

It is also possible to consider weighted fibres (Schwandtke, 1988) where fibres carry additional information. This is directly analogous to a marked point pattern analysis and serves the same purpose, the ability to include more information into the data analysis. The weight can be a quantity or a label and allows for the consideration of additional properties of the fibre during analysis. In a plant system the weight attributed to a given fibre could represent the diameter of plant parts (e.g., roots or shoots), rhizome or stolon branching order, the conductive capacity of a given structure, or another characteristic of interest.

1.2.3.3 Along-fibre and between-fibre process

Another important aspect of the application of linear data analysis in ecology is differentiating between-fibre processes and along-fibre processes. In the case of clonal plants, this presents two somewhat related, but conceptually different fields of focus: (1) the consideration of nodes along fibres that represent below-ground components (root and rhizome branching, ramet formation) and (2) the consideration of nodes that represent the above-ground point pattern of the ramet stems (ramet formation, stolon branching). This conceptual framework is particularly powerful because the spatial arrangement of points can be interpreted very differently when considered in the context of between-fibre and along-fibre processes. For example, Figure 1.2 shows how a distance-based cluster of points might look if the underlying fibre system is not considered (Figure 1.2 A) and how the clustering could change if the points are considered constrained by the fibres (Figure 1.2 B). Above-ground point pattern is what is often considered when examining the spatial arrangement of plants, and reconciling above-ground point pattern analysis and below-ground fibre analyses into a jointed spatial analysis method is a challenging but promising area for further research.

1.2.3.4 Hard-core and soft-core distances

Other important aspects of the theory to be considered are hard-core and soft-core repulsion distances in the context of between-fibre (Figure 1.3 A & B) and along-fibre (Figure 1.3 C) distances. Hard-core and soft-core distances are important in point process models where they summarize important properties regarding the model responsible for the generation of events. In the context of point process models, a hard-core distance is

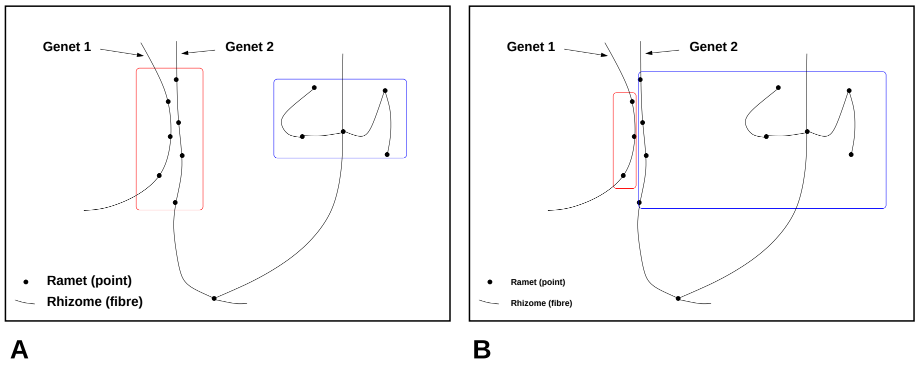


Figure 1.2: Rhizome networks of two genets with associated nodes (locations of ramets), (A) distance-based clusters of points if the rhizome network is not considered (between fibre clusters), (B) distance-based clusters if the rhizome network is considered (along fibre clusters).

where points are not allowed to come within a distance r of one another, while a soft-core distance represents a density function around each point that determines the likelihood of other points occurring within radius r. The pair correlation function can be used to get an indication of repulsion distances of fibres (Krasnoperov and Stoyan, 2004). However, current methods require that fibres will have similar repulsion distances along their lengths because it considers the points generated by the sectioning to be equal regardless of their position along a given fibre. In actual ecological systems, this assumption is most likely not realistic. It may be that certain components of a given fibre system (e.g., the growing tip of a rhizome) may have very different hard- or soft-core properties than a section of rhizome that has been in place for a longer period of time. Particularly challenging is to reconcile the repulsion distances found using sections of fibre processes with methods that consider repulsion distances as well as position of measurement along fibre lengths. Of particular interest is the examination of patterns where these distances become zero (e.g., contact and/or fusion of adjacent structures). This is particularly relevant with respect to root and rhizome placement, the spread of clonal ramets, and the placement of branches and leaves.

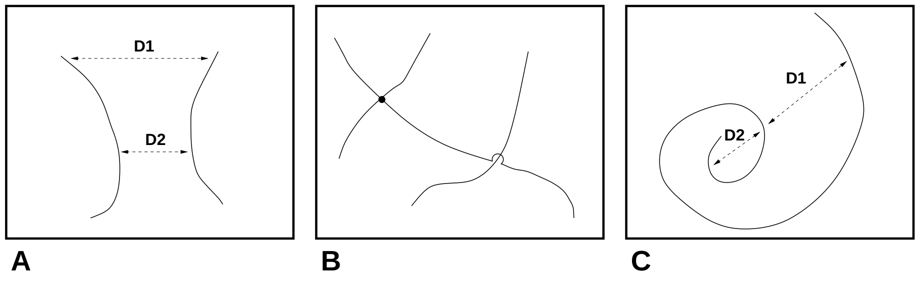


Figure 1.3: Repulsion distances in fibre systems: (A) distances between fibres, (B) intersection of fibres with or without fusion, and (C) distances along fibres.

1.2.4 Practical Application and Examples

A system of fibres can be analyzed by examining the point patterns produced by the intersections of a test system of lines, circles or other fibres (Stoyan et al., 1995). Given a mapping of linear structures, it is possible to create a set of test lines that will overlay the mapping, producing a set of points at intersection locations. These intersections can then be used to calculate measures of the fibre process.

1.2.4.1 Example 1: Segment Process

A segment process could be considered the simplest example of a fibre process. This is a type of Boolean model where the germs are the centre points of the segments and the segments are orientated according to some orientation distribution. Figure 1.4 shows a line segment process (A), a set of test lines overlaying the process (B) and the resultant intersection points that result from the segment process and the test lines (C).

The test line sampling demonstrated in Figure 1.4 makes the assumption that the spatial process is both stationary and isotropic. In a practical sense, it means that the direction of the test lines is irrelevant when sampling the structure to generate intersection points. Regardless of the direction of the test lines, the intensity estimate will be the same. This can be illustrated by sampling a truly isotropic and homogeneous segment process at a range of angles. Figure 1.5(a) shows a random segment process that is isotropic and homogeneous. Figure 1.5(b) shows the intensity (B) (unit length / unit area) for ten instances of processes similar to that of Figure 1.5(a). The intensities for each segment process are plotted as a function of test line sampling angle on a radial plot. Each segment process was repeatedly sampled using straight test lines at various angles, and for each sampling of each process, an intensity estimate (B) was calculated. The generally circular radial plots show that the estimated intensity remains the same regardless of the angle of the straight test lines. This rotational analysis can also be used to determine the level of anisotropy in a system whereby a deformation of the radial plot of linear intensity indicates anisotropy.

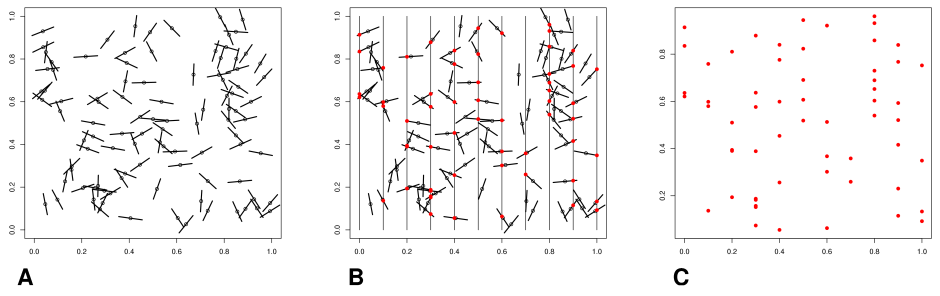
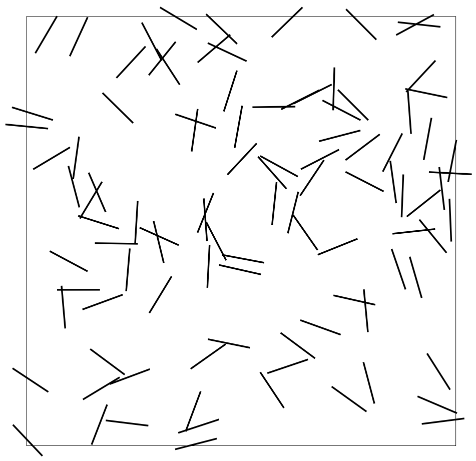
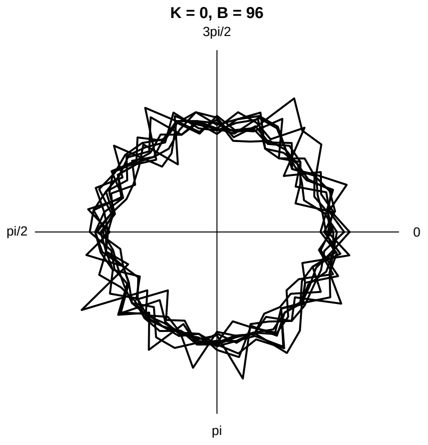


Figure 1.4: An example of the first stages of a simple fibre process analysis. (A) A segment model with 100 randomly located, randomly oriented segments forming a simple fibre system, (B) a series of test lines generating intersection points with segments, (C) the resulting point pattern to which can be applied stereological formulae (Stoyan et al., 1995).


(a) Stationary, isotropic segment process



(b) Radial plot of estimated linear intensity

Figure 1.5: A rotational analysis of a segment process using sets of straight test lines. (a) A stationary segment process (n = 100, segment length=1, total segment length in the plot=96.13). (b) The average estimated linear intensity for 10 segment processes similar to the example in (a) that were sampled using test lines. Each segment process was sampled after being rotated the indicated number of radians.

1.2.4.2 Example 2: *Pteridium* rhizome

It is possible to digitize spatial mappings of vegetation structures (from either hand-mapped data or digital image data) into collections of fibres using a GIS or vector drawing software package. Stoyan et al. (1995) point out that, for practical purposes, smooth fibres can be treated as collections of short line segments. This greatly simplifies computer implementation of analytical methods.

Figure 1.6 shows a representation of a drawing of *Pteridium* rhizome (adapted from Watt (1947)). It has been separated into two datasets for illustration purposes. Pattern 1 and Pattern 2 show two different rhizome patterns that are actually continuous in Watt's paper, and they are used as an example of a potential change in process being indicated by the change in pattern.

One aspect of the data that can be examined is the directional distribution. The rose diagrams in Figure 1.7 show the distributions of lengths of lines composing Pattern 1 and Pattern 2. The rose diagram for Pattern 1 (Figure 1.7(a)) shows that there is preferential stolon growth in the directions $\pi/4$, $(3\pi)/4$ and 0. These coincide with the growth directions in the more regular rhizome pattern in Pattern 1 (Figure 1.6 A). The rose diagram for Pattern 2 (Figure 1.7(b)), however, shows a distribution of radial lengths that fails to indicate a clear preferential growth direction. This coincides well with the less regular growth directions in Pattern 2.

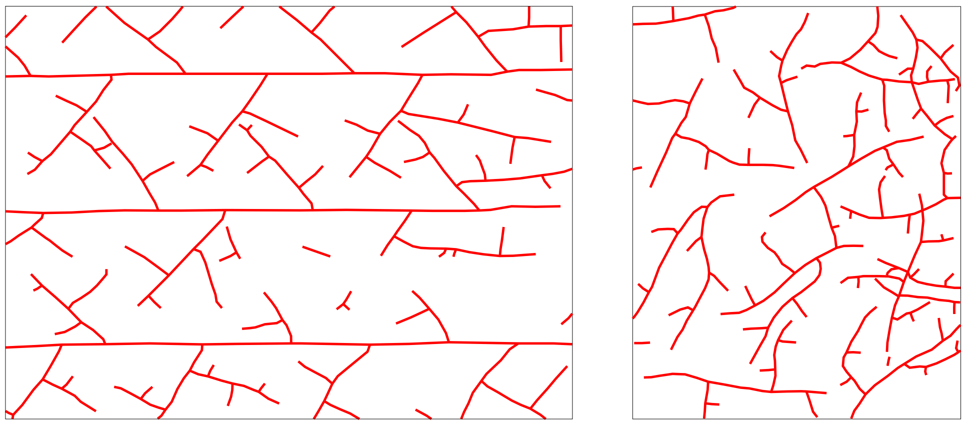
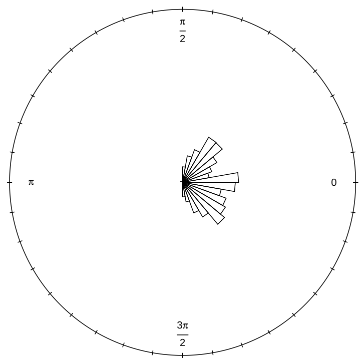


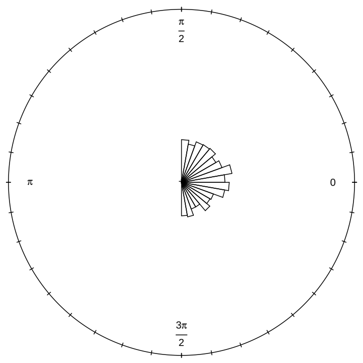
Figure 1.6: Rhizome patterns adapted from Watt (1947).

As previously discussed, another aspect of the data that can be examined is the intensity of the linear structure (length per unit area). As pointed out by Stoyan et al. (1995), test lines can be straight lines, curves or circles. The assumption of isotropy is particularly important in the case of linear data, and anisotropic data can lead to sampling errors when using standard, straight test lines. Fortunately, circular test lines are robust to anisotropic data when estimating B_A, the length of line per unit area.

Figure 1.8 shows Pattern 1 and a set of randomly placed circular test lines along with their intersection points (grey). The intensity can be



(a) A



(b) B

Figure 1.7: The distribution of linear lengths of line segments comprising the digitized *Pteridium* rhizome (rhizome data from Watt (1947)).

estimated using Equation 1.1 where the length of test lines is the total length of circular test line inside the study plot. This method provides an estimate of the average intensity (length/unit area) of the rhizome across the plot. If the data are resampled using a random set of circular test lines, a distribution of intensity estimates can be developed for any given dataset and circle radius. Figure 1.9 shows the box plots of intensity estimates (n = 100 separate samples of the data, each with 100 circles) for each of eight circle radii. The horizontal line indicates the exact intensity of the rhizome (calculated from the exact length of the digitized stolon fragments).

Circular test lines can also be placed non-randomly over the study plot. Figure 1.10 shows Pattern 1 with circular test lines located randomly along rhizomes (on random fibre points). Intersection points between these non-random circles and rhizomes are shown in grey. Intersections between a circle and the rhizome on which its centre resides are not enumerated. By using circles of radius *r*, this approach measures the intensity within a given distance *r* around the rhizome. Figure 1.11 shows the intensity of

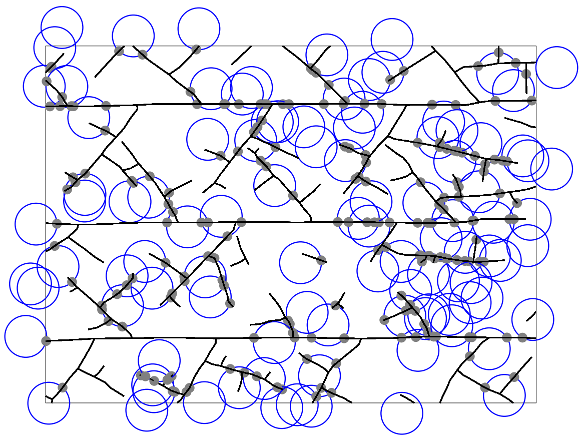
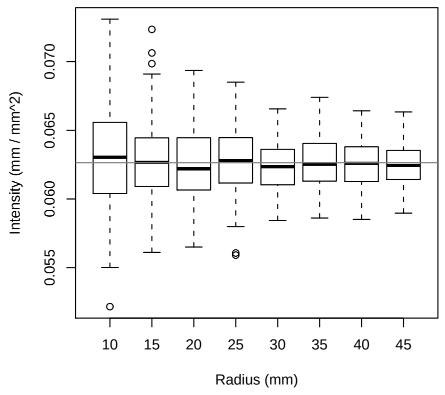
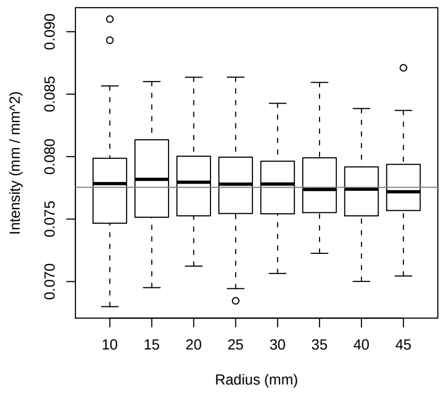


Figure 1.8: Circular test lines of radius 10 units (n = 100) distributed randomly over a portion of the data representing *Pteridium* rhizomes (rhizome data from Watt (1947)).



(a) A



(b) B

Figure 1.9: Box plots of intensity (mm/mm^2) of stolon estimated for Pattern 1 and Pattern 2 using resampling ($nsim$ = 100) with circular test lines (n = 100 circles) randomly distributed over the study plot.

rhizome calculated for Pattern 1 and 2 using sets of circular test lines with radius *r*. At small radii, the intensity of rhizomes around any given rhizome is lower and then increases with distance.

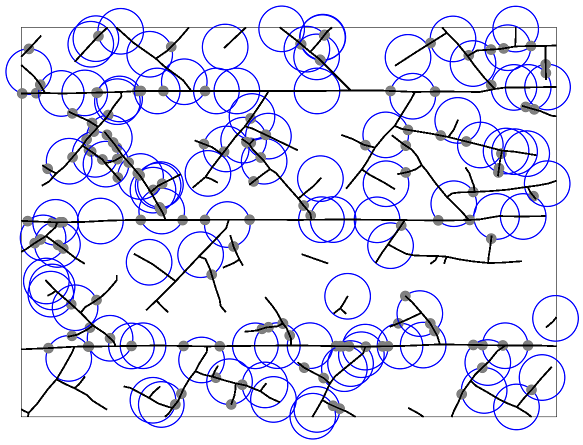


Figure 1.10: Circular test lines (*n* = 100) distributed non-randomly over the study plot but randomly along rhizomes. Intersection points between the circular test line and nearby rhizome sections (rhizome data from Watt (1947)).

1.2.4.3 Example 3: Raster data

Mapping linear data in the field by hand is labour and time intensive. If field conditions allow, it may be more efficient to collect linear data in raster format. Raster data has the benefit of being relatively easy to collect using remote sensing technology (e.g., digital camera). A two-dimensional representation of the study plot is stored using a grid of square pixels, and each pixel is given either a single numeric value (one value between 0-255 as in a greyscale image) or a range of values (e.g., three values between 0-255 for red, green, blue as in RGB images). Image analysis has seen some application in ecology. For example, Donnelly et al. (1995) used pixel-based methods to examine the spatial properties (biomass, growth, fractal dimension) of mycelial systems. Digital photographs have also been used to study spatial structure of vegetation (Zehm et al., 2003). Chadœuf

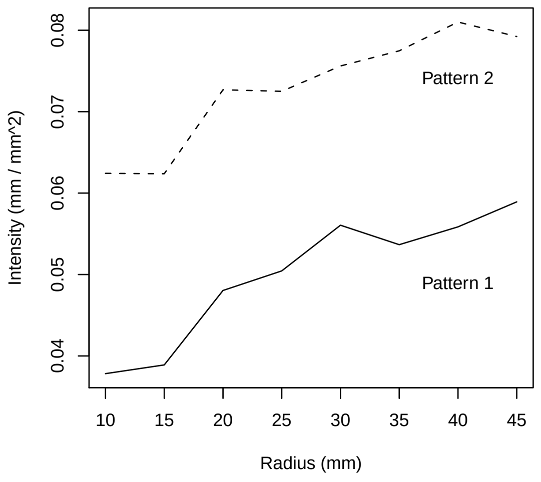


Figure 1.11: Intensity of rhizome in Pattern 1 and Pattern 2 at a range of distances from any particular rhizome estimated using 100 circular test lines for each radius (rhizome data from Watt (1947)).

et al. (2000) provide methods for examining the dependence between two spatial processes using digital images. They consider plots with mixtures of spatial structures (e.g., points, fibres, and areas).

Using image analysis techniques, it is possible to extract linear data from a digital image. Figure 1.12 A shows a colour digital image map of a *F. virginiana* stolon. Figure 1.12 B shows the image map transformed to extract the linear data contained in the original map, and Figure 1.12 C shows a set of test lines sampling the data. Figure 1.12 D shows the identification of the center of the stolon (in x-y coordinate pixel space) marking the intersection of the stolon and a linear test line. Using such an approach, it it possible to extract linear data from digital images and find intersection points between the linear data (in this case, *F. virginiana* stolons) and a suite of test lines. These intersection points can then be used to estimate properties of the system (e.g., linear intensity). Additionally, it is possible to use raster approximations of circular test lines rather than straight test lines.

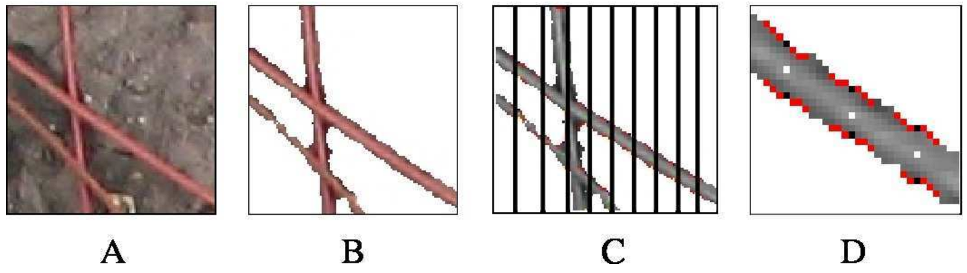


Figure 1.12: Raster data analysis. (A) A colour digital image map of a *F. virginiana* stolon. (B) The image map transformed to extract the linear data contained in the original map. (C) The calculation and identification of the center of the stolon (in x-y coordinate pixel space) marking the intersection of the stolon and a test line set. (D) The resulting intersection points.

1.2.4.4 Example 4: Pixel values

In addition to sampling a colour image, it is possible to examine greyscale images in order to characterize the linear properties of a system. The pixel value in greyscale images, as in colour images, can provide important spatial information provided that some previous information about the system is known. For example, Figure 1.13 shows the greyscale values of pixels from a single test line that has been used to sample an image of conifer needles on the forest floor. Pixel values range between zero (black) and 255 (white). In this example the pixel values of conifer needles are lighter than the shadowed space between the needles, so the peak values along the test line indicate the presence of a needle. The x-y coordinate of each intersection between the test line and the fallen needles can be easily determined whereby the location of the horizontal test line provides the y-coordinate and the test line pixel value maxima provide a set of x-coordinates indicating intersection. By extension, a suite of test lines will produce a set of intersection points over the entire image to which stereological formula can be applied. This approach is similar to the approach of Karkkainen et al. (2001), who estimated the orientation distribution of a fibre process in a digital image using a line transect.

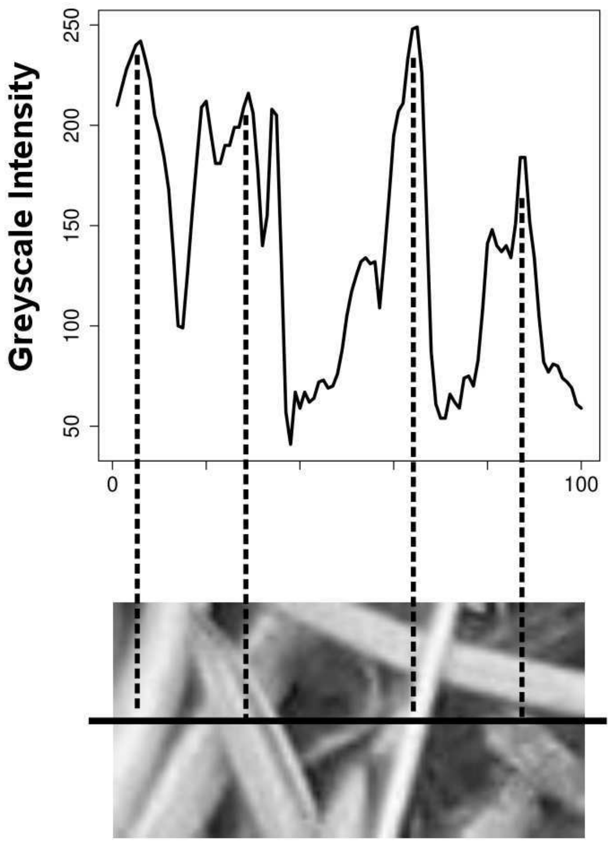


Figure 1.13: The greyscale intensity measured along a test line sampling a digital image of conifer needles.

1.3 Computer Intensive Methods of Inference

While the estimation of measures such as linear intensity can provide a snapshot description of a spatial pattern, making inferences regarding the pattern requires additional consideration. In order to make inferences, the approaches discussed in previous sections require extension so as to allow comparison of observed measures or test statistics and their expected distributions. A common item of interest when examining any set of data is the distribution of data from which the observations are assumed to be drawn, and classical statistical approaches provide distributions against which various attributes of a dataset (estimated measures) might be compared (e.g., see Zar (1999)). Inferences are made based on where the observed measure or test statistic, calculated from the data, lies on a theoretical distribution. There are some classical statistics that might be applied to linear data, such as angular analyses dependent on theoretical distributions (Zar, 1999), but these analyses are limited in number. Non-parametric methods exist to estimate the directional distribution of line and fibre processes, but these assume the process to be stationary (Kiderlen, 2001). Traditionally, classical statistical approaches relied primarily on theoretical distributions, but now computer intensive methods are available as replacements to the classical statistical approach (Manly, 1997). These methods, such as randomization, bootstrap and Monte Carlo methods, provide the benefit of not having to rely on an underlying theoretical distribution. Instead, they rely on distributions that either are produced from the original dataset (e.g., bootstrap) or an underlying model (e.g., Monte Carlo). In the case of spatial data, and linear spatial data in particular, the relative lack of theoretical distributions requires that computer intensive methods, such as Monte Carlo, randomization and bootstrap analyses, be used to make inferences.

1.4 Monte Carlo, randomization, bootstrap

Monte Carlo simulation is a method of statistical inference whereby an underlying model is used to produce a number of expected observations based on a specific null hypothesis. These samples are then compared to the observed data using some measure (or test statistic). The position of the observed test statistic in the distribution of expected observations can then be used as a measure of significance of the observed data (e.g., see Manly (1997) for examples).

An important aspect of Monte Carlo methods is the underlying model, the null model, that is used to generate the expected distribution. The model represents the expected system based on a null hypothesis. The behaviour of the model over many samples will produce a distribution of observations that can be used as a statistical distribution.

That is, the null model, when used in a Monte Carlo simulation, produces an expected distribution of a test statistic that is analogous to the theoretical distribution underlying a classical statistical analysis (e.g., student’s t, Gaussian). Then the observed ecological data can be compared to the reference distribution produced by the model.

Randomization is a type of Monte Carlo where the underlying model is one of complete randomization of some aspect of the system. In the case of a random, two-dimensional point pattern process, for example, the x and y coordinates would be drawn from a distribution of values in which all values are equally likely to arise. Such a case, where the data are completely randomized, can be termed complete spatial randomness, or CSR (Dale, 1999). However, in some cases complete randomization of data or randomizing all parameters of the model may not address relevant ecological questions because it will create unrealistic “expected” distributions. In such cases, a constrained randomization might provide a more ecologically relevant solution. In this case, only certain aspects of the system are randomized, while others are left untouched. This is particularly important when the null model has a number of parameters; the constrained randomization will allow for the investigation of hypotheses relating to a specific aspect of the system.

Bootstrap resampling develops an expected distribution by resampling the observed data. It is based on the idea that the sample data should represent the variability in the population. As such, resampling the data many times can be used to construct confidence intervals for test statistics (see Manly (1997) for additional details). It is possible to adapt this application to spatial data. For example, Figure 1.9 showed the results of resampling the *Pteridium* stolon data using randomly distributed circular test lines. The resampling procedure produced a distribution of linear intensity estimates, which could be used as a component in a bootstrap procedure.

1.5 Conclusions

The study of linear spatial structure in ecology holds promise for the investigation of questions regarding a wide range of ecological systems. There are a range of statistical and computational techniques that can be employed in the investigation of linear data, one of the most important being the field of stochastic geometry. The challenges with working with this data, that of collection and then the subsequent analysis, have been addressed in part by the methods in this chapter, and there is ample opportunity for further extension of the methods in order to address new questions. There is, however, considerably more work to be done, and the theory and application lend themselves to potentially fruitful use in a wide range of systems.

Chapter 2

Characterizing clustering and repulsion in vegetation structure

2.1 Introduction

Plant spatial data can take the form of points, polygons, or lines. To date, polygons (e.g., distribution maps) and points (e.g., plant locations) are the most common forms of spatial data collected in ecological studies. A wide range of analytical tools are available to ecologists for the examination of point pattern data (Dale and Powell, 2001; Mugglestone, 1996; Ripley, 1981). These analyses allow for the estimation of a range of measures that describe the spatial properties of point locations. They can, for example, indicate at what scales point locations are in a clumped, regular, or random spatial distribution across the study area. Such characterization of the data in turn may offer insights into the processes driving the location of plants and vice versa.

However, the spatial aspects of some systems can be better represented as lines. For example, in the case of plant morphology, roots, shoots, stolons and rhizomes are all amendable to representation as lines in two- or three-dimensional space. Examining the placement of plant structures in space is important, particularly when considering possible relationships between spatial pattern and processes. This presents a challenge for ecologists, as there are few methods available for the examination of linear data in the ecological literature. Such analyses could provide insight into the relationships between pattern and process as it relates to the distribution of plant structures in space. In the case of clonal plants, this becomes particularly meaningful, as the distribution of vegetative components (e.g., stolons and rhizomes) can be associated with asexual reproduction, competition, appropriation of ‘territory’, and clonal spread. Being able to quantify and examine the structure of these linear components, then, becomes critical to understanding clonal plant ecology.

The field of stochastic geometry, which provides underlying theory for point processes, also provides theory that can be applied to linear data in ecology (see Chapter 1). In particular, the theory of fibres and fibre processes can be incorporated into analyses of this type.

One way to examine the structure of linear data is to estimate the properties of the data using samples from its spatial structure. For example, it is possible to analyze a system of fibres by examining the point patterns produced by the intersections of a test system of lines, circles or other fibres (Stoyan et al., 1995). Applying this approach to clonal plant data involves overlaying a series of test lines on a mapping of stolons or rhizomes. The intersection between the mapping and the test line set will produce a point pattern that can then be analyzed further.

While straight test lines (see Chapter 1) are the easiest system to manage computationally, they are subject to the assumption of isotropic linear data when estimating intensity (Stoyan et al., 1995). A lattice of test lines can be used, which improves the estimation somewhat, but the assumption of isotropy is still required. However, in ecological systems there is no guarantee that the linear structures under study will be isotropic. For example, Sampaio et al. (2004) have found evidence of directional growth as a means of habitat selection in *Aechmea nudicaulis*. The linear data from such a system would not be amenable to sampling using straight test lines. In order to apply sampling methods where anisotropy cannot be assumed, a method not sensitive to anisotropy is required. Circular test lines are such a solution. They provide robust estimation of linear intensity even in the case of anisotropic data. Stoyan et al. (1995) mentions curved or circular test lines, but papers demonstrating their use are absent in the literature.

Circular test lines also offer the benefit of forming a two-dimensional sampling region. For example, they can be used as a line to find intersections with linear data, and they can also be used as a region in which points can be enumerated. This raises the possibility of using sampling circles for the development of joint point-line analyses. Although not addressed in this research, point-line analyses could be a fruitful area of further research. There has been some work focussed in this area. Berman (1986) offers methods for testing the spatial association between point processes and other stochastic processes, and Foxall and Baddeley (2002) examined the association between points and lines. Such methods, as well as other methods based on stochastic geometry, including a potential point-line analysis using circular test lines, could be useful in plant ecology applications because of the important ecological relationship between linear components (e.g., stolons) and point components (e.g., ramets).

Estimating the intensity of linear data provides a direct estimate of the length of lines per unit area (see Chapter 1). Although very applicable to ecological questions, the density of linear data provides only the simplest glimpse of what ecological processes might be operating in the study plot. An estimate of density, for example, does not give an indication of the relationship between lines at various distances from each other. It also assumes that the properties of the system across the plot are homogeneous; that is, all portions of the plot have the same density. Both of these considerations have direct application to ecological questions, and, although basic sampling of the data using test lines generates a simple estimate of density, it does not address them.

Fortunately, using the concept of test line sampling as a building block, it is possible to construct more advanced analyses that are capable of addressing more complex hypotheses about the data. This is possible because the points given by a sampling test line provide different information depending on how they are generated. For example, a test line can be located randomly over the data or its randomization might be constrained. It might be straight, and thus sensitive to anisotropic data, or it might be curved or circular.

This chapter introduces a novel technique to examine the spatial properties (clumping, inhibition or randomness) of linear data that is based on using circular test lines as sampling units and then applying stochastic geometry theory to interpreting the resulting intersection points between data and test lines. The approach presented here combines two methods of spatial data sampling into a bootstrap resampling procedure in order to make inferences regarding the structure (the spatial pattern or lack thereof) of the linear data (mappings of *F. virginiana* stolons). It provides both an introduction to the technique and applies it to field data of *F. virginiana* surveyed in forest and open-field plots. The technique is applied to field mappings of *F. virginiana* stolons in three plots. The plots differed in the spatial structure of vegetation and vegetation density, and the biological hypothesis is that *F. virginiana* will exhibit different stolon spatial structure as a consequence of the difference in spatial structure of the forest and field plots (e.g., due to the density of neighbours and distribution of light).

2.2 Methods

2.2.1 Resampling analysis

2.2.1.1 Circular test lines

As noted in Chapter 1, straight test lines require an assumption of isotropic data. For example, in the extreme case of having all lines in a spatial pattern perfectly parallel to the straight test lines, the estimated intensity of the data would be zero because there would be no intersections between the test lines and the data. This situation can be overcome by employing what is essentially an isotropic test line, a circle.

Circles can be placed on the study plot by distributing points in x-y coordinate space according to a chosen model (e.g., random x and y values). Each point location is used as the centre for the circular test line of radius r that is defined by the equation of a circle centred on point (x,y) with radius r. The intersections between the linear data and the circular test lines can then be calculated and the resulting point pattern can be further analyzed.

Another method is to examine the region within the circle, rather than events (intersections) on its perimeter. A circle is defined as having radius r with its centre situated on the point as in the method above. The area of the circle thus serves as a sample region, and spatial elements inside this area can be recorded. Using the two-dimensional circle as a sampling device allows for the measurement of both one-dimensional lines and zero-dimensional points (Weibel, 1980). The total length of line inside the circle can be calculated and the points inside the circle can be enumerated. The first method, that of finding intersection points, will be presented in this chapter.

2.2.1.2 Estimating intensity of the stolons across the study plot

As noted in Chapter 1, sampling linear data with straight test lines requires the assumption of stationarity, that is isotropy and homogeneity. As noted above, circular test lines are robust in the case of anisotropic data. However, data might also be inhomogeneous (clumped or regular) which can also lead to inaccuracies in estimates when using regularly placed, straight test lines. This can be avoided by using a random circle process, C_f, to estimate the intensity of lines across the study plot (total linear length / total area). A random point process, P, is initiated over the study area, and circles of radius r are centered on the points. The portions of the circles contained within the study plot comprise the test line set, C.

The intersection points between the test line set and the linear data are then calculated. The linear data (plant stolon mappings) are stored as groups of lines segments called polylines. These polylines approximate closely the irregular curves present in biological data and greatly simplify computational methods. For each circle in the test line set, C, all polylines are iteratively checked for intersection points with the circle perimeter. The density of lines can then be found using a derivative of Eqn. 1.1 and Eqn. 1.3, Eqn. 2.1.

$$B_A = \frac{\pi}{2} \times \{\frac{\#\{C \bigcap \Phi\}}{C_L}\} \tag{2.1}$$

where,

B_A = the length of line per unit area

C = the test line set composed of circular test lines inside the study plot

C_L = the total length of circular test lines in the study plot

Φ = the set of linear data

2.2.1.3 Estimating intensity of other stolons around any one stolon

Randomly located circular test lines provide robust estimates of intensity, but they do not provide information about the second-order properties of the fibre process. In order to investigate the spatial region near a fibre (stolon), a different approach must be taken where sampling is constrained to the fibre of interest. It is possible to do this by centering the sampling circles on fibres. Using many circles will sample an envelope around the fibre with a width equal to the circle diameter (Figure 2.2).

A point process, Ps, is initiated randomly over the fibre process, Φ, and circles with radius r are centred on the points creating a circle process, Cs_f. The portion of Cs_f within the study plot is the test line set, Cs. The intersection between Φ and Cs, $\{Cs \bigcap \Phi\}$, is a collection of points which can be used to estimate B_A using Eqn. 2.1.

This method samples the portion of x-y space surrounding the segment or fibre in the range [0 to *r*]. Figure 2.1 shows a stolon with circles of radius *r* placed randomly along its length. This can be considered a constrained randomization of sampling points because the points are restricted to being placed on a fibre, but are otherwise randomly distributed along the fibre process as a whole. This means that this approach considers the positions of fibres relative to one another, and it will provide estimates of fibre intensity at various distances from any one fibre (see Figure 2.2).

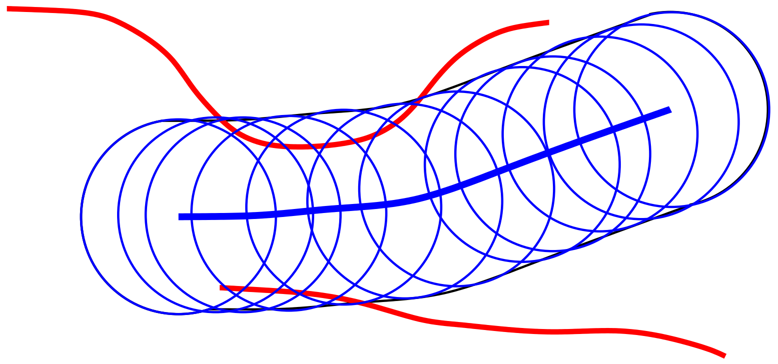


Figure 2.1: A fibre with circular test lines with radius *r* distributed along its length. With many test lines an envelope with radius *r* is approximated.

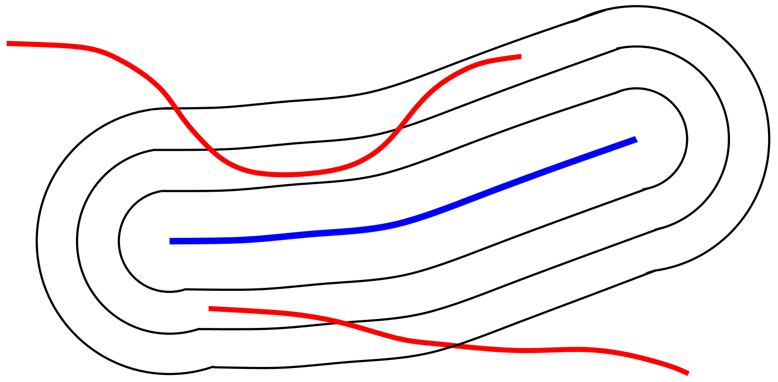


Figure 2.2: A fibre with envelopes at three radii that could be approximated by three sets of circular test lines.

2.2.1.4 Resampling Analysis

These two separate sampling methods, circles distributed randomly over the study area and circles distributed randomly over the fibre process (stolons) can be combined into a resampling analysis that can be used to investigate the second-order properties of the linear data.
The resampling analysis has the following steps:

1. Determine stolon intensity around stolons at a given radius

    i. Distribute N circles randomly on the stolons

    ii. Calculate B_r, the intensity of stolons within a radius r using Equation 2.2.

$$B_r = \frac{\pi}{2} \times I_{Lr} \tag{2.2}$$

where,

B_r = the intensity (length of line per unit area)

I_r = the intensity of intersection points between the linear data and circular test lines of radius r distributed randomly on stolons (# / unit length of test line)

2. Determine stolon intensity across plot

    i. Distribute N circles randomly on the plot

    ii. Calculate B_p using Equation 2.3 using the intersections of the linear data and the randomly distributed circles.

$$B_p = \frac{\pi}{2} \times I_p \tag{2.3}$$

where,

B_p = the intensity (length of line per unit area)

I_p = the intensity of intersection points between the linear data and circular test lines of radius r distributed randomly across the plot (# of points / unit length of test line)

3. Repeat Step #2 $NSIM$ times to produce a distribution of $NSIM$ B_p estimates for the study plot. This is an estimate of the average intensity of stolons within the plot given the sampling circle radius r. That is, it is centred on the total length of the stolons divided by the plot area.

4. Repeat 1, 2, & 3 for a range of circle radii $r = r_1, r_2, r_3....r_n$

5. Compare B_r alongside the distribution of B_p for each radii r.

Where B_r deviates from a sufficient proportion of the B_p distribution, say 95% of the distribution, it can be said that the estimated stolon intensity within a given radius of any stolon is significantly higher or lower than what we would expect if the stolons were distributed randomly. This answers the question, “Is the intensity around stolons, within a specific distance, greater than (clumping) or less than (inhibition) the intensity of fibres across the plot if they were distributed randomly?”

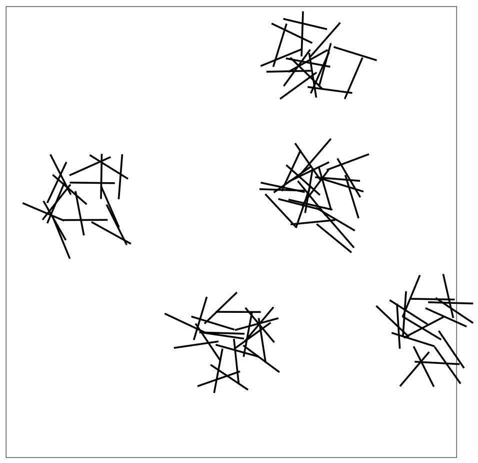
All analyses were conducted in the Python programming language using the *Linear* software library developed as a component of the thesis research (see Appendix B).

2.2.2 Artificial Data

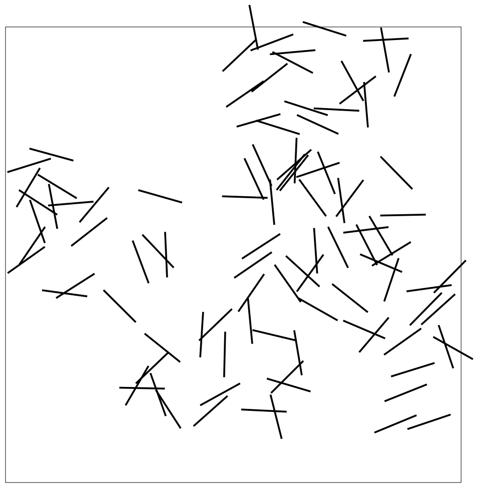
The analysis can be demonstrated using a set of test data (Figure 2.3). Several replicate sets of separate segment processes were created with varying degrees of heterogeneity. For each level of heterogeneity, a cluster point process was produced with randomly distributed parent points as the centres of circles of radius r_p units into which were distributed randomly distributed daughter points. The union of all daughter points compose the point process D. Each point in the process D is then used as the centre of a line segment having length one and an orientation that is random between $[0, 2\pi]$. Figure 2.3 shows the segment processes generated for $r_p = [1.0, 2.0, 4.0, 8.0]$. Five parent points were used for the circle centers, into each of which were placed 20 daughter points (segment centers) at random. A plot size of 10 x 10 units was chosen in order to provide a theoretical linear intensity of 1 unit / unit ^{2}.

2.2.3 *Fragaria virginiana* Data

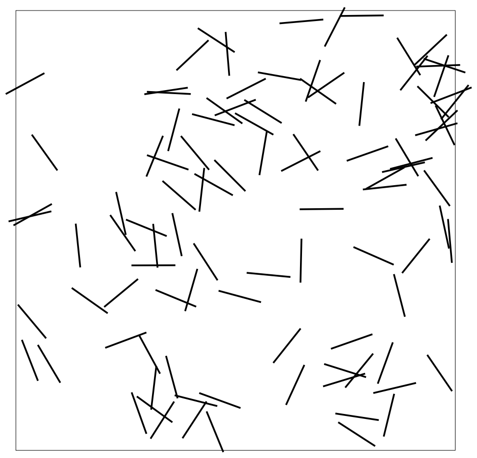
In 2005, three 2 m x 2 m plots were established at the Woodbend Research Forest, approximately 20 km southwest of Edmonton, Alberta. The area is characterized by mixed forest and old-fields with predominantly sandy soils. One plot was established in an abandoned field, and two plots were established in a nearby forested area. The field plot was characterized by grasses and other small herbs; the only shading at this site was from *F. virginiana* ramets and surrounding vegetation. The forest plots were characterized by a patchy, young *Populous tremuloides* overstory and a second layer of vegetation composed of broadleaf shrubs; the vegetation structure in the forest plots resulted in larger patches of light and shade. The plots were surveyed using a sampling frame that partitioned the plot



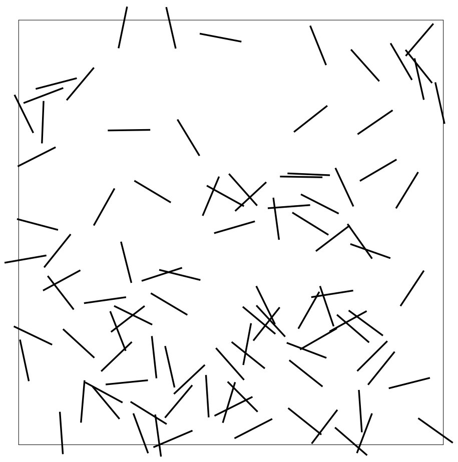
(a) Radius 1.0 units



(b) Radius 2.0 units



(c) Radius 4.0 units



(d) Radius 8.0 units

Figure 2.3: Cluster segment processes. Each process is generated using a set of five randomly located points as the centres for each cluster. Orientations are uniformly random from a von Mises distribution ($K = 0$).

into rows of 10 cm x 10 cm quadrants. A ruler was then used to measure ramet locations to the nearest millimetre. Each plant was marked in the field using a numbered pin inserted in the soil at the ramet base. The number of leaves, flowers, buds and longest leaf length was recorded for each ramet.

Plant location data was stored digitally as x-y point coordinates and printed to form maps of each plot. The stolons were then hand-mapped onto the printed map of ramets using the plant locations in the field and the sampling frame as guides. The analogue stolon maps were then digitized using vector graphics software and combined with the ramet location point data. The stolons were digitized into polylines, collections of straight line segments that approximate a smooth curve, which greatly simplified storage and computation. As discussed in Chapter 1, Stoyan et al. (1995) points out the practicality of treating smooth fibres as collections of short line segments.

Ramet locations were analyzed using point pattern analyses. The Spatstat package (Baddeley and Turner, 2005) and the R statistical system (R Development Core Team, 2011) was used to conduct the point pattern analyses. The $L(r)$ (transformed Ripleys K-function) and $g(r)$ (pair correlation function) statistics were calculated along with simulation envelopes, and the $\hat{L}(r)$ and $g(r)$ functions were plotted as a function of r. For each plot, this analysis was conducted for all the ramets, all ramets with flowers or buds, and all ramets with neither flowers nor buds. Separate analyses were conducted for plants with flowers/buds and for those without flowers and buds in order to characterize the spatial structure of these two groups of ramets. Of particular interest was whether flowering ramets showed clustering and/or inhibition at similar scales to the non-flowering ramets and, ultimately, if the spatial structure of sexual reproduction was similar to the spatial structure of some aspects of vegetative reproduction (i.e., stolon production).

Figure 2.4 shows the field plot mappings of *F. virginiana* ramets in Plots 1, 2& 3. Plot 1 (field) had 707 ramets and 54.79 m of stolons. Plot 2 (forest) had 163 ramets and 9.73 m of stolons. Plot 3 (forest) had 94 ramets and 4.96 m of stolons.

Each *F. virginiana* stolon mapping was approximated by a polyline, *p*, that was composed of connected line segments, *p*1, *p*2, *p*3 . . .. The resampling analysis was conducted for each set of *F. virginiana* mappings.

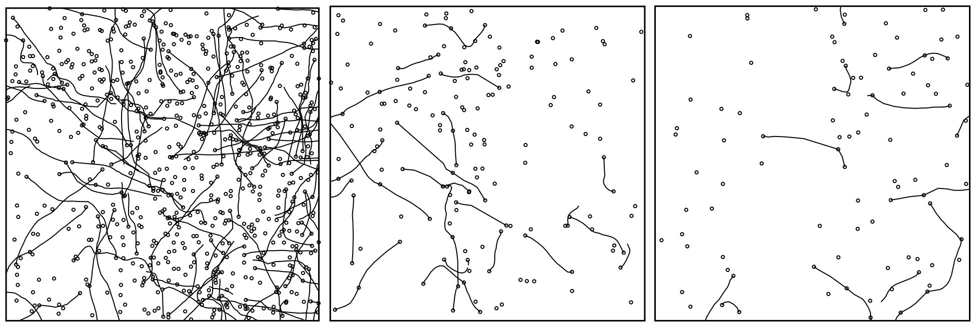

(a) Plot 1 (b) Plot 2 (c) Plot 3

Figure 2.4: The ramet locations and stolon mappings of *Fragaria virginiana* in Plot 1, 2 & 3 located at the Woodbend Research Forest. Plot size is 2 m x 2 m. Plot 1 had 54.79 m of stolon (*n ramets* = 707). Plot 2 had 9.73 m of stolon (*n ramets* = 163). Plot 3 had 4.96 m of stolon (*n ramets* = 94).

2.3 Results

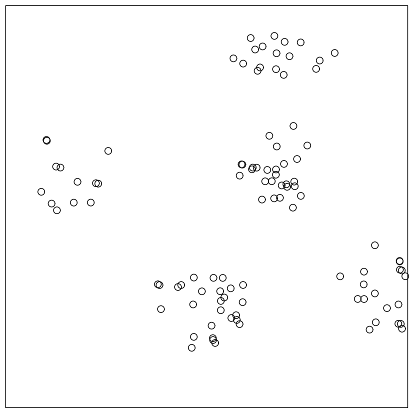
2.3.1 Artificial Data

Figure 2.5 (subplots 2.5(a) through 2.5(d)) show point processes produced by randomly locating points on the segment process in Figures 2.3(a) through 2.3(d) respectively. The size of point clusters increase as the size of segment clusters increase.

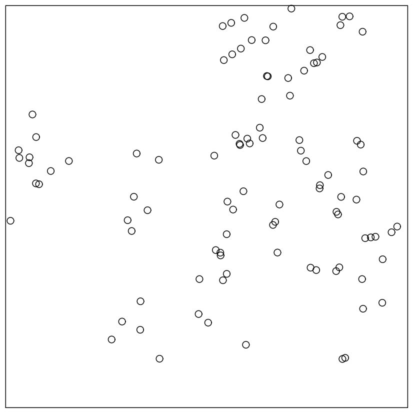
Figure 2.6 shows calculation of $L(r)$ simulation envelopes showing expected $L(r)$ (dashed), observed (solid) and CSR (grey) for the point patterns in Figure 2.5. The solid line represents $\hat{L}(r)$, and the grey region represents the envelope produced from 100 random simulations of the data using the envelope function of the Spatstat package (Baddeley and Turner, 2005). Subplots 2.6(a) through 2.6(d) coincide with subplots 2.3(a) through 2.3(d) respectively. As the radius of the segment clusters increases (from 1.0 to 8.0), $\hat{L}(r)$ of the fibre points correctly indicates reduced clustering of segments.

Figure 2.7 shows calculation of $g(r)$, the pair correlation function, and simulation envelopes showing expected $g(r)$ (dashed), observed (solid) and CSR (grey) for the point patterns in Figure 2.5. The solid line represents $g(r)$, and the grey region represents the envelope produced from 100 random simulations of the data using the envelope function of the Spatstat package (Baddeley and Turner, 2005). Subplots 2.7(a) through 2.7(d) coincide with subplots 2.3(a) through 2.3(d) respectively. Similar to $\hat{L}(r)$, $g(r)$ of fibre points correctly indicates clustering of segments at small segment cluster radii and no clustering at large radii.

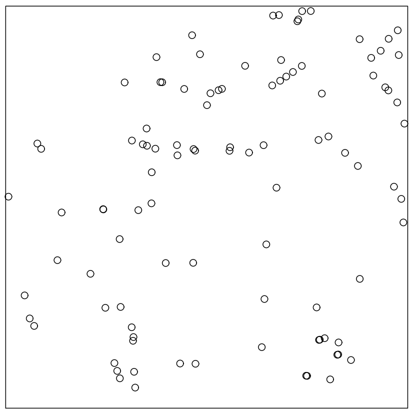
Figure 2.8 shows the intensity of segments estimated by fibre-centred sampling circles at radius r (solid line) plotted against the 90% interval of intensity estimates of segments from randomly placed sampling circles. For each radii, 100 samples of the data were conducted, each one using 100 sampling circles. The circle sampling clearly indicates clustering (higher intensity of segments than expected) at smaller segment cluster radii, and it correctly shows randomness at the largest cluster size (8.0 units).



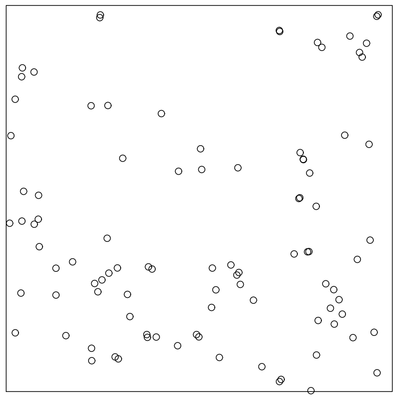
(a) Radius 1.0 units



(b) Radius 2.0 units

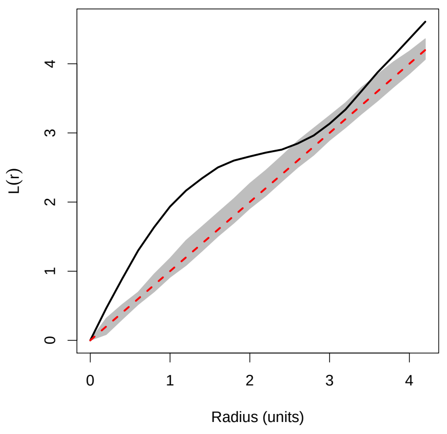


(c) Radius 4.0 units

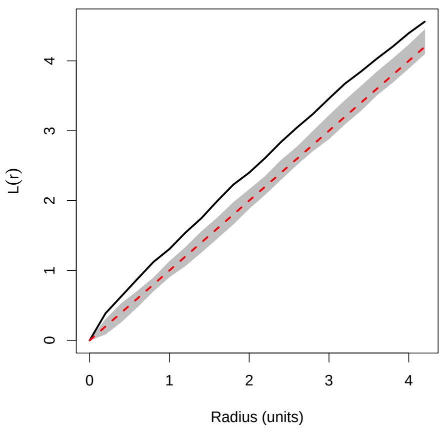


(d) Radius 8.0 units

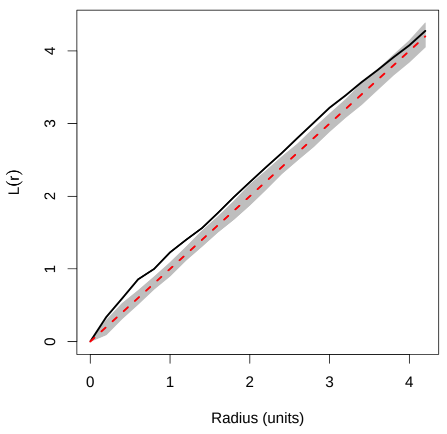
Figure 2.5: Subplots 2.5(a) through 2.5(d) are point processes produced by randomly locating points ($n = 100$) on the segment process in Figures 2.3(a) through 2.3(d) respectively.



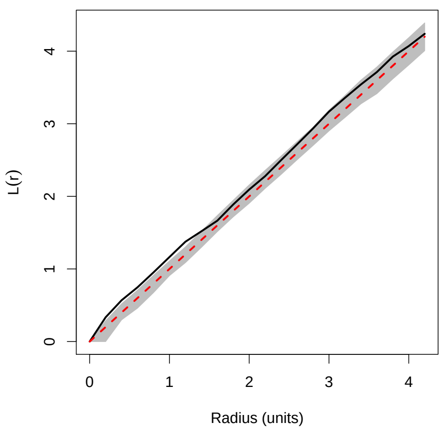
(a) Radius 1.0 units



(b) Radius 2.0 units

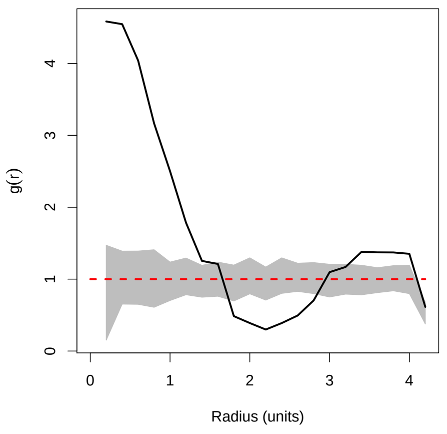


(c) Radius 4.0 units

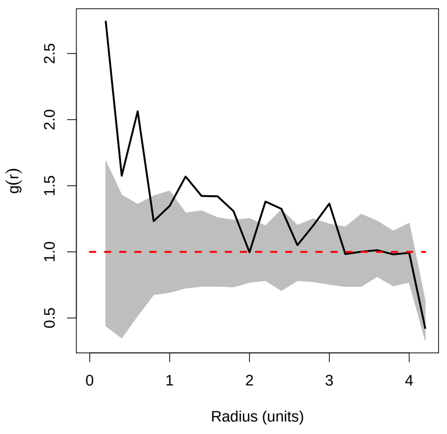


(d) Radius 8.0 units

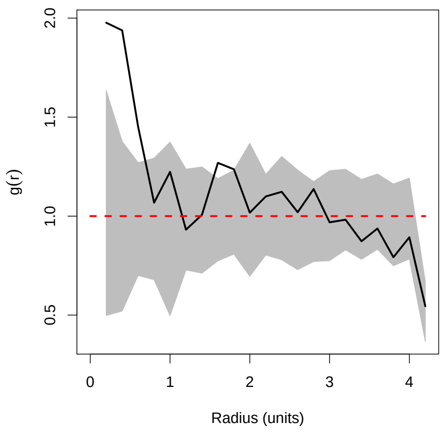
Figure 2.6: Calculation of $L(r)$ simulation envelopes showing expected $L(r)$ (dashed), observed (solid) and CSR (grey) for the point patterns in Figure 2.5. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR realizations. Subplots 2.6(a) through 2.6(d) coincide with subplots 2.3(a) through 2.3(d) respectively.



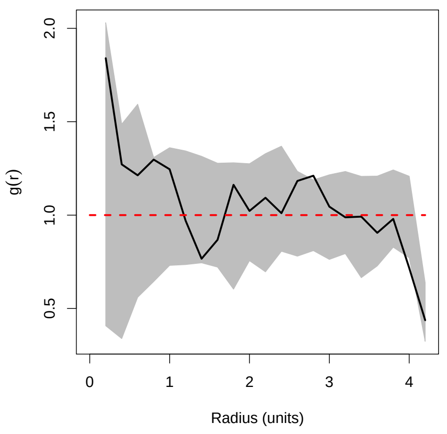
(a) Radius 1.0 units



(b) Radius 2.0 units

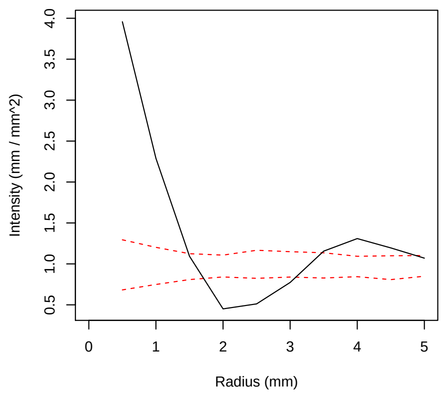


(c) Radius 4.0 units

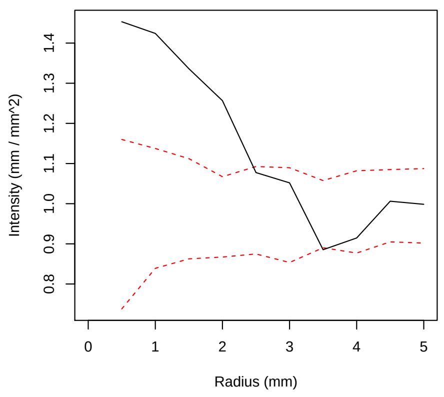


(d) Radius 8.0 units

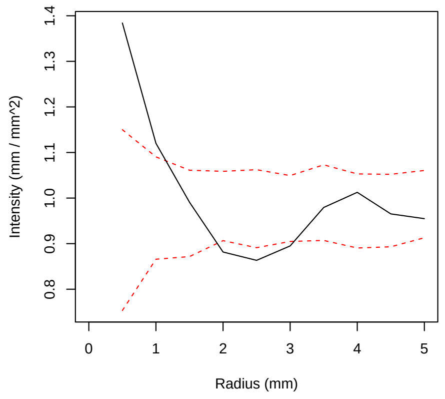
Figure 2.7: Calculation of $g(r)$, the pair correlation function, and simulation envelopes showing expected $g(r)$ (dashed), observed (solid) and CSR (grey) for the point patterns in Figure 2.5. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR realizations. Subplots 2.7(a) through 2.7(d) coincide with subplots 2.3(a) through 2.3(d) respectively.



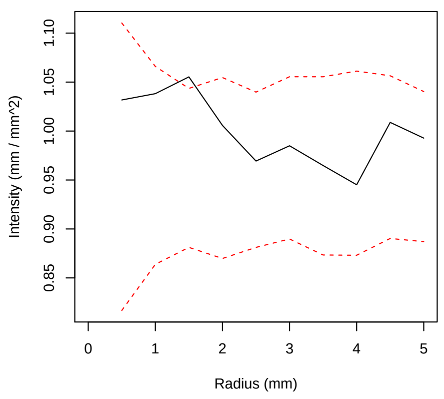
(a) Radius 1.0 units



(b) Radius 2.0 units



(c) Radius 4.0 units



(d) Radius 8.0 units

Figure 2.8: The intensity of segments estimated by fibre-centred sampling circles at radius *r* (solid line) plotted against the 90% interval of intensity estimates of segments from randomly placed sampling circles. For each radii, 100 samples of the data were conducted, each one using 100 sampling circles.

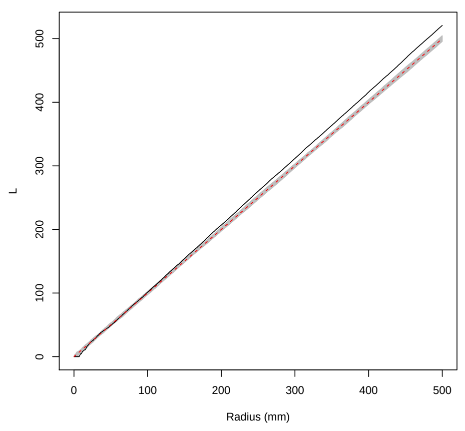
2.3.2 *F. virginiana* Data

2.3.2.1 Point Pattern Analysis

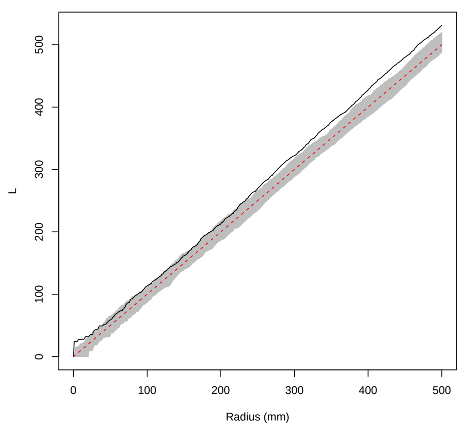
Figure 2.9(a) shows the point pattern analysis for Plot 1 (field). The solid line represents $\hat{L}(r)$, and the dashed lines represents the envelope produced from 100 random simulations of the data using the envelope function of the Spatstat package (Baddeley and Turner, 2005). Deviations above the envelope indicate overdispersion of the observed ramet locations, while deviation below the envelope indicates clumping (underdispersion) of the ramet locations. The analysis indicates significant inhibition at very small scales (<25 mm) and significant clumping at radii above 140 mm. Figure 2.9(b) shows the point pattern analysis for Plot 2 (forest). The only significant clumping occurs at small radii (<25 mm) and above 300 mm. Figure 2.9(c) shows the point pattern analysis for Plot 3 (forest). No significant clustering or inhibition was detected, suggesting that the points are distributed at random.

Figure 2.10 shows $L(r)$ calculated for all ramets that had either flowers and/or buds in Plots 1 and 2. Plot 3 did not have ramets with flowers or buds. Plots 1 and 2 differed in their arrangement of ramets with flowers and/or buds. Ramets in Plot 1 (Figure 2.10(a)) were clumped at distances greater than 200 mm. Ramets in Plot 2 (Figure 2.10(b)) were randomly distributed at all radii.

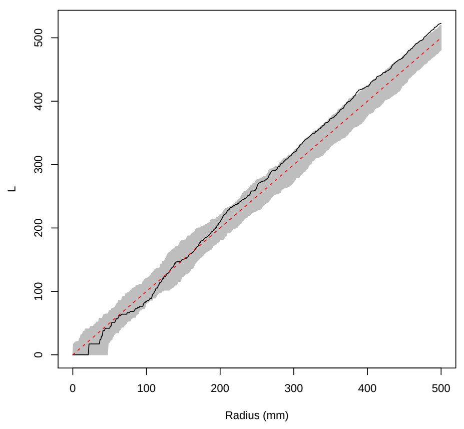
Figure 2.11 shows $L(r)$ calculated for ramets that had neither flowers nor buds in Plots 1, 2, and 3. Ramets in Plot 1 (Figure 2.11(a)) were randomly distributed at distances less than 300 mm but clumped at distances greater than 300 mm. Ramets in Plot 2 (Figure 2.11(b)) showed a tendency toward randomness between radii 50 mm and 200 mm but were clumped at radii above 200mm. Ramets in Plot 3 (Figure 2.11(c)) were randomly distributed at all scales.



(a) Plot 1

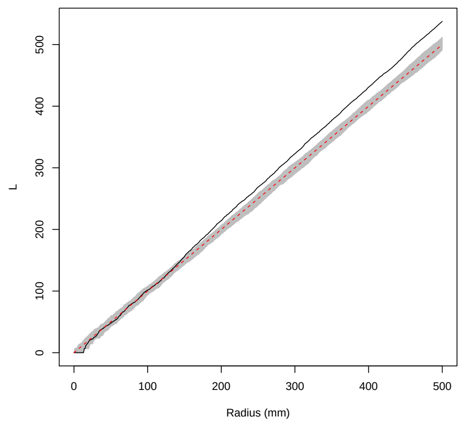


(b) Plot 2

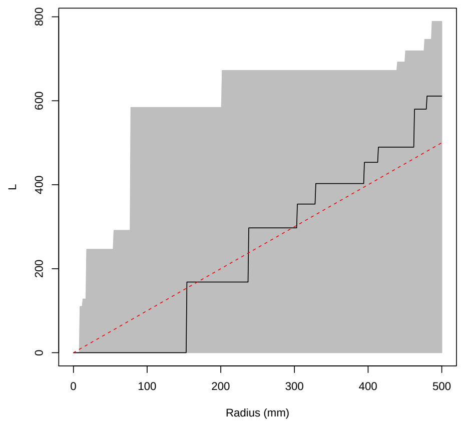


(c) Plot 3

Figure 2.9: $L(r)$ calculated for all ramet locations in Plots 1, 2, and 3. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR realizations.

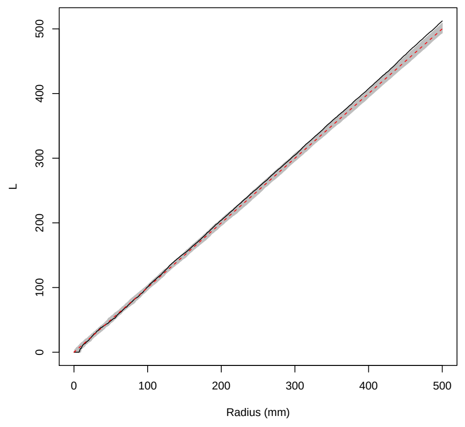


(a) Plot 1

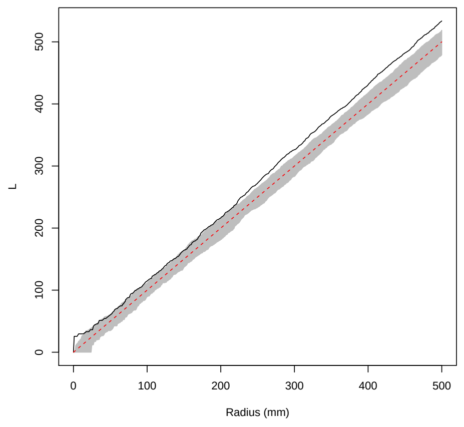


(b) Plot 2

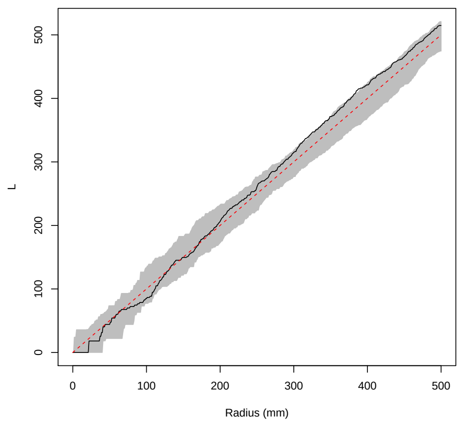
Figure 2.10: $L(r)$ calculated for all ramets that had either flowers and/or buds in Plots 1 and 2. Plot 3 did not have ramets with flowers and/or buds. For Plot 1, $n = 247$. For Plot 2, $n = 10$. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR realizations.



(a) Plot 1



(b) Plot 2



(c) Plot 3

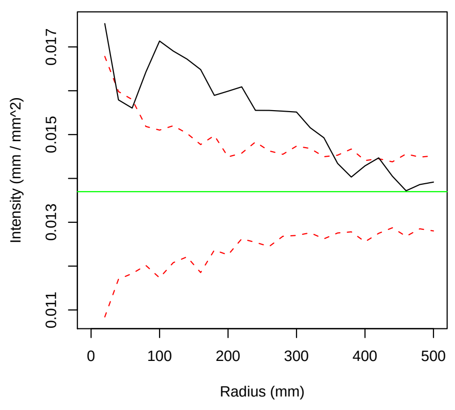
Figure 2.11: $L(r)$ calculated for ramets that had neither flowers nor buds in Plots 1, 2, and 3. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR realizations.

2.3.2.2 Linear Structure Analysis

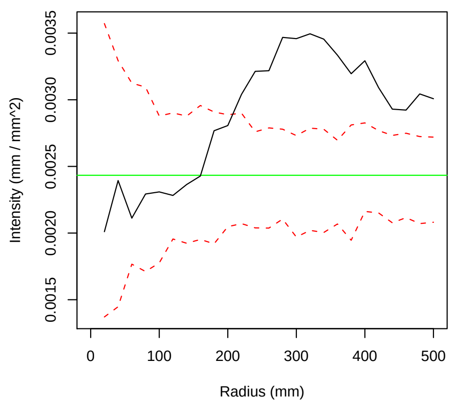
Figure 2.12(a) shows the resampling analysis for Plot 1 (field). Note that the theoretical stolon intensity ($length/m^2$) is represented as a horizontal line. The dashed line represents the upper and lower 90% levels for the estimation of the plot intensity using sampling circles placed randomly over the plot (expected intensity). The stolon-to-stolon intensity is highest at lower radii, and for radii between approximately 60-350mm, it is above the estimated theoretical (expected) intensity, indicating clustering. At radii beyond 350 mm, the stolon-to-stolon intensity is similar to the expected intensity.

Figure 2.12(b) shows the resampling analysis for Plot 2 (forest). At radii smaller than approximately 200 mm, the stolon-to-stolon intensity is not outside the range of observations that might occur if the stolons were arranged randomly. At radii above 200 mm the stolon-to-stolon intensity indicates significantly higher intensity than the expected, indicating clustering at these scales. The maximum intensity occurs at a radius of approximately 325 mm.

