Investigating Model Analysis

A critical examination of current methods for model behavioural analysis in System Dynamics

by

Willem Geert Phaff

Master Thesis Project, Faculty of Technology, Policy and Management, Delft University of Technology
June 2006, Delft
Investigating Model Analysis

A critical examination of current methods for model behavioural analysis in System Dynamics
Preface

This report is the final product of my MSc. thesis project at the faculty Of Technology, Policy and Management. The focus of this work is the assessment and improvement of methods for model behavioural analysis in System Dynamics. Our models are simplifications of reality, built to help us understand a complex world. Model behavioural analysis aims to provide insight into the workings of these crude simplifications. In the end, this insight might help us to understand the complex world they represent.

The reader interested in the background of the problem is referred to the first three chapters of this report. Chapters four through five contain a description of the theoretical concepts needed for model behavioural analysis and an introduction to the techniques themselves. In Chapters six and seven I describe the application of the methods, both the framework used and the application of the methods to two models. Conclusions and recommendations following from this research can be found in Chapters eight and nine.

Model behavioural analysis proved to be challenging and difficult subject matter. The invested effort did bear its fruits however; it provided me with the opportunity to submit a conference paper and have it accepted. In addition, I had the privilege of giving a guest lecture on eigenvalue elasticity analysis in the Advanced System Dynamics course, which I will never forget.

I would like to express my gratitude to my first supervisor, Jill Slinger; without her support and advice, I probably would have never made it through. In addition, I would like to thank the other members of the Committee for their invaluable advice.

Special thanks go to my friends and family, who have unconditionally supported me in this stage of my life.

Willem Geert Phaff
‘gringo temático antikaima’
June 2006

“Leuk hè?” – Oom Cor
Executive Summary

An idea central to System Dynamics (SD) is that the structure of a system determines its behaviour. That is, that the interplay of levels, feedback loops and delays gives rise to the behaviour of the system. Despite the centrality of this idea, determining how structure gives rise to behaviour still poses a challenge to even the most experienced of modellers.

Indeed, more than 40 years after SD was first developed, methods for System Dynamics model analysis are still under development. Yet there is a gap between the relative importance of understanding the causes of model behaviour and available analysis techniques, whether formal or not.

In recent years several methods have been developed in an attempt to fill this gap. These methods range from relatively accessible approaches to those that are more mathematically based.

In spite of the existence of these algorithms and tools, they are not applied routinely and are still in the experimental stage of development. At present, the methods are not developed enough to bring the modeller closer to fully understanding the model. The goal of this study is the assessment and improvement of the applicability of existing formal methods for model analysis.

After an initial survey of existing methods, the most promising methods for model analysis were selected for further analysis. After adjusting these methods and developing a framework to support automation, the improved methods were evaluated using two models as test cases. During the testing and implementation process expert consultation with experts took place.

Of the methods for model analysis, Ford’s behavioural approach and eigenvalue elasticity analysis proved to be the most promising for formal model analysis.

**Ford’s behavioural approach**
Ford’s method of taking out loops during specific intervals of behaviour of a variable of interest provides the analyst with a systematic and relatively easy method of assessing the dominance of a loop in determining model behaviour. At its core,
the method is an extended form of an extreme condition test, where the extreme condition is the sudden elimination of a specific loop. The difference between the model run without the loop and the original model is used to assess the effect of the loop on a specific variable of interest.

**Eigenvalue Elasticity Analysis**

The Eigenvalue Elasticity Analysis uses the fact that, at a given point in time, the behaviour of a model is described by eigenvalues of the linearised system. The method links model structure or parameters to behaviour by determining how these eigenvalues respond to a perturbation of loops and parameters. The concept central to the analysis is the elasticity of an eigenvalue: the response of the eigenvalue relative to a change in the loop gain or parameter value. The elasticities can be related to the behaviour of a specific variable or the system as a whole.

Both selected methods required significant improvement. Ford’s behavioural approach has aspects that require sharper definition, while eigenvalue elasticity analysis still faces several problems with regards to implementation and the singularity of the matrix of right eigenvectors. In order to perform all variants of the eigenvalue elasticity analysis, it was necessary to design and implement a framework for model analysis. This allowed for a consistent and flexible execution of, at least, the eigenvalue elasticity analysis methods.

The strongest argument for the routine application of Ford’s method is how accessible it is. The mechanism of deactivating loops is readily explicable and the concept of a behavioural pattern (whether a variable displays balancing or reinforcing behaviour) agrees with an intuitive understanding of model behaviour.

However, the method is marred by several fundamental weaknesses. First of all, owing to the time-consuming nature of the method, only a selection of the loops can be analysed in larger models. This makes the method methodologically weaker, since it cannot test the role of loops outside of the initial selection.

Second, two assumptions that the method is based on are not valid. Given the method’s focus on intervals displaying a single behavioural pattern, the first assumption is that a change in driving structure is always accompanied by a change in behavioural pattern and vice versa. However, this is not necessarily true. The second assumption is that individual loops can always be eliminated independently. Eliminating one loop may affect the behaviour of another loop though, making it impossible to analyse the effects of individual loops separately.

Overall, the exercise of systematically taking out loops does provide qualitative insight into the role of the eliminated loop, which is useful.

The most significant contribution of eigenvalue elasticity analysis is the visuali-
sation of the fractional influence of loops on eigenvalues and state variables over time. The results can be displayed as a (semi-)continuous graph over time, tracing the influence of structure on behaviour, if automated support is present. Assuming that the model is in analysable form, obtaining these results is only a matter of running the appropriate functions.

In addition, the several forms in which the method is available do not exclude others, so different forms can compensate for each others’ weaknesses. The analyst can choose to perform the method from a system-wide perspective or focus his/her interest on one specific state variable.

Understanding eigenvalue elasticity analysis still poses a challenge to the analyst, however. The major hurdle in applying eigenvalue elasticity analysis is the knowledge required for understanding the technique and interpreting its results.

Eigenvalue elasticity analysis seems to be analytically the strongest method currently available, despite the challenges it still faces regarding its theoretical foundations and practical implementation. However, it is still very much in the experimental phase, necessitating further research into model behavioural analysis, the underlying theory and the application of model analysis to the modelling process.
vi
# Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preface</td>
<td>i</td>
</tr>
<tr>
<td>Executive Summary</td>
<td>iii</td>
</tr>
<tr>
<td>1 Introduction</td>
<td>1</td>
</tr>
<tr>
<td>1.1 Contents</td>
<td>3</td>
</tr>
<tr>
<td>1.2 Audience</td>
<td>3</td>
</tr>
<tr>
<td>2 Problem Description</td>
<td>4</td>
</tr>
<tr>
<td>2.1 Context: System Dynamics</td>
<td>4</td>
</tr>
<tr>
<td>2.2 The System Dynamics Modelling Process</td>
<td>5</td>
</tr>
<tr>
<td>2.2.1 Iteration</td>
<td>6</td>
</tr>
<tr>
<td>2.2.2 Problem Definition</td>
<td>7</td>
</tr>
<tr>
<td>2.2.3 Conceptualisation</td>
<td>7</td>
</tr>
<tr>
<td>2.2.4 Specification</td>
<td>7</td>
</tr>
<tr>
<td>2.2.5 Testing</td>
<td>8</td>
</tr>
<tr>
<td>2.2.6 Explanation of Model behaviour</td>
<td>8</td>
</tr>
<tr>
<td>2.2.7 Policy Analysis</td>
<td>9</td>
</tr>
<tr>
<td>2.3 The Need For Formal Model Analysis</td>
<td>9</td>
</tr>
<tr>
<td>2.3.1 Uses for Model Analysis</td>
<td>9</td>
</tr>
<tr>
<td>2.3.2 The Gap in the System Dynamics Paradigm</td>
<td>11</td>
</tr>
<tr>
<td>2.4 The Problem</td>
<td>12</td>
</tr>
<tr>
<td>3 Research Setup</td>
<td>14</td>
</tr>
<tr>
<td>3.1 Goal</td>
<td>14</td>
</tr>
<tr>
<td>3.2 Limitations</td>
<td>14</td>
</tr>
<tr>
<td>3.3 Questions</td>
<td>15</td>
</tr>
<tr>
<td>3.4 Method</td>
<td>16</td>
</tr>
<tr>
<td>3.4.1 Phase I: Exploration</td>
<td>16</td>
</tr>
<tr>
<td>3.4.2 Phase II: Analysis</td>
<td>17</td>
</tr>
</tbody>
</table>
## 4 Preliminary Concepts and Techniques

4.1 A Graph Theory View of System Dynamics Models
   4.1.1 Central Concepts
   4.1.2 The Loops In The Model
   4.1.3 Causal Paths
4.2 Equation-based view of System Dynamics Models
   4.2.1 Central Concepts
   4.2.2 Linearisation
   4.2.3 The Relation between Eigenvalues and Behaviour

## 5 Understanding Behaviour

5.1 Behavioural approach
   5.1.1 Description
   5.1.2 Review
5.2 Pathway Participation Metrics
   5.2.1 Description
   5.2.2 Review
5.3 Eigenvalue Elasticity Analysis
   5.3.1 Parameter based analysis
   5.3.2 Loop based analysis
   5.3.3 From Eigenvalues to Variables
   5.3.4 Review
5.4 Conclusions

## 6 A Framework for Model Analysis

6.1 The Need
6.2 Requirements
6.3 Conceptual Design
   6.3.1 Information Transferred
   6.3.2 The Overall Procedure
   6.3.3 The Model
   6.3.4 Internal Model Structure
6.4 The Analysis Functions
6.5 Prototype
6.6 Conclusions

## 7 Performing The Analyses

7.1 Selecting The Test Models
7.2 Yeast
   7.2.1 Ford
   7.2.2 Loop Eigenvalue Elasticity
CONTENTS

7.2.3 Parameter Eigenvalue Elasticity ........................................ 94
7.2.4 Elasticities to states ..................................................... 98
7.3 Long Wave ................................................................. 104
  7.3.1 Preliminary Analysis for the Eigenvalue Elasticity Analysis . 105
  7.3.2 Loop Eigenvalue Elasticity Analysis ............................... 109
  7.3.3 Parameter Eigenvalue Elasticity Analysis ......................... 118
  7.3.4 Relating eigenvalues to states .................................... 125
  7.3.5 Ford ................................................................. 135
7.4 Conclusions ............................................................... 144
  7.4.1 Framework ........................................................... 144
  7.4.2 Eigenvalue Elasticity Analysis ................................... 145
  7.4.3 Ford ................................................................. 148

8 Conclusions ........................................................................ 150
  8.1 Promising Methods Available .......................................... 151
    8.1.1 Ford's behavioural approach ..................................... 151
    8.1.2 Eigenvalue Elasticity Analysis .................................. 152
  8.2 Improvements In The Methods ......................................... 153
    8.2.1 The Framework .................................................... 153
    8.2.2 Ford's Behavioural Approach .................................... 154
    8.2.3 Eigenvalue Elasticity Analysis .................................. 155
  8.3 Assessment of the Methods ............................................. 156
    8.3.1 Ford's Behavioural Approach .................................... 156
    8.3.2 Eigenvalue Elasticity Analysis .................................. 158
    8.3.3 Comparative Analysis ............................................ 160
  8.4 Final Remarks ............................................................ 161

9 Recommendations ................................................................ 162
  9.1 Theory ........................................................................... 162
  9.2 Implementation ............................................................ 163
  9.3 Application ................................................................. 164
  9.4 Research Process .......................................................... 166

Bibliography ........................................................................... 168

A The Example Models .......................................................... 173
  A.1 Predator - Prey .............................................................. 173
    A.1.1 System Diagram ..................................................... 174
    A.1.2 Equations ............................................................ 175
    A.1.3 Directed Cycle Matrix ............................................. 176
  A.2 Yeast ............................................................................. 176
Chapter 1

Introduction

Why does a system exhibit undesirable behaviour? Why do policy options that should work have unexpected detrimental effects? What are the underlying dynamics of the solutions that do work? One of the ideas central to System Dynamics (SD) is that the structure of a system determines its behaviour (Richardson and Pugh, 1981; Sterman, 2000; Senge, 1990). However, all of the - fairly commonly occurring - questions above relate to the problem of how structure gives rise to behaviour. Solving this problem poses a challenge to even the most experienced of modellers. In this thesis, the process of determining the link between structure and behaviour is referred to as model analysis.

In System Dynamics, model analysis plays a substantial role in determining the final product offered to a client, and is a part of several steps in the modelling process. The strength of the System Dynamics paradigm comes from the combination of both structure and behaviour; alone they hold little power (Meadows, 1980). The output of a SD study does not just consist of a ready-to-implement set of policy options, but of a combination of strategies and insights into the workings of the system being modelled. The goal is not just to offer solutions, but to explain the dynamics of the system. Model analysis is central to providing insight into these dynamics. The explanation offered can be crucial to model acceptance, especially if the conclusions go against the firmly held beliefs of the client.

An even broader role can be envisaged for model analysis during stages in the modelling process. For instance, the goal of model validation is to find out whether the “right” behaviour is generated for the right reasons. So knowing the cause of behaviour helps with the decision as to whether or not to refute the model. For example, if a growth rate in a real system declines due to a known limiting condition, that same condition should be the cause of the same behaviour in the model. Just displaying the relevant behaviour is not enough, the validity of the model depends
on the reasons for that behaviour.

Furthermore, the strategies designed should be actual solutions, not just temporary fixes that suppress the symptoms of the problem. For example, a policy strategy could have knocked the model out of equilibrium, making the solution appear to be effective, while the system will return to the previous equilibrium state in the long run. Insight into the dynamics of a model is necessary to evaluate solutions in a way that is consistent with the System Dynamics paradigm.

To summarise, finding out how system structure gives rise to behaviour is an important aspect of several steps in the modelling process. The communication of these findings can be instrumental to the impact of the model.

Yet, at this point in time, more than 40 years after SD was developed, techniques for System Dynamics model analysis are still under development. Traditionally practitioners have relied on intuition, experience and the building and testing of hypotheses, as described in Richardson and Pugh (1981). There seems to be a gap between the relative importance of understanding model behaviour and available analysis techniques, formal or not. Sterman (2000), Richardson (1996) and Saleh (2000) have identified this as one of the main research areas for system dynamics, central to the methodology.

In recent years several attempts have been made to fill this gap. Methods focus on identifying feedback structure responsible for a behavioural mode of a variable of interest. These methods range from relatively easy approaches that manually eliminate loops from a model to those that are more mathematically based, using linearisations and analysing the relationship between the structure of the system and its eigenvalues.

However, in spite of the development of algorithms and tools for these methods, their routine application is limited at best. The conditions for the application of a particular method are not fully specified and it is unclear which analysis method is most suitable and when.

This thesis aims to provide insight into the application of the methods of formal model analysis. The results include an overview of the advantages and disadvantages of each method.

To obtain these results, an overview was made of the methods currently available. Of these, the most promising were selected and subjected to further analysis. An analysis framework was designed and implemented to enable a consistent and flexible approach to formal model analysis. Using this, the methods were reviewed and conclusions regarding their applicability were drawn.

The path to deriving these results was not as smooth as expected. The methods

\[^1\text{Saleh (2000) even states that the System Dynamics research programme is ‘under threat of reaching a ‘crisis’ state’.}^\]
turned out to be less developed than initial impressions suggested. It was necessary to design a consistent approach to the implementation of analysis methods. Accordingly, the designing of a framework for model analysis formed a larger component of the research than anticipated at the outset of the study.

1.1 Contents

The first three chapters deal with the problem the research aims to address, research set-up and the approach taken during the study. The problem description provides an overview of the System Dynamics methodology and the related modelling process, after which the need for formal model analysis is discussed. It concludes with the problem statement for this research. The research setup contains a description of the goal of the research, its limitations, the questions addressed, and an overview of the approach taken.

The second part of the thesis focusses on a description and analysis of the available methods for model analysis. Chapter 4 briefly describes the concepts required for the deeper methods of analysis. Chapter 5 deals with the methods for analysing model behaviour. A number of these methods is taken into the next tier of analysis. The framework developed to enable comparison of the methods for model analysis is presented in Chapter 6. The in-depth analysis of the methods, using the formal framework, are presented in Chapter 7.

Finally, the conclusions and recommendations are presented in Chapter 8 and Chapter 9.

1.2 Audience

The readers of this thesis are assumed to be modellers with a strong interest in formal model analysis. To these, it offers an overview of the currently available methods, and insights into strengths and weaknesses of each method. In addition, the framework designed provides the analyst with a prototype tool for the analysis of models.
Chapter 2

Problem Description

The problem at the heart of this research is described in this chapter. An overview of the fundamentals of System Dynamics and the modelling process is presented and the need for formal model analysis explained.

2.1 Context: System Dynamics

System Dynamics (SD) is a methodology for describing, studying and analysing complex feedback systems. It was developed at MIT in the 50s, primarily by Jay W. Forrester. Borrowing from the then recently developed fields of control engineering, cybernetics and organisational theory, Forrester developed a methodology for modelling complex non-linear systems. It has since been applied to a wide variety of (mostly social) systems, from industrial dynamics (Forrester, 1961) to peer-to-peer file-sharing (Pavlov and Saeed, 2004), from the fishing industry in Gloucester, Massachusetts to physiological models of human weight gain and loss (Abdel-Hamid, 2002).

Instead of focusing on exact values of variables, System Dynamics modellers are more interested in the general dynamic behaviour of “their” systems. They concern themselves more with the question of whether a system stabilises, oscillates or grows exponentially than with the value of a performance indicator at the end of a simulation run.

A fundamental assumption within the field is that the dynamic behaviour of a system is driven by its causal structure. Within this structure the feedback loop is seen as the most important determinant of behaviour. Generally, the closed nature of SD ensures that there are relatively few exogenous variables involved (see Section 2.2.3). The interaction of multiple loops, time delays and nonlinearities can give rise to complex behaviour in deceptively simple looking models.
The fundamental building blocks for formal model representations are stocks and flows. Stocks - also known as state variables or levels - generally represent the directly observable aspects of the system. Examples of these are the number of houses in a city, population or energy reserves. Rates are the changes of these stocks, flows of matter or information. Examples of rates are the construction rate of new estate, death rate or daily energy usage. In addition, modellers use auxiliary variables to keep the model readable. Popular presentation of SD models is in the form of system diagrams, which graphically show the structure of the underlying model equations. These diagrams make the models relatively understandable to the inexperienced modeller and clients and facilitate the identification of feedback loops (Figure 2.1). However, when quantifying, parameter values are assigned to the complex system of ordinary differential equations which are then solved numerically during simulation runs.

