

©Copyright 2008
Maureen C. Kennedy

Multi-objective Optimization for Ecological Model Assessment and Theory Development

Maureen C. Kennedy

A dissertation submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

University of Washington

2008

Program Authorized to Offer Degree:
Quantitative Ecology and Resource Management

University of Washington
Graduate School

This is to certify that I have examined this copy of a doctoral dissertation by

Maureen C. Kennedy

and have found that it is complete and satisfactory in all respects,
and that any and all revisions required by the final
examining committee have been made.

Chair of the Supervisory Committee:

E. David Ford

Reading Committee:

E. David Ford

Thomas Hinckley

Mark Kot

Date: _____

In presenting this dissertation in partial fulfillment of the requirements for the doctoral degree at the University of Washington, I agree that the Library shall make its copies freely available for inspection. I further agree that extensive copying of the dissertation is allowable only for scholarly purposes, consistent with "fair use" as prescribed in the U.S. Copyright Law. Requests for copying or reproduction of this dissertation may be referred to ProQuest Information and Learning, 300 North Zeeb Road, Ann Arbor, MI 48106-1346, 1-800-521-0600, to whom the author has granted "the right to reproduce and sell (a) copies of the manuscript in microform and/or (b) printed copies of the manuscript made from microform."

Signature _____
Date _____

University of Washington

Abstract

Multi-objective Optimization for Ecological Model Assessment and Theory Development

Maureen C. Kennedy

Chair of Supervisory Committee:

Professor E. David Ford

College of Forest Resources, Quantitative Ecology and Resource Management

The motivation for building ecological process models is to synthesize observations and explore hypotheses for system functions. In the course of ecological research it is common to choose a trajectory for a research program in order to explain an observed phenomenon, often at the exclusion of other possible explanations. In particular, in optimality studies a single function is often the focus of the optimization. Organisms are rarely optimal for a single measure, and more often represent tradeoffs among competing requirements for growth and survival. We require multiple objectives to be optimized simultaneously in order to make substantial progress in the process of ecological model assessment, and in the use of ecological models in optimality theory development. The multi-objective optimization utilizes Pareto optimality, wherein the non-dominated set of Pareto optimal solutions is generated.

A dynamic simulation model is developed to examine the process of branch morphogenesis in *P. menziesii*, and multi-objective optimization conducted to assess the model structure and evaluate how the morphology compensates for size constraints observed in old-growth systems. The parameters and independent variables in the model

structure represent postulates for the process of morphogenesis in *P. menziesii*. The theoretical optimality objectives show the tradeoffs among major branch growth constraints and functions; the observed morphology emerges in the optimization results as a compromise between hydraulic constraints and foliage display.

The model of plant form and function presented demonstrates how, through the synthesis of multiple phenomena, multi-objective optimization for process models can be a key tool in theory development for ecological systems. The conclusions of the process model are bounded by the context of the model structure, parameters and objectives. The model shows that the old-growth branch morphology is not optimal for any single constraint; the more likely explanation is that it compensates for multiple constraints acting simultaneously in the system. These results are tempered by the bounds of the model structure and they are presented in that context. The modeling results should be integrated with empirical investigation of the physiological processes relevant to branch growth in this and related morphologies.

Table of Contents

	Page
List of Figures.....	iii
List of Tables.....	iv
Chapter I: Introduction	
Motivation.....	1
Optimality theory in biology.....	4
The Pareto optimal frontier.....	6
Methodology.....	7
Example system.....	9
Chapter II: Two-objective optimization for the assessment of a morphological model of branch development and reiteration in <i>Pseudotsuga menziesii</i>	
Introduction.....	11
Plant modeling.....	11
Biological background.....	13
Ecological question.....	15
Summary of Kennedy et al. (2004), simulation of branch growth in <i>P. menziesii</i>	18
Two-objective binary assessment.....	21
Discussion.....	32
Summary of progress in methodology.....	35
Chapter III: Model growth functions require natural history background and biological justification	
Introduction.....	36
Modeling questions.....	39
Optimization objectives.....	41
Model growth functions.....	53
Model bounds.....	69
Discussion	70
Chapter IV: Multi-objective assessment of BRANCHPRO2 uncovers inadequacies in both model process and mathematical structures	
Introduction.....	73
Assessment optimization searches.....	76
Discussion.....	97
Chapter V: <i>P. menziesii</i> old-growth branch morphology emerges as a trade-off among competing growth requirements, with hydraulic path length as the dominant constraint	
Introduction.....	103
Quantitative measures.....	105
Question A.1 Major system constraints.....	107
Question A.2 Relative performance of branch architectures.....	117
Question A.3 <i>P. menziesii</i> compensation.....	134

	Question B.1 Relative importance of branch structural characteristics.....	135
	Discussion.....	136
Chapter VI:	Multi-objective optimization for process models allows for the synthesis of multiple theories that can guide further empirical investigation	
	Multi-objective optimization, Pareto optimality and theory development.....	148
	Morphological compensation (1).....	152
	Bud release from suppression (2).....	154
	SCU development and morphogenesis (3).....	154
	Longevity in <i>P. menziesii</i>	155
	My “truth in advertising”.....	158
	Modeling implications.....	161
	Further theory development: empirical observations.....	164
	Conclusions.....	166
References.....		168
Appendix A:	Multi-objective optimization.....	179
Appendix B:	Source code for BRANCHPRO2 and BRANCHPRO3	
	Branch.h.....	181
	ParamObjDef.h.....	184
	ran.h.....	185
	scu.h.....	185
	brMain.cpp.....	189
	brMaintenance.cpp.....	190
	critCalc.cpp.....	200
	misc.cpp.....	206
	ranNum.cpp.....	206
	SCUmaint.cpp.....	209

List of Figures

Figure Number	Page
1.1	Non-dominance.....6
1.2	Pareto Optimal Model Assessment Cycle.....8
2.1.	Architectural model of <i>P. menziesii</i>17
2.2.	Expected number of daughter shoots.....18
2.3.	Pareto optimal parameter distributions.....25
2.4.	Parameter scatter plots.....28
2.5.	Example branch maps.....30
3.1.	Pareto optimal frontier, binary and continuous errors.....38
3.2.	Shoot order, path length and foliage overlap.....47
3.3.	Saturating function.....55
4.1.	Parameter distributions for optimization Series 1, r function.....78
4.2.	Foliage overlap branch map.....82
4.3.	Foliage overlap saturating function84
4.4.	Series 2 parameter distribution for p_{hydr}87
4.5.	Series 4 parameter densities for r_0 and r_{gen}92
4.6.	Series 4 example branch maps.....94
4.7.	ParetoAll parameter distributions.....95
4.8.	ParetoAll.145 parameter distributions.....96
5.1.	Theoretical objectives v. minimum r_{gen}109
5.2.	Theoretical objective tradeoffs Part1.....114
5.3.	Theoretical objectives box plots.....116
5.4.	Parameter box plots for the r function.....118
5.5.	Parameter box plots for the p function.....119
5.6.	Median g^* v. theoretical objectives.....121
5.7.	$E(k)$ v. n_{over} for Yr90.....123
5.8.	$E(k)$ v. n_{over} for Yr145.....124
5.9.	$E(k)$ v. t_{bud} for Part1.....125
5.10.	$E(k)$ v. t_{bud} for Part2.....126
5.11.	Timing of epicormic initiation.....128
5.12.	SCU turnover.....129
5.13.	Foliated shoot levels for long-term simulations.....132
5.14.	Shoots per SCU for long-term simulations.....133
5.15.	<i>A. grandis</i> branch map.....137

List of Tables

Table Number	Page
2.1. Growth processes in BRANCHPRO.....	19
2.2. Two-objective optimization searches.....	23
2.3. Parameter examples.....	29
3.1. Optimization objectives.....	44
3.2. Load objective examples.....	50
3.3. Independent variables.....	58
3.4. Effect of hydraulic constriction on bifurcation.....	66
3.5. Computational model bounds.....	69
4.1. Series 1 allowable parameter ranges for the r function.....	77
4.2. Half-saturation constants.....	84
4.3. BRANCHPRO3 parameters and variables.....	88
5.1. Series 4 simulated branch examples.....	111
5.2. Allowable parameter ranges for Part1 and Part2.....	112
5.3. Regression slopes of theoretical objectives v. SCUs.....	113
5.4. Summary of g^*	120
5.5. Summary of Rb_{sim}	127
5.6. Proportion of correct shoot levels.....	131
A.1. Non-dominance example, binary errors.....	180
A.2. Non-dominance example, continuous errors.....	180
B.1. BRANCHPRO3 source code file list.....	181

Acknowledgements

I would like to thank my family and friends who have provided support throughout this process: Kortney, Mom, Dad, Colleen, Katie, Luke, Rob, Kristen, Justin, Brenden and Noelle. The dissertation could not have been completed without the advice and guidance of my advisor E. David Ford. I'd like to thank the Quantitative Ecology and Resource Management program for providing me a home during the time it took to complete this degree, especially Joanne Besch and Loveday Conquest. My colleagues, past and present, in the canopy dynamics lab provided a constant source of insight: Joel Reynolds, Roaki Ishii, Marianne Turley, Craig Aumann, Margaret Harris, Rie Komuro, Zoe Edelstein, Bert Loosmore, Derek McClure and Ziya Ma. The Center for Quantitative Science provided valuable teaching experience. Funding for part of this work was provided by a grant from the Mellon Foundation. Support for a related project was given through a joint contract with the USFS in Wenatchee. Valuable advice was given by my committee members, Tom Hinckley, Mark Kot and Charles Laird.

Dedication

For Kortney.

Chapter I

Introduction

Motivation

Ecological process models are developed with a structure intended to mimic an ecological system. Through simulation studies, the emergent properties of the model are used to construct explanations in ecology (Ford 2000, Chapter 12). A process model can be used to synthesize observations and explore postulates for the functioning of the system, and then to guide further empirical observation (Schmitz 2000; Schmitz 2001). Models are also used as optimality tools in biology to construct explanations for the form and function of organisms. Thus, models have great promise to improve ecological theories, yet they are not living up to that potential to provide explanations in ecology (Grimm 1999). The main reason for the gap between the potential of ecological models and the reality of their role in improving theories is the lack of a formal structure for their development and assessment. The simple "fit to data" approach, with an ad hoc analysis such as sensitivity analysis, is insufficient to cope with the size and complexity of most ecological models. For many observed systems there are groups that advocate theories that become competing explanations for a common observation. This tendency to polarize to single explanations severely limits progress in ecology because it is more likely that, rather than a single explanation, multiple mechanisms drive ecological processes (e.g., Franklin et al. 1987). A method that synthesizes multiple phenomena and integrates theories is required to make progress in the explanatory power of ecological process models.

In this dissertation, I will demonstrate that multi-objective optimization for ecological process models is a key tool in the process of theory development for biological systems. When such a model is considered in theory development its conclusions are bounded by the context of the process structure, parameters and objectives. For example, a standard approach in hydrological modeling is to fit a given model by utilizing a single objective (Gupta et al. 1998), which is usually a statistically convenient measure such as the likelihood or mean squared error (MSE). This requires

that the chosen model fulfills additional requirements beyond being the “best fit”; that is, the model errors are independent and identically distributed (*iid*) and follow some probability distribution. Often, this distribution is chosen for mathematical convenience rather than the model errors actually following some inherent probability structure (Gupta et al. 1998; Beven 2006). Even if such assumptions are met, a consequence of this approach is that the current model structure is assumed correct and that the parameter set that optimizes the single objective is the best possible system representation (it is “optimal”; Beven 2006). Yet, in complex systems models with many parameters, the optimum is rarely well-defined (Beven 2006), with the possibility that within reasonable limits multiple solutions fit the data equally well (the problem of non-uniqueness). When a single model is optimized for a single performance measure, the model structure that is chosen to be fit is based on the modeler’s preconceptions of how the system works. When the model is optimized for a single measure and the best fit is found, then what is achieved is a convenient, but unreliable, “confirmation” of those preconceptions. This is true when the model is considered “optimized”, yet that status is gained only for a parameter set relative to other possible parameter sets for the same model structure. The only way to avoid such a modeling pitfall is to thoroughly interrogate the model structure itself, and to evaluate *how* the model is optimizing the objectives.

This process of assessment, or the interrogation of model structure, is difficult when a single measure is used to optimize the model. In such a case, the model parameters could shift in order to accommodate the fit to the data, rather than represent meaningful quantities. This procedure provides no guidance to detect and improve model inadequacies because it provides a relative measure of model performance against just a single objective. It provides means to compare performance relative to only that measure, but for complex ecological process models there are multiple phenomenon that are relevant to model performance. The use of multiple objectives allows us to compare model structures with respect to how they are able to satisfy multiple requirements (Reynolds and Ford 1999). The multiple objectives provide valuable insight into how the model is able to match the observed pattern, and if this insight is not incorporated into theory development, then explanations provided by the model are incomplete.

Furthermore, if the process structure itself is not assessed and evaluated in the context of the assessment objectives, then the conclusions and inference from the model should be suspect. We require a method to integrate the parameter space, the objectives for assessment and the model's process structure, and a framework to facilitate interpretation of the results.

A process model is inherently multivariate because within its structure are representations of the components of the system the model is developed to emulate. The model generates multiple outputs, which includes intermediate results that are hidden in the “black box” of the model structure, and a myriad of system features can be chosen to assess model and system performance. If focus is on a single feature and other relevant measures are ignored, then valuable and pertinent information is lost in the assessment process (Reynolds and Ford 1999). Multiple objectives allow a framework around which the model structure itself can be assessed and competing structures compared (Reynolds 1996; Reynolds and Ford 1999). In the context of optimality theory development, research suggests that organisms are rarely, if ever, optimal for a single feature (Honda and Fisher 1978; Honda and Fisher 1979; Farnsworth and Niklas 1995; Rothley et al. 1997), and the connection between any optimal form and the proposed biological (or ecological) function(s) depends entirely on the structure of the model and objectives utilized to perform the study. It is imperative to assess both the objectives and the model itself in order to draw appropriate conclusions, and to do so the analysis should incorporate both empirical observations and theoretical objectives. In this dissertation I demonstrate that theoretical synthesis from process-based models depends on the entire system of model structure, parameters, empirical objectives and theoretical objectives, and that effective synthesis requires simultaneous optimization of multiple objectives.

In the next section I describe the kinds of models this method is designed for, and the issues that have prevented their use in effectively guiding theoretical syntheses. I then outline other suggestions for model-building that will be incorporated into this method.

Optimality theory in biology

Models are often used as optimization tools to explain the form and function of organisms (Maynard Smith 1978). The model is used to quantify the optimal form for a particular organism, and then the observed organism is evaluated for whether it matches the optimal form. Such studies are taken under the premise that evolution works to optimize (Smith 1978; Gould and Lewontin 1979; Parker and Smith 1990), although they do not rely on the assumption that organisms themselves are optimal (Parker and Smith 1990). If the organism does not match the proposed optimal state, then explanations are drawn to explain why. There are many criticisms of the use of optimization to explain organism form and function: posterior alternatives are proposed when optimality failed, organisms should not be expected to be optimal for any given trait, and alternative explanations and techniques should be utilized (Gould and Lewontin 1979). These criticisms center on the issue that, in the process of forming posterior explanations for why organisms are not optimal, the gap between the empirical observations and the theoretical model tends to widen. Gould and Lewontin (1979) call this “telling stories”, in which the rejection of one explanation simply leads to its replacement by another of the same kind, without regard to alternative explanations. They claim that the criteria for acceptance of these stories are loose, and that researchers consider their work done when the explanation seems plausible. These do not rely on a formal assessment of the explanation.

Despite these criticisms, optimality is an important tool in the construction of explanations in biological systems. It is precisely when the organisms fail to meet the proposed optimal state that useful questions can be put forth to explain why (Parker and Smith 1990), which can guide empirical and theoretical investigations. Researchers should question their formulation of the optimal state inasmuch as it reflects their understanding of the organism or system of interest. This is also part of the model assessment problem because the model calculates objective values from system

parameters and the optimization results are dependent on this calculation. If the structure of the model is assessed, then the inference for the biological system is more valuable.

Multi-objective optimization is a first step in improving the approach of optimization in biology, and in anchoring the optimization model to empirical objectives. An explanation for why an organism is never optimal for a single trait is that there are multiple functions an organism must perform in order to survive and reproduce, and from engineering we know that optimal performance for multiple functions causes sub-optimal performances if a single function is considered (Niklas 1999). We can consider organisms to be adequate compromises among often competing requirements (Farnsworth and Niklas 1995) rather than inevitably optimal entities for any single function. This also explains the biodiversity present among organisms in similar ecosystems with similar constraints and requirements (Niklas and Kerchner 1984; Niklas 1997a,b; Niklas 1999). In general, the conclusion that organisms *should* optimize is not inevitable and optimality studies are best utilized to uncover the trade-offs that prevent organisms from achieving optimal states. Observed structures are not necessarily the result of optimal fitness achieved by natural selection (Gould and Lewontin 1979).

Schmitz et al. (1998) propose utilizing multiple criteria in optimality studies, with an example from optimal foraging (Rothley et al. 1997), to reconcile variability in optimal behavior. Any study that utilizes multiple objectives in a theoretical optimization cannot ignore the relevant criticisms of optimality theory in biology, in particular that a disconnect between the interpretation of the results of an optimality study and solid empirical evidence is easily formed, and must be bridged. The methodology described here seeks to assess a process model both for its form and for insight into the system in terms of its place in theoretical objectives. I treat objectives as proposed limitations and constraints on biological systems and always anchor the analysis with empirical evidence and observation.

The Pareto optimal frontier

The concepts of Pareto optimality and non-dominance are the centerpieces of the multi-objective optimization methodology. Rather than calculated as a scalar-valued function that is optimized, the multiple objectives are evaluated as a vector objective function (Reynolds and Ford 1999). If a single solution optimizes all objectives simultaneously, then that is the single optimal solution. More likely, if there are tradeoffs among performance in the objectives, then no single solution optimizes all simultaneously. To determine optimality I use the concept of non-dominance. When comparing two vector objective functions, they are considered to be non-dominant if any improvement in one objective is to the detriment of another (Cohon 1978, p 70; Figure 1.1). In this manner the non-dominated Pareto frontier (the set of all non-dominated solutions) is calculated. This frontier represents all tradeoffs among objectives in the optimal space; if the objectives were combined linearly with any set of weights and that scalar function optimized, the solution would fall on the Pareto optimal frontier.

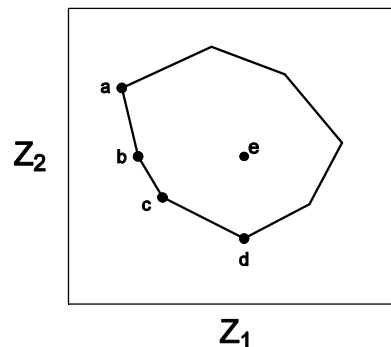


Figure 1.1. Non-dominance example for 2 objectives (Z_1 , Z_2), where the optimization problem is to minimize both objectives. The solid line encloses the feasible space for both objectives. Any solutions along the (a,b,c,d) line belong to the non-dominated Pareto frontier, where any improvement in Z_1 occurs at a cost to Z_2 , and vice versa. The value (e) is an example of a dominated solution in the feasible space.

In the context of optimality in biology, the Pareto optimal frontier visualizes possible compromises in performance for multiple objectives. The optimal frontier can be approximated through a simulation model and optimization algorithm (Reynolds and Ford 1999); when combined with empirical observation the corresponding growth forms characteristic of particular organisms can be evaluated for their placement in the Pareto frontier (e.g., Rothley et al. 1997).

Methodology

I use the guidelines for pattern-oriented modeling proposed by Grimm (1999) and Wiegand et al. (2003), and the process of model assessment presented by Reynolds and Ford (1999) as the basis for the multi-objective optimization methodology. Grimm (1999) suggests that model-building should begin with a simple model that reproduces an observed pattern of interest; the model is then improved by incorporating greater biological detail, and for each modification the model is assessed for whether it continues to replicate the observed pattern. This cycle of model building is repeated until an appropriate level of biological complexity is found that still adequately replicates the observed pattern (Grimm 1999; Wiegand et al. 2003). In this sense the model is designed for a particular system, which anchors interpretation of the results. For this method of model-building, a crucial problem is the procedure to be utilized for model assessment (am I still replicating the pattern?) and choice of the measurements by which the model will be judged (what parts of the pattern am I trying to replicate?). Validation or complete verification of an ecological model as a true representation of the system is never entirely possible (Oreskes et al. 1994), rather one can only confirm (albeit partially) through observation and assessment that the model is consistent with the observed pattern (Oreskes et al. 1994, Rykiel 1996). Rykiel (1996) observes that there is no consistent method or criteria through which the confirmation of a model with respect to its objectives is possible. To contend with these problems, Reynolds and Ford (1999) developed the Pareto Optimal Model Assessment Cycle (POMAC), with the purpose of applying multiple objective decision-making methods to the assessment of complex ecological models.

The POMAC procedure is conducted as a cycle, wherein model performance is measured against a vector of multiple objectives using Pareto optimality (Figure 1.1; Figure 1.2). The goal is to satisfy all objectives simultaneously with feasible parameter values and model structures. If the model cannot satisfy all objectives, then the potential of POMAC is to illuminate sources of model deficiency and guide model improvements in an iterative fashion. If a deficiency is found, the model is modified and reassessed.

POMAC has been used to assess several ecological models (Reynolds and Ford 1999; Turley 2001; Levy et al. 2004; Komuro 2005; Reynolds and Golinelli 2005; Komuro et al. 2006).

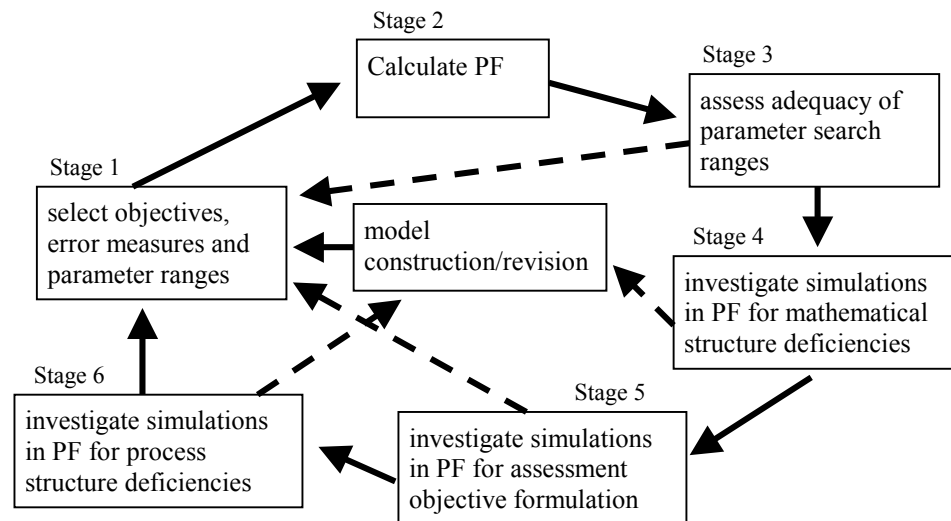


Figure 1.2. Stages of the Pareto Optimal Model Assessment Cycle (adapted from Reynolds and Ford 1999). After initial model construction and selection of objectives, generate the Pareto frontier (PF; the set of all non-dominated solutions) and the Pareto optimal set (the set of all Pareto optimal parameterizations). Through investigation of the resulting PF you can assess the adequacy of the parameter search ranges, then conduct simulations using the PF parameterizations to investigate the three sources of model deficiency: mathematical structure, objective formulation, and process structure. At each stage the PF is recalculated.

The theoretical insights that result from the POMAC methodology are bounded by the synthesis of the objectives, the model postulates and the empirical ecological system. Although this is true in any modeling exercise, it is rarely acknowledged and formally utilized in the process of using a model to synthesize ecological theories. The specification of the methodology represented by Figure 1.2 is incomplete to form theories utilizing the relationship between the objectives and the process model. This methodology fails to incorporate theoretical considerations into the optimization process, and does not realize the full potential of multi-objective optimization. The method of pattern-oriented modeling proposed by Grimm (1999) is best integrated with the POMAC assessment process in order to realize the potential of process-based models in ecological theory building, particularly for optimality models. This is best achieved when conclusions from model results are considered in the context of the objectives used in the assessment.

Example system

Functional structural models of plant development

A class of process models that would benefit highly from the multi-objective optimization methodology is functional structural models (FSMs) of plant development. The investigation of plant form has evolved from empirical observations to structured classification (e.g., Halle et al. 1978) to analytical models and computer simulation (Lindenmayer 1968; Honda 1971). Visual observations lead to qualitative descriptions of characteristic plant architecture, yet the explanation for the relationship between form and function is incomplete. In FSMs, biological processes are coupled with spatial representation of plant form (Godin and Sinoquet 2005). In general these models have two major components, the first of which is a set of rules and functions that control plant growth and development. The second is the translation of growth patterns to a visual/geometric representation of plant form (Kurth 1994) that preserves the lineage of plant parts and their topology. The details and depth represented by the growth functions can vary widely, as can the integration of the geometric representation with the growth processes.

There is no clear methodology for the development of these models (Godin and Sinoquet 2005) or for the assessment of model performance. Insofar as the structure of a plant model represents a working postulate for the relationship between plant form and function, the structure of the model must be assessed in order to inform us of the veracity of that proposition. Since plants are described by multiple characteristics and perform functions that are often competing, individual values that are used to assess plant models in an ad hoc manner fail to inform us completely of the proposed relationships between plant form and function. These models must be assessed thoroughly in order to improve their explanatory power (Ford 2000) and to make progress in the theory of plant form and function. Here, in this dissertation, I apply the multi-objective optimization methodology to a FSM of branch development in old-growth *Pseudotsuga menziesii* at the Wind River Canopy Crane Research Facility (WRCCRF).

For this system, I produce a process model of branch development that is built up from a simple model (Kennedy 2002; Kennedy et al. 2004) to one that incorporates more biological detail, guided by the results of a multi-objective assessment. I will form a theory of the multiple constraints a branch (or tree) is restricted by, and the underlying growth processes that produce the known morphological branching pattern in *P. menziesii* that is one of the compromises that satisfies the constraints. This theory will be framed by the context of the model structure, parameters and objectives. In Chapter II, I describe the existing model of branch growth developed for this system, and the first two-objective assessment. The background for the choice of new growth processes and empirical and biological objectives that are used to analyze the theory are presented in Chapter III. The results of the assessment cycle and subsequent model modifications are given in Chapter IV, and in Chapter V the results are synthesized to form a theory for branch development in the old-growth system. Final conclusions are drawn in Chapter VI.

Chapter II

Two-objective optimization for the assessment of a morphological model of branch development and reiteration in *Pseudotsuga menziesii*

Introduction

The model-building methodology in this dissertation begins with an observed pattern, and a simple model is designed that adequately yields the observed pattern. Schmitz (2001) asserts that models should be built using natural history, and simulation experiments can yield large sets of data that can be compared to field experiments. This is clearly in line with the goals of pattern-oriented modeling, and the use of natural history serves as a key starting point for the process of pattern-oriented modeling (Grimm 1999; Wiegand et al. 2003). The development and assessment of the model must incorporate objectives that are also informed in natural history and empirical observations. Further inference from the results of the model evaluation must consider the model and objective formulation.

In this Chapter, I review the analysis of plant functional structural models, and then describe the prominent biological pattern that motivates the modeling example, including the biological (natural history) background that guides the model and objective development. Finally, I present the simple model that was first produced by Kennedy et al. (2004) to replicate the observed pattern, and the results of a two-objective assessment I conduct for the model.

Plant modeling

Plant architecture is the consequence of many physiological processes including module production and associated bifurcation, hormone regulation, carbon allocation, and the influence of the microenvironment on the plant, and these combine to perform multiple functions of plant growth (e.g., movement of materials, foliage display, mechanical support; Pearcy et al. 2005). Visual observations lead to qualitative descriptions of characteristic plant architecture, yet the explanation for the relationship between form and function is incomplete. A promising method to explore this relationship is the development of functional structural models (FSMs), in which

biological processes are coupled with spatial representation of plant form (Godin and Sinoquet 2005). In many such models, however, morphology is often uncoupled from the underlying physiological processes and utilized for visualization of the results of the growth functions and developmental rules (e.g., Honda 1971; deReffye et al. 1997; Früh 1997; Fournier and Andrieu 1999; Kaitaniemi et al. 2000); that is, the loop of causal effects is broken. This break occurs either where morphology affects growth or where growth functions may affect morphology (morphology → growth functions → morphology). In modeling the relationship between plant form and function, an important goal is to understand how differences in architecture affect the comparative growth and function of trees (e.g., Ford et al. 1990; Sterck et al. 2005).

Models of plant development and architecture are produced for varying systems (Godin and Sinoquet 2005) and no clear criteria are used to assess whether the goals for the particular system are met. The assessment of plant architectural models has relied almost entirely on visual observations of the resulting plant forms (Honda 1971; Aono and Kunii 1984; Prusinkiewicz et al. 1990). In this case, a subjective question is asked: do simulated plants look like observed plants? The answer to this question may change depending on the viewer. In other cases, the validity of an optimization procedure is assessed when the emergent architectural parameters such as angle and length ratio match values observed in nature or in the history of evolution for the species. (Fisher and Honda 1977; Honda and Fisher 1978; Honda 1978; Fisher and Honda 1979; Niklas and Kerchner 1984; Farnsworth and Niklas 1995; Niklas 1997a,b; Niklas 1999). In other model assessments model outputs, such as number of terminal nodes, have been compared to empirical observations (Borchert and Honda 1984; Borchert and Slade 1984). It is obvious that there is no clear single methodology for the development of the model (Godin and Sinoquet 2005) or for the assessment of model performance. Insofar as the structure of a plant model represents a working postulate for the relationship between plant form and function, that structure must be assessed in order to inform the veracity of the postulate. Since plants are described by multiple characteristics and perform functions that are often competing, individual values that are used to assess plant

models in an ad hoc manner fail to inform completely the proposed relationships between plant form and function. These models must be assessed thoroughly in order to improve their explanatory power (Ford 2000, Chapter 12) and to make progress in the theory of plant form and function. In order to best evaluate the model of branch development in old-growth *P. menziesii*, the model must first be placed in the ecological context of the old-growth forest dynamics.

Biological background

Old-growth forest dynamics

It is well known that net primary production of forest stands decreases as the stand ages (Gower et al. 1996; Mencuccini and Grace 1996; Ryan et al. 1997; Bond 2000; Acker et al. 2002; Ryan et al. 2004), which has been observed as a summary of the carbon budget across an entire stand of trees. Others have reported that there is a maximum height and size for individual trees in a particular forest, and that the increment of height growth and crown expansion declines as trees approach the predicted maximum size (Moorby and Wareing 1963; Thomas 1996; Bond 2000; Ishii and Ford 2001; Koch et al. 2004). These two observations imply that growth in general is reduced for trees as they age and grow in size and complexity. Explanations for the reduction in growth as a tree ages include a shift to maintenance respiration that reduces the NPP of a forest, hydraulic limitation that reduces photosynthesis with reduced stomatal conductance, and nutrient limitation in the soils of older forests, which reduces photosynthesis (Ryan and Yoder 1997; Hubbard et al. 1999; Bond 2000; Hubbard et al. 2002; Ryan et al. 2004; Ryan et al. 2006). More recently it has been proposed that shifts in resource use efficiencies as trees differentiate into social classes during stand development (e.g., dominant trees) explains the decline in production with stand age (Binkley et al. 2002; Binkley 2004). These theories have in common a shifting carbon balance as trees age, whether from reduced GPP and/or reduced NPP.

Compensation in old trees

Although it is clear that forest trees are constrained with respect to size and age, the cause of the growth limitation is not clear and there is evidence that trees have evolved compensations for the constraints. In particular, direct tests of hydraulic limitation do not fully support the hypothesis (McDowell et al. 2002a; Barnard and Ryan 2003) and the authors offer hydraulic compensation as an alternative. Compensation may occur through changes in sapwood hydraulic conductivity and storage, or hydraulic lift by both horizontal and vertical roots (Phillips et al. 2003; Čermák et al. 2007; Warren et al. 2007). While conductance does decrease with path length, in *Eucalyptus saligna* the minimum water potential at which stomata closed (ψ_{min}) also decreased with path length, such that the stomatal conductance did not decrease between larger and smaller trees (Barnard and Ryan 2003). They propose that an increase in the sapwood-area to leaf-area ratio and a decrease in ψ_{min} compensated fully for path length and gravitational potential and maintained stomatal conductance in the taller trees. They suggest a shift in research to investigating the cost of such compensatory changes.

In the Wind River Canopy Crane Research Facility (WRCCRF; Shaw et al. 2004) the dominant *P. menziesii* trees have reached their maximum height and crown width for the site, and height growth and crown expansion are negligible (Ishii et al. 2000; Ishii et al. 2003). Yet, on the basis of estimated mortality rates, the trees are projected to persist for another 755 years in the forest stand (Franklin and DeBell 1988). This is possible, in part, due to physiological, architectural and morphological compensations in the old trees.

McDowell et al. (2002a,b) found some evidence in favor of hydraulic limitation in larger trees of *P. menziesii* at the WRCCRF and some in favor of hydraulic compensation. They measured and compared quantities related to water relations for trees 60 m, 32 m and 15 m tall. Their results for hydraulic conductance and carbon isotope discrimination conformed with hydraulic limitation, yet they observed several compensatory changes in large trees. This includes a more negative leaf water potential

in larger trees (ψ_{leaf}), resulting in a larger difference in water potential between the soil and leaves ($\Delta\psi$) and a decrease in leaf-area to sapwood area ratio (LA:SA). Others have found more negative foliar osmotic potentials in large trees (Woodruff et al. 2004). Some of the changes observed in taller trees likely increase the risk of xylem cavitation, and the changes do not compensate fully for the size of large trees compared to smaller trees, as demonstrated by the lower hydraulic conductance of taller trees (McDowell et al. 2002a,b). There is further evidence for compensation given that, in a related study, the hydraulic conductance of 60 m *P. menziesii* trees was actually higher than 32 m tree (Phillips et al. 2002).

In addition to those physiological and architectural compensations, *P. menziesii* at the WRCCRF likely compensate for the size constraint morphologically. Ishii and Ford (2001) present a theory of foliage regeneration through proleptic reiteration within the crowns of these old trees at the WRCCRF. The key to their theory of persistence of *P. menziesii* as a long-lived pioneer is morphological acclimation to the constraints imposed on trees of this age and size (Ishii and Ford 2002). The morphology compensates for the size constraint through regeneration of foliage on existing branch structures within the crown of the old-growth *P. menziesii*. Recently, such mechanisms have been detailed in several other species capable of massive size and longevity (Sillett and VanPelt 2007).

Ecological question

We assume reiteration has some role in maintaining foliage on the crown of trees that are constrained in height increment and crown expansion, yet we do not understand the long-term implications of this growth pattern on the species. In addition, although we know that growth of these trees is constrained, we do not know how the constraints relate to the dominant growth pattern, which leads to the following major ecological question that motivates this dissertation.

How does the pattern of branch development in old-growth *P. menziesii* at the WRCCRF compensate for size constraints on the species?

With the multi-objective optimization methodology, I synthesize postulates for the relationship between factors that control growth in *P. menziesii* and the potential constraints on the large, old trees. The process begins with analysis of the simple geometric simulation model of *P. menziesii* branch development presented by Kennedy et al. (2004). Next, I describe the pattern that characterizes old-growth *P. menziesii* branch development and that drives the model. Then, I give the major results of Kennedy et al. (2004).

Branch growth in P. menziesii

The basic architectural model of *P. menziesii* branch growth in old trees is characterized by a distinguishable first-order main axis with 2–3 daughter shoots per year (Figure 2.1; Ishii and Ford 2001; Kennedy 2002) and limited bifurcation on lateral axes. On old-growth branches this pattern is reiterated, through extensive proleptic growth of epicormic axes, to form spatially distinct shoot cluster units (SCUs; Figure 2.1b). The basis of the reiteration is the release of suppressed buds, which results in extensive growth of epicormic axes. For each successive incident of bud release a new generation of foliage is established (Figure 2.1c). If the new axis survives 1–2 years of reduced growth, it proliferates as if it were an order-1 axis (Kennedy 2002; Kennedy et al. 2004). The consequence of the proleptic growth is that foliage on old-growth *P. menziesii* branches is organized into clearly recognizable SCUs that consist of a distinguishable main axis and several lateral branchlets. Ishii and Ford (2001, 2002) proposed that the ubiquitous proleptic reiteration they observed compensates for the limited height and expansion growth for *P. menziesii* at the WRCCRF.

The model of Kennedy et al. (2004) is designed to replicate the basic observed growth pattern for *P. menziesii* with a simple process structure. In order to understand where the model might be improved in order to answer more detailed questions, we require a formal assessment of the branch model of Kennedy et al. (2004).

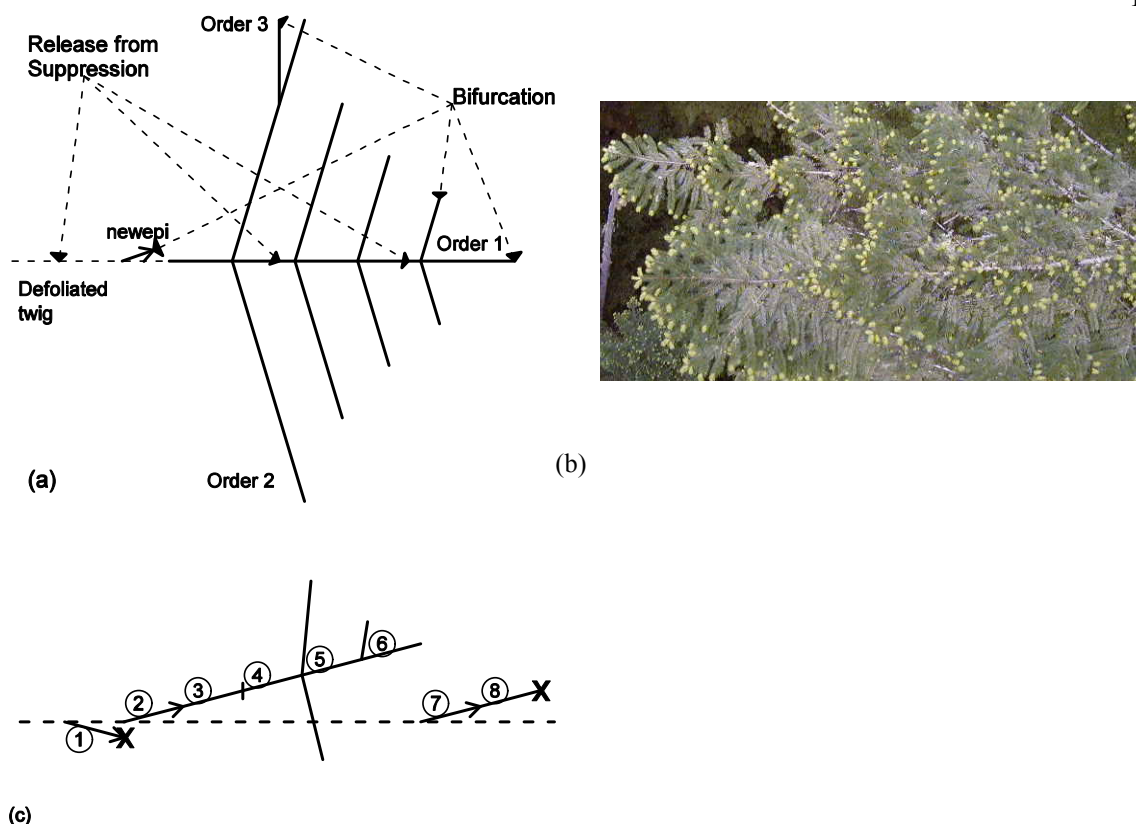


Figure 2.1: (a) Characteristic architectural model of branch growth in *P. menziesii* and two major growth processes in BRANCHPRO. Solid lines are foliated shoots of the shoot cluster unit that is pictured. The new epicormic shoot is assigned a higher generation than the parent SCU. Bifurcation occurs at all active terminal axes, and release from suppression is tested on suppressed buds along order-1 axes (b) Picture of branch growth and reiteration in *P. menziesii* at the WRCCRF (Taken June 2000), pictured from the top. (c) Release from suppression and SCU development along order-1 axes. The dashed line represents an order-1 axis that has lost its own foliage, but is still actively proliferating epicormic shoots and SCUs. Arrows represent new epicormic shoots, and “x” designates an axis that is no longer actively proliferating. Shoots 1, 2 and 7 are new epicormic shoots that resulted from three separate buds that were released from suppression along the parent axis. The number of daughter shoots for each is controlled by the *newepi* parameter. In this example, shoot 1 drew a zero from the Poisson distribution with *newepi* as its rate parameter and it is no longer actively proliferating. Shoots 2 and 7 each drew one daughter shoot. Shoots 3 and 8 are also controlled by the *newepi* rate parameter, and shoot 3 drew a value of one. Shoot 8 drew a value of zero, and that axis is no longer actively proliferating. Shoot 4, as the third year of growth of the new epicormic axis, is now considered an order-1 shoot. Its bifurcation is controlled by a draw from the Poisson distribution with order 1 as its rate parameter, and it drew a value of 3. Shoot 5, also order 1, drew a value of 2. Shoot 6 is the current year’s terminal order-1 shoot. In the simulation this shoot will draw a number from the Poisson distribution with the order-1 rate parameter, and that value will be the number of daughter shoots from shoot 6 in the next year. The axis that proliferated from shoot 2 is becoming its own independent SCU.

For the purposes of this dissertation, the first model of Kennedy et al. (2004) will be called BRANCHPRO (for proleptic reiteration). Next I describe the structure of

BRANCHPRO and I present a two-objective empirical optimization to assess how well the model fits the general growth pattern.

Summary of Kennedy et al. (2004), simulation of branch growth in *P. menziesii*

Regular sequential growth

In BRANCHPRO, shoot proliferation is regulated solely by the architectural status of the shoot with distinct parameters assigned for order-1, order-2 and order-3 bifurcation rates; newly forming epicormic complexes (*newepi*) are given a unique rate parameter for their first two years of growth. This choice of parameters was driven by the architecturally distinct branching orders and the observation that newly forming epicormic complexes take at least two years to begin to reiterate fully the characteristic architectural model (Figure 2.1; Kennedy et al. 2004). The number of daughter shoots for each active terminal axis is drawn from a Poisson distribution whose rate value is the corresponding parameter. The actual number of daughter shoots is limited to three for each terminal node, so the realized expected number of daughter shoots is less than the rate parameter. The relationship between the rate parameter and expected number is monotonically increasing (Figure 2.2).

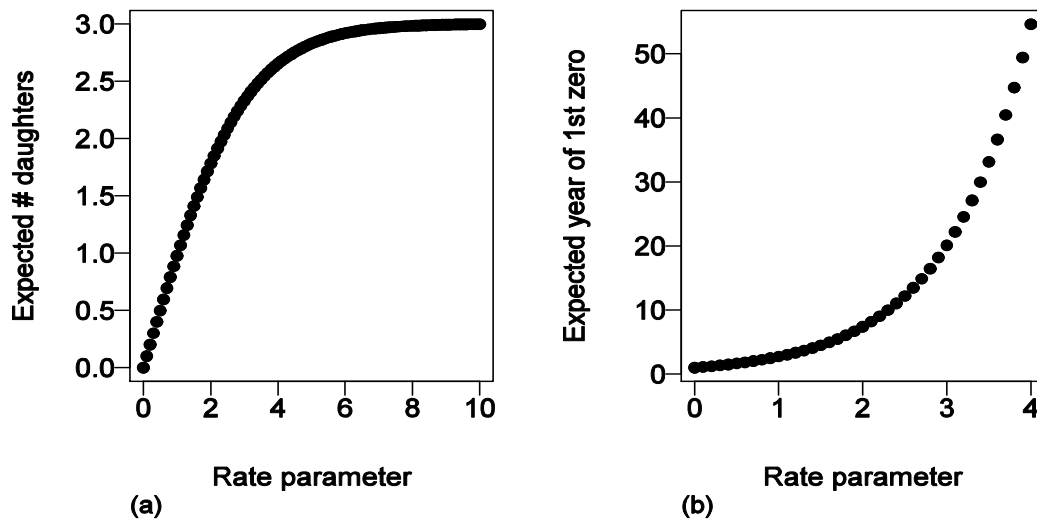


Figure 2.2 (a) Expected number of daughter shoots in BRANCHPRO increases, then levels off at 3 as the bifurcation rate parameter increases. (b) The expected year of the first zero draw increases exponentially with the rate parameter.

Proleptic growth and reiteration

In the BRANCHPRO simulation, epicormic buds are initiated through two stages. First, for each order 1 shoot a single number is drawn from a gamma probability distribution. The parameters of the distribution were determined empirically so that the shape of the curve resembles the observed distribution of timing of epicormic initiation (Kennedy et al. 2004). When the parent shoot of the bud reaches that age, the bud is tested for epicormic initiation according to the default rule: initiation of an epicormic shoot occurs only if either of the axes immediately lateral to the current shoot is no longer proliferating at the designated year. If the bud does not initiate, then it is terminated. Table 2.1 gives some of the basic processes and functions in BRANCHPRO.

Table 2.1. Description of processes for branch development in BRANCHPRO

Process	Function Type	Explanatory variables	Range of values or units
bifurcation	Poisson RV	Order, new epicormic	(0,3)
branching angle	Empirical regression	Parent length	$(-\pi/2, \pi/2)$
length ratio	Empirical regression	Parent order	(0,1)
foliage weight	Empirical regression	Shoot age	grams
foliage area	Empirical regression	Shoot age, distance from stem	cm ²
timing of epicormic initiation	Gamma RV, rules	Immediate lateral axes	year

Kennedy et al. (2004) model analysis

In the first analysis of BRANCHPRO by Kennedy (2002) and Kennedy et al. (2004), more than 80% of simulated branches failed to maintain foliage through the first 90 years (Kennedy 2002). To guarantee minimum growth for the branch and allow sufficient time for the branch to establish its reiterative framework on the first-generation main axis, each simulated branch is deterministically assigned three daughter shoots every year for the first fifty branch years (Kennedy et al. 2004). After fifty years, the first-generation order-1 axis is subject to random draws from the truncated Poisson distribution. Furthermore, long-term simulations showed that without reduction of the bifurcation of new epicormic shoots with increasing generation, the branches grow impossibly large (Kennedy et al. 2004). The value for the *newepi* parameter is thereby decreased with increasing generation for all generations greater than 3. The final

investigation of the model by Kennedy et al. (2004) was performed through simulation experiments and a local sensitivity analysis of the four parameters.

Major results of Kennedy et al. (2004)

Small changes in the parameters that control the capacity for reiteration in the branches resulted in dramatic increases in the numbers of shoots and SCUs on the branch, which indicated that it was the cumulative effect of reiterating structures that caused drastic variability in the model results (Kennedy et al. 2004). Simulation also made clear that reiteration has a profound effect on foliage maintenance in *P. menziesii* and, in the absence of reiteration, the branches terminate quickly. The evidence gathered both empirically (Ishii and Ford 2001; Ishii and Ford 2002; Ishii et al. 2002; Ishii et al. 2003) and through the modeling study (Kennedy et al. 2004) indicate that this observed growth pattern is one of the characteristics that enables *P. menziesii* to persist at the WRCCRF, despite its constrained height growth and crown increment.

Kennedy et al. (2004) suggested that the distinct clusters of foliage are caused by the release of the suppressed bud from some kind of dominance and the subsequent assertion of dominance by the newly forming axis. For this dissertation, I use the definitions of apical dominance and apical control presented by Brown et al. (1967). Apical dominance is the suppression of the outgrowth of lateral buds for the current year. Apical control is the continued suppression of growth of subordinate axes after the current year's bud outgrowth. Conifers are said to have weak apical dominance, but strong apical control because lateral buds do grow in the current year, but have limited subsequent growth year-to-year (Brown et al. 1967). In this dissertation, I assume that a bud has escaped the apical control of the parent axis when it is suppressed more than one year, and then is able to grow as if it is a dominant axis.

The multi-objective optimization problem is formally defined in Appendix A. Next, I present a preliminary two-objective assessment of BRANCHPRO to guide where

more biological detail is required in the growth processes and to begin the assessment and optimization cycle.

Two-objective binary assessment

For the assessment, we utilize a vector objective function and Pareto optimality with a binary error structure. With binary errors, a range of target values for each objective is specified; the model results are assessed for whether they fall within the target range. If the objective is within the target range, it is given an assessment value of 1, and 0 otherwise (Reynolds and Ford, 1999).

A set of vector objective functions is evaluated through the concept of Pareto optimality and non-dominance. For the two-objective case, there are three possible results for each vector objective function: (i) either both objectives failed $\{0,0\}$, (ii) one of the objectives is satisfied but not the other $\{0,1\}$, $\{1,0\}$, (iii) or both objectives are satisfied simultaneously $\{1,1\}$. Case (iii) is the best possible outcome, and it is considered to dominate solutions in cases (i) and (ii). All solutions that satisfy both objectives are considered mutually co-dominant. If no solution satisfies both objectives simultaneously, but fall under case (ii), then these are considered to be mutually co-dominant and to dominate solutions in case (i).

The modeling analysis conducted by Kennedy et al. (2004) is restricted because they chose values for the model parameters that were biologically reasonable and yielded desired model outputs (Kennedy 2002; Kennedy et al. 2004). At face value, this is a reasonable approach. I show in this Chapter, however, that in order to satisfy the empirical observations, a more formal and quantitative determination of the parameter values is desirable. This assessment reveals a model deficiency that they previously overlooked. I present a binary error assessment of BRANCHPRO with two objectives in order to observe how well the modeled processes match the observed pattern of growth and to identify where deficiencies may be.

The SCU, which results from the process of proleptic reiteration, can be considered a fundamental trait of old-growth *P. menziesii* branches. I define the two characteristics essential to define the demography of the *P. menziesii* branches to be the number of foliated shoots and SCUs on the branch. Ishii et al. (2003) present demographic data for old-growth *P. menziesii* branches. They harvested 9 branches from 3 different old-growth *P. menziesii* trees at the WRCCRF (trees were ~400 years old), one each from the lower, middle and upper crown. The lower crown branches were all epicormic in origin, and similar in age to the upper crown branches (near 90 years). The three middle crown branches were all around 145 years in age. Data for each branch included a count of the number of shoots and SCUs on each. To set the binary error intervals for the optimization (Table 2.2), we used the values for the 90-year branches. The lower edge of the binary error interval was defined by the minimum number of foliated shoots (or SCUs) observed, and the upper range was defined by the maximum observed values. A single branch was simulated to calculate each objective. The parameter values allowed to vary freely in the optimization were the bifurcation rates of order 1, order 2 and order-3 shoots, as well as the bifurcation of new epicormic shoots (*newepi*). There are then 2 objectives and 4 parameters in the optimization search (Table 2.2). In the context of the multi-objective optimization problem (Appendix A), the model

$$M(X), \quad (2.1)$$

is the BRANCHPRO model. The vector of decision variables is

$$X = \{x_1, x_2, x_3, x_4\}, \quad (2.2)$$

where each element is: x_1 =order 1, x_2 =order 2, x_3 =order 3, x_4 =*newepi*. The vector objective function is

$$F(M(X)) = (F_1(M(X)), F_2(M(X))). \quad (2.3)$$

The values for each element of the function are F_1 =foliated shoots and F_2 =SCUs, both calculated at year 90. All optimization results are from the 500th generation. See Reynolds and Ford (1999) for a description of the search algorithm.

General strategy for optimization searches

Following the guidelines of Reynolds and Ford (1999; Figure 1.2), the first stage of the Pareto optimization is to investigate the allowable range of parameter values for each optimization search. If a parameter value converges at the edge of its allowable range, the range should be increased and the optimization repeated. This is necessary in order to explore the full possible distribution of parameter values in the Pareto optimal space and in order to uncover any source of model deficiency. With satisfactory parameter ranges, the model structure and objectives are then assessed through simulation of Pareto optimal results.

Two-objective assessment: allowable parameter ranges

The range of allowable parameter values dictates the bounds on the parameters in the optimization; if the resulting parameter distributions converge at either end of the allowable ranges then the ranges should be expanded and the search repeated before any further analysis is conducted. To guide the initial search ranges in the optimization, I assumed the basic architectural model of *P. menziesii* branch growth; the range decreases between order 1, order 2, order 3 and *newepi* bifurcation (search 2.1; Table 2.2).

Table 2.2. Summary of the two-objective optimization searches. Order refers to branch order from primary axis.

Allowable range				
Order 1	(0,3.5)	(0,3.5)	(0,3.5)	(0,4)
Order 2	(0,3.0)	(0,3.5)	(0,3.5)	(0,4)
Order 3	(0,2.5)	(0,3.5)	(0,3.5)	(0,4)
New (<i>newepi</i>) Epicormic	(0,1.0)	(0,3.5)	(0,3.5)	(0,4)
Binary Error Intervals				
Foliated Shoots	(2500,6500)	(2500,6500)	(2500,6500)	(2500,6500)
SCUs	(20,50)	(20,50)	(20,50)	(20,50)
Search Results				
Number in set	4327	2175	632	838
Percentage architectural model	59%	2.80%	69.96%	69.70%

Search 2.1

All solutions in the Pareto frontier simultaneously satisfy both shoots and SCUs, but it is clear that, in the distributions of parameter values in the resulting Pareto frontier, the parameter ranges are inadequate. Figure 2.3a shows histograms of parameter values in the Search 2.1 Pareto optimal set. The distribution of order 2 in particular is shown to converge at the upper edge of its range, with a clear mode at a bifurcation of 3.0 (Figure 2.3a). This indicates that, when achieving the two objectives simultaneously, the model has higher values of order 2 than expected. To understand why the parameter distribution is converging at such high values, it is necessary to search an expanded allowable range and observe the pattern of convergence. Since all three parameters measure bifurcation rate, and the search range for order 2 is clearly inadequate, I apply the same search range for all parameters. Given that bifurcation is limited to three daughter shoots per active node, I increase all search ranges to 3.5 and repeat the optimization (search 2.2; Table 2.2).

Search 2.2

Figure 2.3b shows the parameter distributions in the search 2.2 non-dominated Pareto optimal set. The distribution of order-1 bifurcations shifted dramatically to the left relative to the distribution of order-1 bifurcations in search 2.1 (Figure 2.3a,b); the distribution of order-2 bifurcations continued to converge at the upper edge of its search range. Order-3 bifurcation remains centered about one, whereas *newepi* bifurcation shifted to the right in an almost mirror image to order 1 (Figure 2.3b). A detailed analysis of the model structure in the context of these parameter patterns demonstrates that the shift in distribution was caused by the restrictions of the model and parameter structure in the way in which the two objectives are met simultaneously, and in the ability of the parameters to accommodate the two objectives (see below). The combination of shoots and SCUs on a branch depends on the values of the bifurcation parameters and the rules for epicormic initiation.

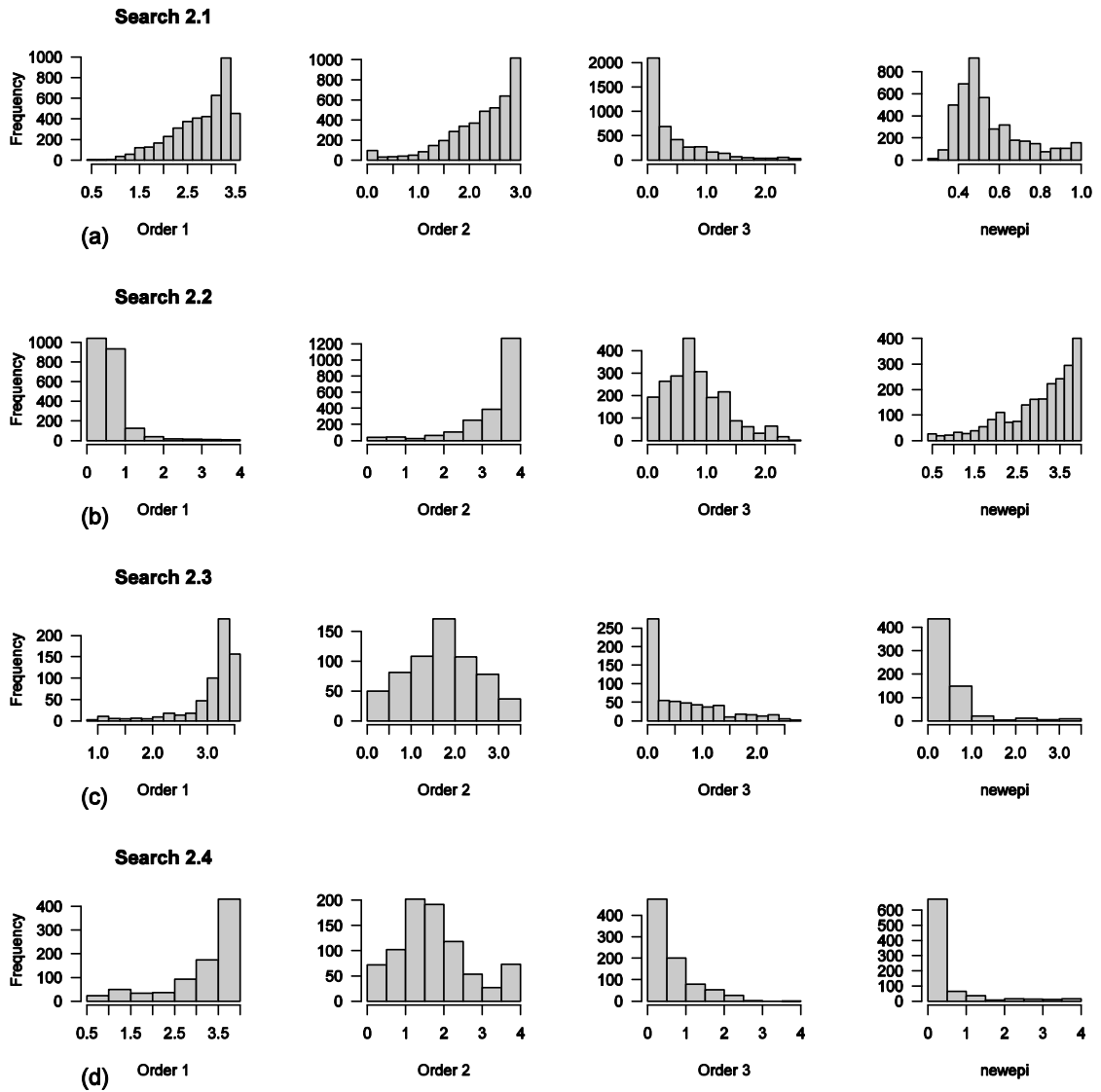


Figure 2.3. Distribution of parameter results in the Pareto set for each two-objective optimization. (a) Search 2.1 with restricted search ranges shows that both order 1 and order-2 bifurcation converged at the high end of their respective search ranges. Order 3 and *newepi* converged at lower values; (b) Search 2.2 with expanded search ranges shows that the mode of order 1 shifted dramatically to the left relative to search 2.1, whereas the mode of *newepi* shifted dramatically to the right; (c) Search 2.3, with completely random main axis, shows the modes of the parameter values to be closer to those proposed by Kennedy et al. (2004) than the modes of values in search 2.1 and search 2.2; (d) Search 2.4 with sequential reiteration also shows the modes of the parameter values to be near those proposed by Kennedy et al. (2004).

Parameter values and model structure

Two consecutive processes control the number of SCUs on a branch. First is the release of suppressed epicormic buds along order-1 main axes, which is controlled by a simple empirical probability distribution that is not assessed here. The second process is the development and growth of a newly initiated axis, as defined by the bifurcation

parameter. When a bud successfully initiates, the first two years of bifurcation are defined by the *newepi* parameter. If the new axis successfully proliferates the first two years, then further growth is defined by the order-1 rate parameter (Figure 2.1c). This sequence of events defines the interactions among order-1, order-2 and *newepi* rate parameters, and the deterministic main axis, and it explains the parameter distribution patterns between search 1 and search 2.

In the parameter ranges for search 1, *newepi* was limited to a maximum value of 1. The expected survival of an axis with a bifurcation rate of 1 is 2.71 (Figure 2.2b), so a new epicormic axis is expected to draw a zero for the number of daughter shoots before its third year; the axis is therefore expected to terminate before its third year and not successfully form an independent SCU. For those new epicormic axes that do survive past the first two years, the values of the order-1 and order-2 bifurcation determine whether there is sufficient shoot proliferation for the formation of an independent SCU. The order-1 axis also provides a structure for additional dormant buds to be initiated, thus ensuring that more SCUs develop. Kennedy et al. (2004) observed, however, that some form of feedback is necessary on the process of reiteration, as the branch can quickly accumulate impossibly high numbers of shoots. In BRANCHPRO, higher values of order-2 bifurcation act as a feedback mechanism, reducing the chances a dormant bud is released by increasing the chance that higher ordered axes are still proliferating. Therefore, the optimal parameter relationship to satisfy the number of foliated shoots and SCUs, when *newepi* is limited to a maximum value of one, is to have high both order-1 and order-2 bifurcation.

In contrast, when *newepi* is allowed to range to 3.5, high-generation order-1 axes are relieved of the burden to provide structure for dormant buds. With such a high value of *newepi*, the majority of buds that initiate successfully are expected to survive the first 2 years (for *newepi*=3, $P(Z>2)=.91$, $E(Z)=20$ years, where Z =number of years before the first zero draw, i.e., termination of the axis; Figure 2.2b), therefore fewer suppressed buds are necessary to initiate in order to accumulate additional SCUs. The deterministic, first-

generation order-1 axis provides sufficient structure for the suppressed buds, therefore higher generation order-1 axes are not necessary for suppressed bud release. High order-2 bifurcation ensures sufficient shoot proliferation for the accumulation of independent SCUs when order-1 bifurcation is low.

The role of order 2 in controlling this bud bank is illuminated by the pattern of branch development that results from the default parameterization. When Kennedy et al. (2004) first analyzed BRANCHPRO, the value of foliated shoots was used to verify the default parameter values whereas SCUs were measured as trends in the sensitivity analysis. In 20 simulated branches with default parameter values, only 1 satisfied both shoots and SCUs simultaneously. The remaining branches had the correct shoots, but too many SCUs. When the order-2 parameter is increased the SCUs tend to decrease. These trends all depend on the rule that deterministically assigns three daughter shoots each year to the first generation main axis, as discussed above.

Parameter dependencies and branch examples

The pair-wise relationships between parameter values gives further insight to the theory of growth represented by the combination of model and assessment objectives. There are negative trends between all parameter pairs with the exception of positive trends between new epicormic and orders 2 and 3 (Figure 2.4). These trends are clearly related to the constraints given by the two objectives. If all bifurcations were high, then the branch would grow too large. Therefore, with a large bifurcation for one architectural order, there must be a smaller bifurcation in other orders. In contrast, if new epicormic bifurcation is high, then, to control the proliferation of new SCUs, order-2 bifurcation must also be high. This explains the positive relationship between the two parameters in the Pareto optimal set. The structure of the model in the formation and development of SCUs dictates these relationships relative to the objectives, and the resulting parameter vectors that emerge conflict with observed patterns of growth in *P. menziesii*.

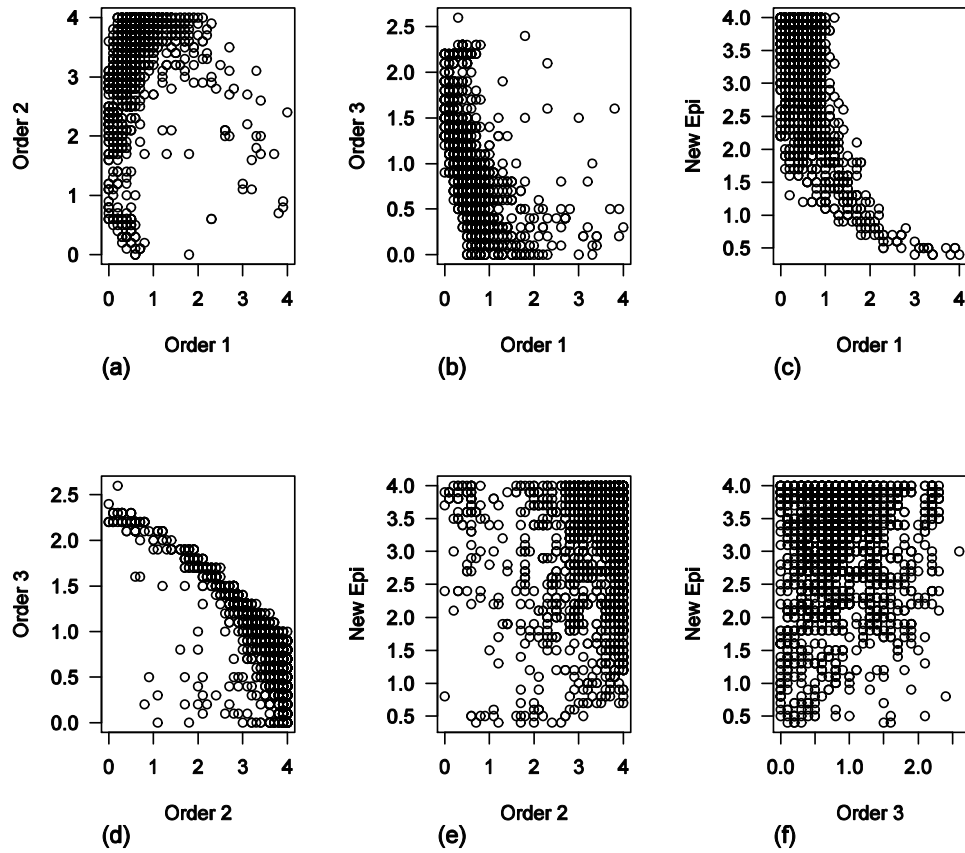


Figure 2.4. Pair-wise scatter plots of parameter values in the search 2.2 non-dominated Pareto frontier. There are negative relationships in plots (a-d), and positive relationships in plots (e-f). This pattern of parameter relationships persists through all four searches.