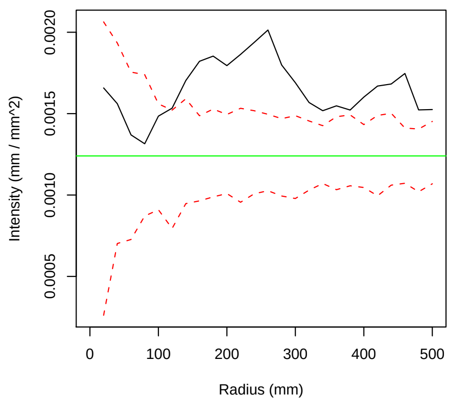
Figure 2.12(c) shows the resampling analysis for Plot 3 (forest). At radii greater than 100 mm, the stolon-to-stolon intensity indicates clustering of stolons, with a maximum intensity at a radius of approximately 250 mm.



(a) Plot 1



(b) Plot 2



(c) Plot 3

Figure 2.12: A resampling analysis for Plots 1, 2, & 3. The solid line is the intensity (mm/mm^2) of stolon at radius *r* from any given stolon estimated using circles placed randomly on the stolons. The dashed line represents the upper and lower interval containing 90% of the estimates of stolon intensity using circles of radius *r* placed randomly across the plot. The horizontal line is the exact average intensity of the plot.

2.4 Discussion

The analysis of the segment processes clearly shows that both the point pattern analysis of fibre points and the resampling analysis are effective methods of characterizing the scale of clustering of linear data.

The circle sampling method reliably indicates the size of clusters up to a segment cluster size of radius=2.0 units, and it suggests some clustering in the case of cluster sizes of radius 4.0 units. It correctly fails to detect clustering for cluster sizes over 4.0 units where the process is producing a spatial structure that is increasingly random. For example, a cluster size of 8.0 produces a segment process that is effectively random across the plot.

The point pattern analysis ($L(r)$ and $g(r)$) of random fibre points also provided reliable indication of the size of clusters. The $L(r)$ statistic most effectively detected the clustering of points, and thus fibres. The $g(r)$ function indicated clustering, but it also showed some ability to detect the repulsion of the clusters of segments. The $g(r)$ statistic provided a more useful characterization of the clustering and inhibition. The $g(r)$ statistic calculated over random fibre points produced results similar to that of the circle sampling analysis.

In the circle sampling analysis, the variation in the estimated theoretical intensity is highest at small circle radii. This is due to the relatively short total test line length in smaller diameter sampling circles. Although the total test line length changes with circle radius, the circle randomization simulation takes this into account by using approximately the same test line length for each simulation for each radius. By producing a distribution of theoretical estimated intensities, it is possible to compare the actual stolon-to-stolon intensity with confidence.

The hypothesis that *F. virginiana* would exhibit different stolon spatial structure as a consequence of the difference in spatial structure of the forest and field plots was tested. The analysis of *F. virginiana* stolon data shows that, regardless of plot, the intensity of stolons with respect to surrounding stolons is not homogeneous in space. The plots, however, differed from each other in terms of at what radii there was clustering or inhibition. The field plot (Plot 1) had ramets with stolons that tended toward clustering of stolons at smaller radii, while the forest plots (Plots 2 and 3) tended to have ramets with stolons that tended toward clustering of stolons at larger radii. The field plot also differed from the forest plots in that the stolon-to-stolon intensity declined as r increased. The forest plots showed lower intensity at lower r, but significantly higher intensities for larger r. It is possible that differences between the forest and field plots resulted in different responses in the *F. virginiana* spatial structure.

Clonal plant research has received considerable attention over the past twenty years (Sammul et al., 2008). Clonal plants are characterized by the ability to reproduce asexually, and they often have above-ground and below-ground stems, stolons or rhizomes respectively, that place daughter ramets distally from the parent ramet. Usually the physical stolon or ramet connection is maintained for a period of time, whereby the parent plant and daughter ramets are physically connected. In addition to the ramets being physically connected, it is well-known that there can be physiological integration within the clone/genet. There is little doubt that the spatial aspect of clonal growth and linear connections between ramets is important when exploring pertinent ecological questions regarding clonal plants. Plants that use stolons and rhizomes to expand into their surroundings, sequester resources (light, nutrients, space, water), transmit materials, and/or distribute ramets have an important spatial aspect to their architecture. The genet becomes a network of connected ramets that allows for clonal integration. This can have important ecological impacts. For example, both nutrient and water transport between ramets has been found in *Carex arenaria* and *C. disticha* (D'Hertfeldt and Falkengren-Grerup, 2002). *Trifolium repens* has been shown to exhibit a spatial division of labour in heterogeneous environments (Stuefer et al., 1996), an indicator of physiological integration. The possibility of physical ramet connections acting as pathways for signaling between ramets during herbivore attack has seen recent discussion (Stuefer et al., 2004). Integrated clones of *Potentilla simplex* also elongate their stolons more so than clones that were not integrated, suggesting, possibly, that integration could allow for movement away from unfavourable locations (Wijesinghe and Handel, 1994). It might also suggest an improved ability for the connected ramets to explore their environment more rapidly, leading to the improved location and acquisition and utilization of resources. *Potentilla simplex* has also been found to benefit from clonal integration in heterogeneous environments, producing more biomass when integrated than when separated (Wijesinghe and Handel, 1994).

The resampling analysis seems to suggest a fundamental difference between field and forest plots in terms of stolon spatial structure. Clustering of *F. virginiana* stolons occurs at a range of scales in both field and forest plots (but not quite the same scales). *F. virginiana* ramets in forest plots seem to have a tendency toward increased clustering at larger radii, while ramets in the open plot seem to show reduced clustering at larger scales. A ramet stolon response to the spatial structure of the surrounding vegetation in the forest and field plots may explain the difference between forest and field plots.The forest plots were characterized by small shrubs and broadleaved herbs, along with a patchy *Populous tremuloides* overstory. This vegetation structure would create a larger scale heterogeneous distribution of light resources compared to the open field plot. *Fragaria* is a genus where clonal growth and physiological integration are important components of its ecology. For example, Alpert and Mooney (1986) have found that clonal integration of *F. chiloensis* ramets increased survival under adverse conditions. *F. chiloensis* has been found to segregate roots between connected ramets, which suggests that integration of clones allows for signal transmission between ramets that allows root separation (Holzapfel and Alpert, 2003).

Wild strawberry species include *Fragaria vesca, F. chiloensis,* and *F. virginiana*. In particular, Wild Strawberry, *F. virginiana,* has an extensive range across North America. It is also a common forest herb that can be found in sites ranging from clearings, such as dry beaver ponds, to heavily forested areas. *F. virginiana* is one species of octoploid strawberry from a genus that has an expansive range across North America and around the world. *Fragaria* spp. are important agricultural species, and wild strawberries serve as a rich genetic resource that can be tapped by the commercial strawberry industry (Hancock and Luby, 1993). While a range of *Fragaria* sub-species in North America have been studied (Hancock and Luby, 1993), more recent investigations have suggested that at least some sub-species designations may not be required (Hancock et al., 2004). Harrison et al. (1997) have examined the morphological and molecular variation among 37 populations of *F. virginiana* and *F. chiloensis* in N. America. They concluded that *F. virginiana* and *F. chiloensis* likely share a common ancestor and that the mesic properties of sites were responsible for driving the differentiation (Harrison et al., 1997). Hancock et al. (2004) examined the 220 genotypes of American strawberry and found that, while morphological and genetic variations warrant the *F. virginiana* and *F. chiloensis* designations, many subspecies designations should no longer be recognized. In fact, they conclude that *F. virginiana* and *F. chiloensis* might actually be distal forms of the same species.

Fragaria spp. genets propagate ramets by stolon growth. The stolons remain connected for the growing season, forming a genet that spatially is composed of ramet locations and stolon connections. Evidence suggests that *Fragaria* spp. forage for resources via clonal growth. For example, *F. vesca* has been found to forage for resources in heterogeneous environments (Roiloa and Retuerto, 2006c). This means that the spatial structure of ramet locations and of stolons could be reflective of responses of clonal expansion to environmental heterogeneity (e.g., light, competition, space, nutrients).

Stolons and rhizomes expand into environments that are almost always heterogeneous. As such, the environment they encounter might generate a response by the genet to alter its strategy. That is, there will be a link between the pattern to ramets and stolons that relates to the underlying ecological processes. Changes in reproductive strategy have been documented. For example, using a combination of spatial mapping and genetic analysis, Wilk et al. (2009) found *F. virginiana* reproductive patterns to have a high degree of variation between sites. Further, there is some evidence that *F. virginiana* is capable of sex-allocation plasticity resulting in variation in hermaphrodite sex-expression (Bishop et al., 2010). In addition, the responses of clonal growth to environmental conditions varies by species and even within a species. As such, it's important that methods for examining the spatial aspects of clonal growth continue to be developed and applied to field and experimental data.

Further complicating the spatial study of vegetation is the need to account for variations in density and spatial structure. The density of ramets can vary from location to location. For example, Angevine (Angevine, 1983, 1981) studied the demography of natural populations of *F. vesca* and *F. virginiana* at three sites. He found that the local conditions of the site had a greater influence on local demography than the identity of the species and that there was a great deal of variation between the sites. Angevine (1983) found a range of *Fragaria* spp. ramet densities from 1.8 $m^{-}2$ to 340 $m^{-}2$. In addition, *F. virginiana* produced higher proportions of stolon biomass in low density field and wooded sites than higher density sites (Holler and Abrahamson, 1977). It is unclear how such variation in stolon biomass might alter the spatial arrangement of stolons.

2.5 Conclusions

The resampling analysis was able to characterize the spatial structure of stolons in terms of clustering, inhibition, or randomness, and it is a promising method for characterizing mappings of linear data. The analysis showed that the spatial structure of *F. virginiana* stolons differed between forest and field plots. It is possible that the spatial structure of resources (light and/or space) in the two sites resulted in foraging responses in the *F. virginiana* genets, and the differences between the two sites with respect to the distribution of these resources could explain the difference in stolon spatial structure.

Chapter 3

Characterization of the directional properties of vegetation structure

3.1 Introduction

The examination of linear data in ecology involves examining not only the density of lines per unit area but also the directional properties of the system. This differentiates fibre processes from point processes in that point processes are characterized only by location in one, two or three dimensions. Fibre processes, however, have both locational and directional properties. In order to understand the properties of a fibre process and consider the ecological implications, methods to assess the directional properties of a system are critical. For example, Sampaio et al. (2004) have found evidence of directional growth as a means of habitat selection in *Aechmea nudicaulis*. Also, *Clintonia borealis* has been found to invade territory in a wedge-shaped fashion due to a combination of internode length and branching angle (Angevine and Handel, 1986).

Circular statistics have been used to examine the orientation of plant segments (stolons) (Macek and Lepš, 2003). However, for the most part, systems that are anisotropic are problematic because they violate the assumptions of many statistical methods (Stoyan et al., 1995). In addition, non-parametric methods exist to estimate the directional distribution of line and fibre processes, but these also assume the process to be stationary (Kiderlen, 2001). Although, there are methods to examine anisotropic linear structures (Cruz-Orive et al., 1985), the initial stages of analysis require an initial characterization of the anisotropic tendencies of the data. The realities of ecological field data, however, call for practical methods of data exploration. Examining methods for both raster and vector data is useful in that it not only illustrates the strengths and weaknesses of these two data types, but it also provides a suite of tools for application in ecology.

Working with raster data presents special challenges and benefits. One benefit is the potential collection of linear data using a digital imaging device (e.g., digital camera). Digital photographs have been used to study spatial structure of vegetation (Zehm et al., 2003). Donnelly et al. (1995) used pixel-based methods to examine the spatial properties (biomass, growth, fractal dimension) of mycelial systems. Chadœuf et al. (2000) provide methods for examining the dependence between two spatial processes using digital images. They consider plots with mixtures of spatial structures (e.g., points, fibres, and areas). Karkkainen et al. (2001) have also shown that it is possible to estimate the orientation distribution of a fibre process using test lines.

This chapter introduces methods to characterize the directional properties of linear data stored in either raster and vector formats. It demonstrates the methods on artificial data, then applies them to sets of field data that illustrate its application. The first set of field data are from a transect of plots where *F. virginiana* stolons were mapped across a forest to field transition zone. The hypothesis is that the directional properties of *F. virginiana* stolons will vary across the forest-field edge, potentially as a consequence of light availability due to variation in vegetation structure along the transect. The second set of data are from a transect of plots where conifer needles were imaged across a gradient of hillside slopes. Here the hypothesis is that the needles will show a preferred direction downslope, potentially as a consequence of physical processes.

3.2 Methods and Data

Methods to extract and characterize the directional attributes of linear data are critical to characterizing any preferred orientation of structure. Quantification of the anisotropy can then inform further analysis or be interpreted in its own right.

The angular distribution of segments and fibres can be measured readily from vector data where fibres are represented as collections of straight lines. Basic geometry can be used to obtain an orientation from each straight line segment, and these orientations will form the new data required to form a distribution of orientations. More challenging, however, is the detection of a preferred orientation when the data are in a raster format.

Figure 3.1 illustrates one method of investigating anisotropy in linear structure. An anisotropic process (in this case line segments) will have a preferred angle (a) and a second angle of variation (b). The preferred angle can be estimated by using test lines to intersect the spatial structure and recording the number of intersections. The test lines (or the structure) are rotated and the process repeated. The number of intersections will be a maximum when the test lines run perpendicular to the preferred angle and a minimum when the test lines run parallel to the preferred angle. A plot of intersection counts will give the estimate of this preferred angle.

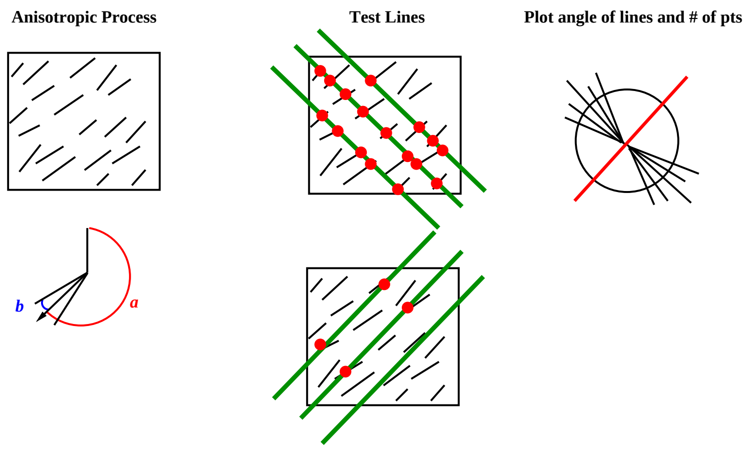


Figure 3.1: Illustration of the examination of anisotropic structure using test lines. Left: An anisotropic process with a preferred angle (a) and level of variation in the distribution of angles around *(a)* (b). Centre: Test lines with intersection points aligned perpendicular and parallel to the preferred direction of segments. Right: The expected number of intersections between straight test lines and the segment process on a circular plot.

3.2.1 Angular distribution

One method to examine the angular distribution is to produce a rose of directions based on fibre orientations measured directly from the data. For smooth fibres this can be a challenge, but sampling the fibre orientation randomly along the fibre can provide a means of obtaining a distribution of orientations. Such an approach can be used in the case of both vector and raster data, although these data types require different implementations.

If the data are stored in vector format, it is possible to employ the method presented below. Fibres are stored as collections of straight line segments. The orientation of the segment is determined in radians where a value horizontal to the right is zero radians and counterclockwise π radians is horizontal to the left. In this approach, the fibre process, Φ, containing n fibres, gives rise to a number of line segments, Φ_{seg}, each with its own orientation, θ. The distribution of orientations, Θ, can then be represented as a rose of directions that uses bins to represent the frequency of orientations. Although this method will provide some measure of the orientation of the system it does not take into account the length of the line segments that comprise the fibres.

A variation of the method above is to create a distribution of orientations whereby each unit length of fibre in Φ is represented by its orientation. This can be achieved by determining the length, l, of each segment comprising Φ and appending its orientation, θ, to the distribution of orientations, Θ_L, for each unit length. This is analogous to dividing all the segments that comprise the entire fibre process, Φ, into segments of unit length so that each new segment has the same length, resulting in a new collection of segments, S. Then, create Θ_L from the orientations of S. Θ_L can then be plotted on a rose of directions where the distribution of orientations is placed into 18 bins spanning 180 degrees (π radians). The length of the bin is representative of the total length of line having an orientation within the range delineated by the bin.

This method takes into account the length of fibres, ensuring that the rose of directions of Θ_L reflects the length of line in any one range of orientations.

3.2.2 Rotational analysis

In some cases, particularly in the case of raster data, it may be difficult to measure directly the orientation of lines. In the case of an isotropic fibre process, the properties of the process will remain stationary under rotations. Further, estimation of B_A using straight test lines requires the assumption that the data are isotropic. This also means that estimates of the linear intensity, B_A, made with straight test lines, which are sensitive to anisotropy, will be constant as the data are rotated. Conversely, the properties of an anisotropic process will vary under rotation. This allows for the characterization of directional properties using straight test lines and a rotational data transformation. It is possible to assess the anisotropy of a fibre process, Φ, by rotating the process through a series of discrete rotations, $\theta = \theta_1, \theta_2, \ldots \theta_N$. Figure 3.2 shows a segment process rotated 0, 30, 50, and 80 degrees. After each rotation, the new fibre process is sampled using straight test lines and the B_A is estimated using Equation 1.2 or 1.1. This will produce a series of length intensity estimates, $B_A(i, N)$, one for each rotation. If the process is isotropic the estimates $B_A(i, N)$ will show little variation. If the process is anisotropic, $B_A(i, N)$ will have a minimum when the test lines are aligned parallel to the majority of the fibres and a maximum when aligned perpendicular to the majority of fibres.

Figure 3.3(a) shows an example of a segment process generated using the von Mises distribution of angles. The von Mises distribution can generate angular distributions ranging from isotropic ($K = 0$) to parallel (K$\rightarrow \infty$). However, it is important to note that K produces anisotropic behaviour at relatively low numbers. For example, at $K = 10.0$, a distribution of angles is strongly anisotropic. The radial plot (Figure 3.3(b)) shows the results from sampling ten separate, isotropic

(K = 0) segment processes (n = 100 segments, a total segment (linear) length = 100 units). Each segment process was rotated in ten degree intervals and the number of intersections between segments and parallel test lines was plotted on the radial plot.

Because the isotropic segment processes are stationary under rotations, the estimations of B_A using test lines is consistent when the data are rotated. This means that B_A will be similar regardless of the angle of rotation, and the radial plot of B_A against rotation angle will be circular.

Each of the ten segment processes was also sampled using a lattice of test lines as a means to estimate the actual length of segments in the plots (the estimated boundary length). The lattice sampling method allows for an acceptable estimation of linear length in the case of anisotropic data. Although the lattice method is used here as a demonstration, sampling circles can also be used and provide a more robust estimation in the case of anisotropic data. Estimating the actual length of lines in the plot is important because with a good estimate of boundary length it is possible to construct Monte Carlo simulations to examine other properties of the spatial structure while holding length constant. The boundary length (B=96 units for Figure 3.3(b)) is the mean for the ten segment processes. Although this is lower than what might be initially expected of a segment process having 100 segments that are one unit long, the segments were allowed to extend over the plot boundary to ensure a homogeneous process across the plot. This results in a total length of line in the plot of 96 units, which is accurately reflected in the lattice-estimated length.

3.2.3 Data

3.2.3.1 Artificial data: Segment processes

A suite of homogeneous segment process models having varying levels of anisotropy were used to create reference systems to demonstrate and test the methods presented in this chapter. Points (N=100) from a random point process were used as centres for line segments of length=1.0 unit in a 10.0 x 10.0 unit study plot. The orientation of each segment was drawn at random from the Von Mises distribution with a preferred angle of zero (horizontal). Segment models ranged in their degree of anisotropy from $K = 0.0$ (isotropic) to $K = 10.0$ (strongly anisotropic) (see Figure 3.4 and Figure 3.10 for example segment processes). These segment processes having known directional distributions were then examined using the methods discussed above. These processes were created and analyzed using the Python programming language and the *Linear* software library (see Appendix B).

3.2.3.2 Field data: *F. virginiana* stolons

Seven contiguous, 2m x 2m, quadrats were sampled in the Woodbend Research Forest in 2006. The plots were labelled Plot 2, 3, 4, 5, 6, 7, & 8. The transect traversed the boundary between a *Populous tremuloides* stand and an abandoned field. Plot 2 was situated in the *P. tremuloides* stand and Plot 8 was situated in the abandoned field. The x-y coordinates of 583 *F. virginiana* ramets were recorded to the millimetre and the stolons were mapped using the methods described in Chapter 2. The mappings were then digitized using vector drawing software. Table 3.1 shows the data collected for each plot, but primarily analysis of the stolon mapping data is being reported in this chapter.

Table 3.1: Data collected from seven contiguous quadrats (2 m x 2 m) that traversed an oldfield-forest edge. $n = 583$ *F. virginiana* ramets

Data Type	**Data**	
	Data Type	**Units**
F. virginiana	$x - y$ location	mm
	leaves	#
	longest leaf length	mm
	stolon	#
	stolon	position mapping
	flowers	#
	buds	#
Herbs	proportion	Braun-Blanquet
Trees	DBH	mm

3.2.3.3 Field data: conifer needles

During the autumn of 2006, fallen pine (*Pinus strobus* and *P. resinosa*) and spruce (*Picea glauca*) needles were sampled using a digital camera (see example image in Figure 3.5(a)) at forested sites approximately 20 km east of Fort Frances, Ontario. Belt transects (one metre in length with 10 cm x 10 cm contiguous quadrats) were positioned along hillslopes and a single digital image was collected from each quadrat. The image was collected with the camera suspended one metre above the needle surface on a custom built sampling pod. A total of 19 transects were established on terrain with slopes ranging from 0 to 43 degrees. Twelve transects with unique slope angles were analyzed. Needles were assumed to be independent. However, *P. resinosa* and *P. strobus* needles are in bundles of 2 and 5 needles, respectively. This means that the needles of these species are not fully independent. This effect was assumed to be minimal due to the narrow nature of most needle bundles, and it was assumed that these bundles would respond to slope similarly to single needles.

Conifer needles provide a convenient system for examining the effect of physical factors (in this case, slope) on a stochastic process (the arrangement of fallen needles). They also provide a suitable system with which to demonstrate methods of analysis suitable for examining field data collected in the raster format. They produce a spatial structure characterized by high length intensity (high length of needle per unit area) which results in poor distinction of individual needles due to overlapping. This makes conversion of the data into vector format difficult, and it makes the application of raster data more practical. At a very small scale, the arrangement of needles creates a range of structures that may be important to the ecology of soil fauna. For example, needles with an anisotropic arrangement tend to pack more tightly, whereas needles arranged in many directions tend to be more loosely packed with more gaps and airspaces. Analysis of images of conifer needles is only one example of where this case might arise. Other examples include blow-down and the roots and shoots of vegetation.

As discussed in Chapter 1, image processing can be used to isolate linear data. Figure 3.5(a) shows a colour image representative of the image samples collected along the transects. Two potential methods of image processing were considered. The first was analyzing images that had been converted to greyscale (Figure 3.5(b)). The second method involved analysing images that had first been colour-processed to enhance the needle data (Figure 3.5(c)) and then converted to greyscale (Figure 3.5(d)).

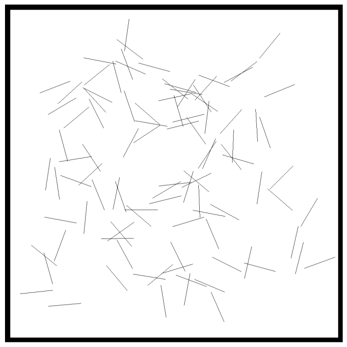
Figure 3.6 shows the pixel values of a test line sampling greyscale and colour-transformed images (Figure 3.6(a) and 3.6(b), respectively).

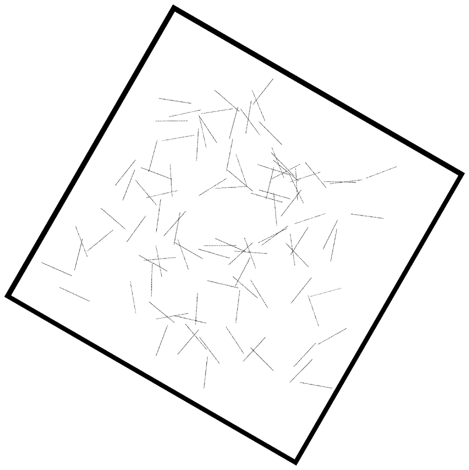
Greyscale transformation provides the most information along the test line. Using a test line to sample a greyscale image with no other pre-processing, however, produces a noisy signal (Figure 3.6(a)). In addition to working with greyscale images, it is possible to colour-process the images (Figure 3.5(c)) in such a way that the desired data remains in the image and any extra signal is removed. The images can then be converted to a greyscale image and obtain signals using test lines (Figure 3.6(b)). Colour processing provides a means to reduce the image complexity and better isolate the maxima that indicate needle centres. This approach was used for the data analysis in this chapter.

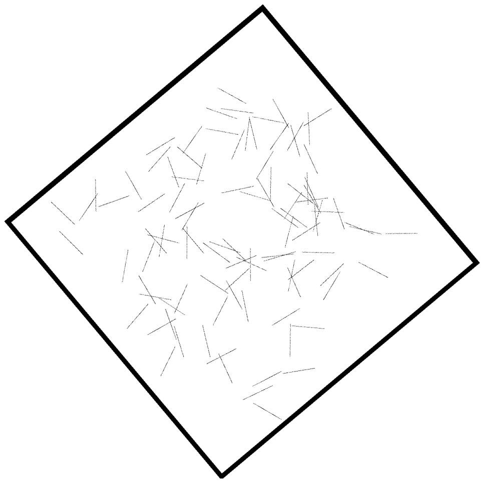
Colour processing involved examining each pixel in the RGB colour image of the needle plot. If the pixel had R, G, and B values within specific ranges, it was left untouched. If the pixel had colour values outside of specific numeric ranges, it was converted to black (R=0, G=0, B=0). For this analysis, pixels that had colour ranges of red=70-100%, blue=40-100%, and green=40-100% were retained. These colour ranges approximate the range of “orange” colours of the needles at these sites.

After the images were colour processed to isolate the orange of the conifer needles from other elements of the forest floor (i.e. twigs, leaves, moss), the images were converted to greyscale images. Examples of the converted images are in Figure 3.5. The greyscale, colour-processed image was then examined using a variation of the rotational analysis. The digital image of each quadrat was rotated through 360 degrees in ten degree increments and the intersections between needles and a set of parallel test lines were enumerated. All image processing used uncompressed TIFF files to ensure that any colour manipulation can be duplicated.

All analyses were conducted in the Python programming language using the *Linear* software library developed during the thesis research.


(a) 0 degrees


(b) 30 degrees


(c) 50 degrees

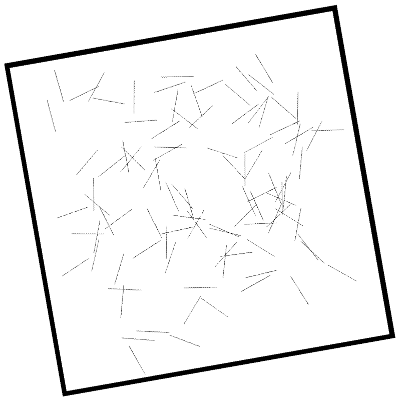
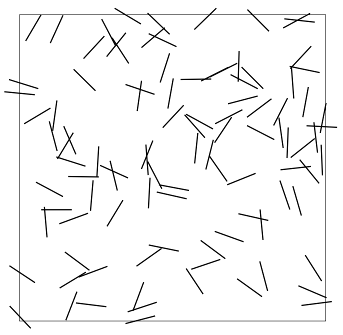
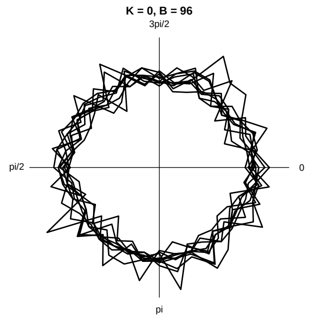

(d) 80 degrees

Figure 3.2: A segment process rotated at 0, 30, 50 and 80 degrees. Plot size is 10 x 10 units, *n segments* = 100, segment length = 1.0, mean angle = 0, von Mises $K = 0$.

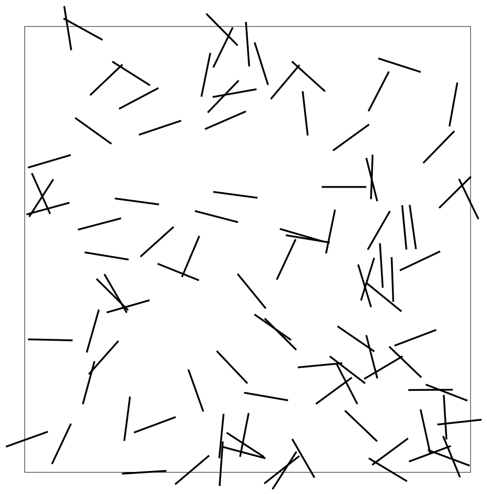


(a)

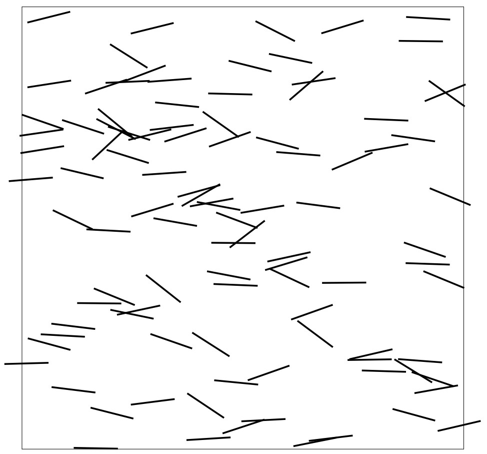


(b)

Figure 3.3: (a) A segment process (n = 100) having segment orientations generated from the von Mises distribution. The plot size is 10 x 10 units. The system is isotropic ($K = 0$). The von Mises distribution produces an isotropic process at $K = 0$ and a completely anisotropic (parallel system) around a preferred orientation at K. (b) The radial plot of intersection counts for ten raster systems ((a) is an example of one such system) that have been rotated in ten degree intervals and sampled with a set of parallel test lines at each rotation. The circular shape of the ten superimposed radial plots indicate isotropic behaviour of the segment process. The boundary length (B) was estimated using a lattice of horizontal and vertical test lines.



(a) Segment Process #1, $K = 0$, $n = 100$



(b) Segment Process #2, $K = 10$, $n = 100$

Figure 3.4: (a) An isotropic segment process (n = 100 segments, von Mises $K = 0.0$, plot size = 10 x 10 units). (b) An anisotropic segment process (n = 100 segments, von Mises $K = 10.0$, plot size = 10 x 10 units).

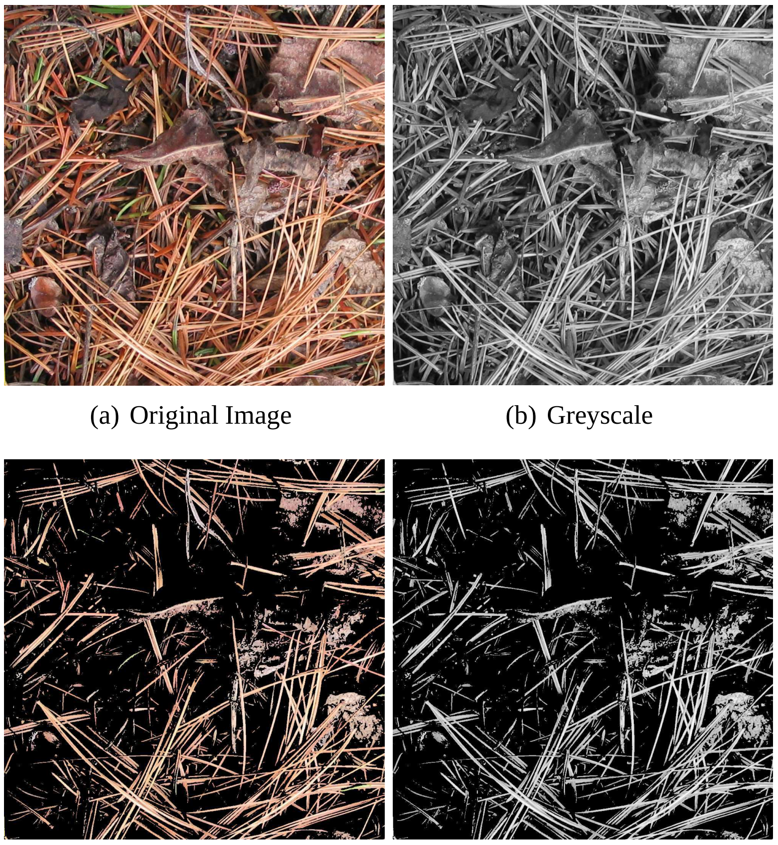
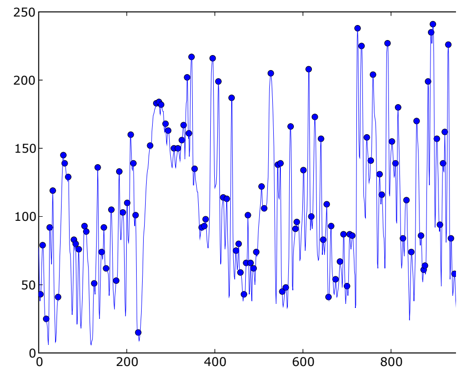
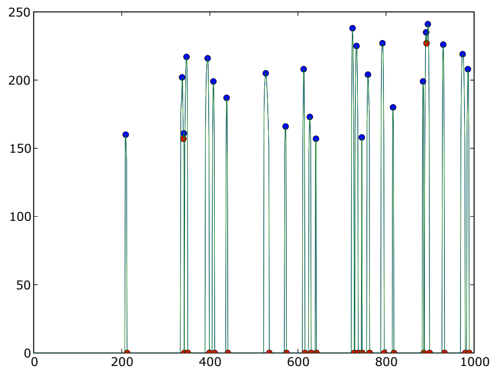

(a) Original Image
(b) Greyscale
(c) Colour Processed
(d) Greyscale After Colour Processed

Figure 3.5: Stages of colour processing in order to extract linear data. (a) Original 10 x 10 cm quadrat (1000 x 1000 pixels). (b) The image in (a) converted to greyscale. (c) The image in (a) with pixels having RGB values R=70-100%, B=40-100%, and G=40-100% retained and pixels outside this range converted to RGB=(0,0,0). (d) The image in (c) converted to greyscale.



(a) Greyscale maxima before colour processing



(b) Greyscale maxima after colour processing

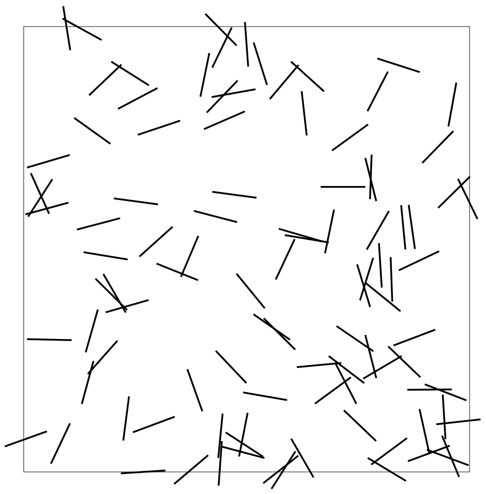
Figure 3.6: (a)The pixel values along a test line across a greyscale image 1000 pixels wide (similar to Figure 3.5(b)). Dots indicate local maxima. The local maxima of the greyscale intensity (high pixel values indicate conifer needles). (b) The pixel values along a test line across a greyscale image that was first colour-processed (similar to Figure 3.5(d)). Dots indicate local maxima.

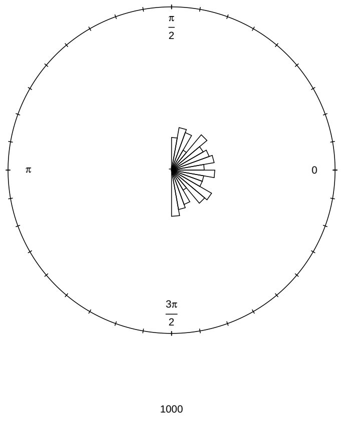
3.3 Results

3.3.1 Angular Distribution

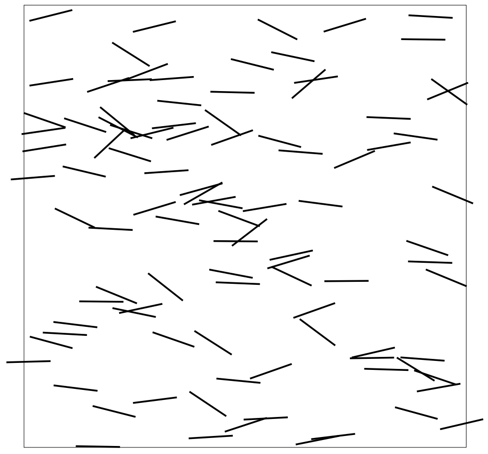
3.3.1.1 Artificial data: Segment processes

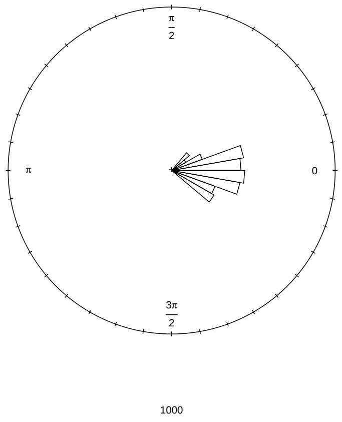
Figure 3.7 shows two segment processes and their respective rose of directions that show their length-weighted distribution of segment orientations. Segment Process #1 (Figure 3.7(a)), which has a von Mises $K = 0.0$, is isotropic. This is clearly demonstrated by the rose of directions which shows that all segment orientations are distributed evenly between $\pi/2$ and $2\pi/3$ (Figure 3.7(b)). Segment Process #2 (Figure 3.7(c)), which has a von Mises $K = 10.0$, is strongly anisotropic with a preferred angle of zero radians (horizontal on this plot). This anisotropic tendency is detected using the rose of directions (Figure 3.7(d)).


(a) Segment Process #1, von Mises $K = 0$, $n = 100$



(b) Rose of directions: segment orientations of Segment Process #1


(c) Segment Process #2, von Mises $K = 10$, $n = 100$



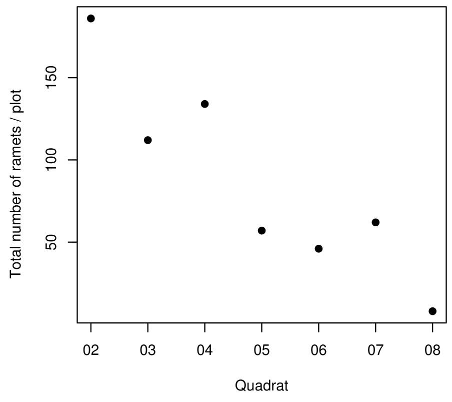
(d) Rose of directions: segment orientations of Segment Process #2

Figure 3.7: Two segment processes (a & c) and their respective roses of directions (b & d) for the length-weighted distribution of segment orientations.

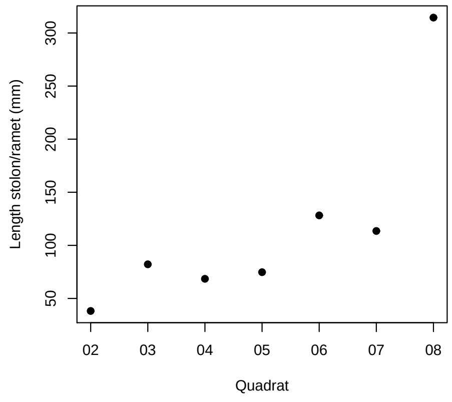
3.3.1.2 Field data: *F. virginiana* stolons

Figure 3.8 shows aspects of the field data from the Woodbend field plots. Figure 3.8(a) shows the number of ramets per plot. There is a general decrease in the number of *F. virginiana* ramets per plot along the transect. Although the number of ramets generally decreases along the transect from forest stand to abandoned field, the length of stolon per ramet increases (Figure 3.8(b)). Figure 3.8(c) shows the total length of stolon per plot vs. the number of ramets per plot. It seems to suggest that there is a unimodal response in total stolon length. At very high ramet density, the total length of stolon decreases. Figure 3.8(d) shows the average length of stolon per ramet vs. the number of ramets per plot. Generally, as the ramet density increases in the transect plots, the length of stolon per ramet decreases.

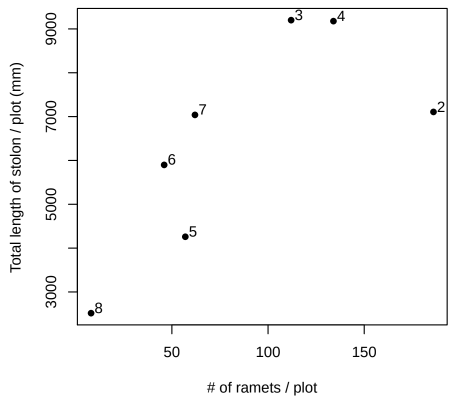
Figure 3.9 shows the rose diagrams for the stolons of ramets in the seven plots. Stolon in plots farthest from the field edge (Plot 02, 03) do not have a preferred direction of spread. Plots progressively closer to the abandoned field, however, tend to show an increasing tendency toward a preferred orientation of stolons parallel to the transect and perpendicular to the forest edge (zero radians on the radial plot). This preferred orientation is particularly evident in Plots 07 and 08.



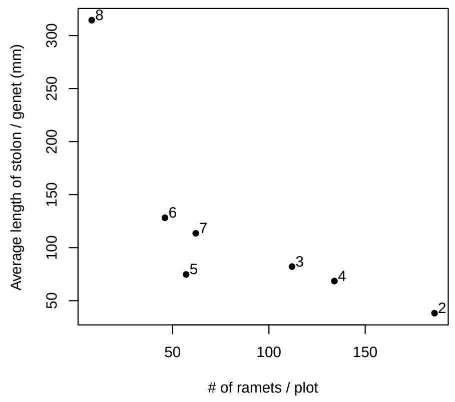
(a) # Ramets per plot



(b) Average length per ramet

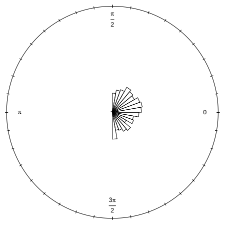


(c) Lenght vs # ramets

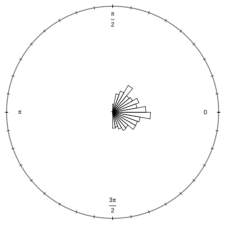


(d) Average length vs # ramets

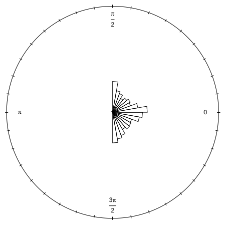
Figure 3.8: Data showing number of ramets and length of stolon for a transect of 2 m x 2 m plots traversing a forest-field edge at the Woodbend Research Forest near Edmonton, AB. Plot 02 is located in the forest, Plot 08 is located in the field, and the forest-field edge is located a approximately Plot 07. (a) The number of ramets per plot, (b) the average length of stolon per ramet, (c) the stolon length vs. the # of ramets and (d) the average stolon length per genet.



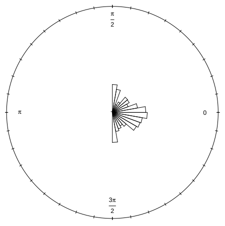
(a) Plot 02



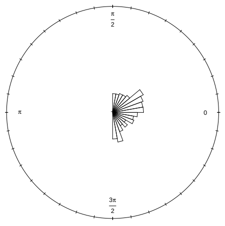
(b) Plot 03



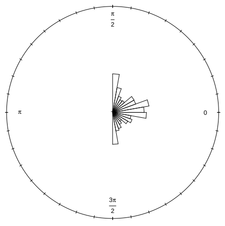
(c) Plot 04



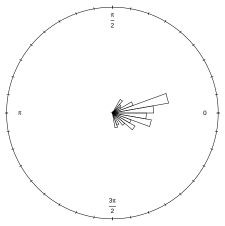
(d) Plot 05



(e) Plot 06



(f) Plot 07



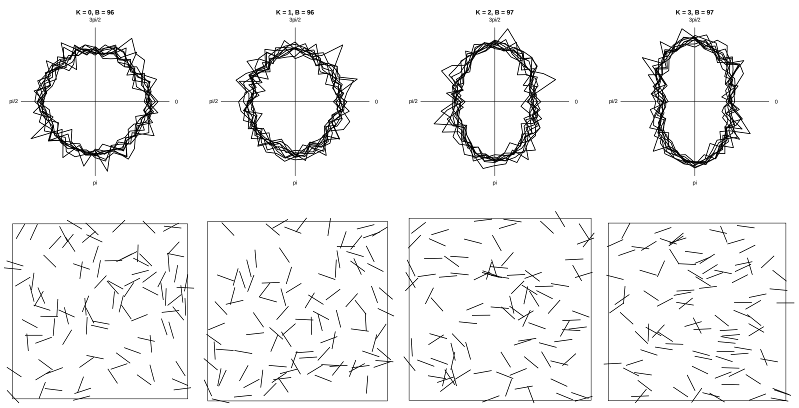
(g) Plot 08

Figure 3.9: Rose diagrams showing the distribution of “angle units” of stolon fibres. A set of orientations was produced where each unit (mm) of line segments comprising stolon fibres was represented by its angle.

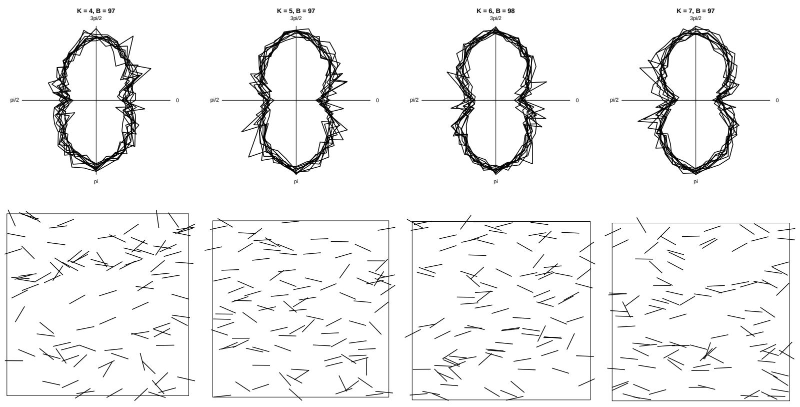
3.3.2 Rotation Analysis

3.3.2.1 Artificial data: Segment processes

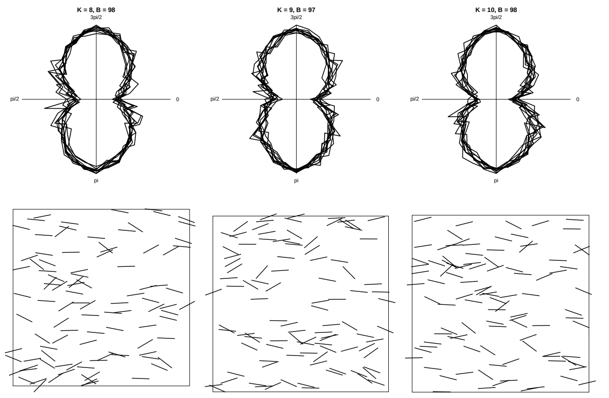
Figure 3.10 shows an analysis where segment processes where generated using eleven values of von Mises K (0-10). Ten segment processes were generated for each value of K and one example is plotted beside the respective radial plot. Radial plots show the number of intersections between parallel test lines and segments. As the value of K is increased, the segment processes begin to exhibit a preferred orientation toward zero (horizontal in segment plots and radial plots). On the radial plots, the preferred orientation is perpendicular to the line drawn through the longest portion of the plot. At $K = 10$ the segment process is almost parallel, and the resulting radial plots from the ten processes clearly indicate the direction (zero radians) where the number of intersections between test lines and segments is a minimum. Note that B, the estimate of total length, is slightly less than 100 units due to portions of randomly distributed segments falling outside the plot boundary. Also note that the estimate of total length is accurate regardless of the level of anisotropy.



(a) K=0.0 B=96.0 (b) K=1.0 B=96.0 (c) K=2.0 B=97.0 (d) K=3.0 B=97.0



(e) K=4.0 B=97.0 (f) K=5.0 B=97.0 (g) K=6.0 B=96.0 (h) K=7.0 B=97.0

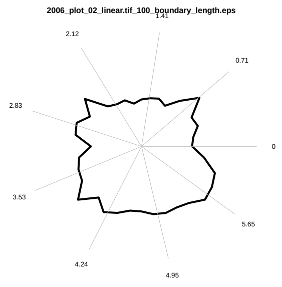


(i) K=8.0 B=96.0 (j) K=9.0 B=97.0 (k) K=10.0 B=96.0

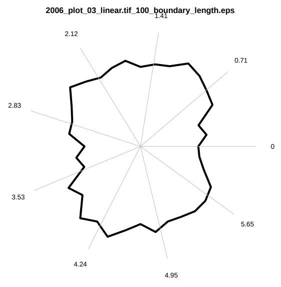
Figure 3.10: Rotational analysis for 11 segment models (each replicated 10 times) where a raster representation of a segment process is sampled using horizontal test lines for each of 36 ten-degree rotational increments. The segment models ranged in their degree of anisotropy from $K = 0.0$ (isotropic) to $K = 10.0$ (strongly anisotropic). The preferred angle of the models was zero. The average length of the segment processes was estimated using a lattice of test lines.

3.3.2.2 Field data: *F. virginiana* stolons

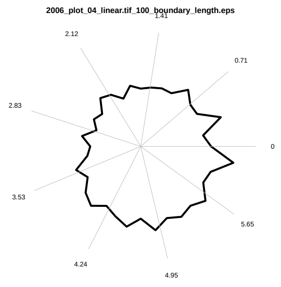
Figure 3.11 shows the results of a rotational analysis based on estimated linear length, *B*, for *F. virginiana* stolons in each plot of the Woodbend field data. The length estimates for each rotation of the data are plotted against the rotation angle on circular plots. The plots are plotted at the same scale, so the diameter of the radial plot can be compared between plots. Generally plots farther into the forest (e.g., Plot 2) have less stolon per ramet compared to plots near the field (e.g., Plot 8). No plots have a circular radial plot. Although centred on the origin, Plots 3, 5, 6, 7, and 8 are not circular, suggesting that the stolons are not isotropically distributed. Plots 2, 4 and 8 are not symmetrical across the plot origin. That is, they are “off centre”, which indicates that the stolons are heterogeneously distributed across the plot. Plots 3, 5, 6, and 7 have clearly reduced estimates of *B* at rotations of zero radians, which indicates preferred stolon growth parallel to the transect and perpendicular to the forest-field edge. Plot 8 also exhibits slightly lower estimates of *B* at a rotation of zero and π, indicating preferential stolon growth perpendicular to the forest edge.



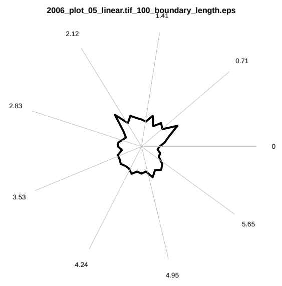
(a) Plot 02



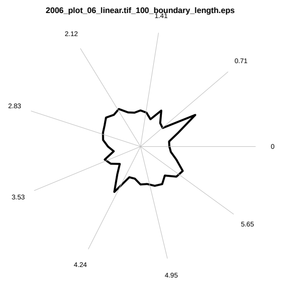
(b) Plot 03



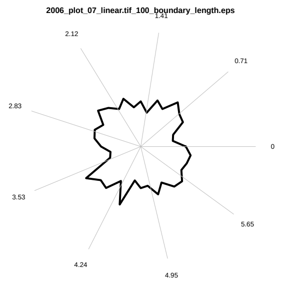
(c) Plot 04



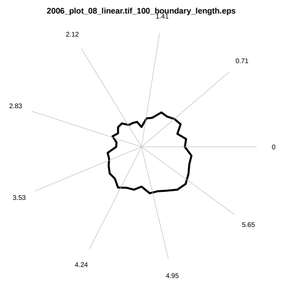
(d) Plot 05



(e) Plot 06



(f) Plot 07



(g) Plot 08

Figure 3.11: Radial plots of estimated boundary length of *F. virginiana* stolons in 2 m x 2 m plots along a transect that traversed a forest-field edge at the Woodbend Research Forest. The intensity was estimated using horizontal, straight test lines plotted against the rotation angle of the original data. At a rotation of zero and π, the test lines align with North and South in the field, are parallel with the transect, and perpendicular with the forest-field edge.

3.3.2.3 Field data: conifer needles

Figure 3.12 shows the results of the rotational analysis of the conifer needle data. Each radial plot shows the mean number of intersections between parallel vertical test lines and needles in contiguous quadrats along transects that were orientated down hill slopes.

At the very steepest slopes (43 and 52.7 degrees), a deformation of the radial plot of intersection counts is evident. There are considerably fewer intersections between needles and test lines at rotation angles near zero and π (horizontal on the radial plot). At these rotation angles, the parallel test lines are vertical in the image and are aligned parallel to the slope of the hill. This elongate shape of the rose of intersections, also shown in Figure 3.10(k), is a clear indication of preferential alignment of linear structure. In the steepest of slopes (43 and 52.7 degrees) the preferential orientation is aligned parallel of the slope.

The rose of intersections, aside from indicating preferential orientation, also provides an indication of the density of needles in the quadrats. All roses of intersections in Figure 3.12 are plotted at the same scale, and their relative size indicates the relative density of needles along the transect. Transects that have fewer needles (illustrated in the example images in Figure 3.12) also have smaller radial plots with smaller diameter.

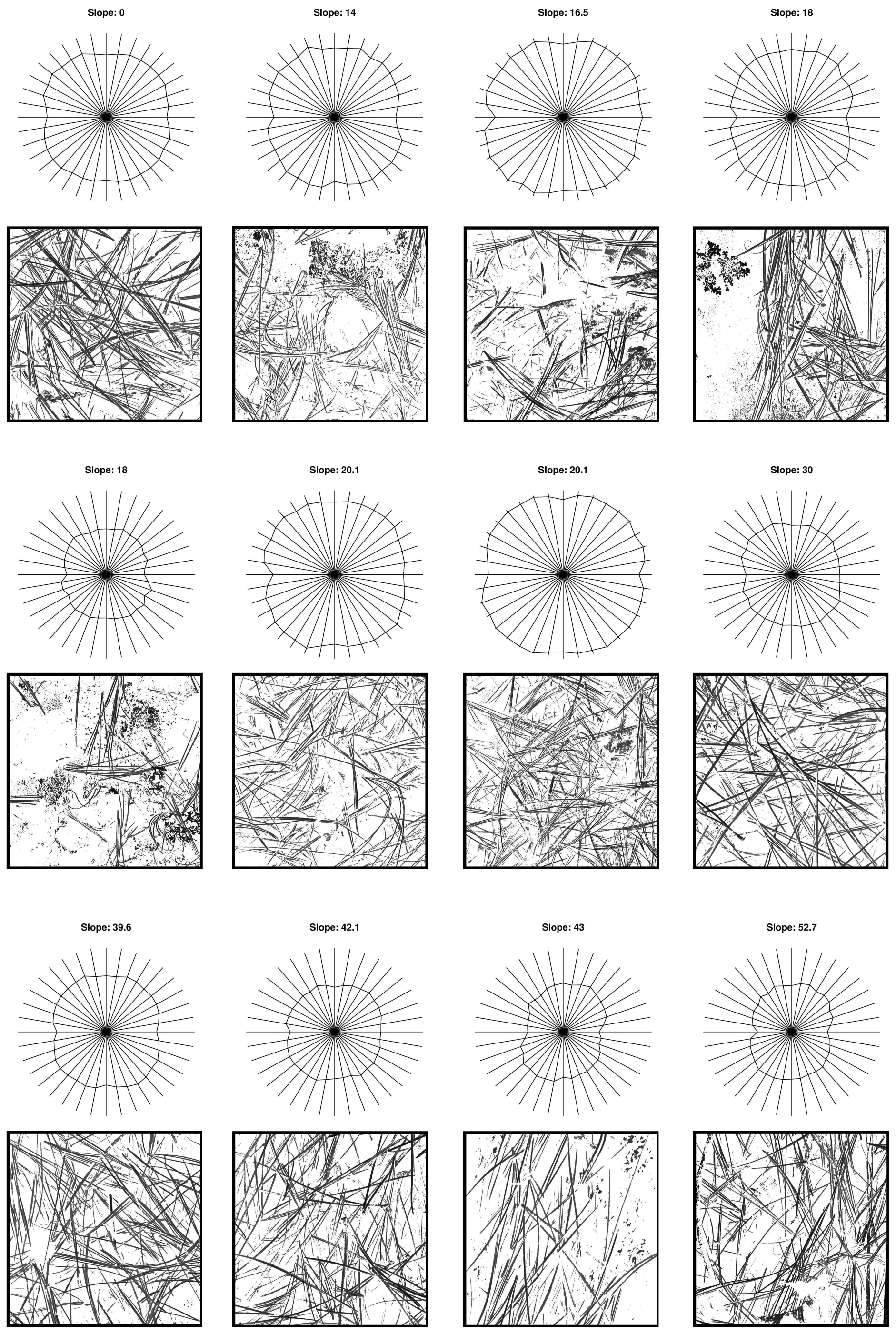


Figure 3.12: The examination of conifer needles from 12 transects 1 m long, each with 10 quadrats (10 cm x 10 cm). Radial plots showing the mean number of intersections between parallel test lines and processed images of conifer needles for the quadrats in the transect (*n* = 10 quadrats).

3.4 Discussion

The analysis of artificial data (segment processes) demonstrates that both the summary of the angular distribution using radial plots and the rotational analysis can provide effective summaries of the directional properties of a fibre system. Plotting the angular distribution is best applied when the data are available in the vector format because the orientation of fibres represented by polylines can be easily measured once the data are loaded into the appropriate software package (e.g., the *Linear* software library). There are methods available to determine orientation from raster data such as “tracing” the fibre at an intersection point with a test line (the *tracing method*), as well as calculating orientation based on the difference between observed and expected fibre width as measured by the test line (the *line width method*). The tracing and line width method both require assumptions that could lead to errors if they are not adhered to, but they are viable methods for examining raster data. Although they were not presented here, the software library *Linear* provides functions that will facilitate the application of both methods. The rotational analysis presented in this chapter provides a means of examining the angular properties of data in raster format that requires fewer assumptions than the line width method and the tracing method. The rotational analysis provides a means to estimate preferential direction of raster data with a relatively simple image transformation and straight test lines.

The segment processes were generated completely in a vector data format and then converted to a raster format prior to analysis in order to demonstrate the practical need for a bridge between raster and vector analysis when working with linear data. Null models and angular distributions are easier to implement and more efficient using vector methods. In the case of completely vectorized data, it is computationally simpler to calculate exact linear properties of the system (when dealing with simple, first order measures, such as length and orientation). However, the conversion of linear data into a vector data format (e.g., through the digitization of stolon mappings using a GIS or vector graphics software) can be very time consuming. In the case of complex linear data (e.g., heavily crossed conifer needles), vectorization of the data might not be practical. In such cases, the use of raster data methods would be preferred.

The rotational analysis was used to test the hypothesis that conifer needles on hillslopes will show a preferred direction downslope, potentially as a consequence of physical processes. The examination of conifer needle data showed that, in the steepest of slopes (43 and 52.7 degrees) needles are preferentially aligned parallel of the slope. The preferential arrangement on steep slopes suggests the presence of a physical process acting on the hill slope, possibly the result of a physical interaction between the needles and water that moves down the slope during storm events. The analysis of this data also demonstrated that, given some knowledge of the system (e.g., colour ranges for fallen conifer needles), it is possible to estimate the characteristics of linear data from complex datasets using methods that are efficient in terms of field and laboratory effort.

The angular distribution and rotational analyses were used to test the hypothesis that the directional properties of *F. virginiana* stolon will vary across a forest-field edge. The analysis of the *F. virginiana* data seem to indicate that stolons have a preferred orientation perpendicular to the forest edge in plots near the old field. This suggests that the properties of the edge may be influencing the growth direction of stolons, perhaps by a foraging response to light at a large scale (on the scale of the transect).

Forest-field edges have been shown to be vegetation gradients (Meiners and Pickett, 1999), and the Woodbend site exhibited gradient properties in terms of species presence/absence and tree CBH.

The lack of preferred direction of *F. virginiana* stolon in many of the plots farther into the forest may reflect a foraging response to light at a small scale (at the scale of the plants in the herb layer). The distribution of light in the heavily forested plots is heterogeneous, and foraging for these smaller scale light patches could explain the lack of anisotropic behaviour in these plots.

The literature suggests that *Fragaria* spp. can respond to heterogeneity. Angevine (Angevine, 1981, 1983) studied the demography of natural populations of *F. vesca* and *F. virginiana* at three sites. He found that the local conditions of the site had a greater influence on local demography than the identity of the species and that there was a great deal of variation between the sites. *F. chiloensis* has been shown to increase vegetative reproduction in areas of low light but high nitrogen (Alpert, 1999b). Alpert (1999b) also showed that integration can have different effects on plasticity at different levels of plant organization. *F. virginiana* has been found to increase biomass allotment to vegetative reproductive organs when plant density is low (Holler and Abrahamson, 1977). It has also been found that *F. virginiana* produced higher proportions of stolon biomass in low density field and wooded sites than higher density sites (Holler and Abrahamson, 1977). This agrees with the field data at the Woodbend site, where average stolon length per genet increased as the density of ramets decreased.

Clonal plant foraging for light in other species has been documented in the literature. Shibaike et al. (1996) found distinct populations of *Oxalis* *corniculata* exhibited a range of foraging responses along a light gradient and a range of plastic responses along a nutrient gradient. Sammul et al. (2004), examined vegetation from forests, wooded meadows, and meadows and concluded that foraging abilities might be favoured in forested environments where light is heterogeneously dispersed and a phalanx strategy is favoured in undisturbed (unmown) meadows. The clonal herb *Glechoma hederacea* has been found to produce longer, unbranched stolon internodes in low-light conditions and short, frequently-branched stolon internodes in high-light conditions (Slade and Hutchings, 1987b). Such a response would allow for utilization of resources in high-resource zones and more rapid spread through zones of low-resource availability. *Lamiastrum galeobdolon* (L.) exhibited a foraging response to light, producing shorter internodes and higher branching intensity under high light conditions (Dong, 1993). *Asarum canadense* genets have been found to have shorter rhizomes in high light forest than genets in low light forest (Cain and Damman, 1997). Some clonal species have exhibited an ability to respond to light heterogeneity through changes in their level of integration. For example, *Solidago canadensis* has been shown to respond to changes in environment that resulted in increased resource (light) heterogeneity by increasing clonal integration (Hartnett and Bazzaz, 1983). This is a somewhat different response than changing structural attributes, but it is an important illustration of the flexibility afforded by the clonal growth form. Similarly, shading can have an impact on spatial structure as well. Huber and Stuefer (1997) found that, although shading did induce changes in branching pattern, the effect was the result of slowed development of ramets that were shaded.

3.5 Conclusions

The characterization of the directional structure of linear ecological data can provide important insights into potential underlying processes when combined with additional data about the system under study. Plotting the orientation distribution directly onto radial plots is the best method if the data are readily available in vector format. If the data are not available in such a format (i.e., it is stored as raster data), a distribution of orientations can be extracted from raster data using test lines using tracing and/or line-width methods, but these require specific assumptions about the data. The rotational analysis has been shown to be an effective and efficient means of determining preferred orientation, and it is recommended for these reasons.

In the case of the *F. virginiana* field data, both the rotational analysis and the plots of directional distribution seem to suggest different ramet behaviour in terms of stolon direction along the forest-field transition zone. The spatial arrangement of stolons farther from the forest edge may be driven by a foraging response to light heterogeneity at small scales. Closer to the forest-field edge, however, *F. virginiana* may be responding to the source of light from the open field.

In the case of the conifer needles on hill slopes, the rotational analysis indicates an effect of slope on needle orientation. It seems likely that physical processes, such as the flow of water during rain events, is responsible for needle alignment.

Chapter 4

Incorporating genet structure into spatial analyses

4.1 Introduction

Characterizing the spatial structure of segments and unrelated fibres in space can involve a wide range of tools. Chapters 1 through 3 have introduced and applied some of these methods. In ecological systems, however, the nature of the system may require additional analyses that take into account specific properties of the system. This is particularly relevant in the case of clonal plants where a genet might be composed of a number of stolons that are connected to more than one ramet. In this case, the connections of the "lines" (stolons) have meaning; their connections result in the potential physiological integration of the genet's ramets. Any analyses that do not take such a higher level structure into account might fail to incorporate the properties of the entire genet. The analyses proposed in Chapters 1 through 3, for example, examine each intact fibre, analogous to a single stolon, separately. They are useful in examining the attributes of a system of fibres, but they cannot provide insight into the explicit connections between fibres. For example, the methods explored in previous chapters cannot determine whether a clumped arrangement of stolons is due to a clumped arrangement of genets.

In order to consider the structure of several interconnected ramets together, the data must include not only the location of stolons (lines) but also their connections and the locations of ramets (points). It might also be important to distinguish between different categories of points (e.g., parent and daughter ramets). With such data in hand, it is possible to begin to address questions, such as whether the clustering of genets leads to clustering of stolons. The examination of the second order spatial properties has been mentioned in previous chapters, but considering these properties at the level of the genet requires further discussion of "randomness".

An important aspect of examining point patterns, which have been used extensively in plant ecology, is the application of complete spatial randomness (CSR) (Dale, 1999). Models of CSR are used to examine point patterns for clumping, randomness, or regularity. In the case of a study plot with *N* plant locations, a logical model of CSR is to randomize each point location. In the case of a clonal plant mapped into 2D space, however, CSR becomes more difficult to define. Should all the ramet locations or stolons/rhizomes be randomized in isolation? The physical structure of the clone would be destroyed in either case. Should the length of stolons, branching angle, or internode length of stolons be randomized? In the case of these variables, randomization is likely not ecologically meaningful, so other models must be investigated. Clearly, the consideration of linear data in the ecological context, particularly in the case of plant ecology, requires careful application of randomization and/or Monte Carlo techniques to investigate the higher level spatial characteristics of these systems.

Despite the challenge of their development, methods of randomization that take into account the properties of entire plants could provide insight into a higher level of spatial structure than can be achieved by considering individual components of a plant separately. For example, *Aralia nudicaulis* has been found to have a clumped distribution of shoots, but the clumps are formed from ramets of many different genets (Edwards, 1984). The clustering of shoots observed by Edwards (1984) could point to a foraging response where ramets are placed in response to resources. This agrees with the foraging concept of de Kroon and Hutchings (1995). *Aralia nudicaulis* has also been found to exhibit regular ramet patterns at the scale of the ramet, demonstrating an inhibition distance near that of the average radius of a ramet (Kenkel, 1993). In this case, the regularity allowed for modelling of the ramet locations. Exploration of such systems using methods incorporating the spatial arrangement of stolons/rhizomes and ramets, while preserving the stolon-ramet connections, could lead to more insight into clonal plant behaviour.

This Chapter introduces randomization and Monte Carlo methods analogous to those methods used in point pattern analysis. It then applies the approaches to artificial data to illustrate their efficacy. It also applies them to the investigation of experimental field data from *F. virginiana* ramets transplanted into clumped, random and regular arrangements. The hypothesis is that stolons will elongate in shade produced from neighbour genets and shorten and produce more ramets in patches of light, and as a consequence the resulting stolon structure will not follow that of the original planting arrangement.

4.2 Methods and Data

4.2.1 Constrained Randomization Analysis

A constrained randomization approach allows for some aspects of a set of spatial data to be randomized while keeping others stationary. Given a mapping of several distinct clonal plants, each genet, G_i, (ramets and stolons) is randomized in the x-y plane while maintaining the structure of the genet itself. The location of the entire genet is randomized without changing the internal structure of the genet itself.

The constrained randomization analysis has the following steps:

1. Determine stolon intensity around stolons at a given radius (B_{Or}) for each genet, G_i, in the study plot, D.

   i. Distribute N circles randomly on the stolons

   ii. Calculate intensity of stolons within a radius r using, Equation 2.2. Ignore all stolons belonging to the same genet. Note that this requires the fibres (stolons) be ‘marked’ with a genet-specific label.

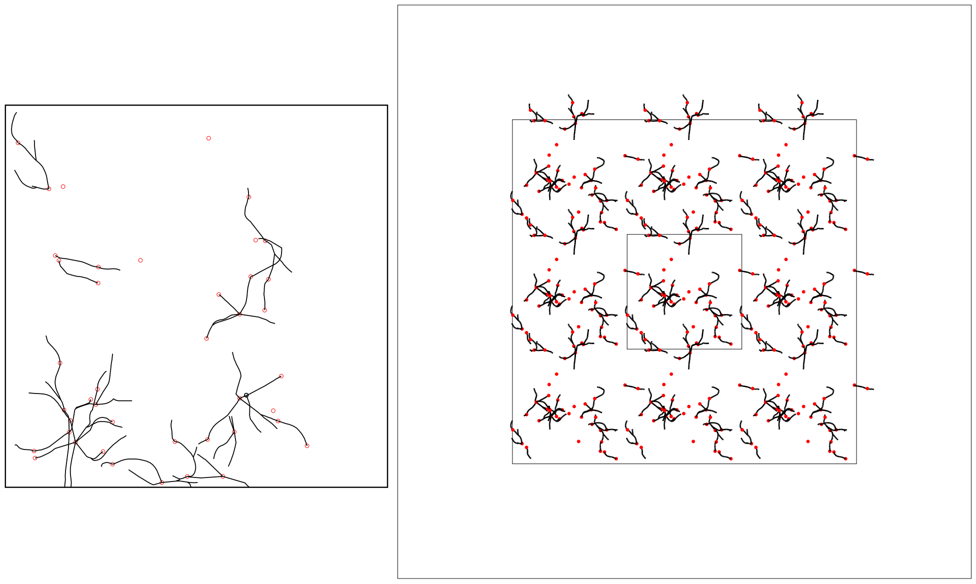
2. Randomize in the $x-y$ plane all genets without changing their individual structure. The parent ramet’s location is randomized, and the rest of the genet is subjected to the same translation. A toroidal edge correction is used to adjust for edge effects caused from randomization. One way to accomplish this is by creating a new plot, E, that is 3 x 3 times the size of the original study plot. The study plot data in D is randomized and copied to each of the nine plots making up E. Calculate B_{Er} using the same method as in Step 1 for $E_{(2,2)}$, the sub-plot of E. Figure 4.1 shows a field plot of ramets and stolons (a) and the new randomized data (b).

3. Repeat Step 2 NSIM times to produce a distribution of B_{Er} estimates for the study plot. This is an estimate of the intensity of stolons from all other genets around genet a at a radius r when the genets are distributed completely at random across the plot.

4. Repeat 1, 2, & 3 for a range of circle radii $r = r_1, r_2, r_3....r_n$

5. Compare B_{Or} alongside the distribution of B_{Er} for each radii r.

Figure 4.1 shows a 2m x 2m plot of mapped *F. virginiana* stolons and ramets (a) and the randomized data with toroidal edge correction (b). In (b), only the data in the innermost region is used for the analysis. The innermost region has dimension 2m x 2m, the second innermost region has dimensions 6m x 6m and the entire randomized plot area has dimensions 10m x 10m.



(a) Randomly Distributed Genets (b) Circle Sampling Analysis

Figure 4.1: A 2 m x 2 m plot of mapped *F. virginiana* stolons and ramets (a) and the randomized data with a toroidal correction (b).

All analyses were conducted in the Python programming language using the *Linear* software library developed as a component of the thesis research (see Appendix B).

4.2.2 Artificial Data

Three sets of artificial spatial data were used to illustrate the genet randomization procedure. The datasets represent genets that are random (Figure 4.2(a)), clumped (Figure 4.3(a)) and regular (Figure 4.4(a)). These datasets were constructed manually, by replicating the mapping of a *F. virginiana* genet and positioning the copied data into the desired arrangement.

4.2.3 Field Data

In the spring to 2005, approximately 1400 *F. virginiana* ramets were transplanted from the Cooking Lake-Blackfoot Provincial Recreation Area into 2 m x 2 m plots at the University of Alberta Ellerslie Research Station. Each plot corner was marked with a permanent stake, and plots were separated by a 1 m buffer. A planting arrangement experiment was conducted in 24 plots using 384 of the *F. virginiana* ramets. Sixteen ramets were planted in either random, clumped and regular arrangements in each plot (n plots = 8, 8, and 8, respectively). The regular planting arrangement was a 4 x 4 ramet array, with ramets approximately 400 mm between array rows and columns. The clustered arrangement was formed by locating four point locations in the center of each 1 m x 1 m quadrant composing the plot. At each quandrant center, four ramets were planted to form a square with a length and width of approximately 250 mm. The same pattern was used for all replicates. Two plots (one random, one clumped) were lost during the experiment. The remaining 22 plots were surveyed in August.