![Figure 2.1: A small Lotka-Volterra model](image)

### 2.2 The System Dynamics Modelling Process

This section contains a description of the System Dynamics modelling process. The view presented is the most commonly held one, although it is not the only
one\(^1\). Also, it is an abstraction of the actual work being done. In practice, the modeller usually deviates from or iterates through this fairly formal view. The process is generally understood to at least consist of the following six phases:

1. Problem definition  
2. Conceptualisation  
3. Formalisation  
4. Testing  
5. Explanation of Model behaviour  
6. Policy Analysis

However, authors have differing opinions on which phases should be included in the process and which not. For instance, Richardson and Pugh (1981) include the fifth phase, while this is absent from Sterman (2000). The fifth phase, “Explanation of model behaviour”, strongly related to the subject of this thesis, permeates large parts of the modelling process and so may be considered not to be restricted to a single phase. It can even be considered as one of the outcomes of the process (Richardson and Pugh, 1981). However, in explaining formal model analysis, I will follow Richardson and Pugh (1981) and distinguish 6 phases.

The communication of results to the client is left out of this overview; this is considered to be beyond the scope of the modelling process itself.

### 2.2.1 Iteration

SD is considered to be a methodology well suited for dealing with complex problems. It would be naïve to assume that when dealing with such systems, modellers move unerringly, step by step through the modelling process in one smooth iteration, choosing acceptable system boundaries and building a valid model that passes all tests flawlessly. In reality, the process is iterative in nature.

This is due to several reasons. First of all, model evaluation may reveal a need for revisions in model structure (Homer, 1996). Secondly, modelling a complex system is an explorative process and activities in most stages help in understanding the system (Richardson and Pugh, 1981). Consequently, knowledge generated in each step of the process feeds back and forward into other steps (Richardson and Pugh, 1981).

\(^1\) For instance, Lyneis (1999) proposes a method that makes client involvement a more specific part of the process. The core of the method is that it moves from small, insight-based models to more complex, detailed models.
2.2.2 Problem Definition

The articulation of a problem defines the perspective for the entire modelling study. It is impossible to represent an entire system, hence, a sharply defined problem is needed to provide focus for the modelling effort (Forrester, 1961, pg. 61).

2.2.3 Conceptualisation

The goal of the conceptualisation phase of the modelling process is to capture the feedback structure that can offer an endogenous explanation of the problem (Sterman, 2000). An endogenous explanation for the problematic behaviour is one where the problem rises from the model, not some exogenous factor.\(^2\)

The stage starts with an activity that forms a strong overlap between the problem definition and conceptualisation; the statement of model purpose. What is it we intend to accomplish by modelling the system? As the basis of modelling is abstraction, the model purpose serves as a guideline for what to leave out and what not. This in turn enables the modeller to define the system boundaries; choosing which aspects of the real world system to include or leave out.

With the above defined, the modeller moves on to the activity often seen as the core of the conceptualisation: the mapping of feedback structure in the form of diagrams. Maani and Cavana (2000) even refer to the entire phase as causal diagramming. More commonly used techniques are causal loop diagrams and stock and flow maps. Given the sheer size of SD models, most modellers use some form of submodelling to create an overview of the general structure of their models.

In addition to the above activities, both Sterman (2000) and Richardson and Pugh (1981) see the generation of the so-called dynamic hypothesis as central to the conceptualisation. Since the model is built to understand or help solve a specific problem, a theory about the causes of the problem serves in guiding the modelling process. The dynamic hypothesis is this theory.

2.2.4 Specification

The specification phase consists of building a formal model in the, (graphical) computer language of choice. It consists of translating the stock-flow structure to a numerically solvable system of Ordinary Differential Equations (ODE). To do this the modeller must a) formulate equations b) make assumptions where data is lacking and c) estimate parameters where data is present. The choices made when making assumptions have to be justifiable and should be tested later on in the modelling process.

\(^2\)System Dynamicists tend to see an exogenous explanation, where model behaviour is largely dependent on exogenous variables, as not an explanation at all. It merely shifts the question to why the exogenous variables behave as they do (Sterman, 2000).
CHAPTER 2. PROBLEM DESCRIPTION

Given the number of software packages and languages available today\(^3\), the eventual selection of the language in which the model is implemented is usually based on expected cost of implementing the model in that language and the quality of the model built in that language.

2.2.5 Testing

If the analyst has built a working model, two questions remain. 1): Has the correct translation from conceptualisation to a formal model been made? And 2): Is the model suited for the purposes it was built for? Answering the first question is referred to as verification, validation refers to answering the second.

The tests in these two phases reflect the question asked in each of them. Verification tests say nothing about the utility of the model. For example, a dimension analysis is only used to check equations, whether each variable in a model represents what it should (according to model conceptualisation). A model can be thoroughly verified, but utterly useless.

Validation\(^4\) uses its tests to check the model against the purpose formulated in the conceptualisation. For instance, any decent model should be able to reproduce historical data accurately enough for the function it is intended to fulfil. Another test the model usually is subjected to in this phase, is the sensitivity analysis, used to see whether the model behaviour changes significantly when values of parameters are changed.

2.2.6 Explanation of Model behaviour

Once the model has been declared suited for its purposes, the next phase consists of understanding the behaviour of the model. Typically, modellers have used informal, experimental approaches\(^5\), relying on the analyst’s expertise for establishing the relation between structure and behaviour and building and refuting hypotheses about model behaviour (Ford, 1999; Richardson and Pugh, 1981). For example, Richardson and Pugh (1981, page 268) mention setting system parameters to extreme values and predicting and describing the resulting model behaviour as one of the methods the analyst can employ.

Creating understanding of a complex system is usually a part of model purpose. If the model is valid, an understanding of model behaviour can be translated to the real world system. A project that has the sole purpose of understanding the reasons for certain undesirable behaviour, ends after this phase.

\(^3\)For instance, Powersim, Simile, Vensim, Stella, DYNAMO, MatLab, Mathematica, Maple, or even regular programming languages such as, python, Java, C++, Visual Basic, etc.

\(^4\)Many practitioners of SD actually prefer the term model utility instead of validity (Meadows, 1980)

\(^5\)The exception being Thissen (1978).
2.2.7 Policy Analysis

The policy analysis phase entails the generation and evaluation of policy alternatives. We can identify two forms of policy design; a) parameter changes and b) structural changes. The first involves changing those parameters identified as being under the control of the client. The sensitivity analysis can help in finding those parameters that are effective leverage points for the client. The second form of policy analysis uses the fact that, in SD, structure determines behaviour. Consequently, effective policies will often involve changes in feedback structure in the model, often using changes in the decision rules.

Once the policy options have been established, they undergo some form of evaluation, which should be consistent with model purpose and the perceived problem. Consistent with the philosophy of SD, the dynamics of a solution are more important than the values of a performance indicator. Also, different options may interact in an unpredictable fashion, so the interferences and synergies between policies should be investigated.

2.3 The Need For Formal Model Analysis

The modelling process described in the above paragraphs relies on the modeller’s understanding of the relation between model structure and behaviour in several of its phases. In addition to that, there is a methodological gap to be filled by formal model analysis.

2.3.1 Uses for Model Analysis

A broad role can be envisaged for formal model analysis during several stages in the modelling process. For instance, the insight into model dynamics can help in simplifying the model to a smaller form that still retains validity. This section describes several ways in which the analyst may use model analysis in the modelling process.

Testing System Boundaries

If the behaviour of the model is strongly dependent on an exogenous explanation, the system boundary we started out with might not be suitable for the model purpose. Assuming that we are looking for an endogenous explanation of behaviour, if we find that behaviour is for a large part dependent on an assumed relation, this

---

6This refers to the term policy analysis as used in Richardson and Pugh (1981); the generation and analysis of alternatives in the SD modelling process.
compromises the utility of the model. We are left with the question why the relation behaves as it does and our model has a weaker endogenous explanation (see Section 2.2.3).

For instance, if the gain of one of the dominant loops is largely dependent on a graph function with a lack of experimental data backing it up, the next question is why that function behaves as it does.

This has to be seen within the context of the purpose of the model under consideration. If, for example, the purpose of the model was to find direction for further research, the recommendation to conduct research on the influential edge of the model would be sufficient.

**Policy Analysis**

As stated above, parameters that are part of dominant loops are more likely to function as leverage points for policy makers. Model behavioural analysis can help in identifying these, along with their associated effects (Richardson, 1986). For example, in the Lotka-Volterra model, if we wanted to stabilise the population, our analysis would have pointed us in the direction of the starvation and predator death loops.

Aside from this fairly obvious application, insight into the dynamics underlying behaviour can aid the modeller in designing more reactive, intelligent strategies. The understanding of dynamics could provide insight into which options create interference and which synergy. Furthermore, it might be easier to define effective structural changes to the model, although the applicability of the analyses in the case of structural changes is not well established.

In addition, as mentioned in the introduction, model analysis also offers a solid foundation for the evaluation of alternatives based on model dynamics.

**Testing the dynamic hypothesis**

The dynamic hypothesis is an attempt to explain observed behaviour in terms of the model. However, the hypothesis may be falsified. The main reasons for specifying and simulating a model rely on our inability to predict the behaviour accurately enough either analytically or by using mental simulation (Diehl and Sterman, 1995; Jensen and Brehmer, 2003). Model analysis can be used to test whether the implemented model corroborates or refutes the hypothesis. This falls under the Model Explanation phase, with the dynamic hypothesis as input from the conceptualisation.
Simplifying the model

As stated by Martinez and Richardson (2001), one of best practices in model formulation is to try to find the smallest model that retains (structural) validity and explanatory power. It should be as small as possible, but still capture the dynamic hypothesis. Furthermore, model simplification has been proposed as a method for understanding model behaviour (Richardson, 1986; Eberlein, 1989). However, this leaves us with a “chicken and egg” problem. The analyst simplifies in order to understand the model, but he needs to understand the model in order to be able to simplify it. Model analysis can aid in identifying those parts of a model that can be left out without having much effect on model dynamics and thereby reducing the model to a smaller form.

Directing development

The opposite of this practice might also be a valuable addition to the modelling process. Parallel to testing the system boundaries, if the modeller is incrementally developing the model, understanding model behaviour can help in guiding the development into the regions most influential for behaviour (Thissen, 1978).

2.3.2 The Gap in the System Dynamics Paradigm

Any white box paradigm, such as SD, makes the structure of the modelled system explicit. Hence, in contrast to black box methods, a white box methodology provides insight into both structure and behaviour. Model analysis, formal or not, provides the opportunity to gain insight into the relationship between structure and behaviour.

However, the current, informal approach (see Section 2.2.6) in obtaining this insight relies, for a large part, on the experience the modeller has with the model when arriving at the stage at which he attempts to explain model behaviour. The process of problem definition, conceptualisation and building and testing the model leaves him with sufficient, often tacit, knowledge of the workings of the model. This results in a subjective explanation of behaviour, while the knowledge of the inner workings of the model largely resides with the person or team building the model. The hypotheses regarding the relation between structure and behaviour are formulated and tested using this knowledge.

Furthermore, performing the informal experiments (see Section 2.2.6) requires considerable effort on part of the analyst. This further erodes the conclusions based on the experimental approach, since a sufficiently thorough analysis of a

---

This is one of the best practices mentioned in Martinez and Richardson (2001) and in Pidd (1996)
large model is expensive, time consuming and not always feasible. Given the relatively smaller portion of a larger model that can be tested, the larger the model, the more likely it is that the model analysis is driven by subjective assumptions of the modeller.

To summarise, finding the relation between structure and behaviour is central to System Dynamics, but the informal process of establishing this relation is subjective, time consuming and the knowledge gained is hard to make explicit. A formal, objective method for understanding behaviour will result in a methodologically sounder modelling study. As such, the formal methods provide a valuable addition to the informal approach.

In addition, a formal method provides a powerful tool for dealing with misperceptions of feedback effects. Previous publications have shown that most people have trouble dealing with feedback systems (Diehl and Sterman, 1995; Jensen and Brehmer, 2003). So, even assuming a completely valid model of a system, the mental model a client or analyst has of its working may be incorrect. A subjective method, based on analyst assumptions is a weaker tool in dealing with personal theories regarding the origins of system behaviour held by the problem owner. A formal method is far more powerful in corroborating or falsifying the client’s perception of the workings of the system, assuming that the method is correct and trusted by the client.

2.4 The Problem

At this point in time, techniques for formal model analysis are still under development. Both Sterman (2000) and Richardson (1996) have identified the absence of these methods as one of the main research areas for system dynamics.

In recent years several attempts have been made to fill this gap, focusing on identifying feedback structure responsible for a behavioural mode of a variable of interest. Richardson (1995) proposed a rigorous definition of dominance, while Ford (1999) set up a behavioural approach to model analysis. A recent publication by Mojtabahzadeh et al. (2004) presents DIGEST, a software package used to identify influential system structure. Several authors have been working on building (automated) methods to perform eigenvalue-based methods for analysis (Kampmann, 1996a; Saleh, 2002; Güneralp, 2005; AbdelGawad et al., 2005).

However, in spite of the development of algorithms and tools, these are not applied routinely and are, apparently, still in the experimental stage of development. As the methods stand now, they are not mature enough to bring the modeller much closer
to fully understanding the model. The applicability of the methods is either unclear or fails to live up to its promises. Reasons for this include, amongst others:

- there are no guidelines when to use which method
- the comparative advantages of the methods are unclear
- there’s incomplete knowledge regarding the conditions under which the methods are applicable
- and, since the methods are still experimental and under development, the process by which they are actually applied and implemented leaves much to be desired;

To summarise the problem: there is a lack of knowledge concerning the application of model analysis techniques.
Chapter 3

Research Setup

3.1 Goal

Model analysis can be considered a crucial part of the System Dynamics paradigm. However, little is known about the effective use of current model analysis tools. A tool without a well understood application is less useful than it could be. Consequently, there is a need to deepen understanding of these analytical techniques, especially with regard to their application. The aim of this project is to assess and improve the applicability of formal methods of model analysis.

The question of how to do this remains. Merely analysing the algorithms and their mathematical background will not suffice, as this does not provide enough information about their actual use, while the application of model analysis to a few cases seems to suffer from a lack of generalisability. The methodological choices in this research attempt to deal with this dilemma.

3.2 Limitations

As this project forms part of an M.Sc. thesis, not all aspects of the problem can be addressed. Aspects that are considered to be beyond the scope of this study are:

- The development of new algorithms. No effort will be made to develop new methods of analysing structure and its connections to behaviour. This research will only deal with applying and evaluating the more formal, recent methods.

- The fit of model analysis to the modelling process is an area of research worth particular attention. For instance, one of the more obvious applications for model analysis is the use of the results of the analysis in the design of policy.
3.3. **QUESTIONS**

There is no literature on if, and how formal methods for model analysis can be used here. However, this particular subject is left for further research when methods for formal analysis are more mature.

- The research will limit itself to the more formal, recent methods available to the analyst. These are at the moment, still relatively untested and their applicability is unknown. In addition, the research will only deal with quantitative methods, the more qualitative variants of understanding systems (Senge, 1990) are considered to be beyond the scope of this research.

### 3.3 Questions

First of all, an inventory and exploratory analysis of available methods will be made. The most promising ones, based on a superficial scan, will be explored in the remainder of the research.

1. **What formal methods are available and which of these are most promising?**

In order to assess the applicability of the methods their strengths and weaknesses will be evaluated. Given the fact the link of the methods to the modelling process is left out, this research can only provide insight into the core process of the method. That is, a model is obtained, a method is applied to it and its results are evaluated; all of which requires resources in terms of knowledge, software and time. Consequently, this research will look at the strengths and weaknesses of the methods specifically in terms of the fundamentals of the method, the results provided, what models it can be applied to and what resources are required to execute the method.

2. **What are the strengths and weaknesses of these methods in terms of**
   a) the fundamentals of the method,
   b) results provided,
   c) the range of models they can be applied to,
   d) the resources required to perform the methods?

The remaining question concerns improving the methods, or, at least, identifying those areas where the methods can be improved.

3. **How can we improve the applicability of these methods?**

In practice an in-depth analysis of promising methods mentioned in question 3 was not feasible without a consistent approach to implementing the analysis methods. Hence, the order was changed to address this dilemma. This meant that the
design of a formal framework for model analysis was undertaken so as to be able to perform the analyses as part of the answer to question 4.

### 3.4 Method

Since the main research question of this focuses on the application of analytical techniques, some way must be found to test and improve the techniques. The method presented here involves a combination of theoretical exploration and empirical testing. It is divided into two phases: exploration and analysis.

#### 3.4.1 Phase I: Exploration

This phase of the research starts with research into relevant literature on model analysis. The goal is to find recently developed, promising methods. The survey will limit itself to recent, formal methods.

Secondly, an overview containing descriptions of available techniques will be presented in order to understand the workings of each (Figure 3.1).

Since the most promising methods are to be selected for the next phase, a quick yet consistent way of exploring the methods is needed. A very simple non-linear model was analysed using the initial list of methods to explore their workings. The advantages of this approach are:

- Applying the methods is necessary in order to understand them
- Using one model will provide consistency in the analysis of the methods
- A well chosen model can highlight the strengths and weaknesses of the methods
- The process will cost some extra time, but the resulting familiarity with the methods is used as an input for Phase II.

A simple, almost linear, model was used for this purpose; the Lotka-Volterra model. The most promising methods will be selected based on the outcome of the exploratory analysis. Methods will be evaluated on their internal consistency and ability to properly deal with the Lotka-Volterra model.

The phase finishes with a re-assessment of the research method. Given the fact that, at the beginning of this research, knowledge regarding the methods was low, I was not in the position to properly estimate the current status of the methods. It is not known how hard they are to apply, or if any, or how much, programming work is necessary obtain the desired results. Hence, the exploratory phase ends with a
re-assessment of the methodology and an inventory of what is needed to progress.

3.4.2 Phase II: Analysis

One of the outcomes of the exploratory analysis was that the methods for formal analysis are in a less mature state than expected. Hence, the decision was made to design a framework for formal model analysis, in which we could easily apply and compare the different methods for formal analysis on different models in order to consistently apply and test the methods. As a result of this decision, the second part of the research follows the approach as described in Figure 3.2.

The first step in the second part is to develop a conceptual design of the formal framework which enables the application of the methods. As a starting point for the design, the requirements for the framework are defined based on the exploratory analysis and the goal of consistently and flexibly applying the methods. The conceptual design defines the components of the framework, the information exchanged between them and their internal workings.

To test the design a prototype is implemented, which is used to analyse a selection of models. The selection was based on the available literature on model analysis. Several models are accepted as more or less standard test models for model analysis methods. The framework should enable reproduction of the results of the analyses performed by previous authors.
As a side effect, the process of explicitly defining how to perform model analysis and implementing the framework, generated insight into the workings of the methods. It resulted in a familiarity with the methods beyond that obtainable by a literature survey or the application of the methods using either existing software or writing code on a per model basis. The implementation phase functioned as a learning process.

As soon as the implemented framework was found suitable for the purpose of performing model analysis, it was used to evaluate the methods in more detail. The same selection of models used to test the framework was used to evaluate the methods; all of the models where analysed with the different available methods. This provided the knowledge needed to answers research question 4 regarding the advantages, requirements and limitations of the formal methods.