I simulated two branches, one using the default parameter values and one from a parameter vector characteristic of the non-dominated Pareto frontier from the search 2.2 (Table 2.3; Figure 2.5a,b). The search 2.2 branch has epicormic development primarily on the first-generation main axis and sequential growth primarily on the order-2 axes throughout the branch (Figure 2.5b). This is evidenced by the maximum order of 8 and a maximum generation of 4 on this branch. In contrast, for a branch simulated with the default values, the highest generation was 6 with the highest order 5 (Figure 2.5a). In order to see the average performance for each of these parameterizations for the number of shoots and SCUs, I simulated an additional 20 branches for each. Branches simulated from both parameterizations are successful for the number of shoots; branches simulated from the default parameterization, however, produce too many SCUs. In addition, in search 2.2, only 2.8% of the solutions result in the characteristic architectural model of *P*.

menziesii (Table 2.2). This violates the basic pattern upon which the model of reiteration is based. From this analysis we see that branches resulting from the default bifurcation values fail to meet both objectives simultaneously, and branches simulated from parameter values that do meet both objectives yield an anomalous architectural model. This seems due to the rule that maintains the main axis deterministically. I thereby remove the deterministic main axis rule from BRANCHPRO and repeat the optimization search (search 2.3).

Table 2.3: Parameter values for the default parameterization and example branch from the search 2.2 non-dominated Pareto frontier.

Parameter	Default	Example Branch
Order 1 (rba)	2.5	0.5
Order 2 (rbb)	1.5	3.5
Order 3 (rbc)	0.5	1
New (rbepia) Epicormic	0.7	3.4
MaxGen	7	4
MaxOrd	5	8

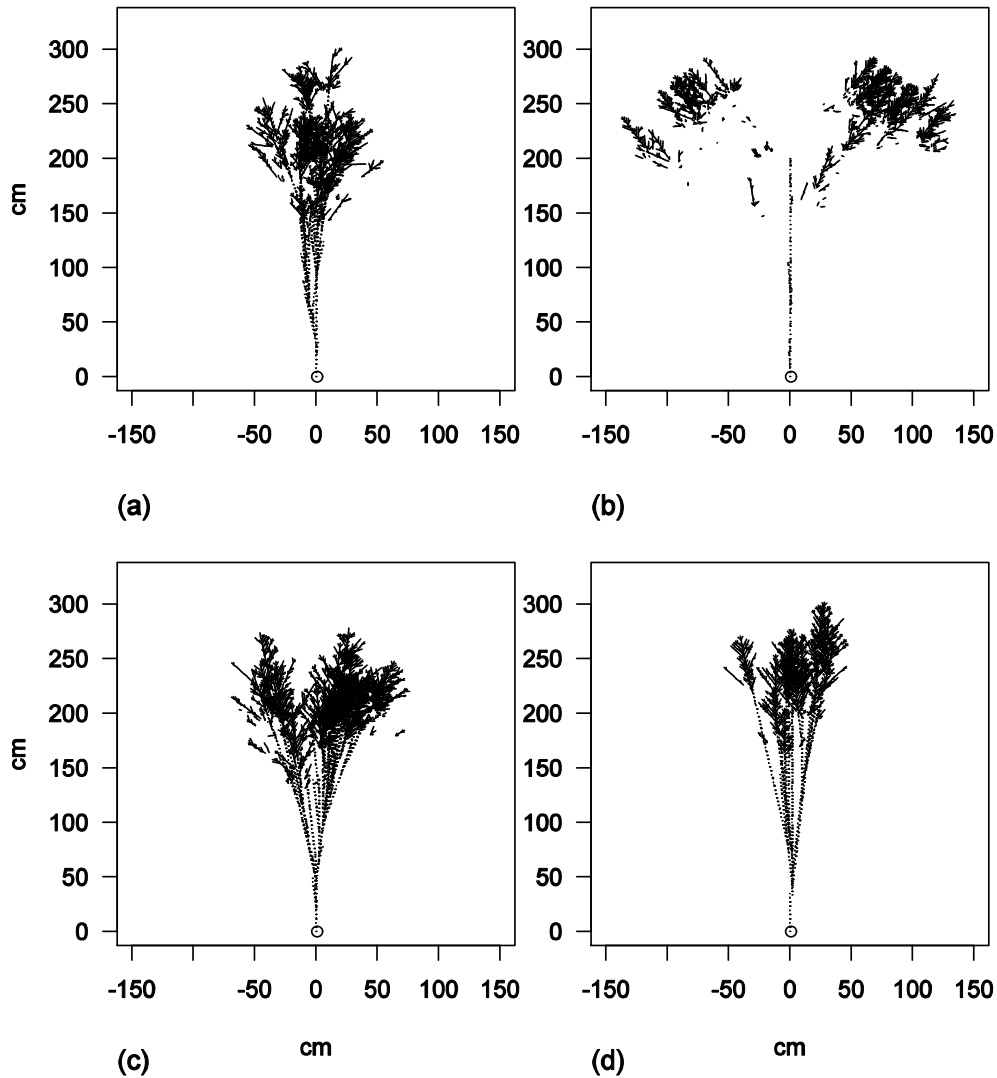


Figure 2.5: Branch maps generated from the original (Kennedy et al. 2004) branch model. The branch is oriented at $x=0$ and $y=0$ and length extends along the y -axis. Dark lines represent foliated shoots and dashed lines are all order 1 non-foliated shoots. (a) Default parameter values for BRANCHPRO yield a branch where most foliage growth occurs along the original main branch axis with some regeneration proleptically along higher generation order 1 axes. This visually resembles the pattern shown in Figure 2.2b, although it does not satisfy the SCU objective; (b) parameter values from the search 2.2 Pareto set with (order 1=0.50; order 2=3.50; order 3=1.00; $newepi=3.40$) with limited growth along the original main axis and extensive growth along higher ordered axes. Such a growth pattern is not observed in the species and is illustrative of the deficiency in the model revealed by the two-objective assessment. (c) branch with sylleptic reiteration, default values. (d) branch with sylleptic reiteration (order 1=4.00, order 2=1.30, order 3=0.10, $newepi=0.30$)

Search 2.3

The results of search 2.3 support the conclusion that the main axis rule is responsible for the relationships observed in the parameter distributions in the previous search (Figure 2.3c). When the main axis rule is removed from the simulation, 69.6% of

the solutions in the new non-dominated set satisfy the characteristic architectural model (Table 2.2), and the mode of each parameter is near the default, empirically derived values. A branch simulated from a representative parameter combination in this search is similar to the default branch (not shown), although several runs had to be discarded because many branches terminated before the full 90 years. It is not unexpected for some branches to die before 90 years, yet in this model analysis we are interested in what is necessary for the long-term survival of branches. The main axis rule provided a convenient means to allow sufficient foliage maintenance for branches to survive to Yr90, but it is clear that the rule is not an appropriate representation of the process of growth in these branches.

Kennedy et al. (2004) recognized the deterministic main axis as a potential place in which BRANCHPRO could be improved. They suggested the incorporation of traumatic sequential reiteration as a mechanism to maintain the first-generation order-1 axis. This process approximates the loss of apical control on sequential lateral axes when the order-1 terminal bud fails to produce new shoots; a sequential lateral shoot assumes the role of the main axis (i.e., assigned order 1) within the usual timing of growth. This has been observed in the vertical growth of branches in various species (e.g., Harding 1986), and along higher-ordered axes of branches when the order-1 axis breaks or dies off (Ishii et al. 2002). In the modification of BRANCHPRO, traumatic sequential reiteration is only applied to the first-generation foliage cluster. BRANCHPRO with sequential reiteration is assessed in the fourth optimization search (search 2.4)

Search 2.4

The distributions of parameter values for a two-objective optimization on BRANCHPRO with traumatic sequential reiteration (search 2.4) are closer to the expected values for bifurcation, with some convergence of order 2 again at the high end of the search range (Figure 2.3d), but order 1 also converges at high values. Both order 3 and new-epicormic bifurcation converge near their default values. In this Pareto frontier, 69.7% of the solutions follow the characteristic architectural model for *P. menziesii*

(Table 2.2). The pattern of pair-wise relationships among parameter values that was evident in search 2.2 (Figure 2.4) persists consistently through all four optimizations.

Based on a simple visual assessment, the overall pattern for a 90 year-old simulated branch (clear dominant order-1 axes, foliage regeneration proximally along major axes) is similar between the default branch and a branch with sequential reiteration (Figure 2.5). Differences do occur, however, in a population of branches simulated for each case. For example, for 50 branches simulated with a random main axis 15 did not survive to year 90 (30%), whereas 100% of branches simulated with sequential reiteration did survive to year 90 ($X=\{4.0, 1.3, 0.1, 0.3\}$).

Discussion

Plant model assessment requires multiple objectives

To assess their model, Kennedy et al. (2004) performed a sensitivity analysis and simulation study. As is common in plant modeling, the modifications they made to their model are mostly ad hoc, and they do not define any formal criteria by which to judge model performance. In this Chapter, I propose two objectives that should be satisfied simultaneously in order for the model to be an adequate representation of branch development in *P. menziesii*, and conduct a formal assessment of the model. I show that by increasing the demands on model structure with two objectives to be satisfied simultaneously, the biological representation of branch development in BRANCHPRO is inadequate. This inadequacy is unveiled in a way that would have been difficult in the kind of analysis originally undertaken. The multi-objective optimization is a necessary step for the first goal of the modeling exercise, i.e., find an adequate representation of the biological theory. Although the deficiencies uncovered in this two-objective assessment do not change the conclusions of Kennedy et al. (2004) with respect to the goal of evaluating the effect of proleptic reiteration on long-term branch development and longevity (sensitivity analysis repeated but results not shown), it does modify the theoretical basis for those conclusions.

The assumption of Kennedy et al. (2004) that I find inadequate is the deterministic maintenance of the first-generation main axis. Biologically, the deterministic main axis can be interpreted as a strongly dominant axis, which proliferates new shoots at its terminal end regardless of external or internal factors. Alternatively, in the context of the set of assessment objectives, parameter space, and model structure, I show that foliage maintenance is accomplished both through adaptive proleptic reiteration and traumatic sequential reiteration. Each of these represents the loss of apical control, where for proleptic reiteration the control is on suppressed buds and for traumatic sequential reiteration the control is on order-2 shoots. The establishment of, and then release from, apical control seems to be the major mechanism by which foliage is maintained in old-growth *P. menziesii*.

Two objectives are insufficient

One of the goals of the multi-objective optimization procedure is to quantify a set of objectives that, if satisfied simultaneously, define an adequate model structure. The assessment presented here is not yet complete, given the observation that inadequate branch representations still satisfy the two objectives simultaneously (search 2.2; Figure 3b). The two objectives are thereby insufficient to describe the pattern, and it is clear that the structure of the branch is defined by more than the numbers of shoots and SCUs (i.e., organization of foliage). Additional objectives should be quantified to describe the growth pattern, such that only an adequate model can satisfy all objectives.

Honda (1971) describes two problems in the study of tree forms: pattern recognition and morphogenesis. He acknowledges that the pictures generated by his geometric model “do not bear a particularly close resemblance to actual trees”, yet the representation of morphogenesis by the model is valuable to examine the effects of different parameters on tree form in a holistic sense. We have developed a model of the process of morphogenesis, yet with our two objectives we clearly have an issue with the problem of pattern recognition (which Honda never quantifies). Other basic descriptions of the branch pattern that can be used as assessment objectives include the length of the

foliated section of the branch and the emergent basic architectural model (dominant order-1 axis). For optimization, there are several functions of branch form that may act as constraints on growth as a branch ages and becomes more complex. These include hydraulic limitation, mechanical stability and foliage display. Optimization of any of these functions may come at the expense of another function, and the plant itself contends with them simultaneously. Quantitative functions for each of these can be formulated and incorporated into the vector objective function.

Progress to be made from Kennedy et al. (2004)

The underlying process structure of BRANCHPRO is valuable because it enabled us to observe the consequences of the branching pattern for the growth and development of the old-growth branches. It is limited, however, to the scope of questions it can address. In BRANCHPRO, most growth relationships are determined from empirical statistical functions, with simple rules to govern the processes of reiteration and bifurcation (Table 2.1). Distinguishing bifurcation rate by shoot order enforces a form of architectural hierarchy that serves as a surrogate for hormonal interactions. The alternative rules set for suppressed bud release represented postulates for the interaction of the suppressed bud with its neighboring axes (Kennedy 2002; Kennedy et al. 2004), with the timing governed by a simple draw from a gamma probability distribution. The incorporation of reiteration in a branch growth model is an advancement in the modeling of plant architecture, and has facilitated formulation of a theory of how processes of growth and reiteration integrate to ensure long-term survival of branches. The theory is incomplete, however, because we have only shown that it does help long-term survival. We need to improve the model and use the assessment process to help us articulate *how* long-term survival is accomplished in this branching system through the choice of growth processes and objectives that represent postulated constraints on growth in this species.

To progress in explaining how the process of proleptic reiteration may occur biologically (i.e., increase the explanatory power of the model) and the physiological consequences, it is necessary to incorporate additional detail in the model. The detail is

necessary to distinguish alternative postulates for the factors that influence branch proliferation and dormant bud release from suppression. From the results presented here, and in order to describe potential physiological consequences of proleptic reiteration in the context of old-growth trees, additional assessment and optimization objectives are required. The two objectives utilized for this assessment are inadequate representations of the pattern of growth, and additional objectives should be used to assess this model of branch development.

Summary of progress in the methodology

Thus far I have followed the guidelines of Grimm (1999); a simple model was produced that answers a simple modeling question and reproduces a simple pattern. A two-objective assessment illuminated deficiencies in the model structure, and further assessment with model modifications showed a more adequate fit. The questions the model was designed to investigate are on the same level as the process specifications of the model, and were answered adequately. For the next stage of investigation the demands on the model will increase with respect to the questions it is to investigate. This requires I add processes that represent the proposed effects at the simplest biological representation, and add objectives that the system is proposed to be optimal for or constrained by (theoretical objectives). The formulation of the growth processes and objectives may overlap, as we think the phenomena that the branch is constrained by may influence the growth processes. We integrate them in the model, then optimize them separately (but simultaneously) in the objectives and investigate, through simulation, the patterns that emerge in the post-processing of the Pareto frontier.

Chapter III

Model growth functions require natural history background and biological justification

Introduction

In the development of an ecological process-based model, one must consider many issues. The first is that an ecological process-based model has underlying it both process and mathematical structures; these represent component postulates for the function of the ecological system. A key component of the assessment and analysis of these process-based models is the clear statement of how the underlying process and mathematical structures represent these ecological postulates. The postulates are distinct from hypotheses in the statistical sense, in that they are not testable at some level of significance. Rather they are working postulates for how the system functions.

Furthermore, when conducting a multi-objective model assessment, one must also specify the set of assessment objectives against which model performance is measured (e.g., the number of shoots and SCUs, Chapter II). During model development one goal is to decide on an appropriate level of biological complexity; the dependence between the assessment objectives and the model functions will guide the decision for an appropriate level, relative to the goals of the modeling exercise. BRANCHPRO was initially developed with the goal to ascertain the long-term consequences of proleptic reiteration on branch development in *P. menziesii* (Chapter II). Through methods of simulation and sensitivity analysis it was concluded that reiteration has a significant impact on the potential growth and longevity of the branching system (Kennedy et al. 2004). A two-objective assessment of BRANCHPRO revealed that the model is deficient with respect to the process of reiteration and morphogenesis (Chapter II). In particular, the processes of epicormic initiation and bifurcation were inadequately specified.

The lack of biological complexity in process specifications in BRANCHPRO limits the further conclusions that can be drawn from its analysis. Given that reiteration has a significant impact on long-term branch growth, longevity and development, the

next step is to understand the underlying processes that control and limit reiteration in order to guide empirical investigation. In this next stage of model development it is important to specify assessment objectives *a priori*. While knowledge gained from the previous modeling exercise guides further development, I treat subsequent model development as a new effort with its own assessment process. For the development of both the optimization objectives and the process model, I continue to use the guidelines of pattern-oriented modeling and the recommendation of Schmitz (2001) to use natural history in the process of model development. The objectives and process model are proposed at a lower level of biological complexity and guided by the natural history understanding of the system.

There are two kinds of optimization objectives that comprise the vector objective function. The first kind is a set of empirical objectives that measure whether the model matches the observed pattern. Valuable information is obtained through the parameters required for the achievement of some or all of the empirical objectives. In Chapter IV, the results of the optimization will be thoroughly interrogated for any possible model deficiencies. Simultaneous achievement of the empirical objectives with satisfactory parameterizations is an indication of an adequate model. The second kind of optimization objective is the theoretical objectives that are utilized in an optimality study in order to evaluate possible constraining forces in the system (Chapter V). The empirical objectives also play a role in the optimality portion of the study. Satisfaction of one or more of the empirical objectives would enable, in the optimal frontier, inclusion of a solution that may not perform as well with respect to the theoretical objectives (Figure 3.1). This creates a partition in the Pareto optimal frontier of solutions that satisfy the empirical objectives. In this manner I am able to compare model results that are consistent with the observed pattern with model results that may not be consistent, but perform better with respect to the theoretical objectives (Figure 3.1).

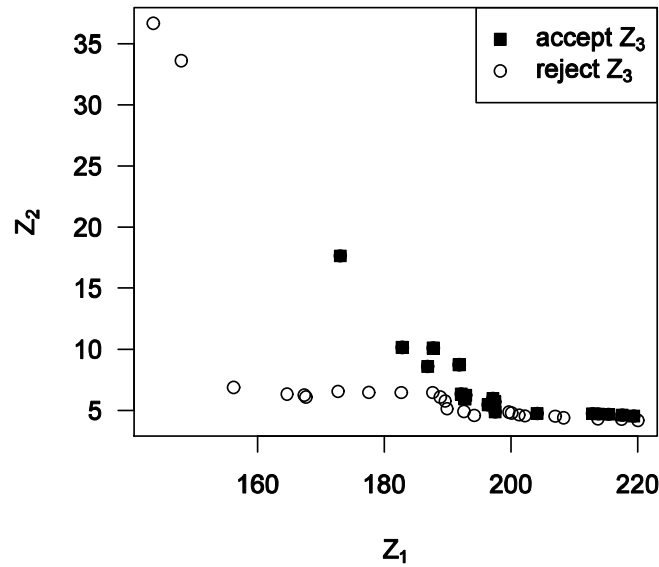


Figure 3.1. Example of a non-dominated Pareto optimal frontier, with two continuous objectives (Z_1 and Z_2) and one binary objective (Z_3). The binary objective is “accepted” when the model result is within the target range, rejected otherwise. We see two distinct areas in the optimal frontier, where solutions that accept Z_3 tend to not perform as well with respect to Z_2 , particularly at low values of Z_1 . At high values of Z_1 , theoretical objective performance is almost indistinguishable between solutions that satisfy Z_3 and those that don’t (note some solutions appear dominated in this plot due to additional objectives in this optimization that are not pictured). This is an example that shows the tradeoff between Z_1 (l_{path}) and Z_2 (n_{over}), which differs between branches that satisfy SCUs (Z_3) and those that don’t.

The modeling study will be used to expand the goals of BRANCHPRO and the questions to be investigated through the analysis. The model structure, parameters and objectives will investigate, through the analysis of the Pareto optimal set and frontier, both specified and unspecified assumptions that are found to be important. The major modeling questions are related to how the old-growth *P. menziesii* trees in the WRCCRF are growth limited with respect to height increment and crown expansion (Ishii and Ford 2001), and an underlying assumption of the optimization is that the observed growth pattern compensates for the constraint on height growth and crown expansion in order to perform major branch functions. Ishii et al. (2007) review how proleptic reiteration may compensate for postulated constraints in old-growth systems under four major categories. It may (1) compensate for an increase in the respiration/photosynthesis ratio, (2) reduce hydraulic limitation, (3) restrict nutrient limitation, and (4) mitigate genetically programmed senescence through the proliferation of younger tissue. The modeling

questions presented here are designed to evaluate through simulation and optimization how the morphology may compensate for the size constraints.

Modeling questions

The larger question the new model seeks to answer is:

How does the branch development pattern of old-growth *P. menziesii* at the WRCCRF compensate for size-related constraints on the species?

There are two major pieces necessary to answer this question: the quantification of (A) the proposed constraints and (B) the processes used to model the branch development pattern. There are three relevant questions for the proposed constraints.

- A.1. What are the major system constraints?
- A.2. Of the architectures possible in the model structure, which architecture (branch development pattern) best compensates for the constraints?
- A.3. For which constraints (if any) does the observed *P. menziesii* branch development pattern compensate?

For question A.1, the set of explanations for what constrains growth in old forests includes a shift to maintenance respiration that reduces the net primary production (NPP) of a forest, hydraulic limitation that reduces photosynthesis with reduced stomatal conductance, and nutrient limitation in the soils of older forests, which reduces photosynthesis (Ryan and Yoder 1997; Hubbard et al. 1999; Bond 2000; Hubbard et al. 2002; Ryan et al. 2004; Ryan et al. 2006). In the new model, I focus on the possible effects of branch architecture on hydraulic limitation and foliage production with respect to effective foliage display. The constraints are then compared through the theoretical objectives quantified for each type of constraint (hydraulic, foliage production) in the system. These are possible alternative theories to explain the constrained growth, and the optimization objectives will allow us to evaluate the relative importance of these theories.

Questions A.2 and A.3 are evaluated through analysis of performance with respect to the theoretical optimization objectives for branches that satisfy the empirical

objectives and branches that don't satisfy the empirical objectives. For example, in Figure 3.1 there is an evident trade-off between objectives Z_1 and Z_2 . If objective Z_3 represents an empirical requirement, then it is obvious that solutions that do not satisfy the empirical requirement perform better with respect to Z_2 than objectives that do satisfy the empirical requirement (Figure 3.1), particularly at lower values of Z_1 . From the pattern it is also clear that, if only the two continuous objectives were minimized, the Pareto optimal frontier may not include solutions that satisfy the empirical requirement. In essence, there are two optimal frontiers that are defined by whether the empirical objective is satisfied or not, and in this case they overlap on the bottom right-hand side of the plot. From analysis of this plot, answers can be proposed for questions A.2 and A.3. If objectives Z_1 and Z_2 were possible constraints, and Z_3 describes the *P. menziesii* branching pattern, then clearly *P. menziesii* performs better with respect to Z_2 only at high values of Z_1 . If the solutions in the Pareto frontier that do not satisfy Z_3 nonetheless reflect an alternative branch morphology that also exists in the old-growth system, then that pattern probably best compensates for Z_1 (Question A.2) and Z_1 may also be constraining in the system (Question A.1). Of course, the analysis is complicated by a higher-dimensional objective space, yet the underlying strategy to answer the three questions is the same and will be applied in the analysis of model results.

For the processes used to model the branch development pattern, I focus on the following question:

- B.1. What is the relative importance of different branch structural characters (e.g., architectural order, foliage overlap, nearby growth) in determining the observed branch development pattern in *P. menziesii*?

The modeling of the branch development pattern is conducted in the context of the two major processes understood to form the basic morphological structure in *P. menziesii*: first proximal development of new foliage clusters is accomplished through delayed break of a dormant bud from suppression, then proliferation of shoots in the newly developing foliage cluster is controlled by the bifurcation of the active nodes (Figure 2.1a). I define variables that characterize conditions that might influence the

probability of bud release from suppression and the bifurcation of active nodes. Their relative importance will be assessed through the resulting distribution of parameter values associated with each variable from the optimization. How these variables control branch development is assessed through the post-processing of simulations on the optimization results. If the variables chosen are inadequate to simulate the *P. menziesii* branch pattern, the variables and their functional forms are re-evaluated.

In the next section, I give the biological background and justification for the optimization objectives. I then give the biological background and justification for the variables used in the growth functions, and how they are combined mathematically to determine bifurcation and release from suppression. For each objective and growth process, I describe the type of relevant process, the biological variable used to represent the process, and the function used to quantify the effect of the biological variable.

Optimization objectives

The optimization objectives bound the theoretical inference that can be accomplished by the modeling effort. Turley (2001) recommends that objectives include three major types: (i) those that test the quantitative domains of the output (i.e., data driven—are my model results consistent with empirical observations?); (ii) those that examine features that theory explicitly predicts (i.e., process driven—are my model results consistent with what the theory predicts?); and (iii) those that provide a means to detect if an alternative theory is more likely (are my model results consistent with theory 1 or with theory 2?). In the context of the current modeling study, the domains of (ii) and (iii) overlap. The theories that I am examining are possible alternative explanations for the constrained branch growth in *P. menziesii*. The theoretical objectives measure both features that the theory explicitly predicts (process A is a constraint on old-growth branch development) and which of the theory predictions are more likely (process A is a more dominant constraint than process B). The three categories proposed by Turley (2001), while useful as a guideline, should not be considered as strictly alternative objective types. In all multi-objective optimization studies, the objectives should be

chosen with a strong understanding of their purpose and their empirical and theoretical implications. In particular, I use the optimization objectives to distinguish among growth constraints and branching patterns, and to identify the constraints that might dominate in the old-growth *P. menziesii* system.

The empirical objectives are designed to capture the basic branching pattern of *P. menziesii*, including the observed foliage demography and the basic observed architectural model. I assume that, for any branch to be considered an adequate representation of branch development, it must at least satisfy the four empirical objectives described below. This constrains a subset of the optimal space to the characteristic observed pattern and would reveal if the model may not be able to satisfy the observed pattern. Simulations from emergent parameter vectors in the optimal set that do satisfy the empirical objectives will then be conducted to uncover whether the model fits the empirical objectives in a manner inconsistent with observed branch development (Chapter IV). Although the goal is for the empirical objectives to capture the observed pattern as fully as possible, it is impossible to account for every relevant aspect of the system. It may be that some other characteristic emerges that does not match the empirical observations, and this would reveal a model inadequacy that should be investigated.

The theoretical objectives are designed to capture major functions and possible constraints on branch growth. The constraints are hydraulics and mechanical load, with foliage display a major branch function. Measures of these constraints will be chosen to differentiate among potential processes for which the observed branch pattern may compensate, as well as yield alternative branch patterns that may perform better with respect to the theoretical objectives. These are optimized within the feasible space dictated by the model structure (e.g., Figure 1.1). In the context of Figure 3.1, Z_1 and Z_2 represent theoretical objectives, and Z_3 represents an empirical objective. Next, I describe the quantitative measures of the empirical and theoretical objectives.

(i) Empirical measures (Table 3.1: 1–4)

In Chapter II, I showed that the numbers of foliated shoots and SCUs are metrics of the branch growth pattern that are necessary (but not sufficient) to describe the *P. menziesii* branch-growth pattern. I therefore include these two measures as the first two components of the empirical vector objective function (*nfoliated*, *nSCUs*; Table 3.1). A third measure that would indicate the simulation is an adequate representation of the branch pattern is the length of the branch. A branch that achieves the correct number of shoots with a branch that is shorter or longer than observed branches would not represent the observed morphology. Branch length (*length*) will be the third component of the empirical vector objective function (Table 3.1). These first three measures of model performance are evaluated with binary error measures with target ranges set by the range of values observed by Ishii and Ford (2001). Simulated branches that have foliated shoots, SCUs and branch length within the respective target ranges are considered to match the observed pattern; in the assessment context, these are optimal relative to simulated branches that are not within the target ranges. The fourth empirical measure (*arch_mod*) evaluates whether the branch follows the basic architectural model of *P. menziesii*, i.e., a dominant main order-1 axis with reduced growth and bifurcation for higher-ordered axes. This is given a value of 1 if, on the simulated branch, the bifurcation rates for order-1, order-2 and order-3 shoots emerge in descending order; zero otherwise. The *arch_mod* objective is optimized with continuous error measures (Appendix A), where the target value is 1.

Table 3.1. Optimization objectives. The name of each is listed in the text. The kind of objective is either data/empirical and theoretical. The first three objectives are labeled data because the target ranges are defined by measurements of each variable on *P. menziesii* branches. For these first three measures, the first target range listed is for Yr90 branches, the second target range is for Yr145 branches. For the measures evaluated with continuous errors, the target is a single value, and for minimization the target is 0. The value of each measure of model performance is detailed in the text.

Name	Type	Target	Value
1. Foliated shoots	Data	(2500,6500) (10000, 17500)	$n_{foliated}$
2. SCUs	Data	(20,50) (85, 160)	n_{SCU}
3. Branch length	Data	(240,430) (490,810)	$length$
4. Architectural model	Empirical	1	1: ($Rb_{ord1} > Rb_{ord2} > Rb_{ord3}$) 0: otherwise
5. # turns	Theoretical	Minimize	$\frac{1}{n_{term}} \sum_{i=1}^{n_{term}} n_{turns}(i)$
6. Path length	Theoretical	Minimize	$\frac{1}{n_{term}} \sum_{i=1}^{n_{term}} l_{path}(i)$
7. Mechanical load	Theoretical	Minimize	$\sqrt[4]{\frac{P_{1/2}}{8P_1}} + \sqrt[4]{\frac{P_{2/3}}{27P_1}}$
8. # overlapping shoots	Theoretical	Minimize	$\frac{1}{n_{foliated}} \sum_{i=1}^{n_{foliated}} n_{over}(i)$

The empirical observation upon which the model relies is a set of 9 branches from old-growth *P. menziesii* at the WRCCRF that were harvested and dissected in 1998 (see Chapter II; Ishii and Ford 2001). There was one branch from each of three trees from the upper, middle and lower crown. The branches in the lower and upper crown were of similar age, as the lower-crown branches were of epicormic origin. Although these are physiologically distinct branches that proliferate in variable light environments, they do represent the range of values that can be achieved by branches in old-growth *P. menziesii* through 90 years of growth. For the current model, the process structure does not distinguish among crown levels, nor does it distinguish whether branches are of epicormic origin. To ensure that the model results encompass what is possible for a Yr90 branch, I group the upper and lower-crown branches in order to set the binary error interval for the first three empirical objectives (see Ishii and Ford 2001 for summaries of the branches). The middle-crown branches were all near 145 years. This establishes two

age points in the development of branch growth that can be compared to simulated values, and binary error ranges are set at each point (Table 3.1).

(ii & iii) Theoretical objectives (Table 3.1: 5–8)

Authors have optimized plant morphologies for various functions individually, including effective leaf area (Fisher and Honda 1977; Honda and Fisher 1978; Honda 1978; Fisher and Honda 1979; Honda and Fisher 1979) and the equitable distribution of leaf clusters (Honda and Fisher 1979). They conclude that a single criterion is insufficient to explain branch form and suggest optimizing the following: exposure to sunlight, mechanical stability, heat exchange and water efficiency. Farnsworth and Niklas (1995) make similar recommendations in their concept of adequate design in plant forms, and they see organisms as representing a multitude of solutions to a multi-goal optimization problem that has no definitive answer. They optimize a single function that combines light interception, mechanical stability and reproductive success and the minimization of total surface area in order to evaluate the adequate design of trees (Niklas and Kerchner 1984; Farnsworth and Niklas 1995; Niklas 1997a,b; Niklas 1999). More recently, Sterck et al. (2005) conducted simulations of plant morphology under different light environments and constraints (no constraints, carbon economy, leaf longevity, self-shading) and utilized simulations to comparatively unravel the interaction effects of the different constraints. They found interactions between apical control and light response that contribute differences in crown architecture and growth between tall vs. short species. In addition, correlated leaf traits result in a growth vs. survival trade-off between pioneer and shade-tolerant species as well as a growth-adult stature trade-off as observed between short-lived vs. long-lived pioneers. Pearcy et al. (2005) identified the following functions and constraints of plant architecture: light capture, competition between plants, reproduction, damage risk, water and heat stress, maintaining upright growth habit, water supply, developmental and allocational constraints.

These studies demonstrate that it is essential to integrate multiple forces and constraints in the analysis of plant form. For the purpose of the branch optimization, I

focus on the first two of the four possible ways proposed by Ishii et al. (2007) by which *P. menziesii* branch morphology may compensate for size constraints (hydraulic limitation, increased respiration:photosynthesis ratio, nutrient limitation and genetically programmed senescence). For hydraulic limitation, I propose both path length and the number of junctions as possible causes of the limitation (Ryan and Yoder 1997; Hubbard et al. 1999; Bond 2000; Hubbard et al. 2002; Tyree and Zimmerman 2002). For the respiration to photosynthesis ratio, Ishii et al. (2007) describe the increasing of productive tissue relative to non-productive tissue as one way in which proleptic reiteration may decrease the respiration to photosynthesis ratio. By regenerating foliage on existing major branch axes, the investment in further wood growth and maintenance would be lower than if the foliage were added at the terminal of the branch. This is a simplification that ignores variable foliage respiration (Brooks et al. 1991 for *Abies amabilis*) and woody tissue respiration (Pruyn et al. 2002) within tree crowns. The ratio of productive to non-productive tissue can be affected morphologically by the placement of foliage so that the investment in further wood growth is minimized along the branch axis. For a given amount of foliage, this would result in a higher ratio of photosynthetic to non-photosynthetic tissue (more foliage, less wood). Another way the ratio of photosynthesis may increase could be effective foliage display to light by the production of epicormic axes (Ishii et al. 2007). A morphology that maximizes foliage display would potentially increase photosynthesis and thereby reduce the respiration to photosynthesis ratio. In the next section I describe the functions for each of these theoretical objectives.

Hydraulic limitation (Table 3.1; 5,6)

There are two possible causes of hydraulic limitation in large, complex trees that I incorporate into the optimization. These are hydraulic constrictions at plant junctions and increased resistance with path length. The cumulative effect of hydraulic constrictions at branch junctions may have a strong negative effect on conductivity (Larson and Isebrands 1978; Zimmerman 1978; Ewers and Zimmerman 1984a,b; Tyree and Alexander 1993; Tyree and Zimmerman 2002). Compensation for such hydraulic limitation would be a branch architecture that results in the placement of shoots such that water flows through

the fewest junctions. For each terminal shoot, I calculate the number of turns from the branch base to the shoot (n_{turns} ; Figure 3.2a), and the value that is minimized is the per-terminal shoot mean of number of turns:

$$\frac{1}{n_{term}} \sum_{i=1}^{n_{term}} n_{turn}(i). \quad (3.1)$$

This per-terminal foliated shoot value is set to minimize in the optimization (Table 3.1).

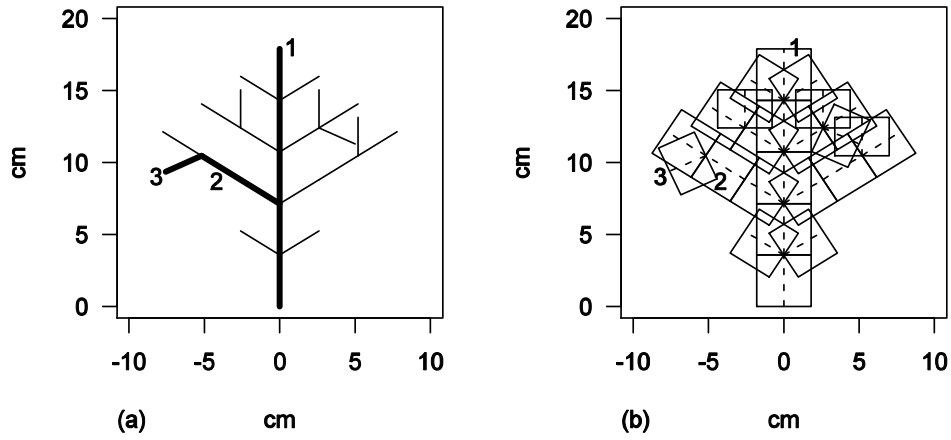


Figure 3.2. (a) Branch ordering system and path length. The thick line represents the path length for shoots 1, 2 and 3. The path length for 3 is the path length for 2 plus the length of shoot 3. The path length for shoot 1 is the length of the structure. The number of turns for shoot 1 is zero, for shoot 2 is one and for shoot 3 is two. (b) Foliage rectangles and overlap. Shoot 1 has an overlap count of four, shoot 2 has an overlap count of four and shoot 3 has an overlap count of two.

A prominent hypothesis to explain reduced growth with increasing size is that hydraulic resistance increases with path length (Ryan and Yoder 1997; Hubbard et al. 1999; Bond 2000; Hubbard et al. 2002), thus reducing conductivity and photosynthesis. A branch architecture that reduces path length to active meristems would provide a compensation for the increased resistance due to the old-growth *P. menziesii* branches with long path lengths. I calculate the per foliated terminal shoot mean path length (cm) to terminal foliated shoots on each branch at the end of the simulation. The path length is calculated as the sum of the lengths of shoots from the base of the branch to the current terminal shoot (l_{path} ; Figure 3.2a), or

$$\frac{1}{n_{term}} \sum_{i=1}^{n_{term}} l_{path}(i) . \quad (3.2)$$

Mechanical load (Table 3.1; 7)

This branch model does not include functions for diameter growth, which is determined both by the mechanical support requirement for the branch and possibly by the branch hydraulic architecture. The requirements for mechanical support increase with the length of the branch and with the location of the mass of foliage distributed along the branch. The objective for mechanical load compares, for a branch of a given length, the support required at the base of the branch relative to that required distally. For two branches of equal length and total foliage mass, the mechanical requirement at the base of each branch is assumed to be equal. The distribution of foliage calculated as point loads (P) along the major branch axis would determine the relative mechanical requirement further along the branch—a more efficient pattern of foliage would require less mechanical support distally along the major branch axis.

In their model of crown architecture of *Psychotria*, Pearcy et al. (2005) use the following equation to calculate deflection of a branch:

$$\frac{4PL^3}{3E\pi R^4} , \quad (3.3)$$

where P is the load on the distal end of the branch, E is the modulus of elasticity, L is the length and R the radius (Morgan and Cannell 1988).

I use this equation to compare the branch radius required at the base of the branch and that required at two points distal: 1/2 the branch length and 2/3 the branch length. For a given branch the modulus of elasticity is the same, and π is a constant. I ignore the distributed load due to extension of the branch (p as defined in Morgan and Cannell 1988) and I focus on the point loads (P), due to lateral growth and reiteration. Since the

branch is held at a constant deflection, then for a point at the base of the branch and a point at 1/2 the length the relative radii are:

$$\frac{4P_1L_1^3}{3E\pi R_1^4} = \frac{4P_{1/2}\frac{1}{2^3}L_1^3}{3E\pi R_{1/2}^4}, \quad (3.4)$$

$$\sqrt[4]{\frac{P_{1/2}}{8P_1}} = \frac{R_{1/2}}{R_1}. \quad (3.5)$$

L_1 is the length of the branch, P_1 is the total number of shoots in the branch, $P_{1/2}$ is the number of shoots attached to lateral and SCU axes distal to the point 1/2 the length of the branch. This equation calculates the ratio of radius required for mechanical support between the base of the branch and the point distal at 1/2 its length. Similarly, for a point distal 2/3 the branch length the equation reduces to:

$$\sqrt[4]{\frac{P_{2/3}}{27P_1}} = \frac{R_{2/3}}{R_1}. \quad (3.6)$$

This allows a calculation of relative mechanical requirements without simulating radial growth and carbon allocation. For example, if there are 6000 total shoots on a 4.3 m branch and the shoots are evenly distributed throughout the branch length ($P_1 = 6000$; $P_{1/2} = 3000$; $P_{2/3} = 2000$), then

$$\frac{R_{1/2}}{R_1} = 0.500, \frac{R_{2/3}}{R_1} = 0.333. \quad (3.7)$$

If, for another example, I assume that the proximal half of the branch length lacks foliage (which may be more typical in an old-growth branch), and the remainder of the shoots are evenly distributed through the distal half ($P_1 = 6000$; $P_{1/2} = 6000$; $P_{2/3} = 4000$), then

$$\frac{R_{1/2}}{R_1} = 0.595, \frac{R_{2/3}}{R_1} = 0.396. \quad (3.8)$$

This illustrates that, as the shoots are clustered more distally on the branch axis, the radius of the distal part relative to the base necessary to support the weight gets larger. This is a measure of the efficiency of the branch to distribute foliage along its major axis. The maximum value for the point 1/2 the length of the branch is 0.595, and for the point

2/3 the length of the branch it is 0.439 (Table 3.2). The objective is to minimize the sum of these two ratios.

Table 3.2: Maximum and example values of mechanical load objective (Table 3.1, 7).

Total foliated shoots	Shoots beyond 1/2 branch length	Shoots beyond 2/3 branch length	Load at 1/2	Load at 2/3	Mechanical load objective
6000	6000	6000	0.595	0.439	1.034
6000	6000	4000	0.595	0.396	0.991
6000	3000	2000	0.5	0.333	0.833
6000	4000	1000	0.537	0.28	0.817

Foliage display (Table 3.1, 8)

An important function of the branch is the effective and efficient display of foliage to harvest light for photosynthesis. Honda (1978) calculates effective leaf area as the area remaining after the area of overlapping foliage clusters is removed from a branch structure (Fisher and Honda 1977; Honda and Fisher 1978; Fisher and Honda 1979; Honda and Fisher 1979). As a less computationally intensive surrogate, I calculate the number of foliated shoots whose foliated area intercept the foliage area of the current shoot (n_{over} ; Table 3.1). In *P. menziesii*, the area occupied by needles can be approximated by a rectangle whose length is the length of the shoot and whose width is twice the needle length (Figure 3.2b). This ignores plasticity in the arrangement of needles in sun and shade environments (e.g., Sprugel et al. 1996), and is calculated only for the foliage within the current branch. Between-branch shading is ignored in the current simulation. Although I do not purport that branches within the crown of *P. menziesii* do not shade each other, *P. menziesii* does have a relatively low branch density, particularly in its middle and lower crown (Ishii and Wilson 2001). Furthermore, the within-crown light environment is highly variable (Ishii and Wilson 2001), and we lack a sufficient description of the light environment that could justifiably be included in the model. The value of n_{over} serves as an approximation of foliage display.

In *P. menziesii*, needle length varies vertically through the crown (Woodruff et al. 2004) and as trees age (Apple et al. 2002). For *P. menziesii* trees greater than 40m in

height (the middle and upper crown) at the WRCCRF, Woodruff et al. (2004) observed needle lengths ranging from 13–22 mm (estimated from Fig. 7 Woodruff et al. 2004), and Apple et al. (2002) report a mean needle length of 18.88 mm in 450 year-old *P. menziesii* trees at the WRCCRF. For comparing branch morphologies, I choose a constant needle length of 18 mm. Rather than the computational burden of calculating effective leaf area (the non-overlapping foliage areas), I approximate overlap with a tally of foliated shoots whose foliage rectangles overlap with the current shoot (n_{over}).

Shoots A and B are assumed to overlap if their foliage rectangles overlap, and for shoots of the same generation overlap is completely symmetric (if A overlaps B, then B overlaps A). Otherwise, a shoot of a higher generation is assumed to overlap shoots of lower generation (if A and B rectangles intercept, and A is generation 2 and B is generation 1, then A overlaps B but B does not overlap A). I assume that the higher number of rectangles that overlap a given shoot, the lower the light interception for that shoot. To summarize the branch, I calculate a per foliated shoot mean of n_{over} , and I assume that the lower this value, the more effectively the branch displays its foliage to available light. The objective that is minimized is the per foliated shoot mean of number of overlapping shoots,

$$\frac{1}{n_{foliated}} \sum_{i=1}^{n_{foliated}} n_{over}(i). \quad (3.9)$$

This full set of empirical and theoretical objectives (Table 3.1) is used to assess the ability of the model to match the observed branch pattern of *P. menziesii* at the WRCCRF, and to differentiate the relative performance of branching patterns with respect to the four theoretical functions of branch growth.

Summary of component ecological postulates: Theoretical objectives

Each theoretical objective represents a postulate for a function of branch development that may be important and/or limiting in old-growth *P. menziesii*, and possibilities for functions for which the observed development pattern might compensate.

The first theoretical optimization objective, hydraulic path length, represents this postulate:

Postulate O.1

P. menziesii old-growth branch morphology minimizes the mean path length to active terminal nodes.

This postulate represents the second theoretical optimization objective, constrictions at cumulative hydraulic junctions:

Postulate O.2

P. menziesii old-growth branch morphology minimizes the mean number of cumulative turns to active terminal nodes.

The third postulate refers to the supposition that old-growth trees are constrained with respect to mechanical requirements. The postulate that the third optimization objective represents is:

Postulate O.1

Through the distribution of foliage load, *P. menziesii* old-growth branch morphology minimizes the radial growth requirement distally on the branch relative to the requirement at the base of the branch.

For the final theoretical objective, it is recognized that foliage display is an important branch function, and the objective (Table 3.1, 8) represents the following postulate:

Postulate O.4

P. menziesii old-growth branch morphology minimizes within-branch foliage overlap.

Given the possible tradeoffs among the theoretical objectives, it is not expected that any one objective will be minimized, or all objectives minimized simultaneously. This analysis will show where among the possible morphologies *P. menziesii* lies with respect to the proposed theoretical constraints, and which alternative morphologies may

perform better for each objective, and combinations thereof. I do not suggest that the model analysis will allow for a strict reject/not reject of any of these postulates in a statistical sense. Rather, I will frame conclusions by evidence in favor of or evidence against each proposition. This will guide suggestions for the next stage of empirical research, which will refine the conclusions from the model analysis. The model that the objectives will be used to assess and optimize will be defined by two major growth functions: release from suppression and bifurcation. These are described in the next section.

Model growth functions

In the original BRANCHPRO model, bifurcation is defined for all shoots differentiated only by architectural order, or if the shoot is a newly-initiated epicormic. In order for the branches to achieve minimum growth without growing too large, a condition was placed that reduced bifurcation of new epicormic shoots with increasing generation (Kennedy 2002; Kennedy et al. 2004). The rules for suppressed bud release are also relatively simple, with little consideration of the condition of the bud when the timing of initiation is determined. However, it is known that bifurcation of a shoot is non-stationary as a branch ages and across the branching complex (Borchert and Slade 1981; Steingraeber and Waller 1986; Kennedy et al. 2004). The growth processes of BRANCHPRO, while informative, are inadequate to explain the non-stationarity of bifurcation and the conditions that improve the probability a bud is released from suppression.

I improve the BRANCHPRO model to incorporate processes that might influence bifurcation and bud release. To develop the growth functions, I follow a similar method to that presented by Sterck et al. (2005). They modeled the production of new metamers as controlled by a flush probability (P_F), which depends on probabilities that relate to (1) metamer position (e.g., order), (2) an axillary inhibiting factor (status of neighboring metamers), (3) the local light level and, indirectly, (4) the carbon status of the tree. In their model, these factors are multiplied to give a final flush probability. For my model, I

will produce an additive function for the rate of bifurcation of active nodes, and an additive function for the probability that a suppressed bud is released. These incorporate simple functional relationships for variables chosen to represent architectural, light and hydraulic status.

Saturating functions

The general strategy for the functional forms that relate each independent variable to the response is to model a saturating effect, such that the effect of the independent variable increases, then levels off. The motivation for using functions that saturate is the expectation that effects tend to have thresholds over large scales (e.g., the effect of an active terminal on bud release would diminish with distance of the terminal to the suppressed bud), and in particular that the branching system needs to be constrained (Kennedy et al. 2004). This assumes that at some point any increase in the independent variable does not change its effect on the response, i.e., the effect diminishes above a certain level. For a starting point, I apply this generic functional form for some independent variable X (Figure 3.3a):

$$f(X) = 1 - \frac{1}{X}. \quad (3.10)$$

The parameters for the function regulate the magnitude of the effect. If β is the parameter that is estimated by the optimization procedure for the independent variable X , then:

$$f(X) = \beta \left(1 - \frac{1}{X} \right). \quad (3.11)$$

This is the general form used to relate the independent biological variables to the bifurcation and probability of release responses.

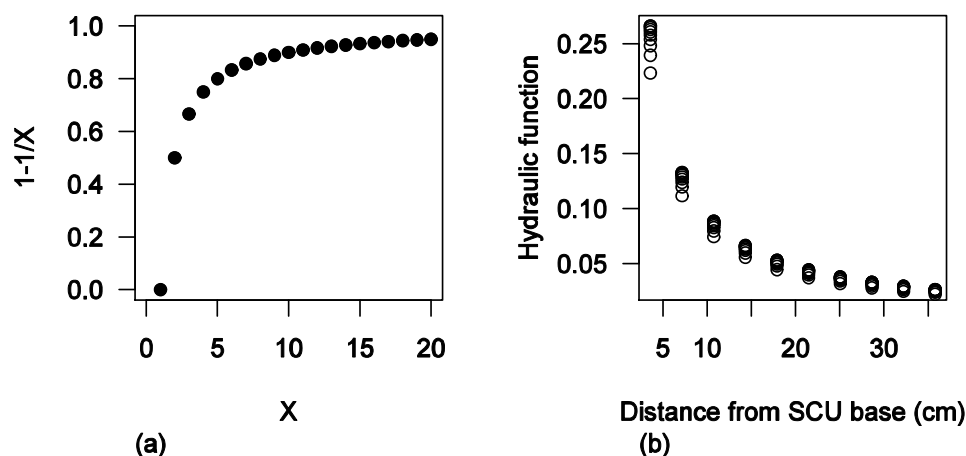


Figure 3.3: (a) Generic saturating function for independent biological variables. (b) Hydraulic function related to d_{base} (distance from SCU base), for a generation 1 order 1 shoot with values of t_{base} from (0,20) by 2 for each level of d_{base} .

Type of process and variable used to represent it

The *P. menziesii* trees at the WRCCRF are clearly constrained in increment of height growth and crown expansion (Ishii et al. 2000; Ishii et al. 2003; Bond et al. 2007). Regardless of the cause of this restriction, the growth pattern in *P. menziesii* trees has somehow compensated for the size constraint in order for individual trees to continue to persist when height growth increment and crown expansion are negligible.

Basic requirements for growth in a tree include photosynthesis (carbon and light), water and nutrients. Hormones play an important role in determining patterns of growth and development, particularly in establishing the apical control evident in the *P. menziesii* growth pattern. To that end, I focus on the local hormonal, light and hydraulic status to determine the bifurcation rate value of the current node and the probability of bud release from suppression. Nutrition status is assumed to be the same for branches on the same tree and not to be influenced by the branch structure. Next, I elaborate on the variables chosen to represent each of the types of factors that influence growth for *P. menziesii*, beginning with the probability a suppressed bud is released.

Probability a suppressed bud is released: background

Epicormic initiation requires the release of a previously suppressed bud. It is unclear how or why this occurs, and only a few studies have evaluated epicormic sprouting in conifer trees (e.g., Bryan and Lanner 1981). The cause of epicormic sprouting is usually studied in deciduous hardwood trees, and mainly in the form of epicormic branching along the bole of the tree (e.g., Kormanik and Brown 1969; Wignall et al. 1987; Nicolini et al. 2001). There is some indication of possible hormonal effects where the bud is suppressed from an active terminal or nearby active lateral axes (Kormanik and Brown 1969; see Discussion in Kennedy et al. 2004), although this is complicated by the correlation of the onset of epicormic sprouting with reduction of cambial growth (Bachelard 1969; Nicolini et al. 2001). Nicolini et al. (2001) investigated suppressed beech trees that had slowed in growth. In this sense they are similar to old-growth *P. menziesii* because both species have reduced growth relative to non-suppressed or younger trees (although the cause of the growth constraint differs markedly in the two cases). Remphrey and Davidson (1992) found no relationship between epicormic formation and stem diameter in *Fraxinus pennsylvanica*, which they interpreted to indicate no correlation with changes in cambial growth. Rather, epicormic formation was associated with an increase of shoot death in older crown regions (Remphrey and Davidson 1992). In oak, Wignall et al. (1987) found that bud emergence is inhibited by exogenous IAA, implying it is also inhibited by endogenous IAA. Wignall and Browning (1988) also found that epicormic buds can escape inhibition despite an increase in cambial zone IAA. Either the buds themselves overcome the inhibition enforced by IAA, or they attract what is necessary from local sources. This is in contrast to release due to the inhibition caused by IAA being reduced. They found no relation between a decrease in cambial region ABA (hence reduction in cambial growth) and bud emergence. There is no evidence that bud release and epicormic sprouting are related to light levels (Wignall et al. 1987; Wignall and Browning 1988), although light levels may play a role in further proliferation of shoots once the bud is released (Quine 2004).

There has been little direct investigation of hydraulic effects for epicormic bud break. With respect to hydraulic architecture, Wignall (1987) noted that variations in the degree of bud development and the development of their vascular connections explained why some buds are able to escape inhibition. In contrast, Bachelard (1969) found that the addition of water (hence a surplus) actually inhibited epicormic shoot formation in eucalyptus (although he offered no mechanistic explanation). Given that there is no clear evidence in favor or against possible hydraulic effects, in the new model structure I allow for a hydraulic effect in the determination of probability of bud release.

Hydraulic conductance decreases with decreases in diameter, and decreases with increased path length; therefore it is lowest at branch tips (Zimmerman 1978; Tyree and Ewers 1991; Tyree and Zimmerman 2002). An epicormic may sprout more proximally along an axis, so gaining the advantage of higher conductances associated with greater diameter and shorter path lengths. For a given stomatal conductance, these would have lower water deficit than shoots at the distal end of the parent branch (Ishii et al. 2007). Hydraulic conductance of epicormic branches is consistently higher than non-epicormic branches at the same diameter of the parent branch (unpubl. data, P.J. Schulte). In addition, most of the resistance to water flow is thought to occur at the last meter or less of the path from the base of the path to its tip (Tyree and Ewers 1991). Therefore, if we assume epicormics can take advantage of higher conductance and lower resistance closer to the base of the branch (or the parent SCU axis), then their initiation may be due to some effect of increasing resistance of the tip of the parent axis as it lengthens, with better hydraulic connections more proximally along the major branch axis. This may be tempered by the length of time the bud is suppressed. I assume initiation is independent of light status, although further SCU development will depend on light status as defined in the function for bifurcation of existing shoots. In this manner, epicormic initiation can result in multiple neighboring clumps of foliage that compete given their status defined by the bifurcation function.

Most observations indicate that the appearance of epicormics on a tree is associated with some kind of slowing or constraint on growth. This observation confounds observed hormonal effects when hormones are manipulated in the system (e.g., Wignall et al. 1987; Wignall and Browning 1988) because active growth affects hormone levels. Overall we assume that the slowing of growth is a positive correlate of bud release, and that might have hormonal implications. In my function for bud break, I represent the slowing of growth as well as potential hydraulic effects.

Process rules and independent variables (Table 3.3)

In the new model, as in BRANCHPRO, bud release from suppression is limited to order-1 axes; only one bud per order-1 parent shoot is allowed to proliferate. The timing of the buds that are released determines the potential location and placement of new SCUs. In the new model there is some baseline probability of release (p_0), which is then increased by favorable local conditions of the bud. These include reduced growth locally on the parent SCU and possible hydraulic influences.

Table 3.3: Independent variables for the growth functions.

Function	Variable	Description	units
p	t_{bud}	time since formation (age) of suppressed bud	years
	a	proportion of inactive axes (two lateral, one extension) relative to current suppressed bud	unitless (0,1/3,2/3,1)
	d_{SCU}	distance (in number of shoots) to the nearest developing SCU on the same axis as current suppressed bud	# segments
	d_{stem}	distance (in cm) to the main tree stem	cm
r	o	architectural order of current shoot	unitless
	n_{over}	the number of overlapping foliated shoots	# shoots
	$n_{turns} + t_{base}$	number of turns to current shoot, age of bud at SCU base when it was released from suppression	unitless + years
	d_{base}	distance from the SCU base to the current shoot	cm

Reduced local growth

I propose that the major cause of bud suppression is the actively proliferating parent SCU. Therefore the probability of release increases as that influence decreases. Ishii and Ford (2001) recorded the timing of initiation for epicormic sprouts on branches of *P. menziesii* at the WRCCRF, and found that the proportion of epicormic buds that were released increased as the age of the bud increased, with a mode near 5 years (Kennedy et al. 2004, their Figure 5). This proportion decreased thereafter, although this is probably an approximation given that not all suppressed buds of all ages were recorded, rather only those that did sprout. In another study, they observed epicormic axes that had originated up to 50 years after bud formation (Ishii et al. 2002). Overall, it is clear that, to a point, the probability a suppressed bud is released increases with its age (time since formation, t_{bud}). I surmise this is due to a lessening of inhibition exerted by the growing main axis as it extends beyond the node at which the bud is formed. The effect of this variable is regulated by the parameter p_{bud} :

$$p_{bud} \left(1 - \frac{1}{t_{bud}} \right). \quad (3.12)$$

For the next variable, I measure the proportion of axes (main axis, lateral axes immediately subtended on either side of the bud; a) that are no longer actively growing to represent the release of inhibition by the slowing of growth local growth. The variable a takes on values $\{0, 1/3, 2/3, 1\}$. High values of a indicate that growth is likely slowing on the parent SCU, and it would correlate with reduction in inhibition factors. This is proposed to increase the probability the bud is released from inhibition, and given the finite range of the variable there is no saturating function. The effect is regulated by the parameter p_a :

$$p_a * a. \quad (3.13)$$

The third variable gives a measure of local proleptic reiteration on the main axis of the parent SCU. If the suppressed bud is in the vicinity of another epicormic that formed on the same parent, then the probability the bud is released decreases relative to a bud that is further away along the axis from other SCUs. This is quantified with the variable d_{SCU} , the number of segments to the nearest epicormic that could produce an SCU. If there are epicormics nearby on the same axis, and it is possible that these could inhibit the bud from release. The effect is regulated by the parameter p_{SCU} :

$$p_3 \left(1 - \frac{1}{d_{SCU}} \right). \quad (3.14)$$

Hydraulic status

Finally, I use one variable to represent the hydraulic status of the bud. I assume that, in general, suppressed buds have poor hydraulic status, but that condition improves the closer the bud is to the tree stem and water under higher water potential and to water at higher conductivity. The bud may respond to higher water potential (ψ). The assumption here is that higher ψ will occur when resistance is less. The variable d_{stem} measures the distance (path length, cm) from the base of the branch to the current suppressed bud, and I assume that the probability of release increases as this variable decreases. The effect is regulated by the parameter p_{stem} :

$$p_{stem} \frac{1}{d_{stem}}. \quad (3.15)$$

Four variables are used to model the probability a bud is released from suppression: t_{bud} , a , d_{SCU} and d_{stem} .

$$p(t_{bud}, a, d_{SCU}, d_{stem}) = p_0 + p_{bud} \left(1 - \frac{1}{t_{bud}} \right) + p_a * a + p_{SCU} \left(1 - \frac{1}{d_{SCU}} \right) + p_{stem} \left(\frac{1}{d_{stem}} \right). \quad (3.16)$$

Summary of component ecological postulates: Growth functions

Each component of the growth function represents a postulate for the local conditions that change the probability of bud release. These postulates must be considered in the context of the model structure and optimization objectives. For each postulate I also list the underlying process assumptions in the model structure.

For the probability of bud release the postulates are similar for each independent variable and can be stated as:

The probability of suppressed bud release along order 1 axes of *P. menziesii* increases with:

- (1) the age of the bud
- (2) the proportion of inactive axes
- (3) the distance to the nearest newly forming SCU
- (4) the proximity to the main stem.

Underlying process assumptions:

Bud release from suppression occurs only on order 1 axes, and can occur anywhere along the order 1 axis.

Postulate p.1:

The probability of suppressed bud release along order 1 axes of *P. menziesii* increases as the age of the bud increases (eqn 3.12).

Underlying process assumptions:

The basis of this postulate is that the as the bud ages, the suppression due to the parent axis decreases. Furthermore, the process of release from suppression is assumed to be distinct from the process of bifurcation.

Postulate p.2

The probability of bud release from suppression increases as the proportion of the immediate axes that are no longer proliferating increases (eqn 3.13).

Underlying process assumptions:

The only axes whose growth suppresses the bud are the ones that immediately subtend to the bud.

Postulate p.3

The probability of bud release from suppression increases as the distance to the nearest developing SCU on the same axis increases (eqn 3.14).

Underlying process assumption:

The distance from the current bud to the nearest SCU can be quantified as the number of segments (which is proportional to distance, as each segment along a given axis is the same length), and that the effect of the nearest SCU occurs both distally and proximally (so, both basipetal and acropetal movement of whatever causes suppression).

Postulate p.4

The probability of bud release from suppression increases with decreasing distance to the main stem (eqn 3.15).

Underlying process assumption:

The hydraulic status of a bud is related to path length in a way that is unaffected by the number of junctions.

Each of these postulates will be evaluated through the distributions of parameter values in the Pareto optimal set. Next, I describe the variables used to model bifurcation.

Bifurcation: biological background

It is known that bifurcation of a given parent shoot is non-stationary as a branch ages and across the branching complex (Borchert and Slade 1981; Steingraeber and Waller 1986), although the value of bifurcation is usually within fixed bounds (e.g., I set it to 0-3 for old-growth *P. menziesii*). Kennedy et al. (2004) found that a reduction in bifurcation of new epicormic shoots with increasing generation was necessary to

constrain branch growth. Wignall et al. (1987) and Wignall and Browning (1988) differentiate the role of thinning in suppressed bud release, and in successful growth and longevity of the newly growing axis. They speculate that available light might have an influence on success of the new axis once the bud had successfully sprouted, whereas the number of new sprouts did not differ with the increased light levels. In a series of modeling exercises, Honda et al. (1981) and Borchert and Honda (1984) modeled by the flux of materials through successive branching nodes, and they regulated bifurcation by the amount of material that reaches the node.

Several factors determine growth capacity and patterns in conifer species. The prominent growth pattern and architectural hierarchy (apical control) are likely controlled by hormones, with a dominant main axis exerting inhibition on the lateral axes (e.g., Cline 1991, 1994; Bollmark et al. 1995; Hao-Jie et al. 1996; Cline 1997; Miguel et al. 1998; DeWit et al. 2002). Nutrients play a role, including nutrient partitioning, but we ignore these factors in the current model. Hydraulic architecture and water supply also play a role, especially in large trees that have reduced growth increment. In particular, it was observed that epicormic growth is restricted in the first few years after release (in both increment and bifurcation; Kennedy et al. 2004; Figure 2.1c). This may have a hydraulic cause, in which it takes the newly expanding epicormic axis 2–3 years to develop a strong hydraulic connection. In addition, through simulation studies Kennedy et al. (2004) concluded that reiteration is restricted by the number of successive generations that can accumulate on a branch. They proposed that there is not only a size limitation in large trees, but also a limitation in complexity measured in generations and branching order. This implies a restriction in the number of branching junctions in *P. menziesii*, which may be explained by hydraulic constrictions observed in branching junctions of some species (Zimmerman 1978; Tyree and Alexander 1993; Tyree and Zimmerman 2002). Tyree and Ewers (1991) report that the leaf specific conductivity (LSC) in stem segments distal to junctions are usually more than that of the junction itself, indicating that the effect of the constriction lessens as the branch extends. The local light environment may also have an influence on the placement of shoots. In the

model, branches are grown in isolation and the light environment is not directly measured, so I use a measure of foliage overlap as a surrogate for the light environment (Figure 3.2b).

Bifurcation: process rules, variables and mathematical functions (Table 3.3)

I measure the successful growth of a node by the number of daughter shoots it produces (bifurcation). The model defines a bifurcation rate that is the rate parameter for the Poisson distribution from which the number of daughter shoots is drawn¹.

Architectural hierarchy

I represent apical control in the model by the designation of a dominant first-order axes; order increases with each lateral bifurcation, and bifurcation is limited on higher-ordered axes relative to the main order-1 axis. The first shoot grown in the simulation is order 1, and reiteration is caused by assigning order 1 to the first shoot grown after a suppressed bud is released (Figure 2.1c). Therefore the new epicormic axis exerts apical control over any lateral axes that may result. For each suppressed bud that is released, generation increases by 1. This is in contrast to lateral growth, which results in increasing order; for each successive lateral bifurcation order increases by 1 (Figure 2.1a). Bifurcation is assumed to decrease with increasing order (o), regulated by the parameter r_{ord} :

$$r_{ord} \left(1 - \frac{1}{o} \right). \quad (3.17)$$

Hydraulic restriction

An increase in resistance at branch junctions has been observed to create a hydraulic constriction in various species, although the effect seems to lessen along the

¹ If this number is greater than 3, then 3 daughter shoots are assigned. This has the effect of reducing the expected value of bifurcation below that of the calculated rate, although the relationship is increasing; Figure 2.2



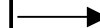
axis distal to the junction (Tyree and Ewers 1991). I assume this constriction is more severe as the age of the bud that results in the junction increases (measured by t_{base} ; base indicates the base of the current axis), and with the accumulation of successive junctions (measured by the number of turns to the current node; n_{turns}). The effect of the constriction is lessened as the axis extends (Figure 3.1b; measured by the distance to the base of the junction; d_{base} , cm); this is considered because LSCs measured distal to the junction are higher than at the junction itself (Tyree and Ewers 1991). Therefore, the higher resistance at the constriction is assumed to not impact growth for foliage at the terminal of the axis after the first few years of growth. I propose the relationship between t_{base} , n_{turns} and d_{base} to explain the observed growth pattern of new epicormic shoots. These have restricted bifurcation for the first two years, then they have higher bifurcation that replicates the growth patterns of other main axes (Figure 2.1c).

Since the constriction was first reported (Zimmerman 1978), it has been more recently argued that the hydraulic constriction has little effect on the water relations of whole trees (Tyree and Ewers 1991; Tyree and Alexander 1993; Tyree and Zimmerman 2002), and that the observed constriction may act as a segmentation to protect the main stem from hydraulic failures (cavitation) in the branches. It is difficult, however, to evaluate the effects of hydraulic constrictions at branch junctions on large, complex old-growth trees. To reach the terminal foliage on old-growth branches, water must travel through many junctions over extreme path lengths. The cumulative effects of these junctions may have a more significant effect when considered for a branch on a large, old-growth tree than for smaller, younger trees. It may be that consecutive accumulations of junctions at the end of a very long path length can have a significant impact on the growth potential of active apices on the branch. The mathematical relationship I propose also models the possible factors contributing to the observed growth pattern in newly forming epicormic structures, i.e., restricted growth the first few years with proliferation like an order-1 axis thereafter. The mathematical relationship for the hydraulic constriction is designed so that the constriction reduces growth for the first shoot that forms from the bud (i.e., the shoot at the base of the axis). For each subsequent shoot

growth, the constriction diminishes (with d_{base}). This is consistent with the observation that the LSC distal to a junction is higher than the LSC at the junction itself (Tyree and Ewers 1991). These effects are combined into a single function that is regulated by the parameter r_{hydr} (Figure 3.3b; Table 3.4):

$$\frac{r_{hydr}}{d_{base}} \left(1 - \frac{1}{n_{turns} + t_{base}} \right). \quad (3.18)$$

Table 3.4. Changes in the effect of the hydraulic constriction on bifurcation as a new axis extends. In this example, the new axis had formed from the release of a bud that had been suppressed for 7 years (t_{base}). This is a second-generation axis (gen), which therefore has 1 turn (n_{turns}). After the shoot that results from bud release is formed, its distance to the base of the junction is its length (d_{base} ; 3.58 cm). The values of these variables result in a reduction of the value of the bifurcation function of $0.224 * r_{hydr}$. As the axis extends, the value of d_{base} increases, whereas the remaining variables retain their original values. This reduces the effect of the constriction each year the axis extends successfully.