A custom-built, four-legged camera pod was constructed that would allow a digital camera to capture images while vertically suspended above the ground. A series of test images taken using the pod were used to calculate the calibration factor for distance measurements, and it was determined that images collected using this pod had a scale factor of 2.4 pixels/mm or 0.42 mm/pixel.

Each plot was digitally imaged during the 2005 season using a camera mounted on the sampling pod. Approximately 40-50 overlapping digital images were taken of each plot in a grid pattern and then compiled using graphics software into a composite image representing the entire 2m x 2m plot. The composite image was cropped to the boundaries of the field plot (using permanent corner markers as a reference) and scaled. The final plot image had dimensions 4730 pixels x 4730 pixels.

Each composite image was converted to a vector dataset representing the plot with points indicating ramets and polylines representing stolons. Using the composite plot image as a background, vector graphics software was used to mark stolons with polylines and ramet locations with points. Parent and daughter ramet locations (identified in field surveys and hand sketches) were identified as parents and daughters, respectively, with marked points. Stolons were marked with polylines that always started at the parent ramet and extended outward toward daughter ramets.

In addition to digital imaging, a hand sketch was drawn of each plot showing ramets and their stolon connections. For each ramet a number of variables were measured (Table 4.1).

Table 4.1: Data collected from *F. virginiana* plants in 22 (2 m x 2 m) plots at the University of Alberta Ellerslie Research Station.

Variable	**Units**
$x - y$ location	mm
leaves	#
longest leaf length	mm
stolon	#
stolon	position mapping
flowers	#
buds	#

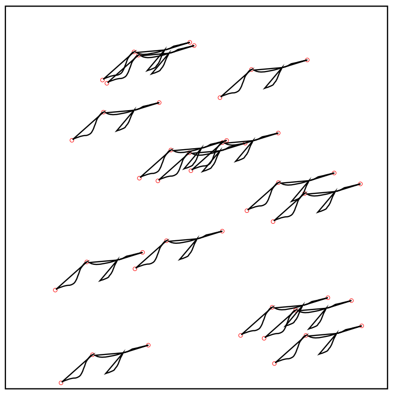
4.3 Results

4.3.1 Artificial Data

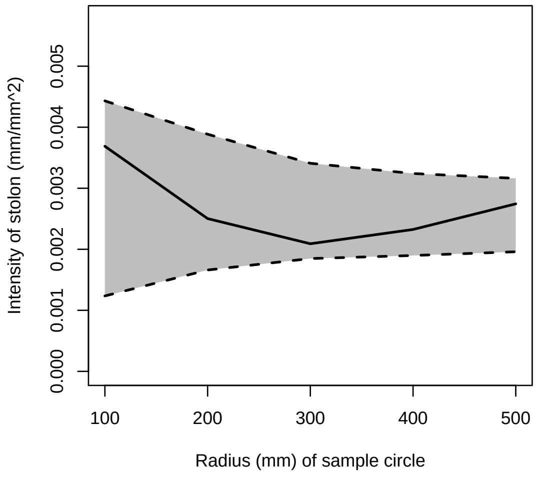
Figure 4.2(a) shows artificial data in a 2m x 2m plot where genets have a random arrangement, and Figure 4.2(b) shows the results from the constrained randomization analysis on the data. The observed intensity of the genet stolons is within 80% of the observations at all scales (radii of sampling circles). This indicates that the intensity of stolons from other genets around any particular genet stolon in the observed data does not significantly differ from what would be expected if all the genets were positioned randomly.

Figure 4.3(a) shows artificial data in a 2m x 2m plot where genets have a clumped arrangement, and Figure 4.3(b) shows the results from the constrained randomization analysis of the data. The observed intensity of the genet stolons is above 80% of the observations at all scales (radii of sampling circles) below 450 mm. This indicates that, at scales below 450 mm, the intensity of stolons from other genets around any particular genet stolon in the observed data is significantly higher than what would be expected if all the genets were positioned randomly.

Figure 4.4(a) shows artificial data in a 2m x 2m plot where genets have regular locations, and Figure 4.4(b) shows the results from the constrained randomization analysis on the data. The observed intensity of the genet stolons is below 80% of the observations at all scales (radii of sampling circles) below 400 mm. This indicates that, as scales below 400 mm, the intensity of stolons from other genets around any particular genet stolon in the observed data is significantly lower than what would be expected if all the genets were positioned randomly.

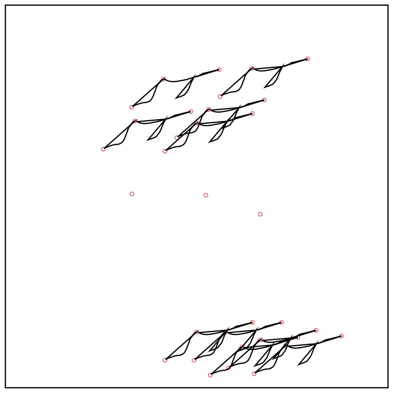


(a) Randomly Distributed Genets

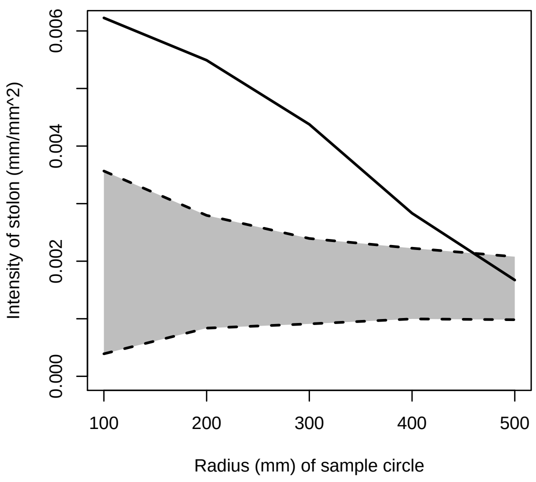


(b) Constrained Randomization Analysis

Figure 4.2: (a) Artificial data simulating randomly distributed genets in a 2 m x 2 m plot. (b) Constrained randomization analysis showing genet-genet stolon intensity (solid line) with an 80% envelope (grey) from 100 randomizations of the genet data (*n* = 100 sampling circles).

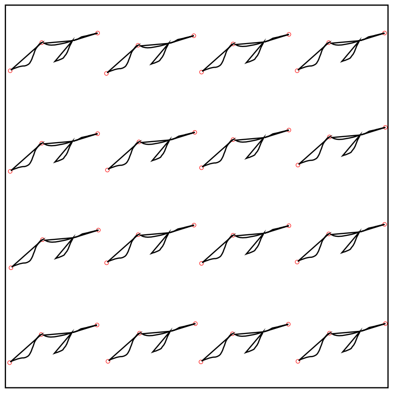


(a) Clumped Genets

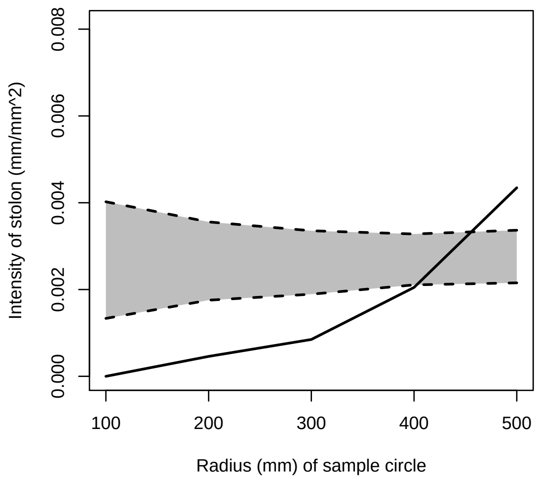


(b) Constrained Randomization Analysis

Figure 4.3: (a) Artificial data simulating clumped genets in a 2 m x 2 m plot. (b) Constrained randomization analysis showing genet-genet stolon intensity (solid line) with an 80% envelope (grey) from 100 randomizations of the genet data (*n* = 100 sampling circles).



(a) Regularly Distributed Genets



(b) Constrained Randomization Analysis

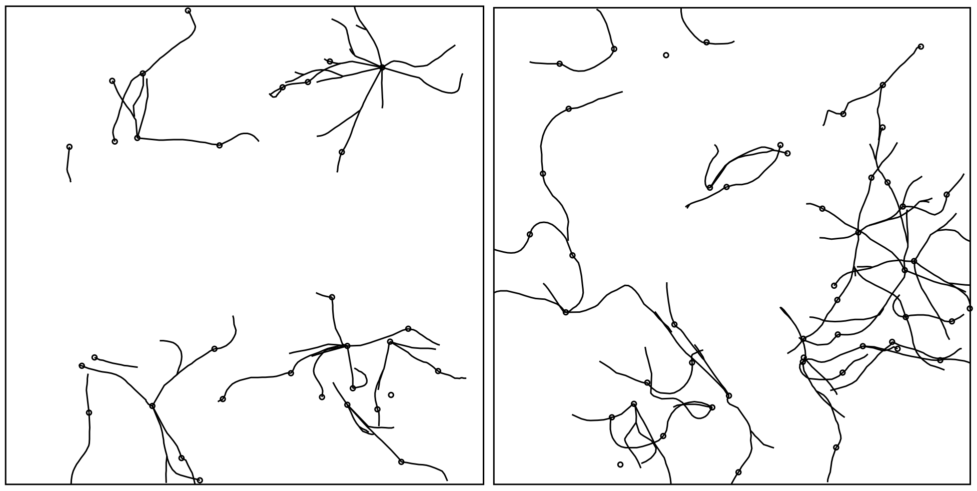
Figure 4.4: (a) Artificial data simulating regularly distributed genets in a 2 m x 2 m plot. (b) Constrained randomization analysis showing genet-genet stolon intensity (solid line) with an 80% envelope (grey) from 100 randomizations of the genet data (*n* = 100 sampling circles).

4.3.2 Field Data

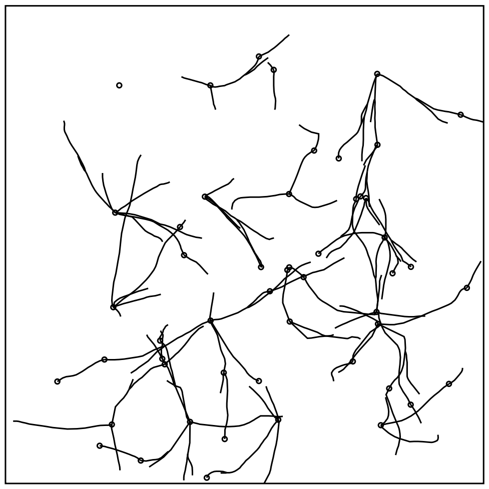
Over the course of the summer, 360 m of stolons were produced by the transplanted ramets, and there were 950 total ramets in the plots.
Figure 4.5 shows one example mapping from each of the clumped, random and regular arrangement treatments.

4.3.2.1 Genet randomization

Figure 4.6 shows the results of the genet randomization for three characteristic plots where the parents had a random arrangement (Subplot 4.6(a)), a clustered arrangement (Subplot 4.6(b)), and a regular arrangement (Subplot 4.6(c)). These plots are illustrative of the general results for the analysis for each initial parent configuration. Full results for plots with parents having a random arrangement, clustered arrangement, and a regular arrangement can be found in Figures A.1 and A.2, A.3 and A.4, and A.5 and A.6 respectively. In the case of the genet randomization for plots where parent ramets had a random distribution (Figure A.1 and A.2), almost all plots had linear spatial structure that was not different than what would be expected if the genets were arranged completely at random. In the case of the genet randomization for plots where parent ramets originally had a clustered distribution (Figure A.3 and A.4), almost all plots had linear spatial structure that was not different than what would be expected if the genets were arranged completely at random. In the case of the genet randomization for plots where parent ramets had a regular distribution (Figure A.5 and A.6), all plots had linear spatial structure that was not different than what would be expected if the genets were arranged completely at random.

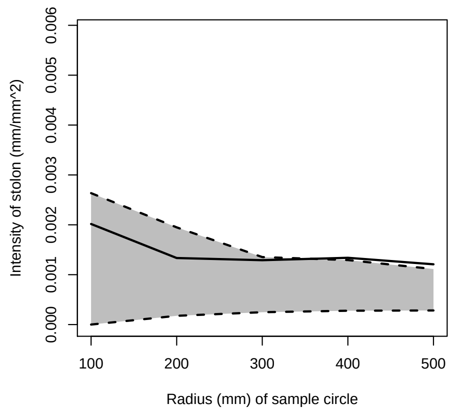


(a) Clumped Arrangement (b) Random Arrangement

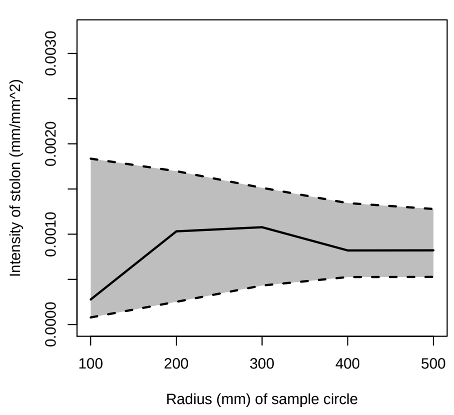


(c) Regular Arrangement

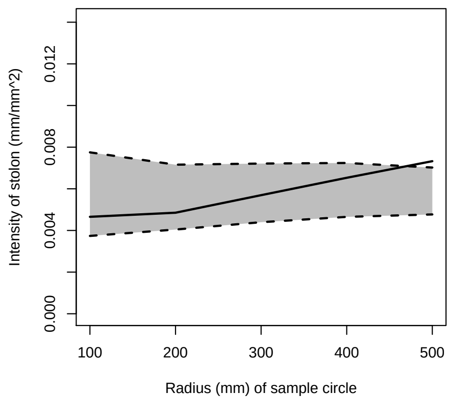
Figure 4.5: Example *F. virginiana* mappings from plots having an initial parent ramet planting arrangement that was either clumped, random and regular. Plot size is 2 m x 2 m, and n = 16 original transplant ramets in each plot. Plots were established at the Ellerslie Research Station, and ramets were grown during the summer of 2005. Ramets were transplanted from the Cooking Lake-Blackfoot Provincial Recreation Area.



(a) Plot 07



(b) Plot 17



(c) Plot 22

Figure 4.6: Constrained randomization analysis of *F. virginiana* plots showing genet-genet stolon intensity (solid line) with an 80% envelope (grey) from 100 randomizations of the genet data (n = 100 sampling circles). Plot size is 2 m x 2 m, and n = 16 original transplant ramets in each plot. Plots were established at the Ellerslie Research Station, and ramets were grown during the summer of 2005.

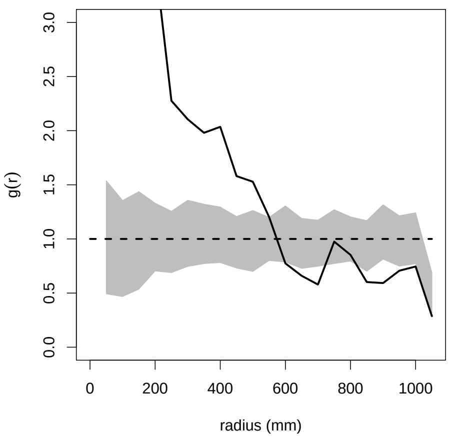
4.3.2.2 Pair correlation, $g(r)$, of fibre points

The pair-correlation function was calculated for random fibre points using the method explained in Chapter 2. Figure 4.7 shows example results for the pair-correlation function for a transplant plot that had a random arrangement of parent ramets (Subplot 4.7(a)), a clustered arrangement of parent ramets (Subplot 4.7(b)), and a regular arrangement of parent ramets (Subplot 4.7(c)). The results for the remainder of the random, clustered and regular plots can be found in Figure A.7 and A.8, A.9 and A.10, and A.11 and A.12 respectively. All plots, regardless of planting arrangement show significant clustering at small (below 200 mm) radii. Although Subplot 4.7(a) and 4.7(b) suggest inhibition at larger radii, almost all plots had fibre point structures that did not significantly differ from a random point process (as demonstrated in Subplot 4.7(c).

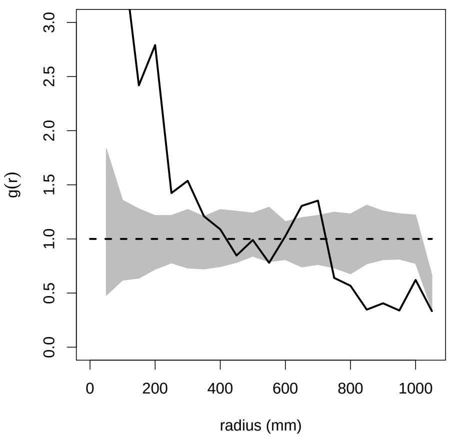
4.3.2.3 Point pattern analyses of ramets

Figure 4.8 shows the pair correlation function calculated for a plot having a random parent arrangement. The remainder of the plots with randomly distributed parents are in Appendix A.1. Figures A.13 and A.14 show the pair correlation function calculated for all ramets in the plots where parent ramets had a random arrangement. Figures A.15 & A.16 show $g(r)$ calculated for all parent ramets for plots where parent ramets had a random arrangement. Figures A.17 & A.18 show $g(r)$ calculated for daughter ramets in random treatment plots. In the majority of plots, both parent and daughter ramets were randomly arranged at all scales.

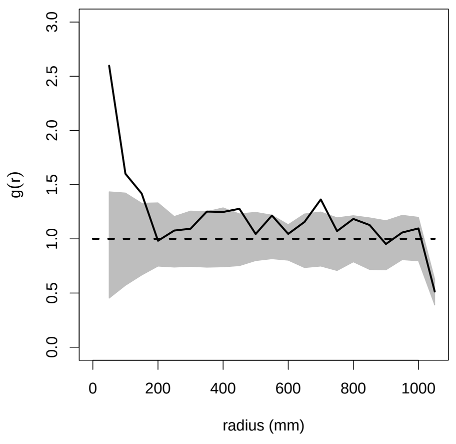
Figures A.19 & A.20 shows the $L(r)$ function calculated for all ramets in the plots where parent ramets had a random arrangement. Figures A.21 & A.22 show $L(r)$ calculated for all parent ramets for plots



(a) Random Parents



(b) Clustered Parents



(c) Regular Parents

Figure 4.7: Pair correlation function calculated on fibre points sampled randomly from stolons in 2 m x 2 m plots when the parent planting arrangement was either random, clustered or regular. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations.

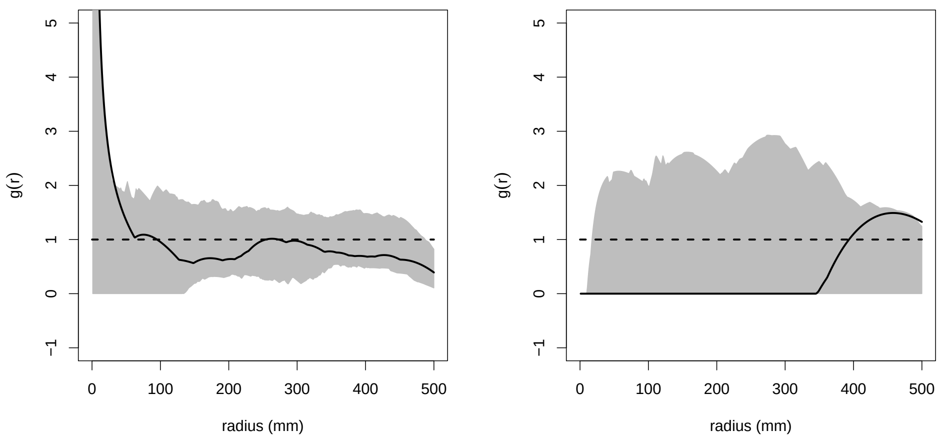
where parent ramets had a random arrangement. Figures A.23 & A.24 show $L(r)$ calculated for all daughter ramets in the random treatment plots.

Figure 4.9 shows the pair correlation function calculated for a plot having a clustered parent arrangement. The remainder of the plots with clustered parents are in Appendix A.1. Figures A.25 & A.26 show $g(r)$ calculated for all ramets in the plots were parent ramets had a clustered arrangement. Figures A.27 & A.28 show $g(r)$ calculated for all parent ramets for plots where parent ramets had a clustered arrangement. The pair-correlation function for parent ramets clearly indicated clustering at radii between approximately 200 to 400 mm. This coincides with the original cluster sizes used for the parent ramets. Figures A.29 & A.30 show $g(r)$ calculated for daughter ramets in random treatment plots. The daughter ramets in almost all plots in the clustered treatment, however, do not show the clear pattern of clustering exhibited by the parent ramets.

Figures A.31 & A.32 show $L(r)$ calculated for all ramets in the plots where parent ramets had a clustered arrangement. Figures A.33 & A.34 show $L(r)$ calculated for all parent ramets for plots where parent ramets had a clustered arrangement. Figures A.35 & A.36 show $L(r)$ calculated for all daughter ramets in the clustered treatment plots.

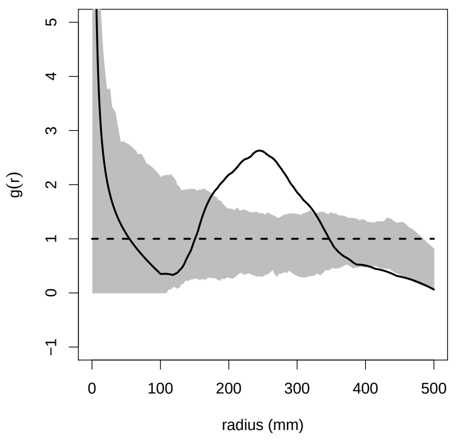
Figure 4.10 shows the pair correlation function calculated for a plot having a regular parent arrangement. Figures A.37 & A.38 shows the pair correlation function calculated for all ramets in the plots were parent ramets had a regular arrangement. Figures A.39 & A.40 show $g(r)$ calculated for all parent ramets for plots where parent ramets had a regular arrangement. Figures A.41 & A.42 show $g(r)$ calculated for daughter ramets in the regular treatment plots. The pair correlation for parent ramets reflects the regular arrangement of parent ramets, showing inhibition at radii below around 200-300 mm. Clustering was detected at large radii (around 400 mm) which is the scale of the array of regularly spaced ramets. The pair correlation function for daughter ramets in the same experimental plots, however, does not detect regularity. Instead, the study plots, including the one in Figure 4.10(b) show a tendency for daughter ramets to be either clustered or random.

Figure A.43 & A.44 shows $L(r)$ calculated for all ramets in the plots were parent ramets had a regular arrangement. Figures A.45 & A.46 show $L(r)$ calculated for all parent ramets for plots where parent ramets had a regular arrangement. Figures A.47 & A.48 show $L(r)$ calculated for all daughter ramets in the regular treatment plots.

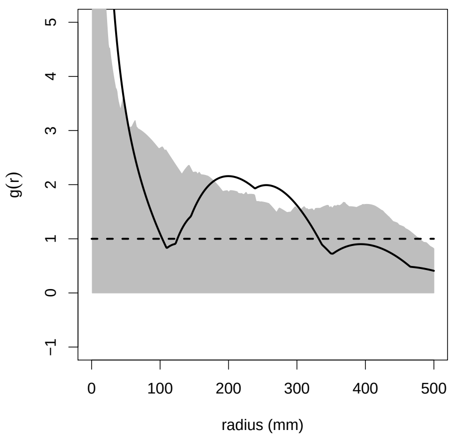


(a) Parent ramets (b) Daughter ramets

Figure 4.8: Pair-correlation function, $g(r)$, calculated for parent ramet locations (a) and daughter ramet locations (b) for Plot 7 where parents were randomly arranged at time of transplant. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations.

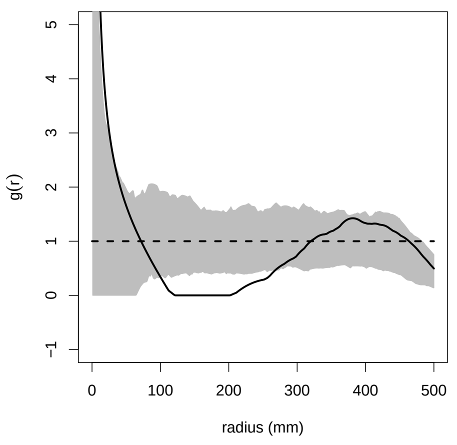


(a) Parent ramets

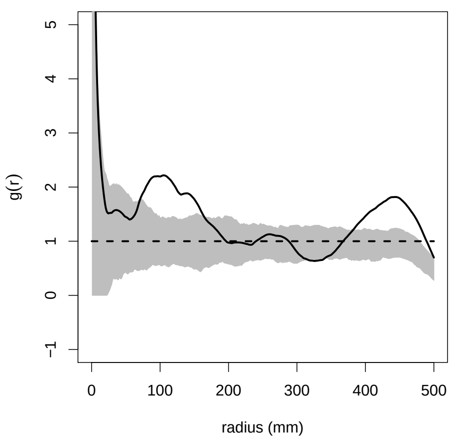


(b) Daughter ramets

Figure 4.9: Pair-correlation function, $g(r)$, calculated for parent ramet locations (a) and daughter ramet locations (b) for Plot 17 where parents were clustered at time of transplant. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations.



(a) Parent ramets



(b) Daughter ramets

Figure 4.10: Pair-correlation function, $g(r)$, calculated for parent ramet locations (a) and daughter ramet locations (b) for Plot 22 where parents were regularly arranged at time of transplant. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations.

4.4 Discussion

The analysis of artificial data shows that the genet randomization procedure is capable of characterizing the spatial structure of genets in terms of clustering, inhibition (regularity) and randomness. The analysis relies on the fundamental concept of the well-known and widely-applied point pattern analysis with a random Poisson point process as a model of CSR. An important aspect of this analysis is that it is a *constrained* randomization. Not all aspects of the data are randomized; the structure of the genet is maintained but translated randomly in the x and y plane so that the model of CSR is a *random genet process*. Further, by sampling only stolons that are from other genets, the analysis ignores clustering or inhibition within the genet structure and instead focuses on the spatial relationship between genets.

The genet randomization procedure was used to test the hypothesis that stolon spatial structure will not follow that of the initial planting arrangement because stolons will respond to neighbour genet shade by elongating stolons and respond to increased light in open areas by producing shorter stolons and more ramets. Analysis of the *F. virginiana* plot data suggests that the initial parent ramet configuration does not seem to influence the spatial structure of either daughter ramets or stolons. The constrained randomization shows that stolons were arranged randomly regardless of the spatial configuration of parents, suggesting that clonal spread of the transplanted *F. virginiana* is not influenced by the initial arrangement of parent ramets. This strongly suggests that *F. virginiana* does not follow an obvious set of growth rules. If simple growth rules dictated stolon growth, it would be expected that the structure of parent ramets would therefore result in corresponding structure in the stolon structure. The point pattern analysis of fibre points failed to detect repulsion or clustering of stolons, supporting the finding of the genet randomization.

The point pattern analyses of ramet locations detected the artificial spatial structure of parent ramets, correctly identifying the clustered, random or regular (inhibited) structure of the original transplants. The fact that daughter ramets did not adopt the spatial structure of their parents supports the idea that stolon structure does not follow repetitive, fixed rules, as such rules would have produced daughter spatial structure that mirrored that of parent ramets. It also may suggest that ramets are not placed preferentially in relation to existing parent plants.

The lack of structure is particularly interesting because *F. virginiana* does show spatial structure in its stolon arrangements in field plots (Chapters 2 & 3). Further, a clumped distribution of ramets at all scales has been found in other species (Hossaert-McKey and Jarry, 1992). One potential explanation for the lack of structure is the homogeneous nature of the transplant plots. The soil was tilled prior to transplanting, and it is assumed that this tilling would have mixed the soil into a homogeneous state in terms of physical structure and nutrients. This could explain the lack of spatial structure in stolons and daughter ramets, as homogeneous soil conditions would be expected not to drive foraging responses as heterogeneous soil conditions might.

Foraging as a unified concept applicable to clonal and non-clonal species has been proposed (de Kroon and Hutchings, 1995). Here, the ramets, leaves and root tips are the “resource acquisition structures” that are distributed in the environment by rhizomes and stolons, and stems and root branches respectively (de Kroon and Hutchings, 1995). de Kroon and

Hutchings (1995) define foraging as “the processes whereby an organism searches, or ramifies within its habitat, which enhances its acquisition of essential resources.”

Some clonal plants can preferentially allocate plant parts (e.g., roots) to take advantage of heterogeneous soil conditions and thus increase productivity in the form of biomass (Birch and Hutchings, 1994). It has been found that resources are translocated along stolons to developing ramets from established ramets (Slade and Hutchings, 1987a). Hartnett and Bazzaz (1985) found that clonal growth in *Solidago canadensis*, measured by length of rhizomes and number of rhizomes per plant, was density-dependent. Sampaio et al. (2004) have found evidence of directional growth as a means of habitat selection in *Aechmea nudicaulis*. Slade and Hutchings (1987a) found that, in *Glechoma hederacea*, high resource availability resulted in more intensive foraging while low resource availability caused more extensive foraging.

There is little doubt that the *F. virginiana* ramets that are physically connected by stolons are also physiologically connected. There is a considerable literature discussing the physiological integration of *Fragaria* spp. Alpert and Mooney (1986) have found that clonal integration of *F. chiloensis* ramets increased survival under adverse conditions. Alpert (1996) found that *F. chiloensis* ramets on a stolon shared nitrogen, but large net transfers only occurred to younger ramets from older ramets. In younger ramets the sharing resulted in an increase in biomass and allocation to stolons, but in older ramets the sharing resulted in a possible decrease in biomass.

Alpert (1999a) found that ramets of *F. chiloensis* taken from homogeneous environments shared fewer resources than ramets taken from heterogeneous environments. Further, Alpert (1991) found that nitrogen is shared between ramets of *F. chiloensis* and heterogeneous soil distribution can influence the architecture of the plant by altering the production of ramets and stolons. This suggests a dynamic development of connected ramets whereby potential site heterogeneity would be expected to influence the clone structure in the plots. Roiloa and Retuerto (2006a) found that physiological integration in *F. chiloensis* was important in stressful, heterogeneous conditions whereby demand for resources by daughter ramets in unfavourable habitats increased the photosynthetic efficiency of parents. Zhang et al. (2009) found that clonal integration in *Fragaria orientalis* allowed the maintenance of ramets in water-stressed conditions. Similarly, Mao et al. (2009) showed that water moves through *Fragaria ananassa* stolons to drought-stressed ramets due to a water potential gradient.

As the density of ramets and stolons changed over time, it is possible that the production of stolons could also vary. *F. virginiana* will increase biomass allotment to vegetative reproductive organs when plant density is low (Holler and Abrahamson, 1977). *F. virginiana* produced higher proportions of stolon biomass in low density field and wooded sites than higher density sites (Holler and Abrahamson, 1977). In addition, Xiao et al. (2011) found that the sexual reproduction, growth, and survival of daughter ramets increased with clonal integration while the parent ramet performance was reduced. The detection of such a phenomena could be possible by sampling the plot repeatedly at evenly spaced intervals and examining changes in stolon production.

However, *F. virginiana* may have other qualities which negate the need to place ramets carefully. *F. chiloensis* has been found to segregate roots between connected ramets, which suggests that integration of clones for signaling between ramets makes possible root separation (Holzapfel and Alpert, 2003).

4.5 Conclusions

The genet randomization analyses provides a method to investigate the spatial arrangement of genets while excluding the internal genet structure. Coupled with methods that examine the general fibre structure, it improves the level of understanding that can be achieved.

The transplant experiment suggested that *F. virginiana* adapts its foraging behaviour to its environment, reacting to a homogeneous resource distribution by random clonal spread and random establishment of daughter ramets in the short term. Longer term monitoring would be required to determine if there is a change in stolon spatial structure formation and placement of ramets over time.

Chapter 5

Characterizing heterogeneity in linear spatial data

5.1 Introduction

Foraging has been defined as a process where an organism explores within its habitat in such a way so as to increase the acquisition of resources (de Kroon and Hutchings, 1995). This is particularly relevant in clonal plants that maintain physiological connections between resource acquisition structures (e.g., leaves and roots). If such plants can locate such structures in resource-rich regions and/or avoid unfavourable areas, it should lead to an overall benefit to the clone and an improvement in fitness. For example, there is some evidence for environmental heterogeneity resulting in adaptation in the form of plastic foraging responses in clonal plants (*Ranunculus reptans*) (van Kleunen and Fischer, 2001). This agrees with Hutchings and Price (1993), who argued that clonal plants respond to environmental quality, and thus environmental heterogeneity.

Such behaviour has been found in various species of clonal plants.

Heterogeneity in soil nutrients has been shown to influence plant competition and the placement of roots (Day et al., 2003b). Day et al. (2003b) grew *Briza media* and *Festuca ovina* in both homogeneous and heterogeneous soil conditions and found an increase in competition (as measured by reduction in yield) and preferential placement of roots in the heterogeneous treatment. *Chardamine hirsuta* has been found to respond to heterogeneous nutrient supplies by producing more root, shoot and total biomass when grown in soils with heterogeneous nutrient distribution (Day et al., 2003a). Li and Wang (2011) found nitrate translocation to be important in *Eichhornia crassipes*, and Sun et al. (2011) found buffalograss (*Buchloë dactyloides*) increased production of ramets in high nutrient patches while at the same time reducing production of ramets in patches that were nutrient poor. Roiloa and Retuerto (2007) found that microtopography causing environmental heterogeneity may have costs for ramet development. Spatially explicit modelling of clonal growth in heterogeneous environments has shown that in some cases plasticity of spacer length (short versus long spacers) can influence the ratio of ramet placement in resource-rich sites, but other environments resulted in spacer length plasticity having no benefit in ramet placement over a null model with random ramet placement (Oborny, 1994). The performance of a clonal species in a heterogeneous environment is not only important to understand the ecology of that particular species, but it may also be important to developing an understanding of other species. For example, Eilts et al. (2011) have found that clonal plants can influence species diversity in heterogeneous environments when they are able to span the distances between resource patches.

Fragaria spp. have been found to exhibit foraging responses as well.

F. chiloensis has been found to shorten stolon length, increase stolon numbers, and increase ramet densities in response to increased amounts of nitrogen (Tworkoski et al., 2001). *F. vesca* has been found to forage for resources in heterogeneous environments (Roiloa and Retuerto, 2006c), and in the case of heterogeneously contaminated soils, *F. vesca* placed ramets randomly but established ramets in non-contaminated (favourable) patches (Roiloa and Retuerto, 2006b). Roiloa and Retuerto (2006b) also found that *F. vesca* parent ramets produced much more reproductive biomass if a daughter ramet had colonized a contaminated region of soil, indicating a possible escape response.

Trifolium repens L. has shown significant increase in branching when growing in soil versus growing in sand (Welham et al., 2002). Such a response could be considered a foraging strategy, as increasing branching could potentially also increase the number of ramets in a given area. Genotypes of *Ranunculus reptans* from competitive heterogeneous environments were found to be more able to respond to competition by altering stolon internode angle and stolon length than genotypes from a homogeneous environment without competition (van Kleunen and Fischer, 2001). Salzman (1985) showed habitat selection in *Ambrosia psilostachya* where non-saline soil was preferred over saline soil. She also found that genotypes varied in their discrimination between saline and non-saline patches.

Increases in soil fertility has been found to increase ramet density, mean height and total biomass in some clonal plants (*Calamagrostis epigejos* (L.) Roth, *Solidago canadensis* L., and *Tanacetum vulgare* L.) (Rebele, 2000). However, in *Trientalis europaea* L., increased nutrients have been found to influence tuber growth in but not stolon growth

(Piqueras et al., 1999). Macdonald and Lieffers (1993) have found *Calamagrostis canadensis* to exhibit a foraging response when confronted with heterogeneous resource distribution, and Kleijn and Van Groenendael (1999) found that *Elymus repens* selectively extended stolons into favourable patches, suggesting foraging in heterogeneous habitats.

Clonal foraging responses are not always consistent across heterogeneity types. For example, when grown in areas of heterogeneous plant densities, *Elymus lanceolatus* ssp. Lanceolatus, did not seem to exhibit clonal foraging, which was opposite its behaviour in the case of nutrient heterogeneity (Humphrey and Pyke, 2001). Despite this work, more work is needed to improve our understanding of foraging. The case has recently been made for the examination of heterogeneity explicitly in growth experiments (Hutchings and John, 2004), and recent work has suggested that the effects of soil heterogeneity and plant foraging precision on plants and plant communities requires further exploration (Kembel and Cahill, 2005).

The study of environmental heterogeneity and its effect on clonal growth presents additional challenges that arise from heterogeneously distributed data. These challenges have been known for some time. Pollard (1971) discusses methods of distance estimators of density and illustrates the bias in density estimates that can occur in cases where the density of plants (points) are not uniform over the study region. Given two areas, both with plants distributed at random, that differ in density, λ, it might be desirable to estimate the densities in these two areas (both density and size of the two regions are unknown). In light of this simple example, the problems associated with an area that has an unknown set of “regions” with unknown mean density and variance become obvious. The analogy can be extended to linear data as well. In another example, circles are placed at random over an area and Poisson-distributed numbers of points are distributed (uniformly and independently) at random within each circle. Only the size of the circles vary between the two realizations. Only the radius of the circles separate the two processes, the smaller radius producing a pattern that would be considered obviously aggregated and the larger radius producing a pattern that would be considered globally heterogeneous. If one considers that a continuum of circle radii exist between these two examples, this is problematic and disheartening because it illustrates the lack of clear distinction between clustering and heterogeneity from the point of view of detection and characterization. The problem of heterogeneity, illustrated by this example is essentially a problem of scale of observation. Early attempts at dealing with spatial heterogeneity focused simply on identifying the pattern as homogeneous or heterogeneous using global analysis. Diggle (1977) suggested a two-stage procedure to detect random heterogeneity where a preliminary test of randomness is followed by a heterogeneity test. This allows for a four-way classification of spatial point patterns into regular, random-homogeneous, random-heterogeneous or aggregated types. This method is a rough, global classification and does not provide for local, exploratory analysis.

Arbia (1990) developed a method that essentially uses local analyzes to describe the non-stationarity of a particular region. This approach focuses on lattice processes (processes operating over a grid) and is based on resampling the spatial data several times using a sliding window over an area of *n* by *m* cells. Instead of a global estimate of the selected measures (e.g. mean, variance), the method generates a set of estimates over the grid.

Pélissier and Goreaud (2001) proposed the delineation of regions of interest that would be homogeneous in terms of point density and of second order characteristics. Local neighbour density and second order neighbour functions can be used to draw “isolines” of density and then use special versions of Ripley’s K in each homogeneous region. However, this is only applicable to “simple cases of heterogeneous vegetation” that lend themselves to objective delineation of homogeneous subplots but not more complex problems such as smooth intensity gradients. The applications of such methods would also be problematic to fibre system patterns due to the increased complexity of the pattern under consideration.

The methods of Brix et al. (2001) and Couteron et al. (2003), however, are promising because they combine some conceptual aspects of moving windows and delineating homogeneous regions. This chapter takes initial steps in applying the methods of Couteron et al. (2003) to examine the heterogeneity of spatial patterns produced by linear features (“fibre processes”) rather than points (“point processes”). It examines the spatial structure of *F. virginiana* that have been grown in heterogeneous soil conditions, and it applies various methods of linear data analysis to characterize the stolon spatial structure that arise under these soil conditions. The biological hypothesis is that *F. virginiana* will respond to nutrient heterogeneity by shortening stolon length, increasing stolon production, and increasing ramet production in regions of elevated nutrient concentration. As a result, stolon spatial structure will vary with nutrient heterogeneity spatial structure.

5.2 Methods

5.3 Local Global Analysis

The general framework of the fibre heterogeneity analysis uses the work of Couteron et al. (2003) as a foundation, but it has been adapted for use with linear data. The methodological procedure follows a series of steps that can be grouped broadly into two main families, components relating to (1) local analysis and (2) global analysis. This combined local-global analysis is applied with the aim of detecting the scale of the heterogeneity in the plot as measured by a trade-off between local variance and global bias.

Local analysis

The local analysis focuses on finding the "optimal" partitioning of the study area into cells of a given size by finding the cell partition size that minimizes the total mean squared error of a statistic calculated both globally and locally over the study plot. For this exercise the mean interpoint distance between points of intersection was used as the metric.

One hundred random fibre points were distributed over the fibres in the study plot. The rhizome mapping was then divided into square cells of width w. In each of the cells a Monte-Carlo simulation was performed that randomized the random fibre points in that cell. The fibre points in the cell were then compared to an equal number of completely random points using the mean interpoint distance. It was assumed that fibres distributed at random across the cell (a homogeneous and isotropic process) would produce fibre points with a specific distribution of interpoint distances. Departures from this distribution will indicate departures from complete spatial randomness of fibre position and orientation. The significance of the test is given by the relative number of times the simulated interpoint distance is greater than the observed distance. The Monte-Carlo simulation was repeated in each cell for a range of cell sizes (divisions of 3, 4, 5, 6, 8, 10, 12, 14, 16). See Couteron et al. (2003) for a description of the Monte-Carlo simulation.

The Monte-Carlo simulation produces a significance value for the test in each cell of the study area. The *p*-value should be low when there is inhibition of intersection points in the quadrat and high when there is clustering in the quadrat. The number of horizontal and vertical partitions determines the quadrat size in which the analysis is carried out. Detecting the optimal quadrat size provides an indication of the scale at which analyses can be conducted whereby the effects of heterogeneity are at a minimum. By extension, it can be expected that the areas within the optimal quadrats are the most homogeneous. Because the quadrats will differ from each other in terms of their internal properties, this approach can be used as a means of identifying the scale of plot heterogeneity.

The optimal quadrat size is defined as the quadrat size that minimizes the total mean squared error (Eqn. 5.9), the trade-off between squared global bias (the squared mean of all local biases, $bs(i, j)$, and variance (Eqn. 5.8) attached to the statistic used for the global test. Couteron et al. (2003) states that the squared bias will likely increase as cell size increases due to the points within the cells becoming increasingly inhomogeneous and that variance will likely be high when quadrats are very small due to small numbers of points. The minimum of the error function, then, represents the balance of these two phenomena.

In order to estimate local bias, $bs(i, j)$, (Eqn. 5.7) first- and second-order intensity functions (Couteron et al., 2003) were calculated for each cell (i, j) based on the density of fibre points within each cell.

The first-order intensity functions include both horizontal (Eqn. 5.1) and vertical (Eqn. 5.2) components. These components examine the density in cells adjacent to the current cell, (i, j), horizontally and vertically. The second-order intensity functions include horizontal (Eqn. 5.3) and vertical (Eqn. 5.4) components, as well as diagonal components (Eqn. 5.5) and (Eqn. 5.6). These components consider the first-order functions adjacent to the current cell, (i, j), horizontally, vertically, and diagonally. These components are listed below:

$$hd_1(i, j) = \frac{(d(i, j+1) - d(i, j-1))}{(2d(i, j))}, \tag{5.1}$$

$$vd_1(i, j) = \frac{(d(i+1, j) - d(i-1, j))}{(2d(i, j))}, \tag{5.2}$$

$$hd_2(i, j) = \frac{(hd_1(i, j+1) - hd_1(i, j-1))}{(2)}, \tag{5.3}$$

$$vd_2(i, j) = \frac{(vd_1(i+1, j) - vd_1(i-1, j))}{(2)}, \tag{5.4}$$

$$vhd_2(i, j) = \frac{(vd_1(i, j+1) - vd_1(i, j-1))}{(2)}, \tag{5.5}$$

$$vhd_2(i, j) = \frac{(hd_1(i + 1, j) - hd_1(i - 1, j))}{(2)}, \tag{5.6}$$

where $d(i, j)$ is the density of intersection points in the cell (i, j).

The intensity functions are stored in the matrices $\mathbf{A_{ij}}$ and $\mathbf{B_{ij}}$ and used to calculate the bias for the cell (i, j) (Eqn. 5.7),

$$\mathbf{A_{ij}} = \begin{bmatrix} hd_1(i, j) & vd_1(i, j) \end{bmatrix}$$

$$\mathbf{B_{ij}} = \begin{bmatrix} hd_2(i, j) & hvd_2(i, j) \\ hvd_2(i, j) & vd_2(i, j) \end{bmatrix}$$

$$bs(i, j) = \frac{n_{ij}}{24} trace(\mathbf{B_{ij}}) - \left(\sum_N n_{ij} \mathbf{u^{Tr}} \mathbf{B_{ij}} \mathbf{u} + \frac{n_{ij}(n_{ij} - 1)}{2} \mathbf{u^{Tr}} (\mathbf{A_{ij}} \mathbf{A_{ij}^{Tr}}) \mathbf{w} \right) / N_{sim}, \tag{5.7}$$

where n_{ij} is the number of points in the cell, **u** and **w** are arrays containing the x and y values of random points, and N_{sim} is the number of simulations in which there are more than one point in the cell.

Global analysis

The global variance that results from a given cell size (or number of divisions) is then calculated (according to Couteron et al. (2003)) via Eqn. 5.8,

$$\sigma^2 = \frac{1}{12N_q}, \tag{5.8}$$

where N_q is the proportion of cells having more than one point (and as a result also have a bias estimate). The variance measure used here is altered slightly from that of Couteron et al. (2003), who recommend N_q as the absolute number of cells having more than one point, as it was found that use of an absolute value for N_q can not produce the behaviour described by their results. In addition, the variances and biases were found to differ widely in their scales, such that the variance estimates were orders of magnitude smaller than the bias estimates. The low value of the variances is obvious from the structure of Eqn. 5.8. To account for this both measures were standardized such that each ranged between [0,1].

A measure of the mean squared error is then given by Eqn. 5.9,

$$E = b^2 + \sigma^2, \tag{5.9}$$

where b^2 is the squared mean bias of all cells having more than one point. The cell size that minimizes this value is then chosen as the cell size that provides the best trade-off between small cell size (and increasing variance) and large cell size (and increasing bias).

In addition to the local-global analysis, a circle sampling analysis, fibre point analysis and point pattern analysis of daughter ramets was conducted on each plot.

All linear data analyses were conducted in the Python programming language using the *Linear* software library developed as a component of the thesis research (see Appendix B). Point pattern analyses were conducted using R as in previous chapters.

Field data

In the summer of 2006, a soil nutrient heterogeneity experiment was conducted in plots at the Ellerslie research station. Five treatments were established that attempted to alter the spatial distribution of soil nutrients in 2 m × 2 m plots. Each treatment was replicated four times for at total of 20 plots, and the plots were randomized. Four *F. virginiana* plants were planted in specific locations in each plot in June. The plants had been transplanted at the University of Alberta Ellerslie Research Station in the spring of 2005. They were then moved from their transplant location into their experimental plot location. The five nutrient treatments are illustrated in Figure 5.1. Grey areas represent regions that received nutrient application. Each 4 m^2 plot received 33.9 mL of 24-8-16 (24% N, 8% P_2O_5, 16% K_2O) all purpose fertilizer. The dry fertilizer was spread evenly by hand inside predefined regions of the plot, and it was assumed that there would be little migration of the nutrients into adjacent non-fertilized areas. The plots were fertilized twice throughout the duration of experiment (June 26 & July 15).

Plots were imaged with a digital camera on (July 17 & July 26 (approx. 2 & four weeks)) using the sampling tripod used for sampling the 2005 Ellerslie plots. Digital images were assembled into a composite image and scaled using the same methodology as in Chapter 4. Each composite image image was then converted from raster to vector data by hand using vector graphics software as in Chapter 4. The software was used to trace manually all stolons, identify stolon connections and branches, and mark the location of daughter and parent ramets. The vector dataset describing each plot was analyzed using the *Linear* software library (see Appendix B).

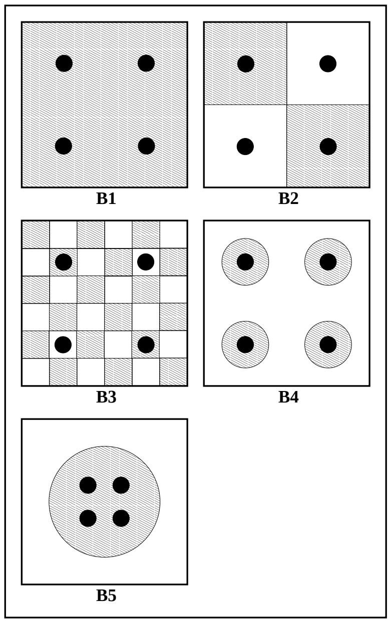


Figure 5.1: Nutrient experimental design. Each treatment was replicated four times. Four plants where transplanted into each plot (5 treatments, 4 replicates = 80 plants in 20 plots). Grey areas represent regions that received nutrient application. Each 4 m^2 plot received 33.9 mL of 24-8-16 (24% N, 8% P_2O_5, 16% K_2O) all purpose fertilizer. Plot size is 2 m x 2 m.

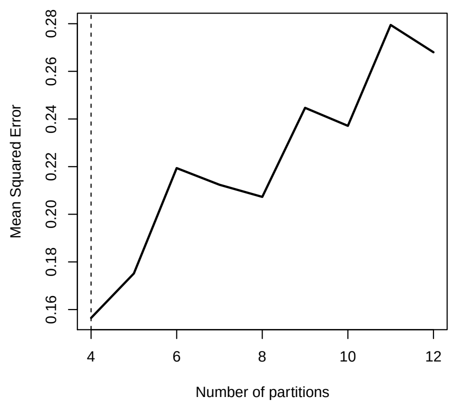
5.4 Results

5.4.1 Local Global Analysis

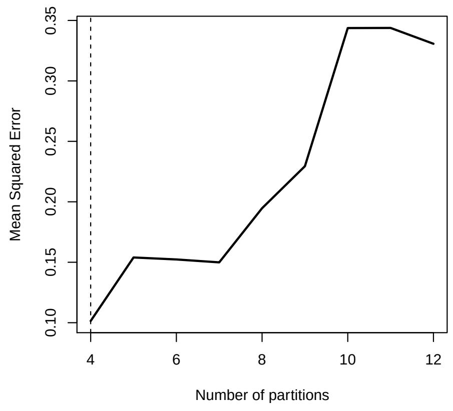
Treatment B1

Figure 5.2 shows the results of the local partitioning analysis for treatment B1. Plots 1, 16, and 21 all show a recommended plot size of 4 partitions (16 cells), while the recommended plot size for Plot 39 is 6 partitions (36 cells). The minimum of the mean squared error represents the best trade-off between bias and variance (the recommended level of partitioning).

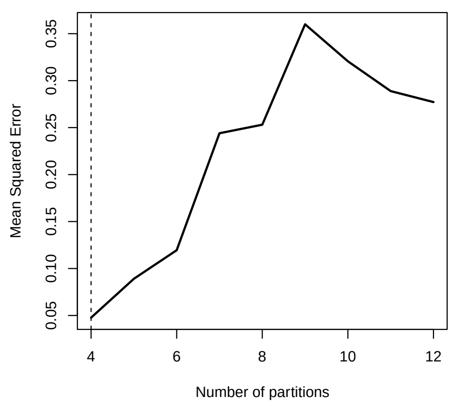
Figure 5.3 shows plots showing the *p*-values for Monte Carlo analysis at the scale of plot division recommended by the partitioning analysis (Figure 5.2). Lighter values indicate plots with high *p*-values indicating clustering of fibre points relative to a random point process. This provides a spatially explicit indication of the location and scale of clustering in the plot.



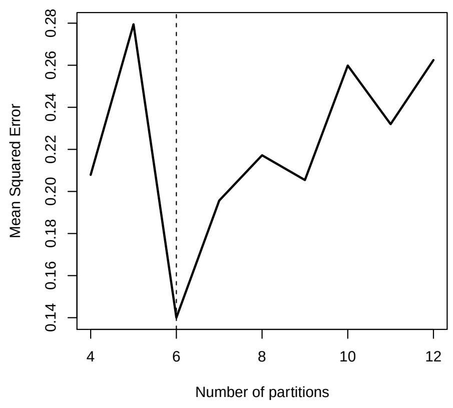
(a) Plot 01



(b) Plot 16

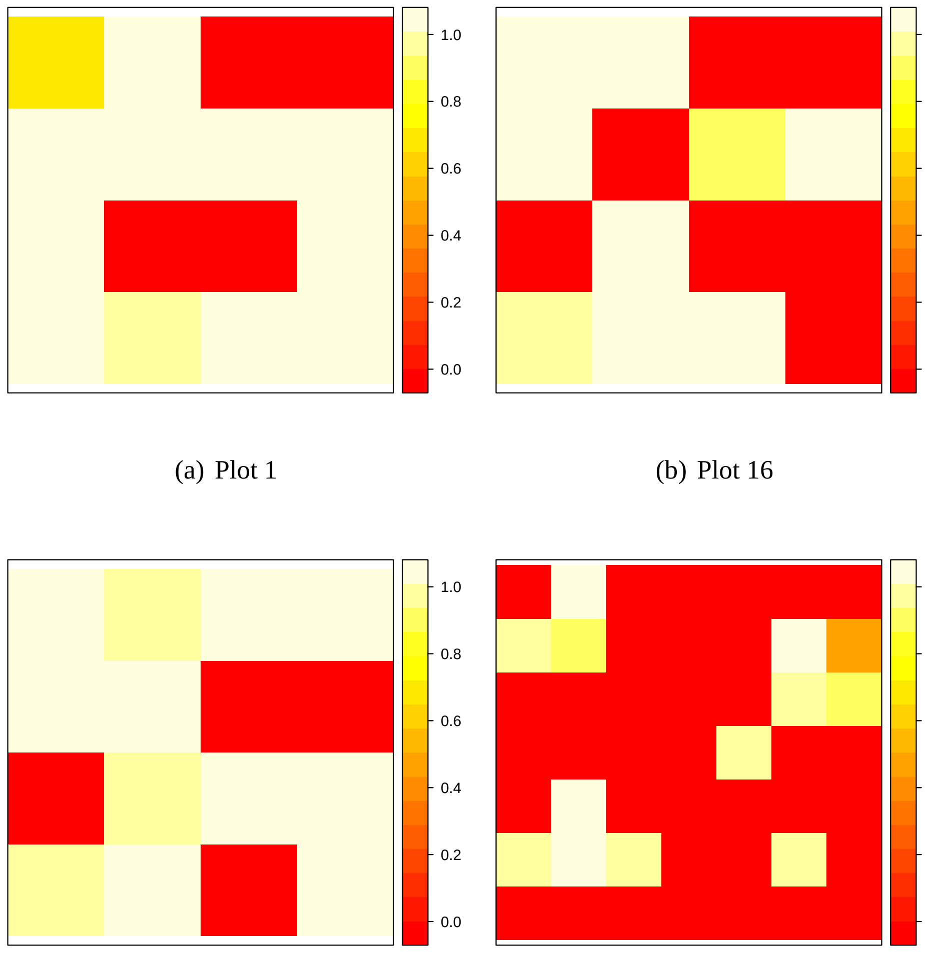


(c) Plot 21



(d) Plot 39

Figure 5.2: The mean squared error calculated for the B1 treatment plots. The mean squared error is plotted against the number of partitions in the x and y direction. The vertical dashed line represents the minimum partition size.



(a) Plot 1 (b) Plot 16

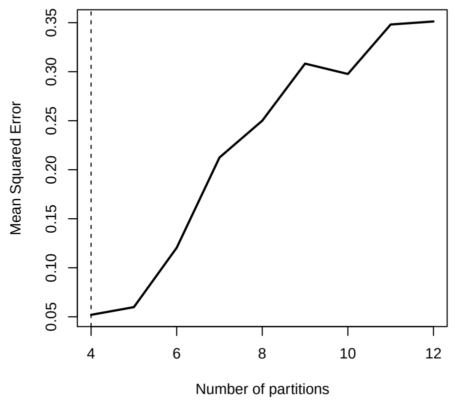
(c) Plot 21 (d) Plot 39

Figure 5.3: The *p*-value maps for the B1 treatment plots. The *p*-value is the significance of a Monte Carlo point pattern analysis conducted in each partition of the plot. The *p*-value map for each plot is shown with the number of partitions indicated in the partitioning analysis. Lighter colours represent higher significance (clustering of points in the partition).

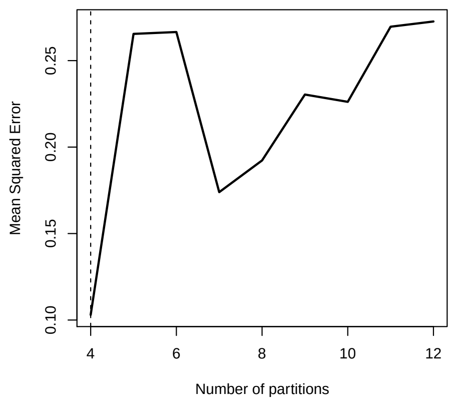
Treatment B2

Figure 5.4 shows the results of the local partitioning analysis for treatment B2. Plots 13, 24, 35, and 37 all show a recommended plot size of 4 partitions (16 cells).

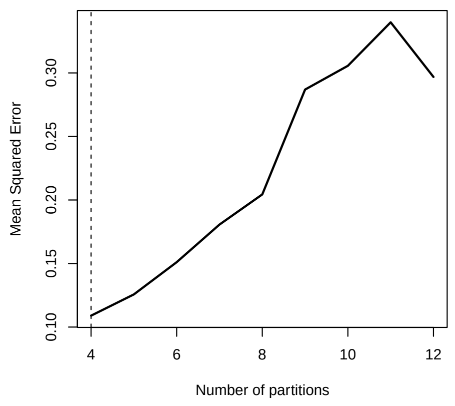
Figure 5.5 shows plots showing the *p*-values for Monte Carlo analysis at the scale of plot division recommended by the partitioning analysis (Figure 5.4). Lighter values indicate plots with high *p*-values indicating clustering of fibre points relative to a random point process.



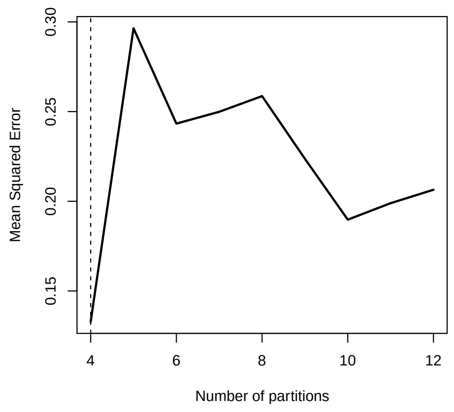
(a) Plot 13



(b) Plot 24

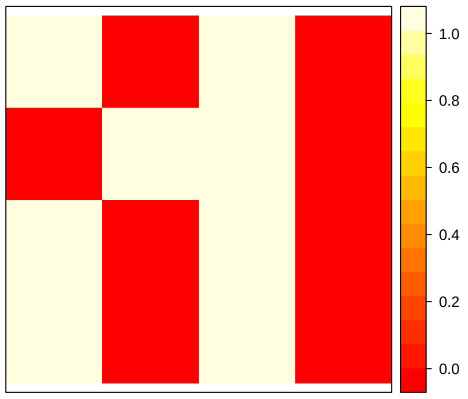


(c) Plot 35

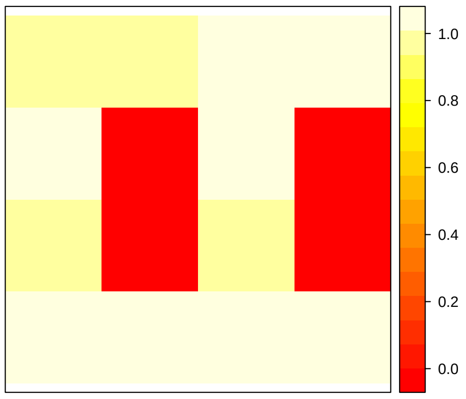


(d) Plot 37

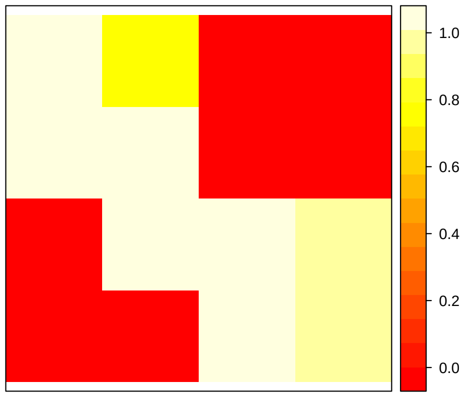
Figure 5.4: The mean squared error calculated for the B2 treatment plots. The mean squared error is plotted against the number of partitions in the x and y direction. The vertical dashed line represents the minimum partition size.



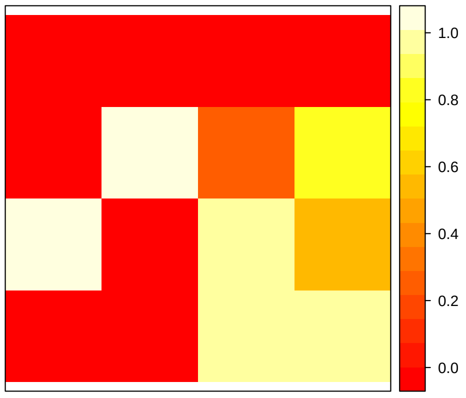
(a) Plot 13



(b) Plot 24



(c) Plot 35



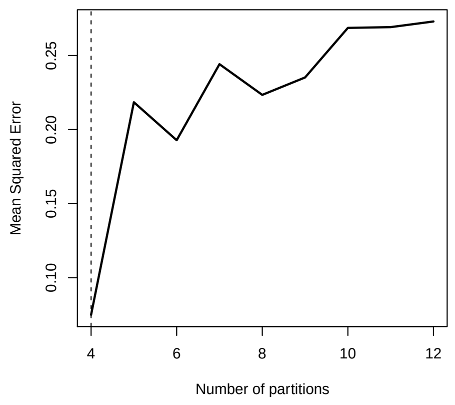
(d) Plot 37

Figure 5.5: The *p*-value maps for the B2 treatment plots. The *p*-value is the significance of a Monte Carlo point pattern analysis conducted in each partition of the plot. The *p*-value map for each plot is shown with the number of partitions indicated in the partitioning analysis. Lighter colours represent higher significance (clustering of points in the partition).

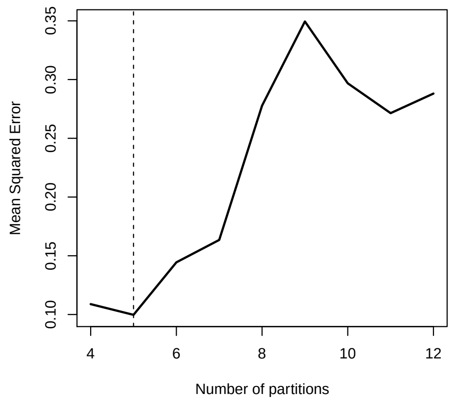
Treatment B3

Figure 5.6 shows the results of the local partitioning analysis for treatment B3. The recommended plot size for Plots 3 and 36 is 4 partitions (16 cells). The recommended plot size for Plot 31 and 38 is 5 and 11 partitions respectively.

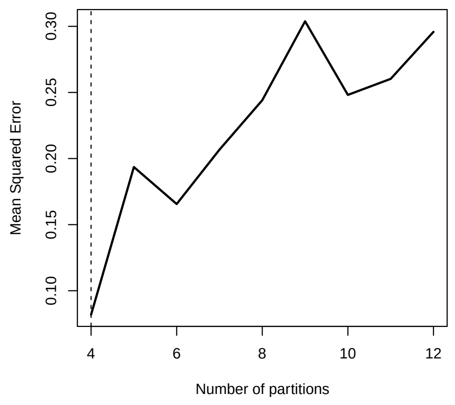
Figure 5.7 shows plots showing the *p*-values for Monte Carlo analysis at the scale of plot division recommended by the partitioning analysis (Figure 5.6). Lighter values indicate plots with high *p*-values indicating clustering of fibre points relate to a random point process.



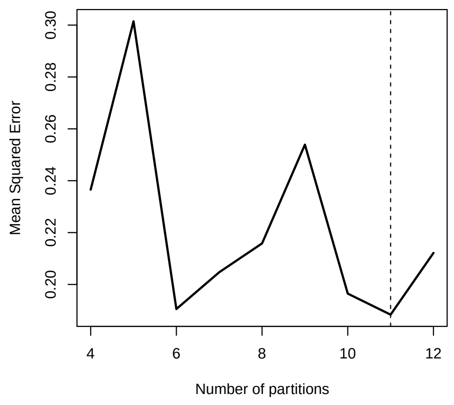
(a) Plot 03



(b) Plot 31

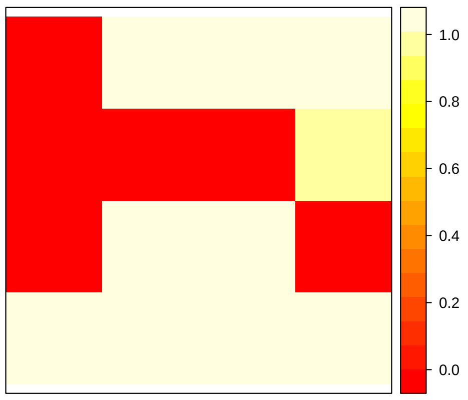


(c) Plot 36

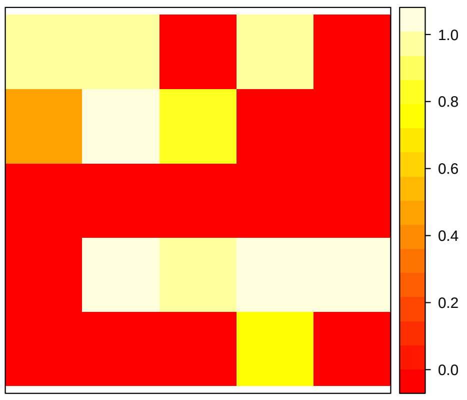


(d) Plot 38

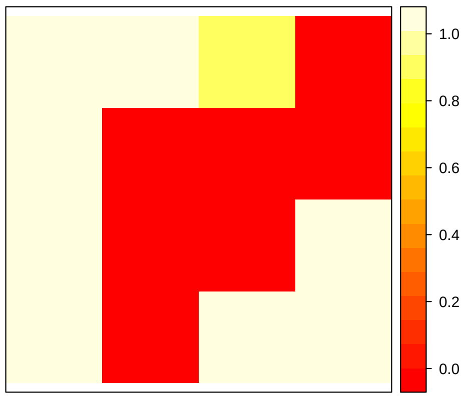
Figure 5.6: The mean squared error calculated for the B3 treatment plots. The mean squared error is plotted against the number of partitions in the x and y direction. The vertical dashed line represents the minimum partition size.



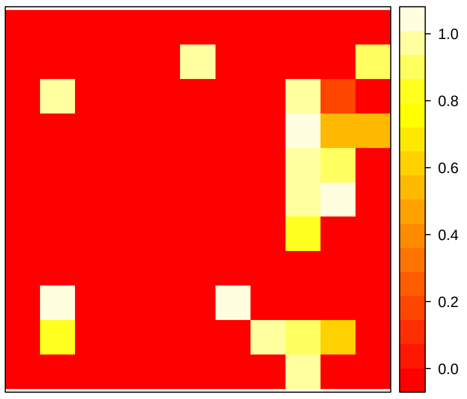
(a) Plot 3



(b) Plot 31



(c) Plot 36



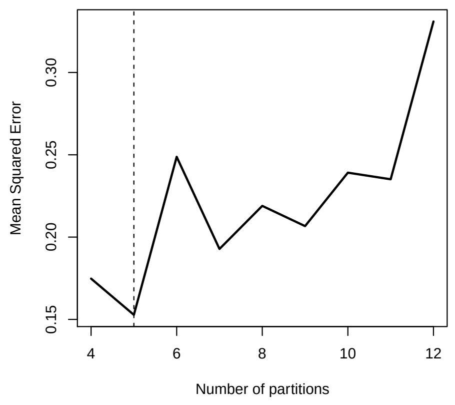
(d) 38

Figure 5.7: The *p*-value maps for the B3 treatment plots. The *p*-value is the significance of a Monte Carlo point pattern analysis conducted in each partition of the plot. The *p*-value map for each plot is shown with the number of partitions indicated in the partitioning analysis. Lighter colours represent higher significance (clustering of points in the partition).

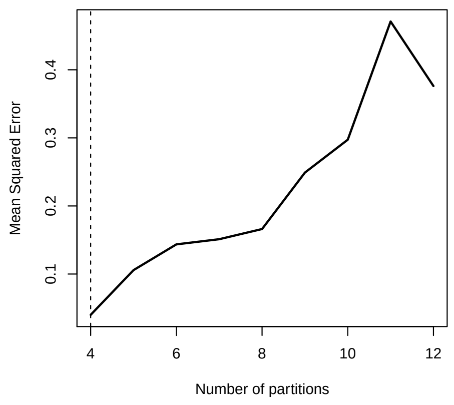
Treatment B4

Figure 5.8 shows the results of the local partitioning analysis for treatment B4. The recommended plot size for plots 30 and 40 is 4 partitions (16 divisions). The recommended plot size for Plots 2 and 48 is 5 and 7 partitions respectively.

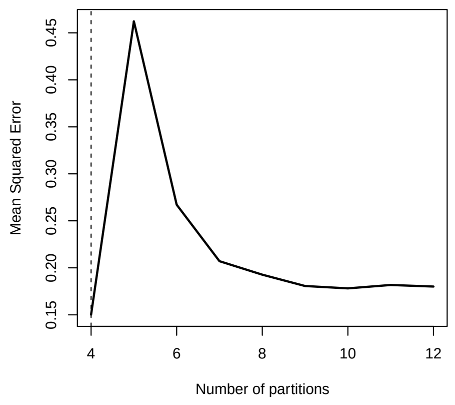
Figure 5.9 shows plots showing the *p*-values for Monte Carlo analysis at the scale of plot division recommended by the partitioning analysis (Figure 5.8). Lighter values indicate plots with high *p*-values indicating clustering of fibre points relative to a random point process.



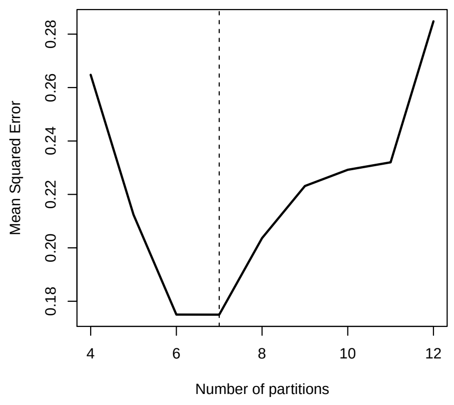
(a) Plot 02



(b) Plot 30

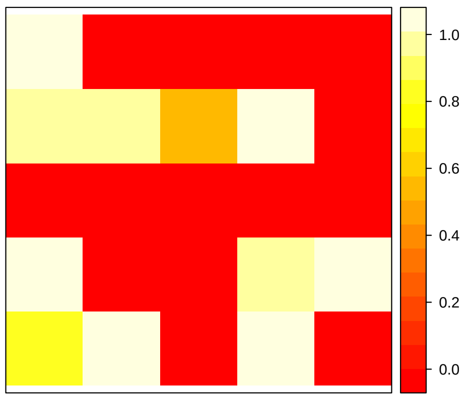


(c) Plot 40

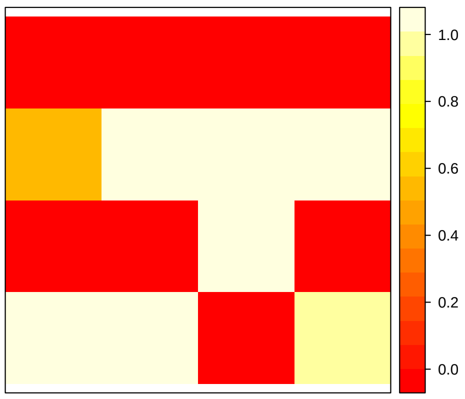


(d) Plot 48

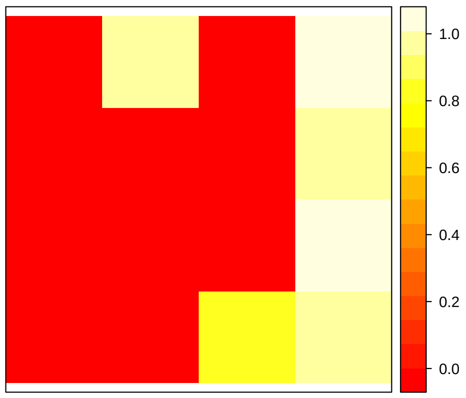
Figure 5.8: The mean squared error calculated for the B4 treatment plots. The mean squared error is plotted against the number of partitions in the x and y direction. The vertical dashed line represents the minimum partition size.