Relevant experts were involved in the development of the framework, its prototype and the evaluation of the methods. For the first of those, experts in the field of model analysis were used to validate the central ideas of the framework. For the second, experts were consulted regarding the implementation of the framework and the generated results.

![Figure 3.2: Second phase of the research approach](image-url)
Chapter 4

Preliminary Concepts and Techniques

The focus of this thesis lies on the analysis of the relation between structure and behaviour. However, to understand the investigated techniques in this research, two different views on System Dynamics models are necessary. The first of these is the graph theory view of System Dynamics models, in which the model is seen as a collection of connected vertices (the variables) and edges (the links between the variables). The second of these is the equation-based view of the model, in which the model is seen as a system of coupled ordinary differential equations. In this chapter these preliminary concepts and techniques are dealt with before proceeding to the deeper methods. The aim is to provide an overview of existing concepts and showing the relationships between them.

The graph theory view is used in structural analysis to investigate properties of the structure of the model without taking behaviour into account. Structural analysis is a valuable tool for its own purposes and a necessary tool for understanding the relation between structure and behaviour. The algorithms for detecting the loops in a system based on a graph representation are used in the analysis of the link between structure and behaviour. Structural analysis is required to provide the analyst with a subset of the loops present in a model.

In the first section I describe the basic concepts used in structural analysis. The following two paragraphs of the section respectively go into finding the loops within a model and an defining an independent subset of the total loopset.

The equation based view on System Dynamics model is more well known; it is also referred to as the formal structure of a model (Kampmann, 1996a). Each level in the

---

1For instance, it can be used to find minimal equation sets for model calibration (Oliva, 2003, 2004)
system diagram corresponds to a state variable in the coupled system. For these state variables, the rate of change is determined by the summation of the incoming and outgoing flows. The auxiliary variables form the non-state equations of the system of equations.

The second section of this chapter contains a discussion of the basic concepts and techniques related to this view of the model. First, the basic form of the model in this notation is described. Second, the concept of linearisation is described, as well as the concept of eigenvalues and how they relate to the behaviour of a linear system.

4.1 A Graph Theory View of System Dynamics Models

In order to understand the relationship between structure and behaviour, the analyst first needs specific knowledge of that structure. For instance, before the dominance of a loop can be assessed, the loop has to be found in the model. In small models, this is often trivial, but in models of realistic size a more formal view of the model is needed to find those properties of structure the analyst needs for the analysis techniques. The graph theory view on a System Dynamics model is a formal representation of the relations in a model, which enables the implementation of algorithms that analyse the structure of the model.

Two publications form the basis of the algorithms and methods presented in the section on the graph theory view, the paper by Oliva (2004) on model analysis through graph theory and the paper by Kampmann (1996a) on Loop eigenvalue elasticity.

4.1.1 Central Concepts

As said above, an SD model can be seen as a directed graph\(^2\), or digraph. To make the translation from model to graph, every variable in the model, be it a rate, an auxiliary, a parameter or a state is represented as a vertex in the graph. The relations between the variables define the edges in the graph; if a variable is directly used in the calculation of another, there exists an edge between the two\(^3\). A SD model is always a directed graph, which means that every edge has a source and a destination. Since the existence of an edge between two variables determines whether the source influences the destination or not, parallel edges between vertices would be meaningless. In addition, self-loops are impossible, since every loop

\(^2\)A directed graph (or digraph) is a pair \((V, E)\) of disjoint sets (of vertices and edges) together with two maps init: \(E \rightarrow V\) and ter: \(E \rightarrow V\) assigning to every edge \(e\) an initial vertex and a terminal vertex\(^3\)(Diestel, 2005, pg. 29)

\(^3\)A causal diagram can also be seen as a digraph, with signed edges.
must contain at least one state variable and one non-state variable. The resulting graph can be represented as an adjacency matrix, which shows the connections between variables; \( A_{ij} = 1 \) if vertex \( i \) has an edge leading to vertex \( j \). If there exists an edge from \( i \) to \( j \), vertex \( j \) is said to be a successor of \( i \), while \( i \) is said to be a predecessor of \( j \). Relative to a directed edge, these vertices are referred to as the destination and the source.

To illustrate the graph perspective, we will use a very small model and its digraph and its matrix representations. See Figure 4.1.

![Diagram](a) Diagram

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>B</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>C</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>D</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>E</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>F</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

(b) Adjacency Matrix

Figure 4.1: Representations of a graph

The adjacency matrix of a graph can be used to construct a reachability matrix. Whereas the adjacency matrix shows the direct connections between the vertices in a graph, the reachability matrix shows whether there exists an \( n \)-step path between variables. If, in a reachability matrix \( R_{ij} > 0 \), then there exists a path leading from vertex \( i \) to \( j \). Appendix B.1 contains an extended discussion on obtaining the reachability matrix.

There are several strategies available for partitioning the model based on the discussed graph representation. First of all, we can establish the concept of a partition level. The first partition level of a model is that set of variables that has no successors out of its own predecessor set in \( R \). They influence nothing beyond themselves. Typically, these are the outcome variables of a model. The second partition level in the model is found by removing the variables in the first partition level and reiterating the same algorithm.

The other form of partitioning is into cycle partitions. Each cycle partition is a set of vertices that have the same set of predecessors and successors in \( R \). All members of a cycle partition are always in the same level and can be traced by an single

---

4Trivially, since only rate variables can influence states, the shortest possible loop in a model is one including just one state and one of its adjacent rates.

5Partitioning means dividing the model into different sections based on its structure.

6This is strongly related to Eberlein’s concept of exact simplifications (Eberlein, 1989)
feedback loop (Oliva, 2004). The partitions are maximally connected subgraphs of the model, i.e. the reachability matrix of the subgraph is filled with ones. There are no loops across cycle partitions. We will use these properties later on in the discussion of loop detection. Appendix B.2 contains an extended discussion on obtaining the reachability matrix.

### 4.1.2 The Loops In The Model

Within System Dynamics the traditional ‘unit of explanation’ is the feedback loop. The loops and the signs these loops have are used in establishing intuitive understanding of the behaviour of a model (Senge, 1990; Sterman, 2000; Richardson and Pugh, 1981). Consequently, when relating structure to behaviour, the methods for model behavioural analysis focus on the loops in the system. Structural analysis is used to detect the loops and define specific subsets of all loops in the model.

A loop is defined as a cycle (Diestel, 2005) in the model which does not include repeated vertices. Traditionally, loops are defined by the edges included in them. However, since there are no parallel edges in SD models, a loop can also be defined by the vertices it includes. To avoid confusion, note that the term “loop” within the context of System Dynamics is used differently than in graph theory; in graph theory a loop refers to an edge in a directed graph where the source of the edge is the same vertex as its destination (Diestel, 2005).

To represent loops, usually the edges or vertices in a loop are noted in the order encountered when traversing the cycle. For instance, if loop $P$ is traced by vertices $A$, $B$ and $C$, than the loop is denoted as $P = A, B, C$. The complete loop set is the set of all possible loops in the model. A set of loops is often represented as a Directed Cycle Matrix. For a set of loops consisting $m$ of loops, which include $n$ edges between them, the DCM would be an $n \times m$ matrix. Each row in this matrix is associated with an edge, each column with a loop. If an edge $e$ (associated with row $i$, is) included in loop $p$ (associated with column $j$) than for a DCM $D$, $D_{ij} = 1$. If the edge is not included in the loop, the corresponding element in the matrix is 0.

**Reduction of the loop set**

In the relatively simple example of the Lotka-Volterra model (Appendix B), there are no less than 8 different loops, more if the same model is written in a different form. The World3 model contains 5 state variables and 81 different loops (Kampmann, 1996a). The number of loops in a SD model tends to grow very fast with model size. To use the loop set in any form of analysis, we need to reduce it to a more manageable form that is still considered to be descriptive of the model.

The form of reduction introduced by Kampmann is to define an independent...
4.1. A GRAPH THEORY VIEW OF SYSTEM DYNAMICS MODELS

Loop set (ILS). Say we have model graph $G$ with edges $\{e_1, e_2, \ldots, e_n\}$ with the complete loopset $C = \{S_1, S_2, \ldots, S_m\}$. Each cycle $S$ is defined by its so-called incidence vector $v = \{v_1, v_2, \ldots, v_n\}$ where $v_i = 1$ if $e_i \in S$ and $v_i = 0$ if $e_i \notin S$. Building a matrix from the incidence vectors of all cycles gives the directed cycle matrix. An independent loopset consists of a maximal set of loops with linearly independent incidence vectors. The set is not unique.

Three properties make the ILS a very useful concept for further analysis of the model. Firstly, the number of independent loops grows only linearly with model size (Kampmann, 1996a), while an independent loop set will always include every edge in the model that is part of a loop. Consequently, it is considered descriptive of the model and is manageable size, relative to the size of the model. Secondly, the linear independence of the loops makes it possible to use an ILS to determine individual properties of the loops within the set. Thirdly, it is a decent starting point for finding the edges in the model that are only related to a single loop, or to find how many loops are associated with a given edge.

The algorithm Kampmann uses to obtain an independent loop set starts with an arbitrarily chosen set of loops. It accepts new edges into the set by picking an edge outside of the set and adding the shortest path back to the set to the loop set (Kampmann, 1996a). The main disadvantage of this algorithm is the fact that it is path dependent; a different starting set results in a different final ILS.

Oliva strives to obtain a comparable independent loop set using a method without the disadvantages of the Kampmann approach. That is, the set should a) not be path dependent and b) unique for a given model. The method starts with the set consisting of geodetic cycles in a cycle partition obtained by the algorithm described above. The basis of the method is selecting the shortest loop that introduces the minimum amount (larger than zero, of course) of new edges into the loop set. This independent loop set is referred to as the shortest independent loop set (SILS). Given the use and existence of this algorithm, it would be redundant to further discuss the algorithm to obtain an independent loop set described by Kampmann (1996a). The algorithm used to obtain this set of loops is explained in further detail in Section 4.1.2.

The eigenvalue analysis proposed by Kampmann (Kampmann, 1996b,a) relies heavily on the use of an independent loop set. The SILS generates a set that is unique for a model and focuses on the shorter - and consequently more intuitive - loops in the model. The addition of the SILS aims to bring more consistency to the application of Kampmann’s method. For a further discussion of this approach, see Section 5.3.
Loop Detection

Given the formal graph representation, several algorithms to detect the loops present in a model are at the disposal of the analyst. The first, naive one, gives back all loops in a model; a variant of this algorithm is also used by Kampmann (Kampmann, 1996a). The second is a description of the algorithm used by Oliva (2004) to find the SiLS.

The algorithm used to find all loops in the system is based on recursion; it calls itself. It remembers the path that it has walked up to the current variable, checks if it has been on the current variable before and, if not, calls itself on the successors of the current variable. The algorithm branches if a vertex has more than one successor. The MatLab code for the algorithm can be found in Appendix B.3.

An example of the algorithm can be found in (Figure 4.2), the resulting DCM in Table 4.1. The path it has walked so far is empty, so it calls itself on the successors of A. The next vertex, B, has not been visited before so it is added to the path and the algorithm moves to C. At C, the recursion branches to A and D. The descendent that moved to A detects that the vertex has already been visited, hence the path walked so far \( P_1 = \{A, B, C, A\} \) is added to the loop collection and that branch of the algorithm is stopped. The algorithm then continues at the second successor of C, namely D, and from there on to E and F. Once it arrives at D for the second time, the next loop is reported; \( P_2 = \{A, B, C, D, E, F, D\} \). Trivially, not all vertices belong in the actual loop, so \( P_1 = \{A, B, C\} \) and \( P_2 = \{D, E, F\} \).

The algorithm has problems with one way causation though. In the above example for instance, if the algorithm would have been started on either vertex D, E or F, it would have never stepped into the upper part of the graph. Kampmann (1996a) remedies this by starting the algorithm on every vertex of the graph and removing the edges connected to the vertex after each iteration of the loop detection algorithm. Since a cycle partition is maximally connected, another solution is to start the algorithm on each cycle partition separately. The algorithm detects all loops within them and there are no loops across different partitions. In our example, we would isolate the upper and lower cycle partitions \( \{A, B, C\} \) and \( \{D, E, F\} \).
Figure 4.2: Recursions of the Kampmann loop detection algorithm. The grey nodes indicate the positions of the algorithm at the indicated recursion depth. The edges are labelled 1 through 7, the vertices A through F.
An alternative approach is introduced by Oliva (Oliva 2004). He first establishes the distance matrix $D$ of a cycle partition by:

$$B = I + A$$

$$D = C + \sum_{i=2}^{[C]-1} (B^i - B^{i-1})$$

All algebra in Equation 4.1 is non-Boolean, except for the powers of B. The distance matrix shows the shortest path between all variables. Taking the second cycle partition from the extended Lotka-Volterra (Appendix A) model get the distance matrix as in Figure 4.3.

![Distance matrix of the second cycle partition](image)

A loop can be seen as the combination of two paths; that from vertex $i$ to $j$ and that from $j$ to $i$. Hence, the length of all loops can be found by adding the transpose of the lower left triangle of the distance matrix to its upper right triangle. This results in the length matrix (Figure 4.4).

The actual loop detection routine uses the $i \rightarrow j$ and the $j \rightarrow i$ paths. If the $i \rightarrow j$ path is of length $n$, it looks for the set of successors of $i$ at distance 1 and the set of predecessors of $j$ at distance $n - 1$. The first vertex that is found in both these sets, is added to the loop. The algorithm then iterates for the successors to the last element added to the loop and the predecessors of $j$ at distance $n - 2$. This continues until all vertices in $i \rightarrow j$ are found and does the same thing for the $j \rightarrow i$ path.

I will illustrate the algorithm on a small Lotka-Volterra model. The aim is to find the loop between prey and predator. This is known to be prey $\rightarrow$ predator births $\rightarrow$ predator $\rightarrow$ encounter kills $\rightarrow$ prey deaths $\rightarrow$ prey.
Figure 4.4: Length matrix of the second cycle partition

<table>
<thead>
<tr>
<th></th>
<th>prey births</th>
<th>prey</th>
<th>prey deaths</th>
<th>encounter kills</th>
<th>predator</th>
<th>predator births</th>
<th>predator deaths</th>
</tr>
</thead>
<tbody>
<tr>
<td>prey births</td>
<td>0</td>
<td>2</td>
<td>4</td>
<td>5</td>
<td>7</td>
<td>7</td>
<td>9</td>
</tr>
<tr>
<td>prey</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>5</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>prey deaths</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>5</td>
<td>7</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>encounter kills</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>7</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>predator</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>predator births</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>predator deaths</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Figure 4.5: Digraph of the Lotka-Volterra model
The algorithm would take 4 steps to find the loop involving prey and predator. We initialise the algorithm by adding the starting node prey to the loop and setting the path length from prey to predator, \( n \), to 2. We then look for the set of successors of the last element added to the loop and the set predecessors of distance \( n - 1 \) away from predator. The vertex, or one of the vertices\(^7\), that is in both these sets is then added to the loop. In this case this is predator births, making the loop path so far \( \text{prey} \rightarrow \text{predator births} \) (Figure 4.6).

![Figure 4.6: First step of the Oliva loop track](image)

At the following step the algorithm arrives at predator, which is then added to the loop. After this, we switch from using the path from prey to predator, to the path from predator to prey. The path length \( n \) between these two variables is 3. The following step adds encounterkills to the loop, which is a direct successor of predator and a predecessor of at distance \( n-1 \) from prey. This makes the loop path \( \text{prey} \rightarrow \text{predator births}, \rightarrow \text{predator} \rightarrow \text{encounterkills} \) (Figure 4.7).

![Figure 4.7: Step 3 in the Oliva loop track routine](image)

The fourth and final step compares the predecessors of prey at distance \( n-2 \) and the successors of encounterkills. See Figure 4.8.

The loop track routine can be called on all loops present in the \( L \) matrix. Starting with the shortest loops, it tracks the first, removes all vertex pairs involved in the loop and tracks the following. Circuits with repeating nodes - such as the

\(^7\)It is at this step that the algorithm fails to distinguish between shortest paths of equal length.
4.1. A GRAPH THEORY VIEW OF SYSTEM DYNAMICS MODELS

4.1.3 Causal Paths

Within model behavioural analysis, the concept of causal paths is used in eigenvalue elasticity analysis (Kampmann, 1996a; Saleh, 2002; Güneralp, 2005; Abdel-Gawad et al., 2005) and in the pathway participation metric based method developed by Mojtahedzadeh et al. (2004). Both methods need to trace the paths in the model through which specific variables influence or are influenced by other variables in the model. Consequently, another use of structural analysis important to this research is the ability to trace and enumerate causal paths in the model.

Since a System Dynamics Model is always a directed graph, the paths in the model are always directed. The definition of a path is similar to that of a path in graph theory, where

A path is a non-empty graph $P = (V, E)$ of the form $V = \{x_0, x_1, \ldots, x_k\}$, $E = \{x_0x_1, x_1x_2, \ldots, x_{k-1}x_k\}$ where the $x_i$ are all distinct. The vertices $x_0$ and $x_k$ are linked by $P$ and are called its ends; […] (Diestel, 2005)

The description of a path can be in terms of edges or vertices, both are equivalent. However, in System Dynamics the search for paths is often performed ‘in reverse’. In that case, both the search and the description of the path follow the ‘influenced by’, not the ‘influences’ direction.

For several types of analysis, the analyst traces paths in a model to find those paths to variables that meet specific criteria. Example criteria are; the variable already occurs in the tree, or the variable has empirical data defined for it, or the variable is a state. The search algorithm often stops following a path when it meets a variable that meets the criterium; this is referred to as clipping.
If the analysis returns more than one path as its result and these paths share a common begin vertex\textsuperscript{8}, then these are often shown together in a tree-like structure (Figure 4.9). Paths starting at the same vertex are drawn together until they no longer share vertices and edges. This gives the appearance of a tree, but it is not a tree in the same sense as used in graph theory.

Causal paths also play a role in the conceptualisation\textsuperscript{9} and calibration. In the latter, the modeller uses structural analysis and the available data for the system to find the smallest calibration problems (Oliva, 2003, 2004).

This type of analysis will be demonstrated by showing how it is used in calibration. The development of strategies for calibration is the main subject in Oliva’s paper on model analysis through graph theory (Oliva, 2004). Calibration is the process of changing model parameters to have its behaviour fit historical data. Automated model calibration is a powerful tool for fitting a SD model to historical data and attempting to reject the model; if no valid combination of parameters can generate realistic behaviour, rejection of the structure of the model as it now stands seems in order. The dilemma is that the more parameters can be used to calibrate a model, the more likely a fit to historical data, but the less powerful it is as a test. With too many adjustable parameters it is possible to fit almost any model structure to any behaviour, so conclusions cannot be made concerning the validity of the model structure. Hence, a rigid approach to calibration aimed maximising its power as a test is a valuable tool. Structural analysis is used in developing such an approach (Oliva, 2003).