			
Year of growth	1st year	2nd year	3rd year
t_{base}	7	7	7
gen	2	2	2
n_{turns}	1	1	1
d_{base}	3.58	7.16	10.74
$\frac{r_{hydr}}{d_{base}} \left(1 - \frac{1}{n_{turns} + t_{base}} \right)$	$r_{hydr} * 0.224$	$r_{hydr} * 0.122$	$r_{hydr} * 0.081$

Foliage display

The local light environment can have an effect on branching potential and patterns. Modeling the actual light environment of a branch or tree is a complicated problem, and it is useful to use other measures to approximate foliage display to light. I approximate foliage overlap with a tally of foliated shoots whose foliage rectangles overlap with the current shoot (n_{over}) as an indirect measure of light exposure. As the number of overlapping shoots increases for a given parent shoot, the rate of bifurcation decreases. This is regulated by the parameter r_{over} . A value of 1 is added to n_{over} in the denominator of the function in order to prevent division by zero, as a value of zero is possible for n_{over} :

$$r_{over} \left(1 - \frac{1}{n_{over} + 1} \right). \quad (3.19)$$

The final functional form for the Rb function for shoot i at year n is

$$r(o, n_{turns}, t_{base}, d_{base}, n_{over}) = r_0 + r_{ord} \left(1 - \frac{1}{o} \right) + \frac{r_{hydr}}{d_{base}} \left(1 - \frac{1}{n_{turns} + t_{base}} \right) + r_{over} \left(1 - \frac{1}{n_{over} + 1} \right). \quad (3.20)$$

For bifurcation, the postulates can be stated as:

Bifurcation rate in active nodes of *P. menziesii* decreases with:

- (1) order
- (2) number of cumulative turns and age of suppressed bud
- (3) number of overlapping shoots.

Underlying process assumptions:

Bifurcation of active nodes is controlled by a modified Poisson process that, for old-growth *P. menziesii*, is truncated at a maximum value of 3; the effects of independent variables are quantified by simple saturating functions (eqn 3.11).

Postulate r.1:

The architectural hierarchy of *P. menziesii*, as defined by the shoot order, decreases the bifurcation rate in active nodes with increasing order (eqn 3.17).

Underlying process assumption:

The architectural hierarchy is represented by shoot order, and this hierarchy is established when a bud is released from suppression and begins to actively proliferate shoots (the assignment of order 1 to new epicormic shoots).

Postulate r.2:

There is a hydraulic constriction in *P. menziesii*, which is controlled by the integrative effects of the number of cumulative turns and age of the suppressed

bud when it originated the axis, which reduces the bifurcation rate in active nodes of *P. menziesii* (eqn 3.18).

Underlying process assumption:

The effects of bud suppression can be separated by the effect on initiation of the bud and the effect on the proliferation of the new axis once the bud is released. The constriction is mitigated so that the effect becomes negligible as the axis extends.

Postulate r.3

The bifurcation rate in active nodes of *P. menziesii* decreases as the number of overlapping foliated shoots at that node increases (eqn 3.19).

Underlying process assumption:

The foliage of *P. menziesii* shoots can be represented by a flat rectangle, whose width is twice the needle length. A fair approximation of foliage overlap is to count the number foliage rectangles that intersect with the current foliage rectangle. Overlap causes shading of foliage below (as defined by the generation of each shoot) and reduces that shoot's vigor.

Growth functions summary

The relationships specified in equations 3.16 and 3.20 are the key improvements to BRANCHPRO. Structurally the model operates as defined in Kennedy et al. (2004) and described in Chapter II (Fig 2.1); the improvement is that the bifurcation rate for each active node is determined by equation 3.20 rather than just the architectural status, and the process of release from suppression is defined by equation 3.16 rather than a draw from a gamma distribution and a simple process rule. The nine parameters that will be searched in the optimization are the vectors of bifurcation parameters ($r_0, r_{ord}, r_{hydr}, r_{over}$) and release from suppression parameters ($p_0, p_{bud}, p_a, p_{SCU}, p_{stem}$). The allowable ranges for each of these will be determined by the initial optimization runs. The model with the functions defined in equations 3.16 and 3.20 will be called BRANCHPRO2.

Model bounds

Once the objectives and model structure are defined, the feasible space of the optimization should be determined. In the case of branch growth, shoot proliferation is highly sensitive to changes in the capacity for reiteration, and simulations show that the branches easily grow to excessive sizes, much larger than any observed branches. In prior simulations, branches could become so large that memory issues caused errors on the computer. To avoid such computational issues, a maximum number of branch segments (MAXSEGS) is set for the optimization. This value is set so that shoot numbers at each time point do not exceed four times the observed values. These values are not too restrictive, given that no solutions in the Pareto optimal sets presented in Chapter IV had any shoot numbers that approached the MAXSEGS values. The simulation of any branch that has a number of shoots greater than MAXSEGS is terminated, and objective values are calculated at that point; multiple runs for that parameter vector are also not performed, and the program progresses to the next individual in the population. This allows for the exploration of objective values for large branches without the size of the branches getting out of hand. The value of MAXSEGS was set at a level higher than the upper limit of the target range for the number of shoots (see Table 3.5), defined separately for Yr90 and Yr145.

Table 3.5. Definition of the computational model bounds

Bound	Description	Value	Theoretical objective value
MAXSEGS	Maximum number of foliated shoots, beyond which the simulation is terminated. Prevents computationally prohibitive branches.	20000; 40000	calculated for the branch at termination of simulation
MINSEGS	Minimum number of foliated shoots, below which the branch is terminated.	5	1000
YMIN	Value along y-axis (branch length; cm), below which the branch is considered outside of the feasible space. A y-value of 0 denotes the main stem of the tree.	-50	1000

The second issue of search stagnation depends on how objective values are calculated for branches that do not survive to the specified year. In this case, both the

numerator and denominator for objectives 5–8 (Table 3.1) are zero. If the objective values are set to zero, then the search would stagnate at dead branches because these would achieve the minimum value for objectives. The alternative is to set the objective values above that which a foliated branch, with the number of foliated shoots below MAXSEGS, would reasonably accomplish. I set the values at 1000 for objectives 5–8 of dead branches, whereas objectives 1–4 are set at the values they achieve when the branch is terminated. This was also the case for any branches that had zero active terminal shoots at the end of the simulation.

Finally, it is possible through the search that the resulting bifurcation values of higher-ordered shoots could result in shoots that bend back toward the trunk, into space that is already occupied. I specify a YMIN value that terminates any simulation that results in a shoot with a y-coordinate that is too far below the plane of the base of the branch. Such branches are also given a value of 1000 for objectives 5–8 (Table 3.5), eliminating them from the non-dominated set. The source code for BRANCHPRO2 is provided in Appendix B.

Discussion

This Chapter outlined the features of the model and objectives that give the structure to the theory represented by the model. Each explanatory component of the growth functions (p and r) represents a particular postulate of the control of branch growth in *P. menziesii*, and the parameter values control the magnitude and direction of the effects. The result of the optimization is bounded by the empirical objectives, which assure that there are optimal parameterizations that result in branches that resemble observed growth patterns. The principle of non-dominance allows for alternative branch patterns that may perform better in the process-based objectives, or for different objective combinations.

At the beginning of this Chapter, I presented a series of questions that bound the context of the model, and the choices of objectives and independent variables were

designed to facilitate answers to the five questions. The main question (How does the branch development pattern of old-growth *P. menziesii* at the WRCCRF compensate for size constraints on the species?) can only be answered by consideration of the subsequent questions A.1-A.3 and B.1. The answer to Question A.1 (What processes are constraining in this system?) depends on the theoretical objectives that are included in the optimization (Table 3.1). This is by no means a complete inventory of the possible constraints on branch development in the old-growth system, but they do provide a plausible set. In my analysis, if there is a clear relationship between growth forms and performance with respect to the theoretical objectives, and if distinct growth forms that reflect patterns observed in the old-growth system perform differently for the theoretical objectives, then it may be inferred that those objectives are potential constraints in the system and should be further evaluated empirically.

Question A.2 (Of the architectures possible in the model structure, which architecture (branch development pattern) best compensates for the constraints?) is related to question A.1 if there is a separation of architectures that perform differently for the objectives (i.e., architecture D performs best for objective 1, architecture E performs best for objective 2), or if a single architecture is found that minimizes the theoretical objectives simultaneously. These two questions (A.1 and A.2) can be answered through optimization results that may or may not replicate the branch pattern observed for *P. menziesii*, such that it is possible that the observed *P. menziesii* architecture does not perform best for any of the theoretical objectives, or combinations thereof. For Figure 3.1, this would be the case if the curve defined by solutions that satisfy Z_3 was shifted up along the Z_2 axis and to the left along Z_1 , with inferior solutions for both Z_1 and Z_2 . In contrast, if the *P. menziesii* architecture performs better in some objectives (or all of the objectives), there is an indication that the architecture does compensate for the constraints that those objectives represent. This evidence should, of course, be interpreted within the limitations of the model structure and assessment objectives. This would provide a guideline to answer question A.3 (For which constraints (if any) does the *P. menziesii* branch development pattern compensate?).

The final question (B.1: What is the relative importance of different branch structural characters (e.g., architectural order, foliage overlap, nearby growth) in determining the observed branch development pattern in *P. menziesii*?) relates to the first three if *P. menziesii* is shown to compensate for the proposed constraints morphologically. The relative values of the parameters (r_0 , r_{ord} , r_{hydr} , r_{over} , p_0 , p_{bud} , p_a , p_{SCU} , p_{stem}) in the results will inform whether and how the proposed independent variables relate to both the corresponding branch morphology and performance of that morphology with respect to the theoretical objectives. This will enable evaluation of the postulates associated with each parameter and variable. This analysis will guide further empirical investigation to focus on relevant features of the branch in the process of morphogenesis.

In order to evaluate these postulates, the adequacy of the model structure must be assessed through a series of optimization searches and post-processing of the results. In Chapter IV, I present the results of optimizations and model modifications that are indicated through the post-processing of the Pareto optimal set.

Chapter IV

Multi-objective assessment of BRANCHPRO2 uncovers model inadequacies in both the process and mathematical structures

Introduction

To best utilize an ecological process model for theory development, its structure must first be assessed against a set of objectives that measure model adequacy. As illustrated in Chapter II, by increasing demands on model structure with just two objectives, model inadequacies were found that would otherwise have been overlooked. In Chapter III, I described the structure of the BRANCHPRO2 process model, and provided theoretical and biological justification for the independent variables of the two growth functions. These represent component postulates for the structure and function of branch development in old-growth *P. menziesii*. In order to use the model to answer the major ecological questions (1a-d), the structure must be assessed. This includes the mathematical formulation of the growth functions, as well as the underlying process rules and descriptions.

The Pareto Optimal Model Assessment Cycle begins with the model structure, objectives and parameters (Appendix A), where

$$M(X) = \text{BRANCHPRO2}, \quad (4.1)$$

$$X = \{r_0, r_{\text{ord}}, r_{\text{hydr}}, r_{\text{over}}, p_0, p_{\text{bud}}, p_a, p_{\text{SCU}}, p_{\text{stem}}\}, \quad (4.2)$$

$$F(M(X)) = (n_{\text{foliated}}, n_{\text{SCU}}, \text{length}, \text{arch_mod}, n_{\text{turns}}, l_{\text{path}}, \text{load}, n_{\text{over}}). \quad (4.3)$$

The parameter ranges the researcher sets for the search algorithm further bound the assessment results, and in the process of determining appropriate allowable ranges for the parameter values, model inadequacies may be found (whether in the model structure, parameters or objective values). It is imperative that in every step of the cycle the results be considered in the context of the connections among the model structure, objectives and parameters.

In the optimal spaces for the optimization series presented in this Chapter, every non-dominated set includes solutions with fewer than the minimum observed foliated shoots, as these more easily minimize the theoretical objectives. For the analyses presented in the remainder of this dissertation (this Chapter and Chapter V), I include only the solutions that achieve at least the minimum number of observed foliated shoots ($n_{foliated} \geq 2500$ for Yr90 optimizations, $n_{foliated} \geq 10000$ for Yr145 optimizations; Table 3.1). This is the space relevant to the questions posed at the outset of this modeling exercise, and it provides a threshold for minimum acceptable growth.

Beginning the assessment cycle: allowable range of parameter values

In order to begin the assessment procedure, the allowable ranges of parameter values must be specified. These ranges define the bounds of the optimization search with respect to the parameters. Although the modeler should use prior knowledge to guide the allowable ranges first set, if the optimization results indicate those ranges to be inadequate, then the modeler should not restrict the search what is expected. As demonstrated in Chapter II, parameter values can emerge outside of what is considered biologically reasonable; this is a clear first indication of model inadequacy and can point to model structures that require improvement.

When the allowable range of parameter values is first set, if the parameters themselves represent established biological quantities (e.g., intrinsic growth rate), then observed values should guide the allowable parameter search ranges used to initiate the search (Chapter II); however, expansion or modification of the allowable ranges may be necessary. In contrast, for BRANCHPRO2, the parameters measure the magnitude and direction of the proposed biological effects. In a sense the parameters are like regression parameters that can theoretically take any value within +/- infinity. If there had been initial empirical observations for the proposed relationships, then those should dictate the allowable parameter values in the search. In the case of the current modeling effort, there is no empirical observation and the search ranges depend on the results of the optimization.

There are several clues in the Pareto optimal parameter distributions that point to inadequacy in the optimization search or in the model itself. First, if in the Pareto optimal set a parameter value converges at the border of its allowable search range, then the allowable range should be increased and the optimization repeated (e.g., Figure 2.3). Second, parameter compensation also implies an inadequacy in the model structure (see Beven 2006 for a discussion of this in hydrological models). Parameters are found to be compensating when the mode of a parameter distribution shifts with no apparent cause. In Chapter II, we saw this when the allowable range of the *newepi* bifurcation parameter was increased; as a result, the mode of order-1 bifurcation shifted dramatically. I explained this pattern through an evaluation of how the parameter values, within the model structure, controlled the achievement of the two objectives. The process rule that controlled the maintenance of foliage on first-generation order-1 axes was found to be deficient relative to the intended effect of the order-1 parameter. This rule diminished the effect of the order-1 parameter on overall branch growth.

The example of parameter compensation in Chapter II (Figure 2.3) illustrated why the sources of evident parameter accommodation should be investigated. It also demonstrated that one of the potential causes of parameter compensation is that a component of the process structure interferes with the intended parameter effect. This may happen when model effects are confounding, when the variables themselves are uninformative, or when the process specification itself is incorrect. The causes of parameter compensations, if observed, must be evaluated in the process of model assessment and improvement.

In this Chapter, I present several series of optimizations conducted to assess the BRANCHPRO2 model structure that is described in Chapter III. For the assessment presented in this Chapter, optimizations are first conducted to determine appropriate allowable parameter ranges. These lead to a series of modifications to the growth functions and process rules. I finally settle on a model structure that will be further

evaluated in Chapter V. The stage of the modeling process presented in this Chapter is an assessment that attempts to find an adequate model structure and complexity that is bounded by the optimization objectives, relative to the parameter distributions of the non-dominated Pareto set.

The assessment I present in this Chapter is divided into several optimization series, each of which includes multiple optimization searches. For each optimization in the series, the pattern of parameter convergence is evaluated. If this pattern indicates an issue in the model structure, then simulations are utilized to uncover model inadequacies. Model improvements are then made and a new optimization series is conducted. In order to simplify the initial searches, the optimizations for Series 1–4 are conducted for branches simulated to year 90 (Yr90). When an adequate model and parameter convergence is found for Yr90 branches, a new series of Yr145 branches is conducted (Series 5). Every search presented in this Chapter is numbered sequentially from the first to the last optimization search (4.1, 4.2, 4.3, ..., 4.33). These searches are divided into five distinct series, so for example optimization Series 1 comprises optimization searches 4.1–4.3.

Assessment optimization searches

Series 1 (Searches 4.1–4.3)

Parameter convergence

Parameter ranges are explored by choosing allowable values for each of the parameters, conducting the optimization for a Yr90 branch and then observing the distribution of parameter values in the non-dominated Pareto set. I begin with general guidelines for the parameter search ranges that reflected the expected relationship under the biological postulates: positive values for each of the parameters associated with p and negative values for each of the parameters associated with r (with the exception of r_0 ; Chapter III).

In the distribution of parameter values for search 4.1, r_0 converges at the upper boundary of its allowable range, whereas r_{ord} and r_{hydr} converge strongly at their lower boundaries (Figure 4.1). The parameters p_0 and p_{bud} also converge at their upper boundaries. The allowable parameter ranges were modified, with an increase in the upper range for r_0 and p_0 , and a decrease in the lower range for r_{ord} and r_{hydr} ; the optimization was repeated with the new search ranges. When the pattern of convergence was similar for the second optimization, the search ranges were modified again and the optimization repeated (see Table 4.1 for allowable ranges of r function parameters). The distributions of parameter values are compared among these three searches.

Table 4.1. Allowable parameter ranges for parameters of the r function across the three searches of optimization Series 1.

	r_0	r_{ord}	r_{hydr}	r_{over}
Search 4.1	(0,4)	(-4,0)	(-15,-5)	(-4,0)
Search 4.2	(0,10)	(-10,0)	(-20,0)	(-4,0)
Search 4.3	(0,10)	(-20,0)	(-20,0)	(-4,4)

In this series of three optimization searches, the distributions of the values for some parameters shift when the allowable ranges for others are modified (Figure 4.1). For example, between the first and second search the allowable ranges of r_0 , r_{ord} and r_{hydr} were all expanded (Table 4.1). The distributions of r_{ord} and r_{hydr} shift left with their new allowable ranges. At the same time, the distribution of r_{over} shifts right without any modification of its allowable range (Figure 4.1). In the next optimization search, r_{ord} and r_{hydr} both shift back to the right, whereas r_{over} shifts to the left border of its allowable range.

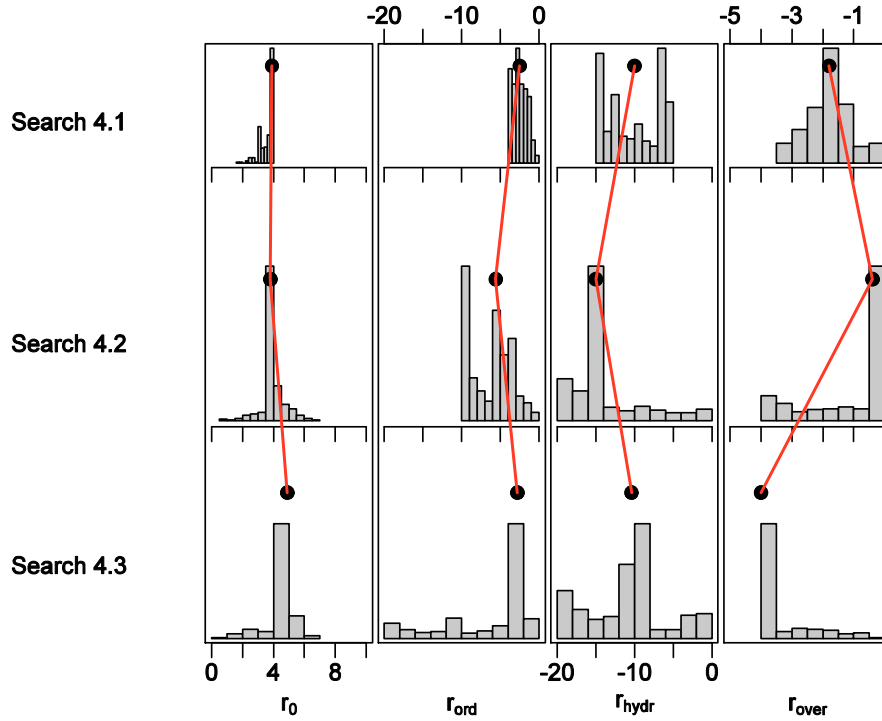


Figure 4.1 Histograms of the parameters for the r function through the three searches in optimization Series 1. The top row is for search 4.1, the middle row for search 4.2 and the bottom row for search 4.3 (see Table 4.1 for allowable search ranges for each). The y-axis is the frequency (not shown). The symbol (•) indicates the median value for each parameter and search, and the red lines connect the median values between searches. Between the second and third searches the distributions of both r_{ord} and r_{hydr} shift from the left of the allowable range to the right. In contrast, the distribution of r_{over} shifts from the right of its allowable range to the left. These distributions shift with no modification to the model structure or objectives, only with changes in the allowable ranges such that as the values of r_{ord} and r_{hydr} converged in one direction, the value of r_{over} converged in the other. This relationship is due to inadequacy in the mathematical structure of the model.

Given that only the allowable ranges change between each search, these distribution shifts indicate that there is a potential dependency of the parameters and their associated variables that accommodates the model structure and optimization objectives (e.g., when one variable is higher, another is lower in order to still meet the objectives). The explanation for why the accommodations occur requires a close examination of the parameters whose distributions shift. It should be clearly understood whether these relationships can be explained within reasonable bounds of the model structure and underlying theory (i.e., in how the parameters relate the variables to the model outputs), or if they represent a deficiency in the model. I first look for signs of model deficiency, as any inference for the relationship between the parameter patterns and the model

outputs should be reserved for a model deemed adequate. This requires a deeper survey of the parameters that seem to be accommodating. In particular the parameter definitions, their associated variables and associated process definitions should be investigated.

I consider the r function first, and the parameters that clearly compensate are r_{ord} , r_{hydr} and r_{over} (Figure 4.1). The parameter r_{ord} decreases the bifurcation of lateral axes relative to lower ordered axes, such that bifurcation decreases with increasing order. The parameter r_{hydr} decreases the bifurcation at the base of axes formed by any junction, whether through lateral sequential growth or from the release of suppressed buds (new epicormic shoots). The parameter r_{over} decreases the bifurcation of any active node with increasing foliage overlap. The effects of each of these parameters clearly coincide in that both r_{ord} and r_{hydr} reduce bifurcation for new lateral axes, and r_{over} reduces bifurcation for all active axes. The direct confounding of r_{ord} and r_{hydr} could explain why they have a similar pattern of distribution across the three searches (Figure 4.1), in that they are causing a similar effect. I address that possibility first, and then I evaluate the possible model deficiency with respect to r_{over} .

Confounding variables

The parameter r_{ord} is associated with the effect of order on bifurcation, and r_{hydr} is associated both with the effect of increasing number of cumulative junctions (n_{turns}) and the effect of increased time of suppression of the bud that originated the junction (t_{base}). The variables order and n_{turns} are directly confounding because n_{turns} is a linear combination of order (o) and generation (g):

$$n_{turns} = o + g - 2. \quad (4.4)$$

This relationship makes it impossible to separate the proposed effects of order in the dominant architectural model and the effects of n_{turns} in the hydraulic function, and because of this the two parameters can possibly continue to accommodate each other in the model structure. The effects of order and generation should be separated in the model

structure, which requires further consideration of the biological intention of n_{turns} as a hydraulic constriction.

Although I included the variable n_{turns} in the hydraulic function because of the possible hydraulic constriction at branching junctions, Schulte and Brooks (2003) found no evidence for hydraulic constriction in *P. menziesii* junctions in younger trees (~24 years old). Constrictions are more often measured at unequal junctions, where there is a distinct disparity in diameter between the segments that comprise the junction (Tyree and Ewers 1991). In red maple, the ratio of conductivity between the branch and the trunk decreases with decreasing diameter ratio (branch diameter:trunk diameter; Eisner et al. 2002). The difference in diameter at is greater for epicormic than for sequential junctions because of the timing of the junction formation. For a sequential junction, the ages of the shoots that comprise the junction are equal. For a bud released from suppression, the new shoot would be very young compared to the parent segment; the longer the bud is suppressed, the greater the disparity in diameter that would result. This is distinct from the hormonal effect of increasing order, which may not be associated with a hydraulic constriction at the resulting sequential junction (Schulte and Brooks 2003).

If we consider the hydraulic constriction to be due solely to junctions caused by bud release from suppression, then the effect of n_{turns} should be restricted to increasing generation. To improve the model, the value of n_{turns} will thereby be replaced with generation (g) and in the model the hydraulic constriction will be solely due to proleptic reiteration. Consequently, the hydraulic r function will only be applied to order-1 axes, t_{base} will continue to be the value of the age of the bud that gave rise to the base of the current SCU, and d_{base} will be the distance along the order-1 axis from the base of the current SCU to the current bud. This function is designed to explain the pattern of initial growth evident in newly released epicormic buds of *P. menziesii* (Kennedy et al. 2004, their Figure 2c),

$$\frac{r_{hydr}}{d_{base}} \left(1 - \frac{1}{g + t_{base}} \right). \quad (4.5)$$

Scale of saturation of independent variables

The third parameter in optimization Series 1 that shows compensation, and shows it most dramatically, is r_{over} (Figure 4.1). This indicates that it may not measure any relevant effect of foliage overlap (under the supposition that, if the parameter and its associated variable represented meaningful quantities in the model structure, then there would be a consistent estimated value within a particular allowable range). This variable appears to shift in opposite to r_{ord} and r_{hydr} , indicating that when these effects are smaller, the effect of r_{over} is greater. The parameter r_{over} is intended to regulate the effect of foliage overlap (n_{over}) on bifurcation. One of the major assumptions in the model structure is that the effect of overlap increases as a simple saturating function (eqn 3.19). The intention of the saturating function is for the effect to increase as the independent variable increases, but at some point any increase in the independent variable does not change its effect on the response, i.e., the effect saturates above a certain level. The chosen saturation function, however, saturates very quickly at low integer values of every variable (Figure 3.3a). This is true regardless of the scale at which the variable may have an effect on the branch, so that variables that have a strong effect at a value of 10 would show no additional effect relative to a value of 3. The function thereby takes on similar values for two shoots whose conditions clearly differ. For example, in the simple saturating function, a shoot with an n_{over} of 5 vs. a shoot with an n_{over} of 15 have similar effects on the overall value of r ; however, the shading on the shoot with an n_{over} of 15 would be greater and therefore should result in a smaller r relative to a shoot with an n_{over} of 5. This issue may mask the effect of the independent variable. In this case, the associated parameter (r_{over}) does not estimate the effect of the independent variable; rather, the parameter value can vary to accommodate some other aspect of the model structure. This manifests as a shifting of its distribution relative to the values of r_{ord} and r_{hydr} . We should consider at what values we might expect the effect to saturate.

I simulated a branch in the current model structure and observed the condition of shoots with varying values of n_{over} . This is a rough approximation of overlap, and does not provide a quantity for the amount of foliage that is actually shaded. Upon visual inspection, shoots with overlap of 10 still have foliage visible through the overlap, whereas shoots with overlap of 20 have no visible foliage (Figure 4.2). Given that the overlap is not actually complete (light is not completely blocked by the foliage, as assumed with the foliage rectangles in the model), the value of overlap at which the shoot lacks illumination is likely greater than 20, yet the effect would clearly saturate at some point above 20 as the shoot is completely covered by overlapping foliage. The current saturating function, however, saturates near a value of three (Figure 3.3a). The current saturating function is an inadequate mathematical structure for the increasing effect of overlap. An alternative function that saturates more slowly and at a higher value should be used instead.

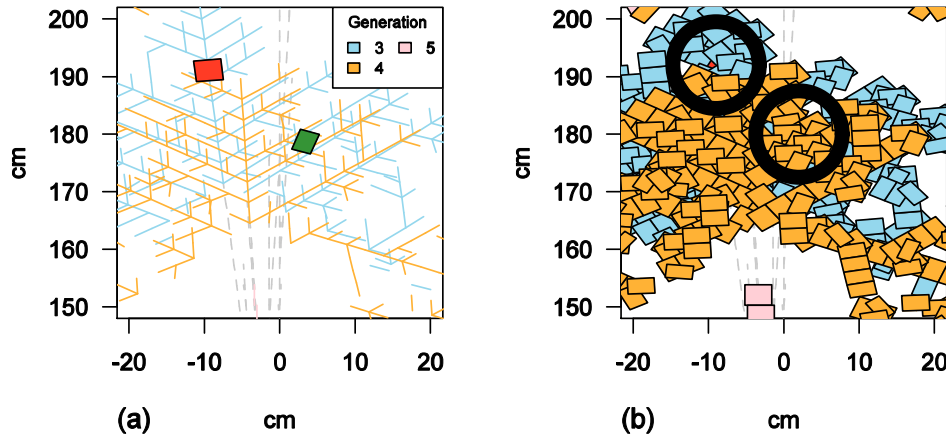


Figure 4.2: (a) Placement of foliage rectangles for two shoots with values of 10 (red) and 20 (green) for n_{over} . Overlap is defined by the number of foliage rectangles of equal or higher generation that intersect the current foliage rectangle (Figure 3.2b). (b) When the remaining foliage rectangles are filled in, there is still some of the red shoot visible (the black circles indicate placement of each shoot), i.e., the red shoot with $n_{over}=10$ is not completely covered by other foliage rectangles. In contrast, the green shoot with $n_{over}=20$ is no longer visible, i.e., it is completely covered by other foliage rectangles. Although this is not a direct indication of complete shading of the green shoot, it does show that as the value of n_{over} increases, the overlap becomes more complete. I assume that, as the number of foliage rectangles from other shoots that cover the current shoot accumulates, the less impact each additional shoot has on the light interception of the current shoot.

Consequently, I replace the function for n_{over} with one that saturates at a scale that better matches the observed scale of n_{over} :

$$y = k_1 \left(\frac{x - k_3}{k_2 + x - k_3} \right), \quad (\text{Haefner 1996, p. 92}) \quad (4.6)$$

where k_1 controls the saturated value of the function and will be the associated parameter value (e.g., r_{over}) allowed to freely vary in the optimization. The value k_3 controls where the function switches from negative to positive and will be set to zero; k_2 is the half-saturation constant that I will set for n_{over} . I choose this function because it provides desired characteristics of the function curve, with a parameter that can control the rate of saturation separately for each variable. Since we lack empirical data for each of these, I choose values for the half-saturation constants. Further empirical investigation should be conducted to better define the curves for observed effects of the independent variables (see Chapter VI).

The value of the half-saturation constant defines the shape of the curve that relates the independent variable to the response; the purpose of the chosen curve is to have a relationship that increases almost linearly over the effective area of the independent variable, then the effect saturates when it has reached its maximum level at larger values of the independent variable. For example, one would expect the growth to decrease as the number of overlapping foliated shoots increases, but at some point the addition of another overlapping shoot would have no further effect on the shoot of interest, as it is already completely shaded. This assumes an isolated branch that is lit directly from above. For the value of the half-saturation constant for n_{over} , I consider an isolated SCU with the observed architectural model (Figure 3.1b) as a guideline. If the order-2 axis does not have lateral growth, then the order-1 shoot would have an overlap value of four (see the order-1 shoot that is second from the base of the structure, Figure 3.1b). I set the half-saturation value at twice that, or 8. The curve defined by the function with a half-

saturation of 8 increases almost linearly until an n_{over} of 20, then the curve slows and levels above n_{over} of 32 (Figure 4.2).

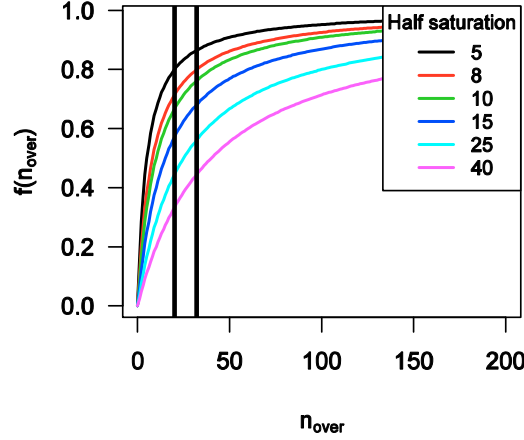


Figure 4.3: Function saturation for varying half-saturation values for n_{over} . Vertical lines are at $n_{over}=20$ and $n_{over}=32$. Large half-saturation values result in the function increasing slowly over values of n_{over} well above that which would reasonably be expected to affect bifurcation. At the half-saturation value of 8 (red curve), the function slows down near $n_{over}=20$, and levels off at values above 32.

Quantitatively, we can calculate the value of the independent variable at 80%, 90% and 95% of the maximum function value (Table 4.2). If q is the decimal of the percent of maximum (e.g., 0.8) and k_2 is the half-saturation constant, then the value of the independent variable at q of maximum is calculated as:

$$\frac{q}{1-q} k_2. \quad (4.7)$$

Table 4.2: Half-saturation constants (HalfSat) for the independent variables, and the value of each at 80, 90 and 95% of the function maximum.

Variable	HalfSat	80%	90%	95%
g	4	16	36	76
n_{over}	8	32	72	152
t_{base}	5	20	45	95
t_{bud}	10	40	90	190
d_{SCU}	15	60	135	285
d_{stem}	100	400	900	1900

For a half-saturation of 8, the function reaches 80% of maximum at n_{over} of 32 (Figure 4.3). This is an adequate representation of the scale of saturation given the

observation that a shoot with an n_{over} of 20 appears to be fully covered by neighboring foliage (Figure 4.2), and the function reaches 80% of maximum at a value well above 20. Therefore the effective range of n_{over} is covered by this function.

The preceding analysis has uncovered a flaw in the original mathematical structure of the saturating function for n_{over} , which explains how the associated parameter value is able to shift its distribution relative to the other parameters in the r function. The flaw is not limited to the function associated with n_{over} . The simple saturating function was applied to most of the independent variables for both functions, yet many of the variables are observed with values well above the scale of saturation in the simple saturating function, particularly g , t_{base} , t_{bud} , d_{SCU} , and d_{stem} . The parameter associated with each of these also shows some indication of parameter compensation in the optimization Series 1 (Figure 4.1). For each of these variables the half-saturation function (eqn 4.6) should be applied, and I will consider each in turn for the value of the associated half-saturation constant.

Half-saturation constants

In this exploratory model, there is no empirical basis for the half-saturation constants, so I make reasonable estimates based on the scale of the independent variable in the observed (or simulated) system. The parameter r_{hydr} is associated with two independent variables, generation (g) and t_{base} . The value of t_{base} is the length of time the bud at the base of the new SCU had been suppressed when it was released to form a new epicormic shoot. The timing of epicormic initiation for epicormic shoots that result in new SCUs has been observed (Ishii and Ford 2001; Kennedy et al. 2004), and the expected value for that timing is 5 years. If the half-saturation is set at five, the function reaches 80% of maximum at 20 years (Table 4.2). This is within the maximum range of timing of epicormic initiation for independent SCUs. The value of generation increases with each cumulative new epicormic axis. The maximum generation observed by Ishii and Ford (2001) is seven. If the half-saturation value is set at half that (rounded up to four), then the function reaches 80% of maximum at generation 16. The half-saturation

value for the hydraulic function is then the sum of the half-saturation constants for t_{base} and generation, 9.

I consider the process of bud release from suppression separately from bifurcation. It is possible for buds to be suppressed longer than the observed timing that results in new SCUs. To that end, I set the half-saturation constant for t_{bud} at twice the expected value of the timing of epicormic initiation (10 years). The function reaches 80% of maximum at 40 years for a half-saturation value of 10 years. The variable d_{SCU} counts the number of segments from the current suppressed bud to the nearest proliferating epicormic structure on the same order-1 axis. An SCU is not an independent structure until it is at least as old as the maximum observed foliage longevity, 10 years. The SCU often survives through 20 years of growth, and in the case of basal reiteration up to 60 years. I set the half-saturation value at 15 segments (which corresponds to 15 years of active growth if the SCU is still proliferating), which has a value of 60 at 80% of the maximum functional value. This enables the function to saturate at values near the maximum observed SCU longevity.

The final variable that occurs at a scale well above the original saturating function is d_{stem} , which is the distance (path length) from the suppressed bud to the base of the branch. The maximum observed branch length for a Yr90 branch (hence the maximum observed d_{stem}) is 430 cm. If the half-saturation constant is set at 100 cm, then the function reaches 80% of maximum near the maximum observed length, 400 cm (Table 4.2).

New model equations

The incorporation of half-saturation constants in the model growth functions yields two new functions:

$$p(t_{bud}, a, d_{SCU}, d_{stem}) = p_0 + p_{bud} \left(\frac{t_{bud}}{10 + t_{bud}} \right) + p_a * a + p_{SCU} \left(\frac{d_{SCU}}{15 + d_{SCU}} \right) + p_{stem} \left(1 - \frac{d_{stem}}{100 + d_{stem}} \right), \quad (4.8)$$

$$r(o, g, n_{over}, t_{base}, d_{base}) = r_0 + r_{ord} \left(1 - \frac{1}{o} \right) + \frac{r_{hydr}}{d_{base}} \left(\frac{g + t_{base}}{9 + g + t_{base}} \right) + r_{over} \left(\frac{n_{over}}{8 + n_{over}} \right). \quad (4.9)$$

The next series of optimization searches are conducted for the model as defined by equations 4.8 and 4.9.

Series 2 (Optimizations 4.4–4.5)

In the second optimization series (Series 2; equations 4.8 and 4.9) there are two optimization searches because a prominent, unexplained pattern in the parameter distributions emerges after only two consecutive searches. The source of the pattern of parameter convergence should be explained before the next optimization series.

Parameter convergence

In the parameter distributions for Series 2, the glaring pattern is the tendency for r_{hydr} to converge at the extreme lower boundary of its search range (Figure 4.4), a pattern that persists through the two optimization searches.

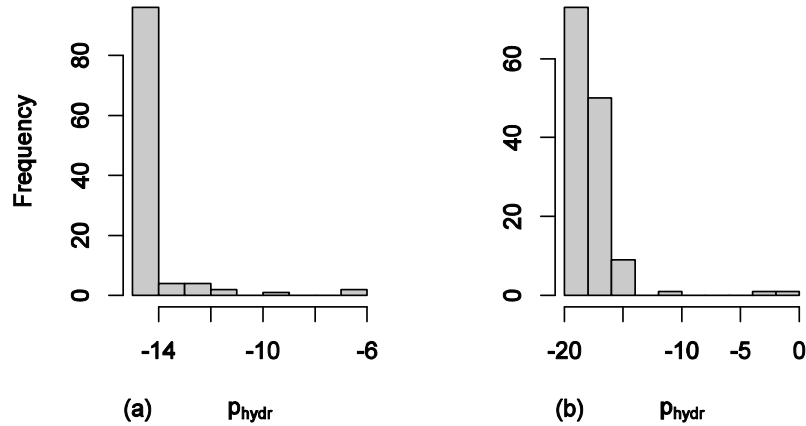


Figure 4.4: Parameter distributions for p_{hydr} in the two searches in optimization Series 2, demonstrating the negative convergence of the hydraulic bifurcation parameter as the left boundary is reduced from (a) -15 to (b) -20.

When investigating the source of that pattern, it is impossible to distinguish the effects of g and t_{base} in the hydraulic function. The original motivation for aggregating

the variables in a single function was to simplify the parameter space by including the smallest number of parameters, with a single parameter for the hydraulic effect. This was at the expense of mathematical complexity in the sense that the hydraulic function contains many variables in a single mathematical expression. In order to discern the effects of each variable, generation and t_{base} should be separated and given individual parameters. The half-saturation constants for each are the components of the total half-saturation constant for the hydraulic function (4 for g and 5 for t_{base} : Table 4.2). This increases the total parameter space to 10 (remove r_{hydr} , add r_{gen} and r_{base} ; Table 4.3), and the model defined by this equation,

$$r(o, g, n_{over}, t_{base}, d_{base}) = r_0 + r_{ord} \left(1 - \frac{1}{o} \right) + \frac{r_{gen}}{d_{base}} \left(\frac{g}{4 + g} \right) + r_{over} \left(\frac{n_{over}}{8 + n_{over}} \right) + \frac{r_{base}}{d_{base}} \left(\frac{t_{base}}{5 + t_{base}} \right), \quad (4.10)$$

and equation 4.9 will be called BRANCHPRO3. At this stage the optimization should be repeated.

Table 4.3: Parameter definitions and associated independent variables for BRANCHPRO3

Parameter	Independent Variable	Description
p_0	NA	Base probability of bud release
p_{bud}	t_{bud}	age of bud
p_a	a	proportion of inactive axes
p_{SCU}	d_{SCU}	distance to the nearest developing SCU on the same axis
p_{stem}	d_{stem}	distance to the main stem
r_0	NA	Base rate of bifurcation
r_{ord}	o	shoot order
r_{gen}	g	shoot generation
r_{over}	n_{over}	number of overlapping foliated
r_{base}	t_{base}	age of bud at SCU base when it was released from suppression

Series 3 (Optimization 4.6)

Parameter convergence

In search 4.6, the value of r_{gen} clearly continues to converge at the lower boundary of its search range. This indicates that generation is the variable that contributes most to the pattern of convergence for r_{hydr} that was observed in Series 2. The value of r_{gen} determines how much bifurcation is reduced for each consecutive new epicormic axis. In order to understand why it might converge to the left, simulations should be conducted with a focus on bud release and early epicormic growth. Branches simulated from the Series 3 Pareto optimal set exhibit bud release from suppression at very high generations, well above that which was observed for *P. menziesii*. SCU development, however, occurred at generations well within the maximum observed value. This indicates a disconnect between the process of bifurcation for a newly initiated epicormic shoot as defined by the r function, and the process of bud release from suppression as defined by the p function. The value of r_{gen} in the bifurcation function determines a possible threshold for the generation at which the r function crosses zero (g^*), or even whether such a threshold exists. When the bifurcation function reaches a value of zero, then the associated shoot will not proliferate. If it is assumed the remaining independent variables are zero, then:

$$r = 0 = r_0 + \frac{r_{gen}}{d_{base}} \left(\frac{g}{4 + g} \right), \quad (4.11)$$

$$g^* = -\frac{4r_0 d_{base}}{r_{gen} + r_0 d_{base}}. \quad (4.12)$$

For a newly initiated epicormic shoot, d_{base} is 3.58 cm, and for these shoots the value of g^* with $r_{gen}=(-20)$ and $r_0=5$ is 10.08. This means that no new epicormic with a generation higher than 11 will bifurcate, regardless of the remaining local conditions. This threshold could possibly be lower depending on the values for t_{base} and n_{over} , so this is the maximum generation an SCU can form on the branch. On the simulated branches,

however, new epicormic shoots are released at generations much higher than the threshold for new SCU development set by the bifurcation function. There is a problem in the model structure for the bud release from suppression process specification that allows suppressed buds to be released at generations much higher than the maximum generation for independent SCUs on the branch. This requires a close examination of the processes involved in bud release from suppression.

Process specification: bud release from suppression

In the probability structure of the p function, every order-1 node has a positive probability of initiating a new epicormic shoot. Since new epicormic shoots are defined as order 1, they also have a positive probability of bud release from suppression. This results in buds released on newly initiated epicormics, even if the parent epicormic did not successfully develop into a new SCU. The implied assumption is that a suppressed bud can be released on any epicormic axis, regardless of whether it is a fully formed SCU structure. This is inconsistent with the theory structure of the model, in which reiteration and epicormic growth are observed to occur along the main order-1 axis of independent SCUs. To be consistent with the observed pattern, if a shoot that results from bud release from suppression fails to proliferate the next year, then that shoot should be shed from the branch when its foliage is lost. Any buds suppressed on this shoot should not be available for release from suppression. To better align the process of bud release from suppression with the current theory of reiteration, I add a new rule to the model that requires a suppressed bud to be located on the order-1 axis of an independent SCU in order to be released from suppression.

A further consequence of the model structure for p is that ALL order 1 nodes on the order-1 axis of independent SCUs have a positive probability of a bud releasing from suppression. Theoretically, then, they will all eventually release a bud. There is no negative feedback to define when a bud might no longer be available for release. It seems that either the relationship with one of the current variables is incorrectly specified, or another variable is missing that would give the negative feedback.

Among the variables currently included in the p function, it seems that the positive effect of t_{bud} may have the weakest justification. The positive relationship between p and t_{bud} is based on the empirical observation that, to a point, the proportion of epicormics released increases as the age of the bud increases. For the process specification I speculate that this could be due to increasing distance to the proliferating axis, but this is really a poor surrogate for that process. Furthermore, the observed proportion of buds released from suppression actually declines beyond five years. In contrast to the positive effect of t_{bud} on the value of p , t_{base} is proposed to have a negative effect on proliferation in the r function. The explanation is that the longer the bud is suppressed, the greater the hydraulic constriction. Further growth requires a few years of development once the bud is released in order to form better hydraulic connections. To be consistent, one would think that such weaker hydraulic connections as the bud is longer suppressed would also have a negative effect on the probability of release. In the optimizations thus far, p_{bud} tends to converge near zero (for example, 180/296 solutions have $p_{bud}=0$, all other parameters in the p function have the first quartile above zero), which further supports changing t_{bud} to negatively affect p . A negative effect would possibly result in a threshold for the length of time a bud could survive in its suppressed state.

This change in the process of bud release from suppression would reflect a possible give-and-take between the suppression of buds by active foliage, and the potential requirement of suppressed buds for some source of carbohydrate in order to be released. The longer a bud is suppressed, the less likely it is that there is active foliage in the immediate vicinity of the bud. This would explain why the effect of t_{bud} should be negative. To enact this change, I extend the search range for p_{bud} is extended to negative values. The optimization is now repeated for BRANCHPRO3 with the new rule for release from suppression and with negative p_{bud} search ranges.

Series 4: BRANCHPRO3 (Optimizations 4.7–4.22)

The optimization search is repeated for BRANCHPRO3, with the modifications included from Series 2 and Series 3. The dominant pattern in Series 4 continues to be the leftward convergence of r_{gen} , with subsequent positive convergence of r_0 (Figure 4.5); there were corresponding shifts in the other r parameters. The leftward convergence tapers when the minimum r_{gen} is set to (-1000), which was achieved after 16 optimization searches.

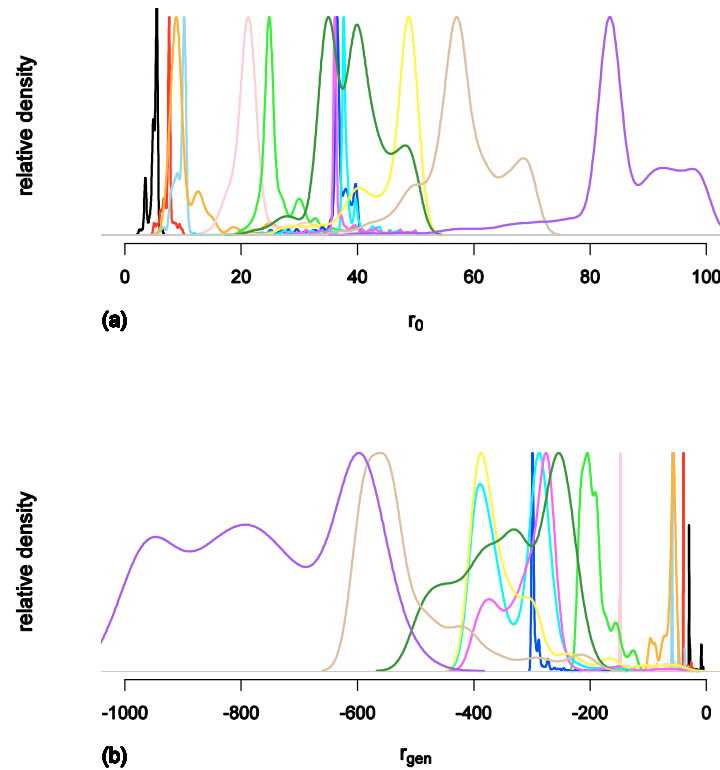


Figure 4.5. Density plots of parameter distributions for (a) r_0 and (b) r_{gen} through optimization searches in Series 4. Each colored line represents the distribution for an individual search (e.g., black is the distribution of r_0 and r_{gen} in search 4.7). For each subsequent search the distribution of r_0 shifts to the right (a), and the corresponding distribution of r_{gen} shifts to the left (b). The distribution of the values in the final search is defined by the purple line. The values on the y-axis vary with each curve and are not shown on the plot. The values would be such that the area under each curve is 1; if the densities were drawn on the same scales the peaks of distributions of the later searches would be much lower compared to the earlier searches.

In this series of optimizations, there are no obvious accommodations among the parameters (e.g., shifting direction of the mode of any parameter). Between searches, the

parameter distributions shift in a consistent manner that relates to how they are integrated in the bifurcation function. For example, if all of the other parameters are ignored, as r_{gen} becomes increasingly negative r_0 becomes increasingly positive (Figure 4.5) such that combinations of r_0 and r_{gen} yield some expected value of bifurcation. This is in contrast to the parameter accommodations observed in Series 1. In that case the modes of the distributions for each parameter shifted depending on the other parameters (Figure 4.1). This change in distribution was not related to the effect of the associated variable, whereas in this case the shifting of the distribution is in direct relation to the effects of the independent variables and how they integrate in the growth function.

Given the final parameter distributions of optimization Series 4, acceptable convergence of the parameter values is obtained and there are no glaring inconsistencies in the process specifications and emergent example branch simulations. I consider the model to be adequate to enter a more detailed analysis to answer the questions posed in Chapter III. Branches simulated from the series of optimizations are visually adequate and they have epicormic growth within the observed number of generations, although they do demonstrate different growth forms through Series 4 (Figure 4.6). We will see in Chapter V that the main factor that differentiates the branch forms at either end of the spectrum is whether they tend to minimize l_{path} or n_{turns} .

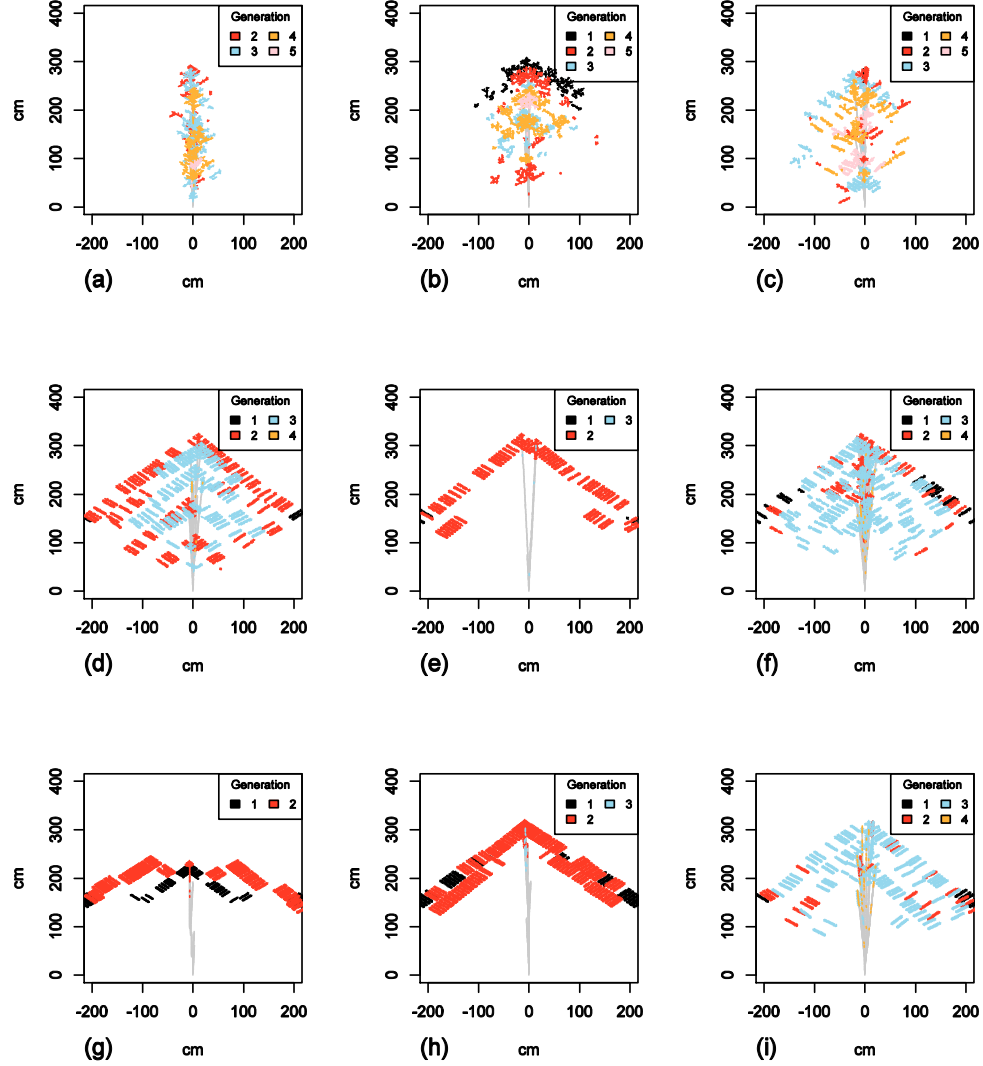


Figure 4.6: Maps of live foliage (in color) for branches simulated from optimization searches in Series 4, showing differing branch forms through the optimization searches. Search progression from (a) to (i) corresponds to parameter distributions from left to right in Figure 4.5a. For the sake of space, not all searches are represented. The light grey lines are for order-1 axes that have lost their foliage, and foliage is differentiated by color for each generation. Through the progression of searches the branch forms clearly differ, which is a consequence of the increasing values of r_0 (which would result in longer lived branch axes) and the decreasing values of r_{gen} (which would result in more severe constriction of new epicormic growth with increasing generation; Figure 4.5) on the process of morphogenesis. Qualitatively, we see that branches for 4.6a-c tend to have many more SCUs of higher generation, with shorter axes. In contrast, branches 4.6g-h have fewer SCUs of lower generation, and individual axes are much longer than 4.6a-c. The implications for the striking differences in growth form are evaluated in detail in Chapter V, where we will see that branches characterized by the growth form in (a) tend to perform better with respect to l_{path} , and branches characterized by the growth form in (i) tend to perform better with respect to n_{turns} .

In this series of optimizations, each search presents a unique partition of the possible parameter space (Figure 4.5), where later searches exclude possible solutions

found in earlier searches. To form a full approximation of the non-dominated frontier, I combine the solutions from the individual optimization searches in this series and re-evaluate dominance. The resulting non-dominated set can be considered the fully approximated Pareto set for BRANCHPRO3 (ParetoAll; Figure 4.7).

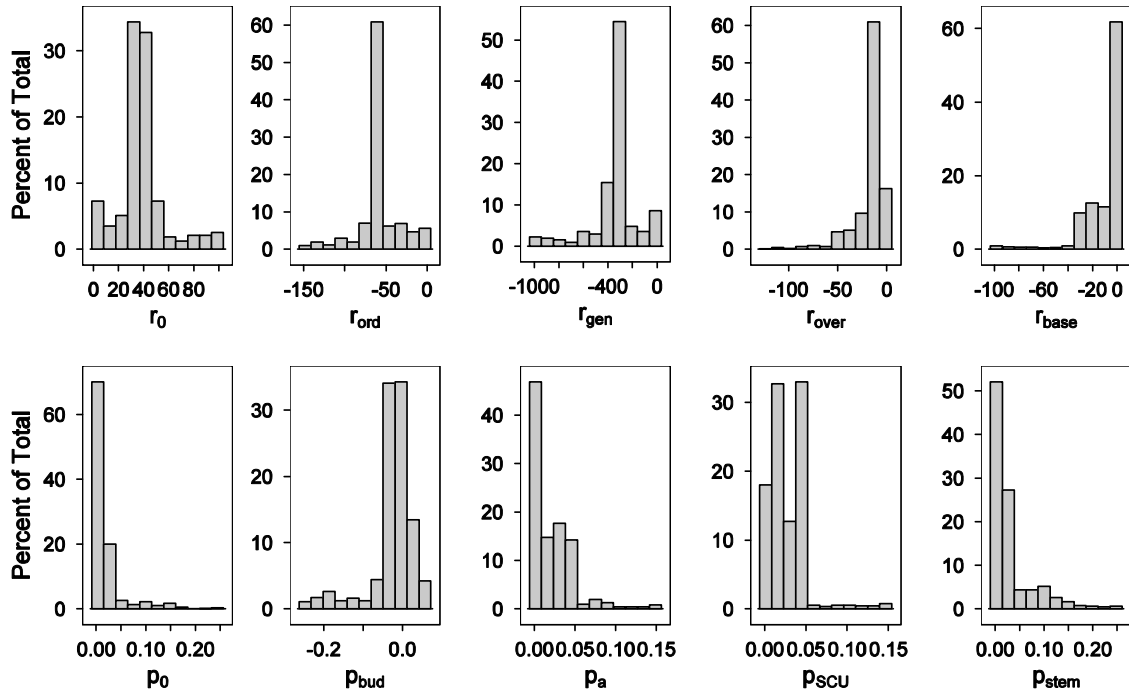


Figure 4.7: Final distribution of parameter values for Yr90 optimizations Series 4 (ParetoAll). This is the non-dominated Pareto set that results when the sets from every optimization search in Series 4 are combined and dominance re-evaluated. This shows that there is adequate convergence for each of the parameters in the full non-dominated Pareto optimal set. See Table 4.3 for description of the parameters.

There are 921 solutions in the Yr90 ParetoAll set, and of those 34.5% (318 solutions) satisfy the four empirical objectives simultaneously. This implies that, while the model does successfully satisfy the observed growth pattern as quantified in the empirical objectives, there are other growth forms that perform better with respect to varying combinations of the theoretical objectives (Figure 4.6). Of the 603 solutions that fail at least one of the empirical objectives, 581 fail the SCUs requirement (539 have too few SCUs). This is clearly a demarcation in the level of reiteration on the branch, and only comparison of relative performance with respect to the theoretical objectives can explain why different growth forms emerge. These patterns will be evaluated in detail in

Chapter V. The analysis to this point has been conducted for Yr90 branches. I repeat the optimization for BRANCHPRO3 simulated for Yr145 branches.

Series 5: Yr145 optimizations (Optimizations 4.23–4.33)

I repeat the Series 4 optimization for branches simulated for 145 years, with the target ranges for the first three empirical objectives modified accordingly (Table 3.1). The primary pattern of strongly negative convergence of the r_{gen} parameter that was observed to drive convergence for Yr90 branches also drives convergence in Yr145 branches. The final lower range for r_{gen} for Yr145 branches that shows acceptable convergence in parameter values is lower than for Yr90 branches (-1348 for Yr145, -1000 for Yr90; Figure 4.8).

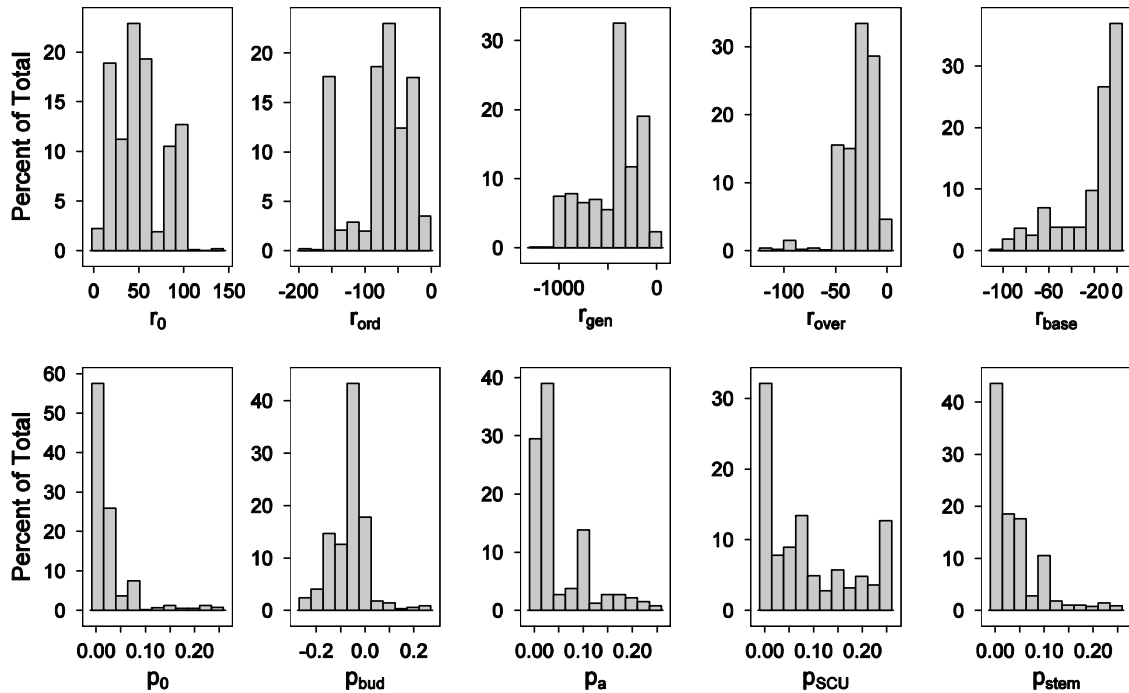


Figure 4.8: Final distribution of parameter values for Yr145 optimizations (ParetoAll.145). This is the non-dominated Pareto set that results when the sets from every optimization search in Series 5 are combined and dominance re-evaluated. This shows that there is adequate convergence for each of the parameters in the full non-dominated Pareto optimal set. See Table 4.3 for description of the parameters.

The branch forms from simulations in the emergent series of optimization searches show patterns similar to the Yr90 Series 3 optimizations, with distinct

differences in visual branch morphology through branches simulated in each optimization search in the series. Of the 999 solutions in the non-dominated set evaluated across the searches in optimization Series 5 (ParetoAll.145), 22% (217) satisfy shoots and SCUs simultaneously. The remaining 78% have too few SCUs, which is similar to the pattern observed for the Yr90 optimization Series 4. In Chapter V, these patterns will be evaluated in more detail.

Discussion

These results further demonstrate that multi-objective optimization is a powerful assessment tool for process-based models of plant development. As recommended in Chapter II, two additional empirical objectives were included in the optimization in order to more fully capture the observed branch pattern. The component parameter values in this model represent the relative magnitudes of the effects of independent variables, and the pattern in their distributions define the eventual branching pattern. By increasing the demands on model structure and requiring the model to satisfy multiple objectives simultaneously, model deficiencies were illuminated that otherwise may be overlooked.

A necessary component of the process model assessment procedure is the evaluation of simulations from example parameterizations in the Pareto optimal set. Although the empirical objectives are designed such that when all are satisfied the model represents an adequate structure, unexpected patterns may emerge that are valuable in uncovering inadequacies both in model structure and in objective formulation. In this case, as is common in plant modeling, a visual assessment of the emergent branch pattern is a useful check on the adequacy of the model to reproduce the observed growth form. In contrast to other plant model exercises, however, I utilize multiple empirical objectives in a formal assessment procedure and the visual assessment serves the purpose of an ancillary check on model adequacy, rather than the basis for the entire assessment. The empirical objectives serve the purpose to narrow all possible parameter combinations to those that satisfy, at some level, the observed growth pattern. Emergent anomalous patterns often come from unexpected sources that are not captured in the empirical

optimization objectives. These can be observed through post-processing simulations of solutions in the Pareto optimal set, which aid to uncover the black box of intermediate and hidden assumptions that are prevalent in process models. In the current example, there was a hidden assumption that all order-1 shoots can release suppressed buds over all time periods. This was replaced with the rule that only order-1 axes on independent SCUs can release buds from suppression.

Role of parameter distributions in assessment process

In the models BRANCHPRO2 and BRANCHPRO3, the parameters are like regression parameters in a statistical analysis. They can take any possible value, and do not have a direct biological interpretation, although they do influence the interpretation of the associated biological variables. As such, the parameter values themselves cannot necessarily be compared against biological observation. What can be compared is how the parameter values combine to form emergent patterns in the post-processing of simulations. Furthermore, the patterns of convergence in the parameter values in the optimization reveal where inadequacies in the corresponding independent variables may be.

In this example, in the optimization Series 1, the values of the independent variables clearly accommodated each other through the shifting of their distributions as the parameter ranges were modified (Figure 4.1). If the model structure was adequate and the effects of the variables that the parameters control were meaningful, then the value of one parameter would not necessarily shift in order to accommodate a change in the value of another parameter. How those accommodations occur provides valuable insight into the internal model structure.

Scale in biological models

Scale is recognized as an important issue in biology and physiology (Parker and Peterson 1998). In Series 1, the shifting of parameter values illuminated the deficiency in the mathematical structure of the growth functions. The scale (i.e., physical dimensions)

of the independent variables is an important consideration that I had overlooked in the initial model formulation. If not for the assessment process, the functional form may not have been found to be deficient in that manner. Clearly, in process-model development, variables should be considered individually for the scale at which they occur and over which they may have an effect. In BRANCHPRO2, the branch sets the basic scale, but this is mitigated by the structure of the branch as defined by the division of foliage into SCUs. Some variables in the model act at the scale of the whole branch (e.g., d_{stem}), whereas other variables act locally within the SCU (e.g., d_{SCU}). These considerations should be (and have been) taken into account in the definition and interpretation of the saturation function. In the current example, the scale over which the independent variables have an effect is defined by the half-saturation constants (Table 4.2).

Implications of assessment for component postulates

In the context of a process model, the component postulates listed in Chapter III must be considered in conjunction with the underlying process rules. The implicit assumption in the original formulation of the hydraulic function (equation 4.5, 4.9) is that the effects of generation and bud suppression are integrated into a single function, and that each has an equal contribution to that function. Furthermore, I modified the assumption that the hydraulic constriction is due to the number of turns. The assumption is now that the hydraulic constriction is most severe when the bud is suppressed and then released, and that this restriction worsens as the number of successive constrictions increases. That is, that there is a cumulative effect of the number of hydraulic constrictions caused by bud suppression and release.

The component postulate of the hydraulic constriction on bifurcation rate,
Postulate r.2:

There is a hydraulic constriction at branch junctions formed by the release of suppressed buds along major branch axes in *P. menziesii*. This constriction is controlled by the integrative effects of the number of cumulative turns and age of

the suppressed bud when it originated the axis, which reduces the bifurcation rate in active nodes of *P. menziesii* (eqn 3.18).

will now be replaced with one postulate with two major components.

Postulate r.2:

A more severe hydraulic constriction forms at branch junctions when a bud is suppressed beyond 1 year relative to those only suppressed 1 year within the usual course of growth. This constriction diminishes as the axis expands beyond the point of constriction (the distance in cm to the constriction).

r.2a. This constriction increases as the number of successive junctions that result from bud release from suppression accumulates to an active node.

r.2b. This constriction increases with the age of the bud when it originates the new axis.