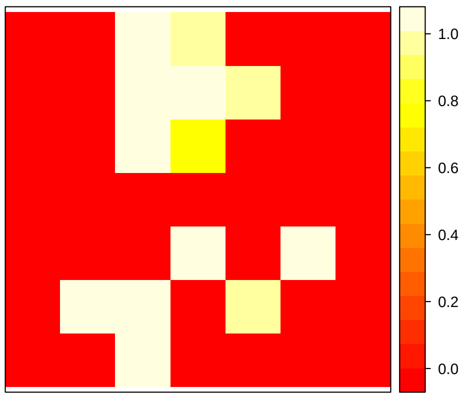
(a) Plot 2



(b) Plot 30



(c) 40



(d) 48

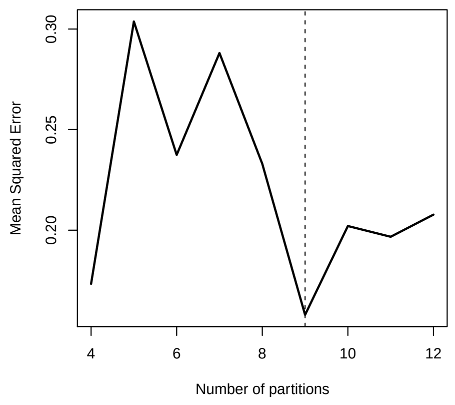
Figure 5.9: The *p*-value maps for the B4 treatment plots. The *p*-value is the significance of a Monte Carlo point pattern analsysis conducted in each partition of the plot. The *p*-value map for each plot is shown with the number of partitions indicated in the partitioning analysis. Lighter colours represent higher significance (clustering of points in the partition).

Treatment B5

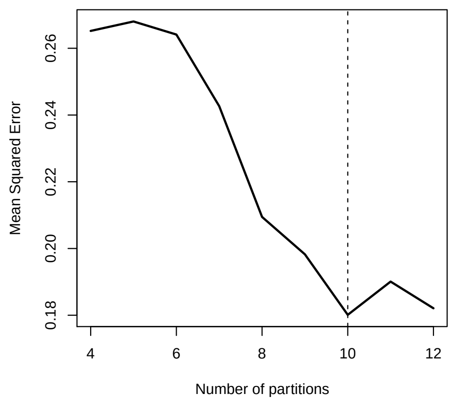
Figure 5.10 shows the results of the local partitioning analsysis for treatment B5. The number of partitions for Plots 5, 19, 42, and 43 is 9, 10, 10 and 8 partitions, respectively.

Figure 5.11 shows plots showing the *p*-values for Monte Carlo analysis at the scale of plot division recommended by the partitioning analysis (Figure 5.10). Lighter values indicate plots with high *p*-values indicating clustering of fibre points relative to a random point process.

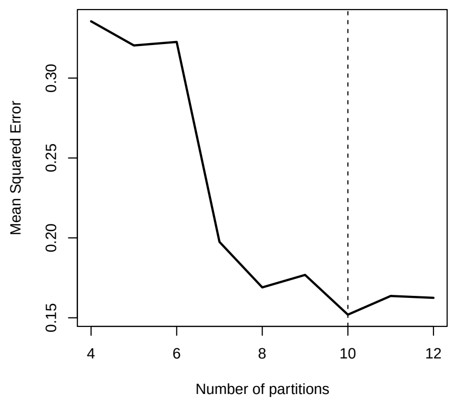
Figure 5.12(a) and 5.12(b) shows the original data for Plot 1 and the results of the partitioning analysis, respectively. Figure 5.12(c) shows the genet data from Plot 1 artificially manipulated so that the parent ramets are in a similar arrangement to those in treatment B5. Figure 5.12(d) shows the corresponding results from the partitioning analysis. In both the original data and the translated data, the partitioning analysis recommended partitioning by 4 (16 cells). The mean squared error curve for the artificial data (Figure 5.12(d)) does differ from that of the original data (Figure 5.12(b)) in that it is unimodal in nature, starting to decrease after a maximum at a partitioning level of 7. The mean squared error curve for the original data is monotonic.



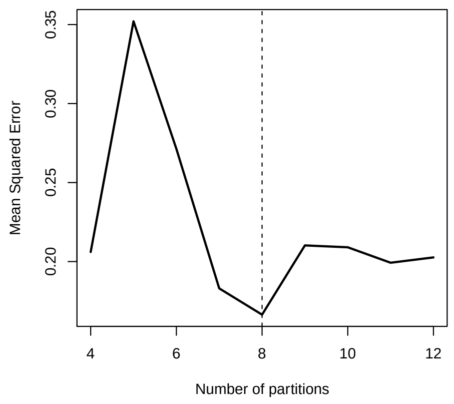
(a) Plot 05



(b) Plot 19

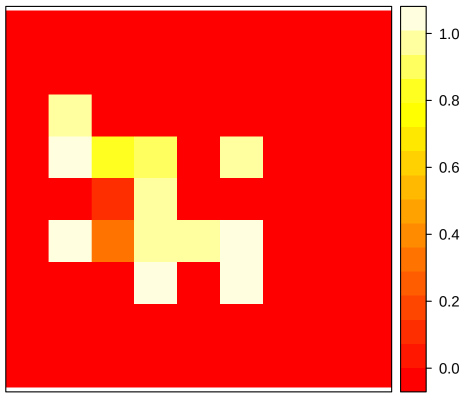


(c) Plot 42

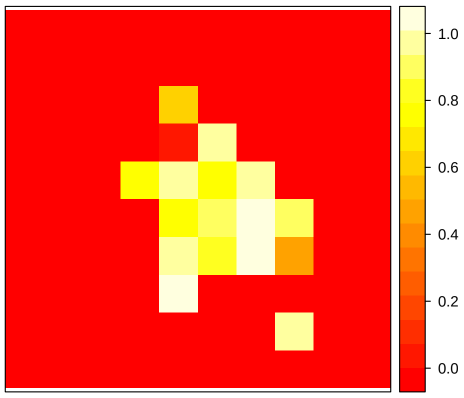


(d) Plot 43

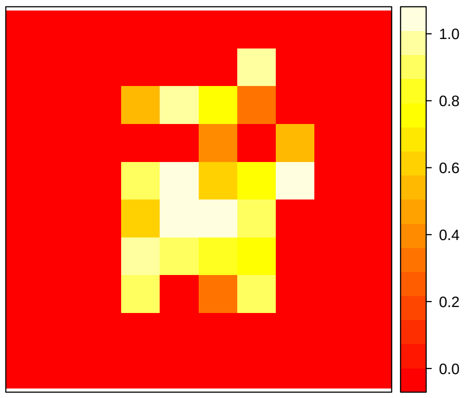
Figure 5.10: The mean squared error calculated for the B5 treatment plots. The mean squared error is plotted against the number of partitions in the x and y direction. The vertical dashed line represents the minimum partition size.



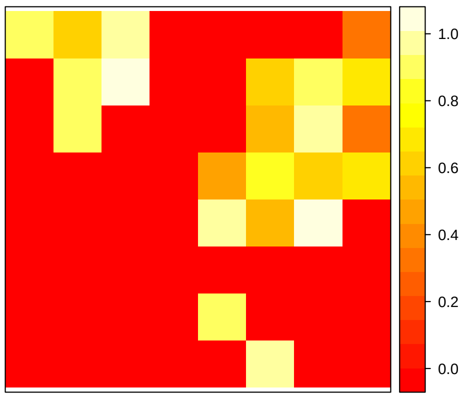
(a) Plot 5



(b) Plot 19

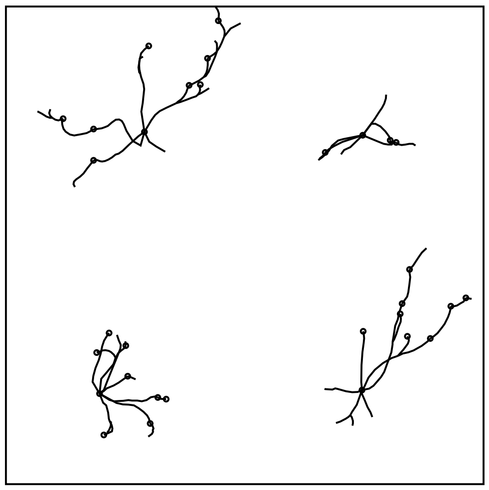


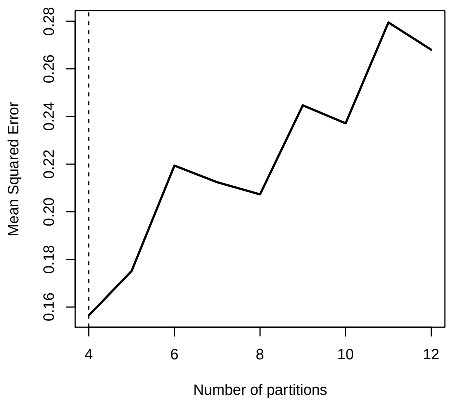
(c) 42



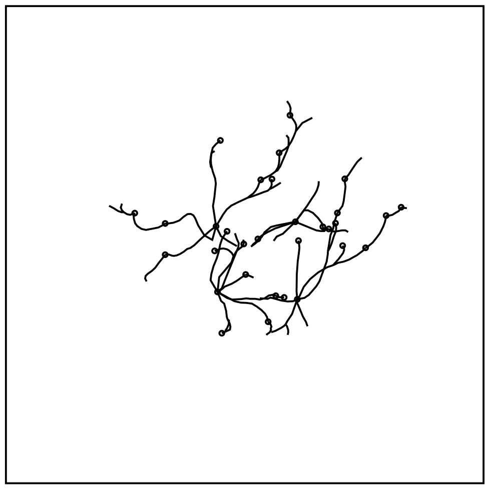
(d) Plot 43

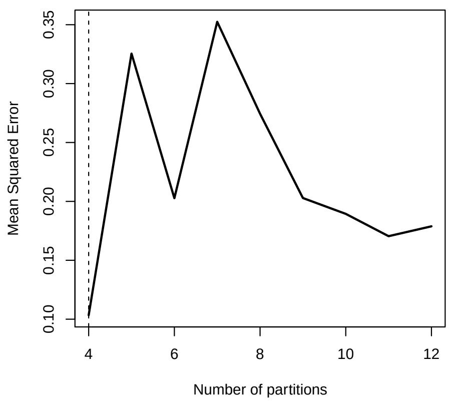
Figure 5.11: The *p*-value maps for the B5 treatment plots. The *p*-value is the significance of a Monte Carlo point pattern analsysis conducted in each partition of the plot. The *p*-value map for each plot is shown with the number of partitions indicated in the partitioning analysis. Lighter colours represent higher significance (clustering of points in the partition).


(a) Plot 1: original data



(b) Partitioning Results


(c) Plot 1: artificial clumping



(d) Partitioning Results

Figure 5.12: Stolon mappings for Plot 1 (a), artificially manipulated data from Plot 1 (c) and the results of the partitioning analysis for those datasets ((b) and (d), respectively).

5.4.2 Circle sampling

Appendix A contains the results of the circle sampling of plots in the various treatments. Generally, treatments B1, B2, B3 and B4 (Figures A.49, A.50, A.51, A.52, respectively) show similar results. There is clustering at small radii, but inhibition at larger radii. Treatment B5 (Figure A.53), however, shows a considerably different structure, with clustering but no inhibition.

5.4.3 Fibre point analysis

Appendix A contains the results of the fibre point analysis of plots in the various treatments. Treatment B1 resulted in similar fibre point results across plots, indicating clustering at small scales, inhibition roughly at the scale of planting (400-800 mm), and clustering at 1000 mm (Figure A.54). Figures A.55, A.56, and A.57 show the results for treatments B2, B3 and B4, respectively. Treatments B2 and B3 do not show a consistent response within the treatment. Treatment B4, does seem to exhibit consistent clustering at small scales and inhibition at medium scales. Treatment B5 (Figure A.58) shows a markedly different spatial structure. Here, there is no inhibition detected at medium scales and no clustering at large scales (as in the case of treatment B1).

5.4.4 Point pattern analysis

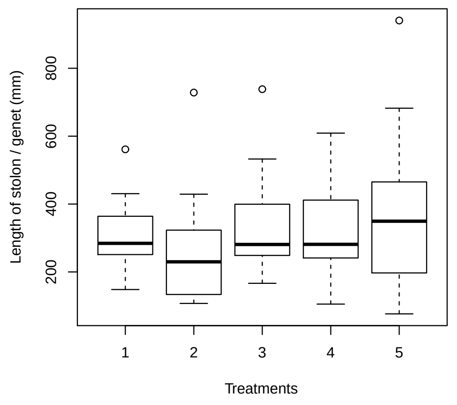
Appendix A contains the results of the point pattern analysis of daughter ramet locations in the various treatments. Generally, the point pattern analysis of treatments B1, B2, B3, B4, B5 (Figures A.59, A.60, A.61, A.62, A.63, respectively) showed similar patterns between the plots.

Daughter ramets tended to be strongly clustered at small scales (< 400 mm), but in many cases they tended toward randomness or inhibition at scales above 400 mm.

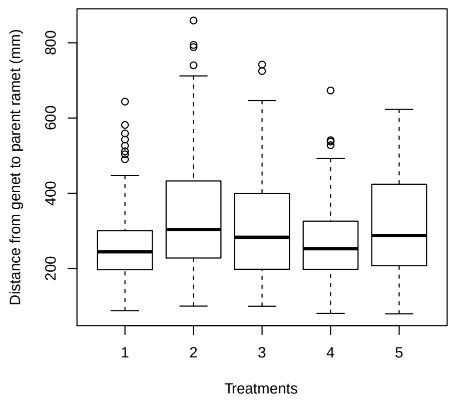
5.4.5 Null models of genet responses

Although not used in the analysis in this chapter, Monte Carlo analyses using spatially explicit null models are a promising avenue of further investigation. A variety of null models might be employed in such analyses. Such models, however, would require some form of parametrization, in which case existing data about the clonal plant is important. The *Linear* software library was used to extract information about the *F. virginiana* mapping in preparation for this work.
Figure 5.13(a) shows the average stolon length per genet for each of the five treatments. Figure 5.13(b) shows the daughter to parent ramet distance for all parent-daughter pairs in each treatment. Figure 5.13(c) shows the number of ramets per plot for each treatment. Figure 5.13(d) shows the total stolon length per plot.

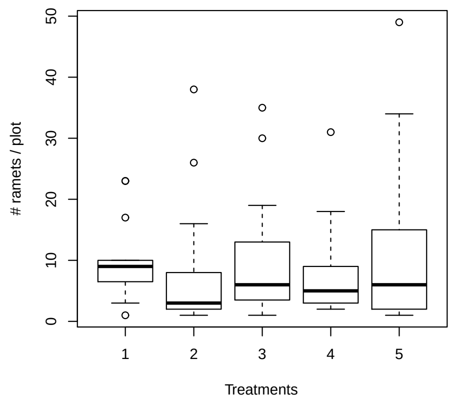
Genet models were developed using the *Linear* software library as a first step to implementing the analyses. Figure 5.14 shows observed *F. virginiana* stolon data (a), a parameterized spatially explicit model of the data (b) and a model with stolon turn angles randomized (c).



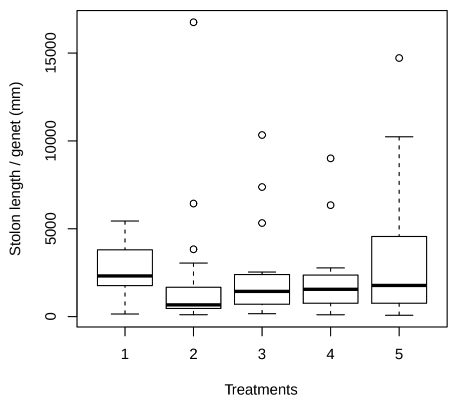
(a) Average stolon length / genet



(b) Distance between parent and daughter

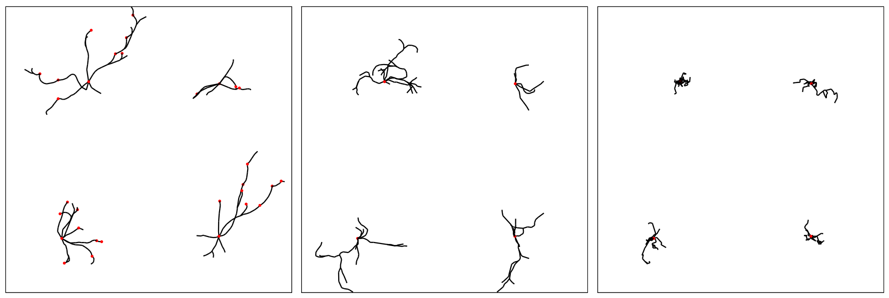


(c) Number of ramets / plot



(d) Total stolon length / genet

Figure 5.13: Boxplots showing (a) average length of individual stolons/genet, (b) average distance between parent and daughter ramets, (c) number of ramets per plot, and (d) total stolon length per genet for each treatment.



(a) Observed (b) Simulation (c) Simulation: random

Figure 5.14: Actual (a) and simulated *F. virginiana* genets (b and c). The simulation model in (b) was parameterized using the observed data in (a). The simulated model in (c) was parameterized using the observed data in (a), but turn angles were random. Model parameters are number of stolon, branching likelihood, initial stolon orientation, turn angle distribution, and distribution of segment lengths.

5.5 Discussion

With some adjustment, the partitioning analysis defined by Brix et al. (2001) and applied by Couteron et al. (2003), shows promise as a means of characterizing the heterogeneity of linear spatial pattern. Using the proportion of points for Nq, as opposed to the point count suggested by Couteron et al. (2003), was the only means by which the variance measure would behave in the fashioned described. The need to standardize the bias and variance may have been the result of having to work with a fundamentally different geometrical structure, although comparison of the results of Couteron et al. (2003) and the formula for the global variance from that paper suggests that a transformation had been performed but not explicitly reported. A potentially fruitful and necessary area of further research is the development of new, more intuitive measures of “variance” and “bias” that are more appropriate for linear features.

The larger cells, with few divisions, provide relatively many random fibre points due to their larger area, but they have a reduced ability to characterize the fibre system due to the decreased resolution. Conversely, small cells, and thus many divisions, result in the fibre points (or test lines) producing point patterns that are very representative of the fibre system, but the number of points in each cell is reduced. This reduction in sample points, although reducing the bias, resulted in an increase in the adjusted variance of the cells. The interpoint distance metric used here requires at least two points in the cell. Metrics that are based on an area, such as the convex hull of a cluster of points, would require at least three points. Realistically, these metrics are much more useful if there are many more than two or three points within the cell. An important consideration is that the number of intersection points can be controlled by varying the interval of the test lines (if test lines are being used to generate fibre points), the number of sampling circles or the number of fibre points distributed on the observed data.

Although the mean interpoint distance was used in the Monte Carlo analysis, this analysis provides ample possibility for extension and alteration. For example, different models could be used in the Monte Carlo analysis. One promising approach not applied here is the use of a random segment model in the Monte Carlo simulation instead of a random point process. This segment model would then be sampled using test lines, sampling circles, or random fibre points. The points generated by the intersections with the test lines and the data would then be compared to the points of the observed intersection or fibre points. Instead of the point intensity remaining constant (as in Couteron et al. (2003)), the intensity of the segment process system within the cell is kept constant across simulations with the position and orientation of the segments being randomized.

In addition to this, the Monte Carlo analysis could be conducted using measures that are not point- or line-based. Such measures might include a directional distribution comparison between the observed data and a random segment process. Using more complex approaches, such as measures relating to the second order measures of fibre processes, are also areas of promising investigation that await further development.

In this investigation, the number of fibre points were held constant per unit area, but alternatively they could have been held constant per unit length of observed data. There exists no recommended number or density of randomly distributed fibre points for use in characterizing linear data. Further, there exists no recommended intensity of test lines or sampling circles used to generate intersection points. Experimentation with test line intervals, circle sampling densities and fibre point densities are areas that require further investigation so that the behaviour of these systems can be better understood when such densities are altered.

The high proportion of cells having zero-value *p*-values is expected due to the considerable regions of relatively open space in the *F. virginiana* mappings. This is likely to be a typical result for rhizome mappings in general as there tends to be a considerable amount of open space relative to the actual fibres (e.g., Maddox et al., 1989; Edwards, 1984). The *p*-values for plot partitions that were non-zero, however, varied considerably, usually between 0.5 and 1.0 (random and strongly clustered, respectively), which indicates that in partitions where there was more than one fibre point, the point patterns ranged between clustering and randomness. This indicates that the heterogeneity of the mapping is being detected by the analysis. Generally, such a result means that caution should be exercised when applying standard analysis approaches across the plot as assumptions of homogeneity might be violated. If deemed substantial enough to warrant treatment, the heterogeneity can be addressed by first partitioning the plot using the local-global analysis and then conducting further analyses inside the cells. While there have been efforts to deal with anisotropic fibre processes (Benes et al., 1997, 1994), the assumption of a stationary process cannot be avoided and global-local methods such as this provide a promising method by which heterogeneous patterns can be examined with analyses requiring the stationarity assumption.

The local-global analysis was used to test the biological hypothesis that *F. virginiana* will respond to nutrient heterogeneity by shortening stolon length, increasing stolon production, and increasing ramet production in regions of elevated nutrient concentration, resulting in stolon spatial structure varying with nutrient heterogeneity spatial structure. The partitioning analysis did indicate varying partition sizes for different plots. Overall, although there was variability within treatments, there was some indication that the nutrient heterogeneity might have influenced the structure of the genets, particularly in the B3, B4 and B5 plots. Potentially, this could reflect a foraging response or some other physiological response to the increased nutrients in some regions of the plots. Many plots in the B1-B4 treatments had a suggested partition size of the minimum 4 x 4 partitions. This relatively large partition size indicates larger areas of homogeneity in the fibre point distribution and, by extension, the *F. virginiana* stolons. However, some plots within each of the treatments B1, B2, B3 and B4 had much higher levels of partitioning (up to 10 partitions). In these cases, the smaller partition size is driven by increased heterogeneity of fibre points at a smaller scale.

The larger partitioning of the B2 plots might be reflective of the larger scale nutrient heterogeneity. They were similar to the B1 plots in that the *p*-value maps indicated large regions of clustering. The smaller partitions (5 and 11 partitions) in two of the four B3 treatment plots indicates heterogeneity at smaller scales in these two plots. This could be an indication of a response to the smaller scale of nutrient heterogeneity in the B3 treatment. The B4 treatment showed high variability between plots in terms of partition size. Two plots had higher numbers of partitions, but the resultant *p*-value mappings didn’t seem to indicate a concentration of stolon clustering around the plants in the higher nutrient zone. The B5 treatment showed the most consistent results across plots. The high number of partitions indicates small scale heterogeneity, and the *p*-value maps clearly indicate the concentrated stolons. It is interesting to note that the small partitions vary considerably in terms of their *p*-values, indicating various degrees of clustering in the partitions.

Although the original planting arrangement of the B5 treatments differed slightly from the other treatments, this planting arrangement can not fully account for the consistent high level of partitioning required in the B5 plots. This is supported by the artificial manipulation of the genet data from Plot 1 (B1) into a “planting” arrangement similar to the plots in B5. Reanalysis of this artificial data showed a recommended partition size that was unchanged from the original data. If planting arrangement (in the case of treatment B5) was solely the cause for the partitioning results, the manipulation of the B1 data would have resulted in partitioning and *p*-values similar to those of B5. This was not the case, which suggests that the structure of the stolons in the B5 plots, rather than simply the aggregation of parent ramets in the planting arrangement, resulted in the observed partitioning and *p*-value maps.

It was expected that *F. virginiana* would exhibit some response to the nutrient heterogeneity. It is known that other species of the *Fragaria* genus do respond to heterogeneous environments. For example, Alpert (1991) found that nitrogen is shared between ramets of *F. chiloensis* and heterogeneous soil distribution can influence the architecture of the plant by altering the production of ramets and stolons. Alpert et al. (2003) found that, although *F. chiloensis* did show differences in performance in the heterogeneous environment, it did not show differences in division of labour as measured by root allocation as a function of ramet connection. In addition, Alpert and Mooney (1986) have found that clonal integration of *F. chiloensis* ramets increased survival under adverse conditions. It was expected that similar responses would be exhibited by *F. virginiana*. It may be that the history of the plants may have also impacted the results. Latzel and Klimešová (2010) suggest that the phenomena of transgenerational plasticity may also be applicable to clonal plants. This supports Roiloa et al. (2007) who found that *F. chiloensis* from more heterogeneous habitats have greater ability for division of labour than *F. chiloensis* from more homogeneous environments. However, other evidence suggested that ramets would respond regardless of past history. D'Hertefeldt et al. (2011) found that clonal plants responded to heterogeneity regardless of the nutrient availability of the previous environment. This included plants from resource poor sites responding to nutrient heterogeneity. Further research is required on *F. virginiana* in this area.

The potential for other clonal species to respond to heterogeneity is well known. Clonal plants have the ability to preferentially allocate plant parts (e.g., roots) to take advantage of heterogeneous soil conditions and thus increase productivity in the form of biomass (Birch and Hutchings, 1994). Foraging has been documented in *Hydrocotyle bonariensis* where ramets were located preferentially outside of unfavourable patches in a heterogeneous environment (Evans and Cain, 1995). Integrated clones of *Potentilla simplex* have been found to elongate their stolons more so than clones that were not integrated, which suggests that integration could allow for movement away from unfavourable locations (Wijesinghe and Handel, 1994). It might also suggest an improved ability for the connected ramets to explore their environment more rapidly, leading to the improved acquisition and utilization of resources.

The clonal herb *Glechoma hederacea* has been found to produce longer, unbranched stolon internodes in low-light conditions and short, frequently-branched stolon internodes in high-light conditions (Slade and Hutchings, 1987b). Such a response would allow for utilization of resources in high-resource zones and more rapid spread through zones of low-resource availability. *G. hederacea* has also been found to exhibit a foraging response to nutrient conditions by forming short, highly-branched internodes in high nutrient areas and long, lowly-branched internodes in low nutrient areas (Slade and Hutchings, 1987c).

However, not all foraging would take place by the alteration of stolon structure. In their review, de Kroon and Hutchings (1995) suggest that morphological plasticity of roots and shoots (i.e., morphology at the level of the ramet) provide mechanisms for foraging, while generally non-plastic length of stolon and rhizome spacers provide mechanisms for a search through the habitat for resources. In addition, precision of root foraging for soil nutrients has been found to vary between species and also vary within species in response to the spatial arrangement of the nutrients (Wijesinghe et al., 2001). *G. hederacea* has also been found to respond to heterogeneity of nutrient patches by increased root allocation in higher nutrient patches (Wijesinghe and Hutchings, 1999). In such cases, a change in spatial structure of stolons would not be expected in heterogeneous conditions, even when foraging is taking place. In addition, the scale of heterogeneity could be important in the case of root foraging. *G. hereracea* showed root foraging in course-scale heterogeneity, but it responded similarly to fine-grained heterogeneity as to homogeneous, poor soil resource distribution (Wijesinghe and Hutchings, 1997).

Even without preferential placement of plant parts (ramets, shoots, stolons, rhizomes, roots or leaves), physical connections between ramets can allow physiological integration, allowing for resources to be transported from sources to sinks throughout the genet. *Potentilla simplex* has been found to benefit from clonal integration in heterogeneous environments, producing more biomass when integrated than when separated (Wijesinghe and Handel, 1994). Slade and Hutchings (1987a) have found that, in the case of heterogeneous environments, developing ramets in resource-poor locations received more resources from ramets in resource-rich sites, which enabled for growth similar to that of ramets in resource rich sites. They also found that when ramets are integrated in heterogeneous conditions, with ramets in both unfavourable and favourable conditions, ramets in resource-poor locations were able to forage for resources more intensely than if they were in uniform, poor-resource conditions (Slade and Hutchings, 1987a). This behaviour can even vary within species. For example, Alpert (1999a) found that ramets of *F. chiloensis* taken from homogeneous environments shared fewer resources than ramets taken from heterogeneous environments.

Modelling of the foraging behaviour of clonal plants also seems to suggest that the phenomena of clonal foraging is far from simple. Cain (1994) modelled clonal growth in an effort to examine foraging behaviour. He found that, even though ramets may have shorter rhizomes in favourable areas, internode length, branching angles, and pattern of branching strongly influence the foraging response. Models have also suggested that foraging responses are likely influenced by environmental conditions, such as resource patch size, number and arrangement (Cain et al., 1996).

Although some species of clonal plants have been found to exhibit clonal traits that favour environmental heterogeneity, many species require further study (Price and Marshall, 1999). As illustrated by the results of this experiment, examining responses to heterogeneity can be challenging, and the research can be further complicated by the questions driving the study and by the species. For example, *F. virginiana* produces relatively long stolons, so experiments examining heterogeneity require relatively large plot sizes in order for genets to develop. In addition, longer time frames allowing for more development of ramets and increased stolon production would be beneficial. However, the requirement of both long time frames and large plot sizes pose significant challenges to the researcher.

Simulation models, however, offer an avenue to overcome these challenges. The case has been made for complex simulation models to serve as experimental systems that can be manipulated in ways that existing systems cannot (Peck, 2004), and Bolker et al. (2003) discuss a range of models of the spatial dynamics of plant communities. Austin (2002) discusses the importance of integrating ecological theory into modelling approaches, and vice versa, when making spatial predictions of species distributions. This chapter has introduced simulation models for individual *F. virginiana* genets, and an important area of future research is the development of multi-genet simulations using these models as a foundation.

Various modelling approaches have been used to investigate clonal growth and foraging behaviour for some time, which is likely in part due to the challenges described above. The clonal growth of *Solidago altissima* has been modelled using deterministic, stochastic simulation and random-walk models that used “clonal growth parameters” (branching angles, rhizome lengths, rhizome initiation points, and numbers of daughter rhizomes) as model parameters (Cain et al., 1991). Cain (1994) modelled clonal growth in an effort to examine foraging behaviour, and found that, even though ramets may have shorter rhizomes in favourable areas, internode length, branching angles, and pattern of branching strongly influence the foraging response. Models have also suggested that foraging responses are likely influenced by environmental conditions, such as resource patch size, number and arrangement (Cain et al., 1996).

Cain (1990) used random-walk models to examine the spread of *S. altissima* and found the models to be good predictors of clonal spread, and the clonal growth of *Trifolium repens* has been modelled using neighbourhood models (Cain et al., 1995). In a non-spatial model, de Kroon and Shieving (1991) modelled the allocation of resources to rhizomes or stolons in clonal plants and predicted that allocation should be low when resources are very high or very low and that allocation should be higher when resources are at a medium level.

Models can also examine the physiological aspects of integration. For example, Ushimaru and Genkai-Kato (2011) modelled different levels of resource translocation. This work could be integrated into spatially explicit models of *F. virginiana*. Herben (2004) uses this approach using a spatially explicit simulation model of clonal growth, competition and translocation. He found translocation provides a competitive advantage provided the costs of that translocation are not too high.

Parameterization of clonal growth parameters, however, might prove challenging. Cook (1988) mapped the rhizomes of *Medeola virginiana* and found that the parameters of clonal growth to be highly variable, suggesting that deterministic models of clonal growth are not appropriate for predicting the spatial properties of clonal plants. This conclusion is supported by the research here, which found considerable variability in the spatial properties of *F. virginiana*. The detailed mappings of the

F. virginiana collected in this experiment, as well as in the field and experimental plots used for the investigations in previous chapters, however, provide for rich datasets that can be used to parameterize various growth models. The variability in some of these data, however, suggest that stochastic models may be more appropriate. This is an area of future research focus.

It is likely that both sexual and asexual reproduction play an important role in *F. virginiana* population maintenance. *F. virginiana* does utilize both methods of propagation, but more research is required to investigate fully the roles of sexual and asexual reproduction in heterogeneous environments. The results of this research could be used to inform these advanced modelling efforts. In other species, genetic analysis has suggested that sexual reproduction is important for establishment of clonal populations while clonal growth is likely important for the maintenance and expansion of the population after establishment (Dong et al., 2006).

5.6 Conclusions

The local-global approach is a promising method of examining spatial data that is heterogeneous in nature. It promises a potentially powerful framework on which analyses for linear data can be developed. Further modification of the global-local analysis will improve the method by taking into account concerns that are specific to linear features (i.e., the development of new measures of “bias” and “variance” to capture the effect of reduction in cell size). This research also suggests several areas of promising new work, notably, the use of a wide range of null models, some newly developed and others based on published models of vegetative spread, to investigate spatial patterns and the investigation of the relationship between sampling line intensity, behaviour of fibre process analysis and ecologically relevant parameters measured on a study organism. *F. virginiana* may have showed some response to the nutrient heterogeneity within experimental plots, but further research is required in this area to understand better the level and type of responses *F. virginiana* may have to nutrient heterogeneity and how this affects spatial structure. In general, the data exploration methods developed in this chapter were successful in characterizing the *F. virginiana* spatial structure and detecting potential responses to nutrient heterogeneity.

Chapter 6

Summary

6.1 Synopsis and further directions

This work has started to address the analysis of linear spatial data in ecology, partly through the application of stochastic geometry. It has introduced stochastic geometry theory, including fibre processes, and applied this theory to clonal plant data in both field and experimental settings. It has also made connections to the theory and application of point processes which have been used extensively in plant ecology. In addition, it has suggested avenues of future research that would expand this initial exploration.

Chapter 2 introduced a new method of examining the second order structure of linear data. A resampling analysis using circular test lines and fibre-based sampling with circular test lines were combined to form a Monte Carlo simulation to assess inhibition or repulsion of stolons. The analysis of the segment processes clearly shows that both the point pattern analysis of fibre points and the resampling analysis are effective methods of characterizing the scale of clustering of linear data. The resampling analysis was used to examine the stolon arrangements of field and forest populations of *F. virginiana*. The analysis of *F. virginiana* stolon data shows that, regardless of plot, the intensity of stolons with respect to surrounding stolons is not homogeneous in space. The plots, however, differed from each other in terms of at what radii there was clustering or inhibition.

Chapter 3 introduced a rotational analysis and applied it to a range of linear data, including stolon mappings of *F. virginiana* along a forest-oldfield transition. The analysis of artificial data (segment processes) demonstrates that both the summary of the angular distribution using radial plots and the rotational analysis can provide effective summaries of the directional properties of a fibre system. The rotational analysis presented in this chapter provides a means of examining the angular properties of data in raster format that requires fewer assumptions than the line width method and the tracing method. The rotational analysis provides a means to estimate preferential direction of raster data with a relatively simple image transformation and straight test lines. Analysis of the stolon mappings showed that ramet density decreased in plots near the old field. It also showed that forest plots had stolons with little directional preference, while plots closer to the field had a preferred orientation perpendicular to the forest-field edge. In the case of the *F. virginiana* field data, both the rotational analysis and the plots of directional distribution seem to suggest different ramet behaviour in terms of stolon direction along the forest-field transition zone. The spatial arrangement of stolons farther from the forest edge may be driven by a foraging response to light heterogeneity at small scales. Closer to the forest-field edge, however, *F. virginiana* may be responding to the source of light from the open field.

Chapter 4 presented a genet randomization analysis and an analysis based on the analysis of random fibre points. The analysis of artificial data shows that the genet randomization procedure is capable of characterizing the spatial structure of genets in terms of clustering, inhibition (regularity) and randomness. The analysis relies on the fundamental concept of the well-known and widely-applied point pattern analysis with a random Poisson point process as a model of CSR. Both of these analyses were effective in describing the structure of stolon mappings from a *F. virginiana* transplanting experiment. Analysis of the *F. virginiana* plot data suggests that the initial parent ramet configuration does not seem to influence the spatial structure of either daughter ramets or stolons. The constrained randomization shows that stolons were arranged randomly regardless of the spatial configuration of parents, suggesting that clonal spread of the transplanted *F. virginiana* is not influenced by the initial arrangement of parent ramets. The transplant experiment suggested that *F. virginiana* adapts its foraging behaviour to its environment, reacting to a homogeneous resource distribution by random clonal spread and random establishment of daughter ramets in the short term. Longer term studies would be required to determine if there is a change in stolon spatial structure formation and placement of ramets over time.

Chapter 5 uses some of the previously developed tools to examine *F. virginiana* data from a nutrient heterogeneity transplant experiment. It also introduced the application of a local-global partitioning analysis that uses a Monte Carlo approach to divide a region into quadrats that have a more homogeneous spatial process. With some adjustment, the partitioning analysis defined by Brix et al. (2001) and applied by Couteron et al. (2003) , shows promise as a means of characterizing the heterogeneity of linear spatial pattern. This is a promising means of partitioning heterogeneous processes, and the methodology lends itself to extension in a number of ways. In particular, the alteration of the underlying null model in the Monte Carlo analysis is a promising means of applying the analysis to a range of questions.

Due to the challenges involved in the heterogeneity experiment, a second area of future focus is the modelling of *F. virginiana* growth using the framework constructed through this research (and demonstrated in Chapter 5). Further modification of the global-local analysis will improve the method by taking into account concerns that are specific to linear features (i.e., the development of new measures of “bias” and “variance” to capture the effect of reduction in cell size). This research also suggests several areas of promising new work, notably, the use of a wide range of null models, some newly developed and others based on published models of vegetative spread, to investigate spatial patterns and the investigation of the relationship between sampling line intensity, behaviour of fibre process analysis and ecologically relevant parameters measured on a study organism. *F. virginiana* may have showed some response to the nutrient heterogeneity within experimental plots, but further research is required in this area to develop a better understanding of the level and type of responses *F. virginiana* may have to nutrient heterogeneity. In general, the methods developed in this chapter were successful in characterizing the *F. virginiana* spatial structure.

6.2 Software library for the analysis of linear data

The analyses in this work were conducted mostly in the Python programming language and required the development of an extensive software library (*Linear*) designed specifically to examine linear data and clonal plants. This library is a collection of Python classes and methods written explicitly to support this work. The library provides basic spatial elements (e.g., circle, segment, point, fibre), higher level spatial elements (e.g., point, segment, and fibre processes), simple measures (e.g., length of fibre, orientation of segment), methods and high level analyses (e.g., circle sampling analysis, randomization of genets). The library uses Python classes (objects) to implement spatial elements and analyses, and it uses methods (functions) to perform specific operations on the objects (data and spatial structures). The current primary use of *Linear* is as a library with which researchers can construct advanced analyses using the Python programming language.

6.3 Unresolved problems

This work explores several theoretical and practical approaches to examining linear data in an ecological context, primarily in relation to clonal plants. However, the exploration was not exhaustive, and there remain unresolved issues. The response of *F. virginiana* to heterogeneous environments requires further exploration, possibly with the use of modelling simulations to overcome the difficulty of experimentation.

The local-global analysis provides a rich framework for further development of the analysis. The various components of the analysis (Monte Carlo null model, method of generating fibre points, etc.) should be more fully explored using simulation models. This would allow a better understanding of the performance of the analysis when using different components in its structure and given data with various fundamental structural properties.

This work introduced basic genet simulation models. These models require application toward further investigation of *F. virginiana*. The development and analysis of simulation models will be useful to develop further new analyses by providing a source of artificial data for testing analysis behaviour. They may also prove useful in the development of null models for Monte Carlo analysis. The use of these models as null models, however, poses a challenge that remains not fully resolved, the issue of developing a null model having many parameters. The usefulness of complex null models remains to be explored, and their use will have to be accompanied by clear assumptions and detailed information on their parameters so that inferences based on these models can be properly interpreted.

6.4 Potential applications

Although the chapters of this work demonstrate potential practical applications for the theory and methods relating to the analysis of clonal plants, there is ample opportunity for extending the application of these approaches. These methods are applicable to examining a wide range of vegetation structures (e.g., roots, shoots, leaf transport systems, rhizomes and stolons) because of the tendency of these structures to manifest themselves as lines. However, these methods can be applied much more broadly in ecology (e.g., animal movement paths, vegetation boundaries, trails, and burrow systems). They can also be applied to any system that produces spatial linear structure (e.g., transportation networks, faults, river channels, or ice fractures).

6.5 Integration of spatial analysis in ecological thinking

This work focused largely on the technical and practical approaches of spatial analysis of linear data in ecology. It was a series of steps made toward improving the ability to understand the spatial structure of plants, in particular, clonal plants. In the case of plant ecology, spatial structure is a critical aspect of understanding the dynamics of plant populations and plant communities. This becomes even more important in the case of clonal plants because of their ability to reproduce asexually, often through the extension of stems (stolons and rhizomes). There is little doubt that in order to understand many of the processes in plant systems, aspects of the spatial pattern must be examined and characterized.

This reasoning can be extended to ecology in general. Many processes in ecology operate over finite distances and thus have a spatial component that might be important in their functioning. While the spatial context is not necessarily always examined in ecological field studies due to the complexity of collecting and analysing such data, the argument can be made that the spatial aspect of any system is a fundamental aspect of how processes unfold. As investigations of any ecological system become more refined and as understanding improves, it is likely that the spatial aspect of the system will be become of greater and greater importance. It is important that new methods of spatial data collection and spatial data analysis be developed continually in order to aid the ecologist in these investigations. In particular, the practical challenges of spatial data analysis (e.g., type of data, applicability to field data, software development) need to be included in this ongoing development so that ecologists can include spatial analysis into their research programs.

References

Agrawal, A. A., Ackerly, D. D., Adler, F., Arnold, A. E., Caceres, C., Doak, D. F., Post, E., Hudson, J. P., Maron, J., Mooney, K. A., Power, M., Schemske, D., Stachowicz, J., Strauss, S., Turner, M. G., Werner, E., 2007. Filling key gaps in population and community ecology. Frontiers in Ecology and the Environment 5 (3), 145–152.

Alftine, K. J., Malanson, G. P., 2004. Directional positive feedback and pattern at an alpine tree line. Journal of Vegetation Science 15, 3–12.

Almeida-Neto, M., Lewinsohn, T., 2004. Small-scale spatial autocorrelation and the interpretation of relationships between phenological parameters. Journal of Vegetation Science 15, 561–568.

Alpert, P., 1991. Nitrogen sharing among ramets increases clonal growth in *Fragaria chiloensis*. Ecology 72 (1), 69–81.

Alpert, P., 1996. Nutrient sharing in natural clonal fragments of *Fagaria chiloensis*. Journal of Ecology 84, 395–406.

Alpert, P., 1999a. Clonal integration of *Fragaria chiloensis* differs between populations: ramets from grassland are selfish. Oecologia 120, 69–76.

Alpert, P., 1999b. Effects of clonal integration on plant plasticity in *Fragaria chiloensis*. Plant Ecology 141, 99–106.

Alpert, P., Holzapfel, C., Benson, J., 2002. Hormonal modification of resource sharing in the clonal plant *Fragaria chiloensis*. Functional Ecology 16, 191–197.

Alpert, P., Holzapfel, C., Slominski, C., 2003. Differences in performance between genotypes of *Fragaria chiloensis* with different degrees of resource sharing. Journal of Ecology 91, 27–35.

Alpert, P., Mooney, H., 1986. Resource sharing among ramets in the clonal herb, *Fragaria chiloensis*. Oecologia 70, 227–233.

Angevine, M., 1983. Variations in the demography of natural populations of wild strawberries *Fragaria vesca* and *F. virginiana*. Journal of Ecology 71, 959–974.

Angevine, M., Handel, S., 1986. Invasion of forest floor space, clonal architecture, and population growth in the perrenial herb *Clintonia borealis*. Journal of Ecology 74, 547–560.

Angevine, M. W., 1981. Demographic variation in populations of the wild strawberries *Fragaria vesca* and *F. virginiana*. Ph.D. thesis, Cornell University.

Arbia, G., 1990. On second-order non-stationarity in two dimensional lattice processes. Computational Statistics and Data Analysis 9, 147–160.

Austin, M., 2002. Spatial prediction of species distribution: an interface between ecological theory and statistical modelling. Ecological Modelling 157, 101–118.

Baddeley, A., Turner, R., 2005. Spatstat: an R package for analyzing spatial point patterns. Journal of Statistical Software 12 (6), 1–42, URL: `www.jstatsoft.org`, ISSN: 1548-7660.

Beever, E. A., Swihart, R. K., Bestelmeyer, B. T., 2006. Linking the concept of scale to studies of biological diversity: evolving approaches and tools. Diversity & Distributions 12 (3), 229–235.

Bell, A., Tomlinson, P., 1980. Adaptive architecture in rhizomatous plants. Botanical Journal of the Linnean Society 80, 125–160.

Benes, V., Chadœuf, J., Kretzschmar, A., 1997. Properties of length estimation in spatial fibre models. Environmetrics 8, 397–407.

Benes, V., Chadœuf, J., Ohser, J., 1994. On some characteristics of anisotropic fiber processes. Mathematische Nachrichten 169, 5–17.

Benhamou, S., 2004. How to reliably estimate the tortuosity of an animal’s path: straightness, sinuosity, or fractal dimension? Journal of Theoretical Biology 229, 209–220.

Benhamou, S., 2006. Detecting an orientation component in animal paths when the preferred direction is individual-dependent. Ecology 87 (2), 518–528.

Berman, M., 1986. Testing for spatial association between a point process and another stochastic process. Applied Statistics 35, 54–62.

Birch, C., Hutchings, M., 1994. Exploitation of patchily disturbed soil resources by the clonal herb *Glechoma hederacea*. Journal of Ecology 82, 653–664.

Bishop, E. J., Spigler, R., Ashman, T.-L., 2010. Sex-allocation plasticity in hermaphrodites of sexually dimorphic *Fragaria virginiana* (Rosaceae). Botany 88, 231–240.

Boddy, L., 1993. Saprotrophic cord-forming fungi: Warfare strategies and other ecological aspects. Mycology Research 97 (6), 641–655.

Bolker, B., Pacala, S., Neuhauser, C., 2003. Spatial dynamics in model plant communities: What do we really know? The American Naturalist 162 (2), 135–148.

Boots, B., 2002. Local measures of spatial autocorrelation. Ecoscience 9 (2), 168–176.

Brisson, J., Reynolds, J. F., 1994. The effect of neighbours on root distribution in a creosotebush (*Larrea tridentata*) population. Ecology 75, 1693–1703.

Brix, A., Senoussi, R., Couteron, P., Chadœuf, J., 2001. Assessing goodness of fit of spatially inhomogenous Poisson processes. Biometrika 88 (2), 487–497.

Cain, M., 1990. Models of clonal growth in *Solidago altissima*. Journal of Ecology 78, 27–46.

Cain, M., 1994. Consequences of foraging in clonal plant species. Ecology 75 (4), 933–944.

Cain, M., Damman, H., 1997. Clonal growth and ramet performance in the woodland herb, *Asarum canadense*. Journal of Ecology 85, 883–897.

Cain, M., Dudle, D., Evans, J., 1996. Spatial models of foraging in clonal plant species. American Journal of Botany 83 (1), 76–85.

Cain, M., Pacala, S., Silander, J., 1991. Stochastic simulation of clonal growth in the tall goldenrod, *Solidago altissima*. Oecologia 88 (4), 477–485.

Cain, M., Pacala, S., Silander, J., Fortin, M., 1995. Neighborhood models of clonal growth in the white clover *Trifolium repens*. The American Naturalist 145 (6), 888–917.

Chadœuf, J., Brix, A., Pierret, A., Allard, D., 2000. Testing local dependence of spatial structures on images. Journal of Microscopy 200, 32–41.

Cook, R., 1988. Growth in *Medeola virginiana* clones: I. field observations. American Journal of Botany 75, 725–731.

Couteron, P., Kokou, K., 1997. Woody vegetation spatial patterns in a semi-arid savanna of Burkina Faso, West Africa. Plant Ecology 132, 211–227.

Couteron, P., Seghiere, J., Chadœuf, J., 2003. A test for spatial relationships between neighbouring plants in plots of heterogenous plant density. Journal of Vegetation Science 14, 163–172.

Cressie, N., 1991. Statistics for Spatial Data. John Wiley & Sons.

Crites, S., Dale, M., 1998. Diversity and abundance of bryophytes, lichens, and fungi in relation to woody substrate and successional stage in aspen mixedwood boreal forests. Canadian Journal of Botany.

Cruz-Orive, L., Hoppeler, H., Mathieu, O., Weibel, E., 1985. Stereological analysis of anisotropic structures using directional statistics. Applied Statistics 34 (1), 14–32.

Dale, M., 1999. Spatial Pattern Analysis in Plant Ecology. Cambridge University Press.

Dale, M., 2000. Lacunarity analysis of spatial pattern: a comparison. Landscape Ecology 15, 467–478.

Dale, M., John, E., Blundon, D., 1991. Contact sampling for the detection of interspecific association: A comparison in two vegetation types. Journal of Ecology 79 (3), 781–792.

Dale, M., MacIsaac, D., 1989. New methods for the analysis of spatial pattern in vegetation. Journal of Ecology 77 (1), 78–91.

Dale, M., Powell, R., 2001. A new method for characterizing point patterns in plant ecology. Journal of Vegetation Science 12, 597–608.

Dale, M. R. T., Dixon, P., Fortin, M. J., Legendre, P., Myers, D. E., Rosenberg, M. S., 2002. Conceptual and mathematical relationships among methods for spatial analysis. ECOGRAPHY 25 (5), 558–577.

Dale, M. R. T., Fortin, M. J., 2002. Spatial autocorrelation and statistical tests in ecology. Ecoscience 9 (2), 162–167.

Day, K., Hutchings, M., John, E., 2003a. The effects of spatial pattern on nutrient supply on the early stages of growth in plant populations. Journal of Ecology 91, 305–315.

Day, K., John, E., Hutchings, M., 2003b. The effects of spatially heterogeneous nutrient supply on yield, intensity of competition and root placement patterns in *Briza media* and *Festuca ovina*. Functional Ecology 17, 454–463.

de Kroon, H., Hutchings, M., 1995. Morphological plasticity in clonal plants: The foraging concept reconsidered. Journal of Ecology 83, 143–152.

de Kroon, H., Shieving, F., 1991. Resource allocation patterns as a function of clonal morphology: A general model applied to a foraging clonal plant. Journal of Ecology 79, 519–530.

D'Hertefeldt, T., Falkengren-Grerup, U., Jónsdóttir, I., 2011. Responses to mineral nutrient availability and heterogeneity in physiologically integrated sedges from contrasting habitats. Plant Biology 13, 483–492.

D'Hertfeldt, T., Falkengren-Grerup, U., 2002. Extensive physiological integration in *Carex arenaria* and *Carex disticha* in relation to potassium and water availability. New Phytologist 156, 469–477.

Diggle, P., 1977. The detection of random heterogeneity in plant populations. Biometrics 33 (2), 390–394.

Dong, M., 1993. Morphological plasticity of the clonal herb *Lamiastrum galeobdolon* (l.) Ehrend. & Polatschek in response to partial shading. New Phytologist 124, 291–300.

Dong, M., Lu, B. R., Zhang, H. B., Chen, J. K., Li, B., Apr 2006. Role of sexual reproduction in the spread of an invasive clonal plant *Solidago canadensis* revealed using intersimple sequence repeat marker. Plant Species Biology 21 (1), 13–18.

Donnelly, D., Wilkins, M., Boddy, L., 1995. An integrated image analysis approach for determining biomass, radial extent and box-count fractal dimension of macroscopic mycelial systems. Binary 7, 19–28.

Druckenbrod, D. L., Shugart, H. H., Davies, I., 2005. Spatial pattern and process in forest stands within the Virginia piedmont. Journal of Vegetation Science 16, 37–48.

Edgar, G. J., Robertson, A. I., 1992. The influence of seagrass structure on the distribution and abundance of mobile epifauna: pattern and process in a Western Australia *Amphibolis* bed. Journal of Experimental Marine Biology and Ecology 160, 13–31.

Edwards, J., 1984. Spatial pattern and clone structure of the perrenial herb, *Aralia nudicaulis* L. (Araliaceae). Bulletin of the Torrey Botanical Club 111 (1), 28–33.

Eilts, J. A., Mittelbach, G. G., Reynolds, H. L., Gross, K. L., 2011. Resource heterogeneity, soil fertility, and species diversity: Effects of clonal species on plant communities. American Society of Naturalists 177, 574–588.

Evans, J., Cain, M., 1995. A spatially explicit test of foraging behaviour in a clonal plant. Ecology 76 (4), 1147–1155.

Fortin, M. J., Dale, M. R. T., 2002. Spatial analysis in ecology. Vol. 4 of Encyclopedia of Environmetrics. John Wiley & Sons, Ltd., Chichester, pp. 2051–2058.

Fortin, M. J., Payette, S., 2002. How to test the significance of the relation between spatially autocorrelated data at the landscape scale: A case study using fire and forest maps. Ecoscience 9 (2), 213–218.

Foxall, R., Baddeley, A., 2002. Nonparametric measures of association between a spatial point process and a random set, with geological applications. Applied 51, 165–175.

Fridley, J. D., Stachowicz, J. J., Naeem, S., Sax, D. F., Seabloom, E. W., Smith, M. D., Stohlgren, T. J., Tilman, D., Holle, V., B., 2007. The invasion paradox: Reconciling pattern and process in species invasions. Ecology 88 (1), 3–17.

Goodall, D., West, N., 1979. A comparison of techniques for assessing dispersion patterns. Vegetatio 40 (1), 15–27.

Gosz, J. R., 1993. Ecotone hierarchies. Ecological Applications 3 (3), 369–376.

Hall, L., Krausman, P., Morrison, M., 1997. The habitat concept and a plea for standard terminology. Wildlife Society Bulletin 25, 173–182.

Halley, J., Hartley, S., Kallimanis, A., Kunin, W., Lennon, J., Sgardelis, S., 2004. Uses and abuses of fractal methodology in ecology. Ecology Letters 7 (3), 254–271.

Hancock, J., Luby, J., 1993. Genetic resources at our doorstep: the wild strawberries. Bioscience 43, 141–147.

Hancock, J., Serce, S., Portman, C., Callow, P., Luby, J., 2004. Taxonomic variation among North and South American subspecies of *Fragaria virginiana* Miller and *Fragaria chiloensis* (L.) Miller. Canadian Journal of Botany 82, 1632–1644.

Hanisch, K., König, D., Stoyan, D., 1985. The pair correlation function for point and fibre systems and its stereological determination by planar sections. Journal of Microscopy 140, 361–370.

Harrison, R., Luby, J. J., Furnier, G., Hancock, J., 1997. Morphological and molecular variation among populations of octoploid *Fragaria virginiana* and *F. chiloensis* (rosaceae) from north america. American Journal of Botany 84 (5), 612–620.

Hartnett, D., Bazzaz, F., 1983. Physiological integration among intraclonal ramets in *Solidago canadensis*. Ecology 64, 779–788.

Hartnett, D., Bazzaz, F., 1985. The regulation of leaf, ramet and genet densities in experimental populations of the rhizomatous perennial *Solidago canadensis*. Journal of Ecology 73, 429–443.

Herben, T., 2004. Physiological integration affects growth form and competitive ability in clonal plants. Evolutionary Ecology 18, 493–520.

Holler, L., Abrahamson, W., 1977. Seed and vegetative reproduction in relation to density in *Fragaria virginiana* (Rosaceae). American Journal of Botany 64 (8), 1003–1007.

Holzapfel, C., Alpert, P., 2003. Root cooperation in a clonal plant: Connected strawberries segregate roots. Oecologia 134, 72–77.

Hossaert-McKey, M., Jarry, M., 1992. Spatial and temporal patterns of investment in growth and sexual reproduction in two stoloniferous species, *Lathyrus latifolius* and *L. sylvestris*. Journal of Ecology 80, 555–565.

Huber, H., Stuefer, J., 1997. Shade-induced changes in the branching pattern of a stoloniferous herb: functional response or allometric effect? Oecologia 110, 478–486.

Humphrey, L., Pyke, D., 2001. Ramet spacing of *Elymus lanceolatus* (thickspike wheatgrass) in response to neighbour density. Canadian Journal of Botany 79, 1122–1126.

Hutchings, M., de Kroon, H., 1994. Foraging in plants: The role of morphological plasticity in resource aquisition. Advances in Ecological Research 25, 159–238.

Hutchings, M., John, E., 2004. The effects of environmental heterogeneity on root growth and root/shoot partioning. Annals of Botany 94, 1–8.

Hutchings, M., Price, E., 1993. Does physiological integration enable clonal herbs to integrate the effects of environmental heterogeneity. Plant Species Biology 8, 95–105.

Hutto, R., 1985. Habitat selection in birds. Academic Press, Ch. Habitat selection by non-breeding, migratory land birds., pp. 455–476.

Irlandi, E. A., Peterson, C. H., 1991. Modification of animal habitat by large plants: mechanisms by which seagrasses influence clam growth. Oecologia 87, 307–318.

Jeltsch, F., Moloney, K., Milton, S., 1999. Detecting process from snapshot pattern: lessons from tree spacing in the southern Kalahari. OIKOS 85, 451–466.

Karkkainen, S., Jensen, E., Jeulin, D., 2002. On the orientational analysis of planar fibre systems from digital images. Journal of Microscopy 207, 69–77.

Karkkainen, S., Penttinen, A., Ushakov, N., Ushakova, A., 2001. Estimation of orientation characteristic of fibrous material. Advances in Applied Probability 33, 559–575.

Kembel, S., Cahill, J., 2005. Plant phenotypic plasticity belowground: A phylogenetic perspective on root foraging trade-offs. American Naturalist 166 (2), 216–230.

Kenkel, N., 1993. Modelling Markovian dependence in populations of *Aralia nudicaulis*. Ecology 74, 1700–1706.

Kiderlen, M., 2001. Non-parametric estimation of the directional distribution of stationary line and fibre processes. Advances in Applied Probability 33, 6–24.

Kleijn, D., Van Groenendael, J., 1999. The exploitationl of heterogeneity by a clonal plant in habitats with contrasting productivity levels. Journal of Ecology 87, 873–884.

Krasnoperov, R. A., Stoyan, D., 2004. Second-order stereology of spatial fibre systems. Journal of Microscopy 216, 156–164.

Latzel, V., Klimešová, J., 2010. Transgenerational plasticity in clonal plants. Evolutionary Ecology 24, 1537–1543.

Legendre, P., 1993. Spatial autocorrelation: Trouble or new paradigm? Ecology 74 (6), 1659–1673.

Legendre, P., Dale, M. R. T., Fortin, M. J., Casgrain, P., Gurevitch, J., 2004. Effects of spatial structures on the results of field experiments. Ecology 85 (12), 3202–3214.

Legendre, P., Fortin, M. J., 1989. Spatial pattern and ecological analysis. Vegetatio 80, 107–138.

Levin, S., 1992. The problem of pattern and scale in ecology: The Robert H. MacArthur Award Lecture. Ecology 73, 1943–1967.

Levin, S. A., 2000. Multiple scales and the maintenance of biodiversity. Ecosystems 3, 498–506.

Li, W., Wang, J., 2011. Influcence of light and nitrate assimilation on the growth strategy in clonal weed *Eichhornia crassipes*. Aquatic Ecology 45, 1–9.

Macdonald, S. E., Lieffers, V. J., 1993. Rhizome plasticity and clonal foraging of *Calamagrostis canadensis* in response to habitat heterogeneity. Journal of Ecology 81, 769–776.

Macek, P., Lepš, J., 2003. The effect of environmental heterogeneity on clonal behaviour of *Prunella vulgaris* l. Plant Ecology 168, 31–43.

Maddox, G., Cook, R., Wimberger, P., Gardescu, S., 1989. Clone structure in four *Solidago altissima* (Asteraceae) populations: Rhizome connections within genotypes. American Journal of Botany 76 (2), 318–326.

Manly, B. F. J., 1997. Randomization, Bootstrap and Monte Carlo Methods in Biology. Chapman & Hall/CRC, U.S.A.

Mao, S.-Y., Jiang, C.-D., Zhang, W.-H., Shi, L., Zhang, J.-Z., Chow, W., Yang, J.-C., 2009. Water translocation between ramets of strawberry during soil drying and its effects on photosynthetic performance. Physiologia Plantarum 137, 225–234.

Mecke, J., 1981. Formulas for stationary planar fibre processes III - Intersections with fibre systems. Mathematische Operationsforschung und Statistik Series Statistics 12 (2), 201–210.

Mecke, J., Stoyan, D., 1980. Formulas for stationary planar fibre processes I - General theory. Mathematische Operationsforschung und Statistik Series Statistics 11 (2), 267–279.

Meiners, S. J., Pickett, S. T. A., 1999. Changes in community and population responses across a forest-field gradient. Ecography 22, 261–267.

Molchanov, I., 1995. Statistics of the boolean model: From the estimation of means to the estimation of distributions. Advances in Applied Probability 27, 63–86.

Molchanov, I., Stoyan, D., 1994. Directional analysis of fibre processes related to boolean models. Metrika 41, 183–199.

Molchanov, I., Stoyan, D., Fyodorov, K., 1993. Directional analysis of planar fibre networks: Application to cardboard microstructure. Journal of Microscopy 172, 257–261.

Mugglestone, M., 1996. The role of tessellation methods in the analysis of three-dimensional spatial point patterns. American Statistical Association and the International Biometric Society Journal of Agricultural, Biological, and Environmental Statistics 1 (2), 141–153.

Oborny, B., 1994. Spacer length in clonal plant and the efficiency of resource capture in heterogeneous environment: A Monte Carlo simulation. Folia Geobot. Phytotax. 29, 139–158.

Odion, D. C., Davis, F. W., 2000. Fire, soil heating, and the formation of vegetation patterns in chaparral. Ecological Monographs 70 (1), 149–169.

Okland, R. H., Bakkestuen, V., 2004. Fine-scale spatial patterns in populations of the clonal moss *Hylocomium splendens* partly reflect structuring processes in the boreal forest floor. Oikos 106, 565–575.

Peck, S. L., 2004. Simulation as experiment: a philosophical reassessment for biological modeling. TRENDS in Ecology and Evolution 19 (10), 530–534.

Pélissier, R., Goreaud, F., 2001. A practical approach to the study of spatial structure in simple cases of heterogeneous vegetation. Journal of Vegetation Science 12, 99–108.

Penttinen, A., Stoyan, D., 1989. Statistical-analysis for a class of line segment processes. Scandinavian Journal of Statistics 16 (2), 153–168.

Perry, G. L. W., Miller, B. P., Enright, N. J., 2006. A comparison of methods for the statistical analysis of spatial point patterns in plant ecology. Plant Ecology 187, 59–82.

Perry, J. N., Liebold, A. M., Rosenberg, M. S., Dungan, J., Miriti, M., Jakomulska, A., Citron-Pousty, S., 2002. Illustrations and guidelines for selecting statistical methods for quantifying spatial pattern in ecological data. Ecography 25, 578–600.

Picard, N., Karemb, M., Birnbaum, P., 2004. Species-area curve and spatial pattern. Ecoscience 11 (1), 45–54.

Piqueras, J., Klimeš, L., Redbo-Torstensson, P., 1999. Modelling the morphological response to nutrient availability in the clonal plant *Trientalis europaea* l. Plant Ecology 141, 117–127.

Pollard, J., 1971. On distance estimators of density in randomly distributed forests. Biometrics 27, 991–1002.

Price, E., Marshall, C., 1999. Clonal plants and environmental heterogeneity. Plant Ecology 141, 3–7.

Pyšek, P., Hulme, P. E., 2005. Spatio-temporal dynamics of plant invasions: Linking pattern to process. Ecoscience 12 (3), 302–315.

R Development Core Team, 2011. R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria, ISBN 3-900051-07-0.
URL `http://www.R-project.org`

Rebele, F., 2000. Competition and coexistence of rhizomatous perennial plants along a nutrient gradient. Plant Ecology 147 (77), 94.

Ripley, B., 1976. The second-order analysis of stationary point processes. Journal of Applied Probability 13 (2), 255–266.

Ripley, B., 1981. Spatial Statistics. John Wiley & Sons.

Roiloa, S., Alpert, P., Tharayil, N., Hancock, G., Bhowmik, P., 2007. Greater capacity for division of labour in clones of *Fragaria chiloensis* from patchier habitats. Journal of Ecology 95, 397–405.

Roiloa, S., Retuerto, R., 2006a. Physiological integration ameliorates effects of serpentine soilsin the clonal herb *Fragaria vesca*. Physiologia Plantarum 128, 662–676.

Roiloa, S., Retuerto, R., 2007. Responses of the clonal *Fragaria vesca* to microtopographic heterogeneity under different water and light conditions. Environmental and Experimental Botany 61, 1–9.

Roiloa, S. R., Retuerto, R., 2006b. Development, photosynthetic activity and habitat selection of the clonal plant *Fragaria vesca* growing in copper-polluted soil. 33 (10), 961–971.

Roiloa, S. R., Retuerto, R., 2006c. Small-scale heterogeneity in soil quality influences photosynthetic efficiency and habitat selection in a clonal plant. 98 (5), 1043–1052.

Sadahiro, Y., 2001. Analysis of surface changes using primitive events. International Journal of Geographical Information Science 15 (6), 523–538.

Salzman, A. G., 1985. Habitat selection in a clonal plant. Science 228 (4699), 603–604.

Sammul, M., Kull, K., Niitla, T., Mols, T., 2004. A comparison of plant communities on the basis of their clonal growth patterns. Evolutionary Ecology 18, 443–467.

Sammul, M., Kull, T., Kull, K., Novoplansky, A., 2008. Generality, specificity and diversity of clonal plant research. Evolutionary Ecology 22 (3), 273–277.

Sampaio, M., Araújo, T., Scarano, F., Stuefer, J., 2004. Directional growth of a clonal bromeliad species in response to spatial habitat heterogeneity. Evolutionary Ecology 18, 429–442.

Schick, R. S., Loarie, S. R., Colchero, F., Best, B. D., Boustany, A., Conde, D. A., Halpin, P. N., Joppa, L. N., McClellan, C. M., Clark, J. S., 2008. Understanding movement data and movement processes: Current and emerging directions. Ecology Letters 11 (12), 1338–1350.

Schwandtke, A., 1988. Second-order quantities for stationary weighted fiber processes. Mathematische Nachrichten 139, 321–334.

Semchenko, M., John, E., Hutchings, M., 2007. Effects of physical connection and genetic identity of neighbouring ramets on root-placement patterns in two clonal species. New Phytologist 176, 644–654.

Semchenko, M., Zobel, K., Hutchings, M., 2010. To compete or not to compete: an experimental study of interactions between plant species with contrasting root behaviour. Evolutionary Ecology 24, 1433–1445.

Shibaike, H., Ishiguri, Y., Kawano, S., 1996. Plastic responses to nutrient and light intensity gradients in populations of *Oxalis corniculata* l. (oxalidaceae). Plant Species Biology 11, 213–223.

Slade, A., Hutchings, M., 1987a. Clonal integration and plasticity in foraging behaviour in *Glechoma hederacea*. Journal of Ecology 75, 1023–1036.

Slade, A., Hutchings, M., 1987b. The effects of light intensity on foraging in the clonal herb *Glechoma hederacea*. Journal of Ecology 75, 639–650.

Slade, A., Hutchings, M., 1987c. The effects of nutrient availability on foraging in the clonal herb *Glechoma hederacea*. Journal of Ecology 75, 95–112.

Sosnová, M., van Diggelen, R., Klimešová, J., 2010. Distribution of clonal growth froms in wetlands. Aquatic Biology 92, 33–39.

Stoyan, D., 1981. On the second-order analysis of stationary planar fibre processes. Mathematische Nachrichten 102, 189–199.

Stoyan, D., 1985. Stereological determination of orientations, second-order quantities and correcations for random spatial fiber systems. Biometrical Journal 27 (4), 411–425.

Stoyan, D., 1990. Stereology and stochastic geometry. International Statistical Review 58 (3), 227–242.

Stoyan, D., 1998. Random sets: Models and statistics. International Statistical Review 66 (1), 1–27.

Stoyan, D., Gerlach, W., 1987. Stereological determination of curvature distributions of spatial fibre systems. Journal of Microscopy 148, 297–305.

Stoyan, D., Kendal, W., Mecke, J., 1995. Stochastic Geometry and its Applications, 2nd Edition. John Wiley and Sons, Ltd.

Stoyan, D., Kuschka, V., 2001. On animal abundance estimation based on pitfall traps. Biometrical Journal 43, 45–52.

Stoyan, D., Mecke, J., Pohlmann, S., 1980. Formulas for stationary planar fibre processes II - Partially oriented fibre systems. Mathematische Operationsforschung und Statistik Series Statistics 11 (2), 281–286.

Stoyan, H., Stoyan, D., 1986. Simple stochastic models for the analysis of dislocation distributions. Physica Status Solidi A-Applied Research 97 (1), 163–172.

Strogatz, S., 2001. Exploring complex networks. Nature 410, 268–276.

Stuefer, J., Erschbamer, B., Huber, H., Suzuki, J., 2002. The ecology and evolutionary biology of clonal plants: An introduction to the proceedings of Clone-2000. Evolutionary Ecology 15, 223–230.

Stuefer, J. F., Gmez, S., Mlken, T. V., 2004. Clonal integration beyond resource sharing: implications for defence signaling and disease transmission in clonal plant networks. Evolutionary Ecology 18, 647–667.

Stuefer, J. F., Kroon, H. D., During, H. J., 1996. Exploitation of environmental heterogeneity by spatial division of labour in a clonal plant. Functional Ecology 10 (328), 334.

Sun, X., Niu, J., Zhou, H., 2011. Buffalograss decreases ramet propogation in infertile patches to enhance interconnected ramet proliferation in fertile patches. Flora 206, 380–386.

Tolvanen, A., Siikamäki, P., Mutikainen, P., 2004. Population biology of clonal plants: Forward to the proceedings from the 7th Clonal Plant Workshop. Evolutionary Ecology 18, 403–408.

Tworkoski, T., Benassi, T., Takeda, F., 2001. The effect of nitrogen on stolon and ramet growth in four genotypes of *Fragaria chiloensis* L.. Scientia Horticulturae 88, 97–106.

Ushimaru, A., Genkai-Kato, M., 2011. A theoretical framework for resource translocation during sexual reproduction in modular organisms. Evolutionary Ecology 25, 885–898.

van der Maarel, E., 1996. Pattern and process in the plant community: Fifty years after A.S. Watt. Journal of Vegetation Science 7, 19–28.

van Kleunen, M., Fischer, M., 2001. Adaptive evolution of plastic foraging responses in a clonal plant. Ecology 82, 3309–3319.

Watt, A., 1947. Pattern and process in the plant community. Journal of Ecology 35, 1–22.

Weibel, E. R., 1980. Stereological Methods Vol. 2: Theoretical Foundations. Academic Press.

Welham, C. V. J., Turkington, R., Sayre, C., 2002. Morphological plasticity of white clover (*Trifolium repens* l.) in response to spatial and temporal resource heterogeneity. Oecologia 130, 231–238.

Wiegand, T., Moloney, K., 2004. Rings, circles, and null-models for point pattern analysis in ecology. OIKOS 104, 209–229.

Wijesinghe, D. K., Handel, S. N., 1994. Advantages of clonal growth in heterogeneous habitats: An experiment with *Potentilla simplex*. Journal of Ecology 82, 495–502.

Wijesinghe, D. K., Hutchings, M. J., 1997. The effects of spatial scale of environmental heterogeneity on the growth of a clonal plant: an experimental study with *Glechoma hederacea*. Journal of Ecology 85, 17–28.

Wijesinghe, D. K., Hutchings, M. J., 1999. The effects of environmental heterogeneity on the performance of *Glechoma hederacea*: the interactions between patch contrast and patch scale. Journal of Ecology 87, 860–872.

Wijesinghe, D. K., John, E. A., Beurskens, S., Hutchings, M. J., 2001. The effects of environmental heterogeneity on the performance of *Glechoma hederacea*: the interactions between patch contrast and patch scale. Journal of Ecology 89, 972–983.

Wilk, J., Kramer, A., Ashley, M., 2009. High variation in clonal vs. sexual reproduction in populations of the wild strawberry, *Fragaria virginiana* (rosaceae). Annals of Botany 104, 1413–1419.

Xiao, Y., Tang, J., Qing, H., Ouyang, Y., Zhao, Y., Zhou, C., An, S., 2010. Clonal integration enhances flood tolerance of *Spartina alterniflora* daughter ramets. Aquatic Biology 92, 9–13.

Xiao, Y., Tang, J., Qing, H., Zhou, C., An, S., 2011. Effects of salinity and clonal integration on growth and sexual reproduction of the invasive grass *Spartina alterniflora*. Flora 206, 736–741.

Zar, J. H., 1999. Biostatistical Analysis, 4th Edition. Prentice Hall.

Zehm, A., Nobis, M., Schwabe, A., 2003. Multiparameter analysis of vertical vegetation structure based on digital image processing. Flora 198, 142–160.