Based on the available data off a modelled system and the structure of the model, a strategy can be devised for obtaining a minimal set of equations for calibration. By starting at one of the variables for which a data series is available, we do a reverse breadth first search along the causal paths leading to this variable and clipping the paths whenever we encounter a variable for which data is available. By this method we end up with a minimal set of equations for calibration (Oliva, 2004).

For instance, following the predecessors (the search is performed in the "is influenced by" direction) for predator in the Lotka-Volterra example and clipping the resulting paths at variables it has previously encountered, the result are the causal paths in Figure 4.9. To obtain a minimal equation set for calibration, the paths are clipped at the variables for which data is available and the resulting part of the model is calibrated.

\textsuperscript{8}Or end vertex, if the tree is drawn in reverse.

\textsuperscript{9}For instance, the list extension method describes a collection of causal paths where the vertices are ordered according to the length of their shortest path to one or more variables of interest in the model list.
4.2. **EQUATION-BASED VIEW OF SYSTEM DYNAMICS MODELS**

(a) Causal paths leading to predator, clipped at vertices that are already in a path

(b) Paths leading to predator, clipped at the vertices that have available data or that are already in one of the paths.

Figure 4.9: Tree-like view of the causal paths leading to predator. Data is available for the underlined vertices.

### 4.2 Equation-based view of System Dynamics Models

The general view of a System Dynamics Model is that of a System Diagram, which is a graphical representation of the states, rates, auxiliary variables in the model. The diagrams are central to the conceptualisation of System Dynamics Models (Saleh, 2000). However, for quantified models, a coupled system of differential equations always underlies the diagrams. It is this system that is numerically solved when a simulation run is performed. For simplicity, the model is assumed to be homogeneous.

#### 4.2.1 Central Concepts

The formal structure of a System Dynamics model can be represented as a coupled system of differential equations, such that

\[
\begin{align*}
\dot{x}(t) &= f(x(t), y(t)) \\
y(t) &= g(x(t), y(t))
\end{align*}
\]

(4.2)

where \( \dot{x} \) is the vector of the net rate of change of the states in the system, \( x \) the vector of states and \( y \) the vector of auxiliaries. Within this form, the behaviour of the model can be seen as the movement of a vector through state space.
Any variable that is not a state variable is considered an auxiliary in this representation. Consequently, the rates in the model are seen as auxiliaries, so the net rates of change of the states are determined by the summation of their incoming and outgoing rates, according to the sign of the rate relative to the level\(^\text{10}\). Although the net rate of change of a state variable can be a direct function of another state variable in this form, this is contradictory to the principles as mentioned by Saleh (2000), since he explicitly states that a state can only be influenced by a rate (Forrester, 1976; Saleh, 2000). Following the mentioned principles, this also means that, within the graph perspective, no edge can lead into a state variable, unless its source is a rate.

Within this form, a rate is seen as part of the \(y\) vector auxiliaries in this form, which means that this form makes no distinction between rates and auxiliaries. A mathematically equivalent model that does not obey the principles as described by Saleh (2000) can be obtained through reordering the equations. It is by definition always possible to simplify the form described above to the reduced form of the model in which all auxiliary variables are eliminated, allowing the model to be rewritten as

\[
\dot{x} = F(x) \quad \text{(4.3)}
\]

The reduced form only shows the state variables in the system. Although it is mathematically equal to the full form of the model, the elimination of the auxiliaries makes the model less readable for both the analyst and the client.

### 4.2.2 Linearisation

A linearisation is the simplification of a model to a linear model. The analysis of a linearised version of a model is considered a powerful technique in its own right, for purposes such as stability analysis (Thissen, 1978). It forms the basis of eigenvalue elasticity analysis, where for each point of analysis the model is linearised and then analysed.

The simplification always takes place around a operating point \(g(\bar{x}, \bar{y})\), defined by the analyst. The use of the linearisation is restricted to a working region around the operating point. For this point, the reduced form of the model can be linearised to the following equations

\[
\dot{x} = G\dot{x} \quad \text{(4.4)}
\]

\[
(4.5)
\]

\(^{10}\text{An incoming rate has a positive sign, an outgoing rate negative. The same rate can be an incoming rate for one level and an outgoing rate for the next. Consequently, the sign of a rate is determined relative to a level.}\)
where G is the so-called gain matrix or system matrix. The linearisation can be extended to a version of the system showing the auxiliary variables; the so-called descriptor form. In this, y is the auxiliary vector, x the state vector.

\[
\dot{x} = A\tilde{x} + B\tilde{y} \quad (4.6)
\]

\[
\tilde{y} = C\tilde{x} + D\tilde{y}
\]

The tilde variables represent deviations from the operating point. In addition, the gain matrix can be expressed in terms of the other four matrices.

\[
G = A + B(I - D)^{-1}C \quad (4.7)
\]

Each element in these matrices is related to the gain of a specific edge, linking the graph perspective to the linearisation. The gain of an edge, \(g(e)\), leading from variable b to a is defined as

\[
g(e) = \frac{\partial a}{\partial b}\quad (4.8)
\]

The source and destination of the edge determine which element in which matrix it is equivalent to. This forms the link between the Systems Diagram representation of the model, the graph representation and the linearised version of the model. An edge leading from an auxiliary to a state would be included in the D matrix. The value of the matrix element is equal to the gain of the edge. Also, the A matrix is always empty, since states can only be influenced by rates Saleh (2000). The B matrix contains just the values 1 and −1 for each rate to state edge. All translations from edges to elements in the matrices are discussed in Table 4.2

The parameters of the model are only implicitly present in this linearisation. Although it is possible to include this, the relation between a parameter and a variable is not associated with any particular element in the discussed matrices.

### 4.2.3 The Relation between Eigenvalues and Behaviour

For any linear system \(\dot{x} = Gx\), the eigenvalues of square matrix G describe the behaviour of this system. That is, any solution of this linear system can be expressed as a linear combination of its eigenvectors and eigenvalues, where the initial conditions determine which linear combination this is. Eigenvalues are also referred to as behaviour modes.

In other words, the solution of a linear nth-order system can be expressed in terms of its fundamental matrix\(^{11}\). That is, for any system

\[
\dot{x} = Gx \quad (4.9)
\]

\(^{11}\)This form of the solution assumes unique eigenvalues.
Table 4.2: Table of the relation between the source and destination of an edge and the placement of the edge in the descriptor form of the linearised model.

<table>
<thead>
<tr>
<th>Source</th>
<th>Destination</th>
<th>Matrix</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>State</td>
<td>State</td>
<td>A</td>
<td>Does not occur.</td>
</tr>
<tr>
<td>State</td>
<td>Rate</td>
<td>C</td>
<td>Value of partial derivative of the rate to the state at the operating point.</td>
</tr>
<tr>
<td>State</td>
<td>Auxiliary</td>
<td>C</td>
<td>Value of partial derivative of the auxiliary to the state at the operating point.</td>
</tr>
<tr>
<td>Rate</td>
<td>State</td>
<td>B</td>
<td>Value is either 1 or −1, depending on whether the rate is incoming or outgoing relative to the state</td>
</tr>
<tr>
<td>Rate</td>
<td>Rate</td>
<td>D</td>
<td>Hardly ever occurs and conflicts with principles as stated by Saleh (2000)</td>
</tr>
<tr>
<td>Rate</td>
<td>Auxiliary</td>
<td>D</td>
<td>Value of partial derivative of the auxiliary to the rate at the operating point.</td>
</tr>
<tr>
<td>Auxiliary</td>
<td>State</td>
<td>B</td>
<td>Does not occur.</td>
</tr>
<tr>
<td>Auxiliary</td>
<td>Rate</td>
<td>D</td>
<td>Value of partial derivative of the rate to the auxiliary at the operating point.</td>
</tr>
<tr>
<td>Auxiliary</td>
<td>Auxiliary</td>
<td>D</td>
<td>Value of partial derivative of the auxiliary to the auxiliary at the operating point.</td>
</tr>
</tbody>
</table>

\[ T = \begin{pmatrix} r_{1,1} & \cdots & r_{1,n} \\ \vdots & \ddots & \vdots \\ r_{n,1} & \cdots & r_{n,n} \end{pmatrix} \] \hspace{1cm} (4.10)

The linear system will have the *fundamental matrix* \( \Psi \) (Boyce and DiPrima, 1996).

\[ \Psi(t) = T \begin{pmatrix} e^{\lambda_1 t} \\ \vdots \\ e^{\lambda_n t} \end{pmatrix} \] \hspace{1cm} (4.11)

And, consequently, the solution

\[ x = \Psi(t)c \] \hspace{1cm} (4.12)

where

\[ x_0 = \Psi(t_0)c \] \hspace{1cm} (4.13)

\[ c = \Psi(t_0)^{-1}x_0 \] \hspace{1cm} (4.14)
The behaviour associated with a particular eigenvalue is as displayed in Figure 4.10. For a complex conjugate eigenvalue pair, the dampening ratio (the ratio of two consecutive peaks in the oscillating system), is represented by \( \cos \theta \). The natural frequency of the eigenvalue is its absolute value \( |\lambda| \); the frequency associated with a purely imaginary eigenvalue with the same absolute value. The dampened frequency \( f_d \), influencing the time between two consecutive peaks, is the absolute value of the imaginary component and the associated period is \( 2\pi / \text{Im}(\lambda) \). The time constant, which represents the speed of dampening or growth, for the exponential envelope is \( 1/\text{Re}(\lambda) \).

Figure 4.10: Eigenvalue and associated behaviour (after Forrester (1982)). The eigenvalue is represented by \( \lambda \), while the \( i \) axis is the imaginary part of the eigenvalue, the \( r \) axis the real part. The angle the eigenvalue makes with the real axis is \( \theta \) and \( f_d \) is the dampened frequency. Note that the real part of the eigenvalue is equal to \( 1/\tau \).
Chapter 5

Understanding Behaviour

This chapter will introduce a selection of recently developed techniques for model behavioural analysis. The most prominent concept in these methods for understanding model behaviour is dominant structure; that part of structure that is most responsible for current model behaviour. This concept, central to model analysis, has been the focus of considerable research effort.

First, I introduce the behavioural approach designed by Ford (1999). Second, the Pathway Participation Method developed by Mojtabazadeh et al. (2004) is discussed. Third, Eigenvalue Elasticity Analysis is explored, focussing on the work of Forrester (1982), Kampmann (1996a) and Güneralp (2005).

5.1 Behavioural approach

5.1.1 Description

Ford introduces a behavioural approach to loop dominance analysis. His analysis consists of a sequence of steps aimed at finding which loops dominate the behaviour of a selected variable of interest. It explicitly aims to be behavioural, meaning that is does not rely on the equations in the model.

The first concept this method introduces is the behaviour pattern, a descriptor of the behaviour displayed by the variable of interest. Ford identifies three Atomic Behaviour Patterns (ABP) based on the rate of change of the variable of interest. These are defined as follows:

- linear atomic behaviour pattern $\frac{\partial(|\partial x/\partial t|)}{\partial t} = 0$
- exponential atomic behaviour pattern $\frac{\partial(|\partial x/\partial t|)}{\partial t} > 0$
- logarithmic atomic behaviour pattern $\frac{\partial(|\partial x/\partial t|)}{\partial t} < 0$
Consider an interval during which the variable of interest displays only one atomic behaviour pattern. Within this interval, loop dominance can be analysed by eliminating a loop and checking if the atomic behaviour pattern changes. That is, if the elimination of a loop causes a variable of interest to switch between ABPs, that loop is considered dominant for the interval in which it was eliminated. Counterintuitively, this implies that multiple loops can dominate at the same time; dominance is not limited to one loop and can be shared.

Furthermore, multiple loops can dominate so strongly at the same time that taking out one loop does not cause the behaviour pattern to change, only if all loops are taken out does the pattern change. If one loop is eliminated, the other loops continue to generate a behaviour that is consistent with the original behaviour pattern. These structures are referred to as shadow feedback structures.

Ford (1999) proposes a rigorous method for finding dominant structure, including shadow feedback structures. The variant presented here has been slightly modified to avoid some unnecessary iteration, but retains the essence of Ford's procedure.

1. Identify a variable of interest.
2. Identify the loops that influence the variable of interest.
3. Identify or build control variables that can switch off loops independently.
4. Select a time interval over which the atomic behaviour pattern stays the same.
5. Select a candidate loop and simulate the time interval with the loop eliminated.
6. If the loop is not dominant on its own, check for combinations with other loops to identify shadow feedback loops by eliminating other loops in combination with the candidate loop. If this combination results in a change of the atomic behaviour pattern either relative to the model with the other loops eliminated, or the original model, we have found a shadow feedback structure.
7. Repeat steps five and six for the next loop.
8. Repeat steps five through seven for the next time interval.

I will illustrate Ford's method by applying it to the Lotka-Volterra model. The selected variable of interest is the predator population in our model of the nature reserve, illustrated in Figure 5.1.

The second step is to identify all loops that influence predator. These include all seven loops as indicated in Figure 5.2 and Table 5.1. Note that although loops 4
Figure 5.1: Behaviour of predator population (shaded areas denote the intervals of ABPs)

through 7 do not include predator directly, they do influence predator. Also, each loop can be controlled individually; by deactivating the right edge each loop can be eliminated without influencing other loops.
5.1. BEHAVIOURAL APPROACH

Table 5.1: Loops in LV

<table>
<thead>
<tr>
<th>Loop Name</th>
<th>#</th>
<th>Variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>Predator births</td>
<td>$L_1$</td>
<td>{ predator, predator birth }</td>
</tr>
<tr>
<td>Predator deaths</td>
<td>$L_2$</td>
<td>{ predator, predator deaths }</td>
</tr>
<tr>
<td>Food</td>
<td>$L_3$</td>
<td>{ predator, encounter kills, total prey deaths, prey, predator births }</td>
</tr>
<tr>
<td>Kills</td>
<td>$L_4$</td>
<td>{ prey, encounter kills, total prey deaths }</td>
</tr>
<tr>
<td>Prey deaths</td>
<td>$L_5$</td>
<td>{ prey, standard deaths, total prey deaths }</td>
</tr>
<tr>
<td>Starvation</td>
<td>$L_6$</td>
<td>{ prey, starvation deaths, total prey deaths }</td>
</tr>
<tr>
<td>Prey births</td>
<td>$L_7$</td>
<td>{ prey, prey birth }</td>
</tr>
</tbody>
</table>

Figure 5.2: Lotka-Volterra model

The following step is to modify the equations so that the loops can be eliminated one by one (Table 5.2). A variable with subscript $s$ retains its value at the beginning of the selected time interval. The time interval over which we select to demonstrate the technique is the second ABP interval (Figure 5.1) for time $t \approx 3$ until $t \approx 14$. This results in the changed behaviours seen in Figure 5.3.

Of the seven loops, only $L_3$ and $L_2$ cause a clear and sudden change in the ABP. Their elimination results in the immediate shift from the logarithmic ABP to the exponential ABP. $L_5$ and $L_7$ do not cause the ABP to change at all during the time interval, but $L_1$, $L_4$, and $L_6$ cause the timing of the shift between ABPs to change. We consider $L_3$ and $L_2$ to share dominance in this phase. Note that both loops
Table 5.2: Eliminating loops in the Lotka-Volterra example

<table>
<thead>
<tr>
<th>Loop</th>
<th>Modified Variable</th>
<th>Old Equation</th>
<th>New Equation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$L_1$</td>
<td>Predator Birth</td>
<td>$PBF \cdot \text{prey} \cdot \text{predator}$</td>
<td>$PBF \cdot \text{prey} \cdot \text{predator}_{s}$</td>
</tr>
<tr>
<td>$L_2$</td>
<td>Predator Deaths</td>
<td>$PDR \cdot \text{predator}$</td>
<td>$PDR \cdot \text{predator}_{s}$</td>
</tr>
<tr>
<td>$L_3$</td>
<td>Predator Birth</td>
<td>$PBF \cdot \text{prey} \cdot \text{predator}$</td>
<td>$PBF \cdot \text{prey}_{s} \cdot \text{predator}$</td>
</tr>
<tr>
<td>$L_4$</td>
<td>Encounter Kills</td>
<td>$KF \cdot \text{prey} \cdot \text{predator}$</td>
<td>$KF \cdot \text{prey}_{s} \cdot \text{predator}$</td>
</tr>
<tr>
<td>$L_5$</td>
<td>Standard Deaths</td>
<td>$SDR \cdot \text{prey}$</td>
<td>$SDR \cdot \text{prey}_{s}$</td>
</tr>
<tr>
<td>$L_6$</td>
<td>Starvation Deaths</td>
<td>$SF \cdot \text{prey}^2$</td>
<td>$SF \cdot \text{prey}_{s}^2$</td>
</tr>
<tr>
<td>$L_7$</td>
<td>Prey Birth</td>
<td>$BR \cdot \text{prey}$</td>
<td>$BR \cdot \text{prey}_{s}$</td>
</tr>
</tbody>
</table>

Figure 5.3: Changes in behaviour of predator after eliminating loops
5.1. BEHAVIOURAL APPROACH

include the variable of interest.

With regard to shadow loops, not even the combination of $L_1$, $L_4$, $L_5$ and $L_6$ changes the ABP as much as $L_5$ or $L_7$. However, if we eliminate $L_1$ in combination with $L_3$ or $L_2$, the ABP remains the same as in the original model.

We can interpret the results of this analysis as the decrease of the prey population, $L_3$, and the deaths of predators, $L_2$, functioning as limiters on the growth of predator population. Without these limiters, the number of predators would increase exponentially, so the balancing behaviour over this time interval can be attributed to $L_2$ and $L_3$. The analysis of shadow loops indicates that the exponential growth that would result in the absence of $L_3$ and $L_2$ can be attributed to $L_1$, the birth of predators.

5.1.2 Review

The technique has significant strengths. These are:

1. We can use this type of analysis on almost any model, with standard software. We do not need Jacobians, eigenvalues or linearisations to eliminate a loop. Standard packages such as Powersim and Vensim offer the functions needed to determine the ABP and a loop can be eliminated using an if-statement in a model equation.

2. The mechanisms employed are understandable and explainable. Taking out part of model structure and running the model again while that structure is inactive, is conceptually relatively simple.

3. The interpretation of outcomes is fairly straightforward. If the ABP changes, the eliminated loop apparently was responsible for determining the reference behaviour.

4. The selective nature of the process can be an advantage compared with other methods. If we wish to just look at the effect of one loop and not bother with the rest of the model, we can.

However, the method has some shortcomings. Firstly, the procedure could benefit by adopting concepts from structural analysis. The reduction of the loops in the model to an independent loop set (ILS) would add rigour to the analysis.