Underlying process assumptions:

The hydraulic status of a suppressed bud can be summarized by its time of suppression. Hydraulic restrictions accumulate through successive epicormic junctions. The constriction is larger relative to a bud that is only suppressed one year (within the usual course of bud out-growth).

Any further analysis of the model is taken under the context of these new postulates, in conjunction with the other postulates listed in Chapter III. Evaluation of the model results depends on the finding of an adequate model structure.

Assessment methodology and model adequacy

There are many issues to consider in the process of evaluating model adequacy, as shown by the assessment process in this Chapter. The first is whether the model can satisfy all of the empirical objectives. This is not the end, however, because adequacy depends on *how* the model meets the objectives. The patterns of convergence in the

parameter distributions can uncover issues of parameter compensation that are due to inadequacies in the model structure (Series 1).

In Series 2, there was no apparent compensation among the distributions of parameter values. There was, however, a clear pattern of convergence in the parameter associated with the hydraulic function to the lower edge of its distribution. The rationale to have a single parameter for the hydraulic effect was to limit the number of parameters in the bifurcation function. There is, however, no justification for that assumption, and it makes it impossible to unravel the individual effects of generation and bud suppression on bifurcation. In this stage in the optimization procedure, the pattern of convergence for the hydraulic parameter did not necessarily indicate an issue in the model structure, but it did require some explanation to understand the pattern and to ensure that the source was not a model deficiency. This led to the separation of the hydraulic effect into two functions, each with an associated parameter.

In Series 3, the pattern of convergence was shown to be caused by the r_{gen} parameter. Given that this was an unexpected pattern, I investigated the number of generations on simulated branches and I evaluated the role of r_{gen} through the value of g^* . The branch maps appeared visually adequate, but another pattern emerged in the generation of epicormic shoots that was inconsistent with observed branch development. Given the set of objectives and parameters, the Pareto optimal branches included those that continued release from bud suppression at the base of other epicormics, regardless of whether they have formed sufficient structure for an independent SCU. The uncovering of this deficiency shows the importance of conducting simulations in conjunction with optimization and parameter estimation in the assessment of process models. At this point, the model was clearly not adequate due to anomalous simulation results.

In the final series of Yr90 optimizations (Series 4), the allowable ranges of parameter values were expanded until an appropriate convergence was found. Emergent simulations did satisfy the empirical objectives, and there was a set that appears to

visually resemble the observed growth pattern. There were no obvious anomalies or deficiencies in the model results or structure. With respect to the parameter space, it is imperative to approximate the full Pareto set and to find adequate convergence in the distribution of parameter values. In this context, adequate convergence is a well-defined distribution that does not converge at either end of its search range. The resulting Pareto optimal set (ParetoAll) yields a wide distribution of parameter values, with well-defined distributions (Figure 4.7). The emergent branch forms (Figure 4.6) present markedly different patterns of development that require further analysis.

These results demonstrate that, through the procedure of multi-objective process model assessment, one must not only consider performance with respect to the empirical objectives, but also emergent patterns in the corresponding simulations. In the case of plant models, these patterns include visual representation of the plant morphology as well as quantitative analysis of that pattern.

Chapter V

***P. menziesii* old-growth branch morphology emerges as a trade-off among competing growth requirements, with hydraulic path length as the dominant constraint**

Introduction

A challenge in the study of quantitative morphology and morphogenesis is to explain how observed plant morphology relates to physiological performance of the plant, tree or branch, and if either the morphology or its relation to physiological performance change through time. All plants contend with multiple growth requirements and constraints. Throughout the course of evolution of land plants, the importance of each requirement or constraint as a selection factor changes with the ecological and evolutionary context (Niklas 1992, pp 481–482; Niklas 1997a,b), as well as the ontogenetic legacy of the organism. For example, early in evolution plants were aquatic. The main priority for aquatic plants is light interception, with little need for mechanical support and little risk of water loss (Niklas 1992, pp 9–14). In order for plants to transition to land, they had to contend first with issues of water loss, and then under competition for light mechanical support was an advantage in vertical growth. Underlying these new selection factors is the basic requirement for photosynthesis and the movement of materials, e.g., a plant needs to perform well with this new requirement without sacrificing much of the previous requirements. A dominant growth constraint may emerge at each stage in the evolution of land plants, and within the space bounded by the growth requirement, the other constraints inform the design details. These sometimes competing requirements and constraints are the basis of the multi-objective optimization methodology for old-growth *P. menziesii*.

For long-lived pioneers, the conditions of survival change throughout the process of old-growth forest succession. Franklin et al. (1987) list temporal changes in the causes and rates of tree mortality in stages of succession in *P. menziesii* dominated forests in the Pacific Northwest. In the first stages of succession (the first 200 years of forest development), competition is listed among the top causes of tree mortality. In later

stages, environmental causes of mortality are more dominant, including wind, pathogens and physiological disorders (Franklin et al. 1987). In the old-growth stage of forest development, it is more the ability of trees to persist in their environment, rather than inter-tree competition, that determines tree longevity and mortality. For *P. menziesii*, this in part is accomplished by successful foliage maintenance under severe growth constraints. Old-growth *P. menziesii* trees exhibit morphology and morphogenesis that are distinct from the growth form of younger trees (Ishii and McDowell 2002), and this shift in growth form relates to the new challenges that the trees face as they age and grow larger. There is empirical evidence for physiological changes in *P. menziesii* that seem to compensate for size constraints (McDowell et al. 2002a,b). I seek evidence that the emergent morphology is also a compensation for the unique environment of the old-growth forest.

The multi-objective optimization procedure that I employ integrates both empirical and theoretical objectives. The empirical objectives anchor the optimization results to the observed pattern of growth in *P. menziesii*, but they do not limit the solution set to that pattern. The Pareto optimal set facilitates the answer to the question: Given all possible sets of branch forms (within the structure of the model), which perform best with respect to the theoretical objectives? In the system of Pareto optimality, if growth forms other than those described by the empirical objectives perform better for individual or other combinations of theoretical objectives then they will also be a part of the solution set. In that case we can consider the empirical objectives to create a conditional space within the Pareto frontier, that is, conditional on the set of branch forms that satisfy the four empirical objectives, which perform best with respect to the theoretical objectives (Figure 3.1)? The branch forms are defined both by the model structure and by the emergent parameter values in the Pareto optimal set. In an integrated analysis of the parameter values in the optimal set and relative objective performance, I can propose answers to the questions presented in Chapter III.

How does the branch development pattern of old-growth *P. menziesii* at the WRCCRF compensate for size-related constraints on the species?

- A.1. What are the major system constraints?
- A.2. Of the architectures possible in the model structure, which architecture (branch development pattern) best compensates for the constraints?
- A.3. For which constraints (if any) does the observed *P. menziesii* branch development pattern compensate?
- B.1. What is the relative importance of different branch structural characters (e.g., architectural order, foliage overlap, nearby growth) in determining the observed branch development pattern in *P. menziesii*?

In this Chapter, I will present the morphological and physiological implications of the results of optimization Series 4 and Series 5 that were described in Chapter IV. This Chapter is organized into six sections. In the first section, I describe the quantitative measures used to evaluate the optimization results. In the second through fifth sections, I present the analyses relevant to questions A.1-A.3 and B.1 above. There is a final section to integrate the overall analyses.

Quantitative measures

Dynamic value of bifurcation

The emergent bifurcation values of simulated branches relate in a dynamic manner to the bifurcation rate parameter as calculated in model equation 4.10, but there is no analytical method to calculate the mean bifurcation ratio across an entire branch for a shoot of a given order and generation. Simulation is necessary to determine the emergent bifurcation of different ordered axes, which are determined by the interactions of other dynamic variables. These are reported as $Rb0_{sim}$, $Rb1_{sim}$ and $Rb2_{sim}$ for order 1, order 2 and order 3 respectively.

Expected value of bifurcation

To supplement understanding of simulated values, an analytical analysis is possible through the calculation of expected bifurcation at various combinations of the independent variables. For the usual Poisson distribution, the expected value is the rate parameter. In the modified Poisson distribution used in BRANCHPRO3, the expected value is less than the rate parameter; this value asymptotes at three as the rate parameter increases (Figure 2.2a). The expected number of daughter shoots (k) is related to the rate parameter (r) in the following manner:

$$E(k) = \sum_k kp(k) = p(1) + 2p(2) + 3p(3); p(3) = 1 - (p(0) + p(1) + p(2)); \quad (5.1)$$

$$E(k) = e^{-r} (r + r^2) + 3 \left\{ 1 - e^{-r} \left(1 + r + \frac{r^2}{2} \right) \right\}. \quad (5.2)$$

I calculate the expected value for each shoot order (order 1, 2 and 3) with increasing values of n_{over} to evaluate how each branch order responds to increased light levels (assuming all other variables are zero). For new epicormic shoots, the expected value is calculated with increasing t_{base} for generation 2, 3, 4 and 5 axes. This will evaluate the effect of both generation and t_{base} on the proliferation of new SCUs.

Generation threshold and axis longevity

The process of shoot proliferation can also be evaluated analytically through threshold values for r at different levels generation. The threshold is defined as the value of the independent variable at which the r function crosses zero. The value of g^* (eqn 4.12) defines the theoretical maximum generation at which a new epicormic shoot proliferates, and hence a new SCU can develop. It is calculated first at zero n_{over} and t_{base} (eqn 4.12). The equation can also be modified to incorporate the value of either of these variables as follows:

$$g^* = - \frac{4 \left\{ r_0 + r_{over} \frac{n_{over}}{8 + n_{over}} + \frac{r_{base}}{d_{base}} \left(\frac{t_{base}}{5 + t_{base}} \right) \right\} d_{base}}{r_{gen} + \left\{ r_0 + r_{over} \frac{n_{over}}{8 + n_{over}} + \frac{r_{base}}{d_{base}} \left(\frac{t_{base}}{5 + t_{base}} \right) \right\} d_{base}} . \quad (5.3)$$

I calculate the value of g^* for zero n_{over} and zero t_{base} , and compare the value of g^* among optimization runs. The value is also calculated for increasing values of n_{over} and t_{base} . Further interpretation of the pattern of branch development is made through the expected longevity of a given axis, which increases exponentially with the bifurcation rate parameter (Figure 2.2b).

Question A.1 Major system constraints

Strategy

To answer question A.1, we must evaluate the performance of model simulations with respect to the theoretical objectives. Of particular interest will be changes in parameter values and respective branch morphologies with changing theoretical objective performances. I begin by evaluating the pattern of convergence for Series 4 and 5, then I compare the dominant branch forms with respect to the theoretical constraints and the process of reiteration.

Analysis of branch development partitions

In order to achieve adequate convergence of the parameter values, a series of optimizations was necessary in the assessment of BRANCHPRO2 and BRANCHPRO3 that was presented in Chapter IV (series 4; Figure 4.5). This series was driven by the consistent convergence of r_{gen} at the lower edge of its search range (Figure 4.4). There were 16 optimization searches in Series 4. Each search in the series can be characterized by the minimum value of r_{gen} allowed in the search, and the searches can be compared by the summaries of each theoretical objective (n_{turns} , l_{path} , $load$, and n_{over}). The minimum, 10th percentile and median theoretical objective values are plotted against the minimum

r_{gen} allowed in the corresponding search in order to explain the patterns of the theoretical objective values through the optimization Series 4.

In optimization Series 4, three of the four theoretical objectives decrease as the r_{gen} parameter becomes more negative, with only l_{path} increasing (Figure 5.1). It is clear that the optimization searches with the highest values for r_{gen} tend to minimize the value of l_{path} (Figure 5.1b), whereas optimization searches with more negative values of r_{gen} tend to minimize n_{turns} (Figure 5.1a). There is a tradeoff between these two objectives through the series of optimization searches, which also emerges in the optimization Series 5 (not shown).

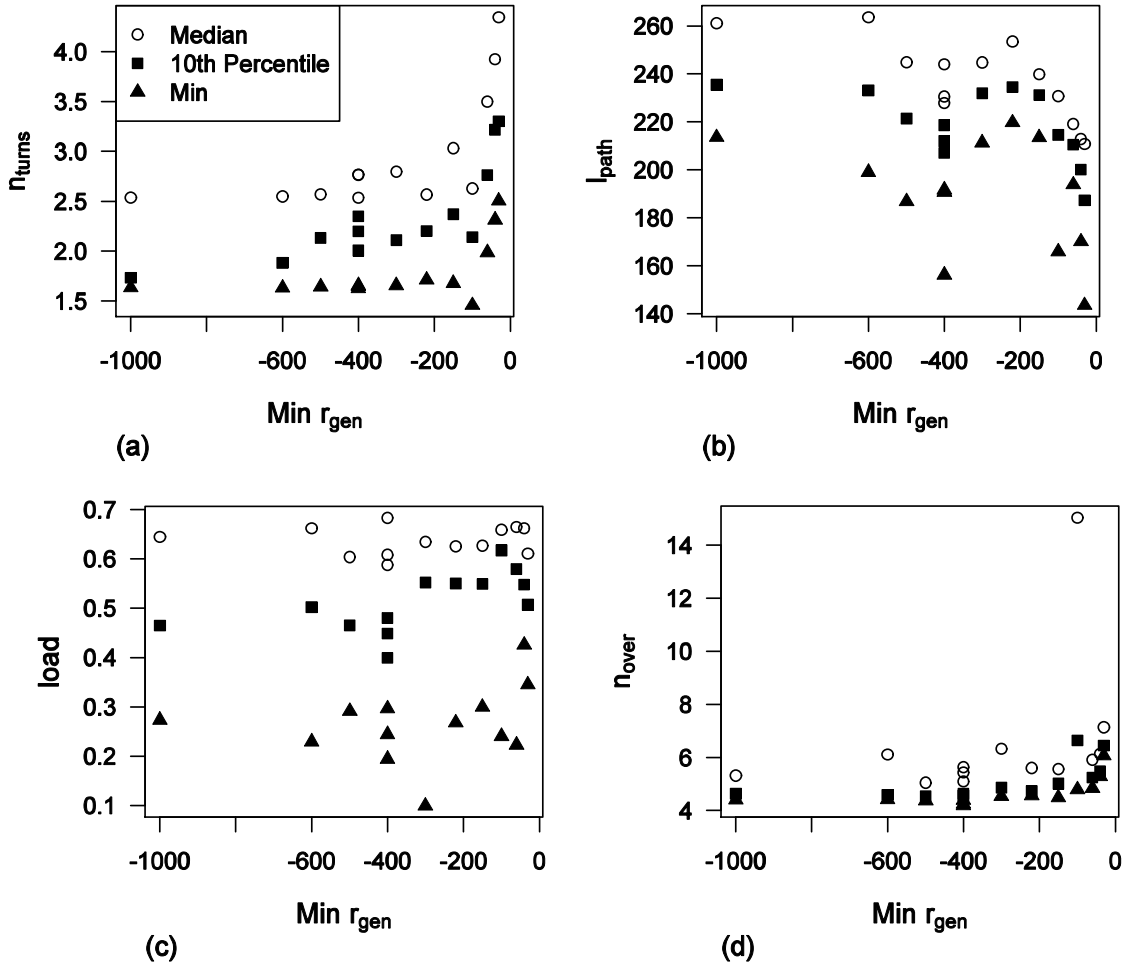


Figure 5.1. Median, 10th percentile and minimum theoretical objective values for each optimization search in Series 4. Each optimization search is represented by the minimum r_{gen} value allowed in the search. The value of (a) n_{turns} tends to decrease with decreasing r_{gen} , whereas (b) l_{path} tends to increase. (c) The relationship between minimum r_{gen} and load is not clear, and (d) n_{over} decreases slightly with increasing minimum r_{gen} . These relationships characterize changes in relative performance of the branches illustrated by Figure 4.6 through the series of optimization searches (Series 4).

Example branches were also simulated from representative parameter combinations in each search in Series 4 in order to compare branch patterns through the optimization series. In Chapter IV, I presented maps for these example branches. Figure 4.6a and 4.6b correspond to parameterizations characteristic of optimization searches that tend to have lower values of l_{path} (the right-hand side of Figure 5.1b). Figures 4.6h and 4.6i correspond to parameterizations characteristic of optimization searches that tend to have lower values of n_{turns} (the left-hand side of Figure 5.1a). There is a qualitative shift in the branch pattern through the series of optimizations (Figure 4.6), which corresponds

to a transition from morphologies that tend to minimize l_{path} and morphologies that tend to minimize n_{turns} . Here we see the first evidence that the major defining constraint on the old-growth branches is hydraulic, yet there are alternative morphologies that compensate for such a constraint through a trade-off between n_{turns} and l_{path} . This initial analysis does not indicate which feature is a more dominant constraint, or if the other theoretical objectives influence the morphologies (see below).

The difference in morphology is explained in part by the shoot bifurcation rates and the longevity of major branch axes (Table 5.1). As the search range for r_{gen} is shifted to smaller values, the values of r_0 shift to larger values (Figure 4.5). The potential longevities of order-1 axes increase with the r_0 parameter because expected longevity of a given axis increases exponentially with the bifurcation rate parameter (Figure 2.2b). As we transition from an l_{path} minimizing morphology to one that minimizes n_{turns} , order-1 and order-2 longevity and emergent bifurcation values increase, whereas emergent order-3 bifurcation transitions from less than one to zero (Table 5.1). This means that on branches that tend to minimize n_{turns} , order-3 shoots that occur as a result of order-2 bifurcation do not proliferate. Furthermore, the value of g^* tends to decrease through the transition between morphologies; an n_{turns} minimizing morphology tends not to proliferate at higher generations. In order to understand these differences in relation to the model structure and objectives, the parameter space is partitioned into the two major patterns and analyzed in detail.

Table 5.1. Nine simulated branch examples from optimization Series 4 (Chapter IV). r_{gen} is the parameter value that modifies bifurcation with increasing generation, $Rb1_{sim}$, $Rb2_{sim}$ and $Rb3_{sim}$ are emergent bifurcation values for order-1, order-2 and order-3 shoots on the simulated branch. Median SCU age and expected longevity are given in years. The value g^* is the maximum generation an SCU can develop on the branch. The branches in the first two rows represent branches from Part1, whereas the branches in the last two rows represent branches from Part2.

r_{gen}	$Rb1_{sim}$	$Rb2_{sim}$	$Rb3_{sim}$	Median SCU age (years)	Expected longevity Ord1 (years)	Expected longevity Ord2 (years)	Expected longevity Ord3 (years)	g^*
-24.8	2.38	1.06	0.39	23	121	7	2	9.02
-39.8	2.71	1.82	0.91	35	1.99×10^3	7	0	8.64
-60	2.94	1.57	0.3	38	4.01×10^4	244	0	6.88
-149.5	3	2.64	0	51	1.61×10^9	1.71×10^4	0	4.12
-220	3	2.67	0	81	5.89×10^{10}	2.43×10^4	0	2.71
-274.3	3	1.87	0	43.5	2.13×10^{16}	1.04×10^3	0	3.85
-398	3	2.73	0	83.5	8.64×10^{17}	5.14×10^5	0	2.36
-581.5	3	2.84	0	58	6.30×10^{23}	3.97×10^8	0	2.03
-595	2.75	2.38	0	59	1.50×10^{36}	3.37×10^{13}	0	4.02

Partition of the parameter space

It is evident that many branch forms emerge in the optimal space, and some of these match the empirical objectives. However, qualitatively different branch morphologies are formed (Figure 4.6), and it is the constraint on l_{path} and the constraint on n_{turns} that drive the divergence in branch forms. To study these two patterns in more detail, the optimizations in Series 4 and 5 are partitioned into two groups: Part1 (l_{path} partition) and Part2 (n_{turns} partition) for both Yr90 and Yr145 (Part1₉₀; Part1₁₄₅; Part2₉₀; Part2₁₄₅; Table 5.2). The values of the theoretical objectives are compared between partitions, and within partitions across SCU levels (low, target, high). I also compare analysis of Yr90 optimizations to the results to Yr145 optimizations.

Table 5.2. Allowable parameter ranges for Part1 and Part2

Parameter	Part1 ₉₀ allowable range	Part2 ₉₀ allowable range	Part1 ₁₄₅ allowable range	Part2 ₁₄₅ allowable range
r_0	(0,10)	(40,100)	(0,10)	(50,150)
r_{ord}	(-15,0)	(-150,-30)	(-15,0)	(-300,-100)
r_{gen}	(-40,0)	(-1000,-500)	(-40,0)	(-1500,-500)
r_{over}	(-20,0)	(-130,0)	(-20,0)	(-130,0)
r_{base}	(-10,0)	(-100,0)	(-10,0)	(-150,-50)
p_0	(0.001,0.25)	(0.001,0.25)	(0,0.25)	(0.001,0.25)
p_{bud}	(-0.25,0.25)	(-0.25,0.05)	(-0.25,0.25)	(-0.25,0.25)
p_a	(0,0.25)	(0,0.15)	(0,0.25)	(0,0.25)
p_{SCU}	(0,0.25)	(0,0.15)	(0,0.25)	(0,0.25)
p_{stem}	(0,0.25)	(0,0.25)	(0,0.25)	(0,0.25)

Objective relationships

The results of linear models produced for the relationship between each objective and the number of SCUs depends on the partition and the SCU level. First I consider all solutions in each partition. In Part1, n_{turns} significantly decreases with SCUs, and the remaining theoretical objectives significantly increase with SCUs (Table 5.3). There is an opposite pattern in Part2, where n_{turns} increases with SCUs and the remaining objectives decrease (Table 5.3). If we consider only low SCU solutions, we see the same pattern in both Part1 and Part2 as was observed in Part2 for all solutions. For target SCUs the pattern is the same for the Part1 solutions regardless of the partition.

Table 5.3. Slope of linear relationship between each theoretical objective value and the number of SCUs for each solution partition. Grey-shaded cells indicate *non-significant* relationships. All remaining coefficients are *significant*.

		Full solution	Low SCUs	Target SCUs
n_{turns}	Part1 ₉₀	-0.017	0.019	-0.048
	Part2 ₉₀	0.05	0.097	0.01
	Part1 ₁₄₅	-0.006	0.069	-0.042
	Part2 ₁₄₅	0.028	0.041	-0.0178
l_{path}	Part1 ₉₀	0.53	-3.36	2.35
	Part2 ₉₀	-1.4	-2.9	0.39
	Part1 ₁₄₅	0.27	-2.87	1.03
	Part2 ₁₄₅	-0.36	-2.19	-0.011
load	Part1 ₉₀	0.0017	-0.007	0.004
	Part2 ₉₀	-0.01	-0.031	0.002
	Part1 ₁₄₅	0.0005	-0.005	0.001
	Part2 ₁₄₅	0.0008	-0.008	0.0004
n_{over}	Part1 ₉₀	0.3	-0.23	0.92
	Part2 ₉₀	-0.027	-0.35	0.29
	Part1 ₁₄₅	0.13	-0.15	0.25
	Part2 ₁₄₅	-0.003	-0.24	0.02

Clearly, the tradeoffs among theoretical objectives depends on the level of reiteration in the branch as defined by the number of SCUs, with solutions in Part1 characterized by greater SCU development and solutions in Part2 are characterized by less SCU development. The main characteristic of these tradeoffs is the opposite trend between n_{turns} and the remaining theoretical objectives, where when n_{turns} decreases with SCUs the remaining objectives increase, and vice versa. This tradeoff is seen clearly with a pair-wise scatter plot between n_{turns} and the remaining three objectives, shown for illustration within Part1₉₀ (Figure 5.2).

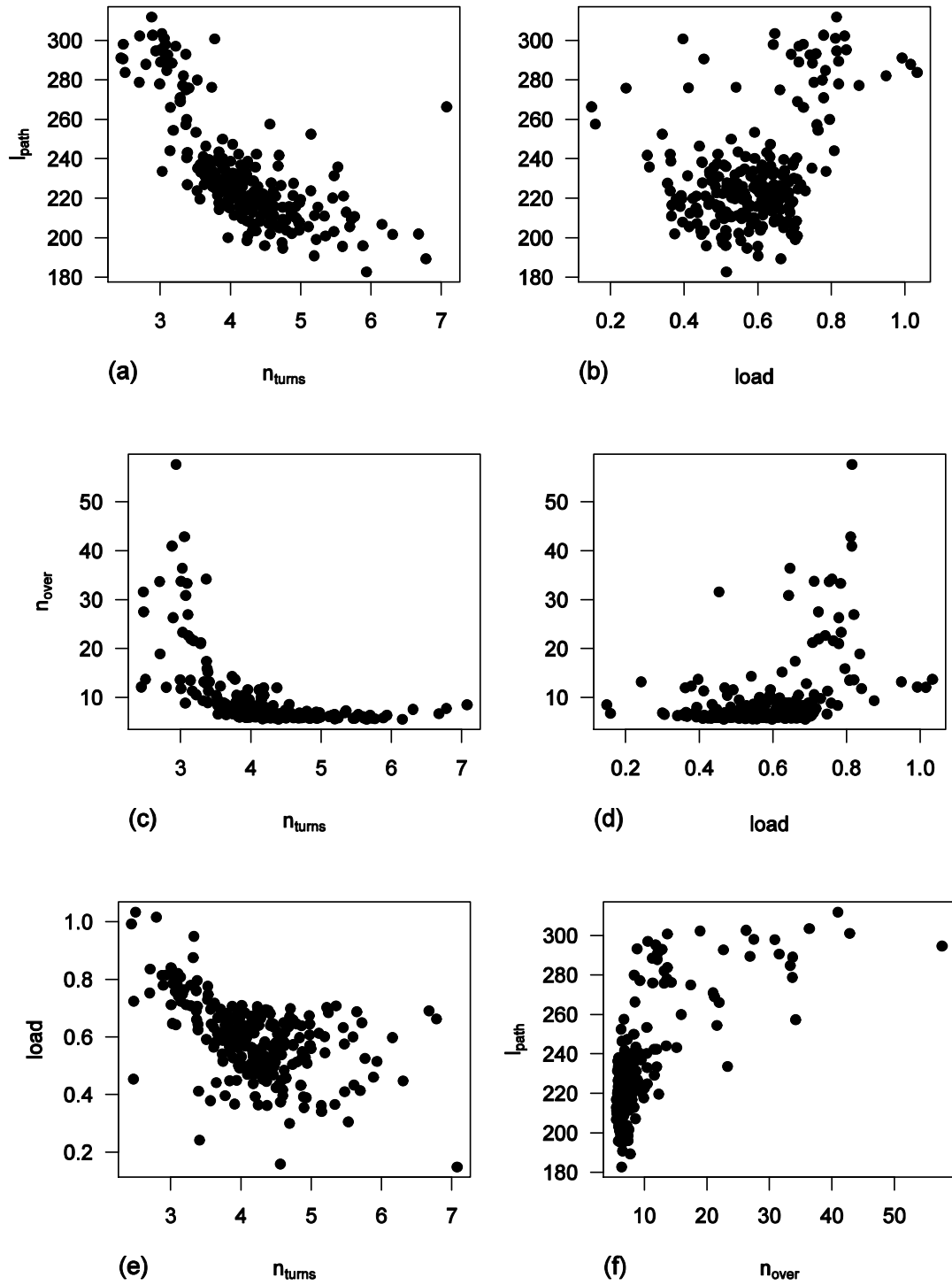


Figure 5.2. Pair-wise scatter plots of the theoretical objective value for Part190. The objective n_{turns} has negative relationships with the remaining three theoretical objectives (a, c, e). In contrast, l_{path} , $load$ and n_{over} exhibit positive pair-wise relationships (b, d, f).

The objective relationships across levels of SCUs are similar between the Yr90 and Yr145 optimizations (Table 5.3). The main differences are in the non-significant relationships within target SCUs with n_{turns} for Part2₉₀ and in l_{path} in both Part2₉₀ and Part2₁₄₅. Finally, in Part2₁₄₅ the relationship between n_{over} and SCUs is non-significant in the target SCUs.

In comparing Part1₉₀ and Part2₉₀ objective values (Figure 5.3), clearly Part1₉₀ performs better with respect to l_{path} and Part2₉₀ performs better in n_{turns} ; this is also true for Part1₁₄₅ and Part2₁₄₅. The values for Part1₁₄₅ and Part2₁₄₅ tend to be higher than the respective Part1₉₀ and Part2₉₀. The differences between the partitions in $load$ and n_{over} are much less clear. Values of $load$ might be slightly higher in Part2₉₀, and n_{over} slightly higher in Part1₉₀. There is a much clearer difference in load between Part1₁₄₅ and Part2₁₄₅ (Figure 5.3c). For objective values, in Part1 the Yr145 optimization has higher values for n_{turns} , l_{path} and load, with the most obvious distinction in l_{path} . In Part2, n_{turns} and l_{path} are higher for Yr145 optimization, whereas there is some evidence that load and n_{over} are lower for Yr145 optimization.

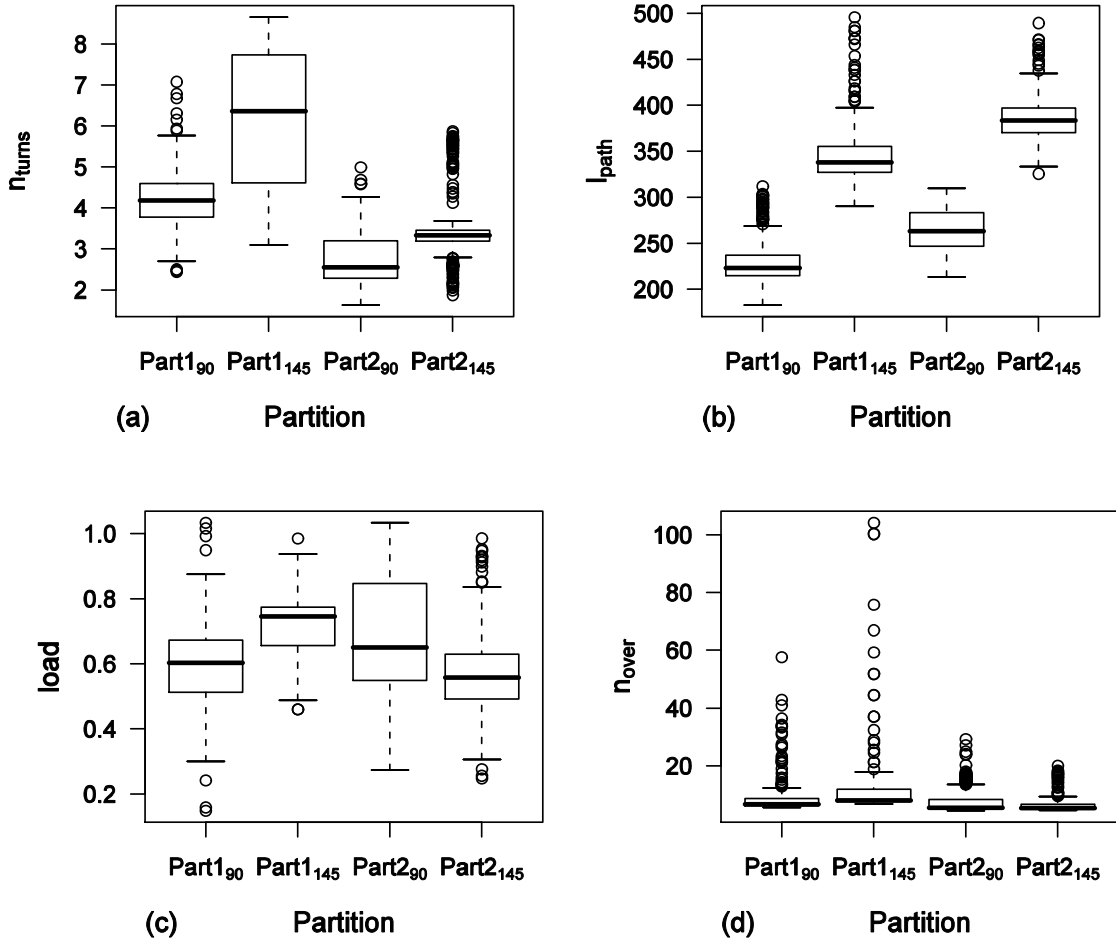


Figure 5.3. Distribution of theoretical objective values compared between Part1 and Part2, for both Yr90 and Yr145 optimizations. The objective n_{turns} tends to decrease between Part1 and Part2 (a), whereas l_{path} increases between Part1 and Part2 (b). There are no clear differences between Part1 and Part2 in $load$ (c), and n_{over} (d).

Summary

The emergence of two clear branch morphologies in the optimization Series 4 and 5 indicates that the dominant constraint in the old-growth branch system, as defined by the model, might be hydraulic. The analysis shows that the major design specification that differentiates each morphology is whether l_{path} (Part1) or n_{turns} (Part2) is minimized. In comparing the two partitions, there are opposite relationships between each of the theoretical objective values and the number of SCUs (Table 5.3). The degree of reiteration evident in Part1 may be a result of the tradeoffs between n_{turns} and the remaining three theoretical objectives (Figure 5.2).

Question A.2 Relative performance of branch architectures

Strategy

We have already seen that branches in the two major partitions perform differently with respect to the theoretical objectives (Question A.1). We must now evaluate the architectures that characterize the branches in each partition in order to understand which kind of architecture performs best for which combination of theoretical objectives. We begin by comparing the parameter distributions between the two major partitions.

Parameter patterns: distributions

When comparing the distributions of parameter values between Part1₉₀ and Part2₉₀ with all SCU levels combined (Figure 5.4), the separation of the partitions by parameters of the r function is obvious. Part2₉₀ shows a higher magnitude (more positive for r_0 , more negative for the remaining parameters) for all of the parameter values relative to Part1₉₀. The distributions of parameter values for the p function are more similar than for the r function between Part1₉₀ and Part2₉₀, but some differences emerge. The distribution of p_0 is shifted to the right for Part2₉₀ relative to Part1₉₀, and the distribution of p_{stem} is shifted far to the left for Part2₉₀ (Figure 5.5).

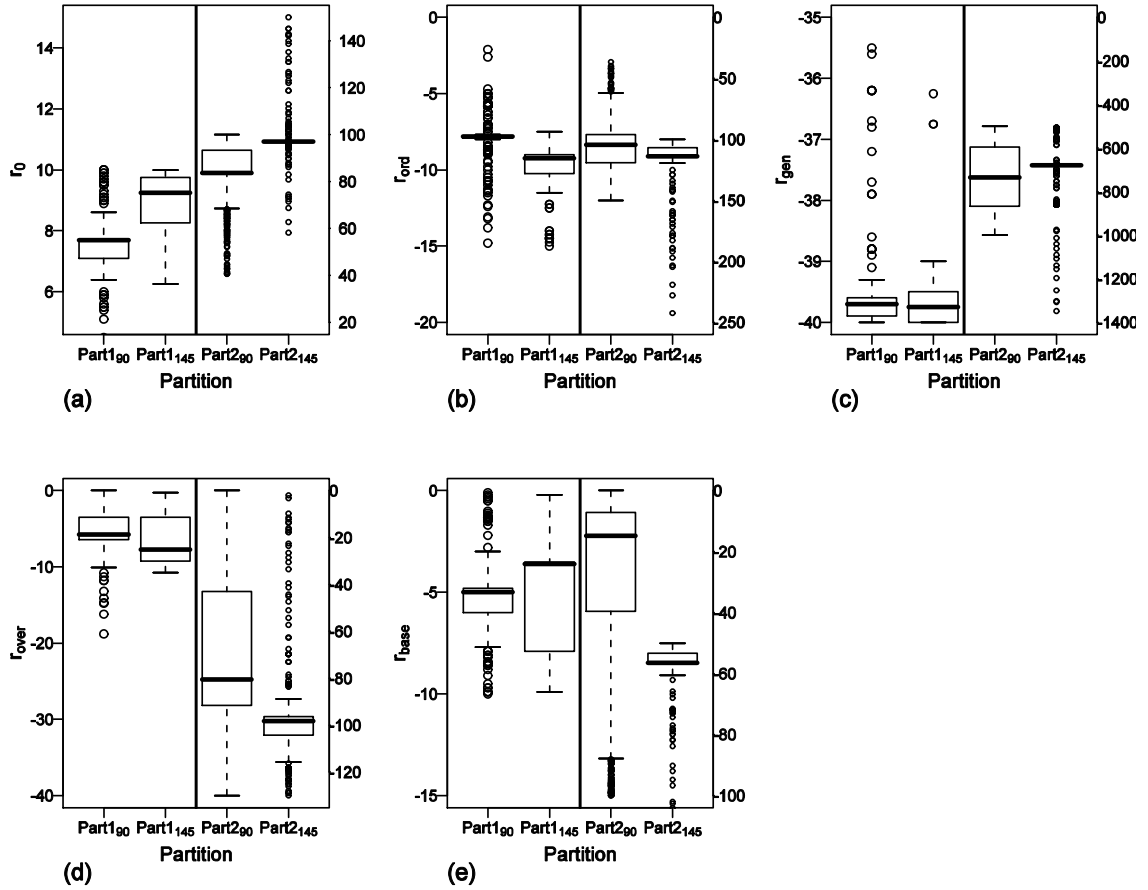


Figure 5.4. Distributions of parameter values for the r function, comparing Part1₉₀, Part1₁₄₅, Part2₉₀, Part2₁₄₅. Note the vertical line separates Part1 and Part2, and the left-hand axis is for Part1 and the right-hand axis is for Part2. For every parameter in the r function, the values in Part2 are of higher magnitude (more positive for r_0 , more negative for the remaining parameters) than Part1.

It is clear that the patterns of convergence in the Pareto optimal space for branches optimized at Yr90 and at Yr145 are very similar with respect to the overall branching pattern and the relative performance in the theoretical objectives (i.e., two obvious partitions driven by l_{path} and n_{turns}). There are differences, however, when the parameter and objective values are compared within each partition between Yr90 and Yr145 optimizations. Within Part1, the distribution of r_0 is shifted higher for Yr145 optimizations relative to Yr90 optimizations (Figure 5.4a), whereas the distributions of r_{ord} and r_{over} are slightly lower for Yr145 optimizations (Figure 5.4b,d). The distributions of p_{bud} and p_{SCU} are shifted higher for Yr145 optimizations (Figure 5.5b,d), and p_a and p_{stem} are shifted lower for Yr145 optimizations (Figure 5.5c,e). There are similar patterns

in Part2 parameter distributions, with r_0 ranging higher for Yr145 optimization, and r_{ord} , r_{ord} , and r_{bud} ranging lower for Yr145 optimization.

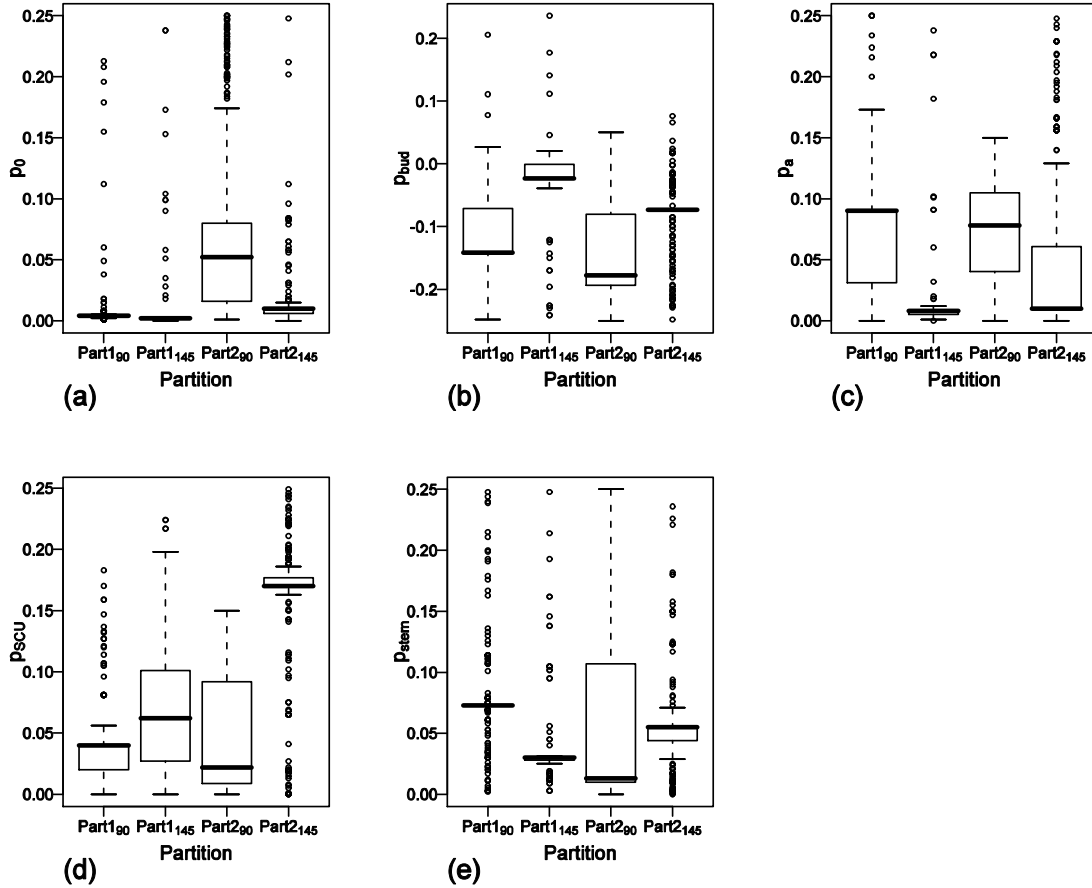


Figure 5.5. Distributions of parameter values for the p function, comparing Part1₉₀, Part1₁₄₅, Part2₉₀, Part2₁₄₅. There are clear differences between the partitions and between years within a partition. These reflect contrasts in morphogenesis between the branch types.

These differences in parameter values between the two partitions have marked consequences for the emergent branch morphologies (Figure 4.6). In order to understand how the parameter values regulate the effects of the biological variables in the growth functions, we must evaluate how they integrate with respect to the overall growth of the branch. This is accomplished both through analytical analyses (g^* and expected bifurcation) and through simulation.

Parameter partitions: g^ and expected bifurcation*

The value of g^* (maximum SCU generation, n_{over} and t_{base} are zero) tends to be higher in Part1₉₀ than in Part2₉₀ (Table 5.4). The first quartile of g^* in Part1₉₀ is 6.8 and the third quartile is 9.1 (Table 5.4). In contrast, the first quartile of g^* in Part2₉₀ is 2.27 and the third quartile is 3.91, well below the values in Part1₉₀. Reiteration, with respect to the successive occurrences of epicormic initiation, is more limited in Part2 than in Part1. The values of g^* are similar between Part2₉₀ and Part2₁₄₅, but Part1₁₄₅ has much higher values than Part1₉₀.

Table 5.4. Summary of g^* values for Part1 and Part2 Pareto optimal sets. Q1 and Q3 are the first and third quartiles, and g^* is the theoretical maximum generation at which an independent SCU can develop.

	Q1	Median	Q3
Part1 ₉₀	6.81	8.9	9.16
Part2 ₉₀	2.27	2.75	3.91
Part1 ₁₄₅	9.24	19.96	28.82
Part2 ₁₄₅	2.2	2.645	3.25

The value of g^* presented thus far in the analysis has been calculated for a shoot with zero n_{over} and it ignores the effect of t_{base} . When g^* is calculated for increasing values of n_{over} (and t_{base} is set to zero; eqn 5.3), the effect of increasing n_{over} on g^* differs between the two partitions and across the levels of SCUs (Figure 5.6a). The change in median value of g^* is similar between low and target levels of SCUs for both partitions. In Part1₉₀, the median value of g^* for high SCUs is lower at low values of n_{over} , but the value declines less steeply than for low and target SCU levels. At higher values of n_{over} , g^* is higher for high SCUs than for low and target SCUs. The values of g^* for Part2₉₀ are lower than for Part1₉₀, and the median values decline less steeply than for Part1₉₀. There is a small separation between the median values of low and target SCUs for Part2₉₀. Target SCUs in Part2₉₀ have a higher median g^* than the low SCU level at low values of n_{over} . As n_{over} increases, the median values of g^* for Part2₉₀ converge between low and target SCU levels. The pattern is similar for Part1₁₄₅ and Part2₁₄₅, except overall

there are much higher values and steeper decline in Part1₁₄₅ than Part1₉₀ (Figure 5.6c). Note that there are no high SCU solutions in Part2₁₄₅.

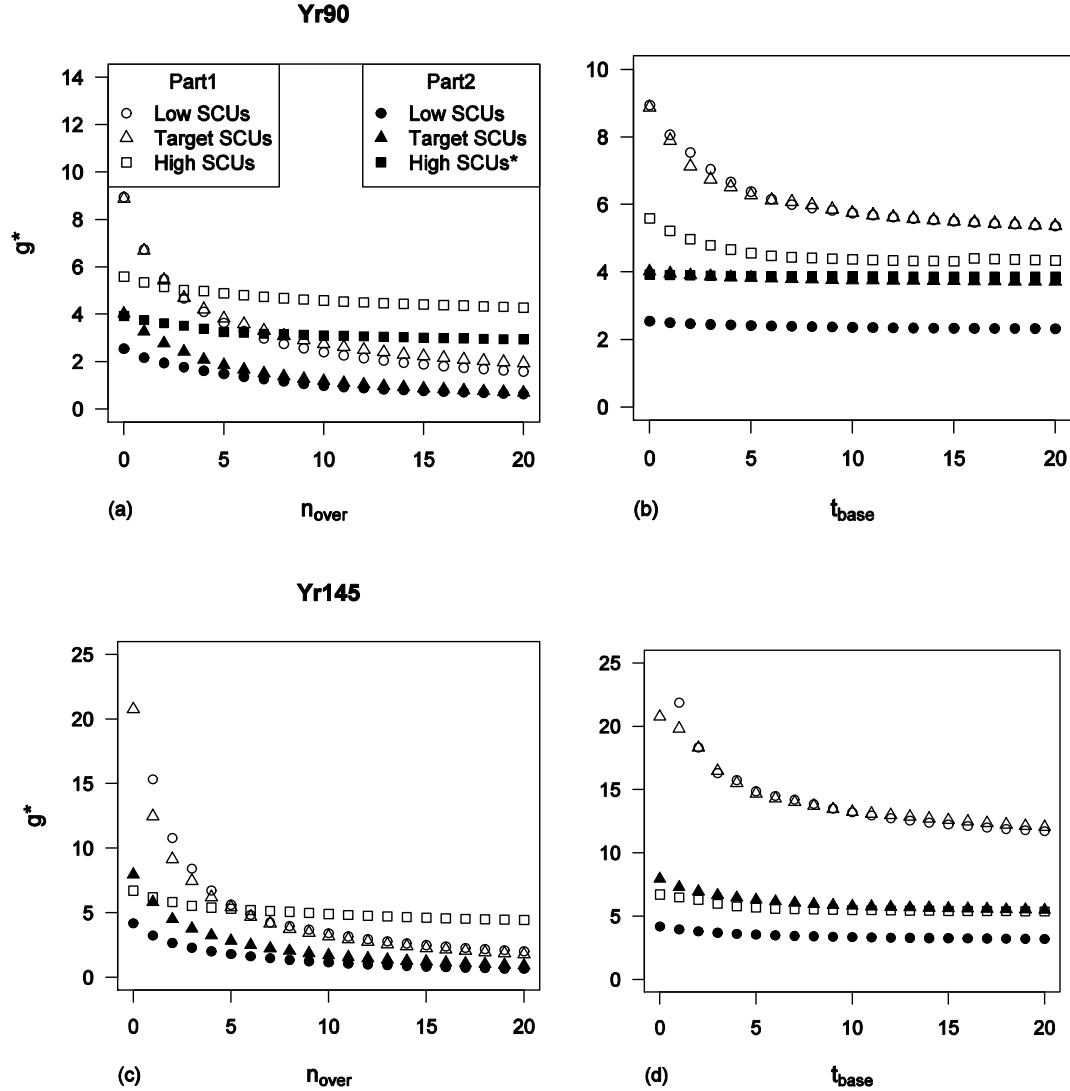


Figure 5.6. Change in median g^* with increasing (a,c) n_{over} and (b,d) t_{base} for low, target and high SCU levels in Part1₉₀ and Part2₉₀ (a,b), and Part1₁₄₅ and Part2₁₄₅ (c,d). The values of g^* tend to be higher for Part1 than Part2, and the relationship differs between SCU levels. The values also tend to be higher for Yr145 than Yr90 Pareto optimal sets. *Note that there are only 3 solutions in Part2₉₀ with high SCU solutions (a,b), and no solutions in Part2₁₄₅ with high SCUs (c,d). The target SCU values are those observed for branches at the WRCCRF, and are listed in Table 3.1).

The effect of increasing t_{base} on g^* also differs between the partitions when n_{over} is zero (Figure 5.6b). In Part1₉₀ the relationship between median g^* and t_{base} is similar between low and target SCUs, with median g^* for high SCUs consistently lower than

target and low SCUs. For high SCUs, median g^* declines more slowly with t_{base} than low and target levels. For Part2₉₀, the values are similar between high and target SCUs, with median g^* of low SCUs consistently lower than high and target SCUs. For Part1₁₄₅, the values of g^* are higher than Part1₉₀ and the values decline more steeply with increasing t_{base} (Figure 5.6d). There is also a greater gap between median values of g^* between high SCUs and target and low SCUs. Median g^* for Part2₁₄₅ also tend to be higher than Part2₉₀, and decline more steeply.

The contrast in the relationship between g^* and n_{over} between the two partitions illustrates how the two distinct morphologies emerge through branch development. In Part1, SCU placement is sensitive to the environment of new epicormic shoots. If the foliage overlap of the new epicormic shoot is high, then it is less likely to proliferate shoots at a level sufficient to develop into a new SCU. This places a threshold on reiteration relative to the foliage overlap, so that higher generation SCUs do not occur at high overlap. In contrast, Part2 branches have very low generation thresholds regardless of overlap or t_{base} . These are not likely to regenerate foliage at high generations, and do not seem to respond much to differences in shoot condition. Another measure of the effect of n_{over} on the branch architecture is the change in expected value of bifurcation with increasing n_{over} .

The expected number of daughter shoots for both partitions declines with increasing n_{over} . The decline is more obvious for order-2 and order-3 shoots than order-1 shoots in Part1 (Figure 5.7). In Part2, n_{over} has no effect on the expected number of daughters for order-1 shoots at low values of n_{over} . The effect is small on expected value for higher n_{over} (Figure 5.7a). Similarly, for order 2 there is only a small effect of n_{over} on expected number of daughters at low values of n_{over} . For order 3, the expected number of daughters is highly variable at low n_{over} , but approaches zero at n_{over} greater than 10 (Figure 5.7c). Overall the pattern of decline of expected bifurcation with increasing n_{over} across shoot orders is similar between Yr90 and Yr145 optimizations (Figure 5.7; Figure

5.8). Within each shoot order, however, the expected values for Yr145 optimizations tend to be higher than Yr90 optimizations.

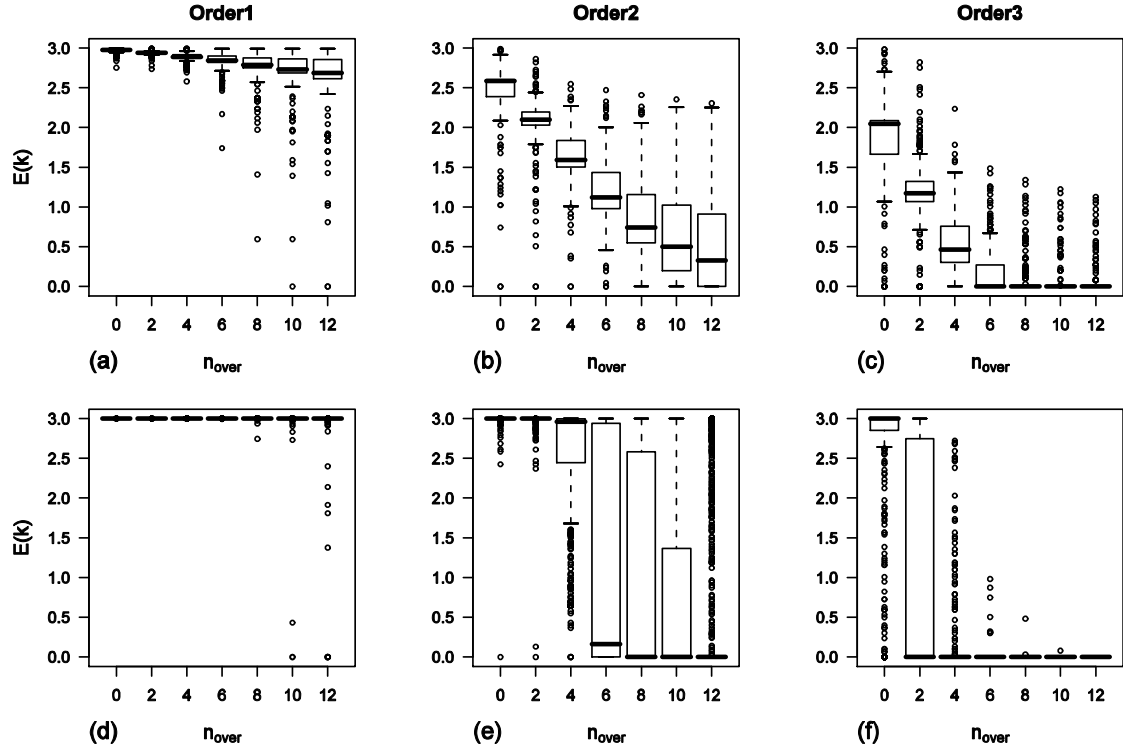


Figure 5.7. The distribution of expected number of daughter shoots ($E(k)$) with increasing n_{over} for solutions in Part1₉₀ (a-c), order 1, order 2 and order 3 respectively and Part2₉₀ (d-f), order 1, order 2 and order 3 respectively. Across shoot orders, the Yr90 Pareto optimal set for Part1 shows continual decline with increasing n_{over} , although the decline is very minor for order 1 axes. In contrast, Part2 exhibits a more extreme decline, with a relationship approaching a threshold for orders 2 and 3, and no visible decline with order 1 up to $n_{over}=12$.

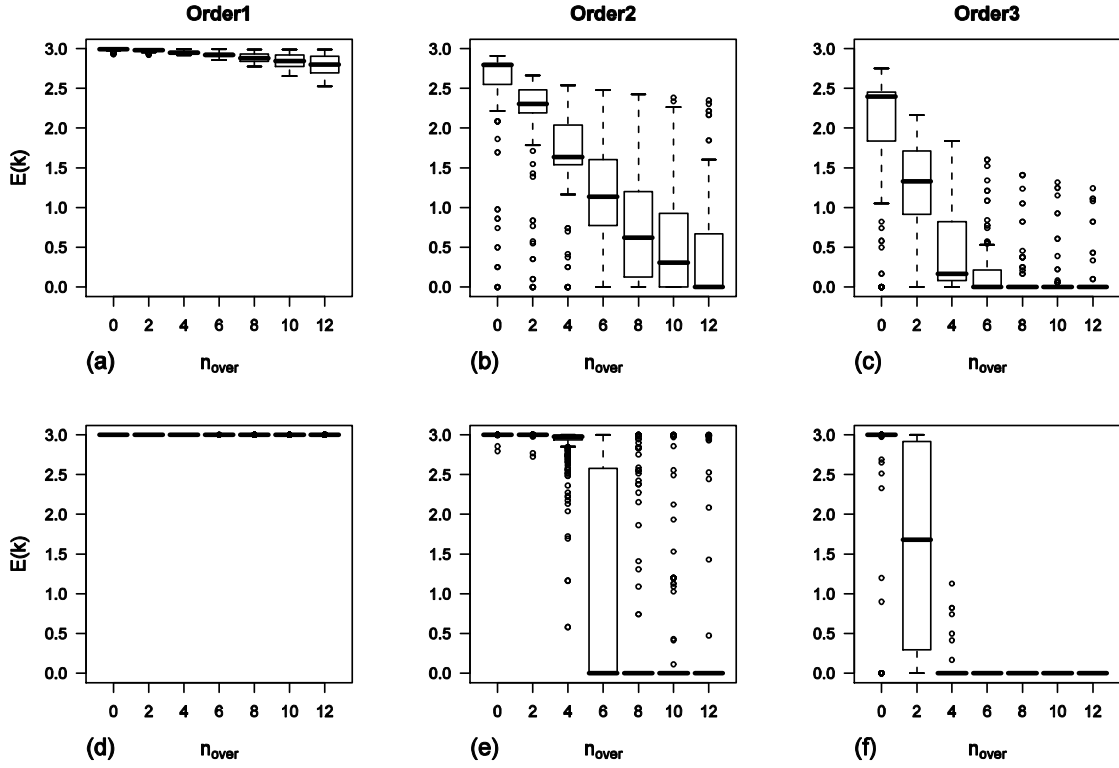


Figure 5.8. The distribution of expected number of daughter shoots ($E(k)$) with increasing n_{over} for Part1₁₄₅ (a-c) and Part2₁₄₅ (d-f) for order 1 (a,d), order 2 (b,e) and order 3 (c,f) shoots. The distributions clearly differ between the two partitions, with Part2₁₄₅ showing a threshold of expected number of daughter shoots for order 2 (falling from near 3 to near 0) at $n_{over}=5$ and at $n_{over}=2$ for order 3. This is more extreme than that observed for Yr90 Pareto optimal sets (Figure 5.7). In contrast, Part1₁₄₅ shows a more gradual decline with increasing n_{over} across all shoot orders.

When we consider the bifurcation of new epicormic shoots, we can calculate the expected number of daughter shoots with increasing t_{base} for different generations. For Part1, the expected number of daughter shoots declines steadily with increasing t_{base} , and the distributions shift downward as generation increases (Figure 5.9). For Part2, the expected number of daughter shoots is more controlled by generation than by t_{base} , with most of the distribution either near 3 for low generation and near zero for higher generations (Figure 5.10). The patterns are similar between Yr90 and Yr145 optimizations, although expected bifurcation is higher for Yr145 optimizations, and tends to decline more slowly for Yr145 optimizations than Yr90 (Figure 5.9; Figure 5.10).

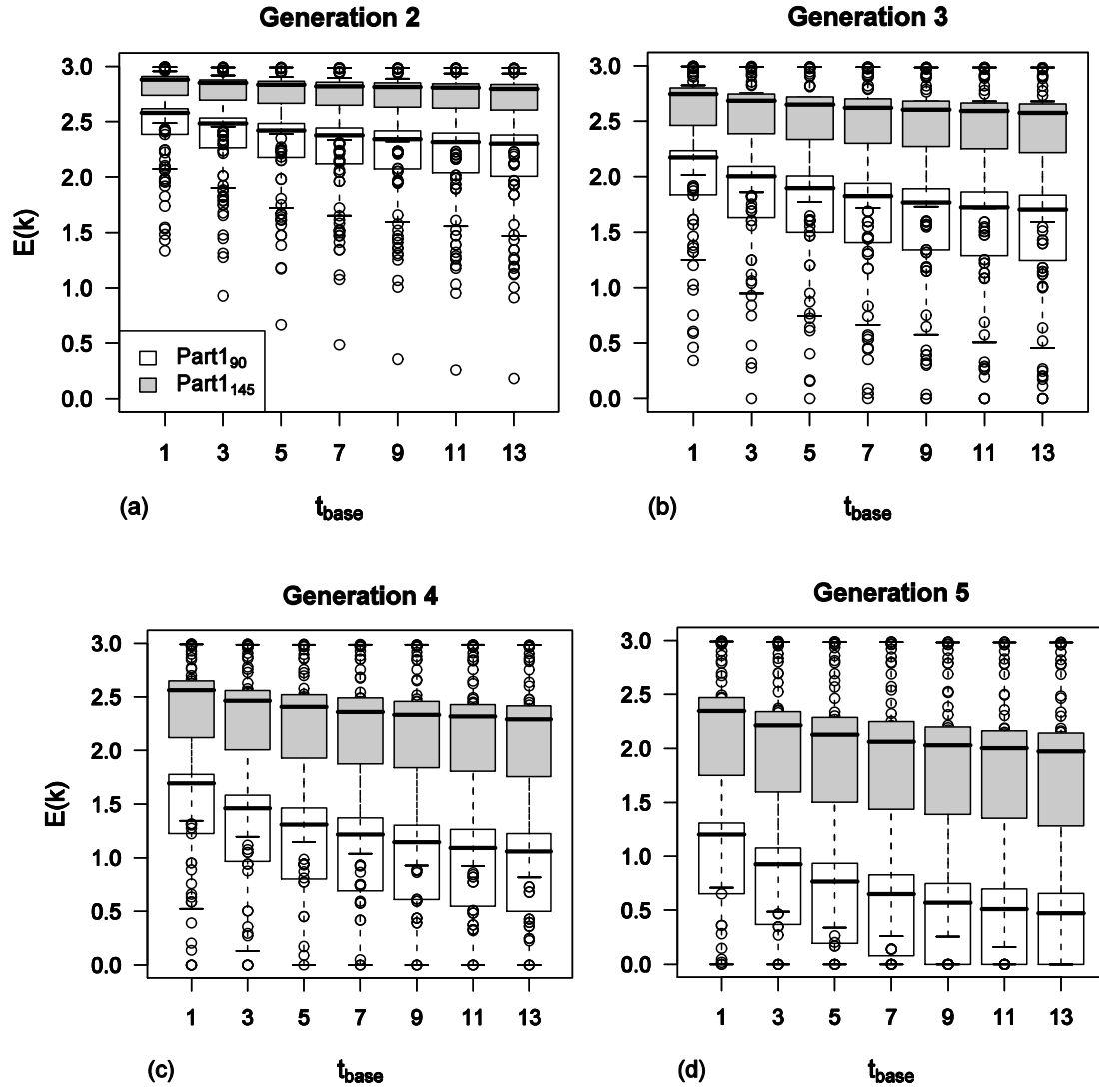


Figure 5.9. The distribution of expected number of daughter shoots ($E(k)$) with increasing t_{base} for generation (a) 2, (b) 3, (c) 4, and (d) 5 new epicormic shoots, comparing Part1₉₀ and Part1₁₄₅. The relationship of $E(k)$ with t_{base} is similar between the Yr90 and Yr145 Pareto optimal sets, yet the values for Yr145 Pareto optimal set tends to be higher across generation.

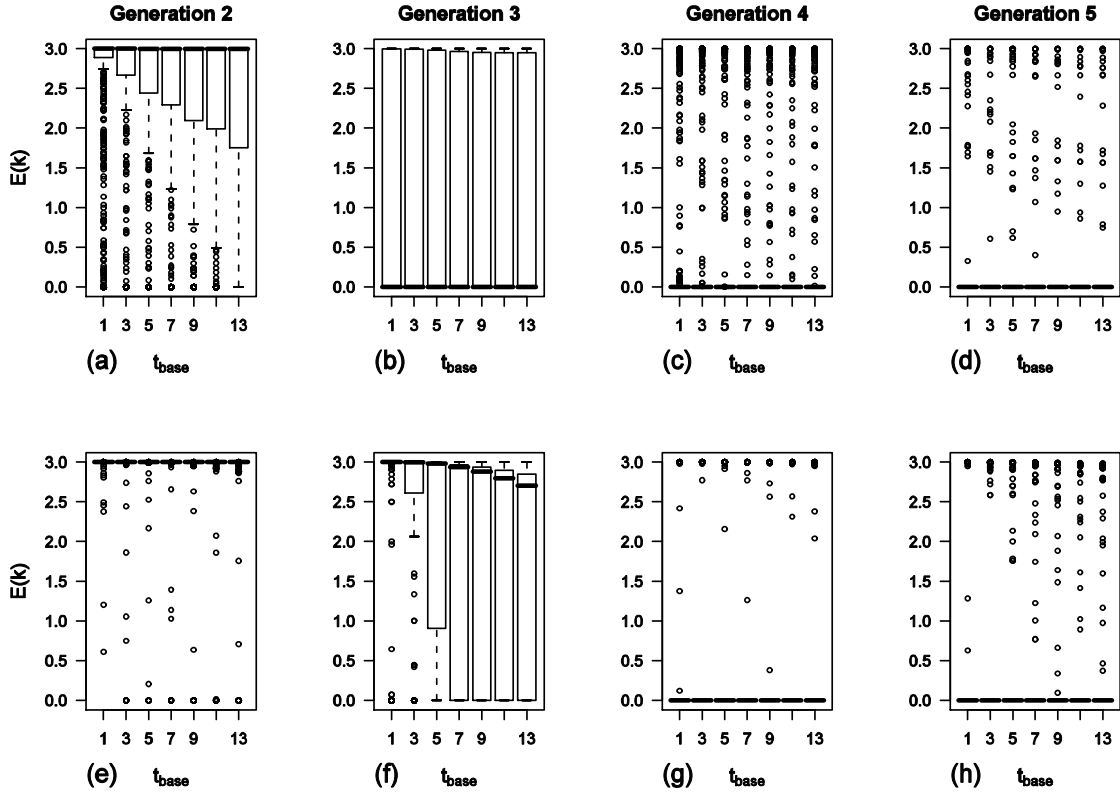


Figure 5.10. The distribution of expected number of daughter shoots ($E(k)$) with increasing t_{base} for generation (a) 2, (b) 3, (c) 4, and (d) 5 new epicormic shoots, Part2₉₀; Part2₁₄₅ generation (e) 2, (f) 3, (g) 4 and (h) 5. The pattern of $E(k)$ with increasing t_{base} is similar between Yr90 and Yr145, with a clear threshold between generation 3 and generation 4 (b,c and f,g). The main difference is that the expected value tends to be slightly higher for the Yr145 Pareto optimal set at generations 2 and 3 (e,f) compared to the Yr90 Pareto optimal set (a,b).

Simulated branches: emergent bifurcation

In order to compare the process of morphogenesis between the parameter partitions, five branches were simulated for every solution in each partition. The values of $Rb1_{sim}$, $Rb2_{sim}$ and $Rb3_{sim}$ differ markedly between the two partitions (Table 5.5). Branches simulated from Part1₁₄₅ and Part2₁₄₅ have values $Rb1_{sim}$, $Rb2_{sim}$ and $Rb3_{sim}$ similar to Part1₉₀ and Part2₉₀, respectively.