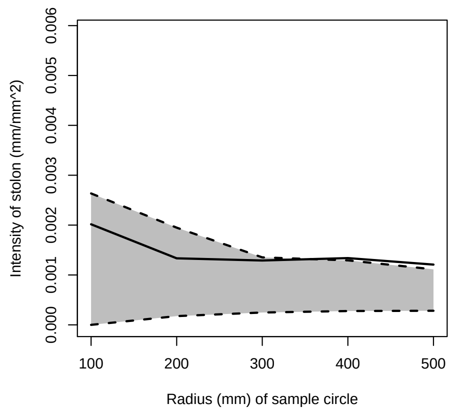
Zhang, Y., Zhang, Q., Luo, P., Wu, N., 2009. Photosynthetic response of *Fragaria orientalis* in different water contrast clonal integration. Ecological Research 24, 617–625.

Appendix A

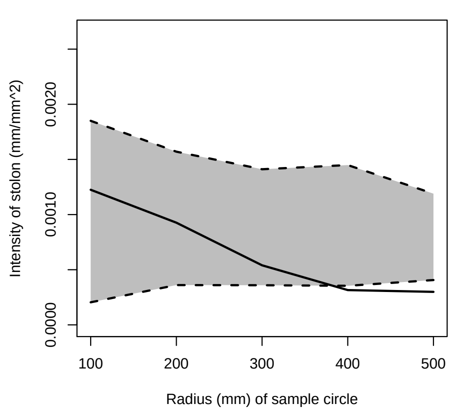
Appendix A

A.1 Chapter 4: Additional Figures

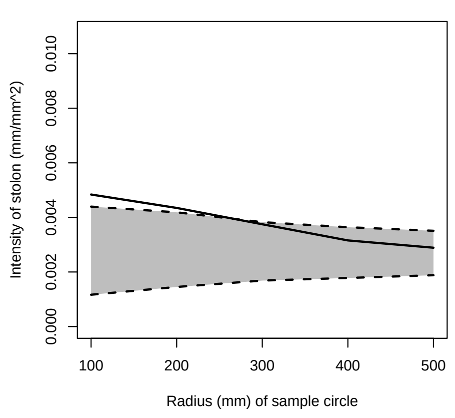
A.1.1 Genet randomization



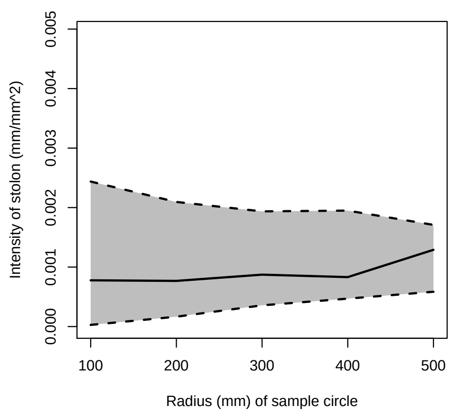
(a) Plot 7



(b) Plot 8

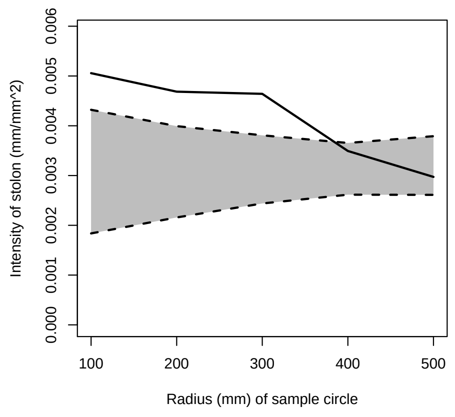


(c) Plot 14

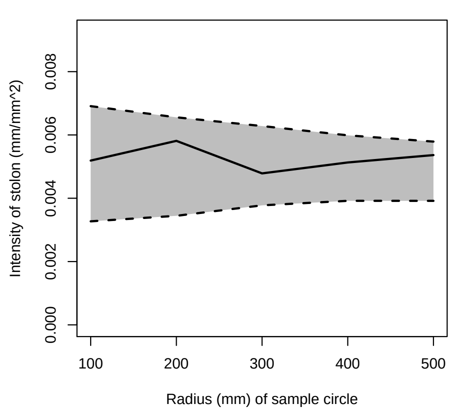


(d) Plot 18

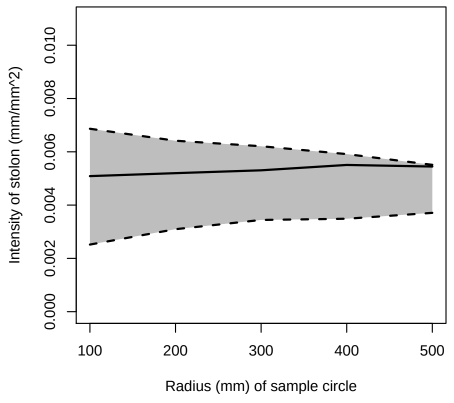
Figure A.1: Constrained randomization analysis of *F. virginiana* plots showing genet-genet stolon intensity (solid line) with an 80% envelope (grey) from 100 randomizations of the genet data (n = 100 sampling circles). Plot size is 2 m x 2 m, and n = 16 original transplant ramets in each plot. Plots were established at the Ellerslie Research Station, and ramets were grown during the summer of 2005. Initial ramet arrangement was random.



(a) Plot 25

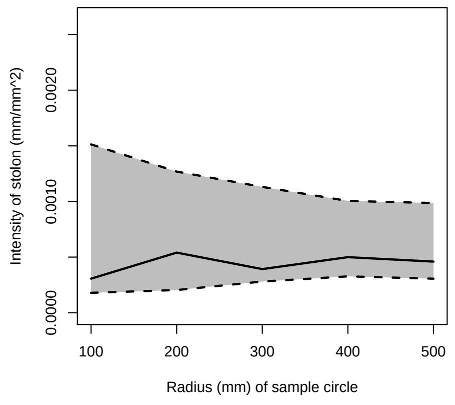


(b) Plot 46

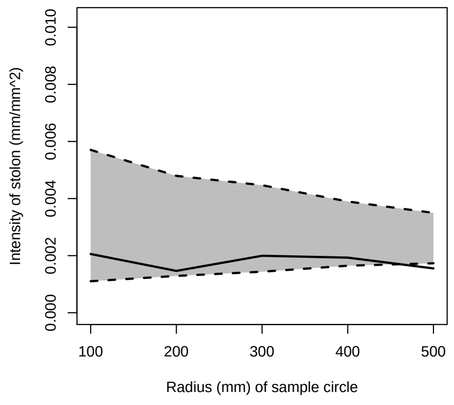


(c) Plot 47

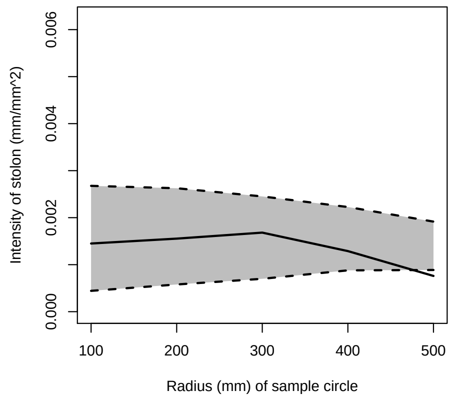
Figure A.2: Constrained randomization analysis of *F. virginiana* plots showing genet-genet stolon intensity (solid line) with an 80% envelope (grey) from 100 randomizations of the genet data (n = 100 sampling circles). Plot size is 2 m x 2 m, and n = 16 original transplant ramets in each plot. Plots were established at the Ellerslie Research Station, and ramets were grown during the summer of 2005. Initial ramet arrangement was random.



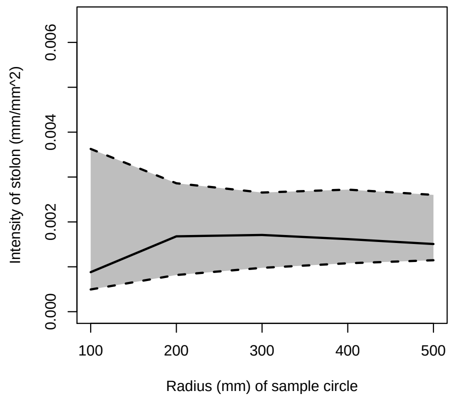
(a) Plot 6



(b) Plot 9

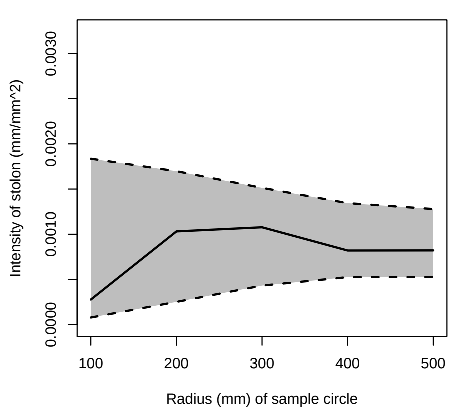


(c) Plot 10

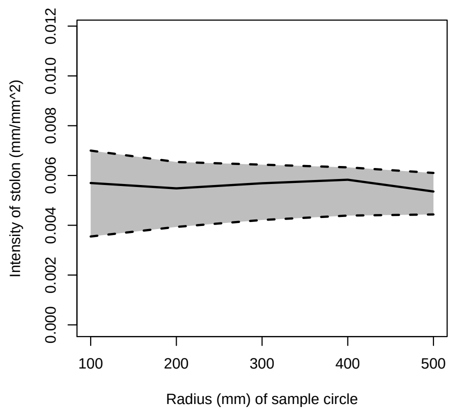


(d) Plot 11

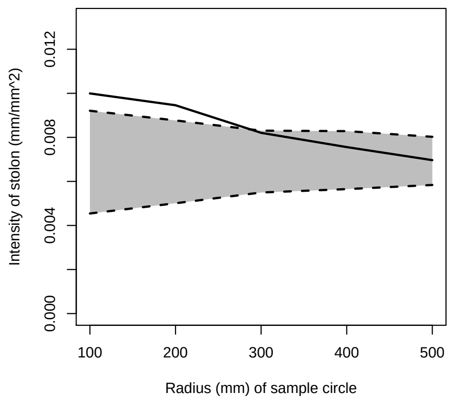
Figure A.3: Constrained randomization analysis of *F. virginiana* plots showing genet-genet stolon intensity (solid line) with an 80% envelope (grey) from 100 randomizations of the genet data (n = 100 sampling circles). Plot size is 2 m x 2 m, and n = 16 original transplant ramets in each plot. Plots were established at the Ellerslie Research Station, and ramets were grown during the summer of 2005. Initial ramet arrangement was clustered.



(a) Plot 17

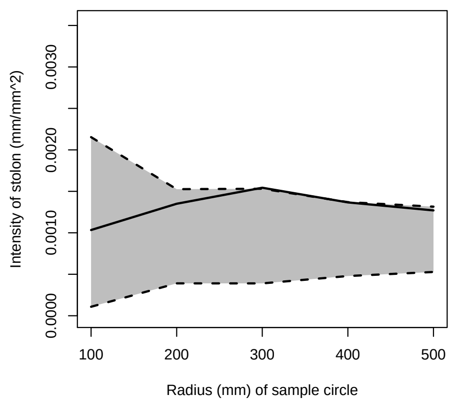


(b) Plot 34

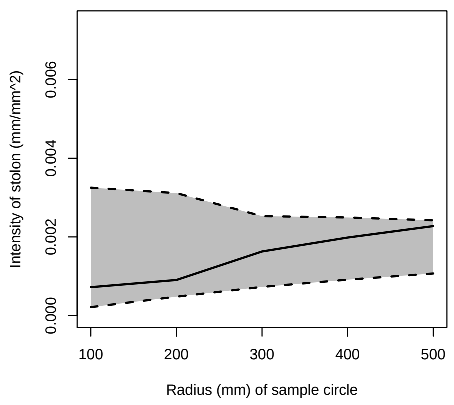


(c) Plot 41

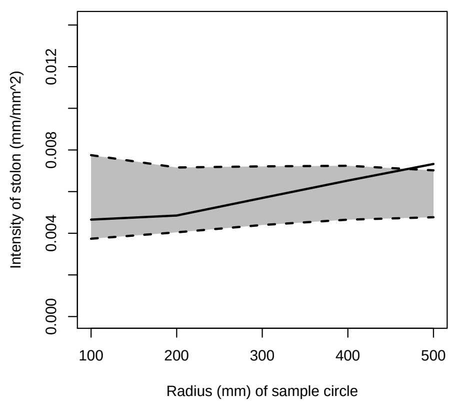
Figure A.4: Constrained randomization analysis of *F. virginiana* plots showing genet-genet stolon intensity (solid line) with an 80% envelope (grey) from 100 randomizations of the genet data (n = 100 sampling circles). Plot size is 2 m x 2 m, and n = 16 original transplant ramets in each plot. Plots were established at the Ellerslie Research Station, and ramets were grown during the summer of 2005. Initial ramet arrangement was clustered.



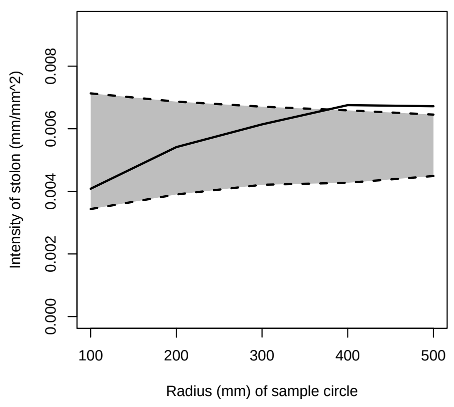
(a) Plot 4



(b) Plot 15

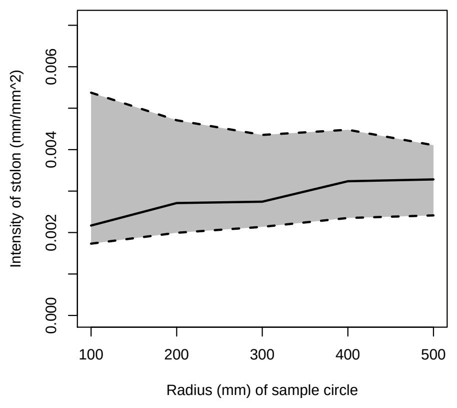


(c) Plot 22

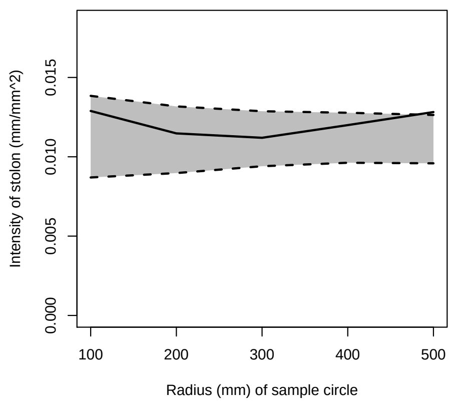


(d) Plot 23

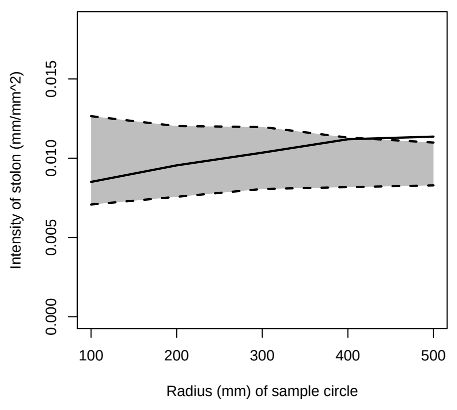
Figure A.5: Constrained randomization analysis of *F. virginiana* plots showing genet-genet stolon intensity (solid line) with an 80% envelope (grey) from 100 randomizations of the genet data (n = 100 sampling circles). Plot size is 2 m x 2 m, and n = 16 original transplant ramets in each plot. Plots were established at the Ellerslie Research Station, and ramets were grown during the summer of 2005. Initial ramet arrangement was regular.



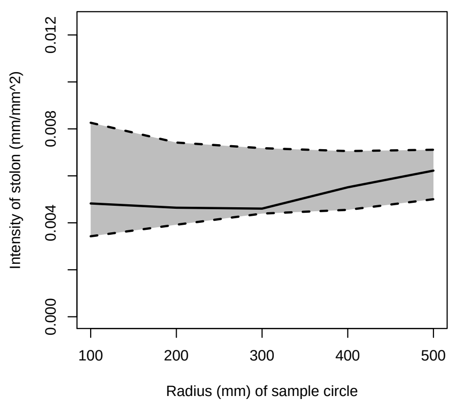
(a) Plot 26



(b) Plot 28



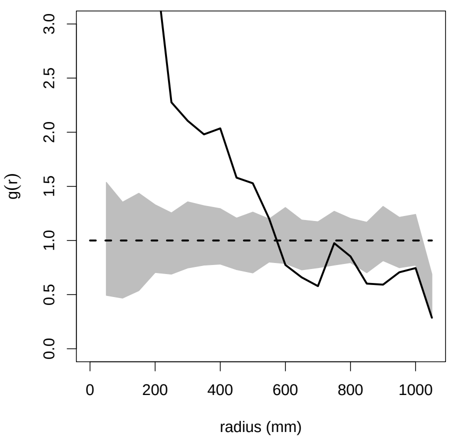
(c) Plot 29



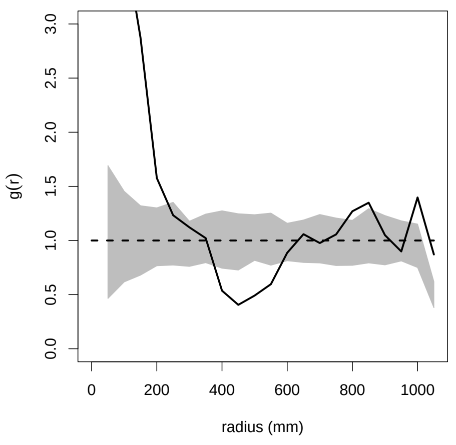
(d) Plot 33

Figure A.6: Constrained randomization analysis of *F. virginiana* plots showing genet-genet stolon intensity (solid line) with an 80% envelope (grey) from 100 randomizations of the genet data (n = 100 sampling circles). Plot size is 2 m x 2 m, and n = 16 original transplant ramets in each plot. Plots were established at the Ellerslie Research Station, and ramets were grown during the summer of 2005. Initial ramet arrangement was regular.

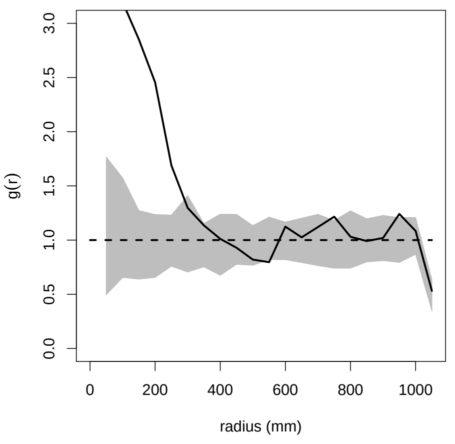
A.1.2 Pair correlation, $g(r)$, of fibre points



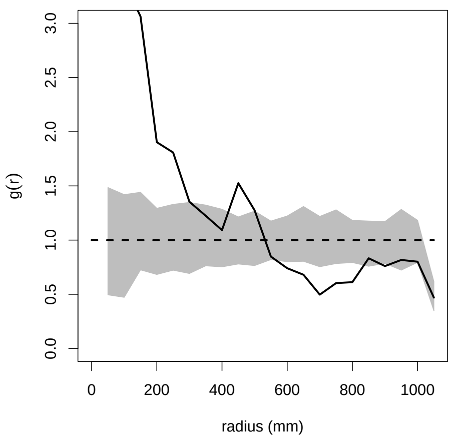
(a) Plot 7



(b) Plot 8

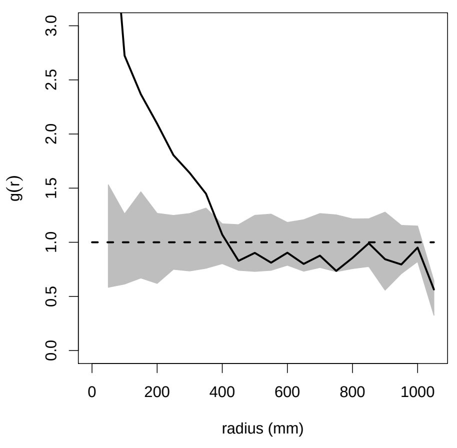


(c) Plot 14

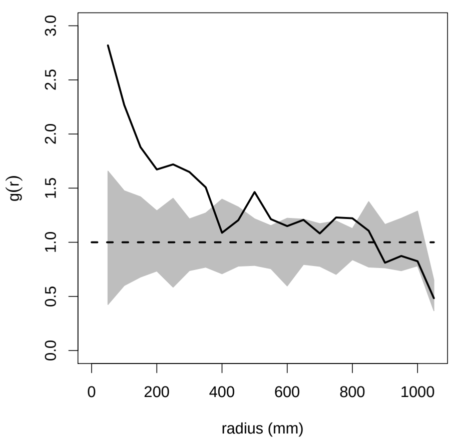


(d) Plot 18

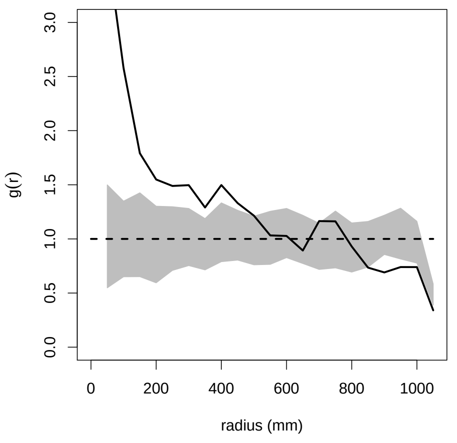
Figure A.7: Pair correlation, $g(r)$, calculated on all random fibre points for plots having an initial ramet arrangement that was random. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.



(a) Plot 25

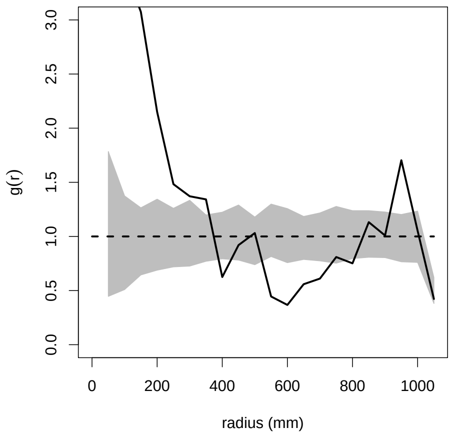


(b) Plot 46

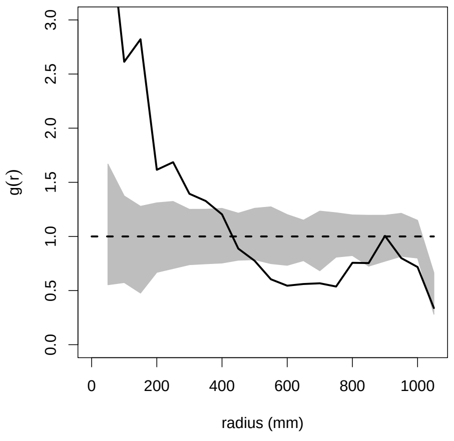


(c) Plot 47

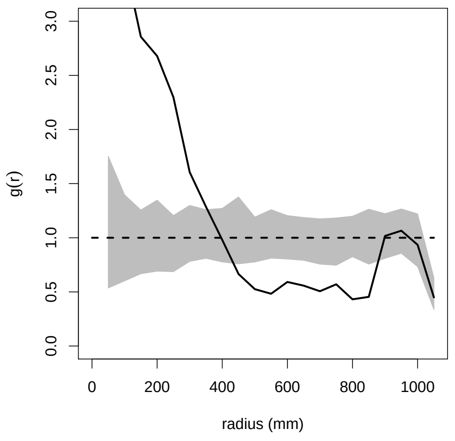
Figure A.8: Pair correlation, $g(r)$, calculated on all random fibre points for plots having an initial ramet arrangement that was random. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.



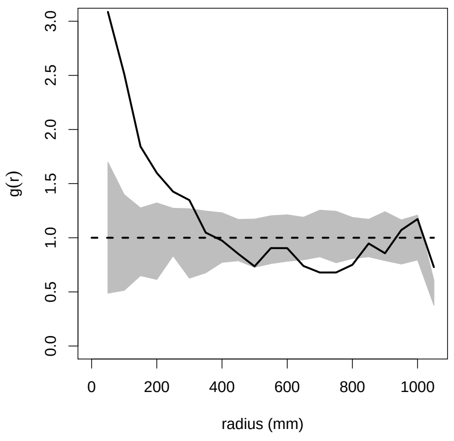
(a) Plot 6



(b) Plot 9

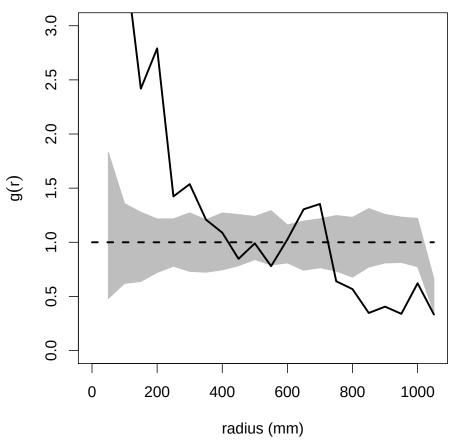


(c) Plot 10

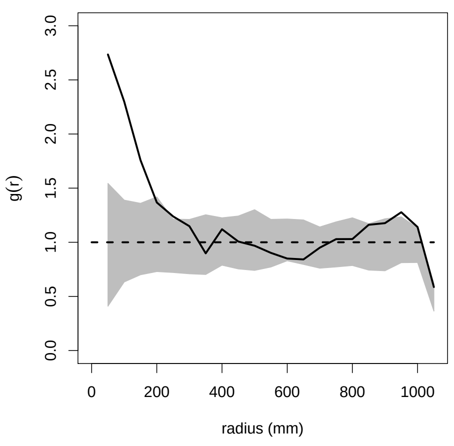


(d) Plot 11

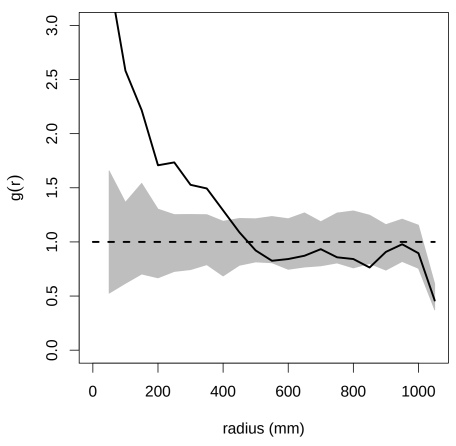
Figure A.9: Pair correlation, $g(r)$, calculated on all random fibre points for plots having an initial ramet arrangement that was clustered. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.



(a) Plot 17

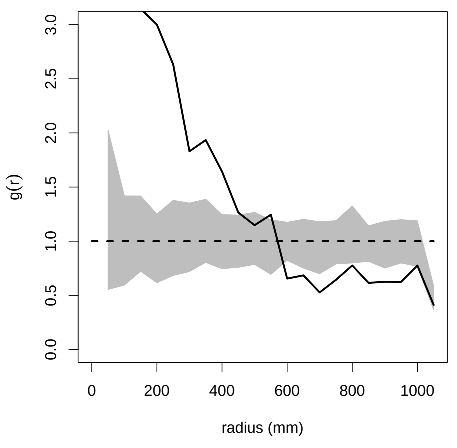


(b) Plot 34

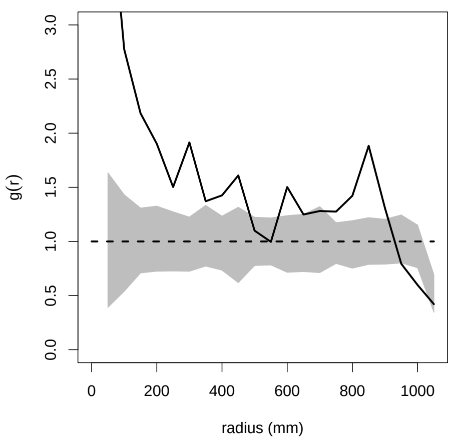


(c) Plot 41

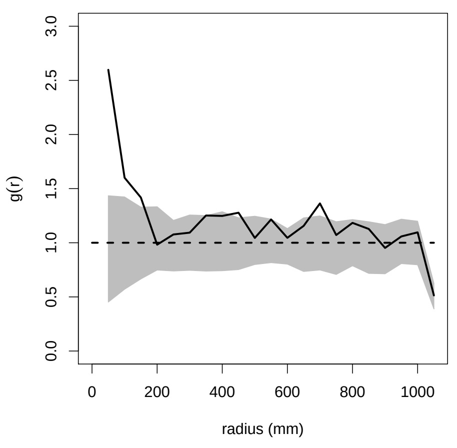
Figure A.10: Pair correlation, $g(r)$, calculated on all random fibre points for plots having an initial ramet arrangement that was clustered. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.



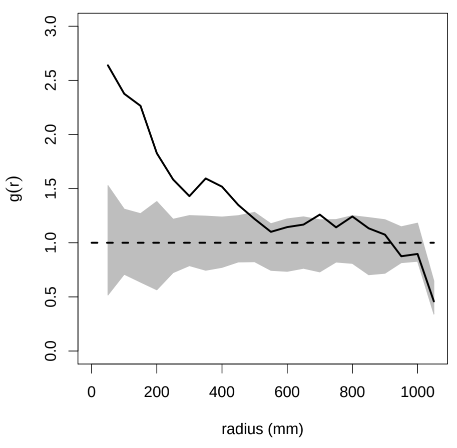
(a) Plot 4



(b) Plot 15

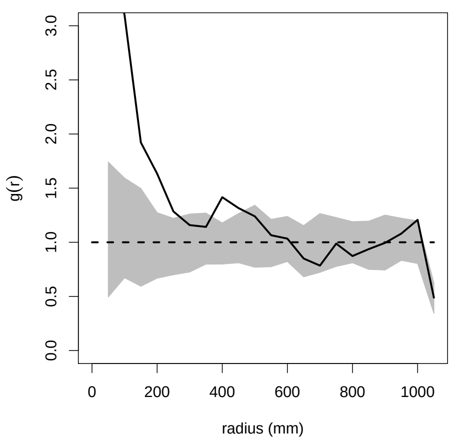


(c) Plot 22

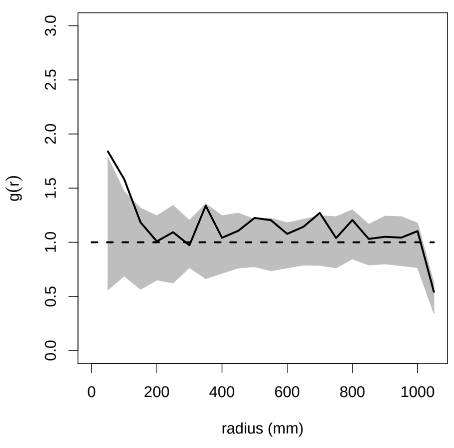


(d) Plot 23

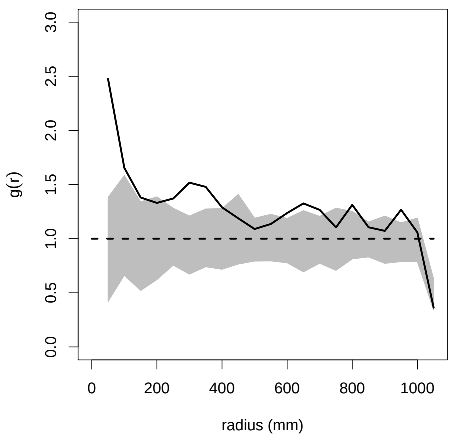
Figure A.11: Pair correlation, $g(r)$, calculated on all random fibre points for plots having an initial ramet arrangement that was regular. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.



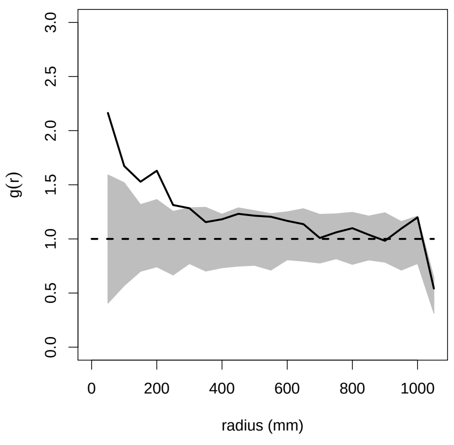
(a) Plot 26



(b) Plot 28



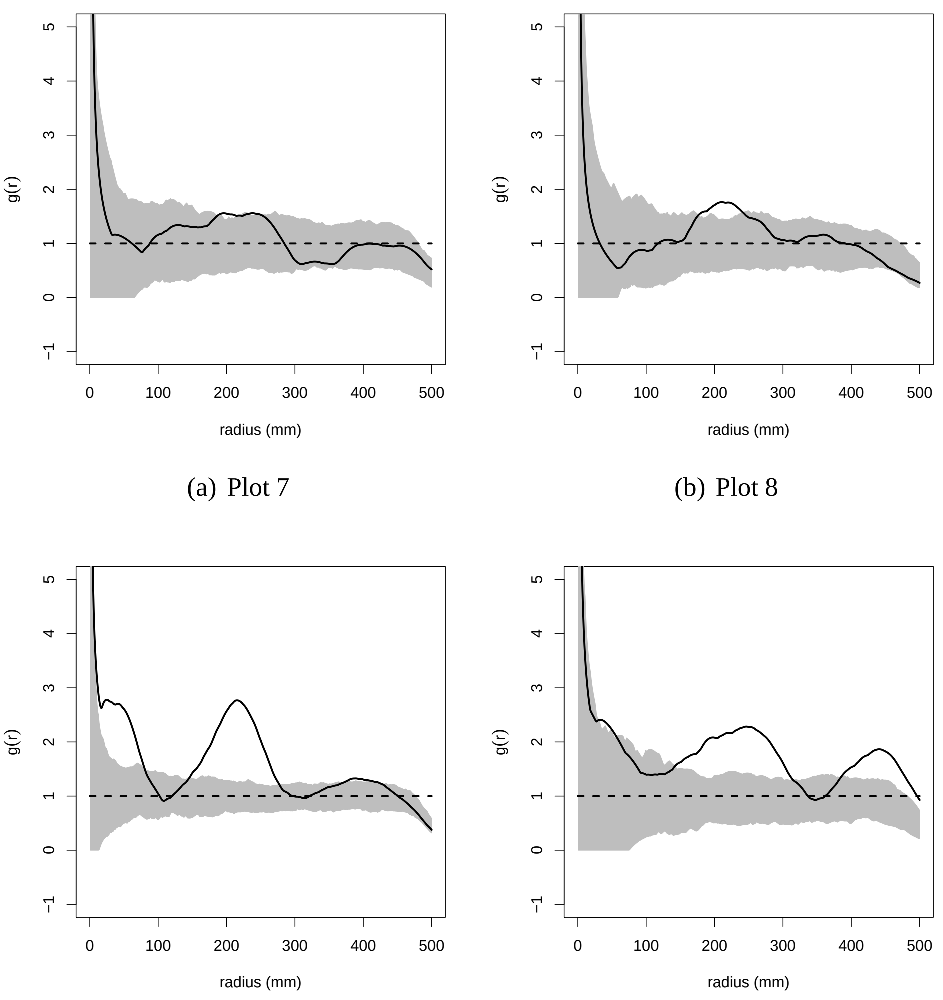
(c) Plot 29



(d) Plot 33

Figure A.12: Pair correlation, $g(r)$, calculated on all random fibre points for plots having an initial ramet arrangement that was regular. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.

A.1.3 Point pattern analyses of ramets



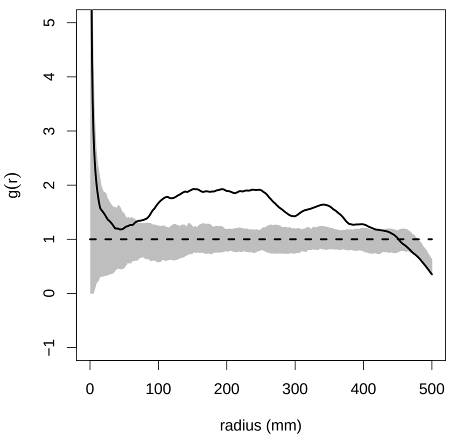
(a) Plot 7

(b) Plot 8

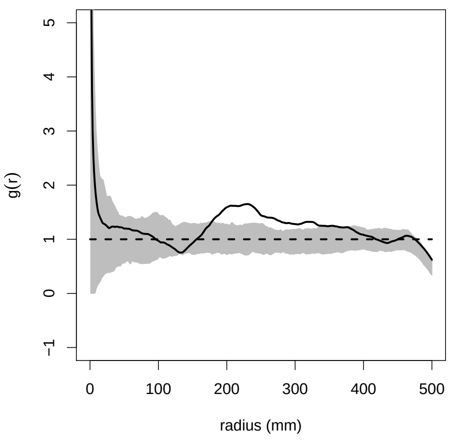
(c) Plot 14

(d) Plot 18

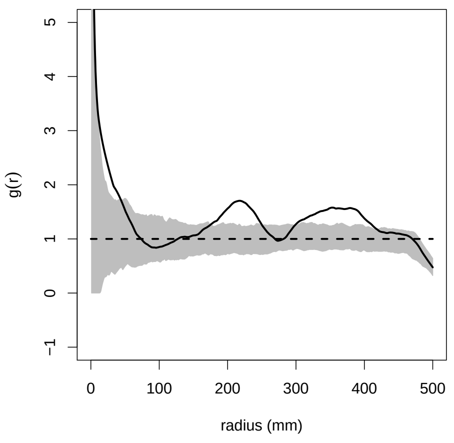
Figure A.13: Pair correlation, $g(r)$, calculated on all ramets (parent and daughter) for plots having an initial parent ramet arrangement that was random. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.



(a) Plot 25

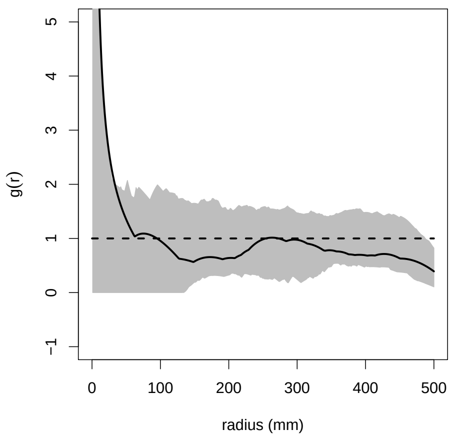


(b) Plot 46

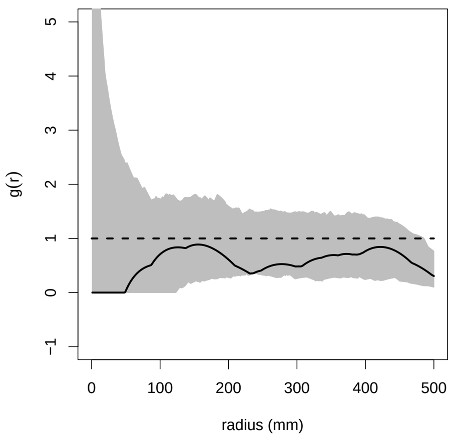


(c) Plot 47

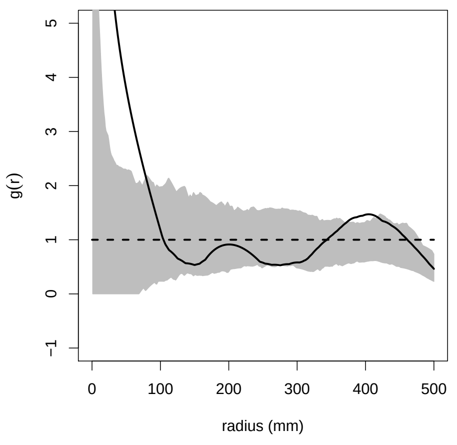
Figure A.14: Pair correlation, *g*(*r*), calculated on all ramets (parent and daughter) for plots having an initial parent ramet arrangement that was random. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.



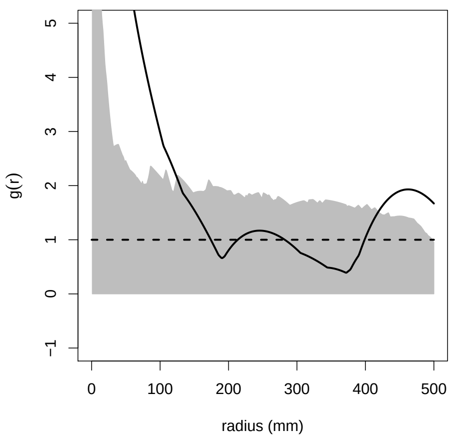
(a) Plot 7



(b) Plot 8

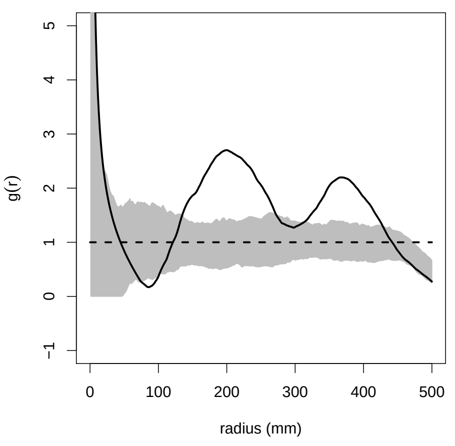


(c) Plot 14

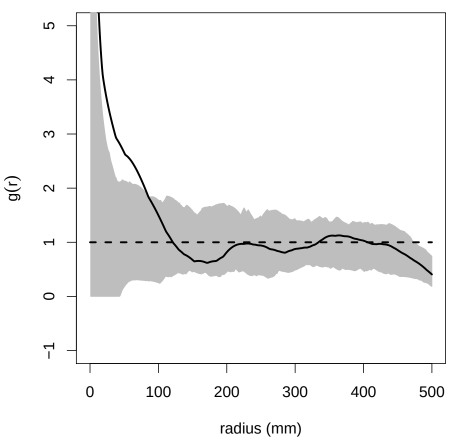


(d) Plot 18

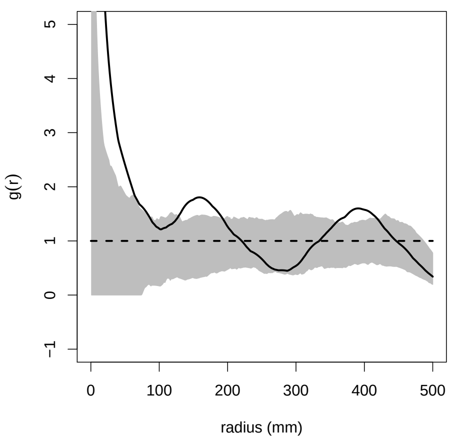
Figure A.15: Pair correlation, $g(r)$, calculated on parent ramets for plots having an initial ramet arrangement that was random. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.



(a) Plot 25

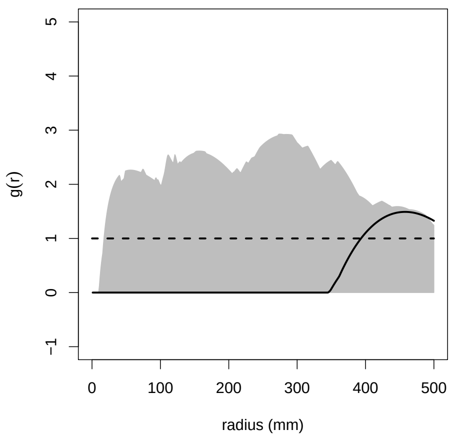


(b) Plot 46

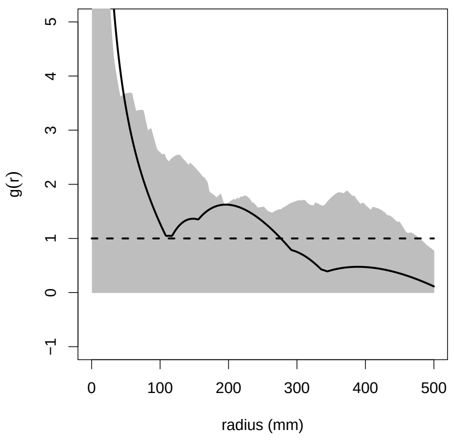


(c) Plot 47

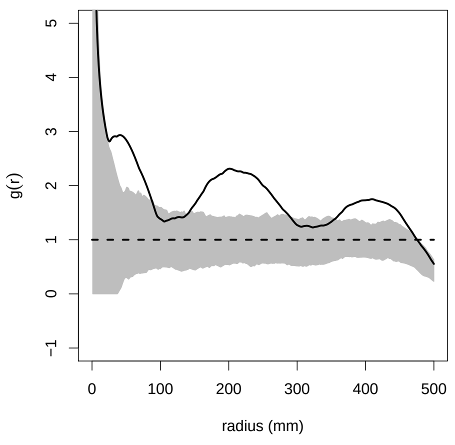
Figure A.16: Pair correlation, $g(r)$, calculated on parent ramets for plots having an initial ramet arrangement that was random. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.



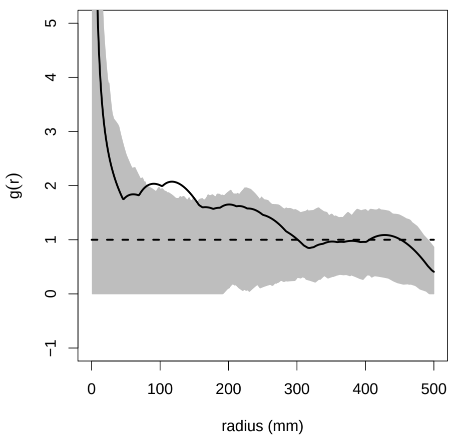
(a) Plot 7



(b) Plot 8

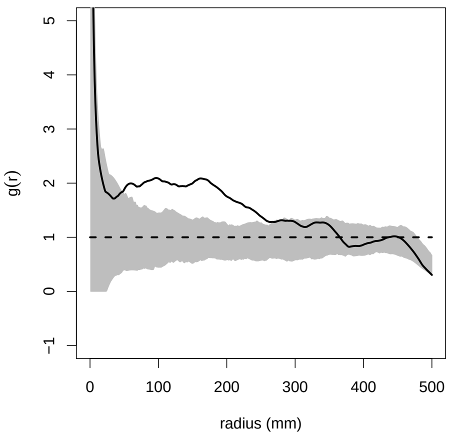


(c) Plot 14

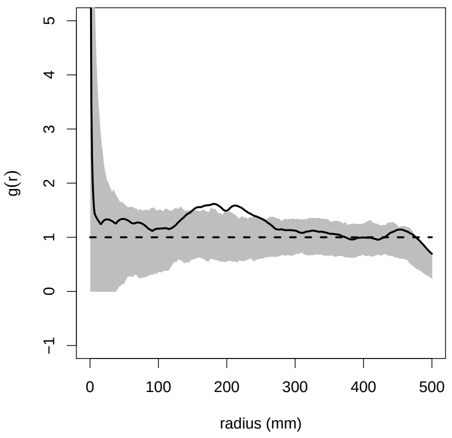


(d) Plot 18

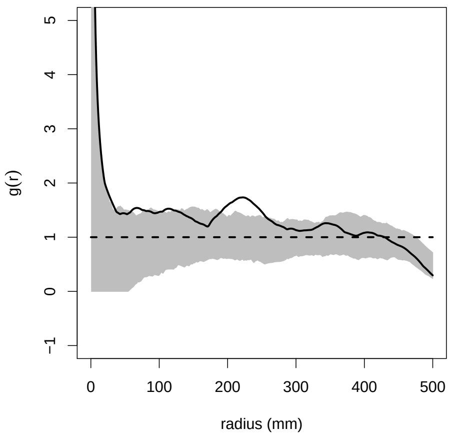
Figure A.17: Pair correlation, $g(r)$, calculated on daughter ramets for plots having an initial ramet arrangement that was random. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.



(a) Plot 25

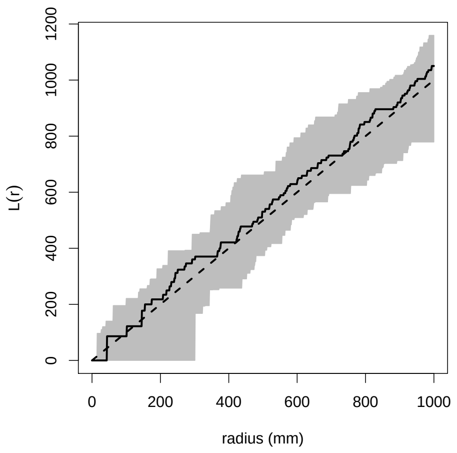


(b) Plot 46

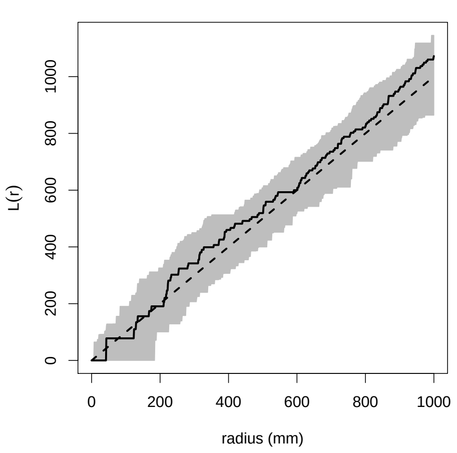


(c) Plot 47

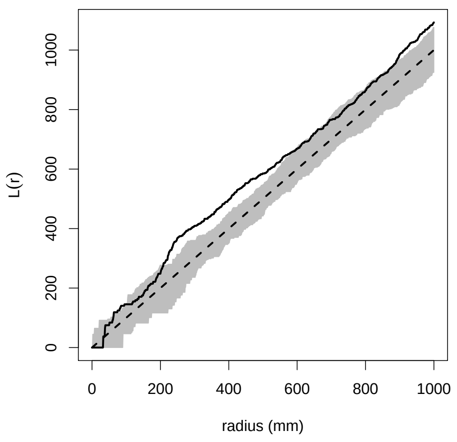
Figure A.18: Pair correlation, $g(r)$, calculated on daughter ramets for plots having an initial ramet arrangement that was random. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.



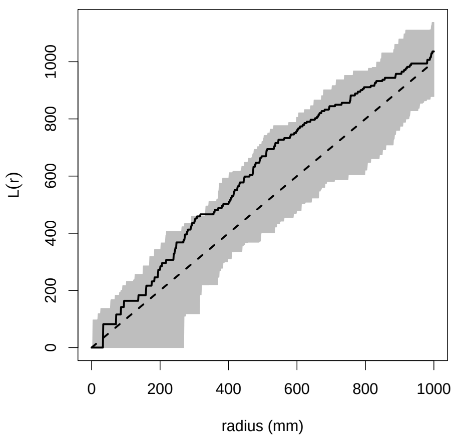
(a) Plot 7



(b) Plot 8

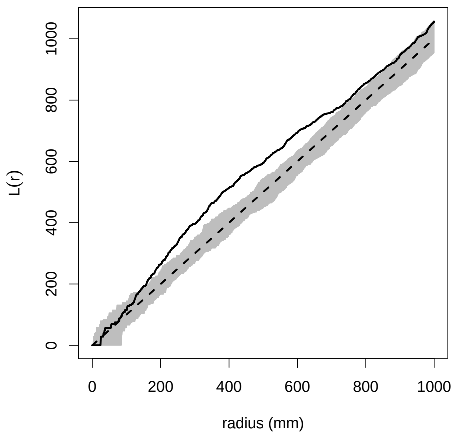


(c) Plot 14

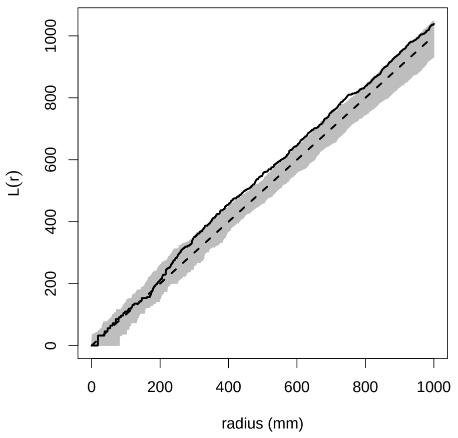


(d) Plot 18

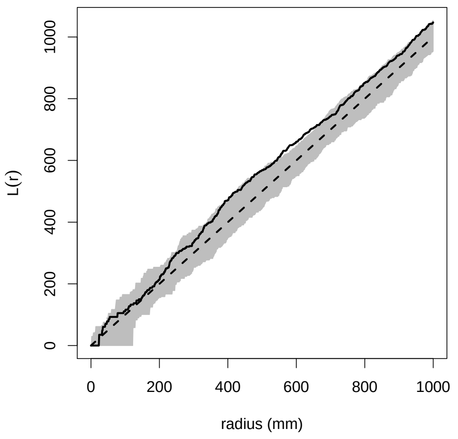
Figure A.19: $L(r)$ calculated on all ramets for plots having an initial ramet arrangement that was random. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.



(a) Plot 25

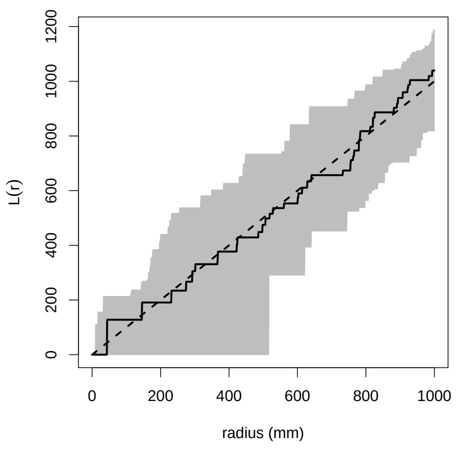


(b) Plot 46

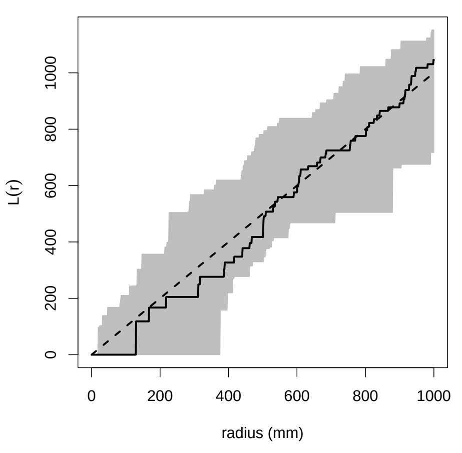


(c) Plot 47

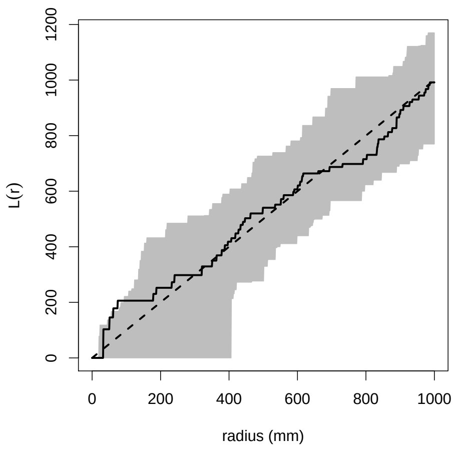
Figure A.20: $L(r)$ calculated on all ramets for plots having an initial ramet arrangement that was random. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.



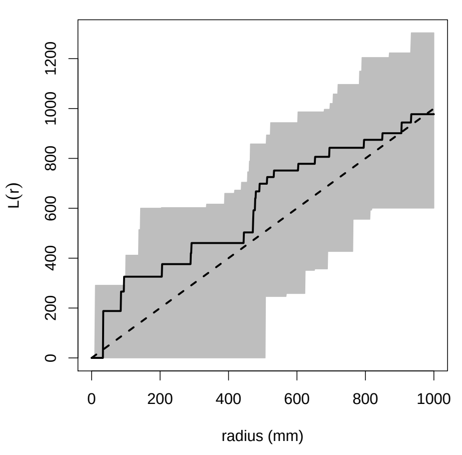
(a) Plot 7



(b) Plot 8

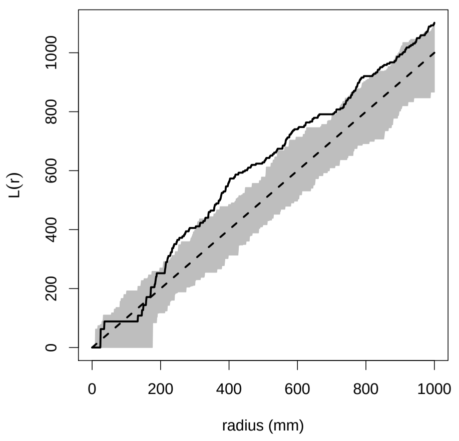


(c) Plot 14

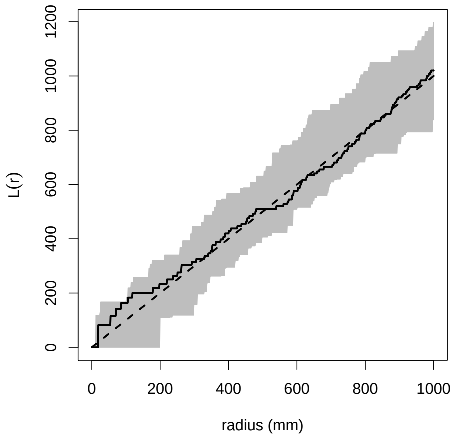


(d) Plot 18

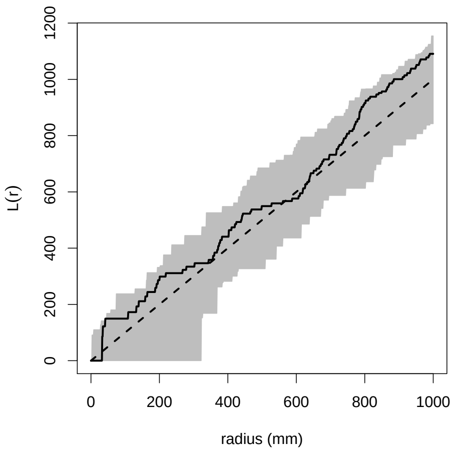
Figure A.21: $L(r)$ calculated on parent ramets for plots having an initial ramet arrangement that was random. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.



(a) Plot 25

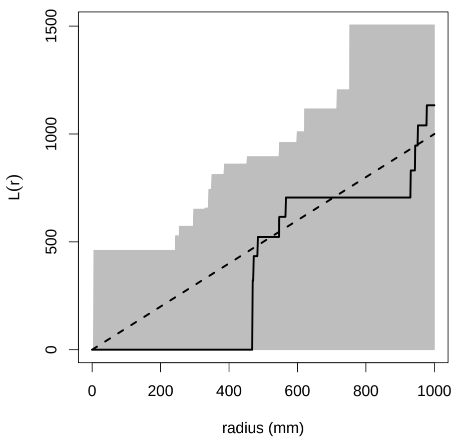


(b) Plot 46

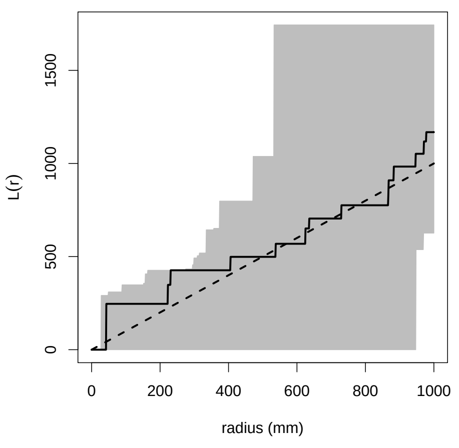


(c) Plot 47

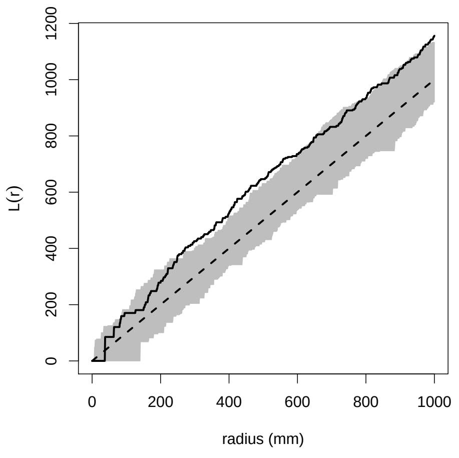
Figure A.22: $L(r)$ calculated on parent ramets for plots having an initial ramet arrangement that was random. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.



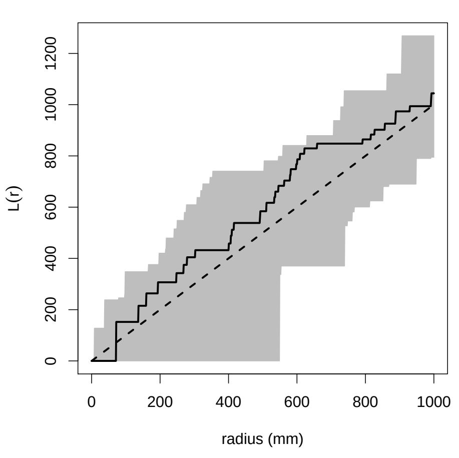
(a) Plot 7



(b) Plot 8

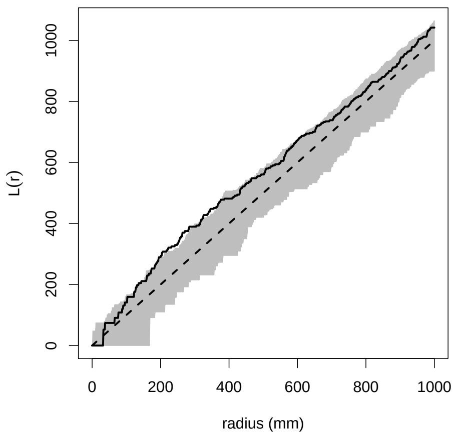


(c) Plot 14

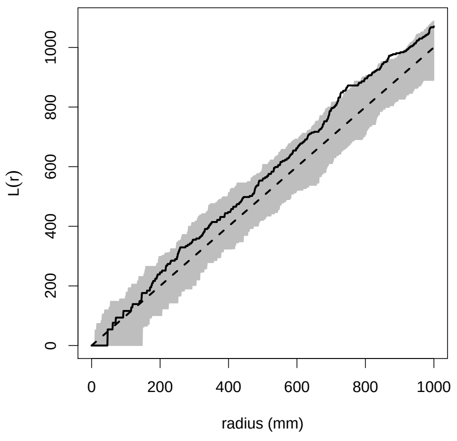


(d) Plot 18

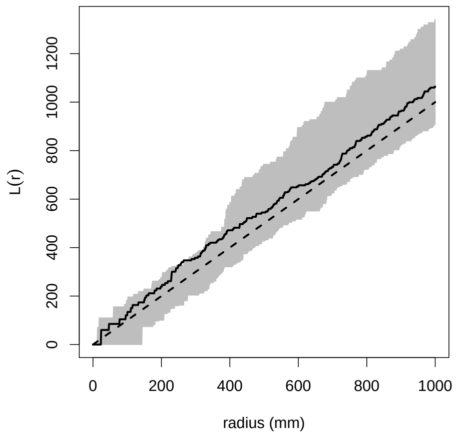
Figure A.23: $L(r)$ calculated on daughter ramets for plots having an initial ramet arrangement that was random. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.



(a) Plot 25

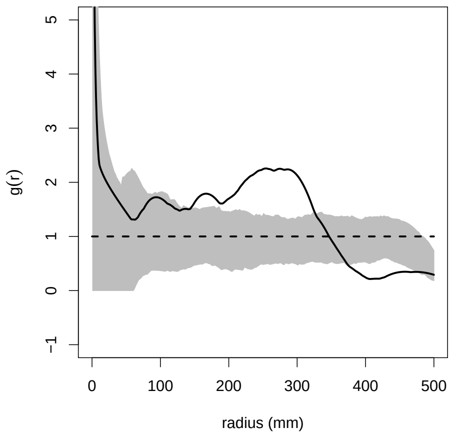


(b) Plot 46

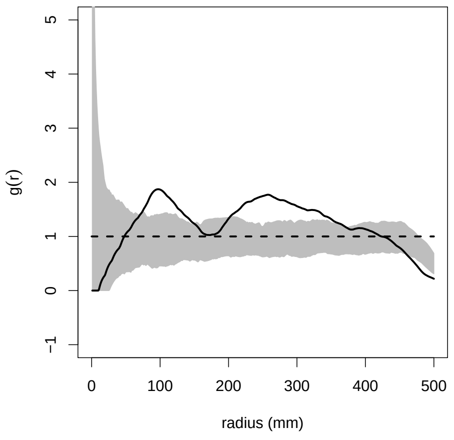


(c) Plot 47

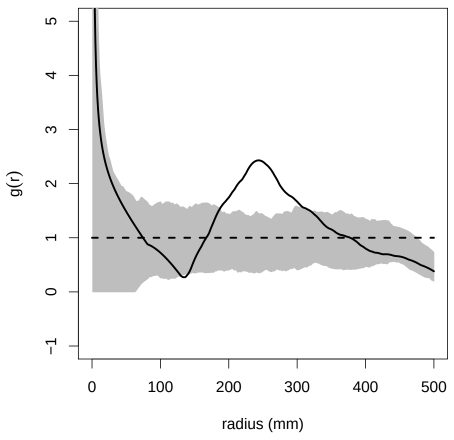
Figure A.24: $L(r)$ calculated on daughter ramets for plots having an initial ramet arrangement that was random. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.



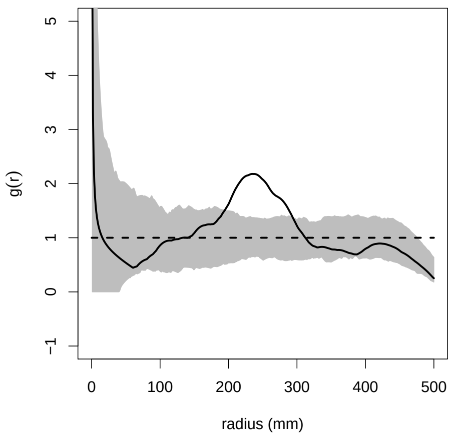
(a) Plot 6



(b) Plot 9

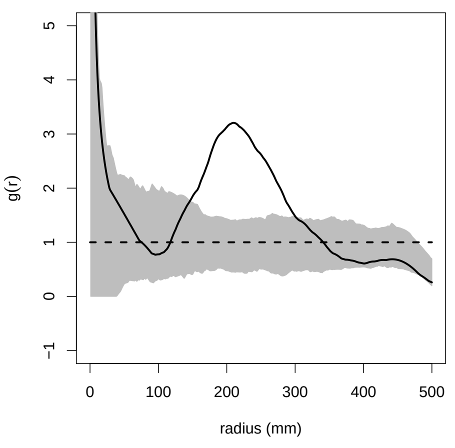


(c) Plot 10

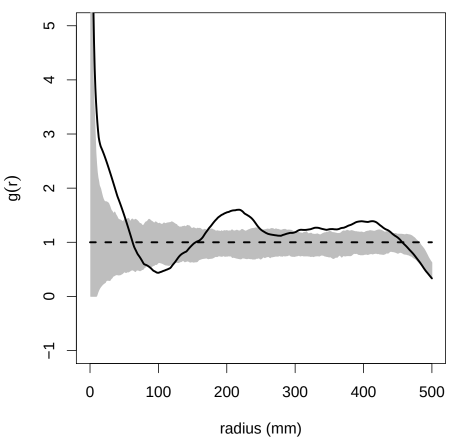


(d) Plot 11

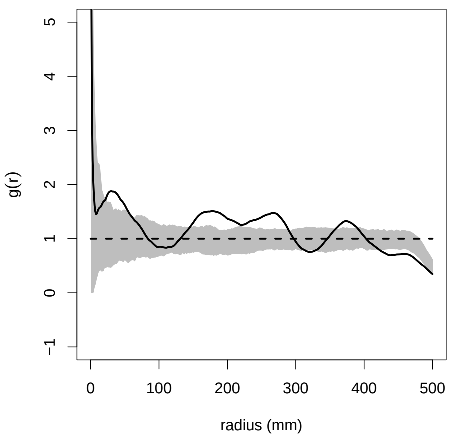
Figure A.25: Pair correlation, *g*(*r*), calculated on all ramets (parent and daughter) for plots having an initial parent ramet arrangement that was clustered. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.



(a) Plot 17

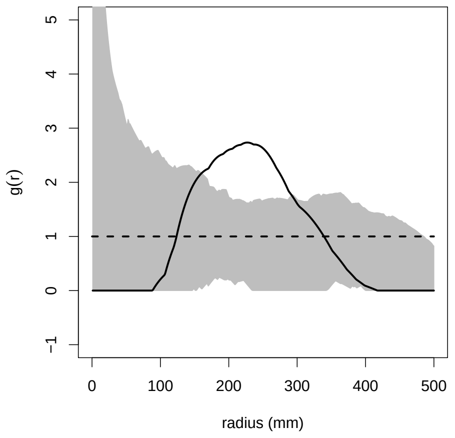


(b) Plot 34

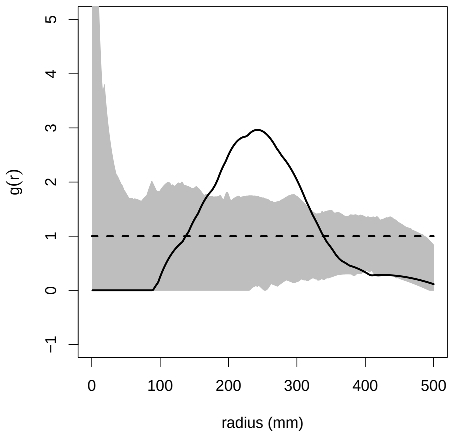


(c) Plot 41

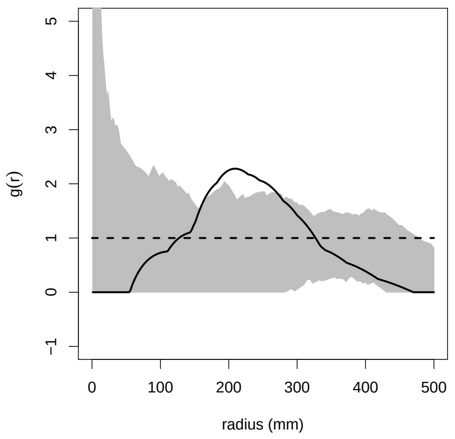
Figure A.26: Pair correlation, $g(r)$, calculated on all ramets (parent and daughter) for plots having an initial parent ramet arrangement that was clustered. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.



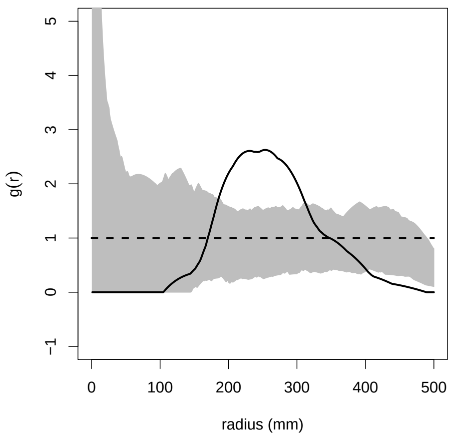
(a) Plot 6



(b) Plot 9

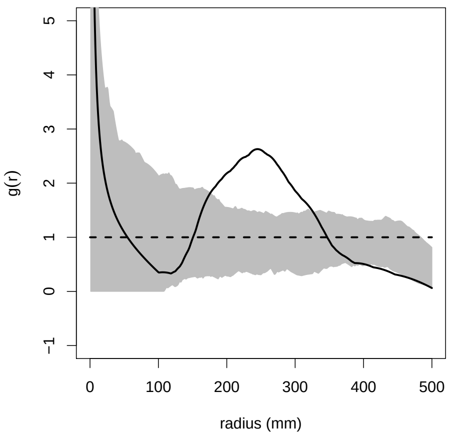


(c) Plot 10

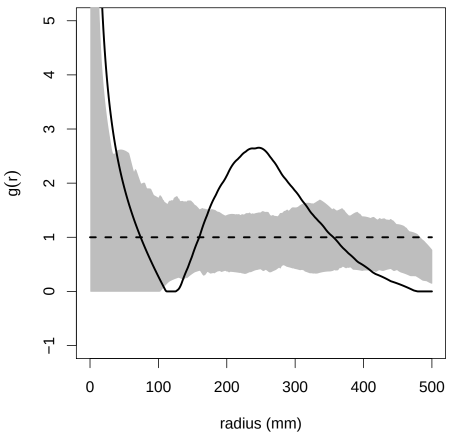


(d) Plot 11

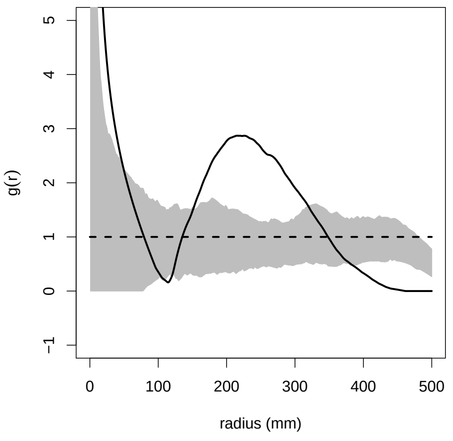
Figure A.27: Pair correlation, $g(r)$, calculated on parent ramets for plots having an initial ramet arrangement that was clustered. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.