Secondly, the definitions of shadow feedback structure and dominance are tricky. As stated by Ford (1999).

Richardson describes feedback structures in complex systems as “a major source of puzzling behaviour and policy difficulties” Richardson (1991).
Forrester (1969) partially attributes this to the influences of multiple feedback loops on system behaviour and the difficulties in identifying the actual causes of behaviour. The identification of loop dominance when two or more loops independently generate similar behaviour can create or contribute to these challenges. The similar behaviour of multiple loops can cause an analyst to misdiagnose which loop dominates the variable of interest or to miss shifts in dominance among the loops by hiding the influence of some loops behind the influence of other loops. [...] This confounding behaviour and dominance can be characterised as the hiding of one loop in the shadow of another loop. Therefore, we refer to these loops as pairs of shadow feedback loops and this type of structure more generally as shadow feedback structures. (Ford, 1999, pg. 9)

In addition, according to step 6 in Ford’s proposed procedure,

Identify time intervals, each of which contains a single atomic behaviour pattern. Within each time interval:

a) If the atomic behaviour pattern in a time interval generated in step 5 is different than the reference pattern identified in step 2, the candidate feedback loop dominates the behaviour of the variable of interest under the system conditions during that time interval.

b) If the atomic behaviour pattern of the variable of interest in step 5 is the same as in the reference pattern identified in step 2, two conditions are possible: (1) the candidate feedback loop does not dominate or (2) the candidate feedback loop dominates, but has a shadow feedback loop. (Ford, 1999, pg. 11)

So to summarise, based on the first paragraph a pair of shadow loops generate a similar behaviour pattern. Both do this to such a degree that if one of the two is eliminated the other continues to generate similar behaviour. In addition, according to the sequence of steps in step 6, the candidate loop can only be part of a shadow structure if it does not dominate on its own.

However, in Ford’s comparison of his procedure to the Loop Eigenvalue Elasticity Analysis (Ford, 1999; Kampmann, 1996a) he finds a certain loop, $L_{14}$, to be dominant in his example and suspects $L_8$ of being a shadow loop. His test for shadow feedback structure amounts to:

$[\text{Eliminating } L_8...] \text{ reduces the length of the entire long wave, but retains the characteristic shapes of the Desired Production cycle found in}$

\footnote{He uses Kampmann as a guideline for selecting loops and eliminates those he considers likely candidates.}
### 5.1. BEHAVIOURAL APPROACH

Table 5.3: Unambiguous conditions for a set of shadow loops

<table>
<thead>
<tr>
<th>Loop A</th>
<th>Loop B</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Eliminated</td>
<td>Active</td>
<td>No change in ABP</td>
</tr>
<tr>
<td>Active</td>
<td>Eliminated</td>
<td>No change in ABP</td>
</tr>
<tr>
<td>Eliminated</td>
<td>Eliminated</td>
<td>Change</td>
</tr>
</tbody>
</table>

the reference behaviour, including the exponential Phase I followed by a logarithmic Phase II, which ends at the peak of the Desired Production cycle (step 5). Deactivating the capacity expansion loop did not change the atomic behaviour pattern. This indicates that either the capacity expansion loop does not dominate Phase II or it dominates with a shadow feedback loop (step 6b). We test whether the economic growth and capacity expansion loops form a shadow feedback loop pair by testing the dominance of the capacity expansion loop (L8) with the economic growth loop (L14) inactive. As described above, the behaviour of the Desired Production with loop L14 inactive and loop L8 active is linear during Phase II. Deactivating both loops also results in linear behaviour. (Ford, 1999, pg. 28)

So, we first thought that a shadow feedback structure resulted in the ABP not changing by removing each loop separately, but changing if we eliminate both loops. That is, two dominant loops generate the same behaviour, one taking over if the other is eliminated. However, following the reasoning of this example it appears to be the case that a shadow feedback loop is found by first taking out a dominant loop and than determining if other loops are dominant in the model that is missing the dominant loop.

In this research the last definition of a shadow loop will not be followed. The first definition, where, if there is a pair of shadow loops, the ABP only changes if both loops are eliminated will be used. In addition, to save time, the version of Ford’s behavioural approach that will be applied in this research will restrict the search for shadow loops to intervals where no single loop is dominant.

Thirdly, it is unclear how to take out loops. In our example for instance, we chose to take out $L_1$ by making a snapshot of the value $predator$ at the beginning of the time interval and using this to calculate further predator births. This approach sets the gain of this particular loop to 0, without influencing the contribution of the other loops to the variable of interest. However, this could also be achieved by setting $predator$ to its equilibrium value. The first option seems the most reasonable one, we continue the simulation without the loop, but from the state it was in at the beginning of the interval under inspection.
In contrast, the number of predators born could have been set to 0, but this would have eliminated \( L_3 \) as well as \( L_1 \), violating the independence constraint. Eliminating one loop in a particular fashion forces us to apply the same method of elimination to the rest of the loops. Otherwise, the lack of consistency in our method of taking out loops would determine which part of structure is considered dominant. The same point is brought forward by Ford in his own evaluation of the technique (Ford, 1999).

Fourthly, the restriction of the analysis to separate time intervals weakens its conclusions when dealing with an oscillating model. Our conclusions are only valid on a time interval displaying one ABP, causing it to ignore effects that span more than one interval. This means that Ford’s analysis will never find the dampening effect of \( L_6 \), the starvation loop.

And, finally, the step of finding shadow loops can be costly and time consuming. If we have \( n \) loops in our ILS, there are \( 2^n \) possible combinations of eliminated or active loops. This would be feasible in models of reasonable size, if we had support from an automated form of this procedure. For now, this is a labour intensive procedure of which the individual steps and concepts are not defined clearly enough to be implemented routinely in any form of software.

\section{5.2 Pathway Participation Metrics}

\subsection{5.2.1 Description}

One of the more recent techniques developed for detecting influential system structure is the pathway participation method. Developed by Mojtabahzadeh et al. (2004) as a mathematical tool for helping the modeller to understand the model, it has since been incorporated in a software package called DIGEST. The algorithm looks for influential structure by walking through the model and choosing its path based on individual contributions of variables to a selected variable of interest.

In the Pathway Participation Metric (PPM), the first step in finding influential structure would be to identify the \textit{causal structure} that determines the behaviour of the variable in the model. The causal paths are clipped at state variables and exogenous functions and do not include constants. To illustrate, in our Lotka-Volterra example the causal structure used for \textit{prey} can be seen in Figure 5.4.

The next step is to identify the \textit{pattern of behaviour} (not to be confused with Ford’s atomic behaviour pattern) of the variable of interest in the phase being analysed. A \textit{phase of behaviour} is a time interval during which the first and second time deriva-
5.2. PATHWAY PARTICIPATION METRICS

Figure 5.4: Causal pathways leading to prey for use in the Pathway Participation Metric.

dervatives of the variable of interest maintain the same sign. Based on this, there are seven patterns of behaviour, as seen in Figure 5.5.
Once the causal structure leading to a variable and the current mode are known, the question is which of the causal pathways - Mojtabazadeh et al. (2004) uses this term instead of causal path - contributes most to the current behaviour of the variable of interest. The first concept involves the Total Participation Metric (TPM) of the variable of interest, which is defined as follows:

\[ TPM = \frac{dx}{dx} \]  

(5.1)

The TPM is a measure of change in the flow of a state variable in response to a change in that variable. It is related to the behavioural pattern since \( \frac{dx}{dx} = \frac{dx}{dt} \). Given that,

\[ \dot{x} = f(x_1, x_2, \ldots, x_n) \]  

(5.2)
5.2. PATHWAY PARTICIPATION METRICS

Then, according to the chain rule,

\[
\frac{d\dot{x}}{dx} = \sum_{i=1}^{n} \frac{\partial f_k}{\partial x_i} \frac{d\dot{x}_i}{d\dot{x}_k}
\]  

(5.3)

The technique only works on an interval where \( x, \dot{x} \) and its partial derivatives are continuous. Now, getting back to the causal tree of prey, we see five different pathways leading to the variable of interest. Namely,

\[
w_1 = \{ \text{prey births, prey} \}
\]

\[
w_2 = \{ \text{prey deaths, encounter kills, prey} \}
\]

\[
w_3 = \{ \text{prey deaths, encounter kills, predator} \}
\]

\[
w_4 = \{ \text{prey deaths, standard deaths, prey} \}
\]

\[
w_5 = \{ \text{prey deaths, starvation deaths, prey} \}
\]

Using the chain rule we can isolate the contribution of a pathway leading to the variable of interest. For instance, the contribution of pathway \( w_3 \) is,

\[
c_3 = \frac{\partial \text{prey deaths}}{\partial \text{prey deaths}} \frac{\partial \text{encounter kills}}{\partial \text{encounter kills}} \frac{\partial \text{predator}}{\partial \text{predator}}
\]

Defining a pathway by the edges leading to variable \( k \) and again using the chain-rule, it is possible to calculate the contribution of a pathway to the TPM of the variable of interest. That is, if we define a pathway by

\[
w_n = \{ e_1, e_2, \ldots, e_m \}
\]  

(5.4)

and the gain of the pathway by

\[
g(w) = \prod_{i=1}^{m} g(e_i)
\]  

(5.5)

Then, its contribution to the TPM is

\[
c_w = g_w \frac{d\dot{x}_i}{d\dot{x}_k}
\]  

(5.6)

And,

\[
TPM = \sum_{i=1}^{n} c_i
\]  

(5.7)

The contribution of a pathway can be normalised according to the definition of the pathway participation metric:
\[ PPM(i, j) = \frac{\frac{\partial f^i}{\partial x^k} \dot{x}_k}{\sum_{i=1}^{n} \sum_{j=1}^{n} \left| \frac{\partial f^j}{\partial x^k} \dot{x}_k \right|} \] (5.8)

The most influential pathway is the one with the largest absolute value of the PPM and with the same sign as the TPM. The PPM provides a measure of the degree to which a particular pathway is responsible for a change in behaviour of a variable of interest. To find a dominant loop, the influential pathways are followed until arriving at a variable we have visited before. To illustrate the above, the Lotka-Volterra model is used. The behaviour of prey is as in Figure 5.6.

![Figure 5.6: prey](image)

Given the fact that \( w_1 \) and \( w_4 \) end in the variable of interest and have a path consisting of only linear relations, their contribution is constant (\( c_1 = 0.15, c_4 = 0.01 \)). The contributions of the other pathways are shown in Figure 5.7.

To select dominant structure, the algorithm chooses the dominant pathway and selects the state variable in which it ends as the next variable of interest. As stated, this continues until the algorithm either arrives at a state variable it has already visited - thereby establishing a dominant loop - or at an exogenous function. The dominant pathways of prey are shown in Figure 5.8.

The only pathway that leads to a different state variable is \( w_3 \). Hence, the other pathways immediately find a dominant loop. In order to find the dominant loop...
5.2. PATHWAY PARTICIPATION METRICS

Figure 5.7: Contributions of pathways leading to prey

Figure 5.8: Dominant pathways during different time intervals
in the remaining intervals, we will continue by repeating the above procedure with predator as the variable of interest. Predator has the causal tree, behaviour and contributions of pathways as seen in Figure 5.9.

Combining these two analyses leads to the following analysis for loop dominance for prey. In the first interval, from $t_0$ to $t \approx 1.8$ the growth of prey is brought to a halt by the increasing predator population, as is shown by the fact that the pathway leading to predator is considered dominant for prey. Predator itself has its own growth as the most influential pathway. From here on, the prey population starts to decrease exponentially, as the predator population continues to grow exponentially until $t \approx 3.25$. At that point, predator population starts to balance as deaths become more influential. This lasts until $t \approx 8$.

However, although we still see an exponential decline of prey, the algorithm chooses the loop containing prey births as dominant, $w_3$. That is, according to the PPM method as it is now, the loop responsible for the exponential decline of the population is the birth loop. The algorithm arrives at this conclusion due to the fact that a) the behaviour is exponential, so we must look for a positive PPM b) the largest positive PPM is the birth loop. The behaviour generated by the rest of the loops (which clearly have a stronger influence on prey, otherwise we wouldn't see exponential decline) is ignored. Apparently, under these conditions, the algorithm decides that exponential decay is caused primarily by a loop that generates exponential growth.

After the switch of behaviour pattern the method views the decline in kills by predators owing to the decrease of prey as most influential in the balancing decline of the prey population. This lasts until $t \approx 18$, when the decrease in predator becomes the primary cause for the decline in kills. The decrease in predator is attributed to the exponential growth loop of predator, owing to the reasoning described above. After a while, so very few kills are made that while the pathway to predator is still dominant for preys, predator is now being dominated by the loop including predator deaths. This remains the case until, finally, births take over the dominant role in the growth of prey.

5.2.2 Review

Although this method has some clear advantages, such as a relatively easy implementation, it suffers from two shortcomings.

Firstly, the analysis focusses on a single variable and says nothing on system as a whole. That is, we lose information by selecting one particular variable of interest. The dominance analysis is valid for that variable, not for the system. In comparison, eigenvalue elasticity analysis looks at the system as whole or a variable of interest.
5.2. PATHWAY PARTICIPATION METRICS

(a) Causal Tree

(b) Behavior

(c) Contributions

(d) Pathways

Figure 5.9: Dominant pathways for Predator
Secondly, the other choice made in the first steps of the algorithm further limits the method. By looking at separated time intervals, the method does not do well in the presence of oscillations and the dampening effects of certain loops. In the example for instance, although the starvation loop is sometimes considered to be dominant, the algorithm says nothing about its dampening role. No conclusions can be drawn about the causes of oscillation or the dampening thereof.

However, in my opinion, there is one flaw that renders the algorithm unusable. The fact that an infinitesimal change in the variable of interest causes the largest change in a pathway’s contribution does not appear to mean that the behaviour of the variable of interest is being dominated by that pathway. This is demonstrated by the dominance of loops that cause exponential growth while the variable of interest displays exponential decline. In other words, the algorithm draws invalid conclusions.

5.3 Eigenvalue Elasticity Analysis

Eigenvalue Elasticity Analysis (EEA) is a way of understanding model behaviour by analysing the response of an eigenvalue to the perturbation of a parameter or a loop gain. The method was developed by Nathan Forrester (1982) and expanded by Kampmann (1996b,a), Güneralp (2005) and Saleh (2002).

In a linear system, the eigenvalues associated with the system describe its behaviour. That is, frequency of oscillation, damping ratios and whether the system has stable equilibria can all be derived from the eigenvalues (See Boyce and DiPrima, 1996; Forrester, 1982). So, model structure can be related to the system eigenvalues, this can function as a form of model behavioural analysis.

Currently, we can distinguish two different forms of EEA, parameter and loop based. The parameter based variant of EEA calculates elasticities based on the influence of different parameters (PEEA), while the loop based variant does the same for model loops (LEEA). First, the parameter based variant will be explained and demonstrated. Next, we will address the structural variant. And, finally, we will address how to relate the eigenvalue elasticities to the behaviour of specific state variables.

5.3.1 Parameter based analysis

Parameter based eigenvalue elasticity analysis measures the response of system eigenvalues to minute changes in selected system parameters. Since the eigenvalues are assumed to describe behaviour, relating them to system parameters establishes the link between parameter values and system behaviour.
Forrester (1982) related system parameters to the system eigenvalues by defining the concept of *eigenvalue elasticity*. This elasticity describes the response of an eigenvalue to a small change in a system parameter. For a linear system $\dot{x} = Ax$, the eigenvalue elasticity of a eigenvalue $\lambda_i$ to a parameter $p_g$ is defined as the complex number

$$
\epsilon = \frac{\partial \lambda_i}{\partial p_g} \frac{p_g}{\lambda_i} 
$$

where

$$
\frac{\partial \lambda_i}{\partial p_g} = i \frac{\partial A}{\partial p_g} r_i 
$$

As stated in Equation 5.9, the calculation of the elasticity involves division of the change in the eigenvalue by the eigenvalue itself. Any component of the change parallel to the eigenvalue translates to a change in the absolute value of $\lambda$, while the perpendicular component measures the change in the angle that the eigenvalue makes with the real axis. Consequently, the real part of the elasticity measures the impact of the parameter change on the natural frequency of the eigenvalue, while the complex part measures the impact on the dampening ratio. That is, how fast the system oscillates and how fast the oscillation dies out, respectively.

An alternative way to calculate the elasticities is to define them as Kampmann (1996a) does, by measuring the relative change in the real and imaginary parts of the eigenvalues separately, according to

$$
\epsilon_1 = \frac{d \text{Re}(\lambda)}{dg} \frac{g}{|\lambda|}, \quad \epsilon_2 = \frac{d \text{Im}(\lambda)}{dg} \frac{g}{|\lambda|} 
$$

The *real elasticity*, $\epsilon_1$, gives the analyst information of the impact on the exponential envelope, while the *imaginary elasticity*, $\epsilon_2$, provides information on the frequency of oscillation. The overall elasticity is then defined as

$$
\epsilon_o = \sqrt{\epsilon_1^2 + \epsilon_2^2} 
$$

The modeller can adopt each measure as he sees fit, according to what he is most interested in. For instance, if a high dampening ratio is crucial to perceived performance of the model, the first measure gives more information.

This method is only applicable to linear systems. Hence, if the analyst wishes to apply this to a non-linear system, the analyst will have to linearise the system first and proceed from there. The approach will be illustrated by applying it to our example.

The model has the set of parameters as shown in Table 5.4. The same parameters as before will be used, but slightly different initial conditions. The behaviour
Figure 5.10: Behaviour and Eigenvalues of the Lotka-Volterra model. The left figure shows the behaviour of both populations according to time. The right figure shows the behaviour of one of the eigenvalues of the complex conjugate pair during the model run.

Table 5.4: Parameters in the LV example

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kill Factor</td>
<td>4e-3</td>
</tr>
<tr>
<td>Starvation Factor</td>
<td>1.2e-4</td>
</tr>
<tr>
<td>Standard Prey Death Rate</td>
<td>0.1</td>
</tr>
<tr>
<td>Prey Birth Rate</td>
<td>0.15</td>
</tr>
<tr>
<td>Predator Death Rate</td>
<td>0.2</td>
</tr>
<tr>
<td>Predator Birth Factor</td>
<td>5e-4</td>
</tr>
</tbody>
</table>

If we determine the elasticities by slightly modifying the parameters and calculating the resulting change in the eigenvalues of the linearised system, we obtain the elasticity to the real part and the elasticity to the imaginary part as described in Equation 5.11. The results can be seen in Figure 5.11.

Both the predator birth factor and the predator death rate have a large, but opposite, influence on the exponential envelope. Hence, adjusting their respective values by 10%, results in the model behaviour and real eigenvalues as seen in Figure 5.12.