Table 5.5. Rb_{sim} values for each partition. $Rb1_{sim}$, $Rb2_{sim}$ and $Rb3_{sim}$ are simulated bifurcation values for order-1, order-2, and order-3 axes, respectively. These values are calculated for 5 branches simulated from each solution in each partition. These are the summaries of those values within each partition (Part1, Part2)

		Q1	Median	Q3
$Rb1_{sim}$	Part1 ₉₀	2.67	2.72	2.77
	Part2 ₉₀	2.52	2.88	2.94
	Part1 ₁₄₅	2.65	2.75	2.81
	Part2 ₁₄₅	2.91	2.93	2.95
$Rb2_{sim}$	Part1 ₉₀	1.49	1.57	1.67
	Part2 ₉₀	2.51	2.75	2.84
	Part1 ₁₄₅	1.41	1.48	1.73
	Part2 ₁₄₅	2.56	2.65	2.72
$Rb3_{sim}$	Part1 ₉₀	0.5	0.57	0.72
	Part2 ₉₀	0	0	0.002
	Part1 ₁₄₅	0.36	0.54	0.78
	Part2 ₁₄₅	0	0	0.001

Bud release from suppression and SCU dynamics

It is difficult to compare the distribution of timing of epicormic initiation for simulated branches to empirical data because the data of Ishii and Ford (2001) and Ishii et al. (2002) are divided into two categories of reiteration: immediate and basal. The observed timing of initiation for the data set of immediate reiteration includes epicormic shoot production from foliated axes, which in the data set reaches a maximum at 20 years. This does not exclude the possibility of epicormics initiating after longer periods of suppression, and indeed that has been observed in the form of basal reiteration (Ishii et al. 2002). To investigate basal reiteration, Ishii et al. (2002) sampled older epicormic junctions on the same branches as immediate reiteration was investigated. These two data sets are not commensurable to create a single distribution of the timing of epicormic initiation throughout the lifespan of a branch, as one was a census of younger buds and the other a sampling of older junctions. The simulated values should similarly be divided, and, as an approximate comparison, I present the timing of epicormic initiation for buds released over a term less than 20 years after formation.

To compare the process of epicormic initiation between the two partitions, a single branch is simulated for every solution in each partition, and the value of t_{bud} for all epicormics that initiated through the branch lifespan is recorded. To summarize each branch, the median value of time to release from suppression less than 20 years is recorded, and the distribution of these median values is compared between partitions and years. The median value of t_{bud} for branches in Part1 tends to be higher than the median t_{bud} for branches in Part2 (Figure 5.11). A Wilcoxon rank-sum test (performed to test whether the median value of each distribution differs between Part1 and Part2) was significant at both Yr90 and Yr145 ($p < 0.05$).

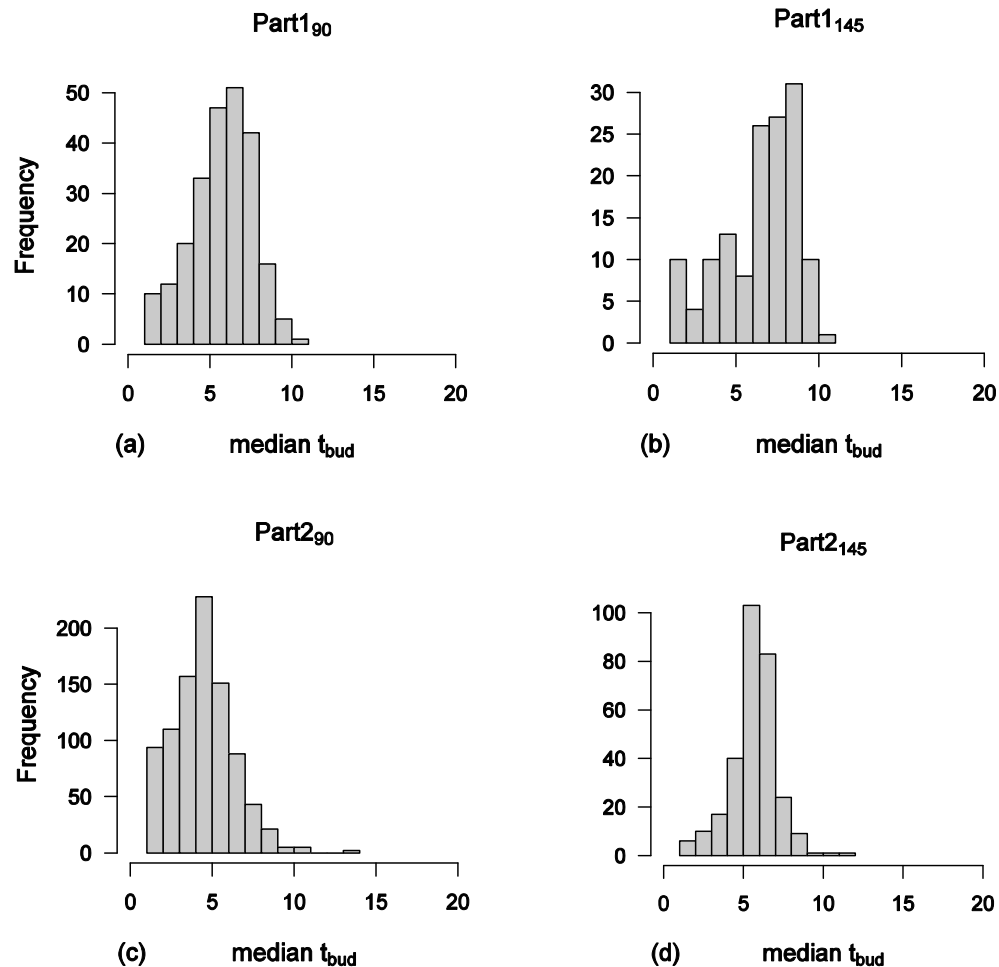


Figure 5.11. Timing of epicormic initiation for branches from each partition and year. A single branch is simulated for each solution in each partition and year and the median value of timing of epicormic initiation recorded. Each histogram is the distribution of these median values for each solution set. The distributions show that solutions in Part2 tend to have a lower median timing of epicormic initiation than solutions in Part1.

To explore the SCU dynamics through the branch lifespan, the proportion of new SCUs that developed throughout the branch lifespan relative to the SCUs that were still actively proliferating at the end of the simulation was calculated for each partition. A value below one indicates that SCUs have terminated throughout the branch lifespan, and a value above one indicates that the original generation 1 axis was still actively proliferating at Yr90. The distributions of these proportions show that, in the majority of solutions in Part2, if an SCU develops on the branch it persists throughout the branch lifespan (Figure 5.12). In contrast, for Part1 there is evident turnover of SCUs as the proportion tends to be below 1.

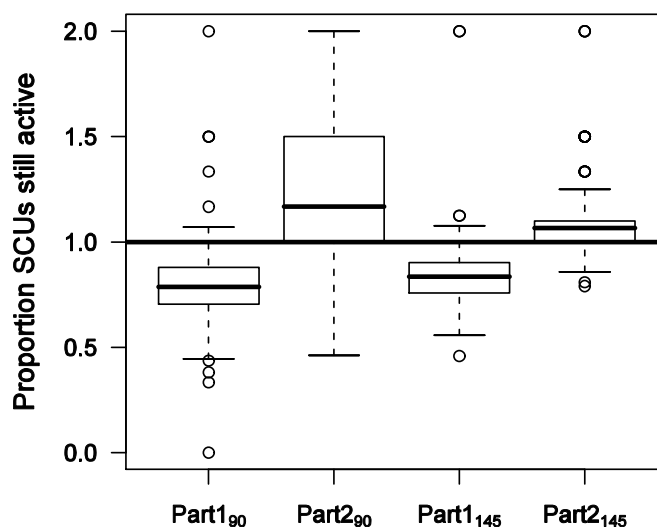


Figure 5.12. Distribution of the proportion of epicormics that develop into independent SCUs throughout the lifespan of the branch and are still actively proliferating foliage at the end of the simulation. The line at 1 shows where the number still proliferating is equal to the number that ever existed. Any number above one indicates that the original branch axis is still actively proliferating, and numbers below 1 indicate branches that exhibit SCU turnover.

Contrasting branch architectures

The pictures painted by Figures 4.6, 5.4–5.12, and Tables 5.4 and 5.5 show distinct morphologies and processes of morphogenesis between the two partitions. These distinctions demonstrate how alternative morphologies can affect relative performance with respect to the theoretical objectives. For the morphologies characteristic of Part2

(and that tend to minimize n_{turns}), there are very few SCUs; those that are present are of low generation (Table 5.4). The SCUs on branches in Part2 that do become independent survive for a very long time (Table 5.1), have very high order-1 and order-2 bifurcation, but no bifurcation on order-3 axes (Table 5.5). There is very little turnover of SCUs (Figure 5.12), and the branch pattern appears to be fairly stable. Bifurcation tends to have an all-or-nothing response to foliage overlap; at high values of n_{over} the axes do not bifurcate, whereas at low values they achieve the maximum values of bifurcation. In contrast, the morphologies characteristic of Part1 (and that tend to minimize l_{path}) have non-zero bifurcation across all three recorded shoot orders (Table 5.5), and the longevity and age of SCUs are much lower compared to Part2 morphologies (Table 5.1). In order to maintain foliage on Part1 branches, there is a higher number of SCUs that do exhibit turnover throughout the course of branch development relative to Part2 (Figure 5.12). The process of SCU development depends on the foliage overlap of active shoots (Figure 5.6; Figure 5.7), where with increasing overlap, bifurcation decreases and SCUs of higher generations fail to develop.

The analysis thus far has been for branches optimized separately at the two time intervals. It is also important to observe dynamics longitudinally throughout the branch lifespan, and compare those between the two partitions. A first explanation for the patterns observed with respect to the objective values between Yr90 optimizations and Yr145 optimizations (Figure 5.3) is that the objective values are calculated for older branches in the Yr145 optimizations. A better comparison would be to calculate values for the Yr145 optimization at Yr90 and compare those to Yr90 optimization values at Yr90. Values should also be compared for both optimizations at Yr145. Next I compare the objective performance between the two optimizations at both Yr90 and Yr145 through simulation of the respective Pareto optimal sets. Branches from the Pareto optimal set in both partitions for Yr90 and Yr145 optimizations were simulated for 145 years, and objective values recorded both at Yr90 and at Yr145. For each set, solutions were identified that produce at most 6500 shoots at Yr90 and at least 10000 shoots at

Yr145. These solutions are considered to have sufficient shoot levels at both Yr90 and Yr145 simultaneously.

Long-term simulations: shoot levels

Of the 242 solutions in Part1₉₀, only 2 have sufficient shoot levels at Yr90 and Yr145 simultaneously (Table 5.6), whereas of the 141 Part1₁₄₅ solutions, 49 have sufficient shoot levels at Yr90 and Yr145 simultaneously. The solutions in Part1₁₄₅ tend to have foliated shoot levels at Yr90 above observed values (Table 5.6), and the solutions in Part1₉₀ tend to have foliated shoot levels at Yr145 below observed values (Table 5.6). Of the 939 Part2₉₀ solutions, 75 satisfy the shoot levels at both time intervals. Of the 296 Part2₁₄₅ solutions 116 satisfy the shoot levels at both time intervals simultaneously (Table 5.6). As in Part1, the Part2₁₄₅ solutions tend to have foliated shoot levels at Yr90 above that which was observed (Table 5.6), and the Part2₉₀ solutions tend to have foliated shoot levels at Yr145 below that which was observed.

Table 5.6. Shoot levels for long-term simulations of Part1 and Part2, listing the proportion of solutions with appropriate shoot levels at both time intervals, those with greater than the minimum observed shoot levels at Yr90 and those with less than the maximum observed shoot levels at Yr145.

	PropCorr	Prop90>6500	Prop145<10000
Part1 ₉₀	0.01	0.05	0.97
Part2 ₉₀	0.08	0.06	0.86
Part1 ₁₄₅	0.35	0.29	0.37
Part2 ₁₄₅	0.39	0.23	0.22

To compare long-term growth dynamics, example branches were simulated for 500 years. For both Part1 and Part2 for Yr90 and Yr145 optimizations, a single solution was chosen at random from the set that achieves appropriate shoot levels at Yr90 and Yr145 simultaneously. For these solutions, five branches were simulated for 500 years and the long-term foliage dynamics recorded. Solutions were also chosen at random for branches that did not simultaneously achieve shoot levels at Yr90 and Yr145, and the long-term growth dynamics compared to those that do.

There are clear differences in the pattern of foliage dynamics over time between Part1 and Part2 branches, although there are no visible differences in branches that achieve shoot levels simultaneously and those that don't. For Part1, the pattern of shoot development is to increase, peak, and then decline over time (Figure 5.13). This is similar to the pattern observed by Kennedy et al. (2004). For Part2, there is a period of increasing shoot levels, then a plateau (Figure 5.13).

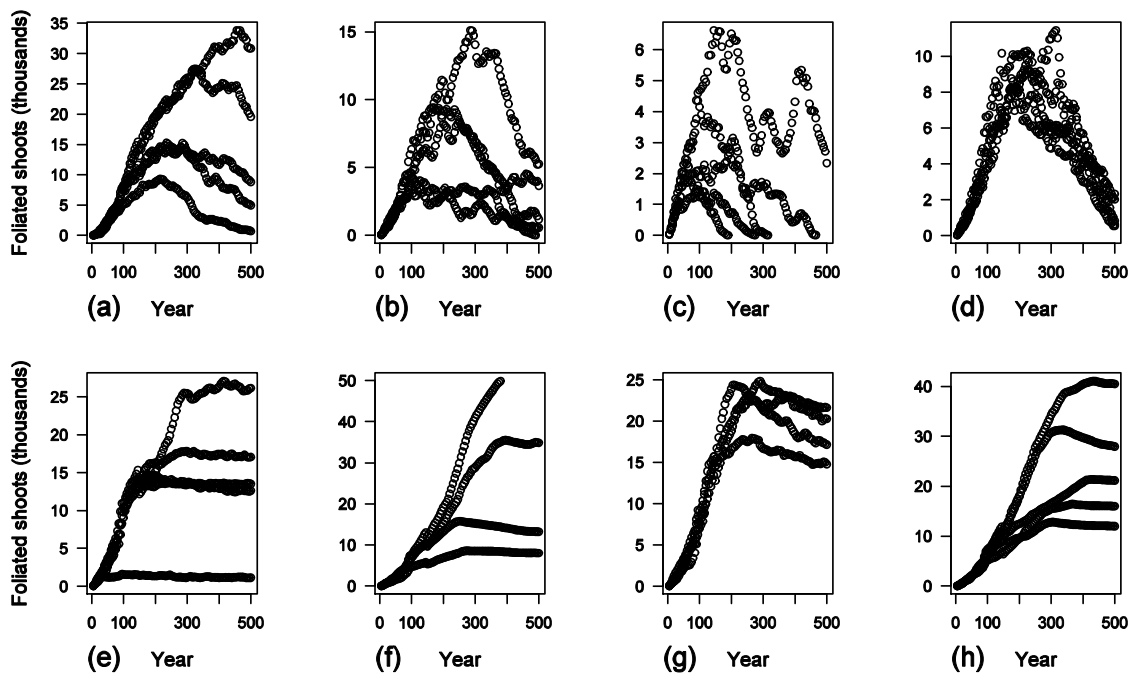


Figure 5.13. Foliated shoots over time for simulations of parameter values chosen at random from Part1 (a-d) and Part2 (e-h) Pareto optimal sets. Each plot shows the realizations of five branches simulated from a single parameter vector, and these are the same branches as for Figure 5.14(a-h). For Part 1, foliated shoots tend to increase over time, peak, then decline. For Part2, the foliated shoots tend to increase over time, then plateau.

For Part1 branches, over the course of branch development, there is an almost constant number of shoots per SCU over time (Figure 5.14). For Part2 branches, however, the foliated shoots per SCU increase, and then plateau over time in a manner similar to the shoot dynamics (Figure 5.14). In the case of the Part1 branches, the reduction in growth shows that while reiteration is prolific in these branches, it is not unlimited (Figure 5.13a-d). The general pattern of growth for the SCUs, however, is

relatively stationary through the branch lifespan. It is the population of SCUs itself that determines the fate of the branch. In the model structure, a maximum generation for SCU development emerges (g^*). As younger generations die-back, any suppressed buds are lost. Regeneration of foliage occurs on higher generation axes, which is limited with respect to SCU development. For Part2, order-1 axes have a much higher expected longevity (Table 5.1). Therefore, in Part2, growth is much less variable and foliage is continually regenerated on existing lower generation axes. If major branch axes can be maintained indefinitely, then so can the parent branch (Figure 5.13 e-h). Since there are so few SCUs, the pattern of shoots per SCU follows that of total shoots where it is the population of shoots (rather than SCUs) that determines the fate of the branch.

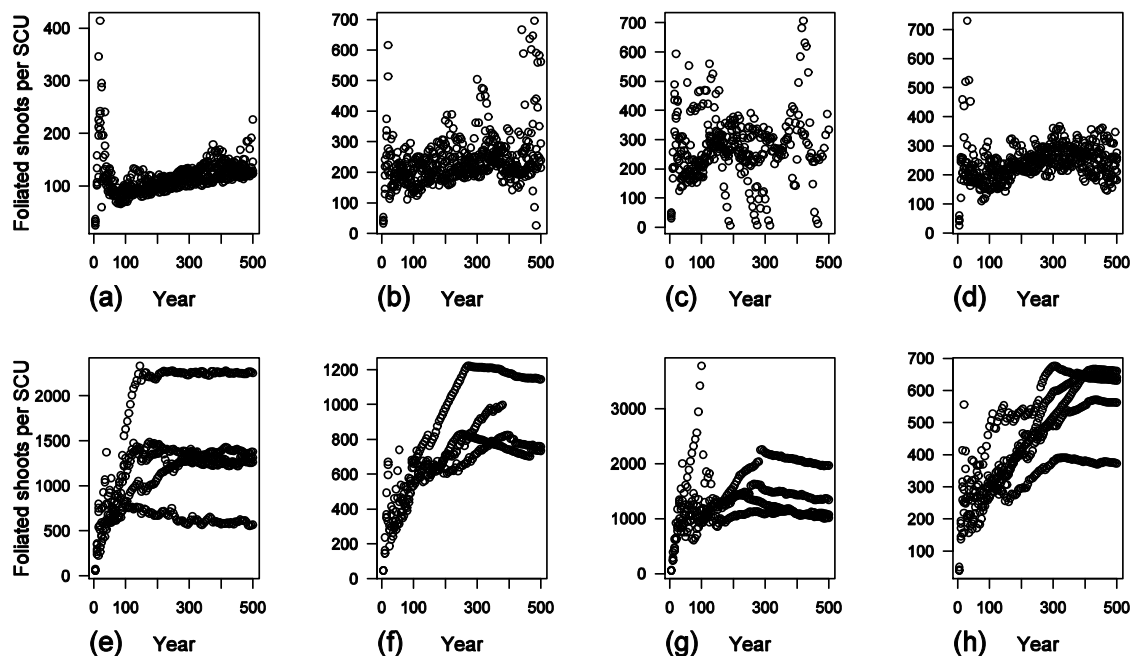


Figure 5.14. Shoots/SCU over time for simulations of parameter values chosen at random from Part1 (a-d) and Part2 (e-h) Pareto optimal sets. Each plot shows the realizations of five branches simulated from a single parameter vector, and these are the same branches as for Figure 5.13 (a-h). In Part 1, the value of shoots/SCU is relatively stable, or slightly increasing over time. In contrast, the trends over time for Part2 resemble the pattern of foliated shoots over time (Figure 5.13).

Performances with respect to the theoretical objectives are similar between the Yr90 and Yr145 optimizations at the two time intervals. This confirms that, with the exception of small differences in the parameter distributions, the overall patterns in

branch form are similar in the two optimizations within each partition and at the two time points (Yr90 and Yr145).

Question A.3 *P. menziesii* compensation

Strategy

To answer the question of which constraints the *P. menziesii* morphology compensates, we must evaluate whether the observed *P. menziesii* morphology resembles either of the major partitions. Since we have established the relative performances of the two major partitions with respect to the theoretical objectives, then if *P. menziesii* matches either of these partitions we have an understanding of the constraints for *P. menziesii*. I compare observed values of bifurcation and the observed pattern of SCU development and bud release from suppression to the two major partitions in order to evaluate where in the solution space the observed *P. menziesii* architecture falls.

Comparison to observed P. menziesii branch morphology

Kennedy et al. (2004) set default values for bifurcation of order-1, order-2, order-3 and new epicormic shoots based on the observed *P. menziesii* growth pattern (Table 2.3). The simulated values of bifurcation in Part1 tend to be near those estimated default values, and may better match the observed growth pattern of *P. menziesii* (Table 5.7); for Part2, Rb2_{sim} tends to be much higher than observed, and Rb3_{sim} tends to be close to zero (Table 5.7). Furthermore, the distribution of g^* values in Part1₉₀ is consistent with the observation that 7 is the highest generation on a *P. menziesii* branch (Ishii and Ford 2001; Table 5.7); the first quartile of g^* in Part1₉₀ is 6.8 and the third quartile is 9.1 (Table 5.4). The distribution of g^* values is shifted much lower for Part2 (Table 5.4).

Table 5.7. Summary measures of simulated branches (Part1₉₀, Part2₉₀; median values) compared to observed and/or expected values for *P. menziesii* branches.

	Empirical	Part1 ₉₀	Part2 ₉₀
Order 1	2.5	2.72	2.88
Order 2	1.5	1.57	2.75
Order 3	0.5	0.57	0
Max SCU generation	7	8.9	2.75

Ishii and Ford (2001) also provided data for the processes of bud release from suppression and the stages of SCU development. The median values in Part1 tend to be near to and above to the mean value observed (near 5 years), whereas the values in Part2 tend to be near and below that observed (Figure 5.11). This analysis is inconclusive as to whether the proposed process of bud release for either Part1 or Part2 is closer to the observed process for *P. menziesii*. For SCU development, however, Ishii and Ford (2001) describe five stages, including two stages of decline. The evident SCU turnover in Part1 (Figure 5.12) is more consistent with the observed stages of SCU development than the pattern for Part2. Upon a visual and quantitative assessment, the branch morphologies characteristic of Part1 better match the observed growth pattern of *P. menziesii* (Table 5.7). The Part2 morphologies that perform best with respect to n_{turns} result in branches that are inconsistent with the observed *P. menziesii* branch pattern. The quantitative measures of the Part1 morphologies also better match the observed pattern, including the age of the SCUs (observed near and up to 20 years by Ishii and Ford 2001), and the turnover of SCUs. Despite this turnover, the population of shoots per SCU on the branch is relatively stable through branch development, although the number of total shoots on the branch is not (Figure 5.13; Figure 5.14).

Question B.1 Relative importance of branch structural characteristics

Strategy

The answer to question B.1 requires evaluation of the parameter distributions in the Pareto optimal set. Since the evidence from model results indicates that Part1 more closely resembles the observed *P. menziesii* branch morphology, then those parameter values will be investigated in great detail. Contrasting the values between Part1 and Part2 will also be valuable to highlight the relative importance of each variable in producing the alternative morphologies. These are considered in the context of the component postulates for the growth functions. See Discussion in this Chapter for the presentation of these postulates.

Discussion

The results presented in this Chapter allow us to propose answers to the questions posed in Chapter III and repeated at the beginning of this Chapter. The question that motivates this study is:

How does the branch development pattern of old-growth *P. menziesii* at the WRCCRF compensate for size-related constraints on the species?

In order to answer this question, we must first give a satisfactory discussion of the four sub-questions.

A.1. What are the major system constraints?

In the modeling analysis presented here, four processes are proposed to be constraining in the system; two of these fall under the category of hydraulic constraint. There is evidence in this analysis that each of these theoretical objectives play a role in the emergent branch pattern. Niklas (1992, p 20) asserts that, in systems analysis for a design with multiple objectives, it is usually a single objective that dominates the major design specification, and then within that other objectives may influence design details. In the optimization results, two major branch patterns emerge, each of which is dominated by a different theoretical objective. The branch pattern that tends to minimize l_{path} resembles closely the observed *P. menziesii* branch pattern, whereas the branch pattern that tends to minimize n_{turns} possibly resembles a more determinate branch pattern such as observed in the true firs (Figure 4.6). For example, Kennedy et al. (2004) also simulated branch development in *Abies grandis* at the WRCCRF, and the overall branching pattern and values of bifurcation resemble those observed in the n_{turns} partition, although without epicormic proliferation (Figure 5.15). Therefore the evidence presented here supports the hydraulic constraint as the objective that dominates the design specification in this old-growth system; the two major designs compensate for two separate possible causes of that hydraulic constraint.

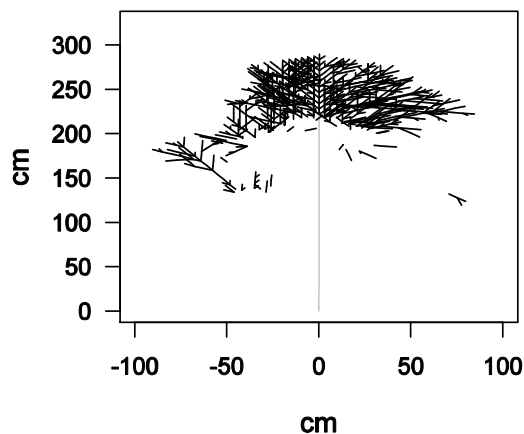


Figure 5.15. *Abies grandis* branch map, simulated with BRANCHPRO and pictured at Yr40. Although this branch exhibits zero reiteration and generation of epicormic shoots, its basic architecture and shape are reminiscent of the branch forms that emerge in Part2, which tend to minimize n_{turns} .

Within those two major partitions of the solution space, other tradeoffs emerge that clearly influence the design details (Table 5.3). If the number of SCUs on the branch can be considered a measure of the degree of reiteration, then performance in the theoretical objectives across the numbers of SCUs illuminates possible tradeoffs within the solution space. Overall, there is an opposite relationship between n_{turns} and SCUs and the remaining three theoretical objectives with SCUs (Table 5.3). In Part1, n_{turns} tends to decrease with SCUs, with the other theoretical objectives tending to increase. In Part2, n_{turns} tends to increase with SCUs and the other objectives tend to decrease. This is also reflected when contrasting low SCUs with target SCUs regardless of the partition (Table 5.3). Within the possible dominant l_{path} constraint, the intermediate values of SCUs observed in *P. menziesii* may be a compromise between the performance with respect to n_{turns} and with the remaining objectives. These emergent patterns provide evidence that the value of n_{over} and load are also constraining in the system, with more evidence for n_{over} . These are not, however, the dominant objectives in the old-growth branches. This gives evidence that foliage display in this case is secondary to hydraulic functioning.

A.2. Of the architectures possible in the model structure, which architecture (branch development pattern) best compensates for the constraints?

It is clear that the answer to this question depends on which objective is considered. The two emergent branch patterns represent distinct strategies of branch development. These strategies are reminiscent of the guerrilla/phalanx spectrum of growth forms described by Doust (1981) for clonal organisms. In the guerrilla strategy, organisms infiltrate the surrounding environment through an opportunistic strategy of rapid spread and sampling of the environment. The phalanx strategy is more conservative, wherein the organism presents a tightly packed advancing front that excludes other plants. The guerrilla species exhibit effective exploitation of favorable patches when encountered (deKroon and Hutchings 1995). A species of the phalanx type is expected to show less plastic response in growth development and morphology (deKroon and Knops 1990). Within the context of the process of branch development as given by the structure of BRANCHPRO3, there are indications of the spectrum of the phalanx and guerrilla growth strategies.

In Part1, the process of reiteration dominates the emergent branch pattern, as shown by the level of SCU development in the branches. In the Part1₉₀ solution space, 59% of the solutions reach at least the minimum number of SCUs observed, whereas in the Part2₉₀ solution space only 17% of solutions reach at least the minimum number of SCUs observed. Reiteration is considered a sign of opportunistic architecture (Halle et al. 1978, Chapter 4; Tomlinson 1983; Begin and Filion 1999), wherein a reserve of buds responds to local conditions and exploits patches of good growth conditions. This is one way of realizing the guerrilla strategy described above. In the context of the old-growth branch environment, the sampling of the environment occurs within the crown because the outward crown expansion is minimal. As the process of reiteration is characterized in the model structure, the reserve of buds represents potential sampling points for the within-crown environment. In the model, bud release depends only on internal conditions including hydraulic status and activity of growing meristems. When a bud is released, the resulting shoot “samples” the environment, and its further proliferation is

regulated accordingly. This is evident in the rapid decrease of g^* with increasing n_{over} (Figure 5.6) and the continual decrease in expected number of daughters with increasing n_{over} (Figure 5.7).

In the optimization, the opportunistic strategy exhibited by Part1 tends to minimize l_{path} at the expense of n_{turns} (when compared to Part2). The major SCU axes have limited life spans, and foliage is continually regenerated proximally along the major branching axis. There is a constant turnover of both foliage and SCUs on the branch. In addition, at higher levels of reiteration this strategy sacrifices the objective of foliage display insofar as the mean number of overlapping foliated shoots increases with the level of SCU development (Table 5.3). The values of g^* in this partition tend to be near or higher than the maximum generation observed by Ishii and Ford (2001), which is consistent with the *P. menziesii* growth pattern. In this context, reiteration, in terms of the accumulation of generations, is limited, but not as limited as in Part2.

In contrast, the major branch pattern of Part2 exhibits limited reiteration and more closely resembles a phalanx growth strategy (Doust 1981). In this growth pattern a plant fills its space and sits there within it, excluding other plants from occupying the same space. It does not actively forage for resources by extending its growth increments differentially through patches of variable resource quality. In Part2, values of important features such as g^* do not change much with increasing n_{over} (Figure 5.6). This is consistent with a strategy that does not preferentially exploit areas of high light. The branches in this partition have no evident turnover of SCUs (Figure 5.12), and have very long-lived branch axes (Table 5.1). As a consequence, they have very low n_{turns} , but they have very long l_{path} (Figure 5.3). Overall the value of g^* is very low for these branches, which is further evidence for their limited reiteration. This is qualitatively similar to observed *A. grandis* branches, which exhibited much less development of epicormic shoots compared to *P. menziesii* at the WRCCRF (Kennedy 2002; Kennedy et al. 2004).

- A.3. For which constraints (if any) does the observed *P. menziesii* branch development pattern compensate?

The answer to this question depends on where in the solution space the observed *P. menziesii* architecture falls. The simplest answer is that Part1, which tends to minimize l_{path} , has a higher proportion of solutions that satisfy all four empirical objectives, whereas Part2 has a very low proportion of solutions that satisfy the SCU objective in particular. There is further evidence that the *P. menziesii* architecture belongs in Part1, including the continuous regeneration of foliage and SCU turnover. The characteristic of this partition as an opportunistic architecture is also consistent with the conclusions of Ishii and Ford (2001) with respect to the consequences of the observed levels of reiteration in *P. menziesii*. Given that the *P. menziesii* architecture conforms to that observed in Part1, there is evidence that *P. menziesii* morphology compensates for hydraulic constraint by lowering the mean path length to terminal active nodes. Within this partition, the number of turns actually decreases with increasing SCU levels, whereas foliage overlap increases. There is clearly a tradeoff between these two objectives within the Part1 space, and the observed levels of SCUs on *P. menziesii* branches may represent a compromise between foliage display and number of cumulative junctions.

- B.1. What is the relative importance of different branch structural characters (e.g., architectural order, foliage overlap, nearby growth) in determining the observed branch development pattern in *P. menziesii*?

The component postulates for the growth functions must be considered in order to answer question B.1. I discuss each postulate in turn. These are not intended as alternative either/or explanations, and evidence in favor of any individual postulate does not negate remaining propositions. Rather, these should be taken as individual pieces, that when considered together form a whole theory for the process of branch development. I evaluate the evidence in favor or against each postulate with respect to the pattern of convergence for the parameter values in the context of the model structure and objectives.

Postulate p.1:

The probability of suppressed bud release along order-1 axes of *P. menziesii* increases as the age of the bud increases.

The results of the optimization searches clearly falsify this postulate, in that the value of p_{bud} converges below zero in all parameter partitions and in both Yr90 and Yr145 optimizations in Series 4 and 5 (Figure 5.5b). In optimization Series 3 (Chapter IV), p_{bud} consistently converged to a value of zero; of 296 solutions, 180 had a p_{bud} value equal to zero. All of the remaining parameters had their first quartile above zero, which indicates that they converged consistently above zero. The evidence from these optimization studies shows that the probability of suppressed bud release *decreases* as the age of the bud increases (Figure 5.5b). This is consistent with one of the assumptions of the r function, in that proliferation of a new epicormic would decrease as the age of the bud that originated the new epicormic increases. These results imply that the longer an individual bud is suppressed the less likely it is to break from suppression. The underlying cause of this pattern cannot be the distance to nearby foliage, as this would increase with increasing t_{bud} . It could be increased hydraulic constriction as the bud is suppressed. It might also be an artifact of the optimization procedure, in that this is the only means to reduce the bud bank; otherwise all suppressed buds would have a positive probability to be released.

Postulate p.2

The probability of bud release from suppression increases as the proportion of the immediate axes that are no longer proliferating increases.

The optimization results lend evidence in favor of this postulate, as the value of p_a consistently converges above zero (Figure 5.5c). This supports the idea that, given the combination of model structure, parameters and objectives, a local slowing of growth might increase the probability of bud release.

Postulate p.3

The probability of bud release from suppression increases with decreasing distance to the main stem.

There is also evidence from the optimization results in support of this postulate. The value of p_{stem} converges consistently above zero (Figure 5.5e). This result, however, may be a consequence of the model structure and parameters in the context of the optimization objectives. This parameter is most related to the position of new epicormics relative to the main stem of the tree, a pattern that is inconsistent with observed trends in *P. menziesii*. A greater proportion of reiterated axes appear more proximally on the branch in simulations than on the observed branches (Figure 5.5e). This would obviously reduce l_{path} to terminal foliated shoots as well as the load parameter as it is calculated. Given that this result emerges in the optimization, there is likely a missing factor in the model and optimization that would prevent such proximal development of epicormics and SCUs in observed branches of *P. menziesii* at the WRCCRF.

A possibility would be the autonomy of branches simulated in BRANCHPRO3. These branches are simulated in isolation, with no accounting for between-branch shading. Ishii and Wilson (2001) report that relative PAR declines within old-growth *P. menziesii* crowns from the upper to the lower crown. Parker et al. (2002) show similar results in a decline in transmittance from the upper to middle and lower crowns in the old-growth WRCCRF forest. They did not record light levels horizontally along branch axes, yet their results indicate that the inner part of the crown for older branches is likely strongly shaded. Such a light environment might limit the development of SCUs proximally along the branch near the main stem. It is possible that if such a factor were included in the model structure, then the effect of p_{stem} would be smaller. A simple fix would be to assign any foliated shoot some relative distance from the main stem a baseline positive value of overlap regardless of the position of other foliated shoots on the same branch. This would especially affect the phalanx-like growth structure, which is shown here to respond quite strongly to an increase in overlap (Figure 5.7; Figure 5.8)

Postulate p.4

The probability of bud release from suppression increases as the distance to the nearest developing SCU on the same axis increases.

The pattern of convergence for p_{SCU} well above zero gives evidence in support of this postulate (Figure 5.5d). As with the convergence of p_{inact} , this implies that active local growth may serve to inhibit the suppressed buds.

Postulate r.1:

The architectural hierarchy of *P. menziesii*, as defined by the shoot order, decreases the bifurcation rate in active nodes with increasing order.

There is clearly a negative effect of order (Figure 5.4b) that yields the observed basic SCU architectural growth pattern in *P. menziesii*. This is not unexpected, as apical control is a well-documented phenomenon in this system. However, there is a limitation in the model structure that is not consistent with the observed branch development pattern. As a process rule, all SCUs in the simulation model originate from epicormic initiation on order-1 axes. However, Ishii et al. (2002) observed some SCU and epicormic development on higher ordered axes, including sequential reiteration that might not be solely traumatic. The strict hierarchy that is enforced in this model does not completely capture the observed growth pattern. Some other factor may decrease the apical control on some branch axes, including possibly a positive effect of light interception on bifurcation of axes regardless of order. Such an effect would mitigate the strict apical control that this postulate represents, although it would not dismiss it. Furthermore, for the branches in Part2, the change in bifurcation with order does not materialize in the expected manner. The expected numbers of shoots for order 1 and order 2 are nearly identical, then order 3 does not bifurcate at all (Table 5.1). This is clearly not consistent with a constant decrease in bifurcation with increasing order.

Postulate r.2:

A more severe hydraulic constriction forms at branch junctions when a bud is suppressed beyond 1 year relative to those only suppressed 1 year within the usual

course of growth. This constriction diminishes as the axis expands beyond the point of constriction (the distance in cm to the constriction).

Postulate r.2a

This constriction increases as the number of successive junctions that result from bud release from suppression accumulates to an active node.

The strongest effect in the pattern of convergence for the optimization is the consistently negative convergence of r_{gen} (Figure 5.4.c). This value controls the maximum generation at which new SCU development can occur. It also accomplishes, at some level, the limitation of reiteration at higher generations. Some clarification is required, however, since the expected value of bifurcation that emerges in simulations for a generation-2 or 3 new-epicormic shoot with no overlap is well above that which was observed by Kennedy (2002) and Kennedy et al. (2004). The expected bifurcation of new epicormic shoots nears the observed values only at higher generations (Figure 5.9). The data set of new epicormic growth that is presented by Kennedy et al. (2004) does not record the generation of the new epicormic axes they observed. The mean number of daughter shoots observed by Kennedy et al. (2004) was 1 for the new epicormic shoot, although this would be higher than that recorded in the simulation because the observed data set excludes zeroes. In the model and optimization results, there is some evidence that bifurcation of new epicormic shoots decreases with increasing generation, but given the model structure this may be an artifact of the need to limit the generations of reiteration that can accumulate on the branch. It does not, however, accomplish the limited bifurcation observed on even lower-generation new epicormic shoots. It is possible that all new epicormic shoots, regardless of generation, could have the same hydraulic constriction and that the maximum observed generation on the branches is due to some other threshold effect. There is no capacity in the current model structure to support this statement, and no biological theory that could explain the generation threshold. There is only evidence in the simulation that such a threshold exists, otherwise given the model structure there would be no limitation on the number of generations of reiteration on the branch.

Postulate r.2a

This constriction increases as the time the bud is suppressed increases.

The pattern of convergence in the optimization series also gives evidence in support of this postulate. The value of r_{base} consistently converges below zero (Figure 5.4e), and it effectively reduces the expected value of bifurcation of new epicormic shoots as well as the maximum generation at which SCU development can occur. This gives evidence in favor of increasing hydraulic constriction each year a bud is suppressed.

Postulate r.4

The bifurcation rate in active nodes of *P. menziesii* decreases as the number of overlapping foliated shoots at that node increases.

There is clear evidence in support of this postulate, particularly in the development of new SCUs in branches that are similar to *P. menziesii*. In Part1, the expected number of daughter shoots declines steadily with n_{over} (Figure 5.5d; Figure 5.7) for each shoot order. The value of g^* also declines with n_{over} , which indicates that overlap in part controls the placement of SCUs on the branch.

Given the analysis of each postulate, not one of the independent variables can be dismissed as a determinant of the branch development pattern in *P. menziesii*. For the bifurcation function, all of the parameter values show significant effects on the pattern of branching and clearly differentiate between the two major branch partitions.

For example, the effect of order differs between the two partitions. In Part2, the bifurcation decreases so steeply with order that order-3 shoots tend not to bifurcate (although order 1 and order 2 have similar values). In contrast, for Part1 the decline is less steep and order-3 shoots exhibit low levels of bifurcation (Figure 5.7). The values of generation, n_{over} and t_{base} determine whether new epicormic shoots proliferate successfully into independent SCUs, and it is clear that the value of r_{gen} is a key determinant of the level of reiteration on the branch. Within that, the value of r_{base} is

such that t_{base} also has an effect on whether the new epicormic proliferates successfully (Figure 5.9). This lends evidence that the longer the bud is suppressed, the less likely it is to form a new SCU. Furthermore, in this opportunistic architecture, the value of n_{over} also has a clear effect on whether a new epicormic proliferates into an independent SCU, which has profound consequences on the resulting branch morphology (Figure 5.6). The expected number of daughter shoots for a given node clearly also changes with n_{over} , and the response is evident at low values of n_{over} (Figure 5.7). Therefore, all four of the proposed independent biological variables have evident effect on the observed branch development pattern in *P. menziesii*.

The effects of the independent variables in the p function are a little more difficult to determine. Although it is clear by the parameter values that there are effects for each of the variables, the associated parameters do not show any particularly obvious pattern across the levels of SCUs in the solution set. They do, however, show differences between Part1 and Part2, particularly in the values of p_0 , p_{stem} and p_{SCU} . The values of p_0 clearly tend to be higher for Part2 (Figure 5.5a), which raises the baseline probability of initiation in Part2 and explains the lower values in the distribution of timing of epicormic initiation for branches simulated from Part2 (Figure 5.11). This relegates much of the epicormic growth to be almost equivalent to regular lateral growth in the Part2, and is evidence of the low capacity for proleptic reiteration on those branches. The value of p_{stem} is highly right-skewed in Part2, with the median value close to zero. In Part1 the value of p_{stem} is tightly distributed near 0.07, well above the values in Part2. These relationships are confounded by the Yr145 results, which show an opposite pattern (Figure 5.5e). The high value of p_{stem} for Part1₉₀ relates to the issue of consistent reiteration on branches in Part1₉₀ that occur more proximally than was observed in *P. menziesii* by giving higher probabilities to suppressed buds nearest to the main stem. The value of p_{SCU} is similarly right-skewed in Part2₉₀, but the mode is more positive than for p_{stem} . The median value of p_{SCU} is higher in Part1₉₀ than in Part2₉₀. As with p_{stem} , there is an opposite relationship in the distributions of p_{SCU} for Part1₁₄₅ and Part2₁₄₅. These differences indicate that the values of p_0 , p_{SCU} and p_{stem} all contribute to the different

morphologies exhibited in the two partitions. The values of p_{bud} and p_a do not tend to differ between Part1 and Part2, which indicates that they may contribute less to the overall morphology that is exhibited by the two partitions.

1. How does the branch development pattern of old-growth *P. menziesii* at the WRCCRF compensate for size constraints on the species at that location?

Now we are prepared to answer this question in the context of the optimization study. The results presented here provide evidence that *P. menziesii* morphology resembles the morphology that emerges in Part1. It exhibits a guerrilla-like growth form, which is accomplished through the preferential production of SCUs in areas of low within-branch overlap (Figure 5.6a,c). Reiteration is thereby an opportunistic process that dynamically exploits area of high light and exhibits a high degree of turnover relative to the Part2 morphology. The *P. menziesii* morphology compensates in particular for the hydraulic constraint by reducing mean path length relative to other possible branch morphologies, and when that is accomplished it exhibits a degree of reiteration that is a compromise between the number of turns and foliage overlap (Table 5.3). This is accomplished through the continual regeneration of foliage through consistent turnover of SCUs proximally along existing branch axes. These conclusions are discussed in the larger model context in Chapter VI.

Chapter VI

Multi-objective optimization for process models allows for the synthesis of multiple theories that can guide further empirical investigation

The model of plant form and function presented here demonstrates how, through the synthesis of multiple phenomena, multi-objective optimization for process models can be a key tool in theory development for ecological systems. When such a model is considered, its conclusions are bounded by the context of the model structure, parameters and objectives.

Multi-objective optimization, Pareto optimality and theory development

Multi-objective optimization

The course of ecological research for a particular program often follows a trajectory set by the original theoretical abstraction and the determination, prior to empirical insight, of the "dynamically relevant components" of the system (Schmitz 2001). This leads to research programs that espouse theories that become competing explanations for a common observation, and that are investigated in isolation with attempts to control other factors (are you in the hydraulic limitation camp, or the respiration hypothesis camp?). Yet, when a theory is tested, it is rarely rejected outright, nor is there conclusive evidence in support of one explanation at the exclusion of others (e.g., Ryan et al. 2006; Niklas 2007). In ecological systems, multiple phenomena interact and system behavior can rarely be explained by a single characteristic (Franklin et al. 1987). Schmitz (2000; 2001) suggests utilizing individual-based models and simulation experiments, formulated from natural history knowledge before extensive empirical research is conducted, to identify the dynamically relevant system components that then set the course for empirical research. In order to identify these components, or to reconcile competing theories, a synthesis of multiple, interacting phenomena is necessary.

This issue is also relevant to optimality studies. When the individual theory is a proposed function for which the organism is insisted to be optimal, treatment of the

theory as a mutually exclusive alternative to other such functions is unacceptable. Rather, if the multiple functions are recognizable features known to be limiting in the system, then the more likely explanation is that they influence the system simultaneously, and they must be treated as such in the analysis. The paradigm for optimality theory must be shifted from the current consideration of optimization of a single value, even if that value incorporates multiple features. The new paradigm should recognize that multiple features act as constraints and that the observed system is a consequence of these acting simultaneously. The individual system represents one among many possible solutions to the requirements imposed by the organism and the constraints (Niklas 1999). The synthesis of these multiple phenomena is achieved through multi-objective optimization of process models. The growth constraints evident in old-growth trees are good examples of multiple features that act simultaneously on an ecological system, and that should be integrated in theory development.

Pareto optimality

The integration of the model with optimization of a vector objective function allows for a contrast of optimal branch morphologies, and compares those to morphologies that match the observed growth pattern. In previous modeling exercises, tradeoffs are uncovered when single objectives are optimized separately (Fisher and Honda 1977; Honda and Fisher 1978; Honda 1978; Fisher and Honda 1979; Honda and Fisher 1979), but this results in non-characteristic plant forms. In contrast, multiple objectives have been combined into a scalar-valued function and, when this function is optimized, multiple forms are found to be optimal (Niklas and Kerchner 1984; Niklas 1997a,b; Niklas 1999); however, the relative contribution of the multiple objectives to each of the forms is difficult to unravel in this case. The optimal space that they compute is a subspace of the Pareto optimal frontier, which includes all weighted combinations of objective values.

Pareto optimality also allows for comparison of alternative Pareto optimal solutions that may not be consistent with the observed system of interest. In the Pareto

optimal frontier the theoretical objectives are optimized regardless of performance with respect to the empirical objectives that are optimized with binary errors.

Satisfaction of one or more of the empirical objectives would precipitate inclusion in the optimal frontier of a solution that may not perform as well with respect to the theoretical objectives (Figure 3.1). These create a partition in the Pareto optimal frontier of solutions that satisfy the empirical objectives, and these may or may not be Pareto optimal with respect to the theoretical objectives. This illustrates how the empirical objectives create a partition in the Pareto optimal frontier that may overlap with the theoretical Pareto optimal frontier, and further supports the value of using Pareto optimality in the optimization. I have utilized this in theory development for branch development in old-growth *P. menziesii*

Theory development

There are several proposed explanations for why net growth declines as a forest stand ages. These include a shift to maintenance respiration that reduces the NPP of a forest, hydraulic limitation that reduces photosynthesis with reduced stomatal conductance, and nutrient limitation in the soils of older forests, which reduces photosynthesis (Ryan and Yoder 1997; Hubbard et al. 1999; Bond 2000; Hubbard et al. 2002; Ryan et al. 2004; Ryan et al. 2006). Regardless of the constraints, compensation for them is an observed phenomenon (McDowell et al. 2002a,b) that must be integrated with these explanations of the growth patterns. Ishii et al. (2007) review how proleptic reiteration in *P. menziesii* (and the *P. menziesii* branch morphology) may compensate for hypothesized constraints in old-growth systems under 4 major categories. (1) It may compensate for an increase in the respiration/photosynthesis ratio, (2) reduce hydraulic limitation, (3) restrict nutrient limitation, (4) and mitigate genetically programmed senescence through the proliferation of younger tissue. Any one or combination of these alternatives may explain how and if the observed *P. menziesii* old-growth morphology compensates for the evident size constraint.

The evidence provided by BRANCHPRO3 and associated objectives support the supposition that *P. menziesii* branch morphology may compensate for hydraulic limitation in the form of reduced path length, and that the level of reiteration is a compromise between foliage display (which would increase photosynthesis) and branch complexity measured by mean number of turns (which could be a further hydraulic limitation). This observation is relative to a second morphology that exhibits much less reiteration, yet emerges in the Pareto optimal set (Part2; Figure 4.6). *P. menziesii* achieves these compensations through the process of proleptic reiteration, which includes bud release from suppression and subsequent bifurcation of active shoots. The spatial placement of SCUs is controlled first by bud release from suppression, then by the condition of the new epicormic shoot when the bud is released. If the shoot is of high generation, or if the shoot is shaded, then it will not proliferate further. Path length is reduced by the reiteration of foliage proximally on the branch axis, and the turnover (dieback) of groups of foliage organized on the SCUs. This demonstrates that no single factor is responsible for the observed growth pattern, and the observed pattern is a compromise among the constraints quantified for the system. This conclusion would not have been possible without multi-objective optimization and Pareto optimality.

In order to synthesize multiple phenomena that are likely acting in a system, these should be optimized as a vector objective function with Pareto optimality. In this example, if only the hydraulic constraint was optimized as the value of n_{turns} , then a single branch pattern would have emerged to answer the question of what pattern best compensates for the size constraint. The analysis presented demonstrates that there is not a single morphology that compensates for size constraints when multiple constraints are included in the optimization. A single objective optimization would have yielded an inadequate representation of the range of morphologies that compensate for constraints in the old-growth system, and such an analysis could not have uncovered the trade-offs that are evident in the morphologies and objectives considered in the multi-objective optimization. The presence of these trade-offs in the optimization results gives evidence that multiple phenomena are acting in the observed system, and any viable theory must

synthesize those phenomena. The theory proposed through the optimization analysis presented here is one of the process of morphogenesis in old-growth *P. menziesii* branches and the constraints for which the morphology compensates.

Proposed theory:

There are three major components of the proposed theory of *P. menziesii* branch morphogenesis: (1) morphological compensation; (3) bud release from suppression; and (2) SCU development and morphogenesis. The *P. menziesii* morphology compensates for size constraints by: (1a) limiting path length to terminal active foliage; (1b) regulating its level of reiteration by the trade-off between the number of cumulative junctions between the main stem and terminal active foliage and by the within-branch foliage overlap. Bud release from suppression is regulated by: (2a) the activity of locally proliferating axes, both sequential and proleptic. The process of SCU development and morphogenesis is regulated by: (3a) the demographics of suppressed buds; (3b) the bifurcation of active terminal nodes. Below I provide a detailed description of each of these theory components.

Morphological compensation (1)

(1a) Limits path length to terminal active foliage

In the old-growth stage of development for *P. menziesii*, the primary challenge is persistence since the dominant trees have attained their maximum height and crown width and have escaped most competition (Ishii and Ford 2002). The series of optimizations conducted here show that growth patterns that tend to minimize hydraulic path length (l_{path}) result in branches that more closely mimic the observed growth pattern (Table 5.4; Table 5.5; Figure 4.6). Niklas (1992, p 20) asserts that, in systems analysis for a design with multiple objectives, it is usually a single objective that dominates the major design specification, and then within that other objectives may influence design details. He applies this as a possibility for general evolutionary processes, and it may also apply to plant design and architecture. In the optimization exercise presented here, it is evident that l_{path} is a dominant constraint that acts as a limiting factor in the

morphological pattern of the old-growth *P. menziesii* branches, when the primary challenge is persistence in the system.

(1b) Tradeoff between number of cumulative junctions and foliage overlap

This does not, however, eliminate the other objectives as constraints that contribute to the observed growth. Two of the other three theoretical objectives (n_{turns} , n_{over}) also represent tradeoffs in the number of SCUs on the branch. The value of n_{turns} tends to decrease with SCUs. In contrast, the value of n_{over} increases with increasing SCUs (Table 5.3). Within the bounds of the model and these objectives, the observed number of SCUs may be a compromise for acceptable levels of n_{turns} and n_{over} , or perhaps n_{turns} is minimized within an acceptable level of n_{over} . Therefore these are not optimization objectives with respect to the minimum values observed, but still serve as constraints on branch form for which the dominant pattern compensates.

In the parameter partition with morphologies that are non-characteristic with respect to *P. menziesii* (Part2), the relationship between the theoretical objectives and the SCU level is different than that in the characteristic parameter partition (Part1; Table 5.3). In Part2, the majority of solutions have too few SCUs. If the morphology exhibited by branches in Part2 is considered to be similar to old-growth species in the system that do not exhibit adaptive proleptic reiteration, then there are two alternative morphologies that compensate for the system constraints. The form of the compensation is dependent on the underlying morphology and the dominant constraint. This should also be integrated with the physiological characteristics of each species. In particular, the differences in the morphological compensations may be accounted for by the type of underlying physiological compensations in the species, or how the constraints manifest in the process of branch development. For example, it may be that a species that exhibits a Part2 morphology has a more severe hydraulic constriction at sequential junctions than *P. menziesii*, which would correspond with a tendency to minimize those junctions. These considerations should guide further empirical investigations, which are outlined later in this Chapter.

Bud release from suppression (2)

(2a) Regulated by the activity of local axes

The model structure and optimization results present a theory for the process by which the *P. menziesii* branch morphology emerges. Buds are suppressed along order-1 axes, and these have little to no connection to the hydraulic architecture of the main SCU axis. A slowing of growth relative to the suppressed bud increases its probability of release, which indicates that active growth serves to suppress proliferation of the buds. This may be a hormonal interaction, perhaps similar to apical control exhibited by the main axis (Chapter II). There is also evidence that the probability decreases the longer the bud is suppressed. It is not clear how this can be interpreted biologically, unless something in the limited growth of the bud each year it is suppressed further removes it from the hydraulic architecture of the main SCU axis (such as bark thickness).

SCU development and morphogenesis (3)

(3a) Controlled by the demographics of suppressed buds

The two partitions allow for a contrast to be made which allows for interpretation of the parameters relative to the two dominant morphologies. The role of bud release from suppression in producing the emergent branch morphology can be ascertained by the relative values of the p function parameters in the two major partitions. In particular, the value of p_0 tends to be larger for Part2, and p_{stem} tends to be smaller (Figure 5.5). The difference in p_0 values would control the timing of epicormic initiation for buds suppressed for short time, in that this is the baseline probability of release from suppression. This explains why so many younger buds are released in Part2 branches (Figure 5.11). The higher value of p_{stem} in Part1 would result in more buds released proximally along the branch. The pattern of SCUs on the branch is first controlled by the pattern of buds released from suppression. The pattern is then determined by the successful proliferation of the new axis.

(3b) Controlled by the bifurcation of active terminal nodes

Once a new epicormic axis is formed by bud release from suppression, the process of bifurcation for the new epicormic shoot is controlled by the order-1 bifurcation. This is reduced by the generation of the shoot, the time since formation of the bud when it was released, and the foliage overlap of the shoot (Figure 5.7; Figure 5.8; Figure 5.9; eqn 4.10). If the bifurcation value that is calculated from these variables is very low, then it is unlikely that the SCU would develop fully. If the new axis reaches a certain distance from the junction (measured by d_{base}), the proliferation is controlled by the bifurcation of each shoot order, which is reduced by the overlap of each shoot. This effect clearly differentiates the two dominant morphologies, where the expected number of daughter shoots in the characteristic Part1 morphology declines steadily with n_{over} and order (Figure 5.7). The Part2 morphology seems to have a more “all or nothing” morphology, where the bifurcation tends to be either 3 or zero with order and n_{over} (Figure 5.7). The consequence of the process of bifurcation in Part1 is that some SCUs terminate over the Yr90 lifespan of the branch. This results in evident SCU turnover that is not exhibited by Part2 morphology, but was observed for *P. menziesii* branches (Ishii and Ford 2001).

Longevity in *P. menziesii*

Extended longevity of trees is attributed to the maintenance of a meristematic stem cell line (Westing 1964), which allows for the rejuvenation of foliage. This is mitigated by the culmination of height and crown growth, as well as the slow accumulation of inhibitory substances and pathogens (Westing 1964). Defense against decay, injury, and pathogens also contributes to increased longevity (Lanner 2002). The lifespan of a tree can be increased by the regeneration of foliage through epicormic sprouting, which has been observed as an ontogenetic shift in growth form (Bryan and Lanner 1981; Ishii and Ford 2001; Lanner 2002; Kennedy et al. 2004). Furthermore, the formation of hydraulically independent sectors on the tree may prolong a tree’s old-age by protecting the whole tree from damage to its parts (Lanner 2002).

There are several implications of this multi-objective optimization study for *P. menziesii* longevity. Given the current model structure, branches simulated for up to 500 years exhibit a decline in foliage past the maximum observed branch age. This implies that, regardless of external factors, branch longevity is intrinsically limited. The potential lifespan of *P. menziesii* branches determined by simulation is well within expected interval of stand-replacing disturbance (Agee 1993). In contrast, branches in Part2 (which I claim better match true fir morphology) tend to plateau in their foliage dynamics, implying that the branches could survive indefinitely. Of course, the model allows these to reiterate (in a limited manner), which regenerates foliage in a behavior inconsistent with observed growth dynamics and basic observed process of reiteration (e.g., Kennedy et al. 2004 for *A. grandis*). This is possibly problematic because true fir trees and their branches are not expected to live as long as *P. menziesii* trees or branches, although the true firs are able to persist in the old-growth canopy. It should be mentioned that the model is not designed to produce true fir branches. The emergence of the morphology in the optimization results reflects the similarities in the basic architectural models of the two species. A model for a true fir would modify the process of suppressed bud release, and should not include the capacity to generate independent SCUs through adaptive proleptic reiteration (Kennedy et al. 2004). This would diminish the capacity for foliage regeneration in the branches, and possibly reduce their potential lifespan.

Further inference of model results for *P. menziesii* longevity depend on comparison of model results to the actual lifespan of major branch axes, as well as a better understanding of the demography of the suppressed bud bank. Of particular importance is the distribution of ages at which the bud aborts. If longevity relies on a healthy reserve of meristematic cells, then how long a branch retains suppressed buds is imperative to its capacity for foliage regeneration. In addition, Lanner (2002) proposes that the generation of hydraulically independent sectors in a tree can prolong a tree's life by protecting it from damage to other tree parts. If bud release from suppression causes a more severe hydraulic constriction in *P. menziesii* than sequential lateral growth, then this can cause hydraulic segmentation (Tyree and Zimmerman 2002), which would isolate the

independent SCUs and prevent mass cavitation due to damage to a single SCU axis. This would also help explain how SCU turnover does not have a catastrophic effect on the entire branch. In contrast, the true fir morphology, as it emerges in the optimization results, relies heavily on maintenance of a few order-1 axes. Any damage to those would have drastic consequences for the longevity of the branch, depending on their own level of segmentation.

One of the major lessons from this model and optimization study is how morphology is shown to integrate with physiological functions to affect branch development. Although the growth functions utilize simple surrogates for physiological phenomenon (e.g., foliage overlap for light interception and photosynthesis, order and distance to nearest foliage for hormonal interactions), these have noticeable interactions with the branch morphology. The full extent of how these integrate can only be understood through further empirical analysis. For example, what is the relationship between hydraulic constrictions at branch junctions and the two contrasting morphologies that have varying performance with respect to n_{turns} ? The model assumes that there is a severe constriction at epicormic junctions, but not at lateral junctions. For *P. menziesii*, this results in a morphology that tends to have higher turns than other possible morphologies. For another example, if true firs can be considered to have a more shade-tolerant physiology than *P. menziesii*, then does that explain why, in the modeling results, it exhibits a phalanx-like morphology (such that it isn't sensitive to higher light levels)? If so, then why does it exhibit the threshold effect with foliage overlap (Figure 5.7)? Perhaps some level of overlap is unacceptable (inefficient) space filling for a phalanx-like growth pattern.

The physiology of wood formation could also be explained by these results, if the wood of true firs is stiffer than that of *P. menziesii*. Then, they are able to sustain longer major order-1 and order-2 axes at an angle that allows for an acceptable level of light interception. We have observed (but not yet quantified) that lateral axes of *P. menziesii* tend to droop vertically, and that new epicormic order 1 axes are held at a higher angle.

Therefore lateral growth may be sufficient to maintain foliage light interception on true firs, whereas in *P. menziesii* this requires epicormic growth. These questions are motivated by the manner in which morphology integrates with physiology in the model. Further empirical research guided by these results would allow for a more detailed modeling of the physiological processes. This would facilitate a more complete theory of how the physiology and morphology integrate. All of these inferences must be considered in the context of the model structure.

My “truth in advertising”

Objective performance in the context of the model structure

Clearly the optimization results are bounded by the quantitative structure of the objectives, as well as the model structure. Although the results give some evidence in favor of the objectives representing constraints in the old-growth system, in that characteristic patterns emerge with divergent performance with respect to the objectives, the results may be a serendipitous artifact of the combination of model structure and objectives. Furthermore, the result of reduced l_{path} may be a characteristic of the growth pattern that is ancillary to more important effects; there may be a constraint that is correlated to l_{path} and better explains the growth pattern, but was not included in the vector objective function. For example, the placement of SCUs near the main stem may have an affect on mechanical load that is not reflected by *load* as quantified in the objectives. A less important consequence may be the shorter l_{path} . I mention l_{path} in particular because the path length along the branch is very small relative to the total path length from the soil to terminal foliage. Sensitivity of growth to l_{path} at this scale is suspect, and whether differences in path length of 200 cm vs. 300 cm would really impact growth should be investigated empirically. There may be an additive effect, wherein at extreme path lengths at the tops of old trees any additional change in path length has a detrimental effect.

Model structure may also be a factor in foliage display not emerging as a dominant requirement that differentiates the two major morphologies. This could be

because both morphologies require a particular level of foliage display, so the value of n_{over} does not clearly differentiate between the two and is secondary to hydraulic constraints as quantified. Alternatively, this is possibly due to the model structure because the value of overlap is a rough surrogate for foliage display. A more complicated algorithm that calculated actual foliage display, allowed for some light transmittance through foliage, and calculated these given a three-dimensional branch structure may show foliage display to be a more dominant objective. This would be particularly relevant when the changes in the sun path across a day and season are taken into account. These calculations should be integrated with the response of photosynthesis to varying light levels (which may differ between species), which depends on data that are not yet available. This would require a significant advancement of the current model structure and more detailed empirical observation of three-dimensional branch structure.

Parameter values (and inference) in the context of the model structure

The effect of each parameter in the optimization results is inextricably associated with the model structure and objectives. For example, a consequence of the probability structure for shoot proliferation is that the bifurcation function not only controls the number of daughter shoots, but also how long a particular axis survives. That is, it controls when an axis draws a zero and can no longer proliferate. For example, in the placement of new SCUs it may be irrelevant whether the new shoot produces 1 or 3 daughters, as long as the young axis survives to form a new SCU. The number of daughters may be secondary to the survival of the axis. These effects may not be distinguishable in the current model structure, if that is even an important distinction to make.

The effect of increasing generation on new epicormic growth in particular might be a consequence of the model structure. The structure only allows for a continuous decrease in bifurcation with increasing generation, which yields poor estimates of new epicormic bifurcation at low generations relative to that observed (Figure 5.9), although

the effect of n_{over} does mitigate these estimates (Figure 5.7). There may be an alternative explanation for the maximum generation that does not conform to the current model structure. If so, then the effect of generation that is in the optimization results is the only means to produce that threshold within the confines of the model structure. All of the effects represented by the bifurcation parameters should be interpreted in the context of the axis longevity and bifurcation thresholds (see Chapter V).

For the p function, the process of bud release from suppression tends to take positive probability across the bud lifespan. This means that eventually all buds will be released. This process may, however, require a negative feedback that could prevent buds from being released within the branch lifespan, as dormant buds have been presumed to eventually abort (Bryan and Lanner 1981). Among the p function parameters, the value of p_{bud} tends to converge below zero (Chapter IV). This trend may reflect a negative effect of increased time of suppression on bud release as it is intended in the model. Alternatively, given the magnitudes of the effects of the remaining parameters, this may be the means by which the negative feedback is accomplished in the model regardless of the actual effect of t_{bud} . The effect, then, may simply be for the purpose of negative feedback that is necessary within the model structure rather than a reflection of some biological process. Of course, all of the other effects may have the same kind of interpretation, yet their consistent positive convergence and the emergence of recognizable patterns from the results lends some evidence that the effects reflect, to some degree, the observed system. In particular there is empirical evidence that slowing of growth induces release from suppression (Chapter III; Kormanik and Brown, 1969; Wignall *et al.*, 1987; Wignall and Browning, 1988; Remphrey and Davidson, 1992), whereas there is no evidence for a direct effect of the timing of initiation on bud release. If further empirical study does not provide such evidence, then the probability structure may need to be supplemented with a different process for the buds to abort, which would replace the effect of t_{bud} in the model structure.

The model results are insufficient to fully form a theory because they are bounded by the model structure, which serves as a qualifier for any parameter inference. The relationships may be serendipitous results of the model structure and objectives, and we require an integration of the model results with empirical investigation. The value is that the results point the direction for the empirical study, described later in this Chapter.

Modeling implications

The modeling system presented here is an improvement of the process of simulating plant form and function, and it allows for exploration of possibilities that are impossible to investigate in the field. Other modeling systems successfully mimic plant form (e.g., Lindenmayer 1968, Honda 1971; Godin et al. 1997), without analysis and assessment of the characteristics that lead to its development, or the multi-objective consequences of contrasting plant forms with differing patterns of morphogenesis. In previous simulations, morphology is often uncoupled from the growth processes and utilized simply for visualization of the results of the growth functions and developmental rules. In the modeling of plant form, it is desirable to understand how morphology itself affects the comparative growth of trees.

In general, models of plant architecture have two major components; the first component is a set of rules and functions that control plant growth and development. The second is the translation of growth patterns to a visual/geometric representation of plant form (Kurth 1994) that preserves the lineage of plant parts and their topology. The details and depth represented by the growth functions can vary widely, as can the integration of the geometric representation with the growth processes. Some models simulate growth requirements such as carbon assimilation or the flux of materials and water (e.g., Ford et al. 1990; Mäkelä et al. 1997; Fröh 1997), but the consequences of growth relationships for the morphological form of the plant is often not considered, nor how the morphological form changes performance on the growth processes (i.e., a feedback between the architecture and morphology and growth processes). The morphology may also be the result of a single physiological requirement that is modeled

in great detail (e.g., light interception, carbon assimilation, material flux; Honda et al. 1981; Borchert and Honda 1984; Evers et al. 2007).

The current modeling effort dynamically calculates the status of each node with basic representations of biological variables that are easy to understand and the effects of which are readily interpretable. From the functional structure, thresholds emerged that determine the relative pattern of shoot and SCU placement (Figure 5.6), and these have direct consequences for the functional performance of the branches. This is accomplished with simple representations of biological processes integrated into a function to determine bifurcation of active nodes. This is an advancement of the model of Kennedy et al. (2004), where, in their model structure, bifurcation is restricted to architectural status (order, generation) and epicormic initiation is restricted to a simple probability distribution.