(a) Plot 17

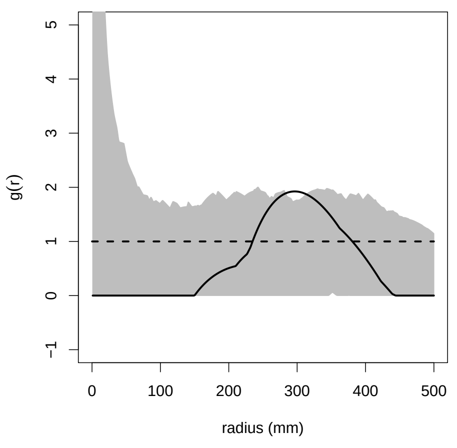


(b) Plot 34

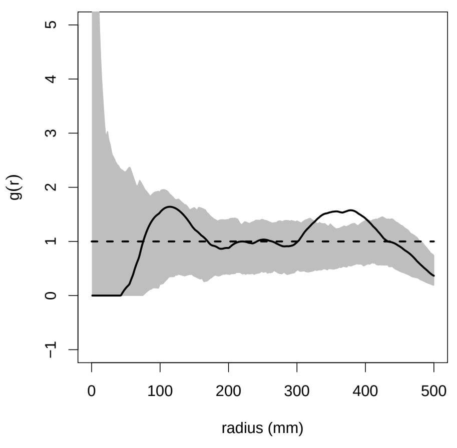


(c) Plot 41

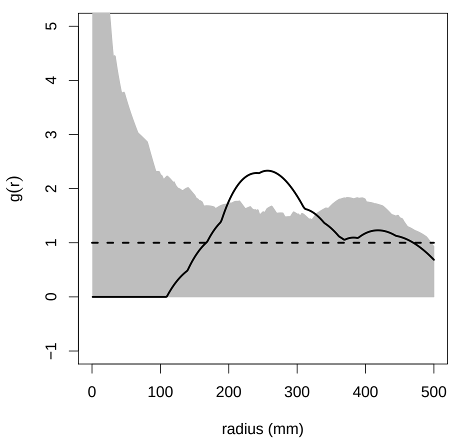
Figure A.28: Pair correlation, *g*(*r*), calculated on parent ramets for plots having an initial ramet arrangement that was clustered. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.



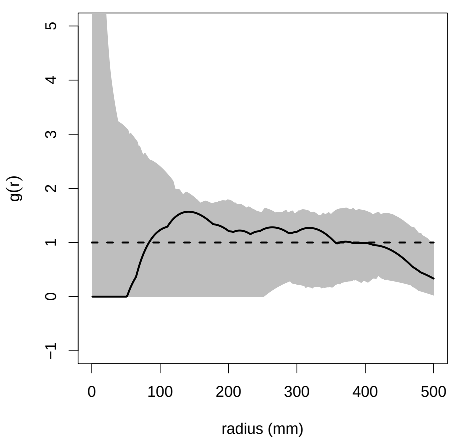
(a) Plot 6



(b) Plot 9

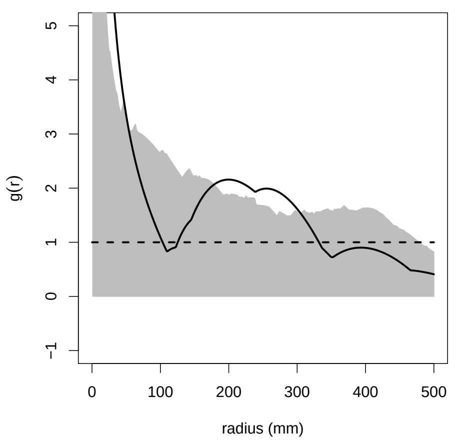


(c) Plot 10

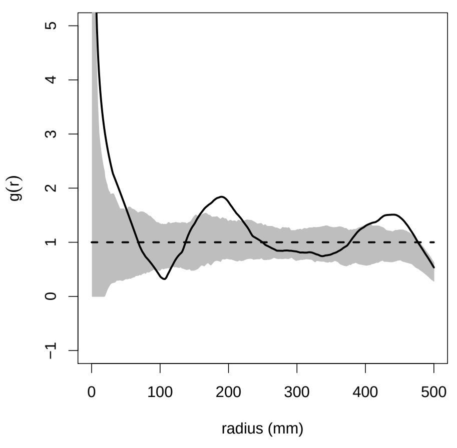


(d) Plot 11

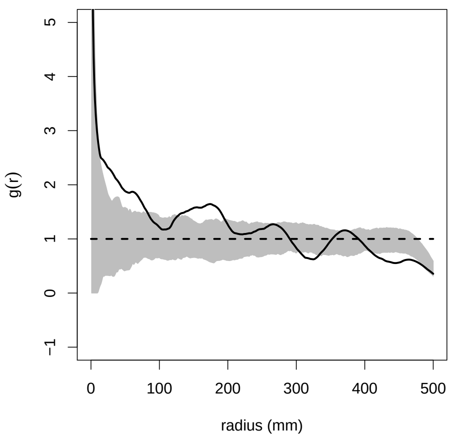
Figure A.29: Pair correlation, $g(r)$, calculated on daughter ramets for plots having an initial ramet arrangement that was clustered. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.



(a) Plot 17

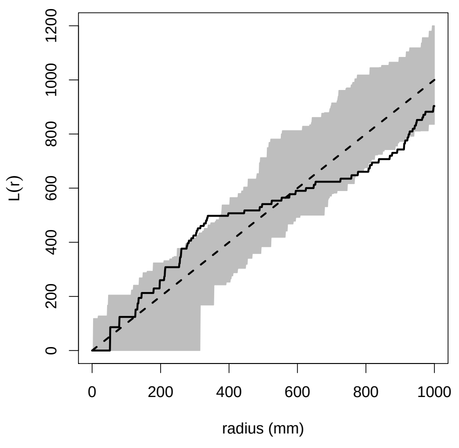


(b) Plot 34

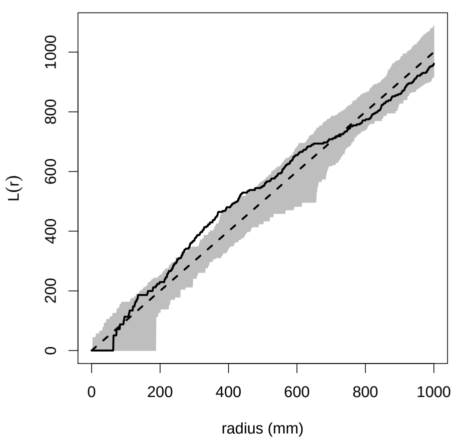


(c) Plot 41

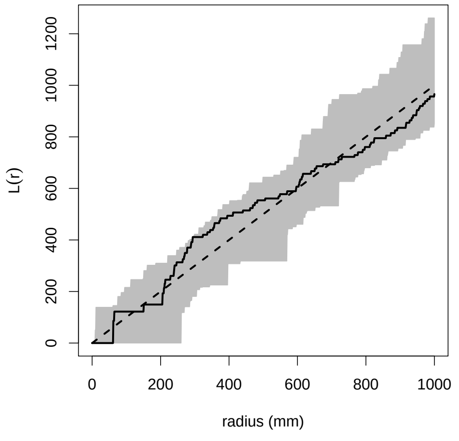
Figure A.30: Pair correlation, $g(r)$, calculated on daughter ramets for plots having an initial ramet arrangement that was clustered. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.



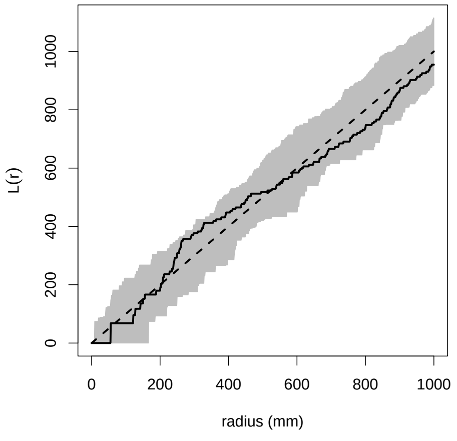
(a) Plot 6



(b) Plot 9

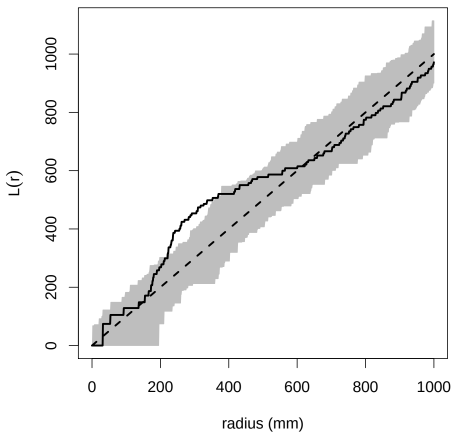


(c) Plot 10

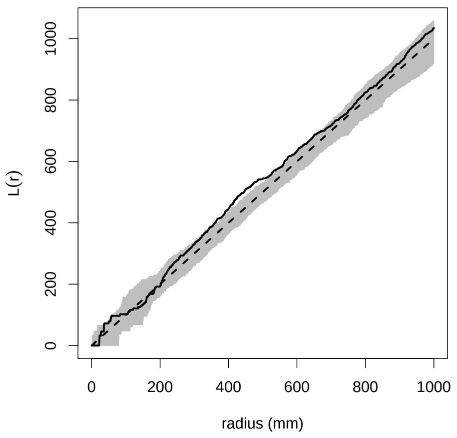


(d) Plot 11

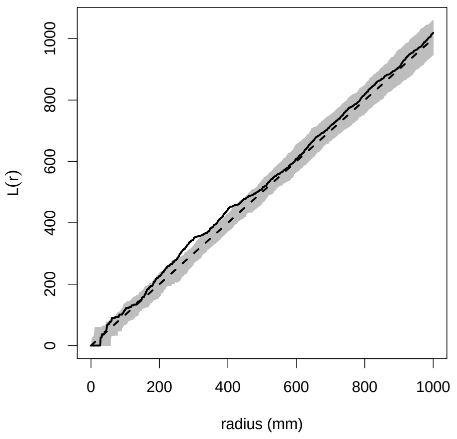
Figure A.31: $L(r)$ calculated on all ramets for plots having an initial ramet arrangement that was clustered. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.



(a) Plot 17

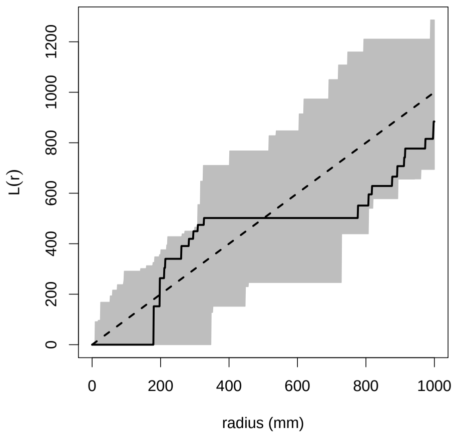


(b) Plot 34

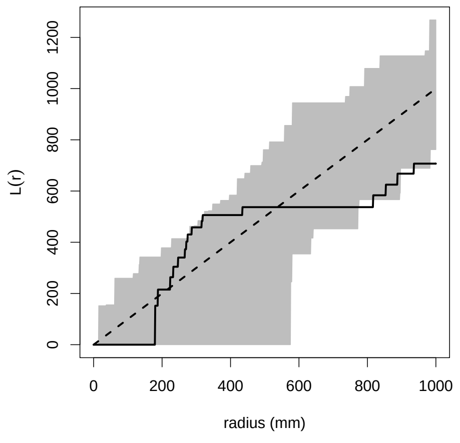


(c) Plot 41

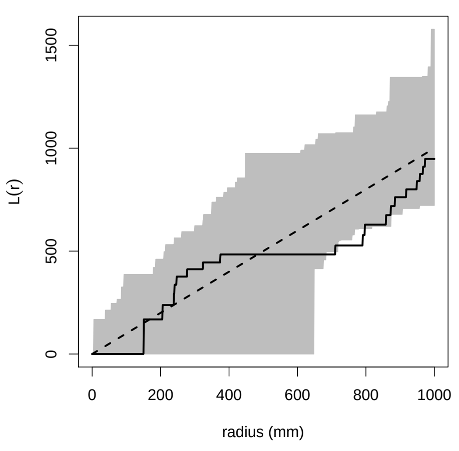
Figure A.32: $L(r)$ calculated on all ramets for plots having an initial ramet arrangement that was clustered. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.



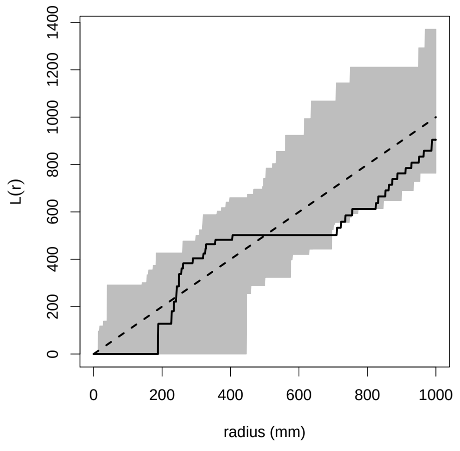
(a) Plot 6



(b) Plot 9

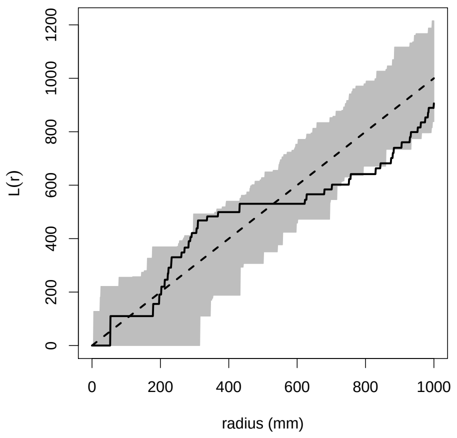


(c) Plot 10

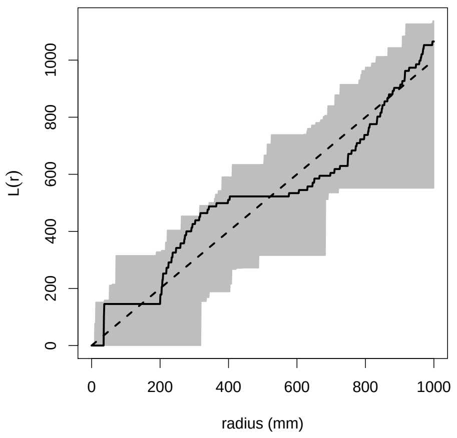


(d) Plot 11

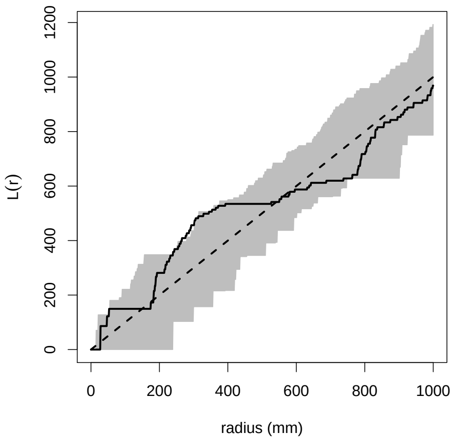
Figure A.33: $L(r)$ calculated on parent ramets for plots having an initial ramet arrangement that was clustered. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.



(a) Plot 17

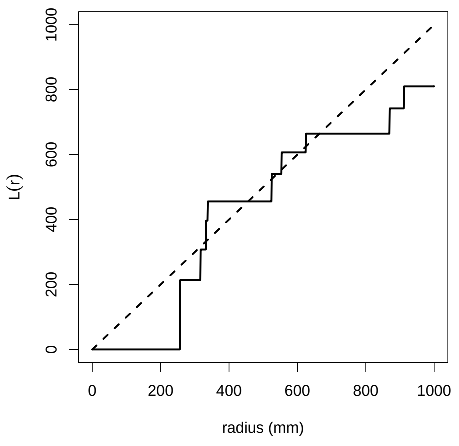


(b) Plot 34

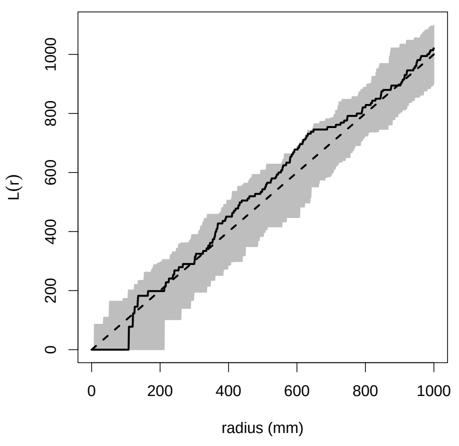


(c) Plot 41

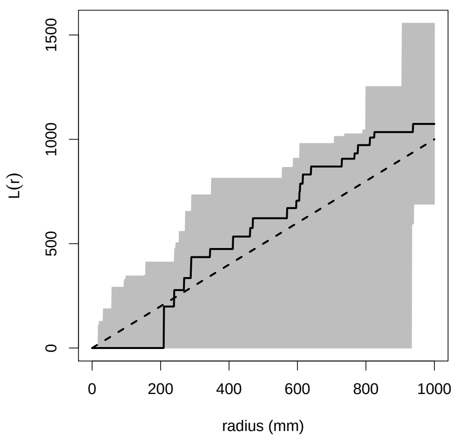
Figure A.34: $L(r)$ calculated on parent ramets for plots having an initial ramet arrangement that was clustered. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.



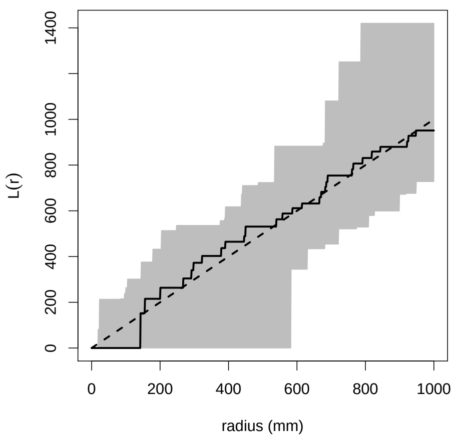
(a) Plot 6



(b) Plot 9

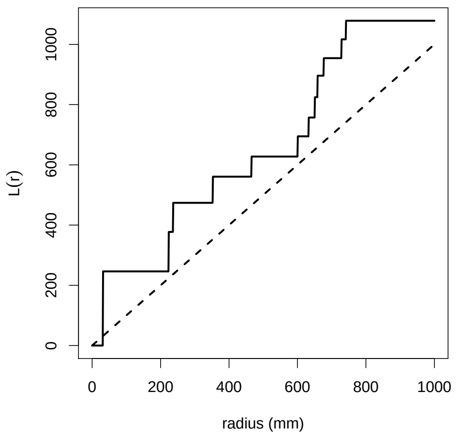


(c) Plot 10

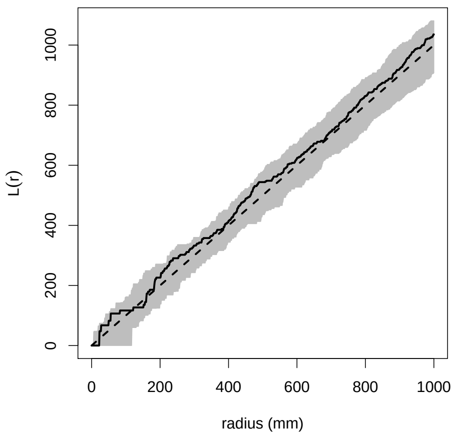


(d) Plot 11

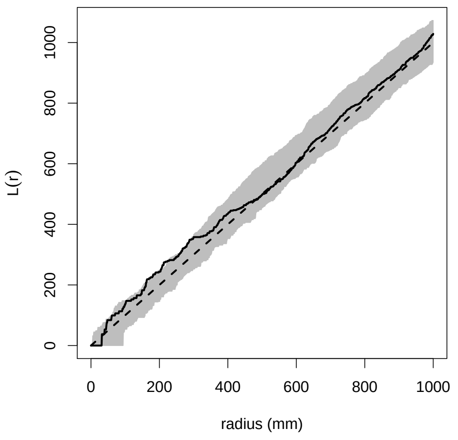
Figure A.35: $L(r)$ calculated on daughter ramets for plots having an initial ramet arrangement that was clustered. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.



(a) Plot 17

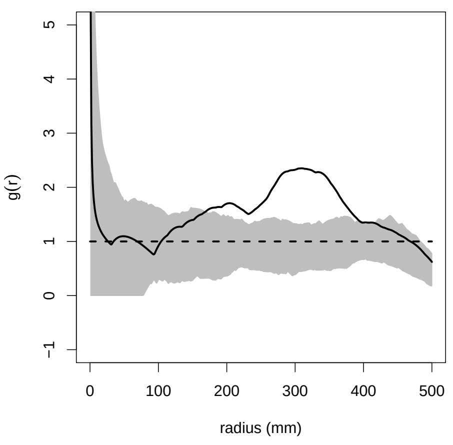


(b) Plot 34

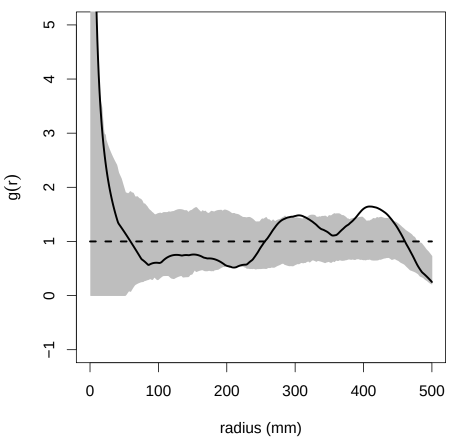


(c) Plot 41

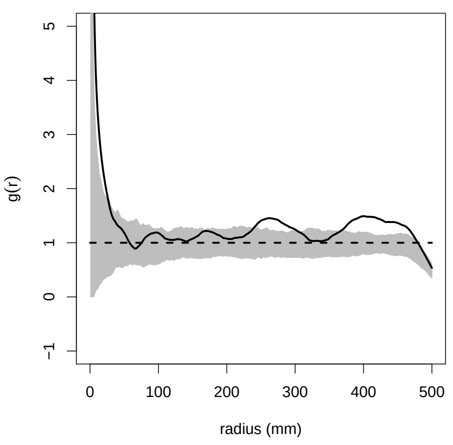
Figure A.36: $L(r)$ calculated on daughter ramets for plots having an initial ramet arrangement that was clustered. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.



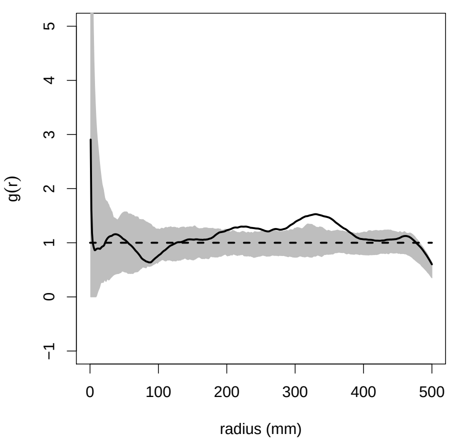
(a) Plot 4



(b) Plot 15

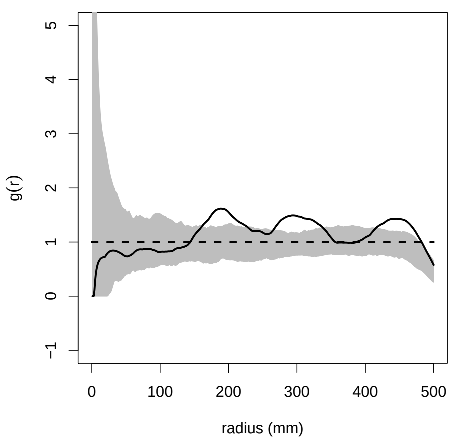


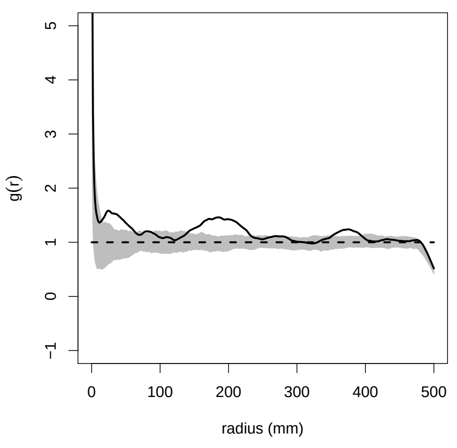
(c) Plot 22

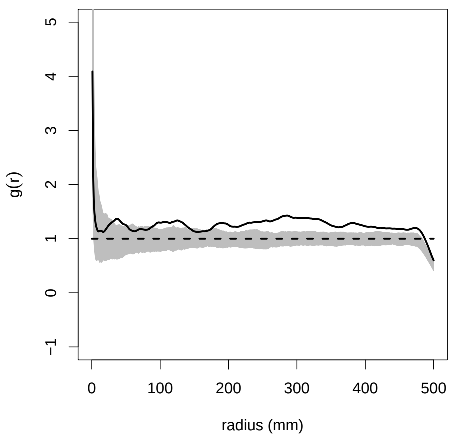


(d) Plot 23

Figure A.37: Pair correlation, $g(r)$, calculated on all ramets (parent and daughter) for plots having an initial ramet arrangement that was regular. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.


(a) Plot 26


(b) Plot 28


(c) Plot 29

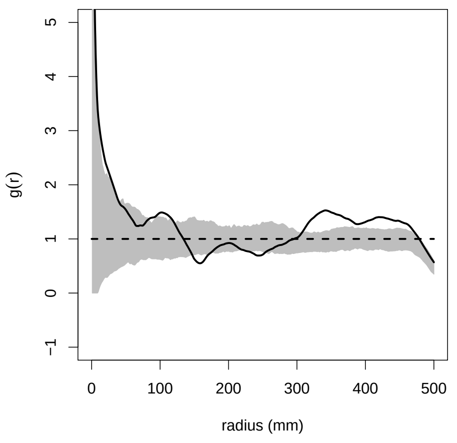
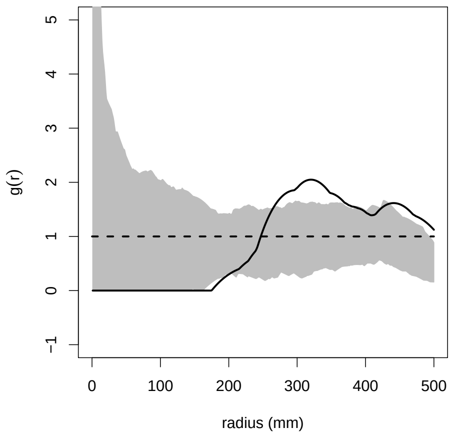
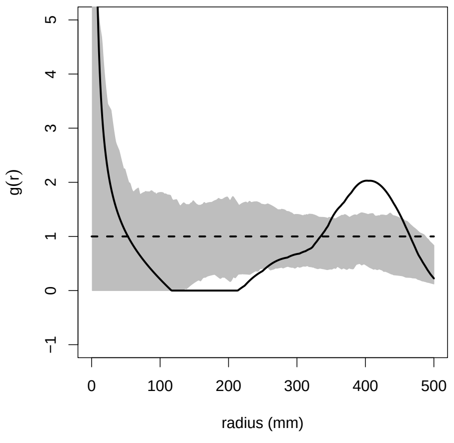

(d) Plot 33

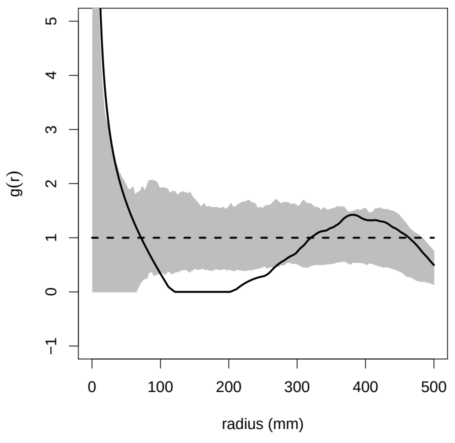
Figure A.38: Pair correlation, $g(r)$, calculated on all ramets (parent and daughter) for plots having an initial ramet arrangement that was regular. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.



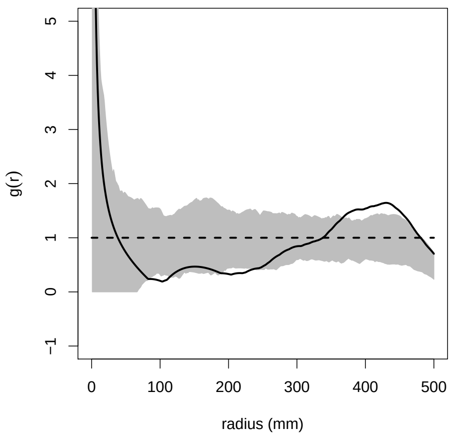
(a) Plot 4



(b) Plot 15

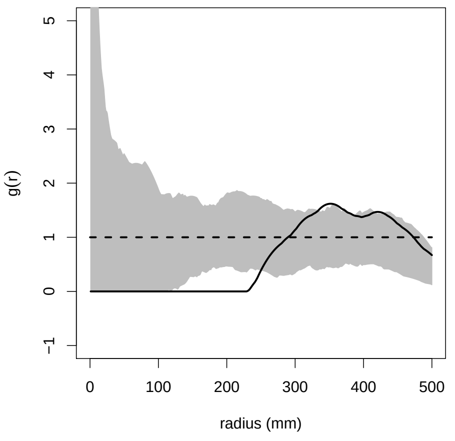


(c) Plot 22

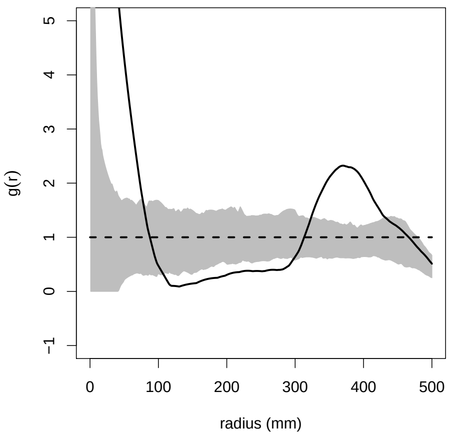


(d) Plot 23

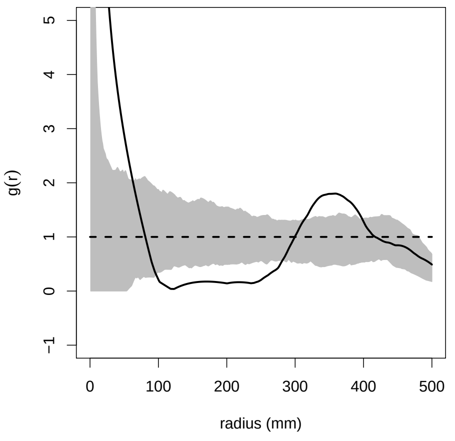
Figure A.39: Pair correlation, $g(r)$, calculated on parent ramets for plots having an initial ramet arrangement that was regular. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.



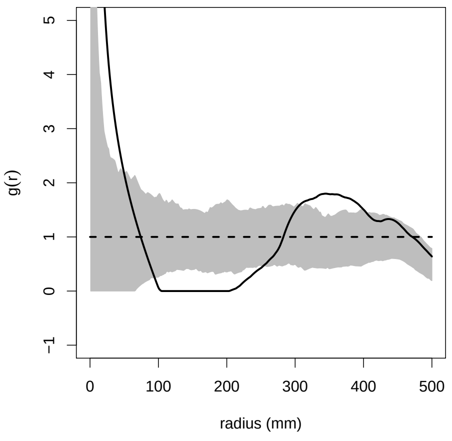
(a) Plot 26



(b) Plot 28

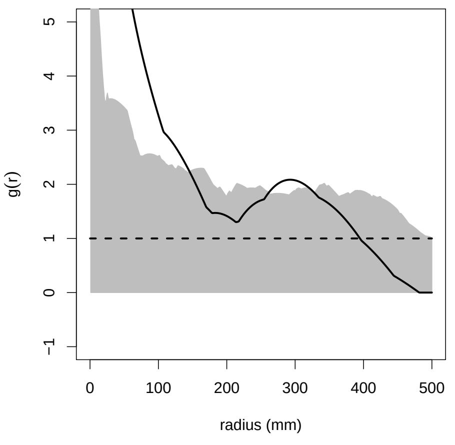


(c) Plot 29

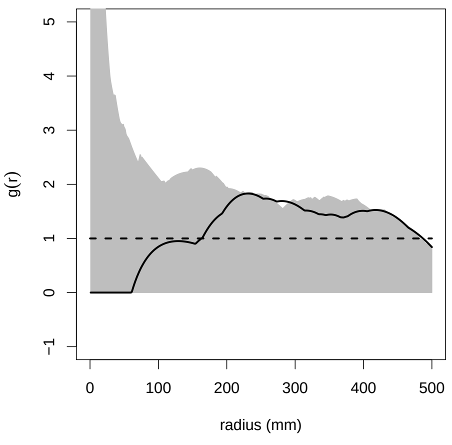


(d) Plot 33

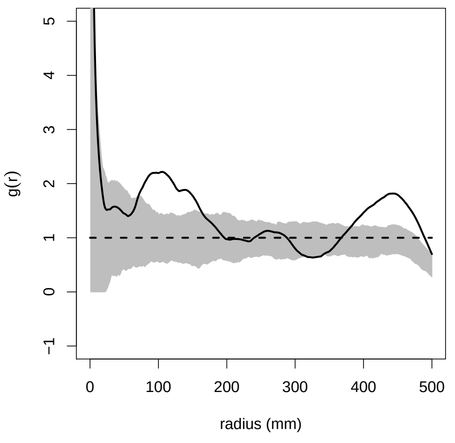
Figure A.40: Pair correlation, $g(r)$, calculated on parent ramets for plots having an initial ramet arrangement that was regular. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.



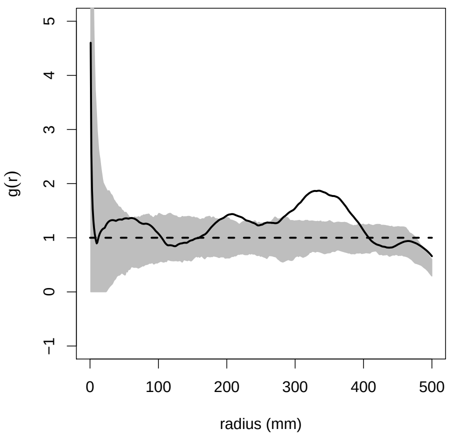
(a) Plot 4



(b) Plot 15

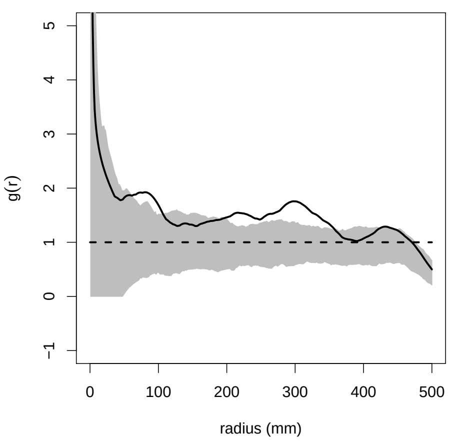


(c) Plot 22

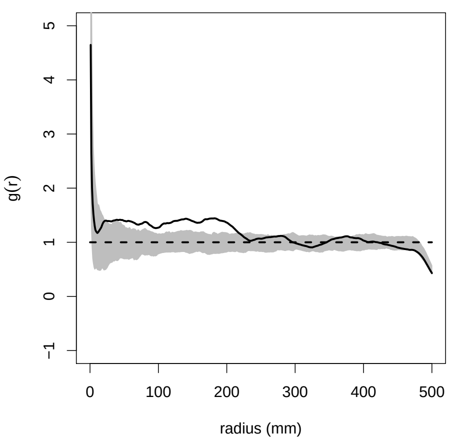


(d) Plot 23

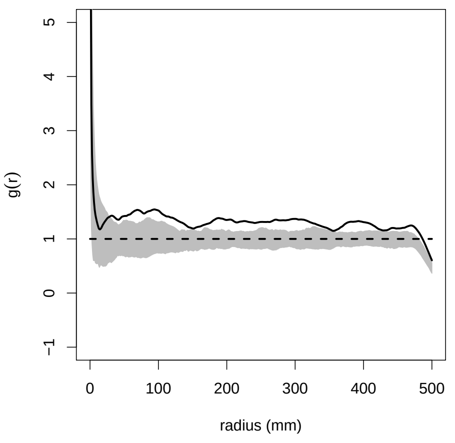
Figure A.41: Pair correlation, $g(r)$, calculated on daughter ramets for plots having an initial ramet arrangement that was regular. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.



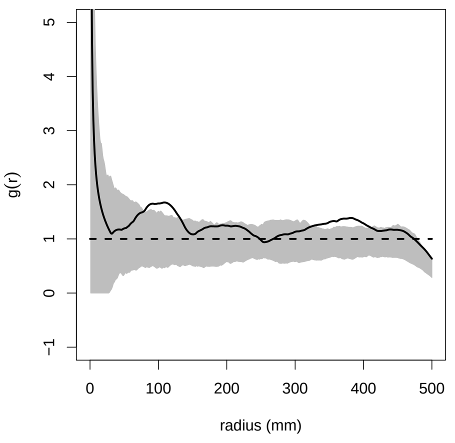
(a) Plot 26



(b) Plot 28

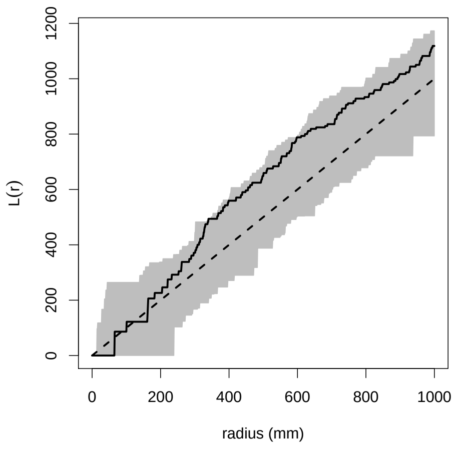


(c) Plot 29

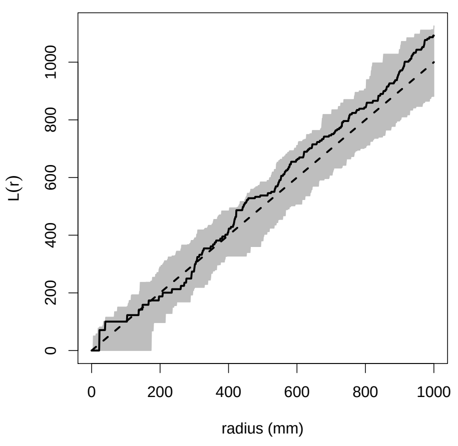


(d) Plot 33

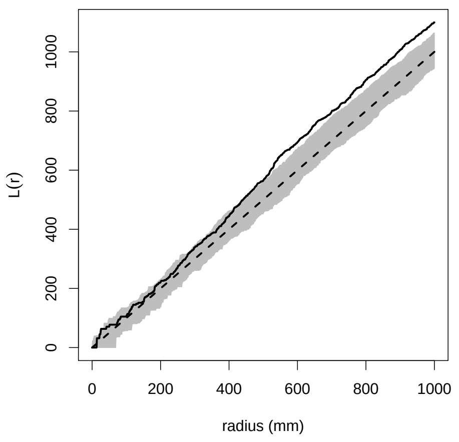
Figure A.42: Pair correlation, $g(r)$, calculated on daughter ramets for plots having an initial ramet arrangement that was regular. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.



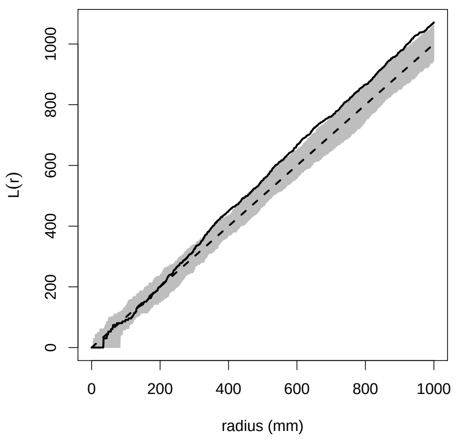
(a) Plot 4



(b) Plot 15

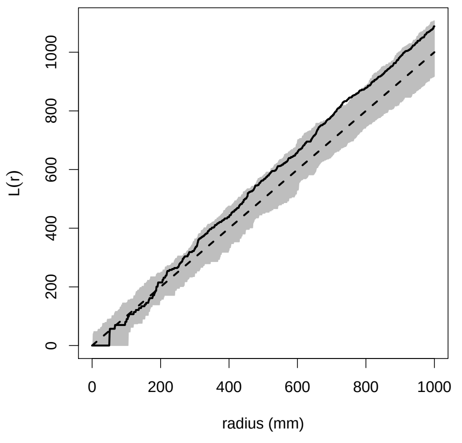


(c) Plot 22

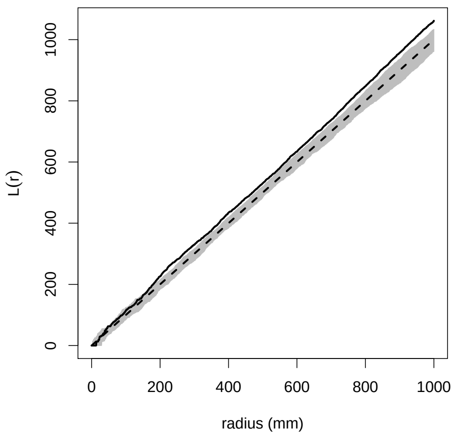


(d) Plot 23

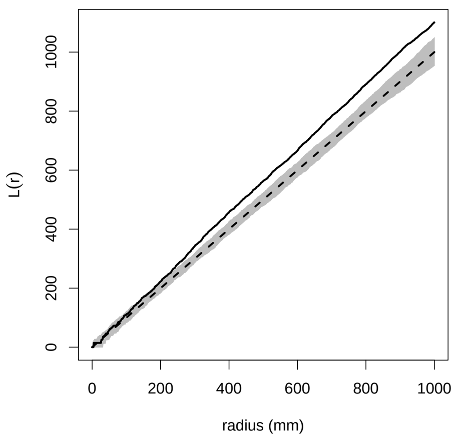
Figure A.43: $L(r)$ calculated on all ramets for plots having an initial ramet arrangement that was regular. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.



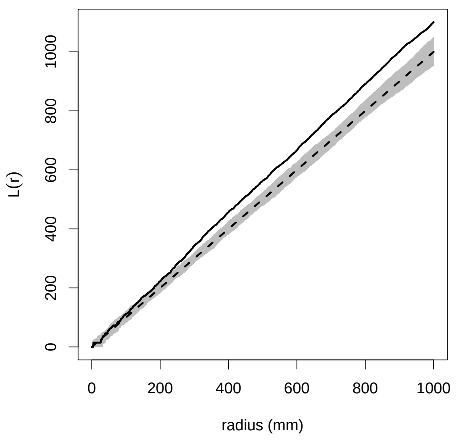
(a) Plot 26



(b) Plot 28

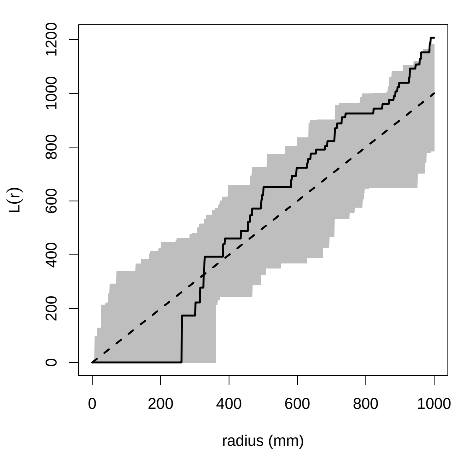


(c) Plot 29

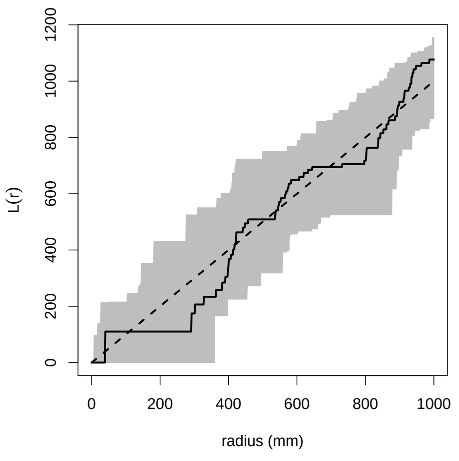


(d) Plot 33

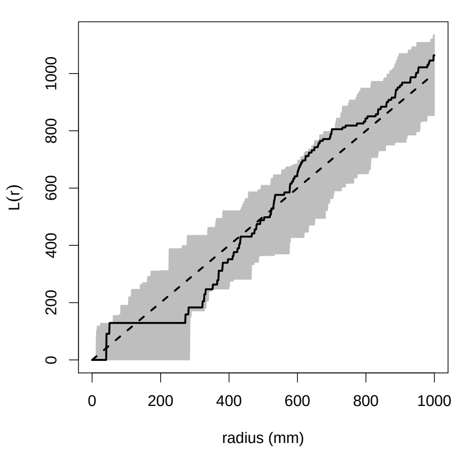
Figure A.44: $L(r)$ calculated on all ramets for plots having an initial ramet arrangement that was regular. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.



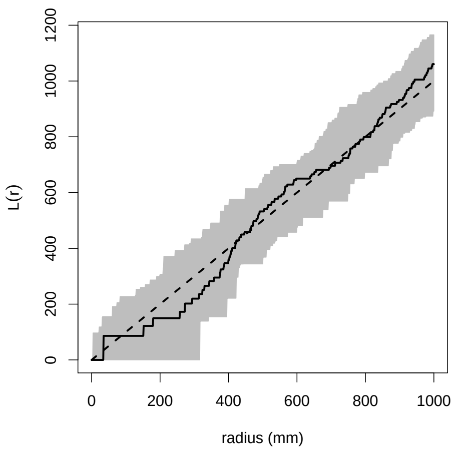
(a) Plot 4



(b) Plot 15

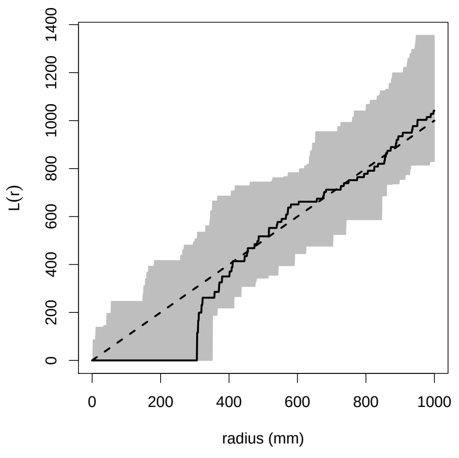


(c) Plot 22

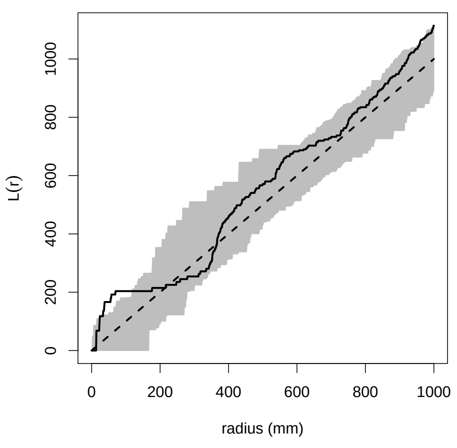


(d) Plot 23

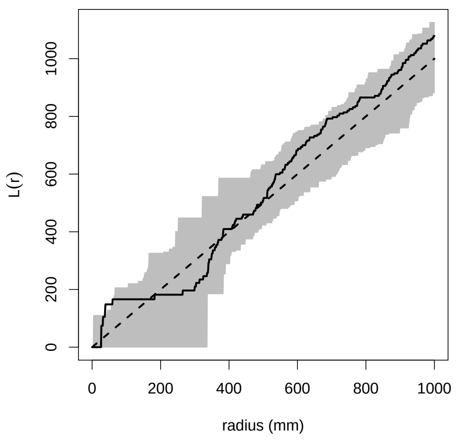
Figure A.45: $L(r)$ calculated on parent ramets for plots having an initial ramet arrangement that was regular. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.



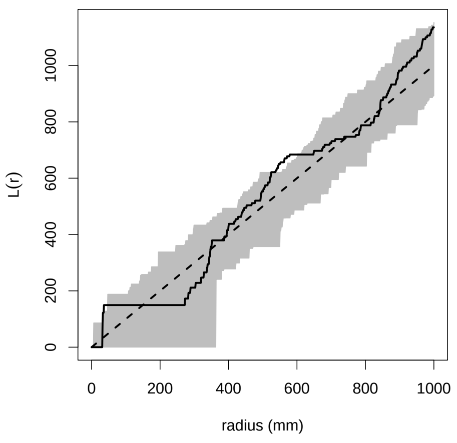
(a) Plot 26



(b) Plot 28

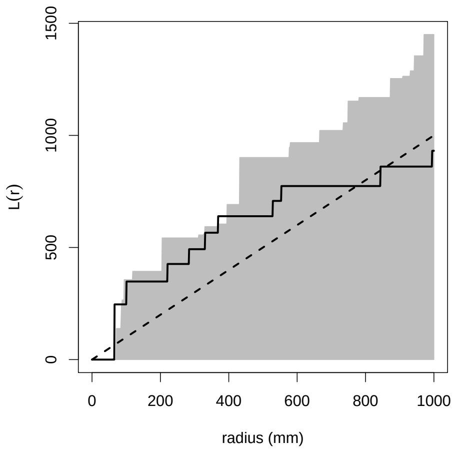


(c) Plot 29

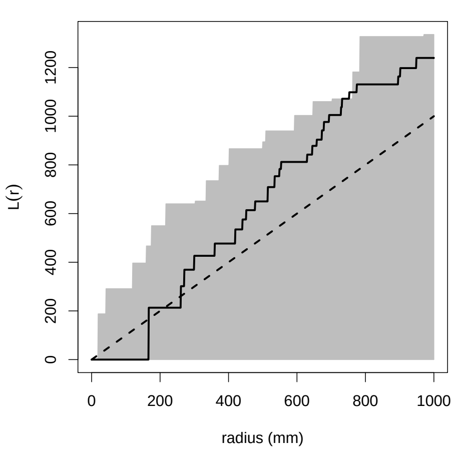


(d) Plot 33

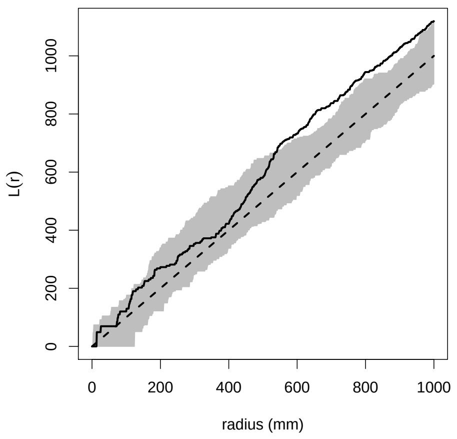
Figure A.46: $L(r)$ calculated on parent ramets for plots having an initial ramet arrangement that was regular. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.



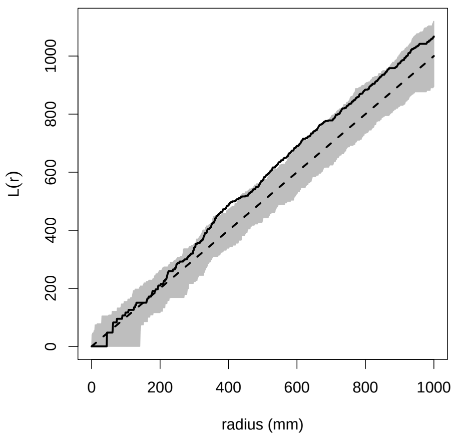
(a) Plot 4



(b) Plot 15

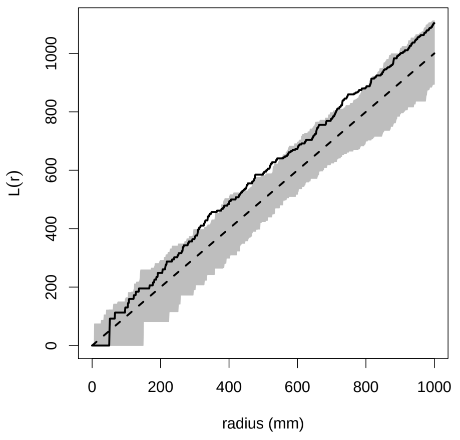


(c) Plot 22

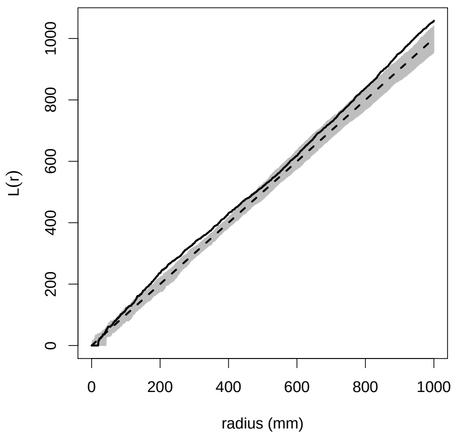


(d) Plot 23

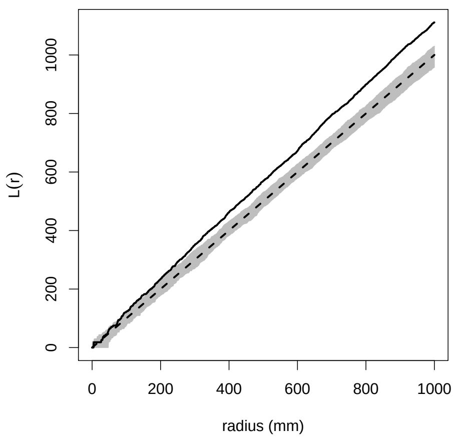
Figure A.47: $L(r)$ calculated on daughter ramets for plots having an initial ramet arrangement that was regular. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.



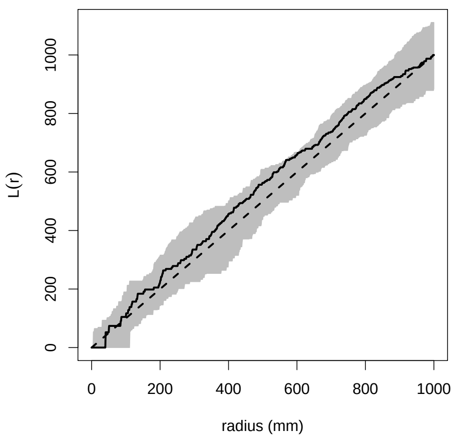
(a) Plot 26



(b) Plot 28



(c) Plot 29



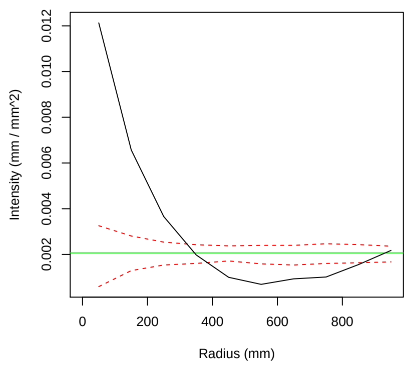
(d) Plot 33

Figure A.48: $L(r)$ calculated on daughter ramets for plots having an initial ramet arrangement that was regular. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.

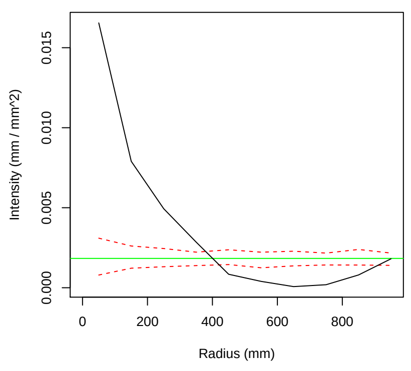
A.2 Chapter 5: Additional Figures

A.2.1 Circle sampling

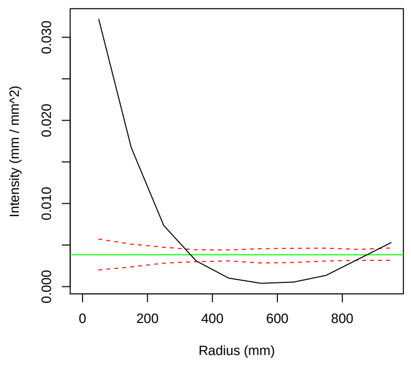
Treatment B1



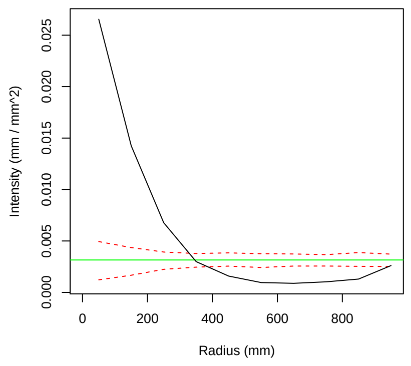
(a) Plot 1



(b) Plot 16



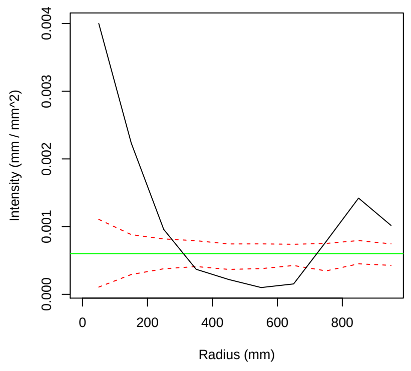
(c) Plot 21



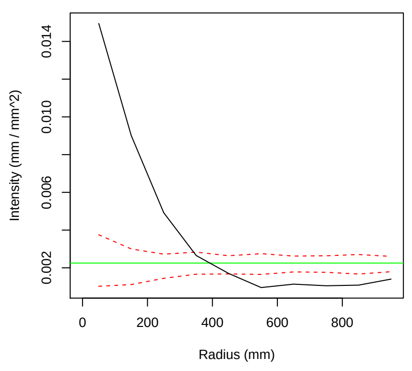
(d) Plot 39

Figure A.49: Circle sampling analysis for B1 treatment plots. Solid line is the observed intensity measured from sampling circles with radius *r* placed randomly on stolons. Dashed lines are the intervals with 80% of the intensity estimates calculated using sampling circles of radius *r* placed randomly over the plot. The horizontal line is the actual intensity of stolons calculated by dividing the total stolon length by plot area.

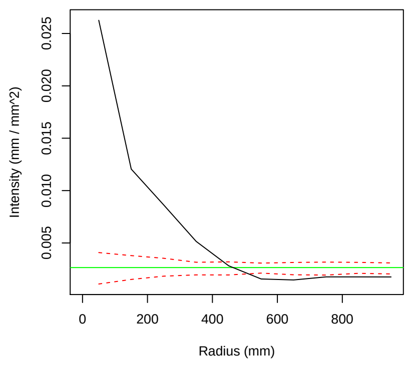
Treatment B2



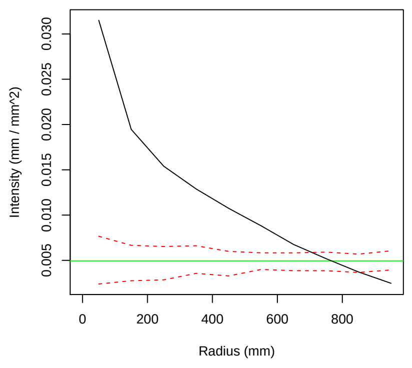
(a) Plot 13



(b) Plot 24



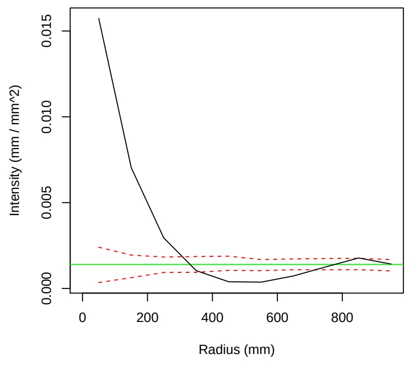
(c) Plot 35



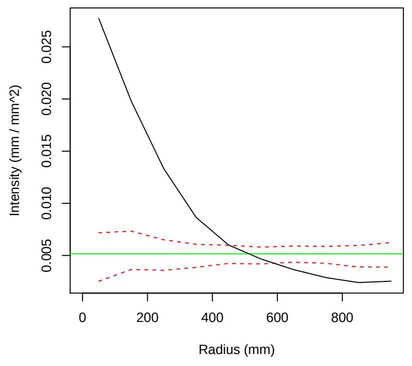
(d) Plot 37

Figure A.50: Circle sampling analysis for B2 treatment plots. Solid line is the observed intensity measured from sampling circles with radius *r* placed randomly on stolons. Dashed lines are the intervals with 80% of the intensity estimates calculated using sampling circles of radius *r* placed randomly over the plot. The horizontal line is the actual intensity of stolons calculated by dividing the total stolon length by plot area.

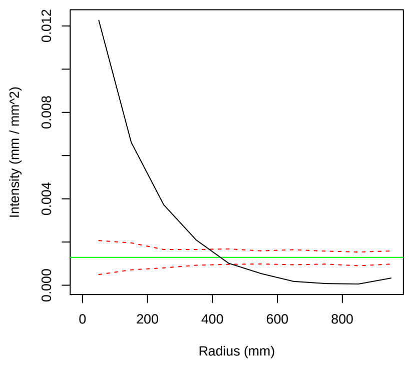
Treatment B3



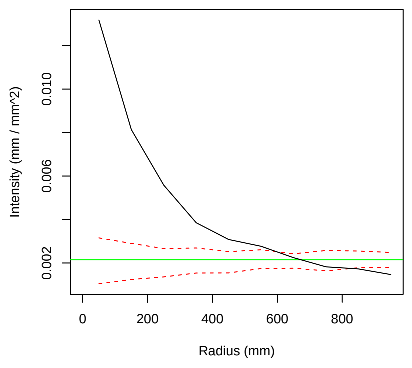
(a) Plot 3



(b) Plot 31



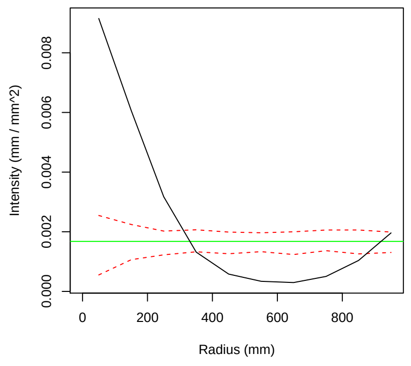
(c) Plot 36



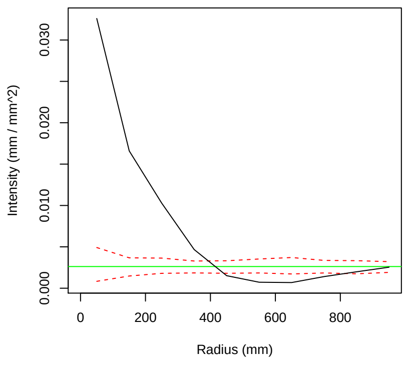
(d) Plot 38

Figure A.51: Circle sampling analysis for B3 treatment plots. Solid line is the observed intensity measured from sampling circles with radius *r* placed randomly on stolons. Dashed lines are the intervals with 80% of the intensity estimates calculated using sampling circles of radius *r* placed randomly over the plot. The horizontal line is the actual intensity of stolons calculated by dividing the total stolon length by plot area.

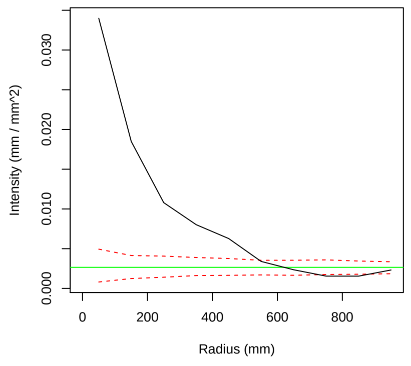
Treatment B4



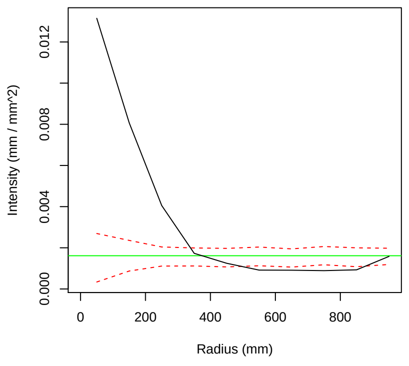
(a) Plot 2



(b) Plot 30



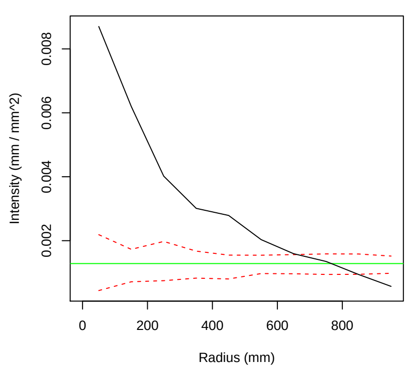
(c) Plot 40



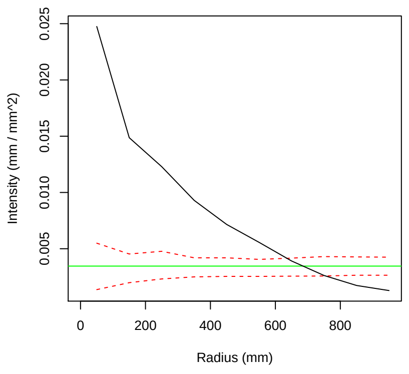
(d) Plot 48

Figure A.52: Circle sampling analysis for B4 treatment plots. Solid line is the observed intensity measured from sampling circles with radius *r* placed randomly on stolons. Dashed lines are the intervals with 80% of the intensity estimates calculated using sampling circles of radius *r* placed randomly over the plot. The horizontal line is the actual intensity of stolons calculated by dividing the total stolon length by plot area.

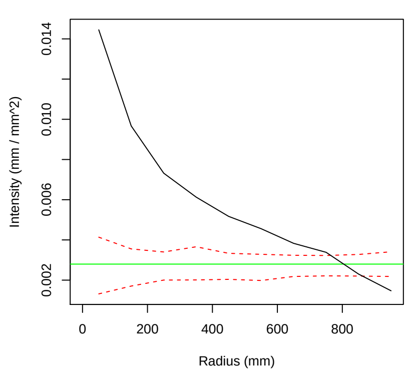
Treatment B5



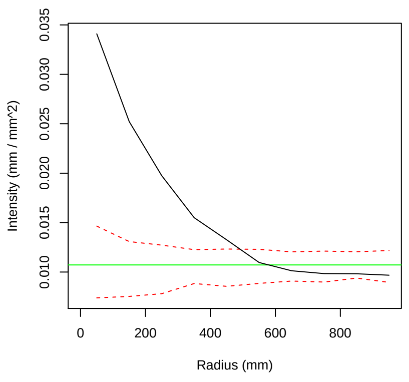
(a) Plot 5



(b) Plot 19



(c) Plot 42

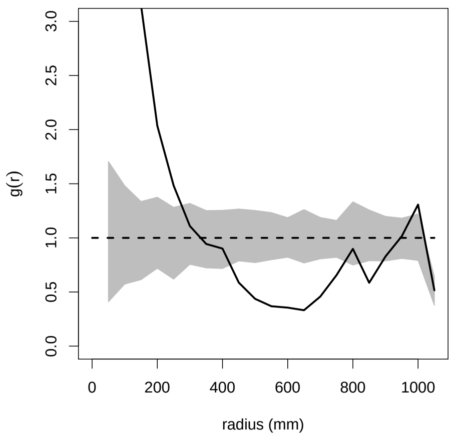


(d) Plot 43

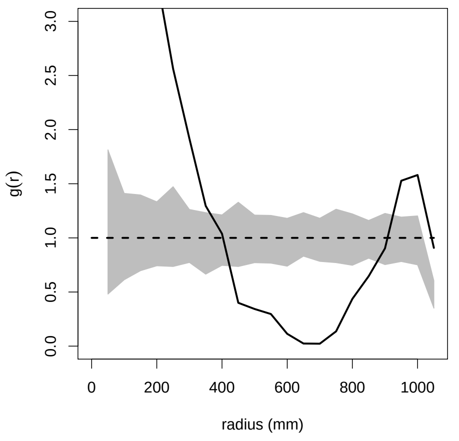
Figure A.53: Circle sampling analysis for B5 treatment plots. Solid line is the observed intensity measured from sampling circles with radius *r* placed randomly on stolons. Dashed lines are the intervals with 80% of the intensity estimates calculated using sampling circles of radius *r* placed randomly over the plot. The horizontal line is the actual intensity of stolons calculated by dividing the total stolon length by plot area.

A.2.2 Fibre Point Analsysis

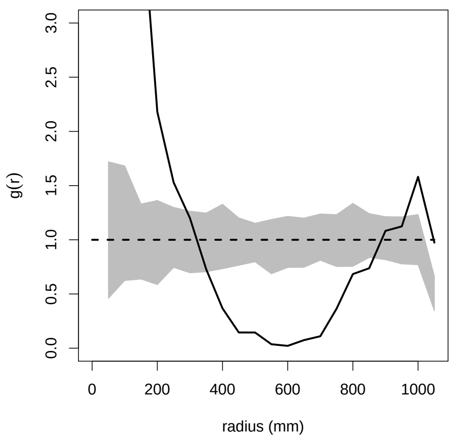
Treatment B1



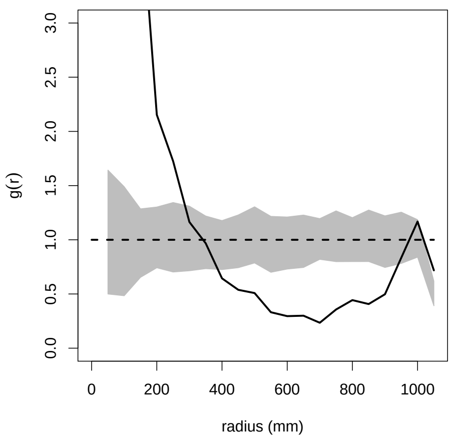
(a) Plot 1



(b) Plot 16



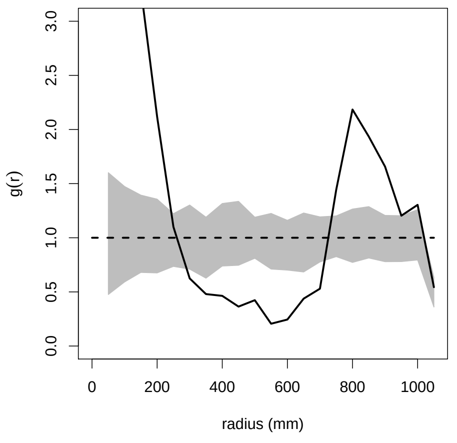
(c) Plot 21



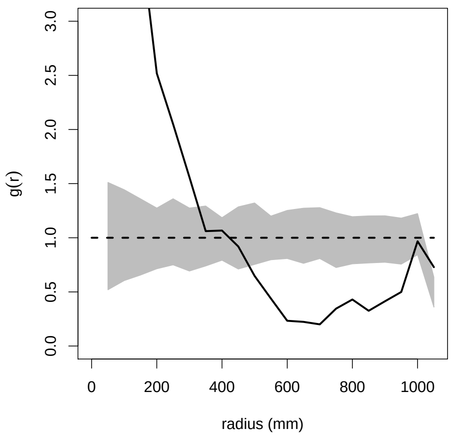
(d) Plot 39

Figure A.54: Pair correlation, $g(r)$, calculated on all random fibre points in plots in the B1 treatment. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.