In Figure 5.11 predator birth factor has a high positive elasticity. Since the real part of the complex conjugate pair is negative, modifying this parameter would lessen

---

3 Obtained by directly calculating the change in the eigenvalues if a given parameter is changed. This was done using the framework as described in Chapter 6.
5.3. **EIGENVALUE ELASTICITY ANALYSIS**

The dampering and cause the oscillation to be maintained for longer periods. The *predator death rate* has a high negative elasticity and should have the opposite effect.

The graphs of the behaviour of predator confirm this, while the graphs of the eigenvalues of the modified model also agree. Since the initial conditions have been preserved between model runs, the behaviour of the model where *predator birth factor* is modified deviates slightly from the prediction. Relative to the other two models it is started further away from the equilibrium.

This form of the method is useful, but limited. It only relates parameters to behaviour, not feedback structure to behaviour. This is useful, especially for designing alternatives, but eigenvalue elasticity analysis can be extended by analysing the relationship between the loops in the model and its behaviour.
5.3.2 Loop based analysis

The parameter based version of eigenvalue elasticity has the disadvantage that it tracks the influence of parameters on eigenvalues, thereby analysing the link between different parameters and the behaviour of the linearised model. What we are looking for however, is the link between structure and behaviour, so if eigenvalue elasticity analysis is to meaningfully contribute as a form of model analysis, there must be a form which can relate the elasticities to individual loops. Kampmann (1996a) addresses this problem by extending the method and showing the link to loop dominance analysis. By transferring the model to descriptor form, the relation between individual edges in the model and the eigenvalues is made. Using a matrix representation of a subset of the loops in the model, he then links the edge elasticities to loop elasticities. Using the descriptor form he relates individual changes in edge gains to changes in a certain eigenvalue and calculates edge elasticities.

The elasticities of the edges provide us with valuable information, but this is only useful for loop dominance analysis if we can establish a relation between changes in loop gains (Equation 4.8) and changes in the eigenvalues. As it turns out, the characteristic polynomial $P(\lambda)$ of the Jacobian $J$ is dependent on a summation of loop gains in the system (see Kampmann, 1996a, pg. 9). Consequently, so are the eigenvalues. The proof of this relation can be found in Reinschke (1988).

Since changes in edge gains affect changes in loop gains, edge elasticities from the basis for the loop dominance analysis. However, in order to draw the necessary conclusions, it should be possible to control the gain of each loop without influencing the gain of the others. This can be done by using an (S)ILS for the analysis (Section 4.1.2), which has the added advantages of growing only linearly with model size and having a DCM with linearly independent columns.

The edge elasticities and the loop elasticities are related via

$$
\begin{pmatrix}
  e_{l1} \\
  \vdots \\
  e_{ln}
\end{pmatrix}
= D
\begin{pmatrix}
  e_{p1} \\
  \vdots \\
  e_{pn}
\end{pmatrix}
$$

in which $e_{li}$ is the elasticity of causal link $i$, $e_{pj}$ is the elasticity of loop $j$ and $D$ the directed cycle matrix (DCM).

Since the DCM has linearly independent columns, we can calculate its pseudo-
5.3. EIGENVALUE ELASTICITY ANALYSIS

Table 5.5: Loops in LV

<table>
<thead>
<tr>
<th>Loop Name</th>
<th>#</th>
<th>Variables</th>
<th>Edge</th>
</tr>
</thead>
<tbody>
<tr>
<td>Predator births</td>
<td>$L_1$</td>
<td>${\text{predator, predator birth}}$</td>
<td>predator $\rightarrow$ predator birth</td>
</tr>
<tr>
<td>Predator deaths</td>
<td>$L_2$</td>
<td>${\text{predator, predator deaths}}$</td>
<td>predator $\rightarrow$ predator deaths</td>
</tr>
<tr>
<td>Food</td>
<td>$L_3$</td>
<td>${\text{predator, encounter kills, total prey deaths, prey, predator births}}$</td>
<td>prey $\rightarrow$ predator birth</td>
</tr>
<tr>
<td>Kills</td>
<td>$L_4$</td>
<td>${\text{prey, encounter kills, total prey deaths}}$</td>
<td>prey $\rightarrow$ encounter kills</td>
</tr>
<tr>
<td>Prey deaths</td>
<td>$L_5$</td>
<td>${\text{prey, standard deaths, total prey deaths}}$</td>
<td>prey $\rightarrow$ standard deaths</td>
</tr>
<tr>
<td>Starvation</td>
<td>$L_6$</td>
<td>${\text{prey, starvation deaths, total prey deaths}}$</td>
<td>prey $\rightarrow$ starvation deaths</td>
</tr>
<tr>
<td>Prey births</td>
<td>$L_7$</td>
<td>${\text{prey, prey birth}}$</td>
<td>prey $\rightarrow$ prey birth</td>
</tr>
</tbody>
</table>

inverse and consequently, the loop elasticities via

\[
\begin{pmatrix}
\frac{e_{p1}}{e_{p1}} \\
\vdots \\
\frac{e_{pn}}{e_{pn}}
\end{pmatrix} = D^{-1}
\begin{pmatrix}
\frac{e_{t1}}{e_{t1}} \\
\vdots \\
\frac{e_{tn}}{e_{tn}}
\end{pmatrix}
\]  \hspace{1cm} (5.14)

The interpretation of the elasticities is the same as for the parameters (Section 5.3.1), but now reveals the influence of the independent loop on the eigenvalue. To illustrate this, the analysis will be performed on a short run of the example model.

The independent loop set is the same as the set used in the discussion of Ford’s behavioural approach. The loops and the associated unique edges can be found in Table 5.5.

The results for calculating the elasticities for these loops can be found in Figure 5.13. The measure used is the one proposed by Kampmann (1996a) as in Equation 5.11. Results for taking the pseudo-inverse and the unique edges are the same. To interpret Figure 5.13, it must be related to the eigenvalues calculated during the model run (Figure 5.10). The real part of the complex conjugate pair has a time constant of about $1/0.04$ at its peak, while the period fluctuates between $2\pi/0.16$ and $2\pi/0.09$ time units. Hence, both play a significant role in the behaviour of the model.

The loop that plays the largest role in creating the oscillatory behaviour is, not surprisingly, the food loop. To test this, we change the gain of the food loop by modi-
fying the predator birth factor. Although this also changes the contribution of the predator birth loop, its effect should still be very visible. So, if we add 40% to the predator birth factor to increase the effect of the food loop, we should see a stronger oscillation, in spite of the opposite, weaker effect of the predator birth loop. This is indeed the case, see Figure 5.14 for the behaviour of prey.

For the real part, we see that the predator death loop has the largest influence on the eigenvalue, $L_3$. This is the loop with the largest negative real eigenvalue elasticity. We can completely isolate this effect. The model approaches its equilibrium significantly faster with the gain of the predator death loop increased by 40%. The effect of modifying the starvation loop turns out to be equivalent; the net effect of modifying each loop is about the same, see Figure 5.15.

---

4 The initial conditions of the model have been slightly modified to make the effect of the changes
5.3. EIGENVALUE ELASTICITY ANALYSIS

Figure 5.15: Modifying dampening loops
5.3.3 From Eigenvalues to Variables

As it stands now, both methods demonstrated do not utilise the full power of eigenvalue elasticity analysis. These versions only relate loops and parameters to overall system behaviour. However, the analyst might be interested in the effect of a given loop on a specific variable. Since we have the link from parameters and loops to eigenvalues, this can be done by relating the eigenvalues to variables, which will complete the path from structure to behaviour of a specific variable. For the sake of clarity, this paragraph will assume we are investigating loop elasticities. Conclusions can easily be transferred to the parameter based form of the analysis.

In order to do this, we begin with a homogeneous, linear $n$th-order system. Our goal is to find the influence of a specific eigenvalue $\lambda$ on the change in slope of the variable of interest. The system has a fundamental matrix $\Psi$.

Since $\Psi$ is a solution of the system and consists of a linear combination of eigenvectors

$$\dot{\Psi}(t) = G\Psi(t) = \Psi(t) \begin{pmatrix} \lambda_1 & \cdots & \lambda_n \end{pmatrix}$$

Then, for the slope vector $s$ of the state variables.

$$s(t) = \Psi(t) \begin{pmatrix} \lambda_1 & \cdots & \lambda_n \end{pmatrix} c = \Psi(t)\alpha$$

So the vector $\alpha$ is

$$\alpha = \begin{pmatrix} \lambda_1 & \cdots & \lambda_n \end{pmatrix} c$$

And parallel to the solution of the system

$$\alpha = \Psi(t_0)^{-1}s(t_0)$$

Now, if we diagonalise $\alpha$, we can obtain the following

$$S(t) = \Psi(t) \begin{pmatrix} \alpha_1 & \cdots & \alpha_n \end{pmatrix}$$

We refer to $S$ as the slope matrix. This contains the individual contribution of the more visible.
eigenvalues to the slope of each state variable and is related to the slopevector \( s \) by a summation along its second dimension.

\[
s_i = \sum_{j=0}^{n} s_{i,j}
\]

Each row of the slope matrix contains the contribution of each specific eigenvalue to the slope of a variable of interest. Since the slopes are either real, or occur in complex conjugates, the imaginary parts of the slope matrix can be ignored when calculating the slope vector at a specific point in time.

Using this, we can calculate the change in slope based on the differences in the slope matrix over a certain timestep \( dt \). Convention is to set \( dt \) equal to the integration timestep, the time at which we linearise as 0. Then,

\[
dS \approx \frac{(S(dt) - S(0))}{dt}
\]

In order to obtain a weighted sum of the elasticities and keep results readable, the contributions of different eigenvalues to specific variable \( v \) are scaled to

\[
C(v, i) = \frac{dS(v, i)}{\sum_{j=0}^{n} |dS(v, j)|}
\]

Here \( C \) is the matrix containing the scaled contributions of the eigenvalues to all state variables.

This can be combined with the information obtained about the eigenvalue elasticities. We multiply the elasticities by the relative contribution of the associated eigenvalue. Say we build an elasticity matrix \( E \) with all loop to eigenvalue elasticities, where each column relates to loop \( p \) and each row to an eigenvalue \( i \). Each entry in the matrix is the elasticity, in whichever form the analyst prefers, of a loop to an eigenvalue, \( e_{pi} \).

\[
E(i, p) = e_{pi}
\]

The next logical step would be to multiply the two matrices to obtain the overall elasticities, \( O \), from a loop to the state variables.

\[
O = CE
\]

And again, to make the results more readable, we reform the overall elasticities to
relative elasticities by
\[ R(v, i) = \frac{O(v, i)}{\sum_{j=0}^{n-1} |O(v, j)|} \]

5.3.4 Review

In contrast to the other two methods, the eigenvalue elasticity analysis can deal with oscillations and has the ability to clearly attribute different behaviour modes to different parts of the feedback structure. Secondly, once the method is understood, it can be implemented and automated relatively easily in, for instance, MatLab. However, most implementations done in this paper so far are model-specific. Additionally, there are several forms of this analysis, none of which is inherently superior to the other, while work is being done on forms of analysis that are different again.

There still are a few drawbacks to the method however. The first of these is its questionable scalability. Can the method and its reasoning be extended to larger models?

The second is the fact EEA relies heavily on abstract mathematical concepts. The math and difficult interpretation of outcomes may influence how willing a client is in accepting the results of this form of analysis, or an analyst in applying the method.

Thirdly, the method as it is formulated right now relies on the linearly independence of the right eigenvectors of the linear system. Aside from this, it appears to offer little insight into models displaying chaotic behaviour (Kampmann and Oliva, 2005). It is still based on linear concepts.

And, finally, although the SILS is a very useful concept, the modeller must beware that the effects of certain loops cannot always be separated in the real system. For instance, it would be very hard indeed to separate the effects of the predator birth and the food loops.

5.4 Conclusions

Selection of Methods

- According the analyses performed here, the disadvantages of the PPM outweigh its advantages. Its myopic form of analysis and the algorithm’s inability
to deal with oscillations\textsuperscript{6} (or vice versa) are reason to not include it in the rest of this research (Section 5.2).

- Ford’s behavioural approach has its drawbacks but still is usable. If any modifications to the method turn out to be necessary, these will be discussed in the case studies. However, useful as it may be, it is not defined strictly enough to be automated.

- Eigenvalue elasticity analysis supported by a Shortest Independent Loop Set seems the most promising and powerful method for dominance analysis. This mainly due to its ability to describe either system behaviour or the behaviour of a single variable and its ability to deal with oscillations properly.

- It appears that structural analysis is a necessary aid for the remaining techniques for understanding model behaviour. The concept of the independent loop set is particularly useful.

Remaining Issues

- The scalability of both remaining methods for understanding behaviour is questionable. It is unknown whether their application scales well to models of medium or larger size.

- The application of different forms eigenvalue elasticity analysis has been done in an ad hoc fashion so far. Current ways of applying this cannot deal with models of normal size, blocking the investigation of the applicability of the methods to higher order models. Given this and the many forms of this type of analysis, what is needed is a clear and consistent way of linking a model to the different forms of analysis. This will the main focus of the next chapter.

- For now, it seems that a different structure for mathematically equal models results in a different output from the analysis.\textsuperscript{7} Consequently, although there is no such thing as the correct model of a problem, it helps if the structure of the model reflects the structure of the real world system.

To illustrate, we will look at different structures possible for our example model. The first option, Figure 5.16(a), separates the effect of starvation and standard deaths, but bases predator birth purely on the interaction between the two populations. In contrast, we could have put the starvation and standard death loops together in one effect on total deaths of prey in the example model, as in Figure 5.16(b). In this case it would have been very hard to find

\textsuperscript{6}This is touched upon in other publications (Kampmann and Oliva, 2005; Güneralp, 2005).

\textsuperscript{7}Model analysis follows the GIGO – Garbage In, Garbage Out – principle: input a badly structure model and the analysis drops in value.
the dampening effect of the starvation loop. On the other end of the spectrum, in the original model we introduced an extra loop by letting predator births be influenced by prey and predator, as in Figure 5.16(c). In that case, the analysis might, for instance, lead us to options relying on a separation of effects that aren’t separable in the real world.

Figure 5.16: Possible structures for the LV model (all system parameters left out)
Chapter 6

A Framework for Model Analysis

This chapter will describe the framework designed to tackle model analysis. The first section will delve into the need for the framework. In the second section I briefly describe the requirements. The third section contains the conceptual design. Finally, the functions used for analysis will be described. Appendix D contains a description of a prototype implementation of the framework.

6.1 The Need

The intention of the framework is to be able to consistently perform model analysis. The framework consists of a method of separating analysis, model and integration, while rigidly defining which information travels between the different parts. A consistent approach, supported by the speed of automation should help in investigating and applying model analysis techniques.

As it is now, current SD applications do not support this at all and it is not easy to automatically obtain the information about the model necessary to perform the analysis\(^1\). The current approach is to do the analysis by hand, write model specific functions, or, for some forms of structural analysis, use PERL to interpret text files and deduce model structure from those (Kampmann and Oliva, 2005).

Doing the analyses by hand is no longer feasible for even medium sized\(^2\) models. The second approach has only very limited applicability and is not flexible enough to easily adjust for the different forms of analysis. For instance, it would not be possible to replace the Kampmann (1996b) algorithms with the Güneralp

\(^1\)For instance, Powersim is not able to provide linearisations of a model.
\(^2\)6 to 10 level variables
(2005) way of relating elasticities to specific variables. Or, to adjust the granularity of the analysis. So, while this approach may work in the short term, it has several disadvantages:

- It only applies to one form of analysis.
- It does not provide the modeller with the flexibility required to deal with larger models.
- The scripts used to transform the model from Vensim often cannot deal with, for instance, if-then-else constructions, severely limiting the number of models that are analysable.

And, in my opinion, it is a dead end for further research. If, for instance, more research is to be done by taking a more graph oriented perspective on a SD model, a script reading a text layout of the model will never be enough. The model needs to built differently from the ground up with the ability to offer a different perspective on demand.

6.2 Requirements

The main requirements for the framework are consistency and flexibility. Consistency in the way of working with different models and analyses; one general approach to multiple forms of analysis. Flexibility in being able to replace parts of the framework and change properties of one part without affecting the other. Lastly, the individual parts should be set up as generically as possible, making them applicable to as broad a range of models as possible.

As stated above, the idea central to the framework would be to separate the integration, the model and the analysis as much as possible. They are to be seen as completely independent entities exchanging information, each with their own responsibilities. If this is setup in a modular and generic enough way, this would enable us to replace and change one of the three without affecting the others. Consequently:

- The integration method and its settings (e.g. RK4, with a timestep of .25) should be independent from the model and analysis. It should, for example, be feasible to do the desired analysis at a much lower (or higher, in certain cases) granularity than the integration.
- The model should be able to interface with the tools for analysis and the integrator so that both can obtain and feed back the information required to run or analyse the model. Meanwhile, the inner workings of the model should be
kept hidden as much as possible. This will require a way of implementing the model that can provide this information by request.

- The analysis part should be able to obtain the information necessary for performing its calculations and work as generically as possible. However, for now no information feeds back to the other two parts. Given the focus on EEA the analyses will mainly request linearisations of the model and representations of structures (Chapter 4).

6.3 Conceptual Design

The concept can be divided in several different parts. First, the information exchanged between the three parts of the framework will be defined. Secondly, the main procedure, which oversees the integration of the model and the analysis will be described. Thirdly, the outer shell of the model that can provide the other parts with the required information. After that, a concept for the internal structure of the model that can generate the information necessary for analysis. And, finally, the functions for the analysis. The integrator required no new concepts, we can hook everything into existing functions.

6.3.1 Information Transferred

What needs to be defined before we design the rest of the framework is the information that one part obtains from the other. For an overview, see Figure 6.1.

![Figure 6.1: Setup for the framework: Information exchanged between the different parts](image)

The integrator uses the rate of change of the state variables to determine their value for the next time step. For every timestep, it feeds their value back into the model,
which then calculates the rate of change for the next timestep.

The structural analysis functions need a representation of the structure of the model. A form on which they can perform. Traditionally, this is an adjacency matrix. The output of these function also serves as input for loop based dominance analysis.

The dominance analysis functions need different linearisations\(^3\) of the model in order to be able to calculate, for instance, eigenvalues and related elasticities. Furthermore, they need references to model parameters for the parameter based variants of analysis.

### 6.3.2 The Overall Procedure

The purpose of the main procedure is to execute model runs and analyses; linking the separate parts of the framework together. All the while it should be able to quickly change any of the underlying components. It calls the functions required for the structural and dominance analyses of the model, while using the integrator to run the model. The procedure needs the following input:

**The model**

A reference to the model in a representation suitable for use by the framework.

**Timesteps**

A vector of time steps for the integrator.

**Analysis times**

A separate vector of times at which to perform the analysis.

**Integrator**

A reference to the integration function to be used.