Model development and assessment

This model is not the first to employ multiple possible constraints and controls on plant growth. Sterck et al. (2005) and Sterck and Schieving (2007) also evaluate possible constraints on growth and morphology for trees in general, yet their models consider those in a piecemeal, stepwise manner. Their analysis is limited, and the methodology presented here provides an escape from those limitations through an evaluation of those constraints simultaneously. They also show how foliage regeneration by the release of dormant buds at damaged axes allows plants to survive beyond plants that do not exhibit such behavior (Sterck and Schieving 2007). Yet, their representation of the process in their model structure does not have grounding in an observed system. They, and others (e.g., Evers et al. 2007) have recently recognized the need for methodologies to test such models and advance their biological interpretation, which would provide an empirical anchor for model conclusions. This dissertation has illustrated a promising methodology for the advancement of plant functional structural models.

The procedure to use process-based models in theory development requires a method to assess the process model structure against empirical and theoretical predictions. In Chapters II and IV, model inadequacies are uncovered only when the results of multi-objective optimization are evaluated in detail. Such an assessment procedure is an essential component of process modeling, and no theory should be proposed before the model is assessed thoroughly. I do not pretend that any model can be shown to be a completely true representation of the system, only that the process structure produces an adequate result relative to the modeling goals. For the current study, the goal was to observe how the process of proleptic reiteration compensates for size constraints on the system. The empirical objectives quantify, at a lower level of biological complexity, what I considered to be the minimum necessary for adequate representation of the branch structure. Model adequacy should be determined both by the ability of the model to satisfy those empirical objectives and by simulation of parameterizations in the Pareto optimal set. Unexpected patterns may emerge that are valuable in uncovering inadequacies both in model structure and in objective formulation. In this case, as is common in plant modeling, a visual assessment of the emergent branch pattern is a useful check on the adequacy of the model to reproduce the observed growth form. It is not, however, the only relevant measure.

The strategy I followed in the process of model development and assessment was similar to that outlined by Grimm (1999) and Wiegand et al. (2003) for individual based models (IBMs). Grimm (1999) suggests that model-building should begin with a simple model that reproduces an observed pattern of interest (Wiegand et al. 2003); the model is then improved by incorporating greater biological detail, and for each modification the model is assessed for whether it continues to replicate the observed pattern. This cycle of model building is repeated until an appropriate level of biological complexity is found that still adequately replicates the observed pattern (Grimm 1999; Wiegand et al. 2003). For BRANCHPRO3, I proposed independent variables that represented complex biological problems at a lower level of biological complexity. For example, rather than incorporate in the model a process representation of hydraulic architecture and cambial

growth (for which we do not have reliable data to justify a detailed model), I summarized the hydraulic effect with gross properties that are understood to influence hydraulic relations (i.e., l_{path} and n_{turns}). The process structure was specified at a similar level of biological complexity, with shoot status defined by generation (rather than hydraulic architecture), foliage overlap (rather than light interception), order (rather than hormonal influence), time since formation of the base of the axis (rather than hydraulic architecture), and distance to the base of the axis (rather than hydraulic architecture). The results of the optimization studies presented here will guide further empirical study, which can increase the credibility of the proposed theory and justify the incorporation of more detailed process specifications and optimization objectives.

Given the difficulty in designing ecological experiments with an appropriate control for statistical hypothesis testing for ecological systems, a valuable methodology in the course of ecological research is the use of an appropriate contrast (Ford 2000, Chapter 7.7). If two systems are compared that are similar in major characteristics, yet differ in an important and identifiable manner, then the consequences of the major difference can be inferred. One of the major results in the optimization is the clear divergence in morphology from the same basic model structure along the optimal frontier. The differentiating factors between the two morphologies are the relative performances with respect to l_{path} and n_{turns} . I have proposed that the Part2 morphology resembles that of true firs in the system such as *Abies amabilis* and *Abies grandis*. Kennedy et al. (2004) choose *A. grandis* as an appropriate contrast to *P. menziesii* in the investigation of the consequences of proleptic reiteration on branch longevity. The results presented here show that contrast to be particularly apt when investigating how morphologies may compensate for size constraints in a given system.

Further theory development: empirical observations

The model results for Part2 should be more explicitly compared to true fir morphology, which requires a detailed empirical analysis of that morphology. We have some preliminary evidence, but it may be that the model superficially replicates the visual

morphology; the details of the branch morphogenesis as defined by the model may be inadequate for true firs. There may be an alternative underlying model structure that would be more relevant to the true firs. Only a careful survey of growth can determine that. Some more detailed observation of the spatial distribution of SCUs in *P. menziesii* branch morphology should also be obtained.

Once it is determined that, indeed, true firs exhibit the characteristics of the morphology in Part2, then physiological contrasts should be made between the two species that integrate with the morphological contrasts. The first physiological measurement should be the resistance at lateral and epicormic junctions for both species. In an effort to integrate physiological processes with the morphology, it may be that true firs have a more severe restriction at sequential junctions than *P. menziesii*, and explain why the true fir morphology may tend to lessen the number of turns to terminal foliage. It has been observed that wood anatomy at branch junctions influence the degree of constriction (Eisner et al. 2002), which can be controlled by hormonal relationships (Kramer and Borkowski 2004). These could differ between species and would then show how the alternative morphologies integrate with alternative physiologies and wood anatomy. For path length, similar measurements could be made of conductance at a sampling of terminal shoots for both species, where the path length to the shoot is also recorded.

For the objective n_{over} , it would be very difficult to measure overlap as it is approximated in the model. However, light interception at various shoot positions could be measured in the field for both species. This would give a contrast of the light intercepted between the two morphologies, and should be measured at various positions along the horizontal branch axis, both along the main axis of SCUs and the lateral axes. For the issue of the load on the branches, a reasonable first approximation would be to measure branch diameter at the base of the branch, then at points along its major axis (or axes). This of course would be tempered by the physiological properties of the wood, which should also be contrasted.

Finally, the morphogenesis of *P. menziesii* should be studied in detail through a longitudinal survey of active growth areas on the branch. Bifurcation and release from suppression can be studied year-to-year for 3–5 years. These observations should be integrated with the light and hydraulic measurements. An intensive study should also be undertaken of the process of release from suppression, where along major branch axes the location, age and surrounding growth should be inventoried for new epicormic shoots. The water relations and light conditions for the first few years of epicormic growth should be recorded, with how those might differ with increased time since formation and generation. The light measurements should be designed to show whether new epicormic shoots preferentially develop into SCUs in better light conditions, in support of the proposal that the process of SCU development is an opportunistic (guerilla) process.

Conclusions

Process models are important tools in ecological theory development, and the results of process model simulations must always be anchored by empirical observation. It is essential that the model structure be interrogated thoroughly in the assessment process. One should never conduct the optimality, assessment, or simulation experiments under the assumption that the model is correct. As an abstraction of the system no model can represent ecological phenomena perfectly, yet modelers strive to build “verifiable” and “valid” models. In this quest models may quickly grow in size and complexity as it becomes clear that simple models are often inadequate for system behavior. In order to use models effectively, the analyst must begin with the supposition that the model itself is incorrect, and they must have an understanding of the model inadequacies as they relate to the research question. This places the context of the theory in the abstraction of the system that is the model.

The theoretical objectives presented in this dissertation were constructed through consideration of the constraints the biological example was operating under. The old-growth system as manifested in *P. menziesii* is obviously constrained, and I used the

multi-objective optimization to investigate those constraints in an optimality context. The use of multi-objective optimization and Pareto optimality for theory development, however, is not restricted to optimality studies. Turley (2001) uses a theoretical objective for assessment of a process model of competition. The theoretical objective is designed to distinguish one type of competition process from another. In a more general sense, the theoretical objectives are formulated from predictions of theory, whether optimality or other theory structures. They can and should be designed to synthesize theories and processes that likely act in concert for ecological systems. Through the synthesis of multiple phenomena, multi-objective optimization for ecological process models can be a key tool in theory development for ecological systems.

References

- Acker SA, Halpern CB, Harmon ME, Dyrness CT. (2002). Trends in bole biomass accumulation, net primary production and tree mortality in *Pseudotsuga menziesii* forests of contrasting age. *Tree Physiology*. 22: 213–217.
- Agee JK. (1993). *Fire Ecology of Pacific Northwest Forests*. Island Press, Washington D.C.
- Aono M, Kunii TL. (1984). Botanical tree image generation. *IEEE: Computer Graphics and Applications*. 4(5): 10–34.
- Apple ME, Tiekotter K, Snow M, Young J, Soeldner A, Phillips D, Tingey D, Bond BJ. (2002). Needle anatomy changes with increasing tree age in Douglas-fir. *Tree Physiology*. 22 (2-3): 129–136.
- Bachelard EP. (1969). Studies on the formation of epicormic shoots on eucalypt stem segments. *Australian Journal of Biological Sciences*. 22: 1291–1296.
- Barnard HR, Ryan MG. (2003). A test of the hydraulic limitation hypothesis in fast-growing *Eucalyptus saligna*. *Plant, Cell and Environment*. 26: 1235–1245.
- Begin C, Filion L. (1999). Black spruce (*Picea mariana*) architecture. *Canadian Journal of Botany*. 77: 664–672.
- Beven K. (2006). A manifesto for the equifinality thesis. *Journal of Hydrology*. 320: 18–36.
- Binkley D, Stape JL, Ryan MG, Barnard H, Fownes J. (2002). Age-related decline in forest ecosystem growth: an individual-tree, stand-structure hypothesis. *Ecosystems*. 5: 58–67.
- Binkley D. (2004). A hypothesis about the interaction of tree dominance and stand production through stand development. *Forest Ecology and Management*. 190: 265–271.
- Bollmark M, Hao-Jie C, Moritz T, Eliasson L. (1995). Relations between cytokinin level, bud development and apical control in Norway spruce, *Picea abies*. *Physiologia Plantarum*. 95: 563–568.
- Bond BJ. (2000). Age-related changes in photosynthesis of woody plants. *Trends in Plant Science*. 5(8): 349–353.
- Bond BJ, Czarnomski NM, Cooper C, Day ME, Greenwood MS. (2007). Developmental decline in height growth in Douglas-fir. *Tree Physiology*. 27: 441–453.

- Borchert R, Honda H. (1984). Control of development in the bifurcating branch system of *Tabebuia rosea*: A computer simulation. *Botanical Gazette*. 145(2): 184–195.
- Borchert R, Slade NA. (1981). Bifurcation ratios and the adaptive geometry of trees. *Botanical Gazette*. 142(3): 394–401.
- Brooks JR, Hinckley TM, Ford ED, Sprugel DG. (1991). Foliage dark respiration in *Abies-amabilis* (Dougl) Forbes—variation within the canopy. *Tree Physiology*. 9(3): 325–338.
- Brown CL, McAlpine RG, Kormanik PP. (1967). Apical dominance and form in woody plants: A reappraisal. *American Journal of Botany*. 54(2): 153–162.
- Bryan JA, Lanner RM. (1981). Epicormic branching in Rocky Mountain Douglas-fir. *Canadian Journal of Forest Research*. 11: 190–199.
- Čermák J, Kucera J, Bauerle WL, Phillips N, Hinckley TM. (2007). Tree water storage and its diurnal dynamics related to sap flow and changes in stem volume in old-growth Douglas-fir trees. *Tree Physiology*. 27(2): 181–198.
- Cline MG. (1991). Apical dominance. *Botanical Review*. 57: 318–358.
- Cline MG. (1994). The role of hormones in apical dominance. New approaches to an old problem in plant development. *Physiologia Plantarum*. 90: 230–237.
- Cline MG. (1997). Concepts and terminology of apical dominance. *American Journal of Botany*. 84: 1064–1069.
- Cohon JL. (1978). Multiobjective Programming and Planning. Volume 140 in *Mathematics Science and Engineering*, edited by Richard Bellman. Academic Press, New York.
- DeWit I, Keulemans J, Cook NC. (2002). Architectural analysis of 1-year-old apple seedlings according to main shoot growth and sylleptic branching characteristics. *Trees*. 16: 473–478.
- Doust LL. (1981). Population Dynamics and Local Specialization in a Clonal Perennial (*Ranunculus Repens*): I. The Dynamics of Ramets in Contrasting Habitats. *The Journal of Ecology*. 69(3): 743–755.
- Eisner NJ, Gilman EF, Grabosky JC, Beeson RC. (2002). Branch junction characteristics affect hydraulic segmentation in red maple. *Journal of Arboriculture*. 28(6): 245–251.

Evers JB, Vos J, Chelle M, Andrieu B, Fournier C, Struik PC. (2007). Simulating the effects of localized red:far-red ratio on tillering in spring wheat (*Triticum aestivum*) using a three-dimensional virtual plant model. *New Phytologist*. 176: 325–336.

Ewers FW, Zimmerman MH. (1984a). The hydraulic architecture of eastern hemlock (*Tsuga-canadensis*). *Canadian Journal of Botany*. 62(5): 940–946.

Ewers FW, Zimmerman MH. (1984b). The hydraulic architecture of balsam fir (*Abies-balsamea*). *Physiologia Plantarum*. 60(4): 453–458.

Farnsworth KD, Niklas KJ. (1995). Theories of optimization, form and function in branching architecture in plants. *Functional Ecology*. 9(3): 355–363.

Fisher JB, Honda H. (1977). Computer simulation of branching pattern and geometry in *Terminalia* (Combretaceae), a tropical tree. *Botanical Gazette*. 138(4): 377–384.

Fisher JB, Honda H. (1979). Branch geometry and effective leaf area: A study of *Terminalia*-branching pattern.1. Theoretical trees. *American Journal of Botany*. 66(6): 633–644.

Fournier C, Andrieu B. (1999). ADEL-maize: an L-system based model for the integration of growth processes from the organ to the canopy. Application to the regulation of morphogenesis by light availability. *Agronomie*. 19: 313–327.

Ford ED, Avery A, Ford R. (1990). Simulation of branch growth in the *Pinaceae*: Interactions of morphology, phenology, foliage productivity and the requirement for structural support, on the export of carbon. *Journal of Theoretical Biology*. 146: 15–36.

Ford ED. (2000). *Scientific Method for Ecological Research*. Cambridge University Press, Cambridge UK.

Franklin JF, Shugart HH, Harmon ME. (1987). Tree death as an ecological process. The causes, consequences and variability in tree mortality. *BioScience*. 37(8): 550–556.

Franklin JF, DeBell DS. (1988). Thirty-six years of tree population change in an old-growth Pseudotsuga-Tsuga forest. *Canadian Journal of Forest Research*. 18: 633–639.

Früh T. (1997). Simulation of water flow in the branched tree architecture. *Silva Fennica*. 31(3): 275–284.

Godin C, Costes E, Caraglio Y. (1997). Exploring plant topological structure with the AMPmod software: an outline. *Silva Fennica*. 31(3): 357–368.

Godin C, Sinoquet H. (2005). Functional-structural plant modeling. *New Phytologist*. 166(3): 705–708.

- Gould SJ, Lewontin RC. (1979). The Spandrels of San Marco and the Panglossian Paradigm: A Critique of the Adaptationist Programme. *Proceedings of the Royal Society of London. Series B, Biological Sciences*. 205(1161): 581–598.
- Gower ST, McMurtie RE, Murty D. (1996). Aboveground net primary production decline with stand age: potential causes. *Trends in Ecology and Evolution*. 11(9): 378–382.
- Gupta HV, Sorooshian S, Yapo PO. (1998). Toward improved calibration of hydrologic models: Multiple and noncommensurable measures of information. *Water Resources Research*. 34(4): 751–763.
- Grimm V. (1999). Ten years of individual-based modelling in ecology: what have we learned and what could we learn in the future? *Ecological Modeling*. 115: 129–148.
- Haefner JW. (1996). *Modeling Biological Systems: Principles and Applications*. Chapman and Hall, New York.
- Hallé F, Oldeman RAA, Tomlinson PB. (1978). *Tropical Trees and Forests. An Architectural Analysis*. Springer-Verlag, New York.
- Hao-Jie, C., Bollmark, M., and Eliasson, L. (1996). Evidence that cytokinin controls bud size and branch form in Norway spruce. *Physiologia Plantarum*. 98: 612–618.
- Harding RB. (1986). Terminal leader failure in white spruce plantations in northern Minnesota. *Canadian Journal of Forest Research*. 16: 648–650.
- Honda H. (1971). Description of the forms of trees by parameters of the tree-like body: Effects of branching angle and branch length on the shape of the tree-like body. *Journal of Theoretical Biology*. 31: 331–338.
- Honda H. (1978). Description of cellular patterns by Dirichlet domains—2-dimensional case. *Journal of Theoretical Biology*. 72 (3): 523–543.
- Honda H, Fisher JB. (1978). Tree branch angle: Maximizing effective leaf area. *Science*. 199: 888–890.
- Honda H, Fisher JB. (1979). Ratio of tree branch lengths: The equitable distribution of leaf clusters on branches. *Proceedings of the National Academy of Sciences of the United States of America*. 76(8): 3875–3879.
- Honda H, Tomlinson PB, Fisher JB. (1981). Computer simulation of branch interaction and regulation by unequal flow rates in botanical trees. *American Journal of Botany*. 68(4): 569–585.

Hubbard RM, Bond BJ, Ryan MG. (1999). Evidence that hydraulic conductance limits photosynthesis in old *Pinus ponderosa* trees. *Tree Physiology*. 19: 165–172.

Hubbard RM, Bond BJ, Senock RS, Ryan MG. (2002). Effects of branch height on leaf gas exchange, branch hydraulic conductance and branch sap flux in open-grown *ponderosa* pine. *Tree Physiology*. 22: 575–581.

Ishii H, Reynolds JH, Ford ED, Shaw DC. (2000). Height growth and vertical development of an old-growth *Pseudotsuga-Tsuga* forest in southwestern Washington State, U.S.A. *Canadian Journal of Forest Research*. 30: 17–24.

Ishii H, Ford ED. (2001). The role of epicormic shoot production in maintaining foliage in old *Pseudotsuga menziesii* (Douglas-fir) trees. *Canadian Journal of Botany*. 79(3): 251–264.

Ishii H, Wilson ME. (2001). Crown structure of old-growth Douglas-fir in the western Cascade Range, Washington. *Canadian Journal of Forest Research*. 31: 1250–1260.

Ishii H, Ford ED, Dinnie CE. (2002). The role of epicormic shoot production in maintaining foliage in old *Pseudotsuga menziesii* (Douglas-fir) trees II. Basal reiteration from older branch axes. *Canadian Journal of Botany*. 80(9): 916–926.

Ishii H, McDowell N. (2002). Age-related development of crown structure in coastal Douglas-fir trees. *Forest Ecology and Management*. 169(3): 257–270.

Ishii H, Ford ED. (2002). Persistence of *Pseudotsuga menziesii* (Douglas-fir) in temperate coniferous forests of the Pacific Northwest Coast, USA. *Folia Geobotanica*. 37: 63–69.

Ishii H, Ford ED, Sprugel DG. (2003). Comparative crown form and branching pattern of four co-existing tree species in an old-growth *Pseudotsuga-Tsuga* forest. *Eurasian Journal of Forest Research*. 6(2): 99–109.

Ishii HT, Ford ED, Kennedy MC. (2007). Physiological and ecological implications of adaptive reiteration as a mechanism for crown maintenance and longevity. *Tree Physiology*. 27(3): 455–462.

Kaitaniemi P, Hanan JS, Room PM. (2000). Virtual sorghum: visualisation of partitioning and morphogenesis. *Computers and Electronics in Agriculture*. 28: 195–205.

Kennedy MC. (2002). A Geometric Simulation Model of Foliage Regeneration in *Abies grandis* and *Pseudotsuga menziesii*. MSc Thesis, University of Washington. Seattle, WA.

- Kennedy MC, Ford ED, Ishii H. (2004). Model analysis of the importance of reiteration for branch longevity in *Pseudotsuga menziesii* compared with *Abies grandis*. *Canadian Journal of Botany*. 82: 892–909.
- Koch GW, Sillett SC, Jennings GM, Davis SD. (2004). The limits to tree height. *Nature*. 428(22): 851–854.
- Komuro R. (2005) Multi-Objective Evolutionary Algorithms for Ecological Process Models. PhD Dissertation, University of Washington. Seattle, WA.
- Komuro R, Ford ED, Reynolds JH. (2006). The use of multi-criteria assessment in developing a process model. *Ecological Modelling*. 197(3–4): 320–330.
- Kormanik PP, Brown CL. (1969). Origin and development of epicormic branches in sweetgum. USDA Forest Service Research Report. SE-54.
- Kramer EM, Borkowski MH. (2004). Wood grain patterns at branch junctions: modeling and implications. *Trees*. 18: 493–500.
- deKroon H, Knops J. (1990). Habitat exploration through morphological plasticity in two chalk grassland perennials. *Oikos*. 59: 39–49.
- deKroon H, Hutchings MJ. (1995). Morphological plasticity in clonal plants: the foraging hypothesis revisited. *Journal of Ecology*. 83: 143–152.
- Kurth W. (1994). Morphological models of plant growth: Possibilities and ecological relevance. *Ecological Modelling*. 75/76: 299–308.
- Lanner RM. (2002). Why do trees live so long? *Ageing Research*. 1: 653–671.
- Larson PR, Isebrands JG. (1978). Functional significance of the nodal constricted zone in *Populus deltoides*. *Canadian Journal of Botany*. 56: 801–804.
- Lindenmayer A. (1968). Mathematical models for cellular interactions in development. I. Filaments with one-sided inputs; II. Simple and branching filaments with two-sided inputs. *Journal of Theoretical Biology*. 18: 280–315.
- Levy PE, Friend AD, White A, Cannell MGR. (2004). The influence of land use change on global-scale fluxes of carbon from terrestrial ecosystems. *Climatic Change*. 67: 185–209.
- Mäkelä A, Vanninen P, Ikonen V-P. (1997). An application of process-based modeling to the development of branchiness in Scots Pine. *Silva Fennica*. 31(3): 369–380.

McDowell NG, Phillips N, Lunch C, Bond BJ, Ryan MG. (2002a). An investigation of hydraulic limitation and compensation in large, old Douglas-fir trees. *Tree Physiology*. 22: 763–774.

McDowell N, Barnard H, Bond BJ, Hinckley T, Hubbard RM, Ishii H, Köstner B, Magnani F, Marshall JD, Meinzer FC, Phillips N, Ryan MG, and Whitehead D. (2002b). The relationship between tree height and leaf area: sapwood area ratio. *Oecologia*. 132(1): 12–30.

Mencuccini M, Grace J. (1996). Hydraulic conductance, light interception and needle nutrient concentration in Scots pine stands and their relations with net primary productivity. *Tree Physiology*. 16: 459–468.

Miguel LC, Longnecker NE, Ma Q, Osborne L, Atkins CA. (1998). Branch development in *Lupinus angustifolius* L. I. Not all branches have the same potential growth rate. *Journal of Experimental Biology*. 49(320): 547–553.

Moorby J, Wareing PF. (1963). Ageing in woody plants. *Annals of Botany*. 21(106): 291–308.

Morgan J, Cannell MGR. (1988). Support costs of different branch designs—effects of position, number, angle and deflection of laterals. *Tree Physiology*. 4(4): 303–313.

Nicolini E, Chanson B, Bonne F. (2001). Stem growth and epicormic branch formation in understory beech trees (*Fagus sylvatica* L.). *Annals of Botany*. 87: 737–750.

Niklas KJ, Kerchner V. (1984). Mechanical and photosynthetic constraints on the evolution of plant shape. *Paleobiology*. 10(1): 79–101.

Niklas KJ. (1992). *Plant Biomechanics. An Engineering Approach to Plant Form and Function*. The University of Chicago Press, Chicago.

Niklas KJ. (1997a). Adaptive walks through fitness landscapes for early vascular land plants. *American Journal of Botany*. 84(1): 16–25.

Niklas KJ. (1997b). Effects of hypothetical barriers and abrupt environmental changes on adaptive walks in a computer-generated domain for early vascular plants. *Paleobiology*. 23(1): 63–76.

Niklas KJ. (1999). Evolutionary walks through a plant morphospace. *Journal of Experimental Botany*. 330: 39–52.

Niklas KJ. (2007). Maximum plant height and the biophysical factors that limit it. *Tree Physiology*. 27(3): 433–440.

Oreskes N, Shrader-Frechette K, Belitz K. (1994). Verification, validation, and confirmation of numerical models in the earth sciences. *Science*. 263(5147): 641–649.

Parker GA, Smith JM. (1990). Optimality theory in evolutionary biology. *Nature*. 348: 27–33.

Parker GG, Davis MM, Chapotin SM. (2002). Canopy light transmittance in Douglas-fir-western hemlock stands. *Tree Physiology*. 22: 147–157.

Pearcy RW, Muraoka H, Valladares F. (2005). Crown architecture in sun and shade environments: assessing function and trade-offs with a three-dimensional simulation model. *New Phytologist*. 166(3): 791–800.

Peterson DL and Parker VT (eds). (1998). *Ecological Scale: Theory and Applications*. Columbia University Press, New York.

Phillips N, Bond BJ, McDowell NG, Ryan MG. (2002). Canopy and hydraulic conductance in young, mature and old Douglas-fir trees. *Tree Physiology*. 22: 205–211.

Phillips NG, Ryan MG, Bond BJ, McDowell NG, Hinckley TM, Čermák J. (2003). Reliance on stored water increases with tree size in three species in the Pacific Northwest. *Tree Physiology*. 22: 237–245.

Prusinkiewicz P, Lindenmayer A, Hanan JS. (1990). *The Algorithmic Beauty of Plants*. Springer-Verlag, New York.

Pruyn ML, Gartner BL, Harmon ME. (2002). Within-stem variation of respiration in *Pseudotsuga menziesii* (Douglas-fir) trees. *New Phytologist*. 154(2): 359–372.

Quine CP. (2004). Development of epicormic sprouts on Sitka spruce stems in response to windthrown gap formation. *Forestry*. 77(3): 225–233.

deReffye Ph, Fourcaud T, Blaise F, Barthélémy D, Houllier F. (1997). A functional model of tree growth and tree architecture. *Silva Fennica*. 31(3): 297–311.

Remphrey WR, Davidson CG. (1992). Spatiotemporal distribution of epicormic shoots and their architecture in branches of *Fraxinus pennsylvanica*. *Canadian Journal of Forest Research*. 22: 336–340.

Reynolds JH. (1996). *Multi-criteria Assessment of Ecological Process Models Using Pareto Optimization*. PhD Dissertation, University of Washington. Seattle, WA.

Reynolds JH, Ford ED. (1999). Multi-criteria assessment of ecological process models. *Ecology*. 80: 538–553.

- Reynolds J H, Golinelli D. (2005). Multi-criteria inference for process models: structural and parametric inference for a stochastic model of feline hematopoiesis. 2004 Proceedings of the American Statistical Association, Section on Statistics & the Environment [cd-rom], Alexandria, VA: American Statistical Association.
- Rothley KD, Schmitz OJ, Cohon JL. (1997). Foraging to balance conflicting demands: novel insights from grasshoppers under predation risk. *Behavioral Ecology*. 8 (5): 551–559.
- Ryan MG, Yoder BJ. (1997). Hydraulic limits to tree height and tree growth. *BioScience*. 47(4): 235–242.
- Ryan MG, Binkley D, Fownes JH. (1997). Age-related decline in forest productivity: pattern and process. *Advances in Ecological Research*. 27: 213–262.
- Ryan MG, Binkley D, Fownes JH, Giardina CP, Senock RS. (2004). An experimental test of the causes of forest growth decline with stand age. *Ecological Monographs*. 74 (3): 393–414.
- Ryan MG, Phillips N, Bond BJ. (2006). The hydraulic limitation hypothesis revisited. *Plant Cell and Environment*. 29 (3): 367–381.
- Rykiel EJ. (1996). Testing ecological models: the meaning of validation. *Ecological Modelling*. 90(3): 229–244.
- Schmitz OJ, Cohon JL, Rothley KD, Beckerman AP. (1998). Reconciling variability and optimal behaviour using multiple criteria in optimization models. *Evolutionary Ecology*. 12: 73–94.
- Schmitz OJ. (2000). Combining field experiments and individual-based modeling to identify the dynamically relevant organization scale in a field system. *Oikos*. 89: 471–484.
- Schmitz OJ. (2001). From interesting details to dynamical relevance: toward more effective use of empirical insights in theory construction. *Oikos*. 94: 39–50.
- Schulte PJ, Brooks JR. (2003) Branch junctions and the flow of water through xylem in Douglas-fir and ponderosa pine stems. *Journal of Experimental Botany*. 54 (387): 1597–1605.
- Shaw DC, Franklin JF, Bible K, Klopatek J, Freeman E, Greene S, Parker GG. (2004) Ecological setting of the wind river old-growth forest. *Ecosystems*. 7 (5): 427–439.
- Sillett SC, VanPelt R. (2007). Trunk reiteration promotes epiphytes and water storage in an old-growth redwood forest canopy. *Ecological Monographs*. 77(3): 335–359.

- Smith JM. (1978). Optimization theory in evolution. *Annual Review of Ecological Systems*. 9: 31–56.
- Sprugel DG, Brooks JR, Hinckley TM. (1996). Effects of light on shoot geometry and needle morphology in *Abies amabilis*. *Tree Physiology*. 16: 91–98.
- Steingraeber DA, Waller DM. (1986). Non-stationarity of tree branching patterns and bifurcation ratios. *Proceedings of the Royal Society of London B*. 228: 187–194.
- Sterck FJ, Schieving F, Lemmens A, Pons TL. (2005). Performance of trees in forest canopies: explorations with a bottom-up functional-structural plant growth model. *New Phytologist*. 166(3): 827–844.
- Sterck FJ, Schieving F. (2007). 3-D growth patterns of trees: effects of carbon economy, meristem activity, and selection. *Ecological Monographs*. 77(3): 405–420.
- Thomas SC. (1996). Asymptotic height as a predictor of height and allometric characteristics in Malaysian rain forest trees. *American Journal of Botany*. 83(5): 556–566.
- Tomlinson PB. (1983). Tree architecture. *American Scientist*. 71: 141–149.
- Turley MC. (2001). Investigating alternative ecological theories using multiple criteria assessment with evolutionary computation. PhD Dissertation, University of Washington. Seattle, WA.
- Tyree MT, Ewers FW. (1991). Tansley Review No. 34. The hydraulic architecture of trees and other woody plants. *New Phytologist*. 119: 345–360.
- Tyree MT, Alexander JD. (1993). Hydraulic conductivity of branch junction in three temperate tree species. *Trees*. 7:156–159.
- Tyree MT, Zimmerman MH. (2002). *Xylem Structure and the Ascent of Sap*. 2nd edition. Springer Series in Wood Science. Springer-Verlag, Berlin.
- Warren JM, Meinzer FC, Brooks JR, Domec JC, Coulombe R. (2007). Hydraulic redistribution of soil water in two old-growth coniferous forests: quantifying patterns and controls. *New Phytologist*. 173: 753–765.
- Westing, AH. (1964). The longevity and aging of trees. *Gerontologist*. 4: 10–15.
- Wiegand T, Jeltsch F, Hanski I, Grimm V. (2003). Using pattern-oriented modeling for revealing hidden information: a key for reconciling ecological theory and application. *Oikos*. 100: 209–222.

Wignall TA, Browning G, MacKenzie KAD. (1987). The physiology of epicormic bud emergence in Pedunculate Oak (*Quercus robur* L.) Responses to partial notch girdling in thinned and unthinned stands. *Forestry*. 60(1): 45–56.

Wignall TA, Browning G. (1988). The effects of stand thinning and artificial shading on epicormic bud emergence in pedunculate oak (*Quercus robur* L.). *Forestry*. 61(1): 45–59.

Woodruff DR, Bond BJ, Meinzer FC. (2004). Does turgor limit growth in tall trees? *Plant, Cell and Environment*. 27: 229–236.

Zimmerman MH. (1978). Hydraulic architecture of some diffuse-porous trees. *Canadian Journal of Botany*. 56: 2286–2295.

Appendix A

Multi-objective optimization

The assessment methodology presented here requires definition of the multi-objective optimization problem. The problem encompasses the model (M), its decision variables (X) and constraints, and the vector objective function (F).

$$\begin{aligned} M(&), \\ X & \subseteq \mathbb{R}^m, \\ F(&) = (F_1(M()), \dots, F_n(M())). \end{aligned}$$

The model (M) produces multiple outputs, whereas there are m decision variables (parameters). The vector objective function (F) measures n distinct features of model performance (Komuro et al. 2006). The optimization problem is to simultaneously minimize the n objectives in the vector objective function. Most likely there is not a single solution that minimizes all objectives simultaneously, so the Pareto optimal set is approximated to investigate the patterns of parameter values that result in ranges of optimal objective values. The Pareto optimal set is the set of all non-dominated solutions with respect to the vector objective function (Komuro et al. 2006). Qualitatively, a solution is determined to be non-dominated if there exists no other feasible solution that will give an improvement in one objective without a subsequent degradation in at least one other objective (Cohon 1978, pp 70; Fig. 1.1). More formally, parameterization X dominates X' ($X \prec X'$ or $X' \succ X$) \Leftrightarrow

$$\begin{aligned} \forall i, 1 \leq i \leq n, F_i(M(X)) \leq F_i(M(X')) \text{ and} \\ \exists i, 1 \leq i \leq n, \text{ such that } F_i(M(X)) < F_i(M(X')). \end{aligned}$$

X is co-dominant to X' ($X \parallel X'$) \Leftrightarrow

$$\begin{aligned} \exists i, j, i \neq j, \text{ such that } F_i(M(X)) < F_i(M(X')), \\ \text{and } F_j(M(X')) < F_j(M(X)). \end{aligned}$$

The Pareto optimal set is the set of all non-dominated solutions, i.e. those that are mutually co-dominant and not dominated by any other feasible solution (Table A.1; Table A.2).

Table A.1. Non-dominance example for binary errors. Solution2 is dominated by all other solutions and Solution4 dominates Solution1. Solution3 and Solution4 are the only two solutions that are mutually co-dominant and not dominated by any other solution.

Objective	Z_1	Z_2	Z_3	Z_4
Solution1	0	0	1	0
Solution2	0	0	0	0
Solution3	1	1	0	0
Solution4	0	0	1	1

Table A.2. Non-dominance example for continuous errors, where the optimization problem is to minimize all four objectives (Z_1, \dots, Z_4). Solution4 is dominated by all other solutions. Solution3 is dominated by Solution1. Only Solution1 and Solution2 are mutually co-dominant and not dominated by any other solutions.

Objective	Z_1	Z_2	Z_3	Z_4
Solution1	4.51	154	0.78	7.68
Solution2	2.67	315	0.85	10.4
Solution3	5.89	240	0.82	9.57
Solution4	5.11	350	0.90	20.4

The optimization problem can be defined for two different error structures, binary or continuous. In the binary error structure a range of target values for each objective is specified, and the model results are assessed for whether they fall within the target range. If the objective is within the target range, it is given an assessment value of 1, and 0 otherwise. All assessment values of 1 are considered "good", 0 "bad" (Reynolds and Ford 1999). For continuous error measures a single target value is specified, and model results are assessed for their Euclidean distance to the target value. These distances are minimized in the optimization.

Inference for the efficacy of model structure to achieve the objectives is performed through investigation of simulation results from parameterizations in the Pareto optimal set. For this problem the optimal set is approximated with an evolutionary algorithm that iterates populations of parameterizations, and produces each new generation of parameterizations from the best solutions in the previous generation. The algorithm converges on an approximation of the non-dominated optimal frontier.

Appendix B.

Source code for BRANCHPRO2 and BRANCHPRO3

Table B.1 List of files in BRANCHPRO3 source code. Code developed and compiled in Microsoft Visual C++ Express Edition.

File type	File	Description	Page
Header	branch.h	Branch-level function and variable declarations	181
	ParamObjDef.h	Parameter and objective names	184
	ran.h	Variables and functions for random number generation	185
	scu.h	SCU and shoot-level function and variable declarations	185
Source	brMain.cpp	Main function for BRANCHPRO	189
	brMaintenance.cpp	Branch-level source code	190
	critCalc.cpp	Source code for objective calculations and file writing	200
	misc.cpp	Source code for non-branch related functions	206
	ranNum.cpp	Source code for random number generation	206
	SCUmaint.cpp	SCU and shoot-level source code	209

Branch.h

```

/*****branch model header file*****/
/* Model and branch definitions and branch-level function */
/* declarations */
/*****/
#define MAXSEGS 50000 //Max # segments before terminating simulation
#define MINSEGS 5 //If segs<5 after year, then branch is dead
#define NUMRUNS 1 // keep odd for median calculations
#define CRIT_TYPE 0 // determines whether the mean (0) or median (1)
// is used. If (2) then both are recorded

#define MAXYR 90
#define SIMTYPE 1
/* 0 for pareto, 1 for other analyses (simulation studies), -1 for both
med and median, 2 for detailed debugging/confirmation*/

#define COMP 1 // 0 for server file directory, 1 for other

#define POPSIZE 100
#define NUMPARAMS 9
#define NUMCRIT 8 // change between OBJTHEOR=1 (5 obj) and OBJTHEOR=0
// (8 obj) and NEWLOAD=1 (6 and 9 obj respectively)
#define OBJTHEOR 0
/*indicator for whether the optimization is with the four empirical
objectives, or just theoretical objectives (including maximizing
shoots)*/
#define NEWLOAD 0 // indicator for a new load objective--a per order 1
//and order 2 junction relative diameter requirement,
//if this is 1 then include the new load objective
#define LOAD2 0 // indicator for whether to use the modified version
//of the original load objective
#define NTURNS 1 // indicator for whether nturns is included as an
//objective (0 if nturns is EXcluded, 1 otherwise)
#define PATH 0 // if 1, then path_length is the only theoretical
//objective, and NUMCRIT=5, set OBJTHEOR to 0 and
//NEWLOAD to 0

```

```

#define HALFSAT 0 // if 1, then half-saturation constants for Rb are
                  //read in as parameters and pBreak parameters are set
#define PBREAK 1 // a flag for whether the full pBreak function is
                  //included (1 or 2), or just p0 (0)

#define SYLL 1 // a flag, 1 for damage sylleptic reiteration, 0 for
               //deterministic main axis
#define NEWFUN 2 // which function for Rb and p_break
#define SCUEPI 1 // indicator for rule change: epis on independent
SCUs only when this is 1
#define DAMAGE 1
/* indicator for whether to allow a low level probability of damage,
begin with 0.02*/
#define DAMAGE_SENS 0 // indicator for whether there is a
                      //sensitivity analysis on damage_prob, the
                      //probability of damage
#define PI 3.141592

#define STOCH 1 // a flag, 1 if stochastic, 0 otherwise. Use zero //for
               constant (e.g. NEEDLE_LENGTH) sensitivity //analysis

#include "scu.h"

#define OVER_TEST 0 // for comparing algorithms to calculate //foliage
                   overlap.
#define PLOTBRANCH 1 // do I plot the branch? accepts multiple //
                   pops, but just one run per pop
#define EPIPLOT 0
#define PLOTYR 90

// for assessment, start with pomac99_branch

// Overlap and array constants
#define NEEDLE_LENGTH 1.8
#define ARR_SEARCH 6
#define YSHIFT 50
// For a 90-yr-old branch
/*
#define XSIZE 400
#define YSIZE 500
*/
// For a 145-yr-old branch

#define XSIZE 500
#define YSIZE 850

#define SEGTEST -1//5434
#define COMPTTEST -1 //5432

struct crit
{
    double final[POPSIZE][NUMCRIT];
    double run[NUMRUNS][NUMCRIT];
};

```

```

typedef crit* crit_ptr;

struct rb_obs
{
    double rb_all[MAXYR][3];
};

typedef rb_obs* rb_ptr;

/*****
class branch
{
public:
    double yr;
    int syll_idx;
    long nscus;
    long nsegs;

    int yDown;

    // scu_rekurs values
    double live_scus;
    double live_segs;
    double SCUs90;
    double shoots90;
    double nsegs_prev1,nsegs_prev2,nsegs_prev3;
    // added 07/13/05 -- to calculate emergent Rb for each order
    double nsegs1,nsegs2,nsegs3;
    // double rb_ord1,rb_ord2,rb_ord3;
    // long nseg1,nseg2,nseg3;
    double Nterm_turns;
    double Nterm;
    double Nterm_path;
    double length;

    long nover;
    int epi_inits;
    int scu_bases;
    double half_segs; // number of live segments beyond 1/2 the branch
                        length
    double third_segs; // number of live segments beyond 2/3 of the
                        branch length

    double njunct;
    double diam_sum;
    double Rb0,Rb1,Rb2,Rb3,Rb4;
    double p0, pForm, pInact, pScu, pStem;
    double sat1,sat2,sat3;
    double damage_prob;

    double rb_obs1,rb_obs2,rb_obs3;

    SCU *scu_head;
    SCU *scu_last;

```

```

    segs *shoot_head;
    segs *shoot_last;

    seg_ptr br_array[XSIZE][YSIZE];

    void free_segs(segs *cur_seg);
    void free_scus(SCU *temp_scu);
    void branch_initialize(SCU *scu1, branch *br, long *seed);
    void year_setup(SCU *first_scu, int pop, int run, branch *br,
                    long *seed, crit *obj, rb_obs *rbAll);
    void param_setup(branch *br, crit *obj, rb_obs *rbAll);
    void add_scu(SCU *newSCU, branch *br);
    void br_reset(branch *br);

};

typedef branch* br_ptr;

/*****functions*****/
void run_calc(crit *obj, branch *br, int pop, rb_obs *rbAll, int run);
void crit_calc(crit *obj, int pop);
void crit_out(crit *obj, int pop, branch *br);
void calc_rb(crit *obj, rb_obs *rbAll, int run);
void swap_array(double crit[], int a, int b);

```

ParamObjDef.h

```

/*****ParamObjDef*****/
/* Declares the parameter and objective names */
/*****
char pnames[NUMPARAMS][11]={
    "rb0", "rbOrd", "rbHydr", "rbOver", "rbform", "p0", "pForm", "pInact",
    "pScu", "pStem"
// NEWFUN==2
// other combinations
// "rb0", "rbOrd", "rbHydr", "rbOver", "p0", "pForm", "pInact", "pScu",
    "pStem" // NEWFUN!=2
// "rb0", "rbOrd", "rbHydr", "rbOver", "rbform", "p0" // PBREAK==0
// "rb0", "rbOrd", "rbHydr", "rbOver", "rbform", "sat1", "sat2", "sat3" //
HALFSAT==1

};

char cnames[NUMCRIT][11]={
    "fol_shts", "tot_scus", "length", "arch_mod", "nturns", "path"
    , "load", "nover" //OBJTHEOR==0
// "fol_shts", "nturns", "path", "load", "nover" //OBJTHEOR==1
// "fol_shts", "tot_scus", "length", "arch_mod", "nturns", "path", "load",
    "nover", "load2" //NEWLOAD==1
// "fol_shts", "tot_scus", "length", "arch_mod", "path", "load", "nover"
//NTURNS==0
// "fol_shts", "tot_scus", "length", "arch_mod", "path" //PATH==1

```

```
};
```

ran.h

```

/*****ran.h*****/
/* From numerical recipes in C.  Used in random */
/* number generation. (Press et al. 1999) */
/*****/

#define MBIG 1000000000
#define MSEED 161803398
#define MZ 0
#define FAC (1.0/MBIG)
#define PI 3.141592
double ran3(long *idum);

#define IM1 2147483563
#define IM2 2147483399
#define AM (1.0/IM1)
#define IMM1 (IM1-1)
#define IA1 40014
#define IA2 40692
#define IQ1 53668
#define IQ2 52774
#define IR1 12211
#define IR2 3791
#define NTAB 32
#define NDIV (1+IMM1/NTAB)
#define EPS 1.2e-7
#define RNMX (1.0-EPS)

double ran2(long *idum);

double poisdev(double xm, long *idum);

```

scu.h

```

/*****scu.h*****/
/* lists data structures and functions for shoots */
/* and SCUs */
/*****/

class branch;
class SCU;

struct segs {
    int segNum; //the segment number, may not need in this program
    int theta_quad; //quadrant of the angle, 1-4
    double length; //length of segment
    double gen;
    //generation of the shoot, increases with epicormics and with
    //traumatic reiteration
    double theta; //angle of segment to its parent

```

```

double year; //year this segment was formed
double order; //order of this shoot wrt to the SCU
int active; //indicates whether the meristem is still growing
double x1; //first x coordinate of segment
double x2; //second x coordinate of segment
double y1; //first y coordinate of segment
double y2; //second y coordinate of segment
double z1;
double z2;
int mort; //flag of mortality-no foliage when = 1
double age;
int epi;
int init;
int scu_base;
double dorage;
double nson;
double live_lin;
double lin;

double diam;
int junct_base;
double axis_length;

double xmid;
double ymid;

double nturns;
double distout;
double path_length;
double dist_turn;
double dist_base;
double tform; // the timing of bud break for the base of the
// current axis, age of the parent shoot, so 1 for all sequential
// growth, then timing of initiation for all epicormic axes

double fol_wt;
double fol_area;

// These give the coordinates for the rectangle that defines
// the foliage area for each shoot.
double theta2, theta3;
// the new angle from which the rectangular coordinates are calculated
double xll;
double yll;
double xul;
double yul;
double xlr;
double ylr;
double xur;
double yur;

double slope[2];
double interc[2][2];

```

```

    int syll;
    double nover;

    struct segs *next;
    // to point to the next of the "siblings" of the parent node
    struct segs *son;
    // to point to the first of the daughter shoots for this node
    struct segs *epi_son;
    // to point to the head of any new epicormic structure/ptl new SCU
    struct segs *par;

    struct segs *br_next;
    struct segs *arr_next;
    SCU* par_scu;
};

typedef segs* seg_ptr;

class SCU
{
    int numshoots;
    int numscus;
    double scu_yr;
    int SCUnum;
    branch *br;

    double ord1,ord2,ord3;
    double ord1_prev,ord2_prev,ord3_prev;
    int mort;

    double dout;
    double term_turns;
    double term_num;
    double term_path;
    double maxY;

    long nover;
    int njunct;
    double diam_sum;

public:
    seg_ptr shoot_head;
    seg_ptr shoot_last;

    int liveshoots;

    double yr;
    class SCU* next_scu;
    class SCU* son1;
    void make_shoots(segs *cur_seg,int flag,SCU *par_scu, double yr,
                    long *seed,branch *br);
    void shoot_extension(segs *cur_seg,double yr,
                        double length);

```

```

void shoot_lateral(segs *cur_seg,segs *prev_seg,int side,
    double yr,double length);
void shoot_rekurs(segs *temp,SCU *par_scu,double yr,
    long *seed,branch *br, int scu_idx, int pop);
void scu_rekurs(SCU *temp, double fol_value,int flag1,
    double yr,long *seed,branch *br, int pop);
void scu_lin(SCU *new_scu,segs *cur_seg,segs *prev_seg,int flag);
// to re-label the par_scu for the lineage of the new SCU
void make_buds(segs *cur_seg,int flag,long *seed,
    SCU *par_scu,double damage_prob);
void make_epini(segs *cur_seg,int side,SCU *par_scu,
    double yr,long *seed,double t_form,double length);
int test_scu(segs *epi_head,segs *epi_par,SCU *par_scu,
    branch *br);
int count_lin(segs *epi_temp,segs *epi_cur,int ind);
void test_epini(segs *cur_seg,SCU *par_scu,int test,
    double yr,long *seed,branch *br);
    void first_scu(SCU *scu1,long *seed,branch *br);
double scu_out(segs *cur_seg);
// to calculate the distance from the base of the SCU to the trunk
void shoot_fol(SCU *par_scu,segs *cur_seg,double yr,branch *br);
// to calculate foliage values based on regression relationships
void comp_loop(segs *cur_seg,segs *comp_seg,int flag);
void comp_vert(segs *cur_seg,segs *comp_seg,int flag);
void calc_lines(seg_ptr cur_seg);
int det_overlap(seg_ptr cur_seg,double xval,double yval,
    int flag);
int det_vert_over(seg_ptr cur_seg,seg_ptr comp_seg,
    double yval,int flag);
int det_horiz_over(seg_ptr cur_seg,seg_ptr comp_seg,
    double xval,int flag);
// Parameter values
double Rb0,Rb1,Rb2,Rb3,Rb4,sat1,sat2,sat3;
double p0, pForm, pInact,pScu,pStem;

void shoot_over(SCU *par_scu,segs *cur_seg,segs *comp_seg,
    long *seed, int flag, int flag2, branch *br);
void shoot_over2(SCU *par_scu,segs *cur_seg,segs *comp_seg,
    long *seed, int flag, int flag2, branch *br);
void calc_coord(seg_ptr cur_seg, seg_ptr par_seg);
void update_array(branch *br, seg_ptr cur_seg);
void array_over(segs *cur_seg,segs *comp_seg,int flag,
    int flag2,branch *br);
void shoot_rekurs_over(branch *br,segs *cur_seg, int flag,
    long *seed,SCU *par_scu,int pop, int run);
void idxOver_loop(branch *br,segs *cur_seg,int flag,
    SCU *par_scu);
void shoot_border(branch *br, segs *cur_seg,int flag);
void shoot_X0border(branch *br, segs *cur_seg, int flag,
    int flag2);
void shoot_X1border(branch *br, segs *cur_seg, int flag,
    int flag2);
void shoot_X2border(branch *br, segs *cur_seg, int flag,
    int flag2);
void comp_parallel(segs *cur_seg,segs *comp_seg,int flag);

```



```

    void plot_branch(branch *br, int pop, int run);
    void plot_rekurs(segs *cur_seg, branch *br, int pop, int run);
    void calc_diam(segs *cur_seg);
};

```

```
typedef SCU* scu_ptr;
```

brMain.cpp

```

/*****branch main*****/
/* Main function for BRANCHPRO */
/*****/

#include <iostream>
#include <fstream>
#include <cstdio>
#include <cmath>
#include <cstdlib>
#include <cstring>
#include <malloc.h>
#include <ctime>

using namespace std;

#include "ran.h"
#include "branch.h"
FILE *fscu;
int main(void)
{
    cout<<"\n\n*****\n\n";
    cout<<"\tBeginning Branch main\n\n";
    cout<<"*****\n\n";

    // Allocate memory for the branch, the objective values and the
    // bifurcations
    br_ptr brMain;
    brMain=new branch;

    crit_ptr obj;
    obj=new crit;

    rb_ptr rbAll;
    rbAll=new rb_obs;

    // Call the param_setup function to enter all remaining functions
    brMain->param_setup(brMain,obj,rbAll);

    // Delete the allocated memory
    delete brMain;
    delete obj;
    delete rbAll;

    cout<<"\n\n*****\n\n";
    cout<<"\tEnd of Branch main\n\n";
    cout<<"*****\n\n";
}

```

```

    return(0); // end of simulation
}

```

brMaintenance.cpp

```

/*****brMaintenance*****/
/* function definitions for branch-level computations */
/*****/

#include <iostream>
#include <fstream>
#include <cstdio>
#include <cmath>
#include <cstdlib>
#include <cstring>
#include <malloc.h>
#include <ctime>
using namespace std;

#include "branch.h"
#include "ParamObjDef.h"

time_t t;

/*****param_setup*****/
/* Begins the population of branches by reading in the parameter */
/* file and initializes the branch and calls the function that */
/* conducts growth over time. */
/* It then writes the criteria to the crit.out file. */
/*****/

void branch::param_setup(branch *br, crit *obj, rb_obs *rbAll)
{
    long seed;
    int run,i,j;
    int pop=0;
    if(STOCH==1)
    {
        srand(time(&t));
        seed=-time(&t);
    }

    // Establish proper format for criteria output file. All files will be
    // assigned to the same directory the executable is placed.
    FILE *fout;
    fout=fopen("critout.txt","w");
    for(i=0;i<NUMPARAMS;i++)
        fprintf(fout,"%s\t",pnames[i]);
    // write out the parameter names (defined in ParamObjDef.h)
    for(i=0;i<NUMCRIT;i++)
        fprintf(fout,"%s\t",cnames[i]);
    // write out the objective names (defined in ParamObjDef.h)

```

```

    fprintf(fout, "\n");
    fclose(fout);

// Open the parameter input file
    FILE *fpar;
    char fileLine[100];
    fpar=fopen("paretoin.txt", "r");

    if (fpar==NULL)
    {
        perror("paretoin.txt");
        exit(EXIT_FAILURE);
    }

    fgets(fileLine, 100, fpar);

// Run through the parameter file line-by-line
    while (!feof(fpar))
    {
// Set the run number to zero to begin simulation
        run=0;
// Scan in the parameter values from the current line of the
// parameter file
        if(PBREAK==1)
        {
            if(HALFSAT==1)
            {
                fscanf(fpar, "%lf %lf %lf %lf %lf %lf %lf %lf\n", &br->Rb0,
                    &br->Rb1, &br->Rb2, &br->Rb3, &br->Rb4, &br->sat1,
                    &br->sat2, &br->sat3);
                br->p0=0.002;
                br->pForm=0;
                br->pInact=0.002;
                br->pScu=0.005;
                br->pStem=0.045;
            }
            else
            {
                if(NEWFUN>=2)
                {
                    fscanf(fpar, "%lf %lf %lf %lf %lf %lf %lf %lf %lf %lf\n",
                        &br->Rb0, &br->Rb1, &br->Rb2, &br->Rb3, &br->Rb4,
                        &br->p0, &br->pForm, &br->pInact, &br->pScu,
                        &br->pStem);
                }
                else
                {
                    fscanf(fpar, "%lf %lf %lf %lf %lf %lf %lf %lf %lf\n",
                        &br->Rb0, &br->Rb1, &br->Rb2, &br->Rb3, &br->p0,
                        &br->pForm, &br->pInact, &br->pScu, &br->pStem);
                    br->Rb4=0;
                }
                br->sat1=4;
                br->sat2=8;
                br->sat3=5;
            }
        }
    }

```

```

    }
}
else
{
    if (HALFSAT==1)
    {
        fscanf(fpar,"%lf %lf %lf %lf %lf %lf %lf %lf\n",&br->Rb0,
            &br->Rb1,&br->Rb2,&br->Rb3,&br->Rb4,&br->sat1,
            &br->sat2,&br->sat3);
        br->p0=0.002;
        br->pForm=0;
        br->pInact=0.002;
        br->pScu=0.005;
        br->pStem=0.045;
    }
    else
    {
        fscanf(fpar,"%lf %lf %lf %lf %lf %lf\n",&br->Rb0,&br->Rb1,
            &br->Rb2,&br->Rb3,&br->Rb4,&br->p0);
        br->Rb4=0;
        br->sat1=4;
        br->sat2=8;
        br->sat3=5;
    }
}

if (DAMAGE_SENS==0)
    br->damage_prob=0.02;
else
{
    fscanf(fpar,"%lf ",&br->damage_prob);
}

printf("Beginning Population # %d\n",pop);

// Reset the objective values to zero
if (SIMTYPE==1)
{
    FILE *outfp;
    char plotout[50];
    char population[10];
    sprintf(population,"%d",pop);
    strcpy(plotout, "SimOutPop");
    strcat(plotout, population);
    strcat(plotout, ".txt\0");

    outfp=fopen(plotout,"w");

    fprintf(outfp,"run\tyr\tlive_segs\tlive_scus\tturns\t
        path\tload\tover\tRb1\tRb2\tRb3\n");
    fclose(outfp);

    FILE *scufp;
    char scuout[50];
    strcpy(scuout, "SimOutSCUPop");

```

```

        strcat(scuout, population);
        strcat(scuout, ".txt\0");

        scuftp=fopen(scuout,"w");
        fprintf(scuftp,"SCU\tlive_shoots\tdaughter_scus\t
                    generation\n");
        fclose(scuftp);
    }

// Begin the NUMRUNS loop--grows NUMRUNS # of branches for each line in
// the parameter file and calculates objective values
    for(j=0;j<NUMCRIT;j++)
    {
        obj->final[pop][j]=1000;
//initialize objective values to non-feasible defaults
        for(int k=0;k<NUMRUNS;k++)
            obj->run[k][j]=1000;
    }

    for(i=0;i<NUMRUNS&&br->live_segs<MAXSEGS;i++)
    {

        if(STOCH==0)
            seed=-2;
        printf("Run #: %d\n",i);

// Allocate memory for the first SCU
        scu_ptr scul;
        scul=new SCU;

// Give some of the initial values for the branch
        br->yr=1;
        br->scu_head=scul;
        br->scu_last=scul;
        br->live_segs=0;
        br->nscus=0;

// Fill in the rest of the information to initialize the branch
        branch_initialize(scul,br,&seed);

// Year_setup calls the primary scu recursion that performs growth
// functions, then calculates necessary yearly summaries
        year_setup(scul,pop,run,br,&seed,obj,rbAll);
        if(br->live_segs>=MAXSEGS)
        {
            for(i=0;i<NUMCRIT;i++)
            {
                obj->final[pop][i]=obj->run[run][i];
            }
        }
    }
// Increment the run #
    run=run+1;
}

```

```

        int runFlg=0;
// Once the set # of runs is accomplished, calculate summary objectives
// for this individual in the parameter population and write the values
// to the critout file
        if(br->live_segs<MAXSEGS)
            crit_calc(obj,pop);
        crit_out(obj,pop,br);
        br->live_segs=0;
        br->nscus=0;
// Increment the population #
        pop=pop+1;
    }

// Once the entire file has been read, close the file
if(fclose(fpar)!=0)
{
    printf("error closing file\n");
    exit(1);
}
return ;
}

/*****year_setup*****/
/* Conducts growth functions each year and calculates criteria */
/* values if the current run is the last for the parameter */
/* vector (defined by NUMRUNS). */
/*****/

void branch::year_setup(SCU *first_scu,int pop,int run,branch *br,
    long *seed,crit* obj,rb_obs *rbAll)
{
// Reset the year to 1, and the branch indicator to 1
    br->yr=1;
    int branch_flag=1;
    int k,j;
// Reset the values for observed RB to 0
    for(k=0;k<MAXYR;k++)
    {
        for(j=0;j<3;j++)
            rbAll->rb_all[k][j]=0;
    }
    br->epi_inits=0;
    br->scu_bases=0;

    scu_ptr cur_scu;

// Keep growing the branch until branch=0 (see below)
    while(branch_flag!=0)
    {

// Increment the branch year
        br->yr=br->yr+1;
        int scu_idx;

```

```

    int plot_yr;

    // To use the year in array indices, make it an integer
    plot_yr=ceil(br->yr);

    // Set the cur_scu to the first SCU to begin all navigations of the
    // SCU linked list
    cur_scu=first_scu;

    // Reset the value of live_scus to zero, to be updated below
    br_reset(br);
    /* The SCU recursion is repeated several times, each returning a unique
    value to branch. In the first run shoot_recursion is called and all
    growth functions are performed. It returns the updated # of foliated
    SCUs. */
    if(cur_scu!=NULL)
        cur_scu->scu_rekurs(cur_scu,0,1,br->yr,seed,br,pop
    else
    {
        printf("Debug something wrong with cur_scu\n");
        exit(1);
    }

    cur_scu=first_scu;

    // The second call sums the total # of foliated shoots
    // Done with SCU growth recursions, calculate the realized RB for
    // each order for this year
    if(br->nsegs_prev1>0)
        rbAll->rb_all[(plot_yr-1)][0]=br->nsegs1/br->nsegs_prev1;
    else
        rbAll->rb_all[(plot_yr-1)][0]=0;

    if(br->nsegs_prev2>0)
        rbAll->rb_all[(plot_yr-1)][1]=br->nsegs2/br->nsegs_prev2;
    else
        rbAll->rb_all[(plot_yr-1)][1]=0;

    if(br->nsegs_prev3>0)
        rbAll->rb_all[(plot_yr-1)][2]=br->nsegs3/br->nsegs_prev3;
    else
        rbAll->rb_all[(plot_yr-1)][2]=0;

    // Test the conditions to end the branch growth (either/or).
    // The maximum year is reached
    // The branch is already impossibly large (saves memory issues)
    // The branch is dead (has zero foliated shoots)
    cur_scu=first_scu;

    int intYr=ceil(br->yr);
    if(intYr%PLOTYR==0&&PLOTBRANCH==1)
        cur_scu->plot_branch(br,pop,run);
    else if(intYr%PLOTYR==0&&EPIPLOT==1)
        cur_scu->plot_branch(br,pop,run);

```

```

if (br->yr==MAXYR)
{
    branch_flag=0;
    cout<<"\nMAXYR\n";
}
if (br->live_segs>=MAXSEGS)
{
    branch_flag=0;
    cout<<"\ntoo many segments!\n";
    if (PLOTBRANCH==1)
        cur_scu->plot_branch(br,pop,run);
}
if (br->live_segs==0)
{
    branch_flag=0;
    cout<<"\nNo more foliated shoots\n";
}

if (br->yDown==1)
{
    branch_flag=0;
    cout<<"\nOut of Feasible Branch Area\n";
}

/* Calculate the sum of the # turns to foliated terminal shoots, and
the total # of foliated terminal shoots at the end of the branch run.*/
int flag2;
if (branch_flag==0)
    flag2=1;
else
    flag2=0;

/* Calculate the number of overlapping foliated shoots for each
foliated shoot*/
cur_scu->shoot_rekurs_over
    (br,br->shoot_head,1,seed,cur_scu,pop,run);

if (SIMTYPE==1)
{
    if (plot_yr%5==0)
    {
        cur_scu=first_scu;
        cur_scu->scu_rekurs(cur_scu,0,2,br->yr,seed,br,pop);
        run_calc(obj,br,pop,rbAll,run);
    }
}

/* Calculate the mean number of turns (meanTurns) to foliated terminal
shoots*/
cur_scu=first_scu;
if (SIMTYPE==0)
{
    cur_scu->scu_rekurs(cur_scu,0,2,br->yr,seed,br,pop);
    run_calc(obj,br,pop,rbAll,run);
}

```



```

    }
    /* Free the memory for the shoot and SCU linked lists to prepare for
    next branch.*/

    cur_scu=first_scu;
    cur_scu->scu_rekurs(cur_scu,-1,3,br->yr,seed,br,pop);
    br->free_scus(first_scu);
    cout<<"Live shoots: "<<br->live_segs<<"\tSCUs"<<
        br->live_scus<<"\t"<<br->yr<<"\n";
}

/*****branch_initialize*****/
/* At the beginning of each run, reset the values for branch */
/* demography, then call the function to fill in the values for the */
/* first SCU (which also fills in the values for the first couple */
/* of shoots). */
/*****/

void branch::branch_initialize(SCU *scu1,branch *br,long *seed)
{
    int i,j;
    // Assign the values for the branch for the first year
    br->nsegs=1;
    br->live_segs=1;
    br->live_scus=1;
    br->SCUs90=0;
    br->shoots90=0;
    br->syll_idx=0;
    br->nscus=1;
    br->epi_inits=0;
    br->scu_bases=0;

    br->shoot_head=NULL;
    br->shoot_last=NULL;

    br->yDown=0;

    for(i=0;i<XSIZE;i++)
    {
        for(j=0;j<YSIZE;j++)
            br->br_array[i][j]=NULL;
    }

    // An SCU function to fill in the information for the first SCU
    scu1->first_scu(scu1,seed,br);

    /* Assign the values for the first year for calculating average
    bifurcation*/
    br->nsegs1=1;
    br->nsegs2=2;
    br->nsegs3=0;
    br->nsegs_prev1=0;
    br->nsegs_prev2=0;

```

```

    br->nsegs_prev3=0;

    br->Nterm=0;
    br->Nterm_path=0;
    br->Nterm_turns=0;

    br->njunct=0;
    br->diam_sum=0;
}

void branch::br_reset(branch *br)
{
    br->live_scus=0;
    br->nsegs1=0;
    br->nsegs2=0;
    br->nsegs3=0;
    br->nsegs_prev1=0;
    br->nsegs_prev2=0;
    br->nsegs_prev3=0;

    br->Nterm=0;
    br->Nterm_path=0;
    br->Nterm_turns=0;
    br->length=0;
    br->live_scus=0;
    br->live_segs=0;

    br->half_segs=0;
    br->third_segs=0;

    br->njunct=0;
    br->diam_sum=0;

    br->nover=0;

    br->shoot_head=NULL;
    br->shoot_last=NULL;

    br->yDown=0;

    br->nscus=1;
}

/*****free_segs*****/
/* Frees the memory for the segs (shoots) between runs of the */
/* model (for each branch) */
/* */
/*****

void branch::free_segs(segs *temp)
{
    seg_ptr temp2;
    while(temp!=NULL)
    {

```

```

    if(temp->epi_son!=NULL)
    {
        if(temp->epi_son->scu_base==0)
            free_segs(temp->epi_son);
    }

    if(temp->syll==0)
    {
        if(temp->son!=NULL) free_segs(temp->son);

        temp2=temp->next;
        delete temp;
        temp=temp2;
    }
    else
    {
        if(temp->scu_base==1) temp=temp->next;
/* if you are the base of a sylleptically reiterated SCU, don't delete,
move to your sibling*/
        else
        {
            if(temp->son!=NULL) free_segs(temp->son);
            temp2=temp->next;
            delete temp;
            temp=temp2;
        }
    }
    delete temp;
}

/*****free_scus*****/
/* Frees the memory for the scus between runs of the model (for */
/* each branch) */
/* */
/*****/
void branch::free_scus(SCU *temp_scu)
{
    scu_ptr temp_scu2;
    while(temp_scu!=NULL)
    {
        temp_scu2=temp_scu->next_scu;
        delete temp_scu;
        temp_scu=temp_scu2;
    }
    delete temp_scu;
}

/*****add_scu*****/
/* Updates the linked list of SCUs on the branch. There are */
/* really two linked lists for SCUs. One that simply allows us to */
/* traverse the branch, another that links SCUs by their lineage */
/* (see function in SCU maintenance for new SCUs. */
/*****/

```

```

void branch::add_scu(SCU *newSCU,branch *br)
{
    br->scu_last->next_scu=newSCU;
    br->scu_last=newSCU;
    return ;
}

```

critCalc.cpp

```

/*****critCalc*****/
/* This file contains the functions that calculate criteria values */
/* for BRANCHPRO */
/*****/

#include "branch.h"
#include <math.h>
#include <stdio.h>
#include <iostream>
#include <fstream>

/*****run_calc*****/
/* Calculate the objective values for each run at the end of the */
/* current simulation. */
/*****/
void run_calc(crit *obj,branch *br, int pop, rb_obs *rbAll,int run)
{
    double meanTurns=0;
    double meanPath=0;
    double turns,terms,paths;
    double meanDiam=0;
    double junctions;

    double meanOver=0;
    turns=br->Nterm_turns;
    terms=br->Nterm;
    paths=br->Nterm_path;

    junctions=br->njunct;

    double load2,load23,load_sum;

    if(br->live_segs<MINSEGS)
    {
        meanTurns=1000;
        meanPath=1000;
        load2=1;
        load23=1;
        meanOver=1000;
        meanDiam=100000;
    }

    else
    {