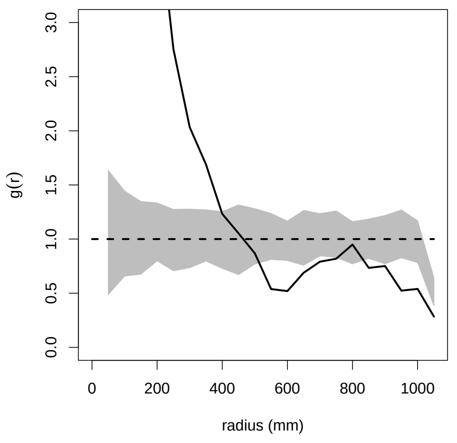
Treatment B2



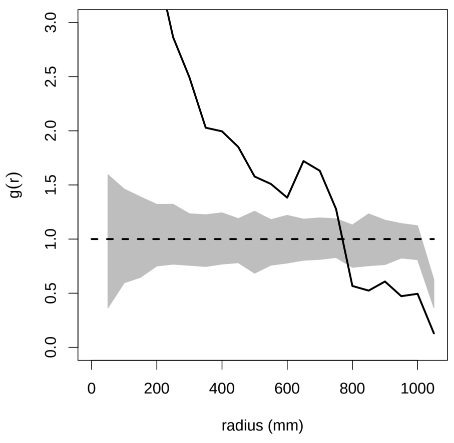
(a) Plot 13



(b) Plot 24



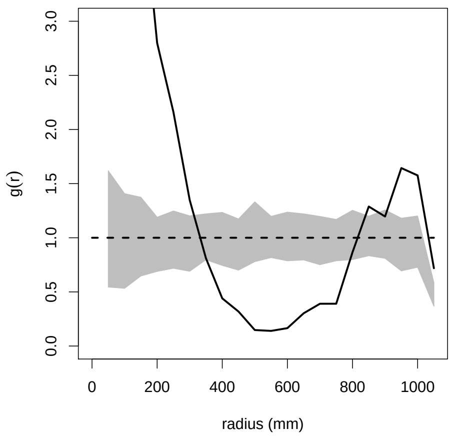
(c) Plot 35



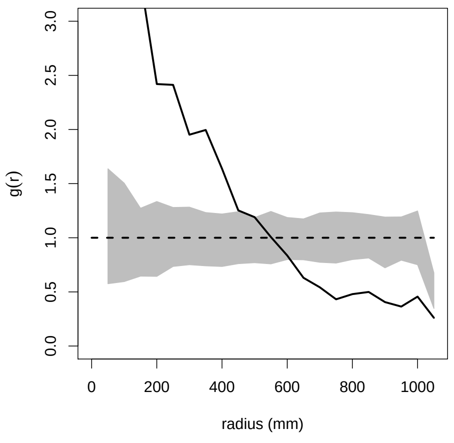
(d) Plot 37

Figure A.55: Pair correlation, $g(r)$, calculated on all random fibre points in plots in the B2 treatment. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.

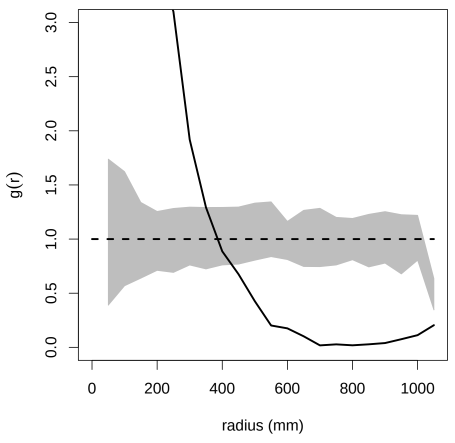
Treatment B3



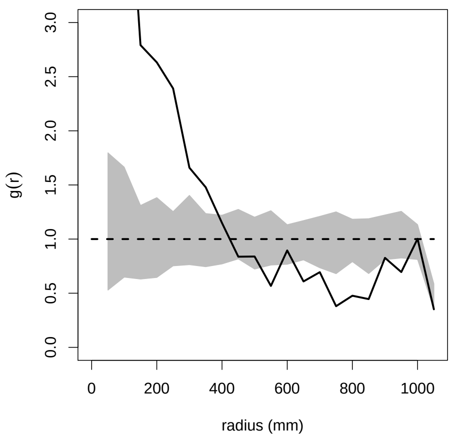
(a) Plot 3



(b) Plot 31



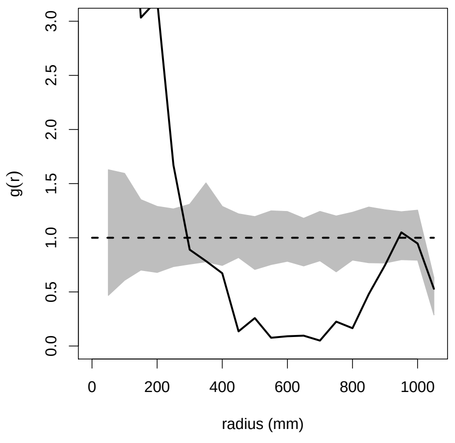
(c) Plot 36



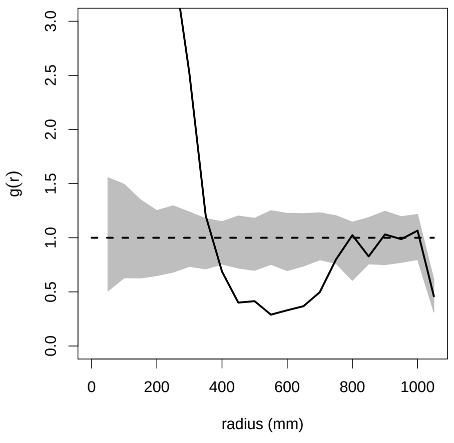
(d) Plot 38

Figure A.56: Pair correlation, $g(r)$, calculated on all random fibre points in plots in the B3 treatment. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.

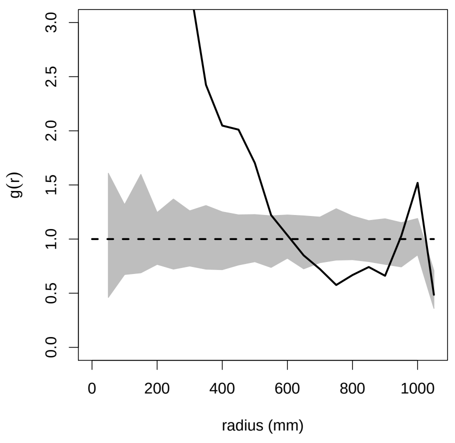
Treatment B4



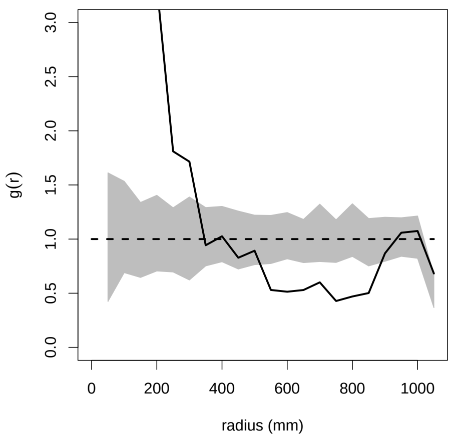
(a) Plot 2



(b) Plot 30



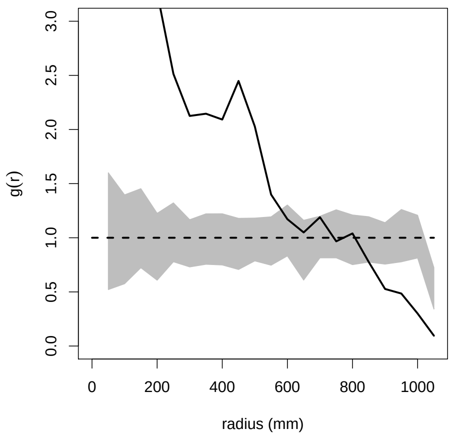
(c) Plot 40



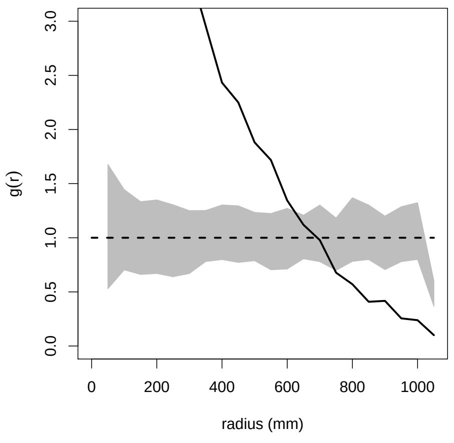
(d) Plot 48

Figure A.57: Pair correlation, *g*(*r*), calculated on all random fibre points in plots in the B4 treatment. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.

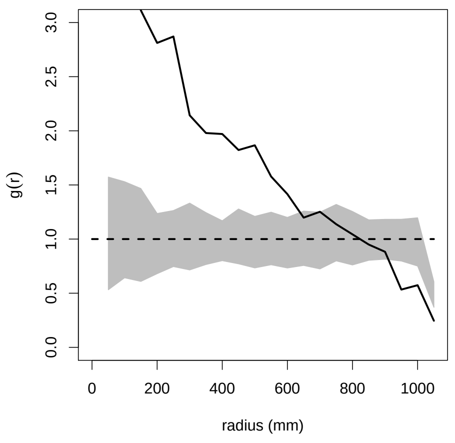
Treatment B5



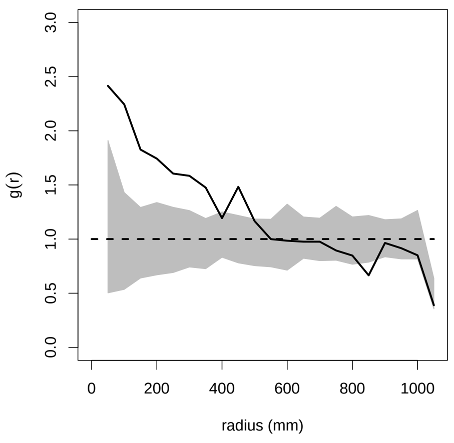
(a) Plot 5



(b) Plot 19



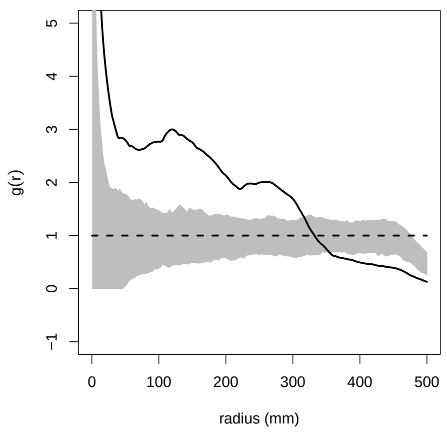
(c) Plot 42



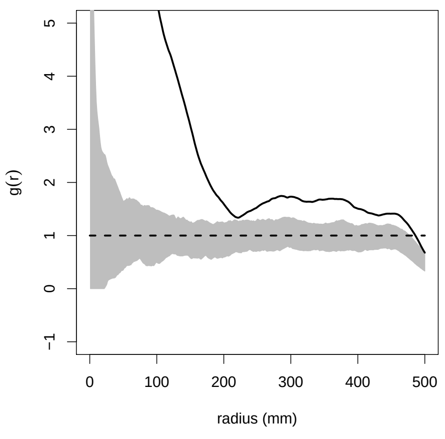
(d) Plot 43

Figure A.58: Pair correlation, $g(r)$, calculated on all random fibre points in plots in the B5 treatment. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.

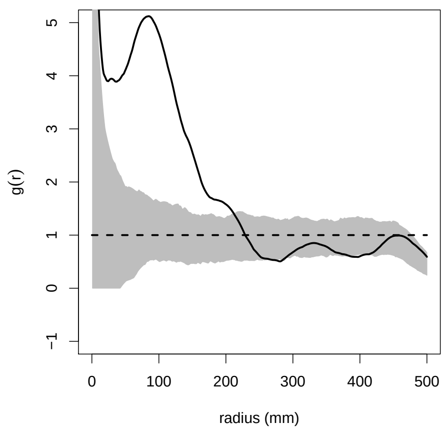
A.2.3 Point patterns of ramets



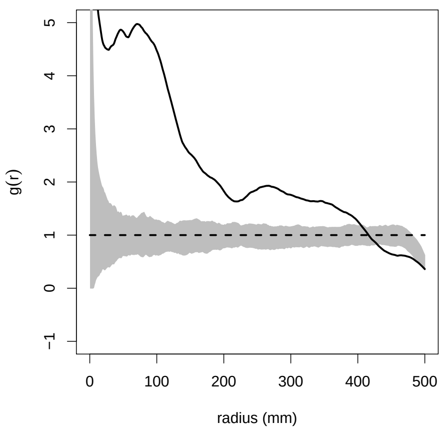
(a) Plot 1



(b) Plot 16



(c) Plot 21



(d) Plot 39

Figure A.59: Pair correlation, $g(r)$, calculated on daughter ramets in the B1 treatment plots. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.

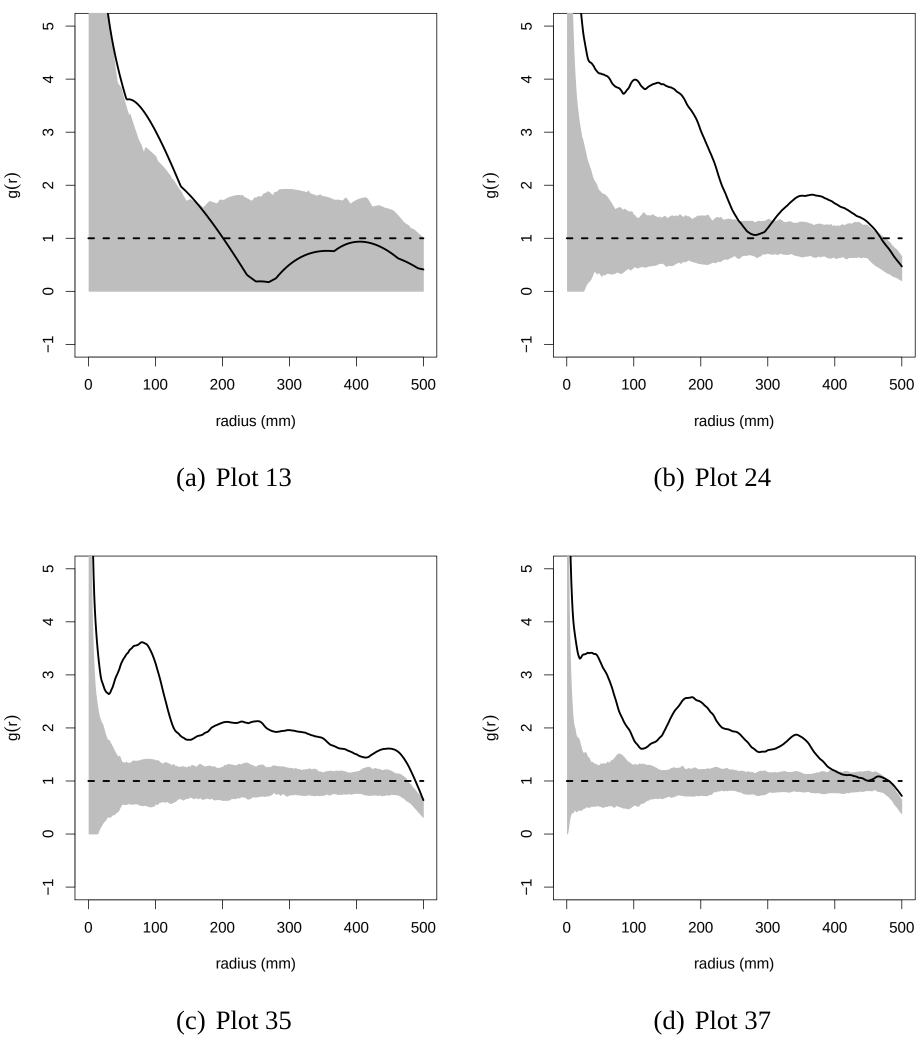
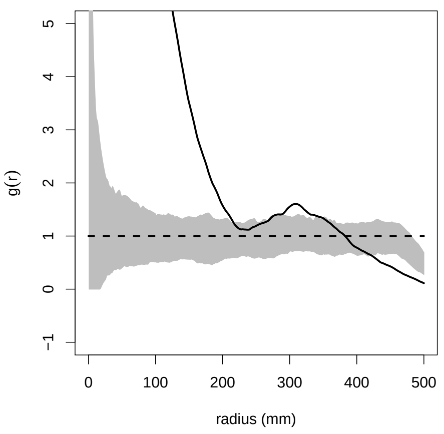
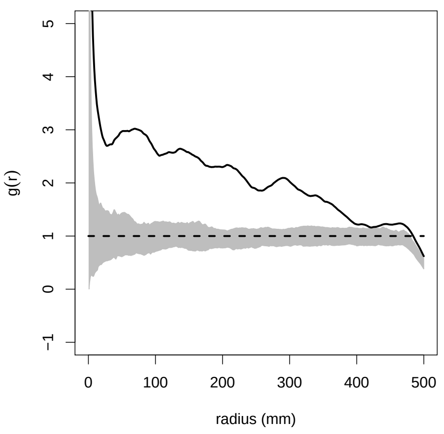


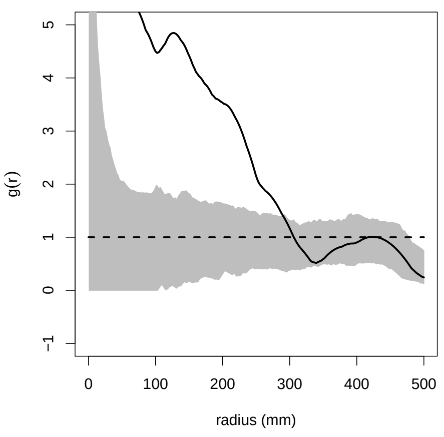
Figure A.60: Pair correlation, $g(r)$, calculated on daughter ramets in the B2 treatment plots. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.



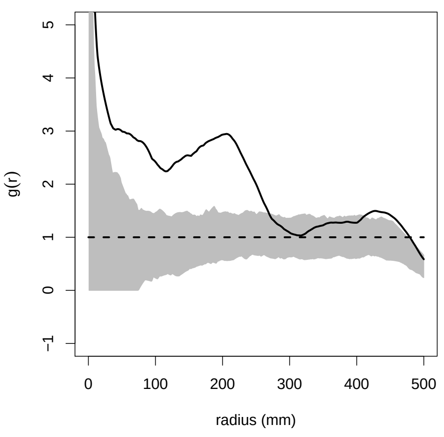
(a) Plot 03



(b) Plot 31

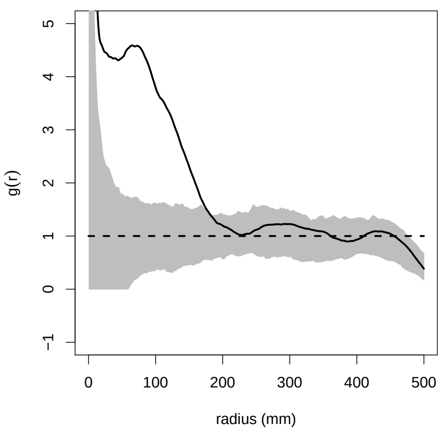


(c) Plot 36

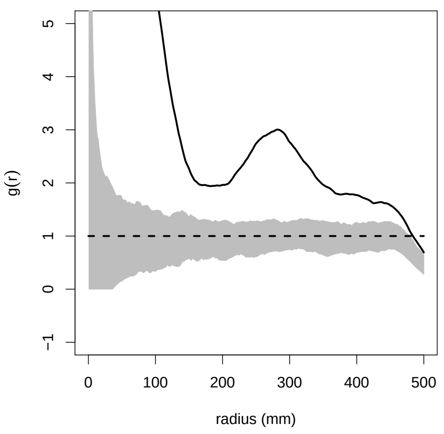


(d) Plot 38

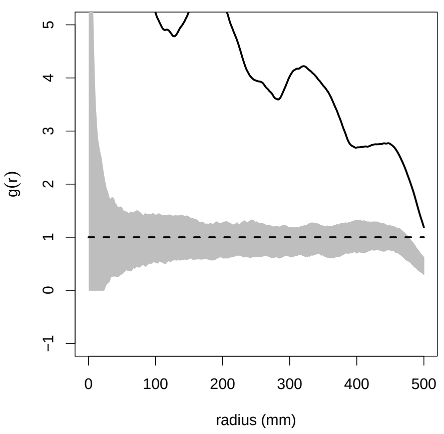
Figure A.61: Pair correlation, $g(r)$, calculated on daughter ramets in the B3 treatment plots. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.



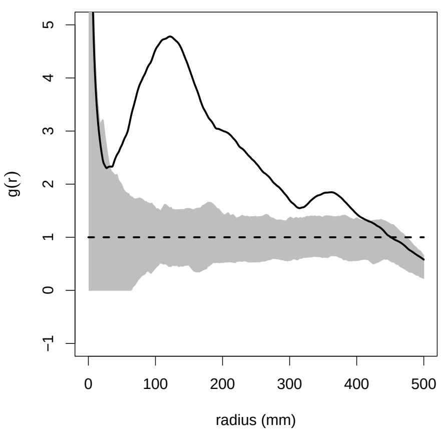
(a) Plot 02



(b) Plot 30

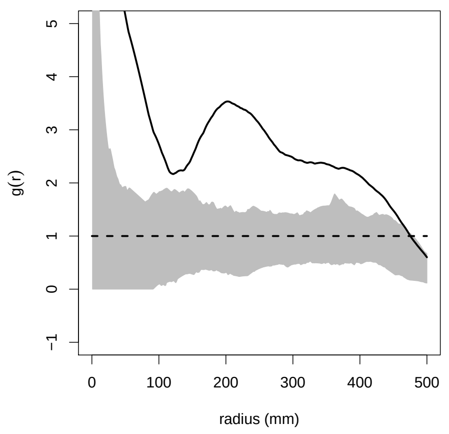


(c) Plot 40

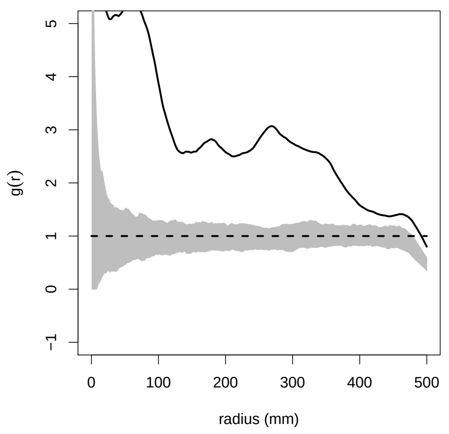


(d) Plot 48

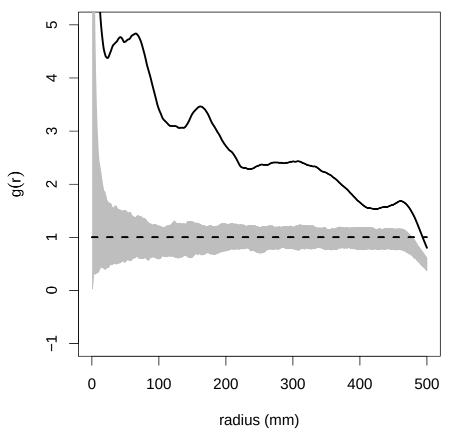
Figure A.62: Pair correlation, $g(r)$, calculated on daughter ramets in the B4 treatment plots. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.



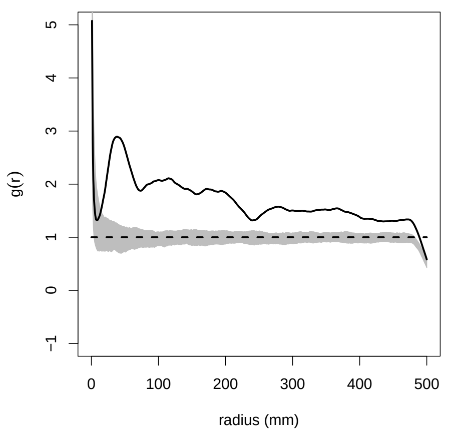
(a) Plot 5



(b) Plot 19



(c) Plot 42



(d) Plot 43

Figure A.63: Pair correlation, $g(r)$, calculated on daughter ramets in the B5 treatment plots. CSR envelopes (grey) are from 99 random simulations of the data and contain 100% of the CSR simulations. Plot size was 2 m x 2 m.

Appendix B

Linear Software Library

B.1 Introduction

The analysis of linear data was conducted in the Python program language using the *Linear* Python library. This library is a collection of Python classes and methods written to allow the analyses of linear data. The library provides basic spatial elements (e.g., circle, segment, point, fibre), simple measures (e.g., length of fibre, orientation of segment), and high level analyses (e.g., circle sampling analysis, randomization of genets). Where required, the software library relies on extension Python libraries (PyX (postscipt drawing), Scypy/Numpy (matrices), and the Python Imaging Library (reading and writing image files)). The R system was used for most of the plots and point pattern analyses, but it is not integrated with the *Linear* package. Given that the nature of ecological field data is likely to vary, this library was designed to provide the researcher as much flexibility as possible to the researcher. On the following pages, a documentation manual is provided that describes the capabilities of the *Linear* software library.

Linear

September 2, 2011

Contents

Chapter 1

Class Index

1.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 3

Class Documentation

3.1 Linear.Analyses.Raster_Analyses.Circle_Sample Class Reference

Public Member Functions

- def __init__

Public Attributes

- **circle_data**
- **total_perimeter_length**
- **L_test_line**
- **n_intersections**
- **I_L**
- **B_A**

Static Public Attributes

- tuple **im_display** = im_in.copy()
- tuple **im_display_write** = im_display.load()
- tuple **circ** = Circle.Circle()
- **coords** = circ.circle_coords
- string **name** = "_N_"

3.1.1 Constructor & Destructor Documentation

3.1.1.1 def Linear.Analyses.Raster_Analyses.Circle_Sample.__init__ (*self, RAW_DATA_FILENAME, im_in, USE_PICKLE_DATA* = `'FALSE'`, *RAW_IMAGE_WIDTH* = `1000`, *RAW_IMAGE_HEIGHT* = `1000`, *BACKGROUND_MIN* = `254`, *BACKGROUND_MAX* = `255`, *BACKGROUND_COL* = `255`, *PIXEL_UNIT_CONVERSION* = `0.01`, *OPEN_RAW_IMAGE* = `'TRUE'`, *CIRC_RADII* = `10`, *CIRC_N* = `1`, *IMAGE_TYPE* = `'L'`, *WRITE_IMAGE* = `'FALSE'`)

```
Initialize an instance of the class Circle_Sample.

   Parameters:

   Returns:
```

The documentation for this class was generated from the following file:

- Raster_Analyses.py

3.2 Linear.Analyses.Raster_Analyses.N_Intersections_Rotate Class Reference

Public Member Functions

- def __init__

Static Public Attributes

- tuple **pickle_filename** = str(RAW_DATA_FILENAME + "_" + str(SCALE_-OF_ANALYSIS) + "_" + "data.pickle")
- tuple **f** = open(pickle_filename, 'w')
- list **data_out** = [boundary_length_estimate_list,boundary_length_intensity_estimate_-list, num_intersections_list, rotations, test_line_set_list]

3.2.1 Constructor & Destructor Documentation

3.2.1.1 def Linear.Analyses.Raster_Analyses.N_Intersections_Rotate.__init__ (*self*, *RAW_DATA_FILENAME*, *SCALE_OF_ANALYSIS*, *USE_PICKLE_DATA* = `'FALSE'`, *CREATE_ROTATED_IMAGES* = `'TRUE'`, *USE_ROTATED_IMAGES* = `'TRUE'`, *RAW_IMAGE_WIDTH* = `1000`, *RAW_IMAGE_HEIGHT* = `1000`, *BACKGROUND_MIN* = `254`, *BACKGROUND_MAX* = `255`, *BACKGROUND_COL* = `255`, *INTERNAL_BORDER_COL* = `0`, *PIXEL_UNIT_CONVERSION* = `0.01`, *PICKLE_DATA* = `'FALSE'`, *OPEN_RAW_IMAGE* = `'TRUE'`, *im_in* = `"NULL"`)

```
Initiate an instance of N_Intersections_Rotate.

Parameters:

Returns:
```

The documentation for this class was generated from the following file:

- Raster_Analyses.py

3.3 Linear.Analyses.Raster_Analyses.Rast_Analysis Class Reference

Public Member Functions

- def __init__
- def Rotate_Image_Intersections
- def Square_Lattice_Sample

3.3.1 Detailed Description

```
Class of type Rast_Analysis. A suite of raster analyses.
```

3.3.2 Constructor & Destructor Documentation

3.3.2.1 def Linear.Analyses.Raster_Analyses.Rast_Analysis.__init__ (*self, RAW_DATA_FILENAME, SCALE_OF_ANALYSIS, USE_PICKLE_DATA =* `'FALSE'`, *CREATE_ROTATED_IMAGES =* `'TRUE'`, *USE_ROTATED_IMAGES =* `'TRUE'`, *RAW_IMAGE_WIDTH =* `1000`, *RAW_IMAGE_HEIGHT =* `1000`, *BACKGROUND_MIN =* `254`, *BACKGROUND_MAX =* `255`, *BACKGROUND_COL =* `255`, *INTERNAL_BORDER_COL =* `0`, *PIXEL_UNIT_CONVERSION =* `0.01`, *PICKLE_DATA =* `'FALSE'`)

```
Initialize an instance of the class Rast_Analysis.

Parameters:

Returns:
```

3.3.3 Member Function Documentation

3.3.3.1 def Linear.Analyses.Raster_Analyses.Rast_Analysis.Rotate_Image_Intersections (*self, RAW_DATA_FILENAME, SCALE_OF_ANALYSIS, USE_PICKLE_DATA =* `'FALSE'`, *CREATE_ROTATED_IMAGES =* `'TRUE'`, *USE_ROTATED_IMAGES =* `'TRUE'`, *RAW_IMAGE_WIDTH =* `1000`, *RAW_IMAGE_HEIGHT =* `1000`, *BACKGROUND_MIN =* `254`, *BACKGROUND_MAX =* `255`, *BACKGROUND_COL =* `255`, *INTERNAL_BORDER_COL =* `0`, *PIXEL_UNIT_CONVERSION =* `0.01`, *PICKLE_DATA =* `'FALSE'`, *ROTATIONS =* `range(0,360`, *OPEN_RAW_IMAGE =* `'TRUE'`, *im_in =* `"NULL"`)

```
Rotate image and find intersections.

Parameters:

Returns:
```

3.3.3.2 def Linear.Analyses.Raster_Analyses.Rast_Analysis.Square_Lattice_Sample (*self, RAW_DATA_FILENAME, SCALE_OF_ANALYSIS, USE_PICKLE_DATA =* `'FALSE'`, *CREATE_ROTATED_IMAGES =* `'TRUE'`, *USE_ROTATED_IMAGES =* `'TRUE'`, *RAW_IMAGE_WIDTH =* `1000`, *RAW_IMAGE_HEIGHT =* `1000`, *BACKGROUND_MIN =* `254`, *BACKGROUND_MAX =* `255`, *BACKGROUND_COL =* `255`, *INTERNAL_BORDER_COL =* `0`, *PIXEL_UNIT_CONVERSION =* `0.01`, *PICKLE_DATA =* `'FALSE'`, *REMOVE_BORDER =* `'FALSE'`, *OPEN_RAW_IMAGE =* `'TRUE'`, *im_in =* `'NULL'`)

```
Sample with a lattice of straight test lines.
```

```
Parameters:

Returns:
```

The documentation for this class was generated from the following file:

- Raster_Analyses.py

3.4 Linear.Analyses.Raster_Analyses.Square_Lattice_Sample Class Reference

Public Member Functions

- def __init__

3.4.1 Constructor & Destructor Documentation

3.4.1.1 def Linear.Analyses.Raster_Analyses.Square_Lattice_Sample.__init__ (*self*, *RAW_DATA_FILENAME*, *SCALE_OF_ANALYSIS*, *USE_PICKLE_DATA* = 'FALSE', *CREATE_ROTATED_IMAGES* = 'TRUE', *USE_ROTATED_IMAGES* = 'TRUE', *RAW_IMAGE_WIDTH* = 1000, *RAW_IMAGE_HEIGHT* = 1000, *BACKGROUND_MIN* = 254, *BACKGROUND_MAX* = 255, *BACKGROUND_COL* = 255, *INTERNAL_BORDER_COL* = 0, *PIXEL_UNIT_CONVERSION* = 0.01)

```
Initialize an instance of the class Square_Lattice_Sample.

Parameters:

Returns:
```

The documentation for this class was generated from the following file:

- Raster_Analyses.py

3.5 Linear.Analyses.Vector_Analyses_v2.Line_based_intensity_circles Class Reference

Public Member Functions

- def line_based_intensity_circles
- def line_based_intensity_circles_sim_rand
- def line_based_intensity_circles_genet_exclude_self
- def line_based_intensity_circles_seg

Static Public Attributes

- tuple **rand_proportions_x** = scipy.rand(N_RAND_PTS)
- tuple **rand_proportions_y** = scipy.rand(N_RAND_PTS)
- list **x_range** = x_limit[1]
- list **y_range** = y_limit[1]
- tuple **x_range_prop** = rand_proportions_x∗(x_range + 2∗radius)
- tuple **y_range_prop** = rand_proportions_y∗(y_range + 2∗radius)
- tuple **x_pts** = x_range_prop+(x_limit[0]-radius)
- tuple **y_pts** = y_range_prop+(y_limit[0]-radius)
- list **n_points_corrected_list** = [ ]
- list **l_circle_perimeter_corrected_list** = [ ]
- list **circ_cen_x** = x_pts[i]
- list **circ_cen_y** = y_pts[i]
- tuple **rad** = (radius/reduction)
- tuple **center_location** = Linear.Object.Point.Point(circ_cen_x,circ_cen_y)
- tuple **circle** = Linear.Object.Circle.Circle()
- **w** = circle.prop_in_rect
- float **perimeter** = 2.0
- **l_circle_perimeter_corrected** = perimeter∗w
- list **intersection_points** = [ ]
- list **pt_x1** = PATTERN.list_of_fibres[m]
- list **pt_y1** = PATTERN.list_of_fibres[m]
- list **pt_x2** = PATTERN.list_of_fibres[m]
- list **pt_y2** = PATTERN.list_of_fibres[m]
- tuple **intersection_points_tmp** = (Linear.Geometry.geom.circle_segment_intersection_-points(pt_x1, pt_y1, pt_x2, pt_y2, circ_cen_x, circ_cen_y, radius))
- tuple **n_points** = float(len(intersection_points))
- tuple **n_points_corrected** = n_points∗(1.0/w)
- **n_points_corrected** = n_points
- tuple **leng** = scipy.sum(l_circle_perimeter_corrected_list)
- tuple **n_pt** = scipy.sum(n_points_corrected_list)

- **I_L** = n_pt/leng
- float **B_A** = 2.0
- tuple **f** = open(filename, 'a')

3.5.1 Member Function Documentation

3.5.1.1 def Linear.Analyses.Vector_Analyses_v2.Line_based_intensity_circles.line_based_intensity_circles (*self*, *x_limit*, *y_limit*, *RADIUS*, *N_RAND_PTS*, *n_perimeter_points*, *INTER_POINT_EDGE_CORRECT*, *filename_prefix*, *PATTERN*)

```
Find the intensity of lines using circles placed randomly over
the linear data. Repeat the process for a range of circle radii.

Points are distributed over the fibres.

Parameters
x_limit : list
    Limits of the plot, [x minimum, x maximum]
y_limit : list
    limits of the plot, [y minimum, y maximum]
RADIUS : list
    List with the range of circle radii to use
N_RAND_PTS : int
    The number of random points to distribute over the linear data.
n_perimeter_points
INTER_POINT_EDGE_CORRECT
filename_prefix
PATTERN
    Object of type Pattern where linear data is stored in a
    list of fibres.
GENET_PROC_OBS
WRITE_DATA

Returns
filename_prefix+"_stat_line_based_intensity.txt" : output file
    Output file with intensity estimates for each circle radius
filename_prefix+"radius_" + repr(r)+"_segments.txt" : output file
    Output file with fibre segments
filename_prefix+"radius_" + repr(r)+"_points.txt" : output file
    Output file with ramet points
filename_prefix+"radius_" + repr(r)+"_circles.txt" : output file
    Output file with sampling circles
```

3.5.1.2 def Linear.Analyses.Vector_Analyses_v2.Line_based_intensity_circles.line_based_intensity_circles_sim_rand (*self*, *x_limit*, *y_limit*, *RADIUS*, *N_RAND_PTS*, *NSIM*, *PLOT_SIM_DATA*, *N_PLOT_SIM_DATA*, *n_perimeter_points*, *INTER_POINT_EDGE_CORRECT*, *filename_prefix*, *PATTERN*)

```
Simulation: Estimate the intensity of lines using circles placed
randomly over the plot.
```

```
Distribute N_RAND_PTS randomly over the plot. Estimate the B_A
(boundary length per unit area) based on the count of intersection
points. Repeat NSIM times.

Points are distributed randomly over the plot.

Parameters
x_limit : list
    Limits of the plot, [x minimum, x maximum]
y_limit : list
    limits of the plot, [y minimum, y maximum]
RADIUS : list
    List with the range of circle radii to use
N_RAND_PTS : int
    The number of random points to distribute over the linear data.
n_perimeter_points
INTER_POINT_EDGE_CORRECT
filename_prefix
PATTERN
    Object of type Pattern where linear data is stored in a
    list of fibres.
GENET_PROC_OBS
WRITE_DATA

Returns
filename_prefix+"_stat_line_based_intensity.txt" : output file
    Output file with intensity estimates for each circle radius
filename_prefix+"radius_" + repr(r)+"_segments.txt" : output file
    Output file with fibre segments
filename_prefix+"radius_" + repr(r)+"_points.txt" : output file
    Output file with ramet points
filename_prefix+"radius_" + repr(r)+"_circles.txt" : output file
    Output file with sampling circles
```

3.5.1.3 def Linear.Analyses.Vector_Analyses_v2.Line_based_intensity_circles.line_based_intensity_circles_genet_exclude_self (*self, x_limit, y_limit, RADIUS, N_RAND_PTS, n_perimeter_points, INTER_POINT_EDGE_CORRECT, filename_prefix, PATTERN, GENET_PROC_OBS, WRITE_DATA*)

```
Find the intensity of lines using circles placed randomly over
the linear data. Repeat the process for a range of circle radii.

Do not collect points from lines on the same genet as the point
centre of the sampling circle.

Parameters
x_limit : list
    Limits of the plot, [x minimum, x maximum]
y_limit : list
    limits of the plot, [y minimum, y maximum]
RADIUS : list
    List with the range of circle radii to use
N_RAND_PTS : int
    The number of random points to distribute over the linear data.
```

```
n_perimeter_points
INTER_POINT_EDGE_CORRECT
filename_prefix
PATTERN
GENET_PROC_OBS
WRITE_DATA

Returns
filename_prefix+"_stat_line_based_intensity.txt" : output file
    Output file with intensity estimates for each circle radius
filename_prefix+"radius_" + repr(r)+"_segments.txt" : output file
    Output file with fibre segments
filename_prefix+"radius_" + repr(r)+"_points.txt" : output file
    Output file with ramet points
filename_prefix+"radius_" + repr(r)+"_circles.txt" : output file
    Output file with sampling circles
```

3.5.1.4 def Linear.Analyses.Vector_Analyses_v2.Line_based_intensity_circles.line_-based_intensity_circles_seg (*self, x_limit, y_limit, RADIUS, N_RAND_PTS, n_perimeter_points, INTER_POINT_EDGE_CORRECT, filename_prefix, SEG_PROC*)

```
Find the intensity of lines using circles placed randomly over
the linear data. Repeat the process for a range of circle radii.

Do not collect points from lines on the same genet as the point
centre of the sampling circle.

Parameters
x_limit : list
    Limits of the plot, [x minimum, x maximum]
y_limit : list
    limits of the plot, [y minimum, y maximum]
RADIUS : list
    List with the range of circle radii to use
N_RAND_PTS : int
    The number of random points to distribute over the linear data.
n_perimeter_points
INTER_POINT_EDGE_CORRECT
filename_prefix
PATTERN
GENET_PROC_OBS
WRITE_DATA

Returns
filename_prefix+"_stat_line_based_intensity.txt" : output file
    Output file with intensity estimates for each circle radius
filename_prefix+"radius_" + repr(r)+"_segments.txt" : output file
    Output file with fibre segments
filename_prefix+"radius_" + repr(r)+"_points.txt" : output file
    Output file with ramet points
filename_prefix+"radius_" + repr(r)+"_circles.txt" : output file
    Output file with sampling circles
```

The documentation for this class was generated from the following file:

- Vector_Analyses_v2.py

3.6 Linear.Analyses.Vector_Analyses_v2.Vector_Analyses Class Reference

Public Member Functions

- def genet_rand_location
- def genet_rand_location_genet_ex_self
- def point_pattern_multiscale_xfig_input
- def point_pattern_xfig_input
- def genet_MC
- def angle_measure_segs

Static Public Attributes

- int **DEBUG** = 0
- int **N_RAND_PTS** = 100
- int **rad_start** = 50
- int **rad_max** = 600
- int **rad_step** = 50
- tuple **RADIUS** = range(rad_start,rad_max,rad_step)
- int **x_limit_min** = 0
- int **x_limit_max** = 2000
- int **y_limit_min** = 0
- int **y_limit_max** = 2000
- list **x_limit** = [x_limit_min,x_limit_max]
- list **y_limit** = [y_limit_min,y_limit_max]
- tuple **PLOT_AREA** = (x_limit[1]-x_limit[0])
- list **xmax** = x_limit[1]
- list **xmin** = x_limit[0]
- list **ymax** = y_limit[1]
- list **ymin** = y_limit[0]
- int **n_perimeter_points** = 400
- string **INTER_POINT_EDGE_CORRECT** = 'FALSE'
- **filename** = FILENAME
- tuple **data_in** = Linear.LinIO.Read_Xfig.Read_data()
- string **PICKLE_DATA** = 'True'
- tuple **gen_proc_obs** = Linear.Process.Proc_genet.GenetProcess()
- tuple **n_segs_per_order** = gen_proc_obs.n_segs_per_order()
- string **obs_filename_prefix** = "obs_"

- **line_analysis_obs** = \
- int **NSIM** = 100
- string **DRAW_SIM** = 'TRUE'
- string **FILE_PLOT_PREFIX** = "func_genet_sim_"
- list **x_limit_sim** = [0,6000]
- list **y_limit_sim** = [0,6000]
- list **simulations** = [ ]
- list **genet_process_list** = [ ]
- tuple **new_genet_process** = copy.deepcopy(gen_proc_obs)
- list **x_old** = new_genet_process.genets[i]
- list **y_old** = new_genet_process.genets[i]
- tuple **x_new** = scipy.random.uniform(high=x_limit[1],low=x_limit[0])
- tuple **y_new** = scipy.random.uniform(high=y_limit[1],low=y_limit[0])
- **x_translation** = x_new-x_old
- **y_translation** = y_new-y_old
- tuple **new_genet_process_ref** = copy.deepcopy(new_genet_process)
- int **nothing** = 0
- list **x_additional_translation** = col∗x_limit[1]
- list **y_additional_translation** = row∗y_limit[1]
- tuple **new_genet_process_tmp** = copy.deepcopy(new_genet_process_ref)
- tuple **filename_plot_eps** = FILE_PLOT_PREFIX+str(SIM)
- float **canvas_extent** = 20.0
- tuple **c** = canvas.canvas()
- tuple **data_extent** = float(x_limit[1]-x_limit[0])
- int **radius** = 1
- int **sim_count** = 0
- list **pt_x1** = new_genet_process.genets[gg]
- list **pt_y1** = new_genet_process.genets[gg]
- list **pt_x2** = new_genet_process.genets[gg]
- list **pt_y2** = new_genet_process.genets[gg]
- tuple **rad** = (radius/data_extent)
- list **x_limit_sim_plot** = [2000,4000]
- list **y_limit_sim_plot** = [2000,4000]
- string **CIRC_SAMP** = 'TRUE'
- string **sim_filename_prefix** = "sim_"
- **line_analysis_sim** = \
- string **WRITE_DATA** = 'FALSE'
- int **DRAW_SIM_N** = 5
- **x_limit_min** = x_limit_min
- **x_limit_max** = x_limit_max
- **y_limit_min** = y_limit_min
- **y_limit_max** = y_limit_max
- list **list_of_fibres** = [ ]

- tuple **temp_pattern** = Linear.Object.Pattern.Pattern(list_of_objects=[list_of_fibres],list_-of_types=['fibre'])
- int **nrand_points** = 100
- string **filename** = "_point_locations.txt"
- tuple **f** = open(filename, 'w')
- list **xlimits** = [ ]
- list **ylimits** = [ ]
- list **point_patterns** = [ ]
- list **limits** = [ ]
- list **npt** = [ ]
- list **density** = [ ]
- list **scale** = [4,5,6,7,8,9,10,11,12]
- list **quadrat_pp** = [ ]
- list **quadrat_size** = x_limit[1]
- list **new_x_limit** = [ ]
- list **new_y_limit** = [ ]
- list **x_limit_tmp_0** = x_limit[0]
- list **y_limit_tmp_0** = y_limit[0]
- **x_limit_tmp_1** = x_limit_tmp_0+quadrat_size
- **y_limit_tmp_1** = y_limit_tmp_0+quadrat_size
- **x_limit_tmp_0** = x_limit_tmp_1
- **y_limit_tmp_0** = y_limit_tmp_1
- list **columns** = [ ]
- list **columns_limits** = [ ]
- list **npt_columns** = [ ]
- list **density_columns** = [ ]
- list **rows** = [ ]
- list **rows_limits** = [ ]
- list **npt_quadrat** = [ ]
- list **density_quadrat** = [ ]
- list **current_x_limit** = [new_x_limit[i][0],new_x_limit[i][1]]
- list **current_y_limit** = [new_y_limit[j][0],new_y_limit[j][1]]
- list **new_pp** = [ ]
- list **pt_x** = temp_pattern.distributed_points[k]
- list **pt_y** = temp_pattern.distributed_points[k]
- list **mc_sim_count** = [ ]
- list **mc_p** = [ ]
- list **random_points_all** = [ ]
- list **row_mc_sim** = [ ]
- list **row_p** = [ ]
- list **random_points_all_r** = [ ]
- list **col_mc_sim** = [ ]
- list **col_p** = [ ]

- list **random_points_all_c** = [ ]
- list **random_points_all_tmp** = [ ]
- list **mc_sim** = [ ]
- list **p** = [ ]
- list **npts** = npt[ndiv]
- list **dist** = [ ]
- tuple **dist_tmp** = sqrt(pow((point_patterns[ndiv][row][col][pt2][0]-point_patterns[ndiv][row][col][pt1][0]),2)+pow((poin patterns[ndiv][row][col][pt2][1]-point_patterns[ndiv][row][col][pt1][1]),2))
- tuple **mean_dist** = mean(dist)
- int **COUNT** = 0
- list **mc_pts** = [ ]
- tuple **x** = random.uniform(limits[ndiv][row][col][0][0],limits[ndiv][row][col][0][1])
- tuple **y** = random.uniform(limits[ndiv][row][col][1][0],limits[ndiv][row][col][1][1])
- list **mcdist** = [ ]
- list **mcdist_tmp** = [ ]
- tuple **mcmean_dist** = mean(mcdist_tmp)
- list **hd_1** = [ ]
- list **vd_1** = [ ]
- list **columns_hd_1** = [ ]
- list **columns_vd_1** = [ ]
- list **quad_hd_1** = [ ]
- list **quad_vd_1** = [ ]
- int **tmp_hd_1** = 0
- int **tmp_vd_1** = 0
- tuple **tmp_hd_1** = (density[ii][i][j+1] - density[ii][i][j-1])
- tuple **tmp_vd_1** = (density[ii][i+1][j] - density[ii][i-1][j])
- list **scale_hd_2** = [ ]
- list **scale_vd_2** = [ ]
- list **columns_hd_2** = [ ]
- list **columns_vd_2** = [ ]
- list **quad_hd_2** = [ ]
- list **quad_vd_2** = [ ]
- int **tmp_hd_2** = 0
- int **tmp_vd_2** = 0
- tuple **tmp_hd_2** = (hd_1[ii][i][j+1] - hd_1[ii][i][j-1])
- tuple **tmp_vd_2** = (vd_1[ii][i+1][j] - vd_1[ii][i-1][j])
- list **scale_vhd_2** = [ ]
- list **columns_vhd_2** = [ ]
- list **quad_vhd_2** = [ ]
- int **tmp_vhd_2** = 0
- tuple **tmp_vhd_2** = (hd_1[ii][i+1][j] - hd_1[ii][i-1][j])

- list **A** = [ ]
- list **B** = [ ]
- list **A_row** = [ ]
- list **B_row** = [ ]
- list **A_column** = [ ]
- list **B_column** = [ ]
- tuple **A_tmp** = array([hd_1[ii][i][j],vd_1[ii][i][j]])
- tuple **B_tmp** = array([[scale_hd_2[ii][i][j],scale_vhd_2[ii][i][j]],[scale_vhd_2[ii][i][j],scale_-vd_2[ii][i][j]]])
- list **bs_pre_sum_levels** = [ ]
- list **bs_sum_levels** = [ ]
- list **bs_pre_sum_columns** = [ ]
- int **bs_sum** = 0
- list **bs_pre_sum_rows** = [ ]
- int **bs_part_temp** = 0
- tuple **mc_u** = zeros([2,1])
- tuple **mc_w** = zeros([2,1])
- tuple **r1** = scipy.random.random_integers(0,len(random_points_all[ii][i][j])-1)
- tuple **r2** = scipy.random.random_integers(0,len(random_points_all[ii][i][j])-1)
- tuple **bs_part_temp** = (npt[ii][i][j]∗float(transpose(matrix(mc_u))∗matrix(B[ii][i][j])∗matrix(mc_-u)))
- **bs_sum** = bs_sum+bs_part_temp
- list **bs_levels** = [ ]
- list **bs_columns** = [ ]
- list **bs_rows** = [ ]
- int **bs** = 0
- tuple **bs** = (npt[ii][i][j]/24.0 ∗ numpy.trace(B[ii][i][j]))
- list **bs_tmp** = [ ]
- list **bs_quad** = [ ]
- list **bs_c** = [ ]
- list **bs_r** = [ ]
- list **var** = [ ]
- list **var_c** = [ ]
- int **quad_count** = 0
- list **var_r** = [ ]
- list **mse** = [ ]
- list **bs_sq_tmp** = bs_quad[ii]
- list **var_sq_tmp** = var[ii]
- tuple **bs_quad** = bs_quad/sum(bs_quad)
- tuple **var** = var/sum(var)
- **rad_start** = rad_start
- **rad_max** = rad_max
- **rad_step** = rad_step

- list **RADIUS** = [ ]
- **rad_tmp** = rad_start
- string **PLOT_SIM_DATA** = "FALSE"
- int **N_PLOT_SIM_DATA** = 1
- string **filename_plot_txt** = "obs_stat_line_based_intensity_simulation.txt"
- tuple **file** = open(filename_plot_txt, "r")
- tuple **data** = file.readlines()
- tuple **n_ints** = len(data)
- float **intensity_tmp** = 0.0
- list **intensity_data** = data[i]
- tuple **intensity_tmp** = intensity_tmp+float(intensity_data[5])
- **intensity_average** = intensity_tmp/n_ints
- tuple **area** = (x_limit[1]-x_limit[0])
- tuple **nrand_points** = int(intensity_average∗area)

3.6.1 Detailed Description

```
Class of vector analyses.
```

3.6.2 Member Function Documentation

3.6.2.1 def Linear.Analyses.Vector_Analyses_v2.Vector_Analyses.genet_rand_location (*FILENAME*)

```
genet_rand_location (v2):
Randomization model for the generation of spatial structures composed of
linear and point data.
A set of linear and point data (in 'genet' structures) is randomized
in the x-y plane and then copied to a 3x3 set of contiguous square regions.
The central region (plot) will then have a complete set of the randomized
data with the effect that linear data extending beyond the plot
boundary will be wrapped to the opposite side of the plot.

Parameters
FILENAME : string
    Filename of xfig file containing genet data

Returns
```

3.6.2.2 def Linear.Analyses.Vector_Analyses_v2.Vector_Analyses.genet_rand_location_genet_-ex_self (*FILENAME*)

```
genet_rand_location_genet_ex_self (v2):
Randomization model for the generation of spatial structures composed of
```

```
linear and point data.
A set of linear and point data (in 'genet' structures) is randomized
in the x-y plane and then copied to a 3x3 set of contiguous square regions.
The central region (plot) will then have a complete set of the randomized
data with the effect that linear data extending beyond the plot
boundary will be wrapped to the opposite side of the plot.

Current genet excludes self from analyses.

Parameters
FILENAME : string
    Filename of xfig file containing genet data

Returns
```

3.6.2.3 def Linear.Analyses.Vector_Analyses_v2.Vector_Analyses.point_pattern_multiscale_xfig_input (*FILENAME*, *x_limit_min*, *x_limit_max*, *y_limit_min*, *y_limit_max*)

```
point_pattern_multiscale_xfig_input (v4):

Conduct a multiscale local-global analysis using random fibre
points distributed over genets.

Parameters
FILENAME : string
    Filename of xfig file containing genet data

Returns
```

3.6.2.4 def Linear.Analyses.Vector_Analyses_v2.Vector_Analyses.point_pattern_xfig_input (*FILENAME*, *rad_start*, *rad_max*, *rad_step*, *x_limit_min*, *x_limit_max*, *y_limit_min*, *y_limit_max*)

```
point_pattern_xfig_input (v1):

Use circle sampling to estimate intensities a various r

Parameters
FILENAME : string
    Filename of xfig file containing genet data

Returns
```

3.6.2.5 def Linear.Analyses.Vector_Analyses_v2.Vector_Analyses.genet_MC (*FILENAME*, *NULL_MODEL_TYPE*, *rad_start*, *rad_max*, *rad_step*, *x_limit_min*, *x_limit_max*, *y_limit_min*, *y_limit_max*)

```
Simulation model for the generation of spatial structures
composed of linear and point data.

Parameters
x_limit_max : float
x_limit_min : float
y_limit_max : float
y_limit_min : float

Returns
```

3.6.2.6 def Linear.Analyses.Vector_Analyses_v2.Vector_Analyses.angle_measure_segs (*FILENAME*, *x_limit_min*, *x_limit_max*, *y_limit_min*, *y_limit_max*)

```
Measure the angles of the segments composing a collection of fibres

Parameters
x_limit_max : float
x_limit_min : float
y_limit_max : float
y_limit_min : float

Returns
"obs_len_angles.txt" : output file
```

The documentation for this class was generated from the following file:

- Vector_Analyses_v2.py

3.7 Linear.Geometry.geom_2.Geom_functions Class Reference

Public Member Functions

- def turn_angle_ccw
- def turn_angle_ccw_test
- def turn_angle_ccw_test_orientation
- def turn_angle_ccw_v2
- def turn_angle_ccw_to_smallest
- def mean_theta
- def theta

- def rad2deg
- def deg2rad
- def cart2polar
- def polar_to_cart
- def intersect_seg
- def intersect_line
- def **intersect_seg_pt**
- def slope
- def ln_eqn_2D
- def pdf_marriot
- def sample_marriot
- def circle_segment_intersection
- def circle_segment_intersection_points
- def circle_segment_intersection_points_version_2
- def **circle_segment_intersection_old**
- def point_point_distance
- def midpoint

Static Public Attributes

- int **turn_theta_tmp** = 2
- **turn_theta_tmp** = turn_theta
- string **inter** = "NO_SOL_HOR"
- tuple **inter** = array([eqn2[1],eqn1[1]])
- list **y** = eqn1[1]
- **eqn** = eqn2
- tuple **x** = (eqn[1][2] - eqn[1][1]∗y)
- list **x** = eqn1[1]
- tuple **y** = (eqn[1][2] - eqn[1][0]∗x)
- tuple **A** = scipy.array([(eqn1[1][0],eqn1[1][1]),(eqn2[1][0],eqn2[1][1])])
- tuple **B** = scipy.array([eqn1[1][2],eqn2[1][2]])
- tuple **y_a** = ((-1)∗b + (sqrt(abs((b∗b)-(4∗a∗c)))))
- **x_b** = pt_x1
- tuple **y_b** = ((-1)∗b - (sqrt(abs((b∗b)-(4∗a∗c)))))
- list **inter_pt_1** = [x_a, y_a]
- int **i_1** = 1
- list **inter_pt_2** = [x_b, y_b]
- int **i_2** = 1
- **y** = pt_y1
- **h** = circ_cen_x
- **k** = circ_cen_y
- **r** = radius

- int **a** = 1
- float **b** = 2.0
- tuple **c** = (h∗h)
- tuple **x_a** = ((-1)∗b + (sqrt(abs((b∗b)-(4∗a∗c)))))
- **y_a** = pt_y1
- tuple **x_b** = ((-1)∗b - (sqrt(abs((b∗b)-(4∗a∗c)))))
- **y_b** = pt_y1
- tuple **a** = (pt_x2 - pt_x1)
- int **b** = 2
- tuple **sol_1** = ((-1 ∗ b) - scipy.sqrt((b∗b)-(4∗a∗c)))
- tuple **sol_2** = ((-1 ∗ b) + scipy.sqrt((b∗b)-(4∗a∗c)))
- list **inter_points** = ["FALSE"]
- **xdiff** = x2-x1
- **ydiff** = y2-y1
- float **deltax** = 2.0
- float **deltay** = 2.0
- **x_mid** = x1+deltax
- **y_mid** = y1+deltay
- list **midpoint** = [x_mid,y_mid]

3.7.1 Detailed Description

```
Generic geometry functions
```

3.7.2 Member Function Documentation

3.7.2.1 def Linear.Geometry.geom_2.Geom_functions.turn_angle_ccw (*seg1*, *seg2*)

```
Calculate the (counter clockwise) turn angles for two line segment
angles (taken from 0/2pi - horizontal).

    Parameters

    Returns
```

3.7.2.2 def Linear.Geometry.geom_2.Geom_functions.turn_angle_ccw_test (*seg1*, *seg2*)

```
# Calculate the (counter clockwise) turn angles for two line segment
# angles (taken from 0/2pi - horizontal).

# The "angle" function in Segment works correctly (used extensively
# for finding intersection points).

    Parameters

    Returns
```

3.7.2.3 def Linear.Geometry.geom_2.Geom_functions.turn_angle_ccw_test_orientation (*seg1*, *seg2*)

```
# Calculate the (counter clockwise) turn angles for two line segment
# angles (taken from 0/2pi - horizontal).

# The "angle" function in Segment works correctly (used extensively
# for finding intersection points).

    Parameters

    Returns
```

3.7.2.4 def Linear.Geometry.geom_2.Geom_functions.turn_angle_ccw_v2 (*seg1*, *seg2*)

```
# Calculate the (counter clockwise) turn angles for two line segment
# angles (taken from 0/2pi - horizontal).

    Parameters

    Returns
```

3.7.2.5 def Linear.Geometry.geom_2.Geom_functions.turn_angle_ccw_to_smallest (*list_of_angles*)

```
turn_angle_ccw_to_smallest

    Parameters

    Returns
```

3.7.2.6 def Linear.Geometry.geom_2.Geom_functions.mean_theta (*list_of_angles*)

```
   # Find the mean angle of a list of angles given in
   # radians.

   Parameters

   Returns
```

3.7.2.7 def Linear.Geometry.geom_2.Geom_functions.theta (*x1*, *y1*, *x2*, *y2*)

```
        # Find the angle (counter-clockwise from polar 0pi rads)
        # to the line defined by the two points.
        #
        # Return:
        #   one angle (radians):
        #       1. small angle (returned by cos and sin)
        #       (NOT ret) 2. large angle (small + pi)

   Parameters

   Returns
```

3.7.2.8 def Linear.Geometry.geom_2.Geom_functions.rad2deg (*radians*)

```
rad2deg

   Parameters

   Returns
```

3.7.2.9 def Linear.Geometry.geom_2.Geom_functions.deg2rad (*degrees*)

```
deg2rad

    Parameters

    Returns
```

3.7.2.10 def Linear.Geometry.geom_2.Geom_functions.cart2polar (*x1, y1, x2, y2*)

```
cart2polar

   Parameters

   Returns
```

3.7.2.11 def Linear.Geometry.geom_2.Geom_functions.polar_to_cart (*theta, r, x, y*)

```
polar_to_cart

   Parameters

   Returns
```

3.7.2.12 def Linear.Geometry.geom_2.Geom_functions.intersect_seg (*seg1*, *seg2*)

```
# Complete a check to see if there is a 'possibility' of the segments
# intersecting. This avoids the need to calculate an intersection
# point if there is no possibility of the segments intersecting.

    Parameters

    Returns
```

3.7.2.13 def Linear.Geometry.geom_2.Geom_functions.intersect_line (*eqn1*, *eqn2*)

```
intersect_line

    Parameters

    Returns
```

3.7.2.14 def Linear.Geometry.geom_2.Geom_functions.slope (*x1*, *y1*, *x2*, *y2*)

```
Generate the segment using two endpoints of type Point

    Parameters

    Returns
```

3.7.2.15 def Linear.Geometry.geom_2.Geom_functions.ln_eqn_2D (*x1*, *y1*, *x2*, *y2*, *m*)

```
# Returns the equation of a line:
#   eq = A*x + B*y = C
#as     eq = [A , B , C]
#   as converted from y - y1 = m(x - x1)

    Parameters

    Returns
```

3.7.2.16 def Linear.Geometry.geom_2.Geom_functions.pdf_marriot (*K*, *psi*)

```
pdf_marriot

    Parameters

    Returns
```

3.7.2.17 def Linear.Geometry.geom_2.Geom_functions.sample_marriot (*K*, *sample_size*)

```
sample_marriot

    Parameters

    Returns
```

3.7.2.18 def Linear.Geometry.geom_2.Geom_functions.circle_segment_intersection (*pt_x1*, *pt_y1*, *pt_x2*, *pt_y2*, *circ_cen_x*, *circ_cen_y*, *radius*, *verbose* = "FALSE")

```
circle_segment_intersection

    Parameters

    Returns
```

3.7.2.19 def Linear.Geometry.geom_2.Geom_functions.circle_segment_intersection_points (*pt_x1*, *pt_y1*, *pt_x2*, *pt_y2*, *circ_cen_x*, *circ_cen_y*, *radius*)

```
circle_segment_intersection_points

    Parameters

    Returns
```

3.7.2.20 def Linear.Geometry.geom_2.Geom_functions.circle_segment_intersection_points_-version_2 (*pt_x1*, *pt_y1*, *pt_x2*, *pt_y2*, *circ_cen_x*, *circ_cen_y*, *radius*)

```
circle_segment_intersection_points_version_2

    Parameters

    Returns
```

3.7.2.21 def Linear.Geometry.geom_2.Geom_functions.point_point_distance (*x1*, *y1*, *x2*, *y2*)

```
point_point_distance

    Parameters

    Returns
```

3.7.2.22 def Linear.Geometry.geom_2.Geom_functions.midpoint (*x1*, *y1*, *x2*, *y2*)

```
midpoint

    Parameters

    Returns
```

The documentation for this class was generated from the following file:

- geom_2.py

3.8 Linear.LinIO.Lin_IO.Read_data Class Reference

Public Member Functions

- def __init__
- def v_read_fib
- def v_read_fib_xfig
- def make_objects
- def read_segs

Public Attributes

- **extents_mat**
- **point_mat**
- **point_list**
- **pt_mat**
- **seg_mat**
- **fib_mat**
- **n_seg**
- **Data_Region**
- **fib_list**
- **list_of_patterns**
- **fib**
- **max_lngth**
- **fib_ID**
- **extent_list**
- **group_list**
- **group_ID_list**
- **fib_ID_list**
- **point_ID_list**

- **circ_points**
- **fib_points**
- **fib_segments**
- **list_of_segments**
- **segment_process**
- **segment_pattern**

3.8.1 Constructor & Destructor Documentation

3.8.1.1 def Linear.LinIO.Lin_IO.Read_data.__init__ (*self*)

```
Initialize the Read_data class
```

3.8.2 Member Function Documentation

3.8.2.1 def Linear.LinIO.Lin_IO.Read_data.v_read_fib (*self, filename*)

```
Read fibres from a text file.

Parameters:
filename : string
    The name of the input data file.
```

3.8.2.2 def Linear.LinIO.Lin_IO.Read_data.v_read_fib_xfig (*self, filename*)

```
Read fibres from an Xfig file.

Parameters:
filename : string
    The name of the input data file.
```

3.8.2.3 def Linear.LinIO.Lin_IO.Read_data.make_objects (*self*)

```
Make Linear objects from the data once it has been read by "v_read_fib".

Parameters:

Returns:
self.list_of_patterns : list
    List of patterns holding the points and fibres contained in the
    original data.
```

3.8.2.4 def Linear.LinIO.Lin_IO.Read_data.read_segs (*self*, *filename*, *x_limit*, *y_limit*, *printsegs* = 'FALSE')

```
Read a segment process from a text file. Each line is one segment with format:

Pt1_x   Pt1_y   Pt2_x   Pt2_y

Return a list of segments, a segment process and a Pattern.
```

The documentation for this class was generated from the following file:

- Lin_IO.py

3.9 Linear.LinIO.Read_Xfig.Read_data Class Reference

Public Member Functions

- def **__init__**
- def **v_read_fib**
- def **v_read_fib_xfig**
- def **make_objects**
- def **make_patterns_colours**

Public Attributes

- **extents_mat**
- **point_mat**
- **point_list**
- **pt_mat**
- **seg_mat**
- **fib_mat**
- **n_seg**
- **Data_Region**
- **fib_list**
- **list_of_patterns**
- **xfig_list**
- **fib**
- **max_lngth**
- **fib_ID**
- **extent_list**
- **group_list**
- **group_ID_list**

- **fib_ID_list**
- **point_ID_list**
- **point_ID_colour_list**
- **point_ID_label_list**
- **fib_ID_colour_list**
- **fib_ID_label_list**
- **group**
- **circ_points**
- **fib_points**
- **fib_segments**
- **colour_ID_list**
- **colour_ID_list_temp**
- **colour_list_of_points**
- **colour_list_of_fibres**
- **list_of_objects**
- **list_of_types**
- **pattern**

Static Public Attributes

- int **count_pt** = 2
- int **count_seg** = 1
- int **max_lngth** = 0
- tuple **max_lngth** = len(self.group_list[h].fib_list[i])
- tuple **vert_correction** = max(extent_mat[:,5])
- float **unit_correction** = 1.0
- **extent_mat** = extent_mat∗unit_correction

The documentation for this class was generated from the following file:

- Read_Xfig.py

3.10 Linear.Object.Circle.Circle Class Reference

Public Member Functions

- def __init__
- def init_rad_Point
- def point_in_circle
- def proportion_in_rectangle

Public Attributes

- **radius**
- **x_cen**
- **y_cen**
- **centre_point**
- **perimeter_points**
- **pts_in_rect**
- **prop_in_rect**

3.10.1 Detailed Description

```
Class of type circle. This is the primary circle class, and all
types of circle will be of this class. The method of initialization
('init_') will differ for different circles.
```

3.10.2 Constructor & Destructor Documentation

3.10.2.1 def Linear.Object.Circle.Circle.__init__ (*self*)

```
Instantiate an object of type circle.

radius = 0
center x = 0
center y = 0
```

3.10.3 Member Function Documentation

3.10.3.1 def Linear.Object.Circle.Circle.init_rad_Point (*self*, *radius*, *Point*)

```
Instantiate a new circle using a radius r and point (x,y)

Arguments:
radius -- circle radius
Point -- a point of type Linear.Object.Point.Point
```

3.10.3.2 def Linear.Object.Circle.Circle.point_in_circle (*self*, *Point*)

```
Test to see if a point of type Point is in the circle

Arguments:
Point -- a point of type Linear.Object.Point.Point
```

3.10.3.3 def Linear.Object.Circle.Circle.proportion_in_rectangle (*self*, *xmin*, *xmax*, *ymin*, *ymax*, *n_perimeter_points*)

```
Calculate the proportion of the circle in a rectangle.

Arguments:
xmin -- the minimum x value of the rectangle perimeter
xmax -- the maximum x value of the rectangle perimeter
ymin -- the minimum x value of the rectangle perimeter
ymax -- the maximum x value of the rectangle perimeter
n_perimeter_points -- the number of equidistant points on the circle
perimeter used to estimate the proportion.
```

The documentation for this class was generated from the following file:

- Object/Circle.py

3.11 Linear.Object.Circle.Circle_circum Class Reference

Public Member Functions

- def __init__
- def get_x
- def get_y

Public Attributes

- **x_cen**
- **y_cen**
- **rad**
- **a**
- **bx**
- **by**
- **c**

3.11.1 Constructor & Destructor Documentation

3.11.1.1 def Linear.Object.Circle.Circle_circum.__init__ (*self*, *Point1*, *Point2*, *Point3*)

```
Class of type circle defined by three perimeter points of type
Point.
```

3.11.2 Member Function Documentation

3.11.2.1 def Linear.Object.Circle.Circle_circum.get_x (*self*, *y*)

```
Find the x coordinates of the circle given a y value.

Arguments:
y -- a y value on the x-y plane
```

3.11.2.2 def Linear.Object.Circle.Circle_circum.get_y (*self*, *x*)

```
Find the y coordinates of the circle given a x value.

Arguments:
x -- a x value on the x-y plane
```

The documentation for this class was generated from the following file:

- Object/Circle.py

3.12 Linear.Object.Circle.Circle_rad_coord Class Reference

Public Member Functions

- def **__init__**

Public Attributes

- **radius**
- **x_cen**
- **y_cen**
- **z_cen**

3.12.1 Detailed Description

```
Class of type circle. Defined by circle center and radius.
```

The documentation for this class was generated from the following file:

- Object/Circle.py

3.13 Linear.Object.Circle.Circle_rad_Point Class Reference

Public Member Functions

- def **__init__**

Public Attributes

- **radius**
- **centre_point**
- **x_cen**
- **y_cen**

3.13.1 Detailed Description

```
Class of type circle. Defined by circle center of type Point
and radius.
```

The documentation for this class was generated from the following file:

- Object/Circle.py

3.14 Linear.Object.Fibre.Fibre Class Reference

Public Member Functions

- def __init__
- def gen_from_segs
- def gen_from_sim
- def gen_from_sim_branches
- def **print_fibre_points**
- def **distribute_random_points**
- def **distribute_regular_points**
- def **distribute_segs_on_points**
- def **set_ID**
- def **length**
- def **effective_length**
- def **rose**
- def **mean_vector**
- def **curvature**
- def **length_distribution**

- def **segment_turn_angles**
- def **fibre_origin_angle**
- def **translate**

Public Attributes

- **instantiated**
- **segments**
- **n_segs**
- **n_points**
- **turn_angles**
- **origin_angle**
- **distn_segment_length**
- **stolon_ramet_index**
- **parent_ramet_index**
- **unique_pts**
- **ID**
- **ID_label**
- **ID_colour**
- **regular_points_list_array**
- **regular_points_list_points**
- **pt_origin**
- **fibre_length**
- **static_length**
- **static_effective_length**
- **static_rose**
- **radii**

3.14.1 Detailed Description

```
Class of type Fibre.
```

3.14.2 Constructor & Destructor Documentation

3.14.2.1 def Linear.Object.Fibre.Fibre.__init__ (*self*)

```
Instantiate instance of class of type Fibre.
```

3.14.3 Member Function Documentation

3.14.3.1 def Linear.Object.Fibre.Fibre.gen_from_segs (*self*, *list_of_segments*)

```
Generate a fibre from a list of segments of type Segment

Arguments:
list_of_segments -- a list of segments of type Linear.Object.Segment
```

3.14.3.2 def Linear.Object.Fibre.Fibre.gen_from_sim (*self*, *n_segs*, *turn_angle*, *orientation*, *angles_distn*, *length_distn*, *origin_orientation*, *origin_x*, *origin_y*)

```
Generate a fibre using a set of simulation parameters

Parameters
n_segs : int
    Number of segments to compose the fibre
turn_angle : boolean
    If "TRUE" the distribution of angles, angles_distn, is
    a distribution of turn angles where the turn angle is the
    deviation of the second segment from the axis of the first
    segment
orientation : boolean
    If "TRUE" the distribution of angles, angles_distn, is
    a distribution of segment orientations where the orientation is the
    deviation in radians from zerio in the counter clockwise direction
angles_distn : list of floats
    A list containing a distribution of angles in radians
length_distn : list of floats
    A list containing a distribution of segment lengths
origin_orientation : float
    The orientation, measured in radians ccw from zero,
    of the first segment
origin_x : float
    The x coordinate in the x-y plane of the fibre begining
origin_y : float
    The y coordinate in the x-y plane of the fibre begining

Returns
segments : list
    A list of segments of type Segment comprising the fibre.
n_segs : int
    Number of segments in the fibre
n_points
    Number of points (endpoints of segments) in the fibre.
```