After performing the necessary structural analysis, the procedure runs the model until the moment for which we have defined the first analysis. It stops running at that point, performs the analysis, stores the results and continues to run until the next time for analysis. This continues until either the last time for analysis or the last timestep has been reached. For an overview see Figure 6.2.

During intervals of rapidly changing loop gains, the times for analysis can be closer together. So, both the integration timesteps and the times at which the analysis is performed are completely up to the analyst. The intervals of interest can be determined by tracing loop gains and/or eigenvalues and checking how variable they are, but this is not included in the function and still up to the modeller. A small prototype MatLab implementation of this procedure can be found in Appendix D.

---

\(^3\)Preferably a descriptor, we can calculate the gain matrix from that.
Figure 6.2: Overview of the main procedure of model analysis
6.3.3 The Model

As stated above, the purpose of the model representation is to provide the rest of the framework with the necessary information. The model itself lies between its contents and the other functions in the framework, keeping all unnecessary complexity away from the other components. It is an outer shell, hiding its internal workings from the rest of the framework. To summarise, the model should be able to:

- Interface with the integrator to provide and obtain the information needed to run the model, namely rates of change and the results of integration for each timestep.
- Provide the tools for structural analysis with a graph representation of the model.
- Provide the tools for dominance analysis with the information needed to perform those analyses.
- Keep the actual implementation of the model completely hidden to all other tools and only provide information by request. It is specifically not the intention for the analysis tools to delve deeply into the model. The goal is to have them request the necessary information. Hence, the model lies between the tools and the variables.

The model can provide the data required to run and analyse the model. This includes building linearisations and adjacency matrices and providing the integrator with the net rates of change of the states. Behind it lies the model structure, which consist of a set of linked variables.

6.3.4 Internal Model Structure

The internal way of building the model should not matter to the rest of the framework, but we need some form of structure that is able to generate the information required. In essence, the loop based variants of model analysis use a graph perspective on System Dynamics models that is very different from the usual approach, which sees a variable as a function instead of a vertex. However, the graph perspective is incorporated into the analysis only at the very last point. The internal structure presented here uses that perspective from the ground up, making sure that a System Dynamics model can be perceived as a graph when necessary.

The purpose of the concept here presented is to show one of the forms that this could take. For an implementation, see Appendix D.4.
6.3. CONCEPTUAL DESIGN

The essence of this concept is the hierarchy of the elements in a System Dynamics model. All elements can be seen as vertices\(^4\) in a graph, where the system is the graph. The model elements are divided into variables and and parameters. See Figure 6.3.

A variable is a model element that is dependent on the value of other parts of the model or time. Each variable should be able to calculate the gains from its predecessors to itself\(^5\). There are two special forms of variables, namely the state (a.k.a stock or level) and the rate (a.k.a. flow).

States are considered different from the rest of the variable classes since

- It’s value changes only through integration of the model and it can (indirectly) accept values from the integrator.
- If the model is stopped at any point during it’s run, the state values are still known\(^6\). The values of other variables have to be deducted from states and parameters.
- All of its predecessors are rates, determining its net rate of change (Saleh, 2000; Forrester, 1976).

The last item also defines the reason that rates are different from the rest of the variables. Namely, they are the only type of variable that can directly influence a state. Consequently, all predecessors to a state must be rates.

Auxiliary variables (constitutive equations) are not considered to be special cases, they have no distinguishing properties.

Parameters form the exogenous influences on the model, the system boundaries. They are assumed constant throughout one model run. The parameters usually represent control parameters for the problem owner, or exogenous influences that cannot be changed by the problem owner.

This way of representing the model can provide us with the graph perspective since every element is already defined as a vertex. Secondly, edge gains can be determined by having the variable estimate the gains from each of its predecessors to itself. This allows us to linearise the model automatically. Thirdly, rates of change can be determined for each timestep using a recursive method call starting at the states of the model. And, finally, the separate definition of parameters allows us to request these for purposes of analysis. To summarise, this can provide us with the information the rest of the framework needs in order to undertake model analysis.

---

\(^4\)Corresponding, each relation would be an edge, but this is left out for now.

\(^5\)In the prototype definition it is possible to define the method of calculating the gain of a specific variable. This for the sake of variables with discrete functions.

\(^6\)In fact, this is one of the ways Sterman (2000) suggests to identify stocks. He suggests the modeller imagines “freezing” the model in a snapshot. The variables that retain meaning or value are very likely to be stocks.
6.4 The Analysis Functions

These are mainly the functions that implement the dominance analysis from Forrester (1982), Kampmann (1996b) and Güneralp (2005) and the structural analysis from Oliva (2004). They have been generalised as far as possible. Some of the functions required rewriting to a more generic form, or the translation from one representation of structure to another (Appendix D.3). For instance, the algorithm used by Güneralp (2005) to determine an eigenvalue’s contribution to the behaviour of a state variable has been translated to a generic method applicable to an \( n \)th-order model. In the current setup, the necessary data can be obtained from the model and the functions perform their calculations.

In line with the setup for the framework, only the required information and the results of the functions are defined. The implementation itself will be made more specific in the Section 6.5.

Table 6.1: Analysis functions currently present in the framework

<table>
<thead>
<tr>
<th>Analysis</th>
<th>Required Input</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Loop Eigenvalue Elasticity Analysis</td>
<td>Model Linearisations, Directed Cycle Matrix</td>
<td>Loop elasticities in the form requested</td>
</tr>
<tr>
<td>Parameter Eigenvalue Elasticity Analysis</td>
<td>Model Linearisations, Model Parameters</td>
<td>Parameter elasticities in the form requested</td>
</tr>
<tr>
<td>Elasticity analysis related to a specific variable</td>
<td>Model Linearisations, Results of the elasticity analysis</td>
<td>Overall elasticity of specific variable to a given parameter or loop</td>
</tr>
</tbody>
</table>
6.5 Prototype

To validate or refute the above concept, a small prototype was built. The prototype is a combination of Java classes for the model and MatLab functions for the overall procedure, integration and analysis.

Given the heavy reliance of the dominance analysis algorithms on linear algebra, MatLab has been used to implement the analysis functions and overall procedure. The language features an external interface to Java, so this made it feasible to link the model defined in Java to the algorithms in MatLab.

The model and the internal model structure have been defined in Java. The object oriented nature of the language allows for an almost direct translation of concept to specification; we can almost directly translate the conceptual hierarchy from Section 6.3.4 into a class hierarchy. Furthermore, the language allows us to strictly define what information is exchanged between classes.

The current prototype allows the integrator and all forms of analysis to request the information they need from the model. The models defined can be analysed automatically. However, the current implementation is still in a very rudimentary form. For a detailed description see Appendix D.

6.6 Conclusions

Based on what has been found in designing the proof of concept and the prototype, the concept seems implementable. Several different methods have been described in the previous chapter and most of the graphs (as in figure) in the later sections have been made using the framework.

At this stage it is possible to build a model that can be easily translated into a graph and at the same time provide the integrator and analysis tools with the requested information. The concept and its implementation will be tested more thoroughly on the Yeast (Section 7.2) and Long Wave (Section 7.3) models.

The current implementation of the internal model structure can be improved. For instance, the way in which models are set up is too labour intensive. Secondly, the implementation could be linked to the JUNG\(^7\) library, further extending the possibilities of viewing a System Dynamics model as a graph.

However, supporting the graph perspective "from the ground up" and linking this to existing graph libraries could open the door to other areas of research. For instance, we could do cluster analysis of model structure using existing algorithms.

\(^7\)see http://jung.sourceforge.net and Appendix C.2
The major disadvantage of the current approach is that we lose existing model base. Translating models is not easily automated and disproportionately labour intensive to do by hand.
Chapter 7

Performing The Analyses

This chapter will compare the different methods of analysis by applying them to two relatively small models.

7.1 Selecting The Test Models

In order to test the selected methods of analysis, two models have been selected as test cases. In the first place, these models have been selected to show several properties of the methods, while not posing too much difficulty to implement. The models should both be relatively small. In addition, at least one the models should still be of sufficient size and complexity to pose a challenge to interpret. And, finally, a comparison of the results obtained in this thesis with those of previous authors is desired. Hence, both models should have been the subject of previous analyses.

Consequently, to meet these requirements, two models have been selected; the Yeast and the Simple Longwave. The Yeast model has been used to check the algorithms for the contributions of eigenvalues to specific variables. In spite of the fact that is a fairly simple 2\textsuperscript{nd} order model, it shows several interesting effects when calculating elasticities. It has been the subject of analyses by Saleh (2002) and Güneralp (2005). The Simple Longwave Model is a small, but highly connected, 3\textsuperscript{rd} order nonlinear model. Its behaviour and structure are complex enough to be considered as a test for the algorithms on whether they can deal with highly nonlinear models. It has previously been analysed by Kampmann (1996a) and Güneralp (2005).
CHAPTER 7. PERFORMING THE ANALYSES

7.2 Yeast

The first model to be analysed is the Yeast model. This representation of the growth of Yeast cells in a vat is a relatively simple, second order, overshoot and decline model. The cells multiply and eventually die out as a result of the alcohol they produce. In the initial stages of the model, there are few Cells present and virtually no Alcohol. This results in an almost exponential increase of the amount of Cells. As the number of Cells grows, so does the Alcohol concentration. At a certain point the influence of alcohol on the deaths and births of cells is so large, the growth of cells begins to decrease until the amount of cells reaches a maximum at \( t \approx 65 \). From there on the effect of Alcohol on the Cells is so great that these decrease in number until the amount of Cells eventually approaches zero.

The model has complex eigenvalues over a certain interval. See Figure 7.1. We will compare the results of Ford (1999), Güneralp (2005) and Kampmann (1996a). Also, when relating the elasticities to specific variables, the difference in outcomes between the different measures of elasticities will be demonstrated.

Our variable of interest for the analysis will be Cells. The graph of its behaviour, divided into phases according to atomic behaviour pattern, can be found in Figure 7.2, together with the behaviour of Alcohol and the eigenvalues of the model.
Figure 7.2: Behaviour of Cells, Alcohol and Eigenvalues in the Yeast model. The roman numerals indicate the phases of the state variables according to atomic behaviour pattern. For the eigenvalues, $\lambda_1$ and $\lambda_2$ represent the real parts of the eigenvalues of the gain matrix of the linear system at the selected points in time, $\lambda_c$ is the (positive) imaginary part of the complex conjugate pair of eigenvalues that exists between $t = 38$ and $t = 78$. 
7.2.1 Ford

Modifications to the Method

As illustrated in Figure 7.2, we can divide the behaviour up into four phases based on the atomic behaviour pattern of the variable of interest. For each of the phases we will take out the loops to see if a change in the atomic behaviour pattern of the variable of interest occurs. Ford’s method has been adjusted to keep it internally consistent and speed up the process of analysis. This is particularly important given that this analysis is still done by hand, although automation might support it. The adjustments include:

- The same method for switching off loops is used for different loops and different intervals. This should make the results between intervals and loops more comparable. Furthermore, by using the same method, the time at which a loop is turned off is easily adjusted and the analysis for one loop over several intervals is quickly performed, thereby speeding up the process. The analysis is presented on a per interval basis, but the execution is done per loop.

- Ford (1999) defines a set of shadow loops as two or more dominant loops that generate the same atomic behaviour pattern, so that taking out one of the loops does not cause the atomic behaviour pattern to change because the other loop continues to generate this\(^1\). Consequently, a set of shadow loops has been found if the behaviour pattern only changes when two loops are eliminated (Table 7.1).

<table>
<thead>
<tr>
<th>Loop A</th>
<th>Loop B</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inactive</td>
<td>Active</td>
<td>No change in behaviour pattern</td>
</tr>
<tr>
<td>Active</td>
<td>Inactive</td>
<td>No change in behaviour pattern</td>
</tr>
<tr>
<td>Inactive</td>
<td>Inactive</td>
<td>Behaviour pattern changes</td>
</tr>
</tbody>
</table>

- The search for pairs of shadow loops dominating behaviour has been restricted to those intervals where no one dominant loop is found. This reduces analysis time again. Multiple loops can still dominate over the same interval.

Applying The Method

For the first phase, as displayed in Figure 7.3, we find, unsurprisingly, that dominance can be attributed to $L_1$, the growth loop. The birth loop drives the exponential growth. Taking out the third loop shows that it functions as a brake on the

\(^1\)This has been discussed in Section 5.1.2
growth of the Cells. The effect of alcohol for the first phase via the two death loops (L2 and L4) is so small that the reference run and the run with the loops taken out are visually indistinguishable.

Figure 7.3: Results of taking loops out during the first phase. The blue, dashed line always shows the original run for reference. The shading denotes phases of behaviour.

For the second phase, we obtain the results displayed in Figure 7.4. Here we see that the balancing growth is dominated by L3, the loop containing the effect of alcohol on birth. Apparently, since taking this loops out causes the model to display exponential growth, its effect on the growth of the number of cells is so large, the growth is declining.

No clearly dominant loop was found during the analysis of the third phase in the first round of analysis. The first four graphs in Figure 7.5 show the effect of taking out the individual loops. According to our procedure, the search for a shadow pair of loops is required for this interval. Hence, Ford’s procedure requires that combinations of loops be taken out, the results of which are displayed in Figure 7.6. The
Figure 7.4: Results of taking loops out during the second phase
only combination that changed the behaviour pattern was that of L3 and L4. This leads us to the conclusion that these loops form a pair of shadow loops causing the exponential decline.

Note that taking out this connection results in the following equation for the behaviour of Cells:

\[
\frac{dC}{dt} = (c_1 - c_2)C
\]

In which \(c_1\) represents the net effect of alcohol on births at the time at which the loops are taken out and the division time of the cells, and \(c_2\) represents the net effect of alcohol on the deaths of cells and the life time of the cells. The resulting first order linear differential equation shows the balancing behaviour displayed, since \(c_2 > c_1\). The large time constant makes the behaviour appear linear, but running the model far past its normal runtime reveals the balancing decline.

Figure 7.5: Results of taking out loops during the third phase

The final phase shows a single dominant loop again. In this case it is L2. The eventual decline towards 0 is caused by the death of Cells. See Figure 7.7.
Figure 7.6: Results of taking out combinations of loops during the third phase
Figure 7.7: Results of taking loops out during the second phase.
CHAPTER 7. PERFORMING THE ANALYSES

Results

It turns out the nature of the current model representation is very well suited for this type of analysis, enabling a similar approach in deactivating an edge leading to any variable, even to state variables. The Java representation allows for each for each variable to be stateful\(^2\) and the analysis uses this to remember past values locally (for a specific variable) and lock loop gains based on the past value.

For instance, for Cell Birth, the variable remembers the value of Cells at the time at which a loop the edge from Cells to Cell Birth is set off and uses this for the rest of run. We simply change the time at which to turn the loop off to analyse a different interval.

A severe disadvantage of the method is the time consuming nature of this type of analysis. The analyst has performed 22 different model runs to perform this analysis on a second order model with four different phases of behaviour. This includes rewriting equations to support the switching off of loops. Analysing a second variable in the same model would require comparable effort, although some work could be reused.

Additionally, the usage of the concept of shadow loops is very doubtful. The suspension of disbelief required when comparing the original model to the model with one loop taken out is small enough to still accept results. However, a model with two loops taken out is so different from the original model that its is questionable whether the conclusions based on the modified model are valid.

In spite of these disadvantages, checking the behaviour of the modified models with the behaviour expected due to these modifications, provides a structured method for assessing the role of the loops in the system. However, contrary to, for instance, an extreme conditions analysis its results are not so predictable that we can use them for validation; the exact role this method of analysis could play in the modelling process is still unclear.

7.2.2 Loop Eigenvalue Elasticity

The second analysis we will perform on the Yeast model is the Loop Eigenvalue Elasticity Analysis (LEEA) as proposed by Oliva and Kampmann. The SILS is obtained using the algorithms as described in Oliva (2004).

Modifications to the Method

First of all, we will divide the behaviour of the model up based on the model eigenvalues, not on the behaviour of a specific variable. This is shown in Figure 7.8.

---
\(^2\)A variable is able remember what has happened at an earlier time in the simulation.
The changes in phases based on eigenvalues do not coincide with the phases identified for the behaviour of Cells in the previous paragraph, except for the change of sign in the imaginary pair at $t \approx 65$. The first phase shows divergent behaviour without oscillation. In Phase II, this changes to divergent oscillatory behaviour. However, the real part of the complex conjugate pair eventually drops below zero, shifting the behaviour to dampened oscillation. At this point in time though, Cells displays non balancing behaviour. In the time spanned by Phase II and Phase III, the shortest period of the oscillations is about $\frac{2\pi}{0.19} = 33$ time units. At $t \approx 78$ the imaginary pair bifurcates into two negative real eigenvalues, after which the model approaches an equilibrium at $Cells = 0$ and $Alcohol \approx 11$.

![Figure 7.8: Phases in the behaviour of the eigenvalues of the Yeast Model. Dashed vertical lines show the changes in phases according to the atomic behaviour pattern of Ford.](image)

Now, using the independent loop set we will calculate the loop eigenvalue elasticities in the Yeast model. First, we will do this by using the measure proposed in Kampmann and Oliva (2005), loop influence. This is defined as

$$ l_i = \frac{\partial \lambda_i}{\partial g} g $$

where $l_i$ is the influence on a particular eigenvalue $\lambda_i$ of a loop $g$. In contrast to the usual measure of elasticity, this measure can deal with eigenvalues that approximate zero since the influence's value is not related to the size of the eigenvalue.

The measures we will use are the real part of $l_i$ to determine the influence of the loops on the exponential envelope, the imaginary part to determine the effect of a loop on the oscillations and the absolute value of $l_i$ as an estimate for the overall influence of a loop. The elasticities of all eigenvalues can be found in Figure 7.9. The analysis was performed with .3 time units between snapshots of the model. The integration was run with a timestep of .1 time unit.

The most prominent feature of these graphs is that all elasticities show peaks
CHAPTER 7. PERFORMING THE ANALYSES

(a) Real elasticities first eigenvalue
(b) Real elasticities second eigenvalue
(c) Imaginary elasticities first eigenvalue
(d) Imaginary elasticities second eigenvalue
(e) Overall elasticities first eigenvalue
(f) Overall elasticities second eigenvalue

Figure 7.9: All elasticities for the Yeast model
around \( t \approx 38 \) and \( t \approx 78 \). Initially, I suspected that the elasticities were supposed to peak at the values shown at these times. To verify that this was the case, the analysis was run again, but only for the interval around \( t = 38 \) and with a much higher granularity; the interval between points of analysis was now \( 3 \cdot 10^{-3} \). The integration time step was kept the same, except for the interval where it needed to adjust for the analysis. The results for the overall elasticity can be found in Figure 7.10.