```

```

if (terms>0)
{
    if (br->yDown==1 || br->live_segs>=MAXSEGS)
    {
        meanTurns=1000;
        meanPath=1000;

        load2=1;
        load23=1;
        meanOver=1000;
    }

    else
    {
        meanTurns=turns/terms;
        meanPath=paths/terms;

        load2=br->half_segs/(br->live_segs*8);
        load23=br->third_segs/(br->live_segs*27);
        meanOver=br->nover/br->live_segs;
    }
}
else
{
    meanTurns=1000;
    meanPath=1000;

    load2=1;
    load23=1;
    meanOver=1000;
}

if (junctions>0)
    meanDiam=br->diam_sum/junctions;
else
    meanDiam=1000000;
}

load2=pow(load2,0.25);
load23=pow(load23,0.25);
load_sum=load2+load23;

calc_rb(obj,rbAll,run);

// Assign the values for the other objectives
if (OBJTHEOR==0)
{
    obj->run[run][0]=br->live_segs;
    obj->run[run][1]=br->live_scus;
    obj->run[run][2]=br->length;

    if (PATH==1)
        obj->run[run][4]=meanPath;
    else
    {

```

```

        if (NTURNS==1)
        {
            obj->run[run][4]=meanTurns;
            obj->run[run][5]=meanPath;
            obj->run[run][6]=load_sum;
            obj->run[run][7]=meanOver;
            if (NEWLOAD==1)
                obj->run[run][8]=meanDiam;
        }
        else
        {
            obj->run[run][4]=meanPath;
            obj->run[run][5]=load_sum;
            obj->run[run][6]=meanOver;
            if (NEWLOAD==1)
                obj->run[run][7]=meanDiam;
        }
    }
}
else
{
    obj->run[run][0]=br->live_segs;
    if (NTURNS==1)
    {
        obj->run[run][1]=meanTurns;
        obj->run[run][2]=meanPath;
        obj->run[run][3]=load_sum;
        obj->run[run][4]=meanOver;
        if (NEWLOAD==1)
            obj->run[run][5]=meanDiam;
    }
    else
    {
        obj->run[run][1]=meanPath;
        obj->run[run][2]=load_sum;
        obj->run[run][3]=meanOver;
        if (NEWLOAD==1)
            obj->run[run][4]=meanDiam;
    }
}

if (SIMTYPE==1)
// results from sim studies of the Pareto frontier
{
    FILE *outfp;
    char plotout[50];
    char population[10];
    sprintf(population,"%d",pop);
    strcpy(plotout, "SimOutPop");
    strcat(plotout, population);
    strcat(plotout, ".txt\0");

    outfp=fopen(plotout,"a");

```



```

        {
            if(i<4)
                obj->final[pop][i]=sum[i]/NUMRUNS;
            else
                obj->final[pop][i]=1000;
        }
    else
    {
        if(i==0)
            obj->final[pop][i]=sum[i]/NUMRUNS;
        else
            obj->final[pop][i]=1000;
    }
}
printf("crit calcs (calculated means):%lf\n",
        obj->final[pop][i]);
}

/* here is for the median, CRIT_TYPE=1, let the binary parameter remain
the mean. First order the values, then pick out the 50th
percentile=ceil(NUMRUNS/2)*
    if(CRIT_TYPE==1||SIMTYPE==1)
    {
        for(i=0;i<NUMCRIT;i++)
        {
            for(j=0;j<NUMRUNS;j++)
            {
                for(k=0;k<NUMRUNS-1;k++)
                {
                    if(obj->run[k][i]>obj->run[k+1][i])
                        swap_array(obj->run[i],k,k+1);
                }
            }
        }
    }
}

/* If there is only one run per individual, then that value is the
value for the objective*/
else
    for(i=0;i<NUMCRIT;i++) obj->final[pop][i]=obj->run[0][i];
}

/*****crit_out*****/
/* Output criteria values to the critout file */
/*****
void crit_out(crit *obj,int pop,branch *br)
{
    FILE *fout;
    fout=fopen("critout.txt","a");

    int j;
    if(PBREAK==0)
    {
        if(HALFSAT==0)

```



```

        fprintf(fout,"%lf\t%lf\t%lf\t%lf\t%lf\t%lf",br->Rb0,br-
>Rb1,
                br->Rb2,br->Rb3,br->Rb4,br->p0);
    else
        fprintf(fout,"%lf\t%lf\t%lf\t%lf\t%lf\t%lf\t%lf\t%lf",
                br->Rb0,br->Rb1,br->Rb2,br->Rb3,br->Rb4,br->sat1,
                br->sat2,br->sat3);
    }
else
{
    if(HALFSAT==1)
        fprintf(fout,"%lf\t%lf\t%lf\t%lf\t%lf\t%lf\t%lf\t%lf",br->Rb0,
                br->Rb1,br->Rb2,br->Rb3,br->Rb4,br->sat1,
                br->sat2,br->sat3);
    else
    {
        if(NEWFUN>=2)
            fprintf(fout,"%lf\t%lf\t%lf\t%lf\t%lf\t%lf\t%lf\t%lf\t
                %lf\t%lf",br->Rb0,br->Rb1,br->Rb2,br->Rb3,
                br->Rb4,br->p0,br->pForm,br->pInact,br->pScu,
                br->pStem);
        else
            fprintf(fout,"%lf\t%lf\t%lf\t%lf\t%lf\t%lf\t%lf\t%lf\t
                %lf\t%lf\t%lf\t%lf\t%lf\t%lf\t%lf\t%lf",
                br->Rb0,br->Rb1,br->Rb2,br->Rb3,br->p0,br->pForm,
                br->pInact,br->pScu,br->pStem);
    }
}

for(j=0;j<NUMCRIT;j++)
    fprintf(fout,"\t%lf",obj->final[pop][j]);
fprintf(fout,"\n");

fclose(fout);
}

/*****calc_rb*****/
/* Calculate mean bifurcations for order 1, order 2 and order 3 */
/* shoots averaged through all of the years. Then assign the value */
/* for the fourth criteria = 1 if Rb's descend with order, 0 */
/* otherwise. */
/*****/

void calc_rb(crit *obj,rb_obs *rbAll,int run)
{
    int j,k;
    double sumRb[3];
    double denomRb=0;
    double ratioRb[3];

// First initialize the sums to zero
for(j=0;j<3;j++)
{
    sumRb[j]=0;
    ratioRb[j]=0;

```

```

    }

// Calculate the means
for(k=0;k<MAXYR;k++)
{
    if(rbAll->rb_all[k][0]>=0) denomRb=denomRb+1;
    for(j=0;j<3;j++)
    {
        if(rbAll->rb_all[k][j]>=0)
            sumRb[j]=sumRb[j]+rbAll->rb_all[k][j];
    }
}

if(denomRb>0)
{
    for(j=0;j<3;j++)
        ratioRb[j]=sumRb[j]/denomRb;
}

else
{
    for(j=0;j<3;j++) ratioRb[j]=0;
}

/* Determine if the characteristic architectural model is observed for
this branch*/
if(ratioRb[0]>ratioRb[1])
{
    if(ratioRb[1]>ratioRb[2]) obj->run[run][3]=1;
    else obj->run[run][3]=0;
}
else obj->run[run][3]=0;
}

```

misc.cpp

```

/*****misc functions*****/
/* This file was created to hold non-branch growth functions. */
/*****
#include "branch.h"

void swap_array(double crit[], int a, int b)
{
    /* This function swaps two values in an array--to sort in ascending
order for the median calculation*/
    double temp=crit[a];
    crit[a]=crit[b];
    crit[b]=temp;
}

```

ranNum.cpp

```

/*****
/* This file includes the code for generating random numbers for */
/* the linked branch model. All code is taken from Numerical */
/* Recipes in C. */
/*****

```

```

#include "ran.h"
#include "branch.h"

#include <cmath>

/*****ran3 *****/
/* From Numerical Recipes in C, 2nd ed, pg. 283. Returns a */
/* uniform (0,1) random deviate. Uses "ranseed", an integer seed. */
/*****/

double ran3(long *idum)
/* returns a uniform random deviate from 0-1. Set idum to any negative
value to initialize or reinitialize the sequence*/
{
    static int inext,inextp;
    static long ma[56];
    static int iff=0;
    long mj,mk;
    int i,ii,k;

    if(*idum<0||iff==0)
    {
        iff=1;
        mj=labs(MSEED-labs(*idum));
        mj %= MBIG;
        ma[55]=mj;
        mk=1;

        for(i=1;i<=54;i++)
        {
            ii=(21*i) % 55;
            ma[ii]=mk;
            mk=mj-mk;
            if(mk<MZ) mk+=MBIG;
            mj=ma[ii];
        }

        for(k=1;k<=4;k++)
            for(i=1;i<=55;i++)
            {
                ma[i] -= ma[1+(i+30) % 55];
                if (ma[i]<MZ)ma[i]+=MBIG;
            }

        inext=0;
        inextp=31;
        *idum=1;
    }

    if (++inext==56) inext=1;
    if (++inextp==56) inextp=1;
    mj=ma[inext]-ma[inextp];
    if (mj<MZ) mj+=MBIG;
    ma[inext]=mj;
}

```

```

        return mj*FAC;

} // end ran3

/*****ran2*****/
/* From Numerical recipes in C, pp 282. Returns a uniform (0,1) */
/* random deviate. Utilizes a non-integer seed. */
/* */
/*****/
double ran2(long *idum)
{
    int j;
    long k;
    static long idum2=123456789;
    static long iy=0;
    static long iv[NTAB];
    double temp;

    if (*idum<=0)
    {
        if (-(*idum) <1) *idum=1;
        else *idum = -(*idum);
        idum2=(*idum);

        for (j=NTAB+7;j>=0;j--)
        {
            k=(*idum)/IQ1;
            *idum=IA1*(*idum-k*IQ1)-k*IR1;
            if(*idum<0) *idum+=IM1;
            if (j<NTAB) iv[j]=*idum;
        }

        iy=iv[0];
    }

    k=(*idum)/IQ1;
    *idum=IA1*(*idum-k*IQ1)-k*IR1;
    if(*idum<0) *idum+=IM1;
    k=idum2/IQ2;
    idum2=IA2*(idum2-k*IQ2)-k*IR2;
    if(idum2<0) idum2+=IM2;
    j=iy/NDIV;
    iy=iv[j]-idum2;
    iv[j]=*idum;
    if(iy<1) iy+=IMM1;

    if((temp=AM*iy)>RNMX)
    {
        return RNMX;
    }

    else

```

```

    {
        return temp;
    }
} // end ran2

/***** poisdev *****/
/* From Numerical Reciped in C, pp 294. Returns a poisson */
/* random deviate with "xm" as the rate parameter. */
/*****/
double poisdev(double xm, long *idum)
/* returns as a floating point number an integer value that is a random
deviate drawn from a Poisson distribution with lambda=xm, using ran3 as
a source of uniform random deviates; uses rejection method as outlined
in book */
{
    static double sq,alxm,g,oldm=(-1,0);
    double em,t;//,y;

    if (xm<12.0)
    {
        if (xm!=oldm)
        {
            oldm=xm;
            g=exp(-xm);
        }

        em=-1;
        t=1.0;
        do
        {
            ++em;
            if (STOCH==0)
                t*=ran3(idum);
            else
                t*=ran2(idum);
        }
        while (t>g);
    }

    return em;
} // end poisdev

```

SCUmaint.cpp

```

/*****SCUmaint*****/
/* This file includes growth and maintenance functions for */
/* individual SCUs */
/* */
/* March 1, 2005 MCK */
/* Modified July 6, 2005 MCK */
/*****/

#include "branch.h"
#include "ran.h"

```

```

#include <iostream>
#include <fstream>
#include <cstdio>
#include <cmath>
#include <cstdlib>
#include <cstring>
#include <malloc.h>
#include <ctime>

using namespace std;

/*****scu_rekurs*****/
/* A function to navigate the linked list of SCUs on the branch. */
/* Each SCU has two pointers, one to its first "son" SCU, then one */
/* to any "sibling" SCUs, i.e. SCUs that arrive on the same main */
/* axis after this one has. */
/* */
/* The flag parameter indicates what calculation is to be performed:*/
/* flag1=0: normal growth functions, tally live SCUs for the branch */
/* and return that value. */
/* flag1=1: end of the run, free the space for the segs */
/* flag1=2: tally the total foliated shoots for the branch */
/* flag1=3: tally the # of sons from order 1 shoots */
/* flag1=4: tally the # of sons from order 2 shoots */
/* flag1=5: tally the # of sons from order 3 shoots */
/* flag1=6: tally the # of order 1 active shoots the previous yr */
/* flag1=7: tally the # of order 2 active shoots the previous yr */
/* flag1=8: tally the # of order 3 active shoots the previous yr */
/* flag1=9: sum the # turns to foliated terminal shoots */
/* flag1=10: sum the # of terminal foliated shoots */
/* */
/* Called by year_setup */
/* Calls shoot_rekurs */
/*****/

void SCU::scu_rekurs(SCU *cur_scu, double fol_value, int flag1,
                    double yr, long *seed, branch *br, int pop)
{
    seg_ptr temp;
    int i=1;
    int scu_idx=1;
    while(cur_scu!=NULL)
    {
        switch(flag1)
        {
            case(1):
// First reset all of the tallies
                cur_scu->ord1=0;
                cur_scu->ord2=0;
                cur_scu->ord3=0;

```

```

    cur_scu->ord1_prev=0;
    cur_scu->ord2_prev=0;
    cur_scu->ord3_prev=0;

    cur_scu->term_turns=0;
    cur_scu->term_num=0;
    cur_scu->term_path=0;

    cur_scu->maxY=0;

    cur_scu->nover=0;

    cur_scu->SCUnum=scu_idx;

    cur_scu->njunct=0;
    cur_scu->diam_sum=0;

    /* Place at the beginning of the shoot linked list for the current SCU
    and navigate the shoot linked list to perform growth functions*/
    temp=cur_scu->shoot_head;
    shoot_rekurs(temp,cur_scu,yr,seed,br,scu_idx,pop);

    // Define a "dead" SCU as one with less than 5 foliated shoots
    if(cur_scu->liveshoots<=5) cur_scu->mort=1;
    if(cur_scu->mort==0) br->live_scus=br->live_scus+1;
    // This is a tally of live (foliated) SCUs

    br->live_segs=br->live_segs+cur_scu->liveshoots;
    br->nsegs1=br->nsegs1+cur_scu->ord1;
    br->nsegs2=br->nsegs2+cur_scu->ord2;
    br->nsegs3=br->nsegs3+cur_scu->ord3;
    br->nsegs_prev1=br->nsegs_prev1+cur_scu->ord1_prev;
    br->nsegs_prev2=br->nsegs_prev2+cur_scu->ord2_prev;
    br->nsegs_prev3=br->nsegs_prev3+cur_scu->ord3_prev;
    if(cur_scu->mort!=1&&cur_scu->maxY>br->length)
        br->length=cur_scu->maxY;
    break;

case(2): // tally values for the objectives
    if(cur_scu->mort!=1)
    {
        br->Nterm_turns=br->Nterm_turns+cur_scu->term_turns;
        br->Nterm=br->Nterm+cur_scu->term_num;
        br->Nterm_path=br->Nterm_path+cur_scu->term_path;

        if (SIMTYPE==1&&yr==MAXYR)
        {
            char population[10];
            sprintf(population,"%d",pop);
            FILE *scufp;
            char scuout[50];
            strcpy(scuout, "SimOutSCUPop");
            strcat(scuout, population);
            strcat(scuout, ".txt\0");

```

```

        scufp=fopen(scuout,"a");

        fprintf(scufp,"%d\t%d\t%d\t%lf\t%lf\n",i,
            cur_scu->liveshoots,cur_scu->numscus,
            cur_scu->shoot_head->gen,
            cur_scu->shoot_head->age);

        fclose(scufp);
        i=i+1;
    }
}

if(LOAD2==0) //So, this is the original load calculation
{
    if((cur_scu->shoot_head->y1>(br->length/2))&&
        (cur_scu->mort!=1))
    {
        br->half_segs=br->half_segs+cur_scu->liveshoots;

        if(cur_scu->shoot_head->y1>(2*br->length/3))
            br->third_segs=br->third_segs+cur_scu->liveshoots;
    }
}
else
{
    if((cur_scu->shoot_head->gen==2)&&(cur_scu->mort!=1))
/* MCK modify load calculation 3/20/07.  only count live_lin of
insertions to the generation 1 axis-not included in dissertation*/
    {
        if(cur_scu->shoot_head->y1>(br->length/2))
        {
            cur_scu->shoot_head->lin=0;  // MCK added 03/20/07
            cur_scu->shoot_head->live_lin=0;
// MCK added 03/20/07
            int space=count_lin(cur_scu->shoot_head,
                                cur_scu->shoot_head,0);
/* MCK added 03/20/07--count the live_lin of an SCU base that is second
generation, so inserts in the generation 1 axis*/
            br->half_segs=
                br->half_segs+cur_scu->shoot_head->live_lin;
// MCK added 03/20/07
            if(cur_scu->shoot_head->y1>(2*br->length/3))
// MCK added 03/20/07
                br->third_segs=
                    br->third_segs+cur_scu->shoot_head->live_lin;
// MCK added 03/20/07
        }
    }
}
if((cur_scu->shoot_head->gen==1)&&(cur_scu->mort!=1))
{
    br->half_segs=br->half_segs+cur_scu->liveshoots;
    br->third_segs=br->third_segs+cur_scu->liveshoots;
}

```



```

        br->nover=br->nover+cur_scu->nover;

        br->njunct=br->njunct+cur_scu->njunct;
        br->diam_sum=br->diam_sum+cur_scu->diam_sum;
        break;

    case(3): // free the memory
        br->free_segs(cur_scu->shoot_head);
        break;

    default:
        cout<<"wrong flag in scu_rekurs\n";
        break;

    }

    cur_scu=cur_scu->next_scu;
    scu_idx=scu_idx+1;
}
if(yr==90)
{
    br->shoots90=br->live_segs;
    br->SCUs90=br->live_scus;
}
return ;
}

/*****first_scu*****/
/* A function to initialize and fill in values for the first SCU */
/* on the branch, including growth of the first four segments. */
/* */
/* Called by branch_initialize */
/* Calls shoot_extension, shoot_lateral, make_buds, shoot_fol */
/*****/

void SCU::first_scu(SCU *scu1,long *seed,branch *br)
{
    // Allocate memory for the first shoot on the branch
    seg_ptr newseg;
    newseg=new segs;

    // Initialize values for the first SCU
    scu1->shoot_head=newseg;
    scu1->next_scu=NULL;
    scu1->son1=NULL;
    scu1->mort=0;
    scu1->term_num=1;
    scu1->term_turns=0;

    scu1->term_path=0;

    scu1->numscus=0;

```

```

scul->ord1=0;
scul->ord2=0;
scul->ord3=0;

scul->ord1_prev=0;
scul->ord2_prev=0;
scul->ord3_prev=0;

scul->Rb0=br->Rb0;
scul->Rb1=br->Rb1;
scul->Rb2=br->Rb2;
scul->Rb3=br->Rb3;

// if (NEWFUN>=2)
scul->Rb4=br->Rb4;
scul->sat1=br->sat1;
scul->sat2=br->sat2;
scul->sat3=br->sat3;

scul->p0=br->p0;
scul->pForm=br->pForm;
scul->pInact=br->pInact;
scul->pScu=br->pScu;
scul->pStem=br->pStem;

// Initialize values for the first shoot
newseg->active=0;
newseg->epi=0;
newseg->mort=0;
newseg->init=0;
newseg->scu_base=1;
newseg->syll=0;

newseg->year=1;
newseg->gen=1;
newseg->age=0;
newseg->dorage=18;
newseg->order=1;
newseg->nson=3;

newseg->length=3.58;
newseg->theta=PI/2;

newseg->x1=0;
newseg->x2=0;
newseg->y1=0;
newseg->y2=newseg->length;
newseg->z1=0;
newseg->z2=0;

newseg->lin=1;
newseg->live_lin=1;
newseg->distout=newseg->length;
newseg->path_length=newseg->length;

```

```

newseg->dist_turn=newseg->length;
newseg->dist_base=newseg->length;

newseg->next=NULL;
newseg->son=NULL;
newseg->epi_son=NULL;
newseg->par=NULL;
newseg->br_next=NULL;
newseg->arr_next=NULL;

newseg->par_scu=scu1;

newseg->nturns=newseg->gen+newseg->order-2;
newseg->tform=1;

newseg->nover=0;

newseg->junct_base=0;
newseg->diam=0;
newseg->axis_length=0;

newseg->xll=newseg->x1-NEEDLE_LENGTH;
newseg->xul=newseg->x1+NEEDLE_LENGTH;
newseg->xlr=newseg->x1+NEEDLE_LENGTH;
newseg->xur=newseg->x1+NEEDLE_LENGTH;

newseg->yll=newseg->y1;
newseg->yul=newseg->y2;
newseg->y1r=newseg->y1;
newseg->y2r=newseg->y2;

newseg->xmid=
    floor(XSIZE/2+newseg->x1+cos(newseg->theta)*newseg->length/2);
newseg->ymid=
    floor(YSHIFT+newseg->y1+sin(newseg->theta)*newseg->length/2);

int idx,idy;
idx=newseg->xmid;
idy=newseg->ymid;

br->br_array[idx][idy]=newseg;

br->shoot_head=newseg;
br->shoot_last=newseg;

calc_lines(newseg);
shoot_fol(scu1,newseg,1,br);
newseg->segNum=br->nsegs;

// Update the first SCU to include the first shoot
scu1->numshoots=1;
scu1->liveshoots=1;
scu1->maxY=newseg->y2;

```

```

// Give the first shoot three sons, and assign their buds
scul->shoot_extension(newseg,2,br->length);
scul->liveshoots=scul->liveshoots+1;
scul->numshoots=scul->numshoots+1;

newseg->br_next=newseg->son;
br->shoot_last=newseg->son;
br->nsegs=br->nsegs+1;
newseg->son->segNum=br->nsegs;

shoot_fol(scul,newseg->son,1,br);
update_array(br,newseg->son);
idxOver_loop(br,newseg->son,1,scul);

scul->shoot_lateral(newseg,newseg->son,1,2,br->length);
scul->liveshoots=scul->liveshoots+1;
scul->numshoots=scul->numshoots+1;

newseg->son->br_next=newseg->son->next;
br->shoot_last=newseg->son->next;
br->nsegs=br->nsegs+1;
newseg->son->next->segNum=br->nsegs;

shoot_fol(scul,newseg->son->next,1,br);

update_array(br,newseg->son->next);
idxOver_loop(br,newseg->son->next,1,scul);

scul->shoot_lateral(newseg,newseg->son->next,2,2,br->length);
scul->liveshoots=scul->liveshoots+1;
scul->numshoots=scul->numshoots+1;

newseg->son->next->br_next=newseg->son->next->next;
br->shoot_last=newseg->son->next->next;
br->nsegs=br->nsegs+1;
newseg->son->next->next->segNum=br->nsegs;

shoot_fol(scul,newseg->son->next->next,1,br);

update_array(br,newseg->son->next->next);
idxOver_loop(br,newseg->son->next->next,1,scul);

scul->shoot_last=newseg->son->next->next;

scul->dout=newseg->distout;
scul->scu_yr=1;
scul->yr=1;

newseg->active=0;

}

```

```

/*****count_lin*****/
/* A function to tally the direct lineage of a single shoot. */
/* This is used to determine whether a new SCU has sufficient */
/* shoots to be independent of its parent axis. */
/* */
/* This function also determines whether the terminal first order */
/* shoot of the structure is active */
/*****/
int SCU::count_lin(segs *seg_temp,segs *seg_cur,int ind)
{
    while(seg_temp!=NULL)
    {
        if(seg_temp->son!=NULL)
        {
            count_lin(seg_temp->son,seg_cur,ind);
        }

        else if(seg_temp->order==1&&seg_temp->active==1)
        {
            ind=1; // If the terminal first order shoot is active, return 1
        }

        if(seg_temp->epi_son!=NULL)
        {
            count_lin(seg_temp->epi_son,seg_cur,ind);
        }

        if(seg_temp->mort==0)
            seg_cur->live_lin++;
        if(seg_temp->order==seg_cur->order)
            seg_cur->axis_length=seg_cur->axis_length+seg_temp->length;
        seg_cur->lin++;
        seg_temp=seg_temp->next;
    }
    return(ind);
}

/*****make_buds*****/
/* This function determines the number of buds out of three that */
/* will sucessfully extend into shoots the next year. This is a */
/* number drawn from truncated poisson distribution, whose mean */
/* differs with order and generation of the parent shoot. */
/* */
/* The observed mean number of sons will be less than the rate */
/* parameter calculated here due to the limit of three sons per */
/* parent (excluding epicormic daughter shoots on order 1 parents). */
/* */
/* For example: RB=3      RB=2      RB=1.5RB=1.0 RB=0.5 */
/* Mean sons:  2.33      1.78      1.41    0.98    0.50 */
/* */
/* called by first_scu, make_shoots */

```

```

/*****/

void SCU::make_buds(segs *cur_seg,int flag,long *seed,
                   SCU *par_scu,double damage_prob)
{
    double num_sons,rb;
    double test=1;
    /* to test for damage, if test~unif(0,1)<prob(damage), then the axis is
    damaged*/
    int damage_flag=0;
    int fun;

    int hydr_flag=0;
    if(cur_seg->order==1)
        hydr_flag=1;
    /* For the new function, only apply the hydraulic effect to first order
    main axes*/
    fun=NEWFUN;
    switch(fun)
    {
    case(0):
        rb=par_scu->Rb0+par_scu->Rb1*(1-1/cur_seg->order)+
            par_scu->Rb2*(1-1/(1*cur_seg->nturns+1*
            cur_seg->tform))/cur_seg->dist_base+par_scu->Rb3*
            (1-1/(1+cur_seg->nover));
        break;
    case(1):
        /* This is the function to enter the second round of optimizations,
        modified after the investigation of parameter search ranges for the
        function above. Half-saturation constants were added to account for the
        scale of the independent variables, and the nturns variable is only
        applied for order1 axes*/
        rb=par_scu->Rb0+par_scu->Rb1*
            (1-1/cur_seg->order)+hydr_flag*par_scu->Rb2*
            ((cur_seg->gen+cur_seg->tform)/
            (9+cur_seg->gen+cur_seg->tform))/cur_seg->dist_base+
            par_scu->Rb3*(cur_seg->nover/(8+cur_seg->nover));
        break;
    case(2):
        rb=par_scu->Rb0+par_scu->Rb1*
            (1-1/cur_seg->order)+hydr_flag*par_scu->Rb2*
            (cur_seg->gen/(par_scu->sat1+cur_seg->gen))/
            cur_seg->dist_base+par_scu->Rb3*(cur_seg->nover/
            (par_scu->sat2+cur_seg->nover))+hydr_flag*
            par_scu->Rb4*(cur_seg->tform/(par_scu->sat3+
            cur_seg->tform))/cur_seg->dist_base;
        break;
    case(3):
        rb=par_scu->Rb0+par_scu->Rb1*
            (1-1/cur_seg->order)+hydr_flag*par_scu->Rb2*
            (cur_seg->gen)/cur_seg->dist_base+par_scu->Rb3*
            (cur_seg->nover/(8+cur_seg->nover))+hydr_flag*
            par_scu->Rb4*(cur_seg->tform/(5+cur_seg->tform))/
            cur_seg->dist_base;
        break;
    }
}

```

```

case(4):
    rb=par_scu->Rb0+par_scu->Rb1*
        (1-1/cur_seg->order)+hydr_flag*par_scu->Rb2*
        (cur_seg->gen/(1+cur_seg->gen))/cur_seg->dist_base+
        par_scu->Rb3*(cur_seg->nover/
            (8+cur_seg->nover))+hydr_flag*par_scu->Rb4*
        (cur_seg->tform/(5+cur_seg->tform))/cur_seg->dist_base;
    break;
default:
    cout<<"wrong function in make_buds\n";
    break;
}

if(rb<0)
    rb=0;
if(DAMAGE==1)
{
    if(hydr_flag==1)
    {
        test=ran2(seed);
        if(test<damage_prob)
        {
            rb=0;
            damage_flag=1;
        }
    }
}

if(SYLL==0) // deterministic regular main axis
{
    if((yr<50)&&(cur_seg->epi==0))
        num_sons=3;
    else
        num_sons=poisdev(rb,seed);
}

else
/* SYLL==1, so allow sylleptic reiteration on first generation main
axis only*/
{
    num_sons=poisdev(rb,seed);
    if(cur_seg->order==1)
    {
        if((cur_seg->epi==0)&&(num_sons==0))
        {
            if(cur_seg->next!=NULL)//&&br.syll_idx<=3)
            {
                cur_seg->next->syll=1;
                //(flag indicating to make any son of this shoot a first order shoot
            }
        }
    }
}
if(num_sons>3)

```

```

        num_sons=3;

        int ord;
        cur_seg->nson=num_sons;
        ord=cur_seg->order;
    }

    /*****test_epini*****/
    /* Any node along the first order axis is available for epicormic */
    /* sprouting, the probability of which will change with an      */
    /* assessment of surrounding conditions                            */
    /*****

void SCU::test_epini(segs *cur_seg, SCU *par_scu, int test, double yr,
                    long *seed, branch *br)
{
    segs *temp_seg;
    int side;
    double p_break, t_form, prop_inact, dist_scu, test_prob;
    /* the parameters that influence probability of sprouting.
    p_break will be the probability of sprouting.
    t_form is the time since formation (i.e. age of parent shoot)
    prop_inact is the proportion of neighboring nodes that are inactive.
    This can be 0, 1/3, 2/3, 1 (each subtended lateral axis and the
    corresponding main axis)*/

    if(PBREAK==1)
    {
        if(cur_seg->age!=0)
            t_form=cur_seg->age;
        else t_form=1;
        prop_inact=0;    // Initialize as zero
        p_break=0;       // Initialize as zero
        dist_scu=1;

        if(cur_seg->son!=NULL)
        // to tally the neighboring inactive axes and update prop_inact
        {
            temp_seg=cur_seg->son;
            if(temp_seg->next!=NULL)
            {
                temp_seg=temp_seg->next;
                while(temp_seg->son!=NULL)
                {
                    temp_seg=temp_seg->son;
                }
                if(temp_seg->active==0)
                    prop_inact=prop_inact+0.3333;
                if(cur_seg->son->next->next!=NULL)
                {
                    temp_seg=cur_seg->son->next->next;
                    while(temp_seg->son!=NULL)
                    {

```



```

        temp_seg=temp_seg->son;
    }
    if(temp_seg->active==0)
        prop_inact=prop_inact+0.33333;
    }
}

temp_seg=cur_seg->son;
while(temp_seg->son!=NULL)
{
    temp_seg=temp_seg->son;
}
if(temp_seg->active==0)
    prop_inact=prop_inact+0.33333;
}
else prop_inact=prop_inact+0.3333;

int cnta=1;
int stop;
if(cur_seg->son!=NULL)
{
    temp_seg=cur_seg->son;
    if(temp_seg->son!=NULL)
    {
        if(temp_seg->epi_son==NULL)
            stop=0;
        else
            stop=1;

        while(stop==0)
        {
            cnta=cnta+1;
            temp_seg=temp_seg->son;
            if(temp_seg->epi_son!=NULL)
                stop=1;
            if(temp_seg->son==NULL)
                stop=1;
        }
    }
}

int cntb=20;
temp_seg=cur_seg->par;
stop=0;
if(temp_seg!=NULL)
{
    if(cur_seg->gen>1)
    {
        while(temp_seg->epi_son==NULL&&temp_seg->par->init!=1)
        {
            cntb=cntb+1;
            temp_seg=temp_seg->par;
        }
    }
    else if(temp_seg->par!=NULL)

```

```

    {
        while(temp_seg->epi_son==NULL&&temp_seg->par!=NULL)
        {
            cntb=cntb+1;
            temp_seg=temp_seg->par;
        }
    }

    if(cnta<=cntb)
        dist_scu=cnta;
    else
        dist_scu=cntb;

/* The function below is the function that entered the first rounds of
optimizations*/

    if(SCUEPI==0)
    {
        if(NEWFUN==0)
            p_break=par_scu->p0+par_scu->pForm*(1-1/t_form)+
                par_scu->pInact*prop_inact+par_scu->pScu*
                (1-1/dist_scu)+par_scu->pStem*
                (1/cur_seg->path_length);
/* This is the function that resulted from investigation of the
parameter search ranges, with the intention of fixing the effective
scale of the independent variables*/
        else
            p_break=par_scu->p0+par_scu->pForm*(t_form/(10+t_form))+
                par_scu->pInact*prop_inact+
                par_scu->pScu*(dist_scu/(15+dist_scu))+
                par_scu->pStem*(1-
                cur_seg->path_length/(100+cur_seg->path_length));
    }
    else
    {
        if(cur_seg->gen==par_scu->shoot_head->gen)
        {
            if(NEWFUN==0)
                p_break=par_scu->p0+par_scu->pForm*(1-1/t_form)+
                    par_scu->pInact*prop_inact+par_scu->pScu*
                    (1-1/dist_scu)+par_scu->pStem*
                    (1/cur_seg->path_length);
/* This is the function that resulted from investigation of the
parameter search ranges, with the intention of fixing the effective
scale of the independent variables*/
            else
                p_break=par_scu->p0+par_scu->pForm*(t_form/(10+t_form))+
                    par_scu->pInact*prop_inact+
                    par_scu->pScu*(dist_scu/(15+dist_scu))+
                    par_scu->pStem*(1-
                    cur_seg->path_length/(100+
                    cur_seg->path_length));
        }
        else p_break=0;
    }

```



```

        cout<<"\nCOMPTEST: "<<cur_seg->epi_son->theta<<"\t"<<
        cur_seg->epi_son->length<<"\t"<<cur_seg->
        epi_son->interc[0][0]<<"\t"<<cur_seg->
        epi_son->xmid<<"\t"<<cur_seg->epi_son->ymid<<"\t"<<
        cur_seg->epi_son->x1<<"\t"<<cur_seg->
        epi_son->y1<<"\t"<<cur_seg->epi_son->year<<"\t"<<
        cur_seg->epi_son->slope[0]<<"\t"<<cur_seg->
        epi_son->interc[0][0]<<"\n";
    }
    else
        cur_seg->epi_son=NULL;
}

/*****make_epini*****/
/* Fills in information for a newly sprouted epicormic shoot.      */
/*                                                                    */
/* called by test_epini                                             */
/*****

void SCU::make_epini(segs *cur_seg,int side,SCU *par_scu,double yr,
                    long *seed,double t_form,double length)
{
    seg_ptr temp_epi;
    temp_epi=new segs;
    par_scu->liveshoots++;
    par_scu->numshoots++;

    cur_seg->epi_son=temp_epi;

    temp_epi->next=NULL;
    temp_epi->son=NULL;

    temp_epi->epi_son=NULL;
    temp_epi->br_next=NULL;
    temp_epi->arr_next=NULL;

    temp_epi->par=cur_seg;
    temp_epi->par_scu=par_scu;

    temp_epi->scu_base=0;

    temp_epi->active=1;
    temp_epi->age=0;
    temp_epi->dorage=-1;
    temp_epi->epi=1;
    temp_epi->gen=cur_seg->gen+1;
    temp_epi->init=1;
    temp_epi->length=3.58;
    temp_epi->mort=0;
    temp_epi->nson=0;
    temp_epi->order=1;
    temp_epi->nson=0;
    if (side==0) temp_epi->theta=cur_seg->theta+0.05;
    else temp_epi->theta=cur_seg->theta-0.05;
}

```

```

temp_epi->theta2=temp_epi->theta+PI/2;
temp_epi->theta3=temp_epi->theta-PI/2;

temp_epi->year=yr;

temp_epi->lin=1;

temp_epi->live_lin=0;
temp_epi->syll=0;

temp_epi->distout=cur_seg->distout;

temp_epi->nturns=temp_epi->order+temp_epi->gen-2;
temp_epi->path_length=cur_seg->path_length+temp_epi->length;
temp_epi->dist_turn=temp_epi->length;
temp_epi->dist_base=temp_epi->length;

temp_epi->tform=t_form;

temp_epi->nover=0;

calc_coord(temp_epi,cur_seg);
calc_lines(temp_epi);
temp_epi->junct_base=0;
temp_epi->diam=0;
temp_epi->axis_length=0;
}

/*****test_scu*****/
/* For every epicormic shoot that represents the initiation of */
/* the epicormic complex, test whether it is ready to be a */
/* new/independent SCU. If so, create the new SCU and delete its */
/* shoots from the parent SCU (by reorienting the pointers in the */
/* linked list, disjoining this linked list from the parent. */
/* */
/* called by shoot_rekurs */
/* calls count_lin */
/*****/

int SCU::test_scu(segs *seg_head,segs *epi_par,SCU *par_scu,
                  branch *br)
{
    int flag=0;
    int ind=0;

    seg_ptr seg_temp;
    scu_ptr scu_temp;
    scu_ptr scu_hold1;
    scu_ptr scu_hold2;

    seg_temp=seg_head;
    seg_head->lin=0;
    seg_head->live_lin=0;
    seg_head->axis_length=0;

```

```

    ind=count_lin(seg_head,seg_head,0);
/* This initializes ind as 0 in the count_lin function, which tests
whether the terminal order 1 shoot is active. Counts the live shoots
from the lineage of the seg_head*/

    while(seg_temp!=NULL)
    {
        if(seg_temp->son==NULL&&seg_temp->active==1)
            ind=1;
        seg_temp=seg_temp->son;
    }

/* If the forming SCU in question has at least 10 shoots, and its
terminal order 1 axis is active, then make it an independent SCU*/
    if((seg_head->live_lin>10)&&(ind==1))
    {
        flag=1; // This SCU is independent
        scu_temp=new SCU; // Create a new SCU
        br->scu_bases=br->scu_bases+1;
/* count the number of epis that become independent SCUs throughout the
lifespan of the branch.*/

/* This next loop finds the place in the list for the new SCU. If it
is the first SCU from this axis, call it son1. If not, it is placed in
the list with its sibling SCU. This loop finds the place in the list
where the last sibling SCU was added (sibling meaning all from the same
parent axis) */
        scu_hold1=par_scu;
        while(scu_hold1!=NULL)
        {
            scu_hold2=scu_hold1;
            scu_hold1=scu_hold2->next_scu;
        }
        scu_hold2->next_scu=scu_temp;

/* Now we fill in the info for the new SCU and remove its shoots from
the parent SCU.*/

        scu_temp->shoot_head=seg_head;
        seg_head->scu_base=1;
        scu_temp->dout=scu_out(seg_head);
        scu_temp->next_scu=NULL;
        scu_temp->son1=NULL;

        par_scu->liveshoots=par_scu->liveshoots-seg_head->live_lin;
        par_scu->numshoots=par_scu->numshoots-seg_head->lin;

        par_scu->numscus=par_scu->numscus+1;

        scu_temp->liveshoots=seg_head->live_lin;
        scu_temp->numshoots=seg_head->lin;
        scu_temp->mort=0;
        scu_temp->numscus=0;

```

```

    scu_temp->ord1=0;
    scu_temp->ord2=0;
    scu_temp->ord3=0;
    scu_temp->ord1_prev=0;
    scu_temp->ord2_prev=0;
    scu_temp->ord3_prev=0;

    scu_temp->scu_yr=seg_head->year;
    scu_temp->term_num=0;
    scu_temp->term_turns=0;
    scu_temp->term_path=0;

    scu_temp->yr=yr;

    scu_temp->Rb0=par_scu->Rb0;
    scu_temp->Rb1=par_scu->Rb1;
    scu_temp->Rb2=par_scu->Rb2;
    scu_temp->Rb3=par_scu->Rb3;
    scu_temp->Rb4=par_scu->Rb4;
    scu_temp->sat1=par_scu->sat1;
    scu_temp->sat2=par_scu->sat2;
    scu_temp->sat3=par_scu->sat3;

    scu_temp->p0=par_scu->p0;
    scu_temp->pForm=par_scu->pForm;
    scu_temp->pInact=par_scu->pInact;
    scu_temp->pScu=par_scu->pScu;

    scu_temp->maxY=seg_head->y2;

    scu_lin(scu_temp,scu_temp->shoot_head->son,
            scu_temp->shoot_head,0);

}
return(flag);
// return the result of the test--flag=1, SCU is independent
}

/*****scu_lin*****/
/* change the pointer for the lineage of a newly independent SCU */
/* to the new SCU. */
/* */
/* Called by test_scu */
/*****/
void SCU::scu_lin(SCU *new_scu,segs *cur_seg,segs *prev_seg,int flag)
{
    while(cur_seg!=NULL)
    {
        if(flag==0)
            cur_seg->par_scu=new_scu;

        if(cur_seg->son!=NULL)
            scu_lin(new_scu,cur_seg->son,cur_seg,flag);
    }
}

```

```

        if (cur_seg->epi_son!=NULL)
            scu_lin(new_scu,cur_seg->epi_son,cur_seg,flag);

        prev_seg=cur_seg;
        cur_seg=cur_seg->next;
    }
    return ;
}

/*****shoot_rekurs*****/
/* A function to traverse the shoots associated with the current SCU. It satisfies some of the growth functions, including
/* SCU. It satisfies some of the growth functions, including
/* test_scu and make_shoots, and updates things like age and
/* mortality.
/*****/

void SCU::shoot_rekurs(segs *temp,SCU *par_scu,double yr,
                      long *seed,branch *br,int scu_idx, int pop)
{
    int ord;
    int flag,flg_syll;

    while(temp!=NULL)
    {
        flag=0;
        flg_syll=0;
        temp->age=yr-temp->year;

        /* To test if the current shoot is actually the base of a sylleptically
        reiterated SCU. If it is, exit the recursion, move on to the next one
        this one will be updated when we get to its new SCU.*/
        if(flgsyll==0)
        {
            temp->distout=par_scu->dout;

            if(temp->y2>par_scu->maxY) par_scu->maxY=temp->y2;

            if(temp->mort==0)
            {
                shoot_fol(par_scu,temp,yr,br);
                if(br->shoot_head==NULL)
                    br->shoot_head=temp;
                if(br->shoot_last!=NULL)
                    br->shoot_last->br_next=temp;
                br->shoot_last=temp;
            }
            temp->br_next=NULL;

            // if the extension exists, repeat recursion on it
            if(temp->son!=NULL)
                shoot_rekurs(temp->son,par_scu,yr,seed,br,scu_idx,pop);
            // This ensures that we traverse all of the nodes

```



```
/* If the epison exists, test whether it is already the base of a new
SCU or if it is time to become a new SCU. If not, repeat the recursion
on it.*/
```

```
    if(temp->epi_son!=NULL)
    {
        if(temp->epi_son->scu_base==1)
            flag=1;
        else if(temp->epi_son->age>10)
            flag=test_scu(temp->epi_son,temp,par_scu,br);
        if(flag==0)
            shoot_rekurs(temp->epi_son,par_scu,yr,seed,br,
                        scu_idx,pop);
    }
```

```
/* Here we grow the buds for all of the terminal shoots, and update the
values for the previous terminal counts to use in calculation of Rb.*/
```

```
    if(temp->active==1)
    {
        make_shoots(temp,0,par_scu,yr,seed,br);
        ord=temp->order;
        switch(ord)
        {
            case(1):
                if(temp->init!=1&&temp->par->init!=1)
                    par_scu->ord1_prev=par_scu->ord1_prev+1;
                break;
            case(2):
                par_scu->ord2_prev=par_scu->ord2_prev+1;
                break;
            case(3):
                par_scu->ord3_prev=par_scu->ord3_prev+1;
                break;
            default:
                break;
        }
    }
```

```
/* Test all order 1 axes for epicormic initiation if they haven't yet
sprouted their dormant bud*/
```

```
    if(temp->order==1)
    {
        if(temp->epi_son==NULL)
            test_epini(temp,par_scu,0,yr,seed,br);
    }

    if(temp->y1<(-YSHIFT))
        br->yDown=1;
```

```
// if the shoot is the base of a junction, calculate its diameter
```

```
    if(temp->junct_base==1&&temp->order==2)
    {
        calc_diam(temp);
        par_scu->njunct=par_scu->njunct+1;
```

```

        par_scu->diam_sum=par_scu->diam_sum+temp->diam;
    }

    // Move on to the next shoot
    temp=temp->next;
}

/*****make_shoots*****/
/* For active shoots, grow the daughters flagged to extend this */
/* year, determined by make_buds the previous year. */
/***** */

void SCU::make_shoots(segs *cur_seg,int flag,SCU *par_scu,
                     double yr,long *seed,branch *br)
{
    int side,ord;
    ord=cur_seg->order;
    double side_test;

    side_test=ran2(seed);
    if(side_test<0.5) side=0;
    else side=1;

    if(cur_seg->nson>0)
    {
        /* The switch is to update the # of sons from terminal shoots of a
        given order for the SCU*/
        switch(ord)
        {
            case(1):
                if(cur_seg->init!=1&&cur_seg->par->init!=1)
                    par_scu->ord1=par_scu->ord1+cur_seg->nson;
                break;
            case(2):
                if(cur_seg->par->init!=1)
                    par_scu->ord2=par_scu->ord2+cur_seg->nson;
                break;
            case(3):
                par_scu->ord3=par_scu->ord3+cur_seg->nson;
                break;
            default:
                break;
        }
    }

    /* For every new shoot, update for the SCU the value for the number of
    terminal shoots, turns to the terminal shoots, live shoots and total
    shoots*/

    // Extension only if nson is 1
    if(cur_seg->nson==1)
    {
        /* for shoot_extension, create a new shoot and have the current shoot
        header point to it*/
        shoot_extension(cur_seg,yr,br->length);
    }
}

```

```

// Extension +1 lateral shoot, whose side is randomly determined
else if(cur_seg->nson==2)
{
/* for shoot_extension, create a new shoot and have the current shoot
header point to it*/
shoot_extension(cur_seg,yr,br->length);

seg_ptr prev_seg;
prev_seg=cur_seg->son;
shoot_lateral(cur_seg,prev_seg,side,yr,br->length);
}

// Extension +2 lateral shoots, one on either side
else if(cur_seg->nson==3)
{
shoot_extension(cur_seg,yr,br->length);

seg_ptr prev_seg;
for(int j=1;j<=2;j++)
{
if(j==1) prev_seg=cur_seg->son;
else prev_seg=cur_seg->son->next;
shoot_lateral(cur_seg,prev_seg,j,yr,br->length);
}
}
}

if(cur_seg->son!=NULL)
{
shoot_fol(par_scu,cur_seg->son,yr,br);
update_array(br,cur_seg->son);
br->nsegs=br->nsegs+1;
cur_seg->son->segNum=br->nsegs;
idxOver_loop(br,cur_seg->son,1,par_scu);

if(br->shoot_head==NULL)
br->shoot_head=cur_seg->son;
if(br->shoot_last!=NULL)
br->shoot_last->br_next=cur_seg->son;
br->shoot_last=cur_seg->son;

par_scu->liveshoots=par_scu->liveshoots+1;
par_scu->numshoots=par_scu->numshoots+1;
par_scu->term_turns=cur_seg->son->nturns+par_scu->term_turns;
par_scu->term_num=par_scu->term_num+1;
par_scu->term_path=par_scu->term_path+cur_seg->son->path_length;

if(cur_seg->son->next!=NULL)
{
shoot_fol(par_scu,cur_seg->son->next,yr,br);
update_array(br,cur_seg->son->next);
br->nsegs=br->nsegs+1;
cur_seg->son->next->segNum=br->nsegs;
}
}

```

```

idxOver_loop(br, cur_seg->son->next, 1, par_scu);

if (br->shoot_head==NULL)
    br->shoot_head=cur_seg->son->next;
if (br->shoot_last!=NULL)
    br->shoot_last->br_next=cur_seg->son->next;
br->shoot_last=cur_seg->son->next;

par_scu->liveshoots=par_scu->liveshoots+1;
par_scu->numshoots=par_scu->numshoots+1;
par_scu->term_turns=
    cur_seg->son->next->nturns+par_scu->term_turns;
par_scu->term_num=par_scu->term_num+1;
par_scu->term_path=
    par_scu->term_path+cur_seg->son->next->path_length;

if (cur_seg->son->next->next!=NULL)
{
    shoot_fol(par_scu, cur_seg->son->next->next, yr, br);
    update_array(br, cur_seg->son->next->next);

    br->nsegs=br->nsegs+1;
    cur_seg->son->next->next->segNum=br->nsegs;
    idxOver_loop(br, cur_seg->son->next->next, 1, par_scu);

    if (br->shoot_head==NULL)
        br->shoot_head=cur_seg->son->next->next;
    if (br->shoot_last!=NULL)
        br->shoot_last->br_next=cur_seg->son->next->next;
    br->shoot_last=cur_seg->son->next->next;

    par_scu->liveshoots=par_scu->liveshoots+1;
    par_scu->numshoots=par_scu->numshoots+1;
    par_scu->term_turns=
        cur_seg->son->next->next->nturns+par_scu->term_turns;
    par_scu->term_num=par_scu->term_num+1;
    par_scu->term_path= par_scu->term_path+
        cur_seg->son->next->next->path_length;
}
}

cur_seg->active=0;
// The current shoot is no longer an active terminal shoot
}

/*****update_array*****/
/* update the branch array for the given midpoint coordinates */
/******/
void SCU::update_array(branch *br, seg_ptr cur_seg)
{
    seg_ptr temp;
    int idx, idy;

```

```

    idx=cur_seg->xmid;
    idy=cur_seg->ymid;
    if (br->br_array[idx][idy]!=NULL)
    {
        temp=br->br_array[idx][idy];
        while(temp->arr_next!=NULL)
// find the end of the list for this place in the array
        {
            temp=temp->arr_next;
        }
        temp->arr_next=cur_seg;
    }
    else
/* if the current place in the array is empty, then point it to the
current shoot*/
        br->br_array[idx][idy]=cur_seg;
}

/*****shoot_extension*****/
/* Fills information for a shoot that is an extension of the */
/* parent axis where the branching angle is the same as the */
/* parent.Also calls make_buds for the shoot. */
/*****/
void SCU::shoot_extension(segs *cur_seg,double yr,double length)
{
    int alpha=8;
    double iniage,beta=0.7;
    int ord;

    int i;

    if (cur_seg->order==1&&cur_seg->init!=1) iniage=0;
    else iniage=-1;

/* Don't allow dormant epicormic buds on lateral axes and on the first
two years of growth of a new epicormic.*/
    seg_ptr ext;
    ext=new segs;

    cur_seg->son=ext;

    ext->son=NULL;
    ext->epi_son=NULL;
    ext->next=NULL;
    ext->br_next=NULL;
    ext->arr_next=NULL;

    ext->active=1;
    ext->age=0;
    ext->dorage=iniage;
    ext->epi=cur_seg->epi;
    ext->gen=cur_seg->gen;
    ext->init=0;
    ext->length=cur_seg->length;
    ext->mort=0;

```

```

ext->nson=3;
if(cur_seg->syll==0)
    ext->order=cur_seg->order;
else
    ext->order=1;

if(cur_seg->syll==0)
    ext->theta=cur_seg->theta;
else
    ext->theta=PI/2;

ext->theta2=ext->theta+PI/2;
ext->theta3=ext->theta-PI/2;

calc_coord(ext,cur_seg);

ext->year=yr;
ext->distout=cur_seg->distout;
ext->path_length=cur_seg->path_length+ext->length;
ext->dist_turn=cur_seg->dist_turn+ext->length;
//this is the dist from the base of the SCU
ext->dist_base=cur_seg->dist_base+ext->length;

ext->lin=1;
ext->live_lin=1;
ext->scu_base=0;

ext->par=cur_seg;

ext->par_scu=cur_seg->par_scu;

ext->syll=0;

ext->nturns=ext->order+ext->gen-2;

ext->tform=cur_seg->tform;

ext->nover=0;

for(i=0;i<2;i++)
    ext->slope[i]=cur_seg->slope[i];

if(ext->theta!=PI/2)
{
    ext->interc[0][0]=ext->y1l-ext->slope[0]*ext->x1l;
    ext->interc[0][1]=ext->y1r-ext->slope[0]*ext->x1r;

    ext->interc[1][0]=ext->y1l-ext->slope[1]*ext->x1l;
    ext->interc[1][1]=ext->y1l-ext->slope[1]*ext->x1l;
}
else
{
    ext->interc[0][0]=-1;
    ext->interc[0][1]=-1;
}

```

```

        ext->interc[1][0]=ext->y1;
        ext->interc[1][1]=ext->y2;
    }

    ord=ext->order;

    ext->junct_base=0;
    ext->diam=0;
    ext->axis_length=0;
}

/*****calc_coord*****/
/* Calculates the x,y coordinates of the center of the shoot and */
/* the surrounding foliage rectangle.                               */
/*****

void SCU::calc_coord(seg_ptr cur_seg, seg_ptr par_seg)
{
    double theta_fix;
    cur_seg->x1= par_seg->x2;
    cur_seg->x2= cur_seg->x1+cos(cur_seg->theta)*cur_seg->length;
    cur_seg->y1= par_seg->y2;
    cur_seg->y2= cur_seg->y1+sin(cur_seg->theta)*cur_seg->length;
    cur_seg->z1=0;
    cur_seg->z2=0;

    cur_seg->xmid= floor(XSIZE/2+cur_seg->x1+cos(cur_seg->theta)*
                        cur_seg->length/2);
    cur_seg->ymid= floor(YSHIFT+cur_seg->y1+sin(cur_seg->theta)*
                        cur_seg->length/2);

    if(cur_seg->xmid<0) cur_seg->xmid=0;
    if(cur_seg->xmid>(XSIZE-1)) cur_seg->xmid=XSIZE-1;

    if(cur_seg->ymid<0)
    {
        cur_seg->ymid=0;
    }
    if(cur_seg->ymid>(YSIZE-1)) cur_seg->ymid=YSIZE-1;

    cur_seg->xll=cur_seg->x1+cos(cur_seg->theta2)*NEEDLE_LENGTH;
    cur_seg->xul=cur_seg->x2+cos(cur_seg->theta2)*NEEDLE_LENGTH;

    cur_seg->xlr=cur_seg->x1+cos(cur_seg->theta3)*NEEDLE_LENGTH;
    cur_seg->xur=cur_seg->x2+cos(cur_seg->theta3)*NEEDLE_LENGTH;

    cur_seg->yll=cur_seg->y1+sin(cur_seg->theta2)*NEEDLE_LENGTH;
    cur_seg->yul=cur_seg->y2+sin(cur_seg->theta2)*NEEDLE_LENGTH;

    cur_seg->y1r=cur_seg->y1+sin(cur_seg->theta3)*NEEDLE_LENGTH;
    cur_seg->yur=cur_seg->y2+sin(cur_seg->theta3)*NEEDLE_LENGTH;

    theta_fix=cur_seg->theta;
    if(cur_seg->theta>2*PI)

```

```

{
    if(cur_seg->theta>4*PI)
    {
        theta_fix=cur_seg->theta-4*PI;
    }
    else
        theta_fix=cur_seg->theta-2*PI;
}

if(cur_seg->theta<(-2*PI))
{
    if(cur_seg->theta<(-4*PI))
    {
        theta_fix=cur_seg->theta+4*PI;
    }
    else
        theta_fix=cur_seg->theta+2*PI;
}

if(theta_fix>0&&theta_fix<=PI/2)
    cur_seg->theta_quad=1;
if(theta_fix>(PI/2)&&theta_fix<=PI)
    cur_seg->theta_quad=2;
if(theta_fix>PI&&theta_fix<=(3*PI/2))
    cur_seg->theta_quad=3;
if(theta_fix>(3*PI/2)&&theta_fix<=(2*PI))
    cur_seg->theta_quad=4;

if(theta_fix<0&&theta_fix>=(-PI/2))
    cur_seg->theta_quad=4;
if(theta_fix<(-PI/2)&&theta_fix>=(-PI))
    cur_seg->theta_quad=3;
if(theta_fix<(-PI)&&theta_fix>=(-3*PI/2))
    cur_seg->theta_quad=2;
if(theta_fix<(-3*PI/2)&&theta_fix>=(-2*PI))
    cur_seg->theta_quad=1;
}

/*****shoot_lateral*****/
/* Fills information for a shoot that is lateral wrt the parent */
/* axis and the branching angle is offset +/- from the the */
/* parent. Also calls make_buds for the shoot. */
/*****/
void SCU::shoot_lateral(segs *cur_seg,segs *prev_seg,int side,
                        double yr,double length)
{
    seg_ptr lat;
    lat=new segs;
    prev_seg->next=lat;

    lat->next=NULL;
    lat->son=NULL;
    lat->epi_son=NULL;
    lat->br_next=NULL;

```



```

lat->arr_next=NULL;

lat->active=1;
lat->age=0;
lat->dorage=-1;
lat->epi=cur_seg->epi;
lat->gen=cur_seg->gen;
lat->init=0;
lat->length=cur_seg->length*0.86;
lat->mort=0;
lat->nson=3;
lat->order=cur_seg->order+1;
lat->nson=0;
if(side==1) lat->theta=cur_seg->theta+1;
else lat->theta=cur_seg->theta-1;

lat->theta2=lat->theta+PI/2;
lat->theta3=lat->theta-PI/2;

calc_coord(lat,cur_seg);
lat->distout=cur_seg->distout;
lat->path_length=cur_seg->path_length+lat->length;
lat->dist_turn=lat->length;
lat->dist_base=lat->length+cur_seg->dist_base;
lat->lin=1;
lat->live_lin=1;
lat->scu_base=0;

lat->par=cur_seg;
lat->par_scu=cur_seg->par_scu;
lat->year=yr;

lat->syll=0;

lat->nturns=lat->order+lat->gen-2;
lat->tform=1;

lat->nover=0;

calc_lines(lat);

lat->junct_base=1;
lat->diam=0;
lat->axis_length=0;
}

/*****scu_out recursion*****/
/* Calculates the distance (path length) from the base of the SCU */
/* to the base of the branch. This distance is used as a */
/* predictor of foliage values. */
/* */
/* Called by add_scu() */
/*****

```

```

double SCU::scu_out(segs *cur_seg)//,double d,int a)
{
    double d=0;
    segs *temp;
    temp=cur_seg->par;

    while(temp->scu_base==0)
    {
        d=d+temp->length;
        temp=temp->par;
    }
    d=d+temp->distout;

    return (d);
} // end scu_out

/*****shoot_fol*****/
/* To calculate foliage area and dry wt for a shoot given */
/* significant regression relationships */
/* */
/*****/
void SCU::shoot_fol(SCU *par_scu,segs *cur_seg,double yr,branch *br)
{
    int crown=3;
    double sna,wtl;

    if (yr<150)
    {
        if(yr<90) crown=1;
        else crown=2;
    }

    switch (crown)
    {
    case(1):
        if(cur_seg->age<5) wtl=pow((8.1743+.03203*cur_seg->age),2);
        else
            wtl=pow((8.1743+.03203*cur_seg->age-1.7214*
                (cur_seg->age-4)),2);
        sna=pow((7.5715-.1611*cur_seg->age-.001*cur_seg->distout+
            .00001986*cur_seg->age*cur_seg->distout),2);
        break;

    case(2):
        if(cur_seg->age<5) wtl=pow((7.1148+0.1948*cur_seg->age),2);
        else wtl=pow((7.1148+0.1948*cur_seg->age-1.3494*
            (cur_seg->age-4.75)),2);
        sna=pow((8.7034-.1611*cur_seg->age-.001*cur_seg->distout+
            .00001986*cur_seg->age*cur_seg->distout),2);
        break;

    case(3):
        if(cur_seg->age<5) wtl=pow((5.9332+.0346*cur_seg->age),2);
        else wtl=pow((5.9332+.0346*cur_seg->age-.8801*

```

```

        (cur_seg->age-4.0)),2);
sna=pow((9.0927-.1611*cur_seg->age-.001*cur_seg->distout+
        .00001986*cur_seg->age*cur_seg->distout),2);
break;

default:
    cout<<"something wrong with crown in shoot_fol\n";
    break;
}
if(wtl<5)
{
    cur_seg->mort=1;
    wtl=0;
    par_scu->liveshoots=par_scu->liveshoots-1;
    idxOver_loop(br,cur_seg,2,par_scu);
    cur_seg->nover=0;
    if(par_scu->liveshoots<0)
        cout<<"negative live shoots! : "<<yr<<"\n";
}

// wtl is mg foliage dry weight per cm shoot length
cur_seg->fol_wt=wtl*cur_seg->length;

// sna is cm^2 foliage area per g foliage dry weight
cur_seg->fol_area=sna*cur_seg->fol_wt/1000;

return ;
}

/*****calc_lines*****/
/* calculate the slope and intercept for each line that defines */
/* the foliage extent for the current shoot. */
/* */
/* The values xll,yll,xlr,ylr,xul,yul,xur,yur give the coordinates*/
/* for the lines that define the foliage extent. xll,yll,xul,yul */
/* give the line parallel to and left of the shoot, the same for */
/* right (xlr, etc). xll,yll,xlr,ylr give the coordinates for */
/* the line perpendicular to the base of the shoot, the same for */
/* the top (xul,yul,xur,yur). */
/*****
void SCU::calc_lines(seg_ptr cur_seg)
{
    if(cur_seg->theta==(PI/2))
    {
        cur_seg->slope[0]=1000;
        cur_seg->interc[0][0]=-10;
        cur_seg->interc[0][1]=-10;

        cur_seg->slope[1]=0;
        cur_seg->interc[1][0]=cur_seg->y1;
        cur_seg->interc[1][1]=cur_seg->y2;
    }

    else

```

```

{
    cur_seg->slope[0]=(cur_seg->y2-cur_seg->y1)/
                    (cur_seg->x2-cur_seg->x1);
    cur_seg->interc[0][0]=cur_seg->y11-cur_seg->slope[0]*
                    cur_seg->x11;
    cur_seg->interc[0][1]=cur_seg->y1r-cur_seg->slope[0]*
                    cur_seg->x1r;

    cur_seg->slope[1]=(cur_seg->yur-cur_seg->yul)/
                    (cur_seg->xur-cur_seg->xul);
    cur_seg->interc[1][0]=cur_seg->y11-cur_seg->slope[1]*
                    cur_seg->x11;
    cur_seg->interc[1][1]=cur_seg->yul-cur_seg->slope[1]*
                    cur_seg->xul;
}
}

/*****comp_loop*****/
/* This function takes two foliated shoots and loops through the */
/* four lines that define their foliage extent to determine      */
/* whether any of the pairs of the four lines intersect,         */
/* indicating the foliage extents of the two shoots overlap.     */
/* It is specifically for shoots that are not perpendicular to the*/
/* x-axis (theta!= 1.57)                                         */
/*                                                                */
/* There are 16 (4x4) pairs of lines, and this loop searches     */
/* through the pairs until it finds an intersection or it has    */
/* checked all pairs.                                            */
/*                                                                */
/*****
void SCU::comp_loop(segs *cur_seg,segs *comp_seg,int flag)
// flag is whether OVERTEST is zero or 1
{
    int over=0;
    int over2=0;
    int i,j,k,l;
    /* i,j for the current seg, k,l for the comp seg. ptr2 is the current
    seg, ptr3 the comparison seg*/
    int check=0;
    int line=0;
    int line2=0;
    double xval,yval,comp_int,comp_slope;

    for(i=0;i<2&&over!=1&&over2!=1;i++)
    {
        for(j=0;j<2&&over!=1&&over2!=1;j++)
        {
            line=line+1;
            for(k=0;k<2&&over!=1&&over2!=1;k++)
            {
                for(l=0;l<2&&over!=1&&over2!=1;l++)
                {
                    check = check +1;
                    line2=line2+1;
                    comp_slope=fabs(cur_seg->slope[i]-comp_seg->slope[k]);

```

```

if(comp_slope>0.0001)
{
    xval=(cur_seg->interc[i][j]-comp_seg->interc[k][l])/
        (comp_seg->slope[k]-cur_seg->slope[i]);
    yval=cur_seg->slope[i]*xval+cur_seg->interc[i][j];
    over=det_overlap(cur_seg,xval,yval,line);
    if(over==1)
    {
        over2=det_overlap(comp_seg,xval,yval,line2);
        if (over2==1)
        {
            if (cur_seg->gen==comp_seg->gen)
            {
                cur_seg->nover=cur_seg->nover+1;
                comp_seg->nover=comp_seg->nover+1;
                if (cur_seg->gen>comp_seg->gen)
                    comp_seg->nover=comp_seg->nover+1;
                if (cur_seg->gen<comp_seg->gen)
                    cur_seg->nover=cur_seg->nover+1;
            }
            else
            {
                if (cur_seg->gen==comp_seg->gen)
                    comp_seg->nover=comp_seg->nover-1;
                else if (cur_seg->gen>comp_seg->gen)
                    comp_seg->nover=comp_seg->nover-1;
                if (comp_seg->nover<0)
                {
                    cout<<"\n*****\nnegative overlap!
                        loop\n*****\n";
                    exit(1);
                }
            }
        }
        else over=0;
    }
}
else
{
    comp_int=fabs(cur_seg->interc[i][j]-
                  comp_seg->interc[k][l]);
    if(comp_int<0.0001)
    {
        if (comp_seg->y1>cur_seg->y1)
        {
            if (comp_seg->y1<cur_seg->y2)
            {
                if(flag==1)
                {
                    if (cur_seg->gen==comp_seg->gen)
                    {
                        cur_seg->nover=cur_seg->nover+1;
                        comp_seg->nover=comp_seg->nover+1;
                    }
                }
            }
        }
    }
}