3.14.3.3 def Linear.Object.Fibre.Fibre.gen_from_sim_branches (*self, n_segs, turn_angle, orientation, angles_distn, length_distn, origin_orientation, origin_x, origin_y, BRANCH_FORM_LIKELIHOOD, MIN_TURN_ANGLE, MAX_TURN_ANGLE, BRANCH_NSEGS*)

```
Generate a fibre with branching using a set of simulation parameters

Parameters
n_segs : int
    Number of segments to compose the fibre
turn_angle : boolean
    If "TRUE" the distribution of angles, angles_distn, is
    a distribution of turn angles where the turn angle is the
    deviation of the second segment from the axis of the first
    segment
orientation : boolean
    If "TRUE" the distribution of angles, angles_distn, is
    a distribution of segment orientations where the orientation is the
    deviation in radians from zerio in the counter clockwise direction
angles_distn : list of floats
    A list containing a distribution of angles in radians
length_distn : list of floats
    A list containing a distribution of segment lengths
origin_orientation : float
    The orientation, measured in radians ccw from zero,
    of the first segment
origin_x : float
    The x coordinate in the x-y plane of the fibre begining
origin_y : float
    The y coordinate in the x-y plane of the fibre begining
BRANCH_FORM_LIKELIHOOD : float
MIN_TURN_ANGLE : float
MAX_TURN_ANGLE : float
BRANCH_NSEGS : list



Returns
segments : list
    A list of segments of type Segment comprising the fibre.
n_segs : int
    Number of segments in the fibre
n_points
    Number of points (endpoints of segments) in the fibre.
```

The documentation for this class was generated from the following file:

- Object/Fibre.py

3.15 Linear.Object.Genet.Genet Class Reference

Public Member Functions

- def __init__
- def set_data
- def generate_from_data
- def genet_n_segs_per_fibre
- def genet_length
- def genet_n_stolon
- def genet_ave_stolon_length
- def genet_n_ramets
- def print_points
- def print_fibres
- def daughter_distance
- def distribute_random_points
- def distribute_random_points_ave_length
- def **find_parent**
- def intersect_points_fibres
- def ang_dist_analysis

Public Attributes

- **data_dict**
- **di**
- **total_length**
- **ave_stolon_length**
- **n_fib_par**
- **n_ramets**
- **list_of_fibres**
- **n_stolon**
- **n_segs_per_fibre**
- **parent_point**
- **daughter_points**
- **fibre_length**
- **cumulative_n_segs**
- **distributed_points**
- **parent_ramet**
- **rad_intensity_L**
- **n_points_corrected_list**
- **l_circle_perimeter_corrected_list**
- **rad_intersection_points**

Static Public Attributes

- list **x2** = rad_intersection_points[i]
- list **y2** = rad_intersection_points[i]
- float **orientation** = 0.0
- tuple **orientation** = scipy.arctan((y2-y1)/(x2-x1))
- **orientation** = orientation
- int **orientation** = 2
- tuple **orient_sum** = sum(orientation_classes_count)
- list **orientation_classes_prop** = [ ]
- tuple **c** = canvas.canvas()
- int **N_ANG_CLASSES** = 18
- int **div_angle** = 360
- tuple **div_angle_list** = range(0,(360+div_angle),div_angle)
- **reg_col** = color.gradient.BlueRed
- list **norm_rad_intensity_L** = [ ]
- list **tmp** = rad_intensity_L[hh]
- list **angle_1** = div_angle_list[j]
- list **angle_2** = div_angle_list[j+1]
- int **r1** = 1
- list **r2** = r1+rad_orientation_classes_prop[i]
- tuple **bx1** = r1∗scipy.cos(angle_2 ∗ 2∗scipy.pi/360.0)
- tuple **by1** = r1∗scipy.sin(angle_2 ∗ 2∗scipy.pi/360.0)
- tuple **arc** = path.path(path.arc(0,0, r2,angle_1,angle_2),path.lineto(bx1,by1),path.arcn(0,0, r1,angle_2,angle_1),path.closepath())

3.15.1 Detailed Description

```
Class of type Genet. An object of type Genet is a class to group
fibres (collections of segments) and points into a structure similar
to a plant genet.
```

3.15.2 Constructor & Destructor Documentation

3.15.2.1 def Linear.Object.Genet.Genet.__init__ (*self*)

```
Instantiate instance of class of type Genet.

Parameters


Returns
self.data_dict = {}
self.di = {}
self.total_length = 'FALSE'
self.ave_stolon_length = 'FALSE'
self.n_fib_par = 'FALSE'
self.n_ramets = 'FALSE'
self.list_of_fibres = []
self.n_stolon = 'FALSE'
self.n_segs_per_fibre = 'FALSE'
self.parent_point = []
self.daughter_points = []
```

3.15.3 Member Function Documentation

3.15.3.1 def Linear.Object.Genet.Genet.set_data (*self, di*)

```
Set the data dictionary that will hold the genet data.
Points and Fibres are held in dictionary entries having unique
names.


Parameters
di : Python dictionary

Returns
self.data_dict : Python dictionary
```

3.15.3.2 def Linear.Object.Genet.Genet.generate_from_data (*self, list_of_fibres, parent_point, daughter_points*)

```
Set the data dictionary that will hold the genet data.
Points and Fibres are held in dictionary entries having unique
names.


Parameters
list_of_fibres : list
parent_point : list
daughter_points : list

Returns
self.di : Python dictionary
```

3.15.3.3 def Linear.Object.Genet.Genet.genet_n_segs_per_fibre (*self*)

```
Find the number of segments per fibre in the genet.


Parameters
```

```
Returns
self.n_segs_per_fibre : list
```

3.15.3.4 def Linear.Object.Genet.Genet.genet_length (*self*)

```
Find the total fibre length of the genet.


Parameters

Returns
self.total_length : float
```

3.15.3.5 def Linear.Object.Genet.Genet.genet_n_stolon (*self*)

```
Find the number of stolons on the parent point

Parameters

Returns
self.n_fib_par : int
```

3.15.3.6 def Linear.Object.Genet.Genet.genet_ave_stolon_length (*self*)

```
Calculate the average stolon length

Parameters

Returns
self.ave_stolon_length : float
```

3.15.3.7 def Linear.Object.Genet.Genet.genet_n_ramets (*self*)

```
Calculate the number of ramets [points] (daughter and parent)

Parameters

Returns
self.n_ramets : int
```

3.15.3.8 def Linear.Object.Genet.Genet.print_points (*self*)

```
Print points

Parameters

Returns
```

3.15.3.9 def Linear.Object.Genet.Genet.print_fibres (*self*)

```
Print fibres

Parameters

Returns
```

3.15.3.10 def Linear.Object.Genet.Genet.daughter_distance (*self*)

```
Calculate the distance of daughter plants from parent

Parameters

Returns
distances : list
    List of distances
```

3.15.3.11 def Linear.Object.Genet.Genet.distribute_random_points (*self, N_RAND_PTS, VERBOSE =* "FALSE")

```
Distribute N random points over the fibres contained in the
list_of_fibres

Parameters
N_RAND_PTS : int
    Number of random points to distribute over the Genet
VERBOSE="FALSE"

Returns
self.list_of_fibres[h].segments[j].point_locations : list
    A list of points for each segment in each fibre of the genet
```

3.15.3.12 def Linear.Object.Genet.Genet.distribute_random_points_ave_length (*self, AVE_LENGTH, VERBOSE =* "FALSE")

```
Distribute N random points over the fibres contained in the
```

```
list_of_fibres where N = length of fibres / n points per unit length

Parameters
AVE_LENGTH : float
    Desired number points per length of fibre
VERBOSE="FALSE"

Returns
self.list_of_fibres[h].segments[j].point_locations : list
    A list of points for each segment in each fibre of the genet
```

3.15.3.13 def Linear.Object.Genet.Genet.intersect_points_fibres (*self*)

```
Find intersection between list_of_points and points comprising fibres.
Mark the fibre/segment points with "is_ramet" if they are in the
list_of_points.
Add the index of the point (in list_of_points) to the fibre for easy
retrieval.
Mark the point (in list_of_points) as belonging to a fibre.
Record the number of linear components that intersect the point.
Add the fibre index (in list_of_fibres) to the point for easy retrieval.


Parameters


Returns
self.list_of_points[k].stolon_fibre_seg_index : list
```

3.15.3.14 def Linear.Object.Genet.Genet.ang_dist_analysis (*self, RADII, N_CLASSES, x_limit, y_limit, filename, DRAW_INDIVIDUALS*)

```
Produce a custom rose of directions with multiple classes that
represent distance classes around the parent ramet.


Parameters
RADII
N_CLASSES
x_limit
y_limit
filename
DRAW_INDIVIDUALS

Returns
```

The documentation for this class was generated from the following file:

- Genet.py

3.16 Linear.Object.Group.Group Class Reference

Public Member Functions

- def **__init__**

Public Attributes

- **ID**
- **fib_list**
- **fib_ID_label_list**
- **fib_ID_colour_list**
- **point_list**
- **point_ID_label_list**
- **point_ID_colour_list**
- **point_mat**

The documentation for this class was generated from the following file:

- Group.py

3.17 Linear.Object.Pattern.Pattern Class Reference

Public Member Functions

- def __init__
- def print_points
- def print_fibres
- def distribute_random_points
- def **distribute_random_points_bak**
- def find_parent
- def **intersect_points_fibres**
- def calc_effective_length_fibres

Public Attributes

- **objects**
- **no_types**
- **no_objects**
- **list_of_patterns**

- **list_of_fibres**
- **list_of_segments**
- **list_of_points**
- **ID**
- **distributed_points**
- **no_points**
- **no_segments**
- **no_fibres**
- **no_patterns**
- **parent_ramet**
- **fibre_length**
- **cumulative_n_segs**

3.17.1 Detailed Description

```
A collection of points, fibres, segments, and/or patterns.
```

3.17.2 Constructor & Destructor Documentation

3.17.2.1 def Linear.Object.Pattern.Pattern.__init__ (*self*, *list_of_objects*, *list_of_types*)

```
Find the intensity of lines using circles placed randomly over
the linear data. Repeat the process for a range of circle radii.

Points are distributed over the fibres.

Parameters
list_of_objects : list
    A list of objects of type Object (patterns, fibres, segments, points)
list_of_types : list
    A list identifying the type of objects in list_of_objects

Returns
self.objects = list_of_objects
self.no_types = len(list_of_types)
self.no_objects = len(list_of_objects)

self.list_of_patterns = []
self.list_of_fibres = []
self.list_of_segments = []
self.list_of_points = []

self.ID = 0

self.distributed_points = []

self.no_points = len(self.list_of_points)
self.no_segments = len(self.list_of_segments)
self.no_fibres = len(self.list_of_fibres)
self.no_patterns = len(self.list_of_patterns)
```

```
self.parent_ramet
```

3.17.3 Member Function Documentation

3.17.3.1 def Linear.Object.Pattern.Pattern.print_points (*self*)

```
Print the points in the Pattern

Parameters

Returns
midpoint : stdout()
```

3.17.3.2 def Linear.Object.Pattern.Pattern.print_fibres (*self*)

```
Print the fibres in the Pattern

Parameters

Returns
midpoint : stdout()
```

3.17.3.3 def Linear.Object.Pattern.Pattern.distribute_random_points (*self*, *N_RAND_PTS*, *x_limit*, *y_limit*, *VERBOSE* = "FALSE")

```
Distribute N random points over the fibres contained in the
list_of_fibres of Pattern.

Parameters
x_limit : list
    Limits of the plot, [x minimum, x maximum]
y_limit : list
    limits of the plot, [y minimum, y maximum]

Returns
```

3.17.3.4 def Linear.Object.Pattern.Pattern.find_parent (*self*)

```
Locate the "double circle" used to identify the "parent" ramet.
Mark the point with "is_parent" tag and remove the duplicated
point from list_of_points (because each list has an extra
point).

NOT COMPLETE

Parameters

Returns
```

3.17.3.5 def Linear.Object.Pattern.Pattern.calc_effective_length_fibres (*self*, *x_limit*, *y_limit*, *filename_prefix*)

```
Calculate the "effective length" of fibres. This is the length of
fibres inside the rectangular plot delimited by x_limit[x_min,x_max]
and y_limit[y_min,y_max].

Parameters
x_limit : list
    Limits of the plot, [x minimum, x maximum]
y_limit : list
    limits of the plot, [y minimum, y maximum]

Returns
filename_prefix+"_effective_linear_length.txt" : file
```

The documentation for this class was generated from the following file:

- Object/Pattern.py

3.18 Linear.Object.Point.Point Class Reference

Public Member Functions

- def __init__
- def translate
- def randomize

Public Attributes

- **x**
- **y**
- **z**
- **meta1**
- **ID**

- **ID_label**
- **ID_colour**
- **is_ramet**
- **is_parent**
- **is_terminus**
- **is_branch**
- **no_in**
- **no_out**
- **stolon_fibre_index**
- **point_marks_array**

3.18.1 Detailed Description

```
Class of type Point. An object of type Point is representative of
the geometric point in the x-y-z space.
```

3.18.2 Constructor & Destructor Documentation

3.18.2.1 def Linear.Object.Point.Point.__init__ (*self, x, y, z =* 0*, meta1 =* 0)

```
Instantiate instance of class of type Point.
```

3.18.3 Member Function Documentation

3.18.3.1 def Linear.Object.Point.Point.translate (*self, x_translation, y_translation*)

```
Translate a Point by values of x and y

Parameters
x_translation : float
    Translate the point by x_translation units in the
    x direction

y_translation : float
    Translate the point by y_translation units in the
    y direction

Returns
x : float
y : float
```

3.18.3.2 def Linear.Object.Point.Point.randomize (*self, x_limit, y_limit*)

```
Randomize Point within the bounds set by x_limit and y_limit
```

```
Parameters
x_limit : list of length 2
    A list having the lower and upper values of x for the plot
    limits

y_limit : list of length 2
    A list having the lower and upper values of y for the plot
    limits
```

The documentation for this class was generated from the following file:

- Object/Point.py

3.19 Linear.Object.Rectangle.Rectangle Class Reference

Public Member Functions

- def __init__
- def check_point
- def intersect_seg
- def intersect_seg_points

Public Attributes

- **pA**
- **pB**
- **pC**
- **pD**
- **ab**
- **bc**
- **cd**
- **da**

3.19.1 Detailed Description

```
Class of type Rectangle. Forms a rectangle from Segments given
four points of type Point.
```

3.19.2 Constructor & Destructor Documentation

3.19.2.1 def Linear.Object.Rectangle.Rectangle.__init__ (*self*, *Point_A*, *Point_B*, *Point_C*, *Point_D*)

```
Generate the segment using two endpoints of type Point
```

```
Parameters
Point_A, Point_B,Point_C,Point_D : Point
    Four points of type Point to be used as corners

Returns
Rectangle boundary (four Segments) : Segment
    ab, bc, cd, da
```

3.19.3 Member Function Documentation

3.19.3.1 def Linear.Object.Rectangle.Rectangle.check_point (*self, Point*)

```
Check to see if a point of type Point is within the rectangle

Parameters
Point : Point
    The point to be tested.

Returns
point_in : int
    Either 0 or 1
```

3.19.3.2 def Linear.Object.Rectangle.Rectangle.intersect_seg (*self, Segment*)

```
Find the portion of a segment inside the rectangle.
Make this portion of the segment into its own segment.

Parameters
Segment : Segment
    The segment to be tested.

Returns
seg_fraction : Segment
```

3.19.3.3 def Linear.Object.Rectangle.Rectangle.intersect_seg_points (*self, Segment*)

```
Find the intersection points of a segment and the rectangle.

Parameters
Segment : Segment
    The segment to be tested.

Returns
inter_points : list
    A list of intersection points
```

The documentation for this class was generated from the following file:

- Rectangle.py

3.20 Linear.Object.Region.Region Class Reference

Public Member Functions

- def __init__
- def set_extents
- def add_Point_type
- def add_Point
- def add_Segment_type
- def add_Segment
- def add_Fibre_type
- def add_Fibre
- def add_Pattern_type
- def add_Pattern
- def Point_intersect_Region
- def Segment_intersect_Region
- def Fibre_intersect_Region

Public Attributes

- **x_max**
- **x_min**
- **y_max**
- **y_min**
- **P1**
- **P2**
- **P3**
- **P4**
- **region_pt**
- **S12**
- **S23**
- **S43**
- **S14**
- **region_seg**
- **Points**
- **Segments**
- **Fibres**
- **Patterns**
- **intercepts**

3.20.1 Detailed Description

```
Class of type Region. An object of type Region is representative of
a 2-dimensional region in x-y space.
```

3.20.2 Constructor & Destructor Documentation

3.20.2.1 def Linear.Object.Region.Region.__init__ (*self*, *x_min*, *x_max*, *y_min*, *y_max*)

```
Instantiate an instance of type Region

Parameters
x_max : float
x_min : float
y_max : float
y_min : float

Returns
self.x_max = x_max
self.x_min = x_min
self.y_max = y_max
self.y_min = y_min
self.P1 = Point.Point(x_min,y_max, 0, 0)
self.P2 = Point.Point(x_max,y_max, 0, 0)
self.P3 = Point.Point(x_max,y_min, 0, 0)
self.P4 = Point.Point(x_min,y_min, 0, 0)
self.region_pt = [self.P1,self.P2,self.P3,self.P4]

self.S12 = Segment.Segment()
self.S12.gen_from_endpoints(self.P1,self.P2)
self.S23 = Segment.Segment()
self.S23.gen_from_endpoints(self.P2,self.P3)
self.S43 = Segment.Segment()
self.S43.gen_from_endpoints(self.P4,self.P3)
self.S14 = Segment.Segment()
self.S14.gen_from_endpoints(self.P1,self.P4)
self.region_seg = [self.S12,self.S23,self.S43,self.S14]

self.Points = []
self.Segments = []
self.Fibres = []
self.Patterns = []
```

3.20.3 Member Function Documentation

3.20.3.1 def Linear.Object.Region.Region.set_extents (*self*, *x_min*, *x_max*, *y_min*, *y_max*)

```
Print the points in the Pattern

Parameters
```

```
Returns
midpoint : stdout()
```

3.20.3.2 def Linear.Object.Region.Region.add_Point_type (*self*)

```
Print the points in the Pattern

Parameters

Returns
midpoint : stdout()
```

3.20.3.3 def Linear.Object.Region.Region.add_Point (*self*, *number*, *point*)

```
Print the points in the Pattern

Parameters

Returns
midpoint : stdout()
```

3.20.3.4 def Linear.Object.Region.Region.add_Segment_type (*self*)

```
Print the points in the Pattern

Parameters

Returns
midpoint : stdout()
```

3.20.3.5 def Linear.Object.Region.Region.add_Segment (*self*, *number*, *segment*)

```
Print the points in the Pattern

Parameters

Returns
midpoint : stdout()
```

3.20.3.6 def Linear.Object.Region.Region.add_Fibre_type (*self*)

```
Print the points in the Pattern

Parameters

Returns
midpoint : stdout()
```

3.20.3.7 def Linear.Object.Region.Region.add_Fibre (*self*, *Fibre*)

```
Print the points in the Pattern

Parameters

Returns
midpoint : stdout()
```

3.20.3.8 def Linear.Object.Region.Region.add_Pattern_type (*self*)

```
Print the points in the Pattern

Parameters

Returns
midpoint : stdout()
```

3.20.3.9 def Linear.Object.Region.Region.add_Pattern (*self*, *Pattern*)

```
Print the points in the Pattern

Parameters

Returns
midpoint : stdout()
```

3.20.3.10 def Linear.Object.Region.Region.Point_intersect_Region (*self*, *point*)

```
Print the points in the Pattern

Parameters
```

```
Returns
midpoint : stdout()
```

3.20.3.11 def Linear.Object.Region.Region.Segment_intersect_Region (*self*, *segment*)

```
Find the portion of a Segment that intersects the Region

Return a tuple seg_in_reg = [New_Segment,"in_reg"] composed
of a segment object and a string indicating the nature
of the intersection.

Parameters
segment : Segment
    The segment of class Segment to be tested

Returns
seg_in_reg : list
    [in_reg, new_segment]

in_reg = either FALSE, TRUE, PART
new_segment = the portion of the segment in the Region
```

3.20.3.12 def Linear.Object.Region.Region.Fibre_intersect_Region (*self*, *Fibre*)

```
Find the portion of a Fibre that intersects the Region

Return a tuple seg_in_reg = [New_Fibre,"in_reg"] composed
of a segment object and a string indicating the nature
of the intersection.

Parameters
fibre : Fibre
    The fibre of class Fibre to be tested

Returns
fib_in_reg : list
    [in_reg, new_fibre]

in_reg = FALSE, TRUE
new_fibre = the portion of the fibre in the Region
```

The documentation for this class was generated from the following file:

- Object/Region.py

3.21 Linear.Object.Segment.Segment Class Reference

Public Member Functions

- def __init__
- def gen_from_endpoints
- def find_midpoint
- def print_seg
- def max_x
- def min_x
- def max_y
- def min_y
- def length
- def effective_length
- def limit_test
- def point_on_seg
- def extend
- def parallel_extended
- def parallel
- def mk_polygon_rect
- def distribute_random_points
- def distribute_segs_on_points
- def make_Ripleys_region_semicircle
- def translate

Public Attributes

- **instantiated**
- **inside_seg**
- **outside_seg**
- **points**
- **x1**
- **y1**
- **z1**
- **metaA**
- **x2**
- **y2**
- **z2**
- **metaB**
- **point1**
- **point2**
- **point1_ordered**

- **point2_ordered**
- **point1_unsorted**
- **point2_unsorted**
- **ID_label**
- **ID_colour**
- **angle**
- **orientation**
- **point_locations**
- **list_of_intersections**
- **list_of_segment_pairs**
- **m**
- **slope**
- **equation**
- **static_length**
- **test_x1**
- **test_y1**
- **test_z1**
- **ep_x1**
- **ep_y1**
- **ep_x2**
- **ep_y2**
- **p1_ep1_pt_ex**
- **p1_ep2_pt_ex**
- **p2_ep1_pt_ex**
- **p2_ep2_pt_ex**
- **par_seg_1_ex**
- **par_seg_2_ex**
- **p1_ep1_pt**
- **p1_ep2_pt**
- **p2_ep1_pt**
- **p2_ep2_pt**
- **par_seg_1**
- **par_seg_2**
- **polygon_rect**
- **segment_length**
- **rect**

3.21.1 Detailed Description

```
Class of type Segment. An object of type Segment is representative of
the geometric line segment in the x-y-z space.
```

3.21.2 Constructor & Destructor Documentation

3.21.2.1 def Linear.Object.Segment.Segment.__init__ (*self*)

```
Instantiate instance of class of type Segment.
```

3.21.3 Member Function Documentation

3.21.3.1 def Linear.Object.Segment.Segment.gen_from_endpoints (*self*, *Point_A*, *Point_B*)

```
Generate the segment using two endpoints of type Point

Parameters
Point_A, Point_B : Point
    Two points of type Point to be used as endpoints

Returns
self.points = [Point_A,Point_B]
self.x1 = Point_A.x
self.y1 = Point_A.y
self.z1 = Point_A.z
self.metaA = Point_A.meta1
self.x2 = Point_B.x
self.y2 = Point_B.y
self.z2 = Point_B.z
self.metaB = Point_B.meta1

self.angle
self.point_locations = []
self.list_of_intersections = []
self.list_of_segment_pairs = []
self.m = Linear.Geometry.geom.slope(self.x1,self.y1,self.x2,self.y2)
self.slope = self.m

self.equation
```

3.21.3.2 def Linear.Object.Segment.Segment.find_midpoint (*self*)

```
Find the midpoint of a line segment of type Segment

Returns
midpoint : list
    List of two floats, [x,y] giving the coordinates of the midpoint
```

3.21.3.3 def Linear.Object.Segment.Segment.print_seg (*self*)

```
Prints the x1, y1, z1, x2, y2, z2 coordinates to stdout()
```

```
#         Returns
#         midpoint : list
#             List of two floats, [x,y] giving the coordinates of the midpoint
```

3.21.3.4 def Linear.Object.Segment.Segment.max_x (*self*)

```
Find the midpoint of a line segment of type Segment

Parameters

Returns
midpoint : list
    List of two floats, [x,y] giving the coordinates of the midpoint
```

3.21.3.5 def Linear.Object.Segment.Segment.min_x (*self*)

```
Find the midpoint of a line segment of type Segment

Parameters

Returns
midpoint : list
    List of two floats, [x,y] giving the coordinates of the midpoint
```

3.21.3.6 def Linear.Object.Segment.Segment.max_y (*self*)

```
Find the midpoint of a line segment of type Segment

Parameters

Returns
midpoint : list
    List of two floats, [x,y] giving the coordinates of the midpoint
```

3.21.3.7 def Linear.Object.Segment.Segment.min_y (*self*)

```
Find the midpoint of a line segment of type Segment

Parameters

Returns
midpoint : list
    List of two floats, [x,y] giving the coordinates of the midpoint
```

3.21.3.8 def Linear.Object.Segment.Segment.length (*self*)

```
Find the midpoint of a line segment of type Segment

Parameters

Returns
midpoint : list
    List of two floats, [x,y] giving the coordinates of the midpoint
```

3.21.3.9 def Linear.Object.Segment.Segment.effective_length (*self*, *x_limit*, *y_limit*)

```
Find the midpoint of a line segment of type Segment

Parameters

Returns
midpoint : list
    List of two floats, [x,y] giving the coordinates of the midpoint
```

3.21.3.10 def Linear.Object.Segment.Segment.limit_test (*self*, *x_limit*, *y_limit*)

```
Find the midpoint of a line segment of type Segment

Parameters

Returns
midpoint : list
    List of two floats, [x,y] giving the coordinates of the midpoint
```

3.21.3.11 def Linear.Object.Segment.Segment.point_on_seg (*self*, *Point*)

```
Find the midpoint of a line segment of type Segment

Parameters

Returns
midpoint : list
    List of two floats, [x,y] giving the coordinates of the midpoint
```

3.21.3.12 def Linear.Object.Segment.Segment.extend (*self*, *rad*)

```
Find the midpoint of a line segment of type Segment

Parameters
```

```
Returns
midpoint : list
    List of two floats, [x,y] giving the coordinates of the midpoint
```

3.21.3.13 def Linear.Object.Segment.Segment.parallel_extended (*self*, *rad*)

```
Find the midpoint of a line segment of type Segment

Parameters

Returns
midpoint : list
    List of two floats, [x,y] giving the coordinates of the midpoint
```

3.21.3.14 def Linear.Object.Segment.Segment.parallel (*self*, *rad*)

```
Find the midpoint of a line segment of type Segment

Parameters

Returns
midpoint : list
    List of two floats, [x,y] giving the coordinates of the midpoint
```

3.21.3.15 def Linear.Object.Segment.Segment.mk_polygon_rect (*self*, *rad*)

```
Find the midpoint of a line segment of type Segment

Parameters

Returns
midpoint : list
    List of two floats, [x,y] giving the coordinates of the midpoint
```

3.21.3.16 def Linear.Object.Segment.Segment.distribute_random_points (*self*, *N_RAND_PTS*)

```
Find the midpoint of a line segment of type Segment

Parameters

Returns
midpoint : list
    List of two floats, [x,y] giving the coordinates of the midpoint
```

3.21.3.17 def Linear.Object.Segment.Segment.distribute_segs_on_points (*self*, *RADIUS*)

```
Find the midpoint of a line segment of type Segment

Parameters

Returns
midpoint : list
    List of two floats, [x,y] giving the coordinates of the midpoint
```

3.21.3.18 def Linear.Object.Segment.Segment.make_Ripleys_region_semicircle (*self*, *RADIUS*)

```
Find the midpoint of a line segment of type Segment

Parameters

Returns
midpoint : list
    List of two floats, [x,y] giving the coordinates of the midpoint
```

3.21.3.19 def Linear.Object.Segment.Segment.translate (*self*, *x_translation*, *y_translation*)

```
Find the midpoint of a line segment of type Segment

Parameters

Returns
midpoint : list
    List of two floats, [x,y] giving the coordinates of the midpoint
```

The documentation for this class was generated from the following file:

- Object/Segment.py

3.22 Linear.Object.Segment.Segment_Polygon_Rect Class Reference

Public Member Functions

- def __init__
- def seg_intercept

Public Attributes

- **seg_poly_rect**
- **intercepts**

3.22.1 Detailed Description

```
Find the midpoint of a line segment of type Segment

Parameters

Returns
midpoint : list
    List of two floats, [x,y] giving the coordinates of the midpoint
```

3.22.2 Constructor & Destructor Documentation

3.22.2.1 def Linear.Object.Segment.Segment_Polygon_Rect.__init__ (*self*, *seg1*, *seg2*)

```
Find the midpoint of a line segment of type Segment

Parameters

Returns
midpoint : list
    List of two floats, [x,y] giving the coordinates of the midpoint
```

3.22.3 Member Function Documentation

3.22.3.1 def Linear.Object.Segment.Segment_Polygon_Rect.seg_intercept (*self*, *seg*)

```
Find the midpoint of a line segment of type Segment

Parameters

Returns
midpoint : list
    List of two floats, [x,y] giving the coordinates of the midpoint
```

The documentation for this class was generated from the following file:

- Object/Segment.py

3.23 Linear.Process.Proc_fibre.Generate_Fibre Class Reference

Public Member Functions

- def __init__
- def **generate**
- def generate_from_list

Public Attributes

- **n_segs**
- **x_limit**
- **y_limit**
- **seg_length**
- **max_len**
- **min_len**
- **seg**

3.23.1 Constructor & Destructor Documentation

3.23.1.1 def Linear.Process.Proc_fibre.Generate_Fibre.__init__ (*self*)

```
A class to generate a fibre process.
```

3.23.2 Member Function Documentation

3.23.2.1 def Linear.Process.Proc_fibre.Generate_Fibre.generate_from_list (*self, list_of_segs*)

```
Generate a fibre process from a list of segments.
```

The documentation for this class was generated from the following file:

- Proc_fibre.py

3.24 Linear.Process.Proc_genet.GenetProcess Class Reference

Public Member Functions

- def __init__
- def generate_from_XFig_input
- def assign_parents
- def assign_fibres
- def append_genet
- def generate_from_list
- def draw_data
- def generate_from_simulation
- def make_pattern
- def probability_of_branching

- def n_segs_per_order
- def calc_length
- def calc_length_pattern
- def calc_effective_length_fibres
- def calc_n_points
- def angle_of_branching
- def distribute_random_points_ave_length
- def ang_dist_analysis

Public Attributes

- **dicts**
- **genets**
- **data_in**
- **pattern**
- **x_limit**
- **y_limit**
- **parent_points**
- **daughter_points**
- **genets_list**
- **tl_di**
- **distributed_points**
- **fibre_length**

Static Public Attributes

- tuple **angles_distn**
- tuple **origin_orientation**
- tuple **fib_tmp** = Linear.Object.Fibre.Fibre()
- string **TURN_ANGLE** = 'TRUE'
- list **previous_order_fibres** = [ ]
- list **current_order_fibres** = [ ]
- list **fib_tmp_prev** = previous_order_fibres[ii]
- tuple **theo_prob** = scipy.random.random()
- list **x_origin_tmp**
- list **y_origin_tmp**
- list **prev_angle** = fib_tmp_prev.segments[len(fib_tmp_prev.segments)-1]
- int **BRANCH_NSEGS_1** = 0
- tuple **BRANCH_NSEGS_1**
- int **BRANCH_NSEGS_2** = 0
- tuple **BRANCH_NSEGS_2**
- **branch_angles_distn_1** = \

- **low** = MIN_TURN_ANGLE,\
- **high** = MAX_TURN_ANGLE,\
- **size** = BRANCH_NSEGS_1)
- **branch_angles_distn_2** = \
- **branch_origin_orientation_1** = \
- **branch_origin_orientation_2** = \
- list **branch_ang** = BRANCHING_ANGLES[scipy.random.random_integers(low=0, high=len(BRANCHING_ANGLES)-1, size=1)]
- tuple **LENGTH_DISTN_1**
- tuple **LENGTH_DISTN_2**
- **LENGTH_DISTN_1** = \
- **LENGTH_DISTN_2** = \
- tuple **fib_tmp_brnch_1** = Linear.Object.Fibre.Fibre()
- tuple **fib_tmp_brnch_2** = Linear.Object.Fibre.Fibre()
- tuple **gen_tmp** = Linear.Object.Genet.Genet()

3.24.1 Constructor & Destructor Documentation

3.24.1.1 def Linear.Process.Proc_genet.GenetProcess.__init__ (*self*)

```
Initialize the class GenetProcess
```

3.24.2 Member Function Documentation

3.24.2.1 def Linear.Process.Proc_genet.GenetProcess.generate_from_XFig_input (*self*, *data_in*)

```
Generate a genet process of type GenetProcess from an XFig input
file that contains the genet structure (lines and points).

Parameters:

Returns:
```

3.24.2.2 def Linear.Process.Proc_genet.GenetProcess.assign_parents (*self*, *list_of_points*)

```
assign_parents
```

3.24.2.3 def Linear.Process.Proc_genet.GenetProcess.assign_fibres (*self*, *list_of_fibres*)

```
assign_fibres
```

3.24.2.4 def Linear.Process.Proc_genet.GenetProcess.append_genet (*self*, *genet*)

```
append_genet
```

3.24.2.5 def Linear.Process.Proc_genet.GenetProcess.generate_from_list (*self*, *list_of_genets*)

```
generate_from_list
```

3.24.2.6 def Linear.Process.Proc_genet.GenetProcess.draw_data (*self*, *filename*, *canvas_extent*, *x_limit*, *y_limit*)

```
draw_data
```

3.24.2.7 def Linear.Process.Proc_genet.GenetProcess.generate_from_simulation (*self*, *DRAW_SIM* = 'TRUE', *FILE_PLOT_PREFIX* = "genet_sim_", *NGENET_SET* = 'RAND', *NGENET_MIN* = 1, *NGENET_MAX* = 10, *NGENET_DEN* = 'FALSE', *PARENT_POS* = 'RAND', *NSTOLON_SET* = 'RAND', *NSTOLON_MIN* = 1, *NSTOLON_MAX* = 10, *NSEGS_RAND* = 'TRUE', *NSEGS_MIN* = 1, *NSEGS_MAX* = 10, *NSEGS_KERNEL* = 'FALSE', *LENGTH_DISTN_RAND* = 'TRUE', *MIN_SEGMENT_LENGTH* = .1, *MAX_SEGMENT_LENGTH* = 100, *LENGTH_DISTN_KERNEL* = 'TRUE', *TURN_ANGLE_SET* = 'RAND', *MIN_TURN_ANGLE* = 0, *MAX_TURN_ANGLE* = .1 * scipy.pi, *ORIENTATION* = 'FALSE', *ORIENTATION_RAND* = 'TRUE', *MIN_ORIENTATION* = 0, *MAX_ORIENTATION* = 2 * scipy.pi, *ORIENTATION_KERNEL* = 'FALSE', *MIN_ORIGIN_ORIENTATION* = 0, *MAX_ORIGIN_ORIENTATION* = 2 * scipy.pi, *ORIGIN_ORIENTATION_KERNEL* = 'FALSE', *x_limit* = [0, *y_limit* = [0, *RAMET_FORM_LIKELIHOOD* = 0.01, *BRANCH_FORM_LIKELIHOOD* = 0.01, *distn_n_stolon_array* = 'FALSE', *distn_n_segs_per_fibre_array* = 'FALSE', *distn_segment_length* = 'FALSE', *distn_turn_angles* = 'FALSE', *distn_origin_orientation* = 'FALSE', *n_segs_per_order* = [.1, *parent_points* = 'FALSE', *BRANCHING_ANGLES* = 'FALSE')

```
Generate genet from simulation.
```

3.24.2.8 def Linear.Process.Proc_genet.GenetProcess.make_pattern (*self*)

```
Make pattern
```

3.24.2.9 def Linear.Process.Proc_genet.GenetProcess.probability_of_branching (*self*)

```
Find the probability of branching at each potential branching
 location
```

3.24.2.10 def Linear.Process.Proc_genet.GenetProcess.n_segs_per_order (*self*)

```
# Return a list of lists containing the number of segments
# for each stolon at each order of branching.
```

3.24.2.11 def Linear.Process.Proc_genet.GenetProcess.calc_length (*self, filename_prefix*)

```
Find the total length of all the segments and fibres
```

3.24.2.12 def Linear.Process.Proc_genet.GenetProcess.calc_length_pattern (*self, filename_prefix*)

```
Find the total length of all the segments and fibres
```

3.24.2.13 def Linear.Process.Proc_genet.GenetProcess.calc_effective_length_fibres (*self, filename_prefix*)

```
Find the total length of all the segments and fibres
```

3.24.2.14 def Linear.Process.Proc_genet.GenetProcess.calc_n_points (*self, filename_prefix*)

```
Find the total number of points
```

3.24.2.15 def Linear.Process.Proc_genet.GenetProcess.angle_of_branching (*self*)

```
# Find the angle of branching at each potential branching
# location (measured as an angle less than pi from the previous node)
```

3.24.2.16 def Linear.Process.Proc_genet.GenetProcess.distribute_random_points_ave_length (*self*, *AVE_LENGTH*, *VERBOSE* = "FALSE")

```
# Pass through the list of genet dictionaries. Find the total fibre
# length and the number of ramets.
```

3.24.2.17 def Linear.Process.Proc_genet.GenetProcess.ang_dist_analysis (*self*, *RADII*, *N_CLASSES*, *x_limit*, *y_limit*, *filename1*, *filename2_prefix*, *DRAW_PLOT*, *DRAW_INDIVIDUALS*)

```
ang_dist_analysis
```

3.24.3 Member Data Documentation

3.24.3.1 tuple Linear.Process.Proc_genet.GenetProcess.angles_distn [static]

Initial value:

```
scipy.random.uniform(
                                low=MIN_ORIENTATION,high=MAX_ORIENTATION,
                                size=NSEGS)
```

3.24.3.2 tuple Linear.Process.Proc_genet.GenetProcess.origin_orientation [static]

Initial value:

```
scipy.random.uniform(
                                low=MIN_ORIGIN_ORIENTATION,
                                high=MAX_ORIGIN_ORIENTATION,size=1)
```

3.24.3.3 list Linear.Process.Proc_genet.GenetProcess.x_origin_tmp [static]

Initial value:

```
fib_tmp_prev.segments[
                                    len(fib_tmp_prev.segments)-1]
```

3.24.3.4 list Linear.Process.Proc_genet.GenetProcess.y_origin_tmp [static]

Initial value:

```
fib_tmp_prev.segments[
                                    len(fib_tmp_prev.segments)-1]
```

3.24.3.5 tuple Linear.Process.Proc_genet.GenetProcess.BRANCH_NSEGS_1 [static]

Initial value:

```
int(
                                        kde_distn_n_segs_per_order[ii+1].resample
      (1)[0][0])
```

3.24.3.6 tuple Linear.Process.Proc_genet.GenetProcess.BRANCH_NSEGS_2 [static]

Initial value:

```
int(
                                        kde_distn_n_segs_per_order[ii+1].resample
      (1)[0][0])
```

3.24.3.7 tuple Linear.Process.Proc_genet.GenetProcess.LENGTH_DISTN_1 [static]

Initial value:

```
scipy.random.uniform(\
                                        low=MIN_SEGMENT_LENGTH,\
                                        high=MAX_SEGMENT_LENGTH,\
                                        size=BRANCH_NSEGS_1)
```

3.24.3.8 tuple Linear.Process.Proc_genet.GenetProcess.LENGTH_DISTN_2 [static]

Initial value:

```
scipy.random.uniform(\
                                        low=MIN_SEGMENT_LENGTH,\
                                        high=MAX_SEGMENT_LENGTH,\
                                        size=BRANCH_NSEGS_2)
```

The documentation for this class was generated from the following file:

- Proc_genet.py

3.25 Linear.Process.Proc_pt.PointProcess Class Reference

Public Member Functions

- def __init__

Public Attributes

- **n_points**
- **x_limit**
- **y_limit**
- **p**

3.25.1 Detailed Description

```
A class of type PointProcess.
```

3.25.2 Constructor & Destructor Documentation

3.25.2.1 def Linear.Process.Proc_pt.PointProcess.__init__ (*self*, *n_points*, *x_limit*, *y_limit*)

```
Instantiate a class of type PointProcess and distribute
the point randomly.

Parameters:
n_points
x_limit
y_limit

Returns
self.n_points
self.x_limit
self.y_limit
self.p
```

The documentation for this class was generated from the following file:

- Proc_pt.py

3.26 Linear.Process.Proc_pt.PtProc Class Reference

Public Member Functions

- def __init__
- def mk_process_uniform
- def mk_process_cluster

Public Attributes

- **n_points**
- **x_limit**
- **y_limit**
- **p**
- **pt_list**

3.26.1 Detailed Description

```
A class of type PtProc
```

3.26.2 Constructor & Destructor Documentation

3.26.2.1 def Linear.Process.Proc_pt.PtProc.__init__ (*self*, *n_points*, *x_limit*, *y_limit*)

```
Instantiate an instance of the PtProc class

Parameters:
n_points
x_limit
y_limit

Returns:
self.n_points
self.x_limit
self.y_limit
self.p
```

3.26.3 Member Function Documentation

3.26.3.1 def Linear.Process.Proc_pt.PtProc.mk_process_uniform (*self*)

```
Distribute the points randomly across the region defined
by x_limit and y_limit.

Parameters:

Returns:
```

3.26.3.2 def Linear.Process.Proc_pt.PtProc.mk_process_cluster (*self*, *seed_pt_proc*, *cluster_radius*)

```
Distribute points in a clustered arrangement.
```

```
Parameters:
seed_pt_proc
    Seed_pt_proc will provide a list of points giving the centres
    of the circles into which the process's points will fall.
cluster_radius
    The radius of the clusters.

Returns:
self.p
```

The documentation for this class was generated from the following file:

- Proc_pt.py

3.27 Linear.Process.Proc_seg.SegmentProcess Class Reference

Public Member Functions

- def __init__
- def generate_from_list
- def generate_orientation
- def generate_process
- def print_segment_proc
- def effective_length
- def distribute_random_points
- def draw_process

Public Attributes

- **n_segs**
- **x_limit**
- **y_limit**
- **seg_length**
- **max_len**
- **min_len**
- **seg**
- **angles**
- **seg_process_eff_length**
- **distributed_points**
- **cumulative_n_segs**

3.27.1 Detailed Description

```
Class of type Segment. An object of type Segment is representative of
the geometric line segment in the x-y-z space.
```

3.27.2 Constructor & Destructor Documentation

3.27.2.1 def Linear.Process.Proc_seg.SegmentProcess.__init__ (*self*, *n_segs*, *x_limit* = [0, *y_limit* = [0, *seg_length* = 1, *rand_seg_length* = 'FALSE', *max_len* = 1, *min_len* = 1)

```
Instantiate instance of class of type Segment.

Parameters
n_segs : int
x_limit=[0,10]
y_limit=[0,10]
seg_length=1
rand_seg_length='FALSE'
max_len=1
min_len=1

Returns
self.n_segs = n_segs
self.x_limit = x_limit
self.y_limit = y_limit
self.seg_length = seg_length
self.max_len = max_len
self.min_len = min_len
```

3.27.3 Member Function Documentation

3.27.3.1 def Linear.Process.Proc_seg.SegmentProcess.generate_from_list (*self*, *list_of_segs*)

```
Generate a segment process from a list of Segments.

Parameters
list_of_segs

Returns
self.seg
```

3.27.3.2 def Linear.Process.Proc_seg.SegmentProcess.generate_orientation (*self*, *orientation* = 'vonmises', *vonmises_mu* = 0, *vonmises_kappa* = 0, *marriot_psi* = 0, *marriot_K* = 0)

```
Generate a distribution of orientations for the segment process.
```

```
Parameters
orientation='vonmises'
vonmises_mu=0
vonmises_kappa=0
marriot_psi=0
marriot_K=0

Returns
self.angles : list
    A list of angles
```

3.27.3.3 def Linear.Process.Proc_seg.SegmentProcess.generate_process (*self*, *point_process* = "FALSE")

```
Generate a segment process with or without a point process.
If point_process is not supplied in the function call,
generate a random point process. Otherwise, use point_process
as the centres of the segments.

Parameters
point_process : Proc_point
    A point process.

Returns
self.seg : list
    A list of Segments
```

3.27.3.4 def Linear.Process.Proc_seg.SegmentProcess.print_segment_proc (*self*)

```
Print out the segments in the segment process

Parameters

Returns
```

3.27.3.5 def Linear.Process.Proc_seg.SegmentProcess.effective_length (*self*, *x_limit*, *y_limit*)

```
Calculate the effective length of the segment process

Parameters
x_limit : list
y_limit : list

Returns
self.seg_process_eff_length : float
```

3.27.3.6 def Linear.Process.Proc_seg.SegmentProcess.distribute_random_points (*self*, *N_RAND_PTS*, *x_limit*, *y_limit*, *VERBOSE* = "FALSE")

```
Distribute random points on the segment process

Parameters
N_RAND_PTS : int
x_limit : list
y_limit : list

Returns
self.seg[h].point_locations
self.distributed_points
```

3.27.3.7 def Linear.Process.Proc_seg.SegmentProcess.draw_process (*self*, *filename*)

```
Draw the data (fibres and points) the segment process

Parameters
filename : string

Returns
```

The documentation for this class was generated from the following file:

- Proc_seg.py

3.28 Linear.Raster.Object.Circle.Circle Class Reference

Public Member Functions

- def __init__
- def init_rad_Point

Public Attributes

- **radius**
- **x_cen**
- **y_cen**
- **circle_coords**

3.28.1 Constructor & Destructor Documentation

3.28.1.1 def Linear.Raster.Object.Circle.Circle.__init__ (*self*)

```
Initialize an instance of class Circle.
```

3.28.2 Member Function Documentation

3.28.2.1 def Linear.Raster.Object.Circle.Circle.init_rad_Point (*self, radius, x_cen, y_cen*)

```
Create a raster circle by providing a radius and centre point.

Parameters:
radius : float
x_cen : int
y_cen : int

Returns:
self.circle_coords : list
    A list of integer coordinates that identify the pixels that make
    up the circle.
```

The documentation for this class was generated from the following file:

- Raster/Object/Circle.py

3.29 Linear.Raster.Object.Collection.Collection Class Reference

Public Member Functions

- def __init__

Public Attributes

- **list_of_regions**
- **ncol**
- **nrow**

3.29.1 Detailed Description

```
Class of type Collection. A rectangular region of the study plot.
```

3.29.2 Constructor & Destructor Documentation

3.29.2.1 def Linear.Raster.Object.Collection.Collection.__init__ (*self*, *im*, *v_scale*, *h_scale*)

```
Initialize an instance of class Collection.
```

The documentation for this class was generated from the following file:

- Collection.py

3.30 Linear.Raster.Object.Fibre.Fibre Class Reference

Public Member Functions

- def **__init__**
- def **set_ID**
- def **length**
- def **rose**
- def **mean_vector**
- def **curvature**

Public Attributes

- **segments**
- **n_segs**
- **n_points**
- **stolon_ramet_index**
- **parent_ramet_index**
- **unique_pts**
- **ID**
- **static_length**
- **static_rose**
- **radii**

The documentation for this class was generated from the following file:

- Raster/Object/Fibre.py

3.31 Linear.Raster.Object.Intersection.Point Class Reference

Public Member Functions

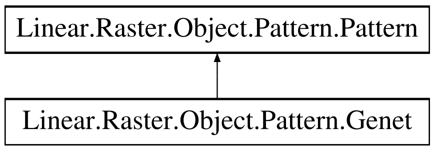
- def **__init__**

The documentation for this class was generated from the following file:

- Intersection.py

3.32 Linear.Raster.Object.Pattern.Genet Class Reference

Inheritance diagram for Linear.Raster.Object.Pattern.Genet:


Linear.Raster.Object.Pattern.Pattern
Linear.Raster.Object.Pattern.Genet

Public Member Functions

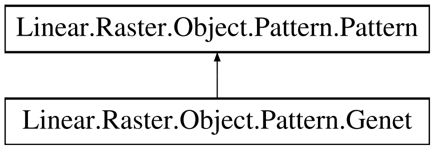
- def **__init__**

The documentation for this class was generated from the following file:

- Raster/Object/Pattern.py

3.33 Linear.Raster.Object.Pattern.Pattern Class Reference

Inheritance diagram for Linear.Raster.Object.Pattern.Pattern:


Linear.Raster.Object.Pattern.Pattern
Linear.Raster.Object.Pattern.Genet

Public Member Functions

- def **__init__**
- def **find_parent**
- def **intersect_points_fibres**

Public Attributes

- **objects**
- **no_types**
- **no_objects**
- **list_of_patterns**
- **list_of_fibres**
- **list_of_segments**
- **list_of_points**
- **ID**
- **no_points**
- **no_segments**
- **no_fibres**
- **no_patterns**
- **parent_ramet**

The documentation for this class was generated from the following file:

- Raster/Object/Pattern.py

3.34 Linear.Raster.Object.Point.Point Class Reference

Public Member Functions

- def __init__

Public Attributes

- **x**
- **y**
- **z**
- **meta1**
- **ID**
- **is_ramet**
- **is_parent**

- **is_terminus**
- **is_branch**
- **no_in**
- **no_out**
- **stolon_fibre_index**

3.34.1 Detailed Description

```
Class of type Point. A point in raster space.
```

3.34.2 Constructor & Destructor Documentation

3.34.2.1 def Linear.Raster.Object.Point.Point.__init__ (*self*, *x*, *y*, *z*, *meta1*)

```
Initialize an instance of class Point.

Parameters:
x
y
z
meta1

Returns:
self.x = float(x)
self.y = float(y)
self.z = float(z)
self.meta1 = float(meta1)
self.ID = 0
self.is_ramet = "FALSE"
self.is_parent = "FALSE"
self.is_terminus = "FALSE"
self.is_branch = "FALSE"

# Each point may have n segments coming "in" or "out"
self.no_in = 0
self.no_out = 0

# A list to hold the indices (in list of fibres) of fibres that
#   intersect the point. The fibres are in list_of_fibres
#   in the Pattern object.
self.stolon_fibre_index
```

The documentation for this class was generated from the following file:

- Raster/Object/Point.py

3.35 Linear.Raster.Object.Region.Region Class Reference

Public Member Functions

- def __init__
- def delete_data
- def to_matrix
- def **im_is_red**
- def load_red
- def find_RGB
- def find_red_fast
- def find_red
- def make_greyscale
- def make_greyscale_matrix
- def fib_prop
- def local_pixels
- def crawl_lin_feature
- def pixel_to_cart
- def test_line_set
- def k_function

Public Attributes

- **region_ID**
- **region_rowID**
- **region_colID**
- **im**
- **type**
- **ncol**
- **nrow**
- **im_mat**
- **have_matrix**
- **is_red**
- **is_red_mat**
- **have_red**
- **im_L**
- **have_L**
- **im_mat_L**
- **have_L_mat**
- **collection_cord_upper_right**
- **collection_cord_lower_right**
- **collection_cord_upper_left**

- **collection_cord_lower_left**
- **a**
- **im_greyscale**
- **im_greyscale_mat**
- **width_total_vert**
- **center_row_v_total**
- **raw_intersection_points**
- **region_intersection_points**
- **n_region_intersection_points**
- **n_test_lines**
- **len_test_lines**
- **area**
- **est_len_fib**
- **est_len_fib_area**
- **lin_feature_v_tline**
- **lin_feature_v_tline_total**
- **im_mat_crawl**
- **theta_total**
- **theta_total_mean**
- **K_hat**

3.35.1 Detailed Description

```
Class of type Region. A rectangular region of pixels.
```

3.35.2 Constructor & Destructor Documentation

3.35.2.1 def Linear.Raster.Object.Region.Region.__init__ (*self*, *im*, *type*)

```
Initialize an instance of the class Region.

Parameters:
im
type

Returns:

# IF type = RGB
# im_mat is a n x m x 3 RGB array with image data

# IF type = L
# im_mat is a n x m x 1 greyscale array with image data
```

3.35.3 Member Function Documentation

3.35.3.1 def Linear.Raster.Object.Region.Region.delete_data (*self*)

```
Delete data associate the the instance of Region.
```

3.35.3.2 def Linear.Raster.Object.Region.Region.to_matrix (*self*)

```
Convert the PIL im to a matrix.
```

3.35.3.3 def Linear.Raster.Object.Region.Region.load_red (*self, filename*)

```
Load a colour-processed image where the red hues
have been extracted.
```

3.35.3.4 def Linear.Raster.Object.Region.Region.find_RGB (*self, R, G, B*)

```
Find the pixels [row, column] having
a specific R, G, B value.
```

3.35.3.5 def Linear.Raster.Object.Region.Region.find_red_fast (*self, R_G_per_diff, R_B_per_diff, image_out_filename*)

```
Find the pixels with a red hue that is defined by the
parameters.
```

3.35.3.6 def Linear.Raster.Object.Region.Region.find_red (*self, R_G_per_diff, R_B_per_diff, image_out_filename*)

```
Find the pixels with a red hue that is defined by the
parameters.
```

3.35.3.7 def Linear.Raster.Object.Region.Region.make_greyscale (*self, im*)

```
Convert the image data (im) to greyscale.

Parameters:

Returns:
```

3.35.3.8 def Linear.Raster.Object.Region.Region.make_greyscale_matrix (*self*, *im*)

```
Make the greyscal image into a matrix.

Parameters:

Returns:
```

3.35.3.9 def Linear.Raster.Object.Region.Region.fib_prop (*self*, *scale*, *bg*, *cen_col*, *dev*, *INT_COORD*, *FIB_WIDTH*, *FIB_THETA*, *VERTICAL*, *HORIZONTAL*, *radius*, *outputfilename*, *filename_cumulative*, *crawlfilename*)

```
Find the properties of the fibres contained in the region.



#
# Write the number, location of intersection points, width and
# orientation of fibres into file.

# Method uses a grayscale version of the image data. There
# is little that can be done with RGB (right now).
#
# Check to see if the Region already has converted the 'im'
# to im_L (linear or grayscale).
# If not, call the convert method.
```

3.35.3.10 def Linear.Raster.Object.Region.Region.local_pixels (*self*, *pixel_row*, *pixel_col*)

```
Find the adjacent pixels (greyscale) to that given by
pixel_row and pixel_column.
#
# Return:
#   A list of two lists: row and column coordinates giving
#   the adjacent pixels having a value between rang1
#   and rang2.
#

# Works on:
#   self.im_mat_L

# Check to see if the Region already has converted the 'im'
# to im_L (linear or grayscale).
# If not, call the convert method.
```

3.35.3.11 def Linear.Raster.Object.Region.Region.crawl_lin_feature (*self*, *pixel_row*, *pixel_col*, *rotation*, *orientation*, *rang1*, *rang2*, *side*)

```
Crawl the edge of a linear feature.
```

```
# rotation:
#   cw = clockwise
#   ccw = counter clockwise

# orientation:
#   The orientation of the test line that has produced
#   the intersection point.
#
#   hor = horizontal test line
#   ver = vertical test line

# Get all the adjacent pixels:
#   adj_pixels = [adj_row, adj_col]

# side:
#   The side of the linear feature that the starting pixel
#   is on (upper, lower).
#   upper = the first pixel encountered
#   lower = the last pixel encountered
```

3.35.3.12 def Linear.Raster.Object.Region.Region.pixel_to_cart (*self*, *nrow*, *col*, *row*)

```
Convert pixel coordinates into cartesian coordinates.
# Pixel row values INCREASE as we move "DOWN" the image,
# but cartesian coords require row (y) values INCREASE as
# we move "UP" the image.
#
# X coord is not affected, but will take and return as
# a pair to the Y coord.
#
# Have to take the Y coord and measure the distance from
# its row location to the bottom of the image matrix.
```

3.35.3.13 def Linear.Raster.Object.Region.Region.test_line_set (*self*, *region*, *orientation*, *scale*, *bg*, *dev*, *input_filename* = None, *output_filename* = None)

```
Get all the points that make up the center of the line.
# The orientation does not matter as we are sampling at a
# scale of 1 (one).
#
# That is, we are finding the centre of the linear features
# and discarding the remainder.
```

3.35.3.14 def Linear.Raster.Object.Region.Region.k_function (*self*, *distance_class*, *list_of_points*, *xmax*, *xmin*, *ymax*, *ymin*)

```
Calculate the k-function for the fibre process.
The analysis is based on points sampled from the image.
```

The documentation for this class was generated from the following file:

- Raster/Object/Region.py

3.36 Linear.Raster.Object.Segment.Segment Class Reference

Public Member Functions

- def __init__
- def max_x
- def min_x
- def max_y
- def min_y
- def length
- def extend
- def parallel_extended
- def parallel
- def mk_polygon_rect

Public Attributes

- **points**
- **x1**
- **y1**
- **z1**
- **metaA**
- **x2**
- **y2**
- **z2**
- **metaB**
- **point1**
- **point2**
- **angle**
- **m**
- **equation**
- **static_length**
- **ep_x1**
- **ep_y1**
- **ep_x2**
- **ep_y2**
- **p1_ep1_pt_ex**

- **p1_ep2_pt_ex**
- **p2_ep1_pt_ex**
- **p2_ep2_pt_ex**
- **par_seg_1_ex**
- **par_seg_2_ex**
- **p1_ep1_pt**
- **p1_ep2_pt**
- **p2_ep1_pt**
- **p2_ep2_pt**
- **par_seg_1**
- **par_seg_2**
- **polygon_rect**

3.36.1 Detailed Description

```
Class of type Segment. A segment in raster space.
```

3.36.2 Constructor & Destructor Documentation

3.36.2.1 def Linear.Raster.Object.Segment.Segment.__init__ (*self, Point_A, Point_B*)

```
Initialize an instance of class Segment.

Parameters:
Point_A : Point
Point_B : Point

Returns:
```

3.36.3 Member Function Documentation

3.36.3.1 def Linear.Raster.Object.Segment.Segment.max_x (*self*)

```
Find the maximum x of the segment endpoints.
```

3.36.3.2 def Linear.Raster.Object.Segment.Segment.min_x (*self*)

```
Find the minimum x of the segment endpoints.
```

3.36.3.3 def Linear.Raster.Object.Segment.Segment.max_y (*self*)

```
Find the maximum y of the segment endpoints.
```

3.36.3.4 def Linear.Raster.Object.Segment.Segment.min_y (*self*)

```
Find the minimum y of the segment endpoints.
```

3.36.3.5 def Linear.Raster.Object.Segment.Segment.length (*self*)

```
Calculate length of segment
```

3.36.3.6 def Linear.Raster.Object.Segment.Segment.extend (*self, rad*)

```
Extend the length of a segment.

Parameters:
rad

Returns:
```

3.36.3.7 def Linear.Raster.Object.Segment.Segment.parallel_extended (*self, rad*)

```
Make parallel segments for the extended segment.

Parameters:

Returns:
```

3.36.3.8 def Linear.Raster.Object.Segment.Segment.parallel (*self, rad*)

```
Make parallel segments.

Parameters:

Returns:
```

3.36.3.9 def Linear.Raster.Object.Segment.Segment.mk_polygon_rect (*self*, *rad*)

```
Make a rectangle.

Parameters:

Returns:
```

The documentation for this class was generated from the following file:

- Raster/Object/Segment.py

3.37 Linear.Raster.Object.Segment.Segment_Polygon_Rect Class Reference

Public Member Functions

- def __init__
- def seg_intercept

Public Attributes

- **seg_poly_rect**
- **intercepts**

3.37.1 Constructor & Destructor Documentation

3.37.1.1 def Linear.Raster.Object.Segment.Segment_Polygon_Rect.__init__ (*self*, *seg1*, *seg2*)

```
Initialize an instance of the class Segment_Polygon_Rect.

Parameters:

Returns:
```

3.37.2 Member Function Documentation

3.37.2.1 def Linear.Raster.Object.Segment.Segment_Polygon_Rect.seg_intercept (*self*, *seg*)

```
Initialize an instance of the class Segment_Polygon_Rect.
```

`Parameters:`

`Returns:`

The documentation for this class was generated from the following file:

- Raster/Object/Segment.py

3.38 Linear.Raster.Object.Test_Line.Test_Line Class Reference

Public Member Functions

- def __init__
- def get_line_efficient
- def get_line
- def remove_border
- def remove_border_2
- def find_local_maxima
- def find_local_minima
- def find_width
- def convert_px_metric
- def angular_from_width
- def write_angular_to_file
- def find_num_maxima
- def get_intersections
- def get_intersections_2
- def del_im_data

Public Attributes

- **orientation**
- **coordinate**
- **test_line**
- **column**
- **row**
- **max_value**
- **max_coord_row**
- **max_coord_col**
- **max_coord_col_plot**
- **min_value**
- **min_coord_row**

- **min_coord_col**
- **min_coord_col_plot**
- **width_column_values**
- **width_column_values_metric**
- **angular_deviation**
- **num_maxima**
- **intersections_center**
- **intersections_start**
- **intersections_end**
- **fibre_width**
- **n_intersections**

3.38.1 Constructor & Destructor Documentation

3.38.1.1 def Linear.Raster.Object.Test_Line.Test_Line.__init__ (*self*)

```
Initialize an instance of class Test_Line.
```

3.38.2 Member Function Documentation

3.38.2.1 def Linear.Raster.Object.Test_Line.Test_Line.get_line_efficient (*self, im, coordinate, orientation, start, stop*)

```
Get all the pixels that make up a straight test line across the
plot area.
```

3.38.2.2 def Linear.Raster.Object.Test_Line.Test_Line.get_line (*self, region, coordinate, orientation, start, stop*)

```
Get all the pixels that make up a straight test line across the
plot area.

# region:
#   The image region to work on. Must be a
#   matrix (e.g. reg.im_mat).

# coordinate:
#   The row or column from which to start
#   the test line.

# orientation:
#   'VERTICAL' or 'HORIZONTAL'
#   The test line should be either a row
#   (HOR) or column (VERT).

# start/stop:
```

```
#   The length of the test line. The pixel
#   values that delimit the line.
```

3.38.2.3 def Linear.Raster.Object.Test_Line.Test_Line.remove_border (*self*, *interior_border_width =* `3`, *interior_border_colour =* `255`, *background_colour =* `0`)

```
Remove the border from a pre-processed image.
```

3.38.2.4 def Linear.Raster.Object.Test_Line.Test_Line.remove_border_2 (*self*, *interior_border_col =* `255`, *exterior_border_col =* `0`)

```
Remove the border from a pre-processed image.
```

3.38.2.5 def Linear.Raster.Object.Test_Line.Test_Line.find_local_maxima (*self*)

```
Find the local maxima.
# Find local maxima
#
# Cycle through the test line data
# (1) Find matrix coordinates of each grayscale intensity maximum
# (2) Find the width of the fibre around each intensity maximum

# The maximum intensity, followed by a minimum will constitute a
# fibre centre. The minimum intensity will constitute a reset of the
# fibre counter.

# The test line (self.test_line) is a one-dimensional vector (array)

# We don't need the upper level array
```

3.38.2.6 def Linear.Raster.Object.Test_Line.Test_Line.find_local_minima (*self*)

```
Find the local minima.

# Find local maxima
#
# Cycle through the test line data
# (1) Find matrix coordinates of each grayscale intensity maximum
# (2) Find the width of the fibre around each intensity maximum

# The maximum intensity, followed by a minimum will constitute a
# fibre centre. The minimum intensity will constitute a reset of the
# fibre counter.

# The test line (self.test_line) is a one-dimensional vector (array)

# We don't need the upper level array
```

3.38.2.7 def Linear.Raster.Object.Test_Line.Test_Line.find_width (*self*)

```
Find distance between local minima (width)
```

3.38.2.8 def Linear.Raster.Object.Test_Line.Test_Line.convert_px_metric (*self, conversion_factor*)

```
Convert pixels.
```

3.38.2.9 def Linear.Raster.Object.Test_Line.Test_Line.angular_from_width (*self, true_width*)

```
Calculate the angular direction of the line base on the actual
(expected) width.
```

3.38.2.10 def Linear.Raster.Object.Test_Line.Test_Line.write_angular_to_file (*self, filename*)

```
Write angular directions to file.
```

3.38.2.11 def Linear.Raster.Object.Test_Line.Test_Line.find_num_maxima (*self*)

```
Find the maximum.
```

3.38.2.12 def Linear.Raster.Object.Test_Line.Test_Line.get_intersections (*self, bg*)

```
Get the intersections from the test line.
Ignore the bg pixel values.

# bg:
#   Two element list holding the min and
#   max values that delimit the greyscale
#   values to be considered the background.
# bg[0]: background min greyscale value
# bg[1]: background max greyscale value
```

3.38.2.13 def Linear.Raster.Object.Test_Line.Test_Line.get_intersections_2 (*self, bg*)

```
Get the intersections from the test line.
Ignore the bg pixel values.

# bg:
#   Two element list holding the min and
#   max values that delimit the greyscale
#   values to be considered the background.
# bg[0]: background min greyscale value
# bg[1]: background max greyscale value
```

3.38.2.14 def Linear.Raster.Object.Test_Line.Test_Line.del_im_data (*self*)

```
Delete the image data.
```

The documentation for this class was generated from the following file:

- Test_Line.py

3.39 Linear.Raster.Object.Test_Line.Test_Line_efficient Class Reference

Public Member Functions

- def **__init__**
- def get_line_efficient

Public Attributes

- **orientation**
- **coordinate**
- **test_line**
- **column**
- **row**

3.39.1 Detailed Description

```
An efficient version of the class Test_Line.
```

3.39.2 Member Function Documentation

3.39.2.1 def Linear.Raster.Object.Test_Line.Test_Line_efficient.get_line_efficient (*self, im, coordinate, orientation, start, stop*)

```
Get line. Efficient version.
# im_name:
```

```
#   The PIL image in which the test line will
#   be located.

# coordinate:
#   The row or column from which to start
#   the test line.

# orientation:
#   'VERTICAL' or 'HORIZONTAL'
#   The test line should be either a row
#   (HOR) or column (VERT).

# start/stop:
#   The length of the test line. The pixel
#   values that delimit the line.
```

The documentation for this class was generated from the following file:

- Test_Line.py

3.40 Linear.Raster.Object.Test_Line_Set.Test_Line_Set Class Reference

Public Member Functions

- def __init__

Public Attributes

- **intersections_center**
- **intersections_start**
- **intersections_end**
- **fibre_width**
- **n_intersection_points**
- **n_test_lines**
- **len_test_lines**

3.40.1 Constructor & Destructor Documentation

3.40.1.1 def Linear.Raster.Object.Test_Line_Set.Test_Line_Set.__init__ (*self*, *region*, *orientation*, *scale*, *bg*, *dev*)

```
Initialize an instance of the class Test_Line_Set. This is a set
of straight test lines.
```

```
Parameters:
region
orientation
scale
bg
dev

Returns:
```

The documentation for this class was generated from the following file:

- Test_Line_Set.py

3.41 Linear.Raster.Object.Test_Line_Set.Test_Line_Set_eff Class Reference

Public Member Functions

- def **__init__**
- def **find_local_maxima_set**
- def **find_local_minima_set**
- def **print_test_line_matrix**
- def **find_width_set**
- def **get_intersections_set**
- def **get_intersections_2_set**
- def **convert_px_metric_set**
- def **angular_from_width_set**
- def **remove_border_set**
- def **remove_border_2_set**
- def **find_num_maxima_set**
- def **write_angular_to_file_set**
- def **del_im_data_set**

Public Attributes

- **orientation**
- **list_of_test_lines**
- **n_test_lines**
- **len_test_lines**
- **orig_len_test_lines**
- **max_value_list**

- **max_coord_row_list**
- **max_coord_col_list**
- **max_coord_col_plot_list**
- **min_value_list**
- **min_coord_row_list**
- **min_coord_col_list**
- **min_coord_col_plot_list**
- **width_column_values_list**
- **intersections_center_list**
- **intersections_start_list**
- **intersections_end_list**
- **fibre_width_list**
- **n_intersections_list**
- **width_column_values_list_metric**
- **angular_deviation_list**
- **num_maxima_list**

3.41.1 Detailed Description

```
An efficient version of the class Test_Line_Set.
```

The documentation for this class was generated from the following file:

- Test_Line_Set.py

3.42 Linear.Raster.Object.Test_Line_Set.Test_Line_Set_new Class Reference

Public Member Functions

- def __init__
- def find_local_maxima_set
- def find_local_minima_set
- def print_test_line_matrix
- def find_width_set
- def get_intersections_set
- def get_intersections_2_set
- def convert_px_metric_set
- def angular_from_width_set
- def remove_border_set
- def remove_border_2_set

- def find_num_maxima_set
- def write_angular_to_file_set
- def del_im_data_set

Public Attributes

- **list_of_test_lines**
- **n_test_lines**
- **len_test_lines**
- **orig_len_test_lines**
- **max_value_list**
- **max_coord_row_list**
- **max_coord_col_list**
- **max_coord_col_plot_list**
- **min_value_list**
- **min_coord_row_list**
- **min_coord_col_list**
- **min_coord_col_plot_list**
- **width_column_values_list**
- **intersections_center_list**
- **intersections_start_list**
- **intersections_end_list**
- **fibre_width_list**
- **n_intersections_list**
- **width_column_values_list_metric**
- **angular_deviation_list**
- **num_maxima_list**

3.42.1 Constructor & Destructor Documentation

3.42.1.1 def Linear.Raster.Object.Test_Line_Set.Test_Line_Set_new.__init__ (*self*, *region*, *coordinate*, *orientation*, *start*, *stop*, *scale*)

```
Initialize an instance of the class Test_Line_Set. This is a set
of straight test lines.

# region:
#   The image region to work on. Must be a
#   matrix (e.g. reg.im_mat).

# coordinate:
#   The row or column from which to start
#   the first test line.

# orientation:
```

```
#   'VERTICAL' or 'HORIZONTAL'
#   The test line should be either a row
#   (HOR) or column (VERT).

# start/stop:
#   The length of the test line. The pixel
#   values that delimit the line.

# distance:
#   The range of pixels across which we
#   disperse our test lines.
```

3.42.2 Member Function Documentation

3.42.2.1 def Linear.Raster.Object.Test_Line_Set.Test_Line_Set_new.find_local_maxima_set (*self*)

```
Find local maxima
#
# Cycle through the test line data
# (1) Find matrix coordinates of each grayscale intensity maximum
# (2) Find the width of the fibre around each intensity maximum

# The maximum intensity, followed by a minimum will constitute a
# fibre centre. The minimum intensity will constitute a reset of the
# fibre counter.
```

3.42.2.2 def Linear.Raster.Object.Test_Line_Set.Test_Line_Set_new.find_local_minima_set (*self*)

```
# Find local maxima
#
# Cycle through the test line data
# (1) Find matrix coordinates of each grayscale intensity maximum
# (2) Find the width of the fibre around each intensity maximum

# The maximum intensity, followed by a minimum will constitute a
# fibre centre. The minimum intensity will constitute a reset of the
# fibre counter.
```

3.42.2.3 def Linear.Raster.Object.Test_Line_Set.Test_Line_Set_new.print_test_line_matrix (*self, filename*)

```
Print the test line matrix
```

3.42.2.4 def Linear.Raster.Object.Test_Line_Set.Test_Line_Set_new.find_width_set (*self*)

```
Find distance between local minima (width)
```

3.42.2.5 def Linear.Raster.Object.Test_Line_Set.Test_Line_Set_new.get_intersections_set (*self, bg*)

```
Find intersections
```

3.42.2.6 def Linear.Raster.Object.Test_Line_Set.Test_Line_Set_new.get_intersections_2_set (*self, bg*)

```
Find intersections
```

3.42.2.7 def Linear.Raster.Object.Test_Line_Set.Test_Line_Set_new.convert_px_metric_set (*self, conversion_factor*)

```
Convert the width
```

3.42.2.8 def Linear.Raster.Object.Test_Line_Set.Test_Line_Set_new.angular_from_width_set (*self, true_width*)

```
Find the orientation of the linear feature from the width for the
test line set.
```

3.42.2.9 def Linear.Raster.Object.Test_Line_Set.Test_Line_Set_new.remove_border_set (*self, interior_border_width =* `3`*, interior_border_colour =* `255`*, background_colour =* `0`)

```
Remove the border from a pre-processed image for the test line
set.
```

3.42.2.10 def Linear.Raster.Object.Test_Line_Set.Test_Line_Set_new.remove_border_2_set (*self, interior_border_col*)

```
Remove the border from a pre-processed image for the test line
set.
```

3.42.2.11 def Linear.Raster.Object.Test_Line_Set.Test_Line_Set_new.find_num_maxima_set (*self*)

```
Find the maximum value for the test line set.
```

3.42.2.12 def Linear.Raster.Object.Test_Line_Set.Test_Line_Set_new.write_angular_to_file_set (*self, filename*)

```
Write angle measurements to file for the test line set.
```

3.42.2.13 def Linear.Raster.Object.Test_Line_Set.Test_Line_Set_new.del_im_data_set (*self*)

```
Delete the test line data from memory for the test line set.
```

The documentation for this class was generated from the following file:

- Test_Line_Set.py