```

```

    }

    if (cur_seg->gen > comp_seg->gen)
        comp_seg->nover = comp_seg->nover + 1;
    if (cur_seg->gen < comp_seg->gen)
        cur_seg->nover = cur_seg->nover + 1;
    }
    else
    {
        if (cur_seg->gen == comp_seg->gen)
            comp_seg->nover = comp_seg->nover - 1;
        else if (cur_seg->gen > comp_seg->gen)
            comp_seg->nover = comp_seg->nover - 1;
        if (comp_seg->nover < 0)
        {
            cout << "\n*****\nnegative overlap!
                    loop\n*****\n";
            exit(1);
        }
    }
    over = 1;
    over2 = 1;
}
}
else if (comp_seg->y2 > cur_seg->y1)
{
    if (comp_seg->y2 < cur_seg->y2)
    {

        if (flag == 1)
        {
            if (cur_seg->gen == comp_seg->gen)
            {
                cur_seg->nover = cur_seg->nover + 1;
                comp_seg->nover = comp_seg->nover + 1;
            }

            if (cur_seg->gen > comp_seg->gen)
                comp_seg->nover = comp_seg->nover + 1;
            if (cur_seg->gen < comp_seg->gen)
                cur_seg->nover = cur_seg->nover + 1;
        }
        else
        {
            if (cur_seg->gen == comp_seg->gen)
                comp_seg->nover = comp_seg->nover - 1;
            else if (cur_seg->gen > comp_seg->gen)
                comp_seg->nover = comp_seg->nover - 1;
            if (comp_seg->nover < 0)
            {
                cout << "*****\nnegative overlap!
                        loop\n*****\n";
                exit(1);
            }
        }
    }
}

```

```

        over=1;
        over2=1;
    }
}
}
}
}
}
line2=0;
}
}
}

}

/*****det_overlap*****/
/* This function determines whether the intersection of a pair of */
/* lines that define the foliage extent of the current and      */
/* comparison shoot is found within the x and y values that bind */
/* each line (e.g. xll,yll).  If so, then the foliage areas      */
/* overlap and the function returns a 1.  If not, there is no    */
/* overlap and the function returns a zero.                      */
/*                                                                */
/* called by comp_loop(...)                                     */
/* This is the case where both shoots are not perpendicular to the*/
/* x-axis.                                                       */
/*****

int SCU::det_overlap(seg_ptr cur_seg,double xval,double yval, int flag)
{
    int overlap=0;

    switch (flag)
    {

    case(1):
        if(cur_seg->xll>=cur_seg->xul)
        {
            if(xval<cur_seg->xll&& xval>cur_seg->xul)
            {
                if(cur_seg->yll<cur_seg->yul)
                {
                    if(yval>cur_seg->yll&& yval<cur_seg->yul)
                        overlap=1;
                }

                else if(yval<cur_seg->yll&& yval>cur_seg->yul)
                    overlap=1;
            }
        }

    else
    {
        if(xval<cur_seg->xul&& xval>cur_seg->xll)
        {
            if(cur_seg->yll<cur_seg->yul)

```

```

        {
            if(yval>cur_seg->yll&& yval<cur_seg->yul)
                overlap=1;
        }

        else if(yval<cur_seg->yll&& yval>cur_seg->yul)
            overlap=1;
    }
}
break;

case(2):
    if(cur_seg->xlr<=cur_seg->xur)
    {
        if(xval>cur_seg->xlr&& xval<cur_seg->xur)
        {
            if(cur_seg->yur>cur_seg->ylr)
            {
                if(yval<cur_seg->yur&& yval>cur_seg->ylr)
                    overlap=1;
            }
            else if(yval>cur_seg->yur&& yval<cur_seg->ylr)
                overlap=1;
        }
    }
    else
    {
        if(xval<cur_seg->xlr&& xval>cur_seg->xur)
        {
            if(cur_seg->yur>cur_seg->ylr)
            {
                if(yval<cur_seg->yur&& yval>cur_seg->ylr)
                    overlap=1;
            }
            else if(yval>cur_seg->yur&& yval<cur_seg->ylr)
                overlap=1;
        }
    }
}
break;

case(3):
    if(cur_seg->yll<=cur_seg->yur)
    {
        if(yval>cur_seg->yll&& yval<cur_seg->yur)
        {
            if(cur_seg->xll<cur_seg->xur)
            {
                if(xval>cur_seg->xll&& xval<cur_seg->xur)
                    overlap=1;
            }
            else if(xval<cur_seg->xll&& xval>cur_seg->xur)
                overlap=1;
        }
    }
}

```



```

else
{
    if(yval<cur_seg->yll&& yval>cur_seg->y1r)
    {
        if(cur_seg->x1l<cur_seg->x1r)
        {
            if(xval>cur_seg->x1l&& xval<cur_seg->x1r)
                overlap=1;
        }
        else if(xval<cur_seg->x1l&& xval>cur_seg->x1r)
            overlap=1;
    }
}
break;

case(4):
    if(cur_seg->yul<=cur_seg->yur)
    {
        if(yval>cur_seg->yul&& yval<cur_seg->yur)
        {
            if(cur_seg->xul<cur_seg->xur)
            {
                if(xval>cur_seg->xul&& xval<cur_seg->xur)
                    overlap=1;
            }
            else if(xval<cur_seg->xul&& xval>cur_seg->xur)
                overlap=1;
        }
    }
    else
    {
        if(yval<cur_seg->yul&& yval>cur_seg->yur)
        {
            if(cur_seg->xul<cur_seg->xur)
            {
                if(xval>cur_seg->xul&& xval<cur_seg->xur)
                    overlap=1;
            }
            else if(xval<cur_seg->xul&& xval>cur_seg->xur)
                overlap=1;
        }
    }
}
break;

default:
    cout<<"line out of bounds in det_overlap\n";
    break;
}
return overlap;
}

```

```

/*****comp_parallel*****/
/* loops through and determines overlap for two shoots with the */
/* same angle (so the lines are parallel) */
/*****/
void SCU::comp_parallel(segs *cur_seg,segs *comp_seg,int flag)
{
    int over=0,par_line=0;
    int quad=cur_seg->theta_quad;
    //parallel, so they're oriented in the same quadrant

    /* lines 1 and 2 (slope[0], inter[0][0:1]) are between left and right
    (vertical)
    lines 3 and 4 (slope[1], inter[1][0:1]) are between upper and lower
    (horizontal)
    if the intercepts for the same line are the same, then use this to
    compare
    if the intercepts for different lines are the same, then no overlap
    if all of the intercepts are different, then use comp_loop or
    comp_vert*/

    double comp_int;
    comp_int=fabs(cur_seg->interc[1][1]-comp_seg->interc[1][1]);
    if(comp_int<0.0001)
    {
        par_line=3; // top line coincides
    }

    comp_int=fabs(cur_seg->interc[1][0]-comp_seg->interc[1][0]);
    if(comp_int<0.0001)
    {
        par_line=2; // bottom line coincides
    }

    if(cur_seg->theta==PI/2)
        comp_int=fabs(cur_seg->xll-comp_seg->xll);
    else
        comp_int=fabs(cur_seg->interc[0][0]-comp_seg->interc[0][0]);
    if(comp_int<0.0001)
    {
        par_line=1;
        /* right and left lines coincide (bec needle length is the same, if
        right line coincides so does left)*/
        comp_int=fabs(cur_seg->interc[1][0]-comp_seg->interc[1][0]);
        if(comp_int<0.0001) // bottom coincides
            par_line=4;
        comp_int=fabs(cur_seg->interc[1][1]-comp_seg->interc[1][1]);
        if(comp_int<0.0001)
            par_line=4; // top coincides
    }

    if(cur_seg->theta==PI/2)
        comp_int=fabs(cur_seg->xll-comp_seg->xlr);
    else

```

```

    comp_int=fabs(cur_seg->interc[0][0]-comp_seg->interc[0][1]);
if(comp_int<0.0001) //left line coincides with right line
{
    par_line=5;
if(comp_seg->segNum==COMPTTEST&&cur_seg->segNum==SEGTEST)
    cout<<"comp_parallel1: "<<par_line<<"\n";
else if(cur_seg->segNum==COMPTTEST&&comp_seg->segNum==SEGTEST)
    cout<<"comp_parallel1: "<<par_line<<"\n";
}
if(cur_seg->theta==PI/2)
    comp_int=fabs(cur_seg->xlr-comp_seg->xll);
else
    comp_int=fabs(cur_seg->interc[0][1]-comp_seg->interc[0][0]);
if(comp_int<0.0001) //right line coincides with left line
    par_line=5;
comp_int=fabs(cur_seg->interc[1][0]-comp_seg->interc[1][1]);
if(comp_int<0.0001) // top coincides with bottom
    par_line=5;
comp_int=fabs(cur_seg->interc[1][1]-comp_seg->interc[1][0]);
if(comp_int<0.0001) // bottom coincides with top
    par_line=5;

switch(par_line)
{
case(0):
    if(cur_seg->theta!=(PI/2)&&comp_seg->theta!=(PI/2))
        comp_loop(cur_seg,comp_seg,flag);
    else
    {
        comp_vert(cur_seg,comp_seg,flag);
    }
    break;
case(1): // right and left coincides
    switch(quad) // which quadrant are the shoots oriented?
    {
case(1):
        if(comp_seg->y1>cur_seg->y1)
        {
            if(comp_seg->y1<cur_seg->y2)
            {
                over=1;
            }
        }
        else if(cur_seg->y1<comp_seg->y2)
        {
            over=1;
        }
        if(comp_seg->y2>cur_seg->y1)
        {
            if(comp_seg->y2<cur_seg->y2)
            {
                over=1;
            }
        }
        else if(cur_seg->y2<comp_seg->y2)

```

```

{
    over=1;
}
break;
case(2):
    if(comp_seg->y1>cur_seg->y1)
    {
        if(comp_seg->y1<cur_seg->y2)
            over=1;
    }
    else if(cur_seg->y1<comp_seg->y2)
        over=1;
    if(comp_seg->y2<cur_seg->y2)
    {
        if(comp_seg->y2>cur_seg->y1)
            over=1;
    }
    else if(cur_seg->y2>comp_seg->y1)
        over=1;
    break;
case(3):
    if(comp_seg->y2>cur_seg->y2)
    {
        if(comp_seg->y2<cur_seg->y1)
            over=1;
    }
    else if(cur_seg->y2<comp_seg->y1)
        over=1;
    if(comp_seg->y1<cur_seg->y1)
    {
        if(comp_seg->y1>cur_seg->y2)
            over=1;
    }
    else if(cur_seg->y1>comp_seg->y2)
        over=1;
    break;
case(4):
    if(comp_seg->y2>cur_seg->y2)
    {
        if(comp_seg->y2<cur_seg->y1)
            over=1;
    }
    else if(cur_seg->y2<comp_seg->y1)
        over=1;
    if(comp_seg->y1<cur_seg->y1)
    {
        if(comp_seg->y1>cur_seg->y2)
            over=1;
    }
    else if(cur_seg->y1>comp_seg->y2)
        over=1;
    break;
default:
    cout<<"wrong quadrant in comp_parallel 1: "<<
        cur_seg->theta<<"\n";

```

```

        break;

    }
    break;

    case(2):
    /* bottom lines coincide: for each quad there are 3 possibilities: cur
    to the right of comp, cur to the left of comp, cur inside of comp
    (which should correspond to right and left coinciding due to common
    needle length)*/
    switch(quad)
    {
    case(1):
        if(comp_seg->xll>cur_seg->xll)
        {
            if(comp_seg->xll<cur_seg->xlr)
                over=1;
        }
        if(comp_seg->xlr>cur_seg->xll)
        {
            if(comp_seg->xlr<cur_seg->xlr)
                over=1;
        }
        if(cur_seg->xlr<comp_seg->xlr)
        {
            if(cur_seg->xlr>comp_seg->xll)
                over=1;
        }
        break;
    case(2):
        if(comp_seg->xlr>cur_seg->xll)
        {
            if(comp_seg->xlr<cur_seg->xlr)
                over=1;
        }
        if(comp_seg->xll>cur_seg->xll)
        {
            if(comp_seg->xll<cur_seg->xlr)
                over=1;
        }
        if(cur_seg->xlr<comp_seg->xlr)
        {
            if(cur_seg->xlr>comp_seg->xll)
                over=1;
        }
        break;
    case(3):
        if(comp_seg->xll>cur_seg->xlr)
        {
            if(comp_seg->xll<cur_seg->xll)
                over=1;
        }
        if(comp_seg->xlr>cur_seg->xlr)
        {
            if(comp_seg->xlr<cur_seg->xll)

```

```

        over=1;
    }
    if (cur_seg->xll > comp_seg->xlr)
    {
        if (cur_seg->xll < comp_seg->xll)
            over=1;
    }
    break;
case (4):
    if (comp_seg->xlr > cur_seg->xlr)
    {
        if (comp_seg->xlr < cur_seg->xll)
            over=1;
    }
    if (comp_seg->xll > cur_seg->xlr)
    {
        if (comp_seg->xll < cur_seg->xll)
            over=1;
    }
    break;
default:
    cout << "wrong quad in comp_parallel 2: " << cur_seg->theta << "\n";
    break;
}
break;

case (3): // top coincides
    switch (quad)
    {
    case (1):
        if (comp_seg->xul > cur_seg->xul)
        {
            if (comp_seg->xul < cur_seg->xur)
            {
                over=1;
            }
        }
        if (comp_seg->xur > cur_seg->xul)
        {
            if (comp_seg->xur < cur_seg->xur)
            {
                over=1;
            }
        }
        break;
    case (2):
        if (comp_seg->xur > cur_seg->xul)
        {
            if (comp_seg->xur < cur_seg->xur)
                over=1;
        }
        if (comp_seg->xul > cur_seg->xul)
        {
            if (comp_seg->xul < cur_seg->xur)

```

```

        over=1;
    }
    break;
case(3):
    if(comp_seg->xul>cur_seg->xur)
    {
        if(comp_seg->xul<cur_seg->xul)
            over=1;
    }
    if(comp_seg->xur>cur_seg->xur)
    {
        if(comp_seg->xur<cur_seg->xul)
            over=1;
    }
    break;
case(4):
    if(comp_seg->xur>cur_seg->xur)
    {
        if(comp_seg->xur<cur_seg->xul)
            over=1;
    }
    if(comp_seg->xul>cur_seg->xur)
    {
        if(comp_seg->xul<cur_seg->xul)
            over=1;
    }
    break;
default:
    cout<<"wrong quad in comp_parallel 3: "<<cur_seg->theta<<"\n";
    break;
}
break;
case(4):
/* bottom and right and left coincide and/or top and right and left
coincide*/
    over=1;
    break;

case(5): // bottom coincides with top
    over=0;
    break;

default:
    cout<<"wrong flag in comp_parallel\n";
    break;
}

if(over==1)
{
    if(flag==1)
    {
        if(cur_seg->gen==comp_seg->gen)
        {
            cur_seg->nover=cur_seg->nover+1;
            comp_seg->nover=comp_seg->nover+1;

```

```

    }

    if(cur_seg->gen>comp_seg->gen)
        comp_seg->nover=comp_seg->nover+1;

    if(cur_seg->gen<comp_seg->gen)
        cur_seg->nover=cur_seg->nover+1;
    }
    else
    {
//      comp_seg->nover=comp_seg->nover-1;
      if(cur_seg->gen==comp_seg->gen)
          comp_seg->nover=comp_seg->nover-1;
      else if(cur_seg->gen>comp_seg->gen)
          comp_seg->nover=comp_seg->nover-1;
      if(comp_seg->nover<0)
      {
          cout<<"\n*****\nnegative overlap! parallel\n*****\n";
          exit(1);
      }
    }
}

}

/*****comp_vert*****/
/* This function takes two foliated shoots and loops through the */
/* four lines that define their foliage extent to determine */
/* whether any of the pairs of the four lines intersect, */
/* indicating the foliage extents of the two shoots overlap. */
/* It is called when one or both of the shoots is/are */
/* perpendicular to the x-axis (theta== 1.57) */
/* */
/* There are 16 (4x4) pairs of lines, and this loop searches */
/* through the pairs until it finds an intersection or it has */
/* checked all pairs. */
/*****/

void SCU::comp_vert(segs *cur_seg,segs *comp_seg,int flag)
{
    int over=0;
    int over2=0;
    double yval,xval;
    int line=0;

    int i,j;

    if(cur_seg->theta==(PI/2))
    {
        if(comp_seg->theta==(PI/2))
        {
            if(comp_seg->x11<=cur_seg->x1r&&comp_seg->x11>=cur_seg->x1l)
            {
                if(comp_seg->y1>=cur_seg->y1&&comp_seg->y1<cur_seg->y2)
                    over=1;
            }
        }
    }
}

```



```

else if(cur_seg->y1>=comp_seg->y1&&
        cur_seg->y1<comp_seg->y2)
    over=1;
}
else if(comp_seg->xlr>=cur_seg->xll&&
        comp_seg->xlr<=cur_seg->xlr)
{
    if(comp_seg->y1>=cur_seg->y1&&comp_seg->y1<cur_seg->y2)
        over=1;
    else if(cur_seg->y1>=comp_seg->y1&&
            cur_seg->y1<comp_seg->y2)
        over=1;
}
}
else
{
    line=0;
    for(i=0;i<2&&over!=1;i++)
    {
        for(j=0;j<2&&over!=1;j++)
        {
            line=line+1;
/* where on the y-axis does the cur_seg cross the left line of the
comp_seg?*/
            yval=comp_seg->slope[i]*cur_seg->xll+
                    comp_seg->interc[i][j];
// determine whether that crossing intersects both foliage rectangles
            over=det_vert_over(cur_seg,comp_seg,yval,line);

            if(over!=1)
            {
/* if comp_seg doesn't cross xll, where on the y-axis does the cur_seg
cross the right (xlr) line of the comp_seg?*/
                yval=comp_seg->slope[i]*cur_seg->xlr+
                        comp_seg->interc[i][j];
                over=det_vert_over(cur_seg,comp_seg,yval,line);
            }

            if(over!=1)
            {
/* where on the x-axis does the comp_seg cross the bottom line of the
cur_seg?*/
                xval=(cur_seg->y1-comp_seg->interc[i][j])/
                        comp_seg->slope[i];
                over=det_horiz_over(cur_seg,comp_seg,xval,line);
                if(over!=1)
                {
                    xval=(cur_seg->y2-comp_seg->interc[i][j])/
                            comp_seg->slope[i];
                    over=det_horiz_over(cur_seg,comp_seg,xval,line);
                }
            }
        }
    }
}
}

```

```

    }
}

else
{
    line=0;
    for(i=0;i<2&&over!=1;i++)
    {
        for(j=0;j<2&&over!=1;j++)
        {
            line=line+1;
            yval=cur_seg->slope[i]*comp_seg->xll+cur_seg->interc[i][j];
            over=det_vert_over(comp_seg,cur_seg,yval,line);

            if(over!=1)
            {
                yval=cur_seg->slope[i]*comp_seg->xlr+
                    cur_seg->interc[i][j];
                over=det_vert_over(comp_seg,cur_seg,yval,line);
            }
            if(over!=1)
            {
                xval=(comp_seg->y1-cur_seg->interc[i][j])/
                    cur_seg->slope[i];
                over=det_horiz_over(comp_seg,cur_seg,xval,line);
                if(over!=1)
                {
                    xval=(comp_seg->y2-cur_seg->interc[i][j])/
                        cur_seg->slope[i];
                    over=det_horiz_over(comp_seg,cur_seg,xval,line);
                }
            }
        }
    }
}

if(over==1)
{
    if(flag==1)
    {
        if(cur_seg->gen==comp_seg->gen)
        {
            cur_seg->nover=cur_seg->nover+1;
            comp_seg->nover=comp_seg->nover+1;
            if(comp_seg->segNum==SEGTEST)
                cout<<cur_seg->segNum<<"\t"<<cur_seg->gen<<"\t";
            if(cur_seg->segNum==SEGTEST)
                cout<<comp_seg->segNum<<"\t"<<comp_seg->gen<<"\t";
        }

        if(cur_seg->gen>comp_seg->gen)
        {
            comp_seg->nover=comp_seg->nover+1;
            if(comp_seg->segNum==SEGTEST)

```

```

        cout<<cur_seg->segNum<<"\t"<<cur_seg->gen<<"\t";
    }

    if(cur_seg->gen<comp_seg->gen)
    {
        cur_seg->nover=cur_seg->nover+1;
        if(cur_seg->segNum==SEGTEST)
            cout<<comp_seg->segNum<<"\t"<<comp_seg->gen<<"\t";
    }

}
else
{
    if(cur_seg->gen==comp_seg->gen)
    {
        comp_seg->nover=comp_seg->nover-1;
        if(comp_seg->segNum==SEGTEST)
            cout<<"-"<<cur_seg->segNum<<"\t"<<cur_seg->gen<<"\t";
    }
    else if(cur_seg->gen>comp_seg->gen)
    {
        comp_seg->nover=comp_seg->nover-1;
        if(comp_seg->segNum==SEGTEST)
            cout<<"-"<<cur_seg->segNum<<"\t"<<cur_seg->gen<<"\t";
    }
    if(comp_seg->nover<0)
    {
        cout<<"\n*****\nnegative overlap! vert\n*****\n";
        exit(1);
    }
}
}

}

/*****det_vert_over*****/
/* This function determines whether the intersection of a pair of */
/* lines that define the foliage extent of the current and      */
/* comparison shoot is found within the x and y values that bind */
/* each line (e.g. xll,yll).  If so, then the foliage areas      */
/* overlap and the function returns a 1.  If not, there is no     */
/* overlap and the function returns a zero.                       */
/*                                                                */
/* called by comp_vert(...)                                       */
/* This is the case where one of the shoots is perpendicular to  */
/* the x-axis, and this looks for intersections of the lines     */
/* parallel to the perpendicular shoot.                           */
/*****
int SCU::det_vert_over(seg_ptr cur_seg,seg_ptr comp_seg,double yval,
                      int flag)
{
    int over=0;
    if(yval>=cur_seg->y1)
    {
        if(yval<=cur_seg->y2)
        {

```

```

switch(flag)
{
case (1):
    if(yval<comp_seg->yul&& yval>comp_seg->yll)
        over=1;
    break;
case (2):
    if(yval<comp_seg->yur&& yval>comp_seg->ylr)
        over=1;
    break;
case (3):
    if(comp_seg->yur>=comp_seg->yll)
    {
        if(yval>comp_seg->yll&& yval<comp_seg->yur)
            over=1;
    }
    else
    {
        if(yval<comp_seg->yll&& yval>comp_seg->yur)
            over=1;
    }
    break;
case (4):
    if(comp_seg->yur>=comp_seg->yul)
    {
        if(yval<comp_seg->yur&& yval>comp_seg->yul)
            over=1;
    }
    else
    {
        if(yval>comp_seg->yur&& yval<comp_seg->yul)
            over=1;
    }
    break;
default:
    cout<<"line out of bounds in det_vert_over\n";
    break;
}
}
return over;
}

```

```

/*****det_horiz_over*****/
/* This function determines whether the intersection of a pair */
/* of lines that define the foliage extent of the current and */
/* comparison shoot is found within the x and y values that bind */
/* each line (e.g. xll,yll). If so, then the foliage areas */
/* overlap and the function returns a 1. If not, there is no */
/* overlap and the function returns a zero. */
/* */
/* called by comp_vert(...) */
/* This is the case where one of the shoots is perpendicular to */
/* the x-axis, and this looks for intersections of the lines */
/* perpendicular to the perpendicular shoot. */
/*****

```

```

int SCU::det_horiz_over(seg_ptr cur_seg,seg_ptr comp_seg,
                        double xval,int flag)
{
    int over=0;
    if(xval<=cur_seg->xlr)
    {
        if(xval>=cur_seg->xll)
        {
            switch(flag)
            {
                case (1):
                    if(comp_seg->xul<=comp_seg->xll)
                    {
                        if(xval<comp_seg->xll&& xval>comp_seg->xul)
                            over=1;
                    }
                    else
                    {
                        if(xval<comp_seg->xll&& xval>comp_seg->xul)
                            over=1;
                    }
                    break;
                case (2):
                    if(comp_seg->xur<=comp_seg->xlr)
                    {
                        if(xval>comp_seg->xur&& xval<comp_seg->xlr)
                            over=1;
                    }
                    else
                    {
                        if(xval<comp_seg->xur&& xval>comp_seg->xlr)
                            over=1;
                    }
                    break;
                case (3):
                    if(comp_seg->xlr>=comp_seg->xll)
                    {
                        if(xval>comp_seg->xll&& xval<comp_seg->xlr)
                            over=1;
                    }
                }
            }
        }
    }
}

```

```

    }
    else
    {
        if(xval<comp_seg->xll&& xval>comp_seg->xlr)
            over=1;
    }
    break;
case (4):
    if(comp_seg->xur>=comp_seg->xul)
    {
        if(xval<comp_seg->xur&& xval>comp_seg->xul)
            over=1;
    }
    else
    {
        if(xval>comp_seg->xur&& xval<comp_seg->xul)
            over=1;
    }
    break;
default:
    cout<<"line out of bounds in det_horiz_over\n";
    break;
}
}
}
return over;
}

/*****shoot_over2*****/
/* Loop through the remaining shoots in the current branch grid, */
/* called from shoot_over, to determine whether the comp_seg of */
/* this loop overlaps with the cur_seg of the shoot_over loop. */
/******/
void SCU::shoot_over2(SCU *par_scu, segs *cur_seg, segs *comp_seg,
                     long *seed, int flag, int flag2, branch *br)
{
    while(comp_seg!=NULL)
    {
        if(cur_seg!=comp_seg)
        {
            /* code to compare the cur_seg to the comparison seg to tally overlap
            different code for shoots perpendicular to the x-axis and those not*/
            if(cur_seg->theta!=(PI/2)&& comp_seg->theta!=(PI/2))
                comp_loop(cur_seg, comp_seg, 1);
            else
                comp_vert(cur_seg, comp_seg, 1);
        }

        comp_seg=comp_seg->br_next;
    }

    return ;
}

```

```

/*****shoot_rekurs_over*****/
/* recurs through every shoot to calculate its foliage overlap, */
/* using the branch array. */
/*****/
void SCU::shoot_rekurs_over(branch *br,segs *cur_seg,int flag,
                           long *seed, SCU *par_scu,int pop, int run)
{
    while(cur_seg!=NULL)
    {
        par_scu=cur_seg->par_scu;
        if(cur_seg->active==1)
            make_buds(cur_seg,0,seed,par_scu,br->damage_prob);

        if(cur_seg->mort==0)
            par_scu->nover=par_scu->nover+cur_seg->nover;
        cur_seg=cur_seg->br_next;

    } // end while
}

/*****idxOver_loop*****/
/* loops through the neighboring indices in the array that can */
/* possibly overlap the current shoot */
/*****/
void SCU::idxOver_loop(branch *br,segs *cur_seg,int flag, SCU *par_scu)
{
    int i,j;
    int idx,idy;

    int k;//,1;
    double mult[4][2];

    mult[0][0]=1;
    mult[0][1]=1;

    mult[1][0]=-1;
    mult[1][1]=1;

    mult[2][0]=1;
    mult[2][1]=-1;

    mult[3][0]=-1;
    mult[3][1]=-1;

    int flagx1=0,flagy1=0,flagx2=0,flagy2=0;
    idx=cur_seg->xmid;
    idy=cur_seg->ymid;
    idy:"<<idy<<"\n";
    array_over(cur_seg,br->br_array[idx][idy],flag,1,br); // change this
    such that flag2 indicates whether nover is to increase or to decrease
    if(cur_seg->xmid<=ARR_SEARCH||cur_seg->ymid<=ARR_SEARCH||

```

```

        cur_seg->xmid>=(XSIZE-ARR_SEARCH)||
        cur_seg->ymid>=(YSIZE-ARR_SEARCH))
/* if any of these are true, then the shoot is at the border of the
array. need to cut it off at the border*/
    shoot_border(br,cur_seg,flag);
    else
/* we are not outside of the border of the array, look at array indices
+/- 6 of the current array*/
    {
        for(i=1;i<=ARR_SEARCH;i++)
        {
            flagx1=0;
            flagx2=0;
            idx=cur_seg->xmid;
            idy=cur_seg->ymid+i;
            if(idx<0||idy<0||idx>XSIZE-1||idy>YSIZE-1)
            {
                cout<<"Error, array out of bounds shoot_border\n";
                exit(1);
            }

            if(br->br_array[idx][idy]!=NULL)
                array_over(cur_seg,br->br_array[idx][idy],flag,1,br);

            idy=cur_seg->ymid-i;
            if(idx<0||idy<0||idx>XSIZE-1||idy>YSIZE-1)
            {
                cout<<"Error, array out of bounds shoot_border\n";
                exit(1);
            }

            if(br->br_array[idx][idy]!=NULL)
                array_over(cur_seg,br->br_array[idx][idy],flag,1,br);

            idy=cur_seg->ymid;
            idx=cur_seg->xmid+i;
            if(idx<0||idy<0||idx>XSIZE-1||idy>YSIZE-1)
            {
                cout<<"Error, array out of bounds shoot_border\n";
                exit(1);
            }

            if(br->br_array[idx][idy]!=NULL)
                array_over(cur_seg,br->br_array[idx][idy],flag,1,br);

            idx=cur_seg->xmid-i;
            if(idx<0||idy<0||idx>XSIZE-1||idy>YSIZE-1)
            {
                cout<<"Error, array out of bounds shoot_border\n";
                exit(1);
            }
        }
    }

```



```

        if(br->br_array[idx][idy]!=NULL)
            array_over(cur_seg,br->br_array[idx][idy],flag,1,br);

        for(j=1;j<=ARR_SEARCH;j++)
        {
            flagy1=0;
            flagy2=0;
            for(k=0;k<4;k++)
            {
                idx=cur_seg->xmid+i*mult[k][0];
                idy=cur_seg->ymid+j*mult[k][1];
                if(idx<0||idy<0||idx>XSIZE-1||idy>YSIZE-1)
                {
                    cout<<"Error, array out of bounds shoot_border\n";
                    exit(1);
                }

                if(br->br_array[idx][idy]!=NULL)
                    array_over(cur_seg,br->br_array[idx][idy],flag,1,br);
            }// end k

        }//end j
    }//end i
}

/*****array_over*****/
/* Navigates the linked list for the current array indices */
/******/
void SCU::array_over(segs *cur_seg,segs *comp_seg,int flag,
                    int flag2,branch *br)
{
    double comp_theta;//,comp_pi;
    while(comp_seg!=NULL)
    {
        if(cur_seg!=comp_seg&&comp_seg->mort!=1)
            comp_parallel(cur_seg,comp_seg,flag);
        /* code to compare the cur_seg to the comparison seg to tally overlap
        different code for shoots perpendicular to the x-axis and those not
        scenarios with the same intercept for any of the lines (flush) vs.
        same intercept for two of the same lines (almost complete overlap) vs.
        same intercept for all of the lines (complete overlap)*/
        else
        {
            if(cur_seg->theta!=(PI/2)&&comp_seg->theta!=(PI/2))
                comp_loop(cur_seg,comp_seg,flag);
            else
                comp_vert(cur_seg,comp_seg,flag);
        }

        comp_seg=comp_seg->arr_next;
    }
}

```

```

    return ;
}

/*****shoot_border*****/
/* Navigate the array for shoots that are at the border of */
/* the array */
/*****
void SCU::shoot_border(branch *br, segs *cur_seg, int flag)
{
    int flag1=0;
    int flag2=0;
    // Figure out which border we're at
    /* flag1 indicates which x-border, flag2 indicates which y-border. =0
    for each if not at that border*/
    if(cur_seg->xmid<=ARR_SEARCH)
        flag1=1;
    else if(cur_seg->xmid>=(XSIZE-1)-ARR_SEARCH))
        flag1=2;

    if(cur_seg->ymid<=ARR_SEARCH)
        flag2=1;
    else if(cur_seg->ymid>=(YSIZE-1)-ARR_SEARCH))
        flag2=2;

    switch(flag1)
    {
    case(0):
        shoot_X0border(br,cur_seg,flag,flag2);
        break;

    case(1):
        shoot_X1border(br,cur_seg,flag,flag2);
        break;

    case(2):
        shoot_X2border(br,cur_seg,flag,flag2);
        break;

    default:
        cout<<"Error, wrong flag1 in shoot_border\n";
        break;
    }
}

/*****shoot_X0border*****/
/* Navigate the array for shoots that are not at either x border */
/*****
void SCU::shoot_X0border(branch *br, segs *cur_seg, int flag,
                        int flag2)
{
    int idx,idy;
    int i,j,k;

```

```

double mult[4][2];

int bord_dist;

mult[0][0]=1; //add to idx
mult[0][1]=1; //add to idy

mult[1][0]=-1; // subtract from idx
mult[1][1]=1; // add to idy

mult[2][0]=-1; // subtract from both idx and idy
mult[2][1]=-1;

mult[3][0]=1; // add to idx
mult[3][1]=-1; // subtract from idy

idx=cur_seg->xmid;
idy=cur_seg->ymid;

switch(flag2)
{
case(0):
    cout<<"Error, shoot_border, shouldn't be here\n";
    exit(1);
    break;

case(1):
    for(i=1;i<=ARR_SEARCH;i++)
    {
        idx=cur_seg->xmid;
        idy=cur_seg->ymid+i;
        if(idx<0||idy<0||idx>XSIZE-1||idy>YSIZE-1)
        {
            cout<<"Error, array out of bounds shoot_border\n";
            exit(1);
        }

        if(br->br_array[idx][idy]!=NULL)
            array_over(cur_seg,br->br_array[idx][idy],flag,1,br);

        if(i<=cur_seg->ymid)
        // can only subtract up to ymid in the y-direction
        {
            idy=cur_seg->ymid-i;
            if(idx<0||idy<0||idx>XSIZE-1||idy>YSIZE-1)
            {
                cout<<"Error, array out of bounds shoot_border\n";
                exit(1);
            }

            if(br->br_array[idx][idy]!=NULL)
                array_over(cur_seg,br->br_array[idx][idy],flag,1,br);
        }
    }
}

```

```

idx=cur_seg->xmid+i;
idy=cur_seg->ymid;
if(idx<0||idy<0||idx>XSIZE-1||idy>YSIZE-1)
{
    cout<<"Error, array out of bounds shoot_border\n";
    exit(1);
}

if(br->br_array[idx][idy]!=NULL)
    array_over(cur_seg,br->br_array[idx][idy],flag,1,br);

idx=cur_seg->xmid-i;
if(idx<0||idy<0||idx>XSIZE-1||idy>YSIZE-1)
{
    cout<<"Error, array out of bounds shoot_border\n";
    exit(1);
}

if(br->br_array[idx][idy]!=NULL)
    array_over(cur_seg,br->br_array[idx][idy],flag,1,br);

for(j=1;j<=ARR_SEARCH;j++)
{
    for(k=0;k<2;k++)
// can add to all 6 spaces in the y direction
    {
        idx=cur_seg->xmid+i*mult[k][0];
        idy=cur_seg->ymid+j*mult[k][1];
        if(idx<0||idy<0||idx>XSIZE-1||idy>YSIZE-1)
        {
            cout<<"Error, array out of bounds shoot_border
                                     j loop\n";
            exit(1);
        }

        if(br->br_array[idx][idy]!=NULL)
            array_over(cur_seg,br->br_array[idx][idy],flag,1,br);
    }// end k

} //end j1
for(j=1;j<=cur_seg->ymid;j++)
{
    for(k=2;k<4;k++)
// can only subtract up to ymid in the y-direction
    {
        idx=cur_seg->xmid+i*mult[k][0];
        idy=cur_seg->ymid+j*mult[k][1];
        if(idx<0||idy<0||idx>XSIZE-1||idy>YSIZE-1)
        {
            cout<<"Error, array out of bounds shoot_border
                                     j loop\n";
            exit(1);
        }
    }
}

```

```

        if(br->br_array[idx][idy]!=NULL)
            array_over(cur_seg,br->br_array[idx][idy],flag,1,br);
    }// end k

    }//end j1
} //end i
break;

case(2):
    bord_dist=(YSIZE-1)-cur_seg->ymid;
    for(i=1;i<=ARR_SEARCH;i++)
    {
        idx=cur_seg->xmid;
        if(i<=bord_dist)
        {
            idy=cur_seg->ymid+i;
            if(idx<0||idy<0||idx>XSIZE-1||idy>YSIZE-1)
            {
                cout<<"Error, array out of bounds shoot_border\n";
                exit(1);
            }

            if(br->br_array[idx][idy]!=NULL)
                array_over(cur_seg,br->br_array[idx][idy],flag,1,br);
        }

        idy=cur_seg->ymid-i;
        if(idx<0||idy<0||idx>XSIZE-1||idy>YSIZE-1)
        {
            cout<<"Error, array out of bounds shoot_border\n";
            exit(1);
        }

        if(br->br_array[idx][idy]!=NULL)
            array_over(cur_seg,br->br_array[idx][idy],flag,1,br);

        idx=cur_seg->xmid+i;
        idy=cur_seg->ymid;
        if(idx<0||idy<0||idx>XSIZE-1||idy>YSIZE-1)
        {
            cout<<"Error, array out of bounds shoot_border\n";
            exit(1);
        }

        if(br->br_array[idx][idy]!=NULL)
            array_over(cur_seg,br->br_array[idx][idy],flag,1,br);

        idx=cur_seg->xmid-i;
        idy=cur_seg->ymid;
        if(idx<0||idy<0||idx>XSIZE-1||idy>YSIZE-1)
        {
            cout<<"Error, array out of bounds shoot_border\n";
            exit(1);
        }
    }
}

```

```

if(br->br_array[idx][idy]!=NULL)
    array_over(cur_seg,br->br_array[idx][idy],flag,1,br);

for(j=1;j<=ARR_SEARCH;j++)
{
    for(k=2;k<4;k++) // can subtract all 6 in the y-direction
    {
        idx=cur_seg->xmid+i*mult[k][0];
        idy=cur_seg->ymid+j*mult[k][1];
        if(idx<0||idy<0||idx>XSIZE-1||idy>YSIZE-1)
        {
            cout<<"Error, array out of bounds shoot_border
                                     j loop\n";
            exit(1);
        }

        if(br->br_array[idx][idy]!=NULL)
            array_over(cur_seg,br->br_array[idx][idy],flag,1,br);
    } // end k

} // end j1
for(j=1;j<=bord_dist;j++)
{
    for(k=0;k<2;k++)
// can only add up to the distance to the border in the y-direction
    {
        idx=cur_seg->xmid+i*mult[k][0];
        idy=cur_seg->ymid+j*mult[k][1];
        if(idx<0||idy<0||idx>XSIZE-1||idy>YSIZE-1)
        {
            cout<<"Error, array out of bounds shoot_border
                                     j loop\n";
            exit(1);
        }

        if(br->br_array[idx][idy]!=NULL)
            array_over(cur_seg,br->br_array[idx][idy],flag,1,br);
    } // end k

} // end j2
} // end i
break;

default:
    cout<<"wrong flag in shoot_border second switch\n";
    break;

} // end switch2
} // end switch

```

```

/*****shoot_Xlborder*****/
/* Navigate the array for shoots that are at the lower x */
/* border of the array */
/*****

void SCU::shoot_Xlborder(branch *br, segs *cur_seg, int flag,
                        int flag2)
{
    int idx,idy;
    int i,j;
    double mult[4][2];

    int bord_dist;

    mult[0][0]=1; //add to idx
    mult[0][1]=1; //add to idy

    mult[1][0]=-1;// subtract from idx
    mult[1][1]=1;// add to idy

    mult[2][0]=-1;// subtract from both idx and idy
    mult[2][1]=-1;

    mult[3][0]=1;// add to idx
    mult[3][1]=-1;// subtract from idy

    idx=cur_seg->xmid;
    idy=cur_seg->ymid;

    switch(flag2)
    {
    case(0):
/* uninhibited in the y-direction, but on the left border of the x
(inhbits subtraction from x)*/
        for(i=1;i<=ARR_SEARCH;i++)
        {
            idx=cur_seg->xmid;
            idy=cur_seg->ymid+i;
            if(idx<0||idy<0||idx>XSIZE-1||idy>YSIZE-1)
            {
                cout<<"Error, array out of bounds shoot_border\n";
                exit(1);
            }

            if(br->br_array[idx][idy]!=NULL)
                array_over(cur_seg,br->br_array[idx][idy],flag,1,br);

            idy=cur_seg->ymid-i;
            if(idx<0||idy<0||idx>XSIZE-1||idy>YSIZE-1)
            {
                cout<<"Error, array out of bounds shoot_border\n";
                exit(1);
            }
        }
    }
}

```

```

if (br->br_array[idx][idy] != NULL)
    array_over(cur_seg, br->br_array[idx][idy], flag, 1, br);

idy = cur_seg->ymid;
idx = cur_seg->xmid + i;
if (idx < 0 || idy < 0 || idx > XSIZE-1 || idy > YSIZE-1)
{
    cout << "Error, array out of bounds shoot_border\n";
    exit(1);
}

if (br->br_array[idx][idy] != NULL)
    array_over(cur_seg, br->br_array[idx][idy], flag, 1, br);

if (i <= cur_seg->xmid)
{
    idx = cur_seg->xmid - i;
    if (idx < 0 || idy < 0 || idx > XSIZE-1 || idy > YSIZE-1)
    {
        cout << "Error, array out of bounds shoot_border\n";
        exit(1);
    }

    if (br->br_array[idx][idy] != NULL)
        array_over(cur_seg, br->br_array[idx][idy], flag, 1, br);
}

for (j = 1; j <= ARR_SEARCH; j++)
{
    // can add to both directions
    idx = cur_seg->xmid + i * mult[0][0];
    idy = cur_seg->ymid + j * mult[0][1];
    if (idx < 0 || idy < 0 || idx > XSIZE-1 || idy > YSIZE-1)
    {
        cout << "Error, array out of bounds shoot_border j
                                                    loop 1\n";
        exit(1);
    }

    if (br->br_array[idx][idy] != NULL)
        array_over(cur_seg, br->br_array[idx][idy], flag, 1, br);
    idx = cur_seg->xmid + i * mult[3][0];
    idy = cur_seg->ymid + j * mult[3][1];
    if (idx < 0 || idy < 0 || idx > XSIZE-1 || idy > YSIZE-1)
    {
        cout << "Error, array out of bounds shoot_border j
                                                    loop 2\n";
        exit(1);
    }

    if (br->br_array[idx][idy] != NULL)
        array_over(cur_seg, br->br_array[idx][idy], flag, 1, br);
}

```



```

if(i<=cur_seg->xmid)
{
    idx=cur_seg->xmid+i*mult[1][0];
    idy=cur_seg->yamid+j*mult[1][1];
    if(idx<0||idy<0||idx>XSIZE-1||idy>YSIZE-1)
    {
        cout<<"Error, array out of bounds shoot_border j
                                loop 3\n";
        exit(1);
    }

    if(br->br_array[idx][idy]!=NULL)
        array_over(cur_seg,br->br_array[idx][idy],flag,1,br);

    idx=cur_seg->xmid+i*mult[2][0];
    idy=cur_seg->yamid+j*mult[2][1];
    if(idx<0||idy<0||idx>XSIZE-1||idy>YSIZE-1)
    {
        cout<<"Error, array out of bounds shoot_border j
                                loop 4\n";
        exit(1);
    }

    if(br->br_array[idx][idy]!=NULL)
        array_over(cur_seg,br->br_array[idx][idy],flag,1,br);
}
} //end j1
} //end i
break;

case(1):
for(i=1;i<=ARR_SEARCH;i++)
{
    idx=cur_seg->xmid;
    idy=cur_seg->yamid+i;
    if(idx<0||idy<0||idx>XSIZE-1||idy>YSIZE-1)
    {
        cout<<"Error, array out of bounds shoot_border\n";
        exit(1);
    }

    if(br->br_array[idx][idy]!=NULL)
        array_over(cur_seg,br->br_array[idx][idy],flag,1,br);

    if(i<=cur_seg->yamid)
// can only subtract up to ymid in the y-direction
    {
        idy=cur_seg->yamid-i;
        if(idx<0||idy<0||idx>XSIZE-1||idy>YSIZE-1)
        {
            cout<<"Error, array out of bounds shoot_border\n";
            exit(1);
        }

        if(br->br_array[idx][idy]!=NULL)

```

```

        array_over(cur_seg,br->br_array[idx][idy],flag,1,br);
    }

    idx=cur_seg->xmid+i;
    idy=cur_seg->ymid;
    if(idy<0||idy>YSIZE-1||idx>XSIZE-1||idx<0)
    {
        cout<<"Error, array out of bounds shoot_border\n";
        exit(1);
    }

    if(br->br_array[idx][idy]!=NULL)
        array_over(cur_seg,br->br_array[idx][idy],flag,1,br);

    if(i<=cur_seg->xmid)
// can only subtract up to xmid in the x-direction
    {
        idx=cur_seg->xmid-i;
        if(idy<0||idy>YSIZE-1||idx>XSIZE-1||idx<0)
        {
            cout<<"Error, array out of bounds shoot_border\n";
            exit(1);
        }

        if(br->br_array[idx][idy]!=NULL)
            array_over(cur_seg,br->br_array[idx][idy],flag,1,br);
    }

    for(j=1;j<=ARR_SEARCH;j++)
    {
// can add to all 6 spaces in both the x and y direction
        idx=cur_seg->xmid+i*mult[0][0];
        idy=cur_seg->ymid+j*mult[0][1];
        if(idy<0||idy>YSIZE-1||idx>XSIZE-1||idx<0)
        {
            cout<<"Error, array out of bounds shoot_border j
                                loop\n";
            exit(1);
        }
        if(br->br_array[idx][idy]!=NULL)
            array_over(cur_seg,br->br_array[idx][idy],flag,1,br);

        if(i<=cur_seg->xmid)
// can only subtract up to xmid in the x-direction
        {
            idx=cur_seg->xmid+i*mult[1][0];
            idy=cur_seg->ymid+j*mult[1][1];
            if(idy<0||idy>YSIZE-1||idx>XSIZE-1||idx<0)
            {
                cout<<"Error, array out of bounds shoot_border j
                                loop\n";
                exit(1);
            }
            if(br->br_array[idx][idy]!=NULL)

```

```

        array_over(cur_seg, br-
>br_array[idx][idy], flag, 1, br);
    }
    } //end j1

    for(j=1; j<=cur_seg->ymid; j++)
// add all in x-direction, subtract only up to ymid in y-direction
    {
        idx=cur_seg->xmid+i*mult[3][0];
        idy=cur_seg->ymid+j*mult[3][1];
        if(idy<0||idy>YSIZE-1)
        {
            cout<<"Error, array out of bounds shoot_border j
                                loop\n";
            exit(1);
        }
        if(br->br_array[idx][idy]!=NULL)
            array_over(cur_seg, br->br_array[idx][idy], flag, 1, br);

        if(i<=cur_seg->xmid)
// can only subtract up to xmid and ymid
        {
            idx=cur_seg->xmid+i*mult[2][0];
            idy=cur_seg->ymid+j*mult[2][1];
            if(idy<0||idy>YSIZE-1)
            {
                cout<<"Error, array out of bounds shoot_border j
                                loop\n";
                exit(1);
            }
            if(br->br_array[idx][idy]!=NULL)
                array_over(cur_seg, br->br_array[idx][idy], flag, 1, br);
        }
    } //end j2
} //end i
break;

case(2):
    bord_dist=(YSIZE-1)-cur_seg->ymid;
    for(i=1; i<=ARR_SEARCH; i++)
    {
        idx=cur_seg->xmid;
        if(i<=bord_dist) // only add up to bord_dist
        {
            idy=cur_seg->ymid+i;
            if(idy<0||idy>YSIZE-1)
            {
                cout<<"Error, array out of bounds shoot_border\n";
                exit(1);
            }

            if(br->br_array[idx][idy]!=NULL)
                array_over(cur_seg, br->br_array[idx][idy], flag, 1, br);
        }
    }
}

```

```

idy=cur_seg->yamid-i;
if (idx<0||idy<0||idx>XSIZE-1||idy>YSIZE-1)
{
    cout<<"Error, array out of bounds shoot_border\n";
    exit(1);
}

if (br->br_array[idx][idy]!=NULL)
    array_over(cur_seg,br->br_array[idx][idy],flag,1,br);

idx=cur_seg->xamid+i;
idy=cur_seg->yamid;
if (idx<0||idy<0||idx>XSIZE-1||idy>YSIZE-1)
{
    cout<<"Error, array out of bounds shoot_border\n";
    exit(1);
}

if (br->br_array[idx][idy]!=NULL)
    array_over(cur_seg,br->br_array[idx][idy],flag,1,br);

if (i<=cur_seg->xamid)
{
    idx=cur_seg->xamid-i;
    if (idx<0||idy<0||idx>XSIZE-1||idy>YSIZE-1)
    {
        cout<<"Error, array out of bounds shoot_border\n";
        exit(1);
    }

    if (br->br_array[idx][idy]!=NULL)
        array_over(cur_seg,br->br_array[idx][idy],flag,1,br);
}

for (j=1;j<=ARR_SEARCH;j++)
{
    /* can subtract all 6 in the y-direction, but only up to xamid in the x-
    direction*/
    idx=cur_seg->xamid+i*mult[3][0];
    idy=cur_seg->yamid+j*mult[3][1];
    if (idx<0||idy<0||idx>XSIZE-1||idy>YSIZE-1)
    {
        cout<<"Error, array out of bounds shoot_border j
        loop\n";
        exit(1);
    }

    if (br->br_array[idx][idy]!=NULL)
        array_over(cur_seg,br->br_array[idx][idy],flag,1,br);

    if (i<=cur_seg->xamid)
        // only subtract up to xamid in x-direction

```

```

{
    idx=cur_seg->xmid+i*mult[2][0];
    idy=cur_seg->ymid+j*mult[2][1];
    if(idx<0||idy<0||idx>XSIZE-1||idy>YSIZE-1)
    {
        cout<<"Error, array out of bounds shoot_border j
                                         loop\n";
        exit(1);
    }

    if(br->br_array[idx][idy]!=NULL)
        array_over(cur_seg,br->br_array[idx][idy],
                    flag,1,br);
}
} //end j1
for(j=1;j<=bord_dist;j++)
{
    /* can add only bord_dist in the y-direction, but only up to xmid in
    the x-direction*/
    idx=cur_seg->xmid+i*mult[0][0];
    idy=cur_seg->ymid+j*mult[0][1];
    if(idx<0||idy<0||idx>XSIZE-1||idy>YSIZE-1)
    {
        cout<<"Error, array out of bounds shoot_border j
                                         loop\n";
        exit(1);
    }

    if(br->br_array[idx][idy]!=NULL)
        array_over(cur_seg,br->br_array[idx][idy],flag,1,br);

    if(i<=cur_seg->xmid)
    //only subtract up to xmid in x-direction
    {
        idx=cur_seg->xmid+i*mult[1][0];
        idy=cur_seg->ymid+j*mult[1][1];
        if(idx<0||idy<0||idx>XSIZE-1||idy>YSIZE-1)
        {
            cout<<"Error, array out of bounds shoot_border j
                                         loop\n";
            exit(1);
        }

        if(br->br_array[idx][idy]!=NULL)
            array_over(cur_seg,br->br_array[idx][idy],flag,1,br);
    }
} //end j2
} //end i
break;

default:
    cout<<"wrong flag in shoot_border second switch\n";
    break;

} // end switch2

```

```

} // end switch

/*****shoot_X2border*****/
/* Navigate the array for shoots that are at the upper x border */
/* of the array */
/*****/
void SCU::shoot_X2border(branch *br, segs *cur_seg, int flag,
                        int flag2)
{
    int idx, idy;
    int i, j;
    double mult[4][2];

    int bord_xdist, bord_ydist;

    bord_xdist = (XSIZE-1) - cur_seg->xmid;

    mult[0][0] = 1; //add to idx
    mult[0][1] = 1; //add to idy

    mult[1][0] = -1; // subtract from idx
    mult[1][1] = 1; // add to idy

    mult[2][0] = -1; // subtract from both idx and idy
    mult[2][1] = -1;

    mult[3][0] = 1; // add to idx
    mult[3][1] = -1; // subtract from idy

    idx = cur_seg->xmid;
    idy = cur_seg->ymid;

    switch(flag2)
    {
    case(0):
    /* uninhibited in the y-direction, inhibited in adding to the x-
    direction*/
        for(i=1; i<=ARR_SEARCH; i++)
        {
            idx = cur_seg->xmid;
            idy = cur_seg->ymid+i;
            if(idx<0 || idy<0 || idx>XSIZE-1 || idy>YSIZE-1)
            {
                cout<<"Error, array out of bounds shoot_border\n";
                exit(1);
            }

            if(br->br_array[idx][idy] != NULL)
                array_over(cur_seg, br->br_array[idx][idy], flag, 1, br);

            idy = cur_seg->ymid-i;
            if(idx<0 || idy<0 || idx>XSIZE-1 || idy>YSIZE-1)

```

```

{
    cout<<"Error, array out of bounds shoot_border\n";
    exit(1);
}

if(br->br_array[idx][idy]!=NULL)
    array_over(cur_seg,br->br_array[idx][idy],flag,1,br);

idy=cur_seg->ymid;
if(i<=bord_xdist)
{
    idx=cur_seg->xmid+i;
    if(idx<0||idy<0||idx>XSIZE-1||idy>YSIZE-1)
    {
        cout<<"Error, array out of bounds shoot_border\n";
        exit(1);
    }

    if(br->br_array[idx][idy]!=NULL)
        array_over(cur_seg,br->br_array[idx][idy],flag,1,br);
}

idx=cur_seg->xmid-i;
if(idx<0||idy<0||idx>XSIZE-1||idy>YSIZE-1)
{
    cout<<"Error, array out of bounds shoot_border\n";
    exit(1);
}
if(br->br_array[idx][idy]!=NULL)
    array_over(cur_seg,br->br_array[idx][idy],flag,1,br);

for(j=1;j<=ARR_SEARCH;j++)
{
    if(i<=bord_xdist)
    {
        // can add to both directions
        idx=cur_seg->xmid+i*mult[0][0];
        idy=cur_seg->ymid+j*mult[0][1];
        if(idx<0||idy<0||idx>XSIZE-1||idy>YSIZE-1)
        {
            cout<<"Error, array out of bounds shoot_border j
                                loop\n";
            exit(1);
        }

        if(br->br_array[idx][idy]!=NULL)
            array_over(cur_seg,br->br_array[idx][idy],flag,1,br);

        idx=cur_seg->xmid+i*mult[3][0];
        idy=cur_seg->ymid+j*mult[3][1];
        if(idx<0||idy<0||idx>XSIZE-1||idy>YSIZE-1)
        {
            cout<<"Error, array out of bounds shoot_border j
                                loop\n";
            exit(1);
        }
    }
}

```

```

    }

    if(br->br_array[idx][idy]!=NULL)
        array_over(cur_seg,br->br_array[idx][idy],flag,1,br);
}

idx=cur_seg->xmid+i*mult[1][0];
idy=cur_seg->ymid+j*mult[1][1];
if(idx<0||idy<0||idx>XSIZE-1||idy>YSIZE-1)
{
    cout<<"Error, array out of bounds shoot_border j
                                         loop\n";
    exit(1);
}

if(br->br_array[idx][idy]!=NULL)
    array_over(cur_seg,br->br_array[idx][idy],flag,1,br);

idx=cur_seg->xmid+i*mult[2][0];
idy=cur_seg->ymid+j*mult[2][1];
if(idx<0||idy<0||idx>XSIZE-1||idy>YSIZE-1)
{
    cout<<"Error, array out of bounds shoot_border j
                                         loop\n";
    exit(1);
}

if(br->br_array[idx][idy]!=NULL)
    array_over(cur_seg,br->br_array[idx][idy],flag,1,br);
} //end j1
} //end i
break;

case(1):
//limited in adding in the x-direction, subtracting in the y-direction
for(i=1;i<=ARR_SEARCH;i++)
{
    idx=cur_seg->xmid;
    idy=cur_seg->ymid+i;
    if(idx<0||idy<0||idx>XSIZE-1||idy>YSIZE-1)
    {
        cout<<"Error, array out of bounds shoot_border\n";
        exit(1);
    }

    if(br->br_array[idx][idy]!=NULL)
        array_over(cur_seg,br->br_array[idx][idy],flag,1,br);

    if(i<=cur_seg->ymid)
// can only subtract up to ymid in the y-direction
    {
        idy=cur_seg->ymid-i;
        if(idx<0||idy<0||idx>XSIZE-1||idy>YSIZE-1)
        {
            cout<<"Error, array out of bounds shoot_border\n";

```



```

        exit(1);
    }

    if(br->br_array[idx][idy]!=NULL)
        array_over(cur_seg,br->br_array[idx][idy],flag,1,br);
}

idy=cur_seg->yamid;

if(i<=bord_xdist)
// can only add up to bord_xdist in the x-direction
{
    idx=cur_seg->xmid+i;
    if(idx<0||idy<0||idx>XSIZE-1||idy>YSIZE-1)
    {
        cout<<"Error, array out of bounds shoot_border\n";
        exit(1);
    }

    if(br->br_array[idx][idy]!=NULL)
        array_over(cur_seg,br->br_array[idx][idy],flag,1,br);
}

idx=cur_seg->xmid-i;
if(idx<0||idy<0||idx>XSIZE-1||idy>YSIZE-1)
{
    cout<<"Error, array out of bounds shoot_border\n";
    exit(1);
}

if(br->br_array[idx][idy]!=NULL)
    array_over(cur_seg,br->br_array[idx][idy],flag,1,br);

for(j=1;j<=ARR_SEARCH;j++)
{
    /* can add to all 6 spaces in the y-direction, but only up to
    bord_xdist in the x-direction*/
    if(i<=bord_xdist)
    {
        idx=cur_seg->xmid+i*mult[0][0];
        idy=cur_seg->yamid+j*mult[0][1];
        if(idx<0||idy<0||idx>XSIZE-1||idy>YSIZE-1)
        {
            cout<<"Error, array out of bounds shoot_border j
                                loop\n";
            exit(1);
        }
        if(br->br_array[idx][idy]!=NULL)
            array_over(cur_seg,br->br_array[idx][idy],flag,1,br);
    }
}

// can subtract all 6 the x-direction
idx=cur_seg->xmid+i*mult[1][0];
idy=cur_seg->yamid+j*mult[1][1];
if(idx<0||idy<0||idx>XSIZE-1||idy>YSIZE-1)

```

```

    {
        cout<<"Error, array out of bounds shoot_border j
                                   loop\n";
        exit(1);
    }
    if(br->br_array[idx][idy]!=NULL)
        array_over(cur_seg,br->br_array[idx][idy],flag,1,br);
} //end j1

for(j=1;j<=cur_seg->yamid;j++)
{
    if(i<=bord_xdist)
// can only add up to bord_xdist, subtract up tp ymid
    {
        idx=cur_seg->xmid+i*mult[3][0];
        idy=cur_seg->yamid+j*mult[3][1];
        if(idx<0||idy<0||idx>XSIZE-1||idy>YSIZE-1)
        {
            cout<<"Error, array out of bounds shoot_border j
                                   loop\n";
            exit(1);
        }
        if(br->br_array[idx][idy]!=NULL)
            array_over(cur_seg,br->br_array[idx][idy],flag,1,br);
    }

    idx=cur_seg->xmid+i*mult[2][0];
    idy=cur_seg->yamid+j*mult[2][1];
    if(idx<0||idy<0||idx>XSIZE-1||idy>YSIZE-1)
    {
        cout<<"Error, array out of bounds shoot_border j
                                   loop\n";
        exit(1);
    }
    if(br->br_array[idx][idy]!=NULL)
        array_over(cur_seg,br->br_array[idx][idy],flag,1,br);
} //end j2
} //end i
break;

case(2): // restricted in adding both cases, unrestricted in
subtracting
    bord_ydist=(YSIZE-1)-cur_seg->yamid;
    for(i=1;i<=ARR_SEARCH;i++)
    {
        idx=cur_seg->xmid;
        if(i<=bord_ydist)
        {
            idy=cur_seg->yamid+i;
            if(idx<0||idy<0||idx>XSIZE-1||idy>YSIZE-1)
            {
                cout<<"Error, array out of bounds shoot_border\n";
                exit(1);
            }
        }
    }

```



```

    }
    // can subtract all 6 in both directions
    idx=cur_seg->xmid+i*mult[2][0];
    idy=cur_seg->ymid+j*mult[2][1];
    if(idx<0||idy<0||idx>XSIZE-1||idy>YSIZE-1)
    {
        cout<<"Error, array out of bounds shoot_border j
                                   loop\n";
        exit(1);
    }

    if(br->br_array[idx][idy]!=NULL)
        array_over(cur_seg,br->br_array[idx][idy],flag,1,br);

    }//end j1
    for(j=1;j<=bord_ydist;j++)
    {
    // can subtract all 6 in the y-direction
        if(i<=bord_xdist)
    // only add up to bord_xdist in the x-direction
        {
            idx=cur_seg->xmid+i*mult[0][0];
            idy=cur_seg->ymid+j*mult[0][1];
            if(idx<0||idy<0||idx>XSIZE-1||idy>YSIZE-1)
            {
                cout<<"Error, array out of bounds shoot_border j
                                           loop\n";
                exit(1);
            }

            if(br->br_array[idx][idy]!=NULL)
                array_over(cur_seg,br->br_array[idx][idy],flag,1,br);
        }

        idx=cur_seg->xmid+i*mult[1][0];
        idy=cur_seg->ymid+j*mult[1][1];
        if(idx<0||idy<0||idx>XSIZE-1||idy>YSIZE-1)
        {
            cout<<"Error, array out of bounds shoot_border j
                                           loop\n";
            exit(1);
        }

        if(br->br_array[idx][idy]!=NULL)
            array_over(cur_seg,br->br_array[idx][idy],flag,1,br);
    }//end j2
    }//end i
    break;

default:
    cout<<"wrong flag in shoot_border second switch\n";
    break;

    }// end switch2
} // end switch

```

```

/*****plot_branch*****/
/* Writes segment information to a file for branch maps. */
/* When PLOTBRANCH==1 */
/*****/
void SCU::plot_branch(branch *br, int pop, int run)
{
    char sgspplot[21];
    char population[10];
    char year[10];
    int plot_yr;
    plot_yr=ceil(br->yr);
    sprintf(population,"%d",pop);
    sprintf(year,"%d",plot_yr);

    strcpy(sgspplot,"c:\\segout");
    strcat(sgspplot,year);
    strcat(sgspplot,"Pop");
    strcat(sgspplot,population);
    strcat(sgspplot,".txt\0");
    FILE *fpname;
    fpname=fopen(sgspplot,"w");
    fclose(fpname);

    char epiplot[21];
    strcpy(epiplot,"epiPlot");
    strcat(epiplot,year);
    strcat(epiplot,"Pop");
    strcat(epiplot,population);
    strcat(epiplot,".txt\0");
    FILE *fpepi;
    fpepi=fopen(epiplot,"w");
    fclose(fpepi);

    segs *cur_seg;

    cur_seg=br->shoot_head;

    SCU *cur_scu;
    cur_scu=br->scu_head;

    while(cur_scu!=NULL)
    {
        plot_rekurs(cur_scu->shoot_head,br,pop,run);
        cur_scu=cur_scu->next_scu;
    }
}

```

```

/*****plot_rekurs*****/
/* Writes segment information to a file for branch maps. */
/* When PLOTBRANCH==1 */
/*****/
void SCU::plot_rekurs(segs *cur_seg, branch *br, int pop, int run)
{
    FILE *fp;
    char sgsp[21];
    char population[10];
    char year[10];
    int plot_yr;
    plot_yr=ceil(br->yr);
    sprintf(population,"%d",pop);
    sprintf(year,"%d",plot_yr);

    strcpy(sgsp,"c:\\segout");
    strcat(sgsp,year);
    strcat(sgsp,"Pop");
    strcat(sgsp,population);
    strcat(sgsp,".txt\0");

    char epiplot[21];
    strcpy(epiplot,"epiPlot");
    strcat(epiplot,year);
    strcat(epiplot,"Pop");
    strcat(epiplot,population);
    strcat(epiplot,".txt\0");

    FILE *epifp;
    while(cur_seg!=NULL)
    {
        if(EPIPLOT==1&&cur_seg->init==1)
        {
            epifp=fopen(epiplot,"a");
            fprintf(epifp,"%d\t%d\t%d\t%d\t%f\t%f\t%f\t%f\t%f\t\n",run,
                cur_seg->init,cur_seg->scu_base,cur_seg->segNum,
                cur_seg->order,cur_seg->gen,cur_seg->tform,
                cur_seg->path_length,cur_seg->age);
            fclose(epifp);
        }
        if(PLOTBRANCH==1)
        {
            if(cur_seg->mort==0)
            { //22 items printed
                fp=fopen(sgsp,"a");
                fprintf(fp,"%f\t%f\t%f\t%f\t%d\t%f\t%f\t%f\t%f\t\t
                    %f\t%f\t%f\t%f\t%f\t%f\t%d\t%f\t%d\t%f\t%d\t\t
                    %f\t%f\t%d\t%d\n",cur_seg->x1,cur_seg->x2,
                    cur_seg->y1,cur_seg->y2,cur_seg->mort,cur_seg->gen,
                    cur_seg->xll,cur_seg->xlr,cur_seg->yll,
                    cur_seg->y1r,cur_seg->nover,cur_seg->xul,
                    cur_seg->xur,cur_seg->yul,cur_seg->yur,run,
                    cur_seg->tform,cur_seg->init,br->yr,cur_seg->segNum,
                    cur_seg->par_scu->SCUnum,cur_seg->order,cur_seg->age,

```


Vita

Education:

Ph.D. University of Washington, Seattle

Quantitative Ecology and Resource Management. Expected 2008

Multi-objective Optimization for Ecological Model Assessment and Theory Development

E. David Ford (chair), Tom Hinckley, Mark Kot, Charles Laird

M.S. University of Washington, Seattle

Quantitative Ecology and Resource Management. August 2002

A geometric simulation model of foliage regeneration in Abies grandis and Pseudotsuga menziesii.

E. David Ford, committee chair

B.S. University of San Francisco

Biology. May 1999.

Summa Cum Laude.

Publications:

Kennedy MC, Ford ED, Singleton P, Finney M, Agee JK. (2008) Informed multi-objective decision-making in environmental management using Pareto optimality. **Journal of Applied Ecology**. 45(1):181-192.

Lehmkuhl J, Kennedy M, Ford ED, Singleton PH, Gaines WL, Lind RL. (2007). Seeing the forest for the fuel: Integrating ecological values and fuels management. **Forest Ecology and Management**. 246: 73-80.

Ishii HT, Ford ED, Kennedy MC. (2007). Physiological and ecological implications of adaptive reiteration as a mechanism for crown maintenance and longevity. **Tree Physiology**. 27:455-462.

Kennedy MC, Ford ED, Ishii H. (2004). Model analysis of the importance of reiteration for branch longevity in *Pseudotsuga menziesii* compared with *Abies grandis*. **Canadian Journal of Botany**. 82: 892-909.