It turns out that the closer the analysis comes to \( t = 38 \), the higher the peak of the elasticities. At both this point and \( t = 78 \) two real eigenvalues join in a complex pair, or a complex conjugate pair bifurcates into two real eigenvalues. As noted in Kampmann and Oliva (2005), the analysis cannot deal with repeated eigenvalues and linearly dependent eigenvectors. This agrees with the analytical formulation for an influence of a particular gain \( g_{ij} \) in the gain matrix on \( \lambda_k \), for which

\[
\frac{\partial \lambda_k}{\partial g_{ij}} g = l_k(i)r_k(j)g
\]

where \( l_k \) is the left eigenvector associated with \( \lambda_k \) and \( r_k \) the right eigenvector. Left eigenvectors are the row vectors for which, for a square matrix \( A \), \( l_k A = \lambda_k l_k \). \( L \) is the matrix consisting of the left eigenvectors of \( A \), \( R \) is the matrix consisting of the right eigenvectors of \( A \). And, since, as in Forrester (1982)

\[
L^{-1} = R
\]

The left eigenvectors are undetermined when \( R \) has linearly dependent columns. The last is the case when there are repeated eigenvalues.

In the current model, our eigenvalues are unique, except around the points such...
as $t = 38$, where, if we approach the point where the eigenvalues merge into the pair, they are almost equal. At the exact point of merging, the eigenvalues are not unique, the right eigenvectors form a singular matrix and the left eigenvectors are undetermined. At these points, the elasticities are undetermined, running off towards infinity. The fact that LEEA cannot deal with a singular matrix of right eigenvectors is mentioned in Kampmann and Oliva (2005). But not only does this influence the scalability of the method\(^3\), this also restricts our analyses to certain intervals within which the eigenvalues do not change from real separate eigenvalues to a pair of complex conjugates or vice versa.

Two measures can be taken in order to deal with this behaviour. First of all, to keep the results of the analysis readable we can scale the elasticities to values between $-1$ and $1$ by dividing them by the sum of the absolute of all elasticities. G"uneralp (2005) already does this, but he never makes the issues with the bifurcation points explicit. Secondly, it is very questionable whether intervals of analysis can contain points with undetermined elasticities. Consequently, this is a valid reason to divide the analyses up into several open intervals, each between the points at which the elasticities are undetermined.

The matrix has poorly conditioned eigenvalues (Deuflhard and Hohmann, 1995) round these points. A poorly conditioned eigenvalue is highly sensitive to small perturbations of its matrix. Consequently, not only are the elasticities high, but so are the potential errors in calculating the eigenvalues due to, say, roundoff errors. The condition number for the eigenvalues is shown in Figure 7.11; the higher this number gets, the worse the conditioning of the eigenvalues is. Since the entire analysis relies on the calculation of the eigenvalues of the gain matrix, its validity becomes questionable in the neighbourhood of these points.

It might still be the case that our results are valid close to the switch of the eigenvalues, but this has to be verified. In order to verify this, the sensitivity of the results of loop dominance analyses around these point should be investigated. That is, if we change the model slightly, without affecting the overall behaviour, does the analysis stay consistent around the bifurcation and merge points?

To continue the analysis, we will rescale the the graphs so that the loop elasticities fall between $1$ and $-1$. Given the problems above, we will divide the analysis into four phases, as show in Figure 7.8. The divisions between phase 1 and phase 2 and between phase 3 and 4 coincide with the singularities described above. The division between phase 2 and phase 3 is due to the change in signs of the eigenvalues.

\(^3\)Larger matrices are more likely to have singular or close to singular matrices of right eigenvectors (Kampmann and Oliva, 2005)
7.2. YEAST

Applying The Method

As can be seen from Figure 7.12, the elasticities agree with the analysis based on Ford’s approach. L1 and L3 play a significant role in determining the behaviour, with L3 having the opposite influence to L1 on both eigenvalues. The second eigenvalue is, for most of the interval, much smaller than the first, having little effect on overall behaviour. L1 is primarily responsible for the exponential growth, while L3 restrains that growth. While the overall elasticities give an indication of the magnitude of the influence of the loops, they tell us nothing about the direction of that influence.

As seen in Figure 7.13, the dominance of the first loop in relation to the real part of the elasticity continues in the second phase. However, L3 has a strong influence on the imaginary part. As both the influence of the first loop on the imaginary part of the complex pair and the absolute value of the real part of the complex pair are dropping, the overall elasticity of the third loop is larger than that of the first loop from $t = 50$ onwards. This agrees with the Ford analysis.

Although Ford and the behavioural analysis produced comparable results for the first two phases, this is not the case in phase 3. Where Ford finds the L3 and L4 loops dominant, the elasticity analysis assigns the most importance to L2 and L4, while the contribution of L3 is almost negligible during this phase. The elasticity analysis holds the two death loops responsible for the exponential decline.

During the fourth phase we see that the absolute value of the first eigenvalue quickly

---

4 The eigenvalues form a complex pair, elasticities only shown for the eigenvalue with a positive imaginary part.
CHAPTER 7. PERFORMING THE ANALYSES

Figure 7.12: Elasticities and associated $\lambda$s for the Yeast model, phase 1
7.2. YEAST

Figure 7.13: Elasticities and associated $\lambda$ for the Yeast model, phase 2.
CHAPTER 7. PERFORMING THE ANALYSES

Figure 7.14: Elasticities and associated $\lambda$ for the Yeast model, phase 3
becomes much larger than the second one. Consequently, we consider this eigenvalue to be dominant during this phase. L2 has by far the largest elasticity for this eigenvalue. L4 has an opposite elasticity, but quickly drops off to zero. L2 is the loop responsible for the last phase of exponential decline.

![Graphs showing elasticities and associated λ for the Yeast model, phase 3](image)

**Figure 7.15: Elasticities and associated λ for the Yeast model, phase 3**

**Results**

This analysis demonstrated that although the Yeast model shows smooth behaviour for its eigenvalues, it is undefined at points where the eigenvalues merge into conjugate pairs, or when a pair bifurcates into two real eigenvalues. This necessitates dividing the analysis up into particular phases according to the eigenvalues and makes the validity of the results of the analysis around these points questionable. The graphs are kept readable by rescaling them to values between $-1$ and 1.

Ford and this form of LEEA are not completely comparable. The first focuses on a specific variable while the other takes a system wide perspective. However, their results agree for the most part and are interpretable for this model. This is not
CHAPTER 7. PERFORMING THE ANALYSES

the case for the entire analysis though. The fact that the validity of the Ford method is questionable when taking out more than one loop is confirmed by the elasticity analysis. LEEA cannot be accepted as absolute truth though, but the fact that its results do not agree with Ford does cast doubt on the Ford analysis.

Also, using the framework the LEEA analysis is much faster than Ford’s behavioural approach. Assuming that the model is in analysable form, the analysis itself is only a matter of running the appropriate functions. However, although the results are more easily obtained, their interpretation is far less obvious than those of Ford. It takes practice and effort to be able to analyse eigenvalue elasticity graphs, especially when models become larger and display less intuitive behaviour. As Kampmann and Oliva also stress, there is a need for a way of representing eigenvalue elasticity analysis results that facilitates interpretation, while not oversimplifying a complex method.

7.2.3 Parameter Eigenvalue Elasticity

Eigenvalue elasticity analysis is not restricted to just the loops in the model, but we can also use this to measure the effect of different parameters on the eigenvalues of the system. The changes in the parameters are always made in the direction of the sign they currently have. That is, positive parameters are adjusted towards positive infinity, negative parameters towards negative infinity. The following definitions where used for eigenvalue elasticities, where $p_i$ is the parameter under investigation:

$$
\epsilon_1 = \frac{d \Re \{ \lambda \} \ p_i}{dp_i \ |\lambda|}, \ \epsilon_2 = \frac{d \Im \{ \lambda \} \ p_i}{dp_i \ |\lambda|}
$$

Applying The Method

The division into phases and the necessity for rescaling the elasticities apply to the parameter based variant of EEA as well. For the sake of brevity we will not include the unscaled elasticities and show the four phases in one graph only. See Figure 7.16, Figure 7.17 and Figure 7.18.

During the first phase behaviour is dominated by the parameters associated with the birth of cells. The increase in the division time for cells would have a negative effect on the dominant eigenvalue in the system. That is, less division would cause slower growth. Comparable results can be seen for the Add. Effects Births and Mult. Effect Births, but in the opposite direction. This suggests that if these were increased in this phase of the modelling run, the eigenvalue would be made larger and growth would speed up.

In the beginning of the second phase the two parameters that have the largest positive influence on the frequency of oscillation are the Alcohol Generation Per
Figure 7.16: Parameter Elasticities and associated $\lambda$ for the Yeast model, Part 1
Figure 7.17: Parameter Elasticities and associated $\lambda$ for the Yeast model, Part 2
7.2. YEAST

Figure 7.18: Parameter Elasticities and associated $\lambda$ for the Yeast model, Part 3
Cell and the multiplicative effect of alcohol on the births of cells. Meanwhile the additive effect has the opposite effect. The same goes for the exponential effect on the deaths of cells, which becomes the dominant influence around the switch from Phase I to Phase III. However, its effect mainly restricts the frequency of oscillation. Since the imaginary part isn't expressed very clearly, it is unclear what the effect of increasing the parameters would be. The exponential envelope remains primarily influenced by the three parameters influencing the birth loop. Increasing the additive effect of alcohol on births would speed up growth.

The third phase is primarily dominated by the exponential effect of alcohol on the deaths of cells. This agrees with the loop based analysis as this effect influences the gain of both death loops.

In the final phase, the exponential effect of alcohol on Cell Deaths is dominant and there is a very minor effect from the lifetime of Cells. This agrees with the dominance of L2 according to the loop based analysis in the previous paragraph, as both parameters influence the gain of L2.

Results

In contrast to the loop elasticity analysis, this method deepens our insight into the specific role of parameters during a model run. The advantage of this is that parameters are often the leverage points for the problem owner. This shows us what effect each leverage point has during the course of the model run. The disadvantage of this method is that it actually doesn't relate structure to behaviour. The mechanisms (loops, pathways) via which the parameters influence the eigenvalues are not clarified.

However, one method does not exclude the other and they can compensate for each others weaknesses. For instance, we can use the loop based elasticity analysis as a means of explaining the influence of a specific parameter in more detail. The parameter based variant compensates for the loop based variant's lack of relation to leverage points and helps to relate it to the language and system understanding of the client.

The hard part however, lies in the interpretation. In order to interpret these graphs the analyst needs to know what elasticity analysis means and how it works. It is not immediately clear from the graph what the effect of increasing or decreasing a parameter would be, especially when the eigenvalues are complex, or when there are more than 2 eigenvalues present.

7.2.4 Elasticities to states

It is possible to link eigenvalue elasticities to the behaviour of states in the model. The algorithms used to do this are based on the ones from Güneralp (2005). In
order to do this, we use the same division into phases as found in Section 7.2.2.

**Modifications To The Method**

In the methodology proposed by Güneralp (2005) he first delves into listing all causal links and nodes in the model and identifying the loops in the SILS. Fortunately, by using the frame work we could now request the adjacency matrix of the model and quickly run this through the structural analysis functions as described in Oliva (2004). However, the output of the functions written by Oliva\(^5\) contains descriptions of all loops, but not in the form of a Directed Cycle Matrix. The output had to be rewritten to a form usable for the rest of the analysis.

As it turned out, Güneralp (2005) used a rather unusual definition for the eigenvalue elasticities. Whereas Kampmann (1996b) used

\[ \epsilon_1 = \frac{d\text{Re}\{\lambda\}}{dg} \frac{g}{|\lambda|}, \quad \epsilon_2 = \frac{d\text{Im}\{\lambda\}}{dg} \frac{g}{|\lambda|} \]

The elasticity measure used by Güneralp (2005) is

\[ \epsilon_1 = \frac{d\text{Re}\{\lambda\}}{dg} \frac{g}{\text{Re}\{\lambda\}}, \quad \epsilon_2 = \frac{d\text{Im}\{\lambda\}}{dg} \frac{g}{\text{Im}\{\lambda\}} \]

On the one hand, the latter measure has two advantages. The first being that the sign of elasticities is defined relative to the eigenvalue. Consequently, if a loop or a parameter draws the eigenvalue towards zero, decreasing the magnitude of effect it currently has, the elasticity is negative, making it suited for determining the influence of a particular loop to the behaviour of a state. The second is that the change in the exponential envelope or frequency of oscillation is measured relative to the appropriate part of the eigenvalue. So, even if the eigenvalue has a very small real part and a large time constant, a loop that has a large relative impact on that time constant, is given a large elasticity. The same goes for the imaginary part of the elasticity and associated frequency of oscillation. This property of the measure makes it more explicit in defining the elasticity to a particular part of the eigenvalue.

On the other hand, there is a disadvantage to this measure. If an eigenvalue has a real part that is close to zero while the imaginary part is significantly larger, the real part has relatively little influence over the associated dynamics in the short term. However, using this measure results in any loop having a significant impact on the real part of the eigenvalue being attributed a large elasticity, while its influence on the dynamics of the model may be small.

\(^5\)Obtained via R. Oliva’s resource page
For purposes of comparing the implementation of the framework with results from other authors, the measure as defined by Güneralp (2005) will be used for the analysis of the Yeast model. However, given the issues described in the above paragraphs, its results will be compared to that of the usual definition of the elasticity. This measure does not relate an elasticity to the sign of the eigenvalue, making it unsuited for relating it to the slope contributions of the eigenvalues. This is solved trivially by multiplying it with the sign of the relevant part of the eigenvalue under inspection, so that

$$
\epsilon_1 = \text{sgn}(\text{Re}\{\lambda\}) \frac{d\text{Re}\{\lambda\}}{dg} \frac{g}{|\lambda|}, \quad \epsilon_2 = \text{sgn}(\text{Re}\{\lambda\}) \frac{d\text{Im}\{\lambda\}}{dg} \frac{g}{|\lambda|} \quad (7.2)
$$

With regards to the contributions of the eigenvalues to the behaviour of Cells, see the graph in Figure 7.19. This is somewhat different than the graph seen in Güneralp (2005). One of the main differences being that the contributions of the conjugate pair are separate. Where Güneralp (2005) keeps these together to form a single positive contribution of one, the method used here assign a contribution of 1/2 for each member of the pair. In the original publication, the method used calculates the contribution of each eigenvalue by hand, which would necessitate rewriting this step in the method for each new model analysed. The method used here is generic and can be applied to a n-th order model without modification (Appendix D.3.3).

The second difference is that around the point of inflection for Cells, the contributions of the conjugate pair switch sign. This is consistent with the definition of a contribution of an eigenvalue as posed in Güneralp (2005). To quote the article:

The rescaled changes [in slope] vary between -1 and 1. A negative rescaled change at any time step means that behaviour mode decreases the slope of the state variable of interest in that interval. A positive change, however, means that the behaviour mode increases the slope of the state variable of interest in that interval.

The strange thing is that the sign switch does not occur in the graph accompanying the Güneralp’s analysis of the Yeast model. Between the inflection points at $t = 50$ and $t = 75$ the slope of Cells is decreasing. And since both eigenvalues must have the same contribution, their contribution must be negative according to the rule quoted above. Further discussion with the author of the article clarified the anomalous behaviour.

In drawing the graph for the elasticities, the initial assumption was that during the entire analysis, the overall elasticity, the magnitude of the imaginary and the

---

6It was a slight calculation error (see Appendix C)
7.2. YEAST

Figure 7.19: Contributions of the eigenvalues in the Yeast model. When the curvature of Cells changes sign ($t = 50$ and $t = 70$), the contributions change sign.

real elasticity combined, was used. Hence, we have no negative elasticities and the overall elasticity of a loop to the behaviour of a variable can only be negative due to a negative contribution of the eigenvalues influenced by the loop. This resulted in the graph for the overall elasticities for Cells seen in Figure 7.20. However, between $t = 0$ and $t = 38$ the sign for the elasticity of $L3$ is negative in the original graph.

Figure 7.20: Overall elasticities for Cells in the Yeast model. The results for $50 < t < 75$ switch sign due to the contributions of the eigenvalues.

Assuming that we are using the overall elasticity throughout the entire analysis, the elasticity cannot be negative for this loop. Hence, its negative influence should be due to negative contribution of one of the eigenvalues. But, as can be seen in Figure 7.19 the eigenvalue with the negative contribution has a contribution that is negligible to that of the eigenvalue with the positive contribution. This falsifies the assumption regarding the overall elasticities.

The only other logical form would be to use real elasticities when the eigenval-
ues for which the elasticity is being calculated is real. This would result in no longer having positive elasticities throughout the analysis. See Figure 7.21.

This again is almost equal to the graph as in Güneralp (2005). The difference now is the sign of the third loop. The influence of the second eigenvalue is negligible again though. So, the negative sign of the third loop must be due to a negative elasticity to the largest eigenvalue. The behaviour displayed does agree with the analysis from the previous section and with the textual description of the elasticities by Güneralp (2005).

![Figure 7.21: Overall elasticities for Cells in the Yeast model only applied when there is a complex conjugate pair of eigenvalues.](image)

**Applying The Method**

As said previously, the measure being used for eigenvalue elasticity in this method is questionable. This only is a problem if the results of the analysis are changed significantly by changing the method of calculating the elasticities. Changing this to the Kampmann (1996a) elasticity and keeping the rest of the method the same results in the analysis displayed in Figure 7.22. In Phase III, $L_3$ emerges as the loop most responsible for behaviour, while $L_4$ dominates phase III. Previously, both loops were considered as less influential than $L_2$. Hence, for that interval the conclusion of the analysis changes depending on which version of the elasticity is taken.

The system wide analysis points to the fourth loop being responsible for the exponential decline and $L_3$ being dominant in the time interval displaying balancing growth. The parameter based analysis also indicates that the effect of alcohol on death is the most influential parameter during this phase. Finally, the effect of alcohol on Cell Death displays a sharp increase over the third phase. This causes the exponential decline as seen in phase III, providing yet another argument for iden-
Figure 7.22: Overall elasticities for Cells in the Yeast model

tifying the loop containing this variable as dominant in Phase III. See Figure 7.23.

Figure 7.23: Effect of alcohol on Cell death

The method again attributes dominance to L2 and L4, in contrast to Ford and in agreement with the system wide analysis.

The results of analysis are almost identical to that of the system-wide variant. The main difference is that the interpretation of this variant is much clearer than in that form. That is, the analysis actually indicates that in Phase I $L_3$ restricts the growth of $Cells$. 
CHAPTER 7. PERFORMING THE ANALYSES

Results

First of all, there were disagreements between the results shown in this chapter and those as presented in the original paper presenting the algorithm. However, differences mentioned have all been explained.

Secondly, this analysis shows that there are qualitative differences in outcome depending on which elasticity measure the analyst uses. The measure chosen by Güneralp (2005) seems highly doubtful since both the outcome of the inspection of the actual behaviour of the variables and elasticity analysis based on parameters differ from Güneralp.

Thirdly, although this version of the analysis technique used a finite difference method for determining the link eigenvalue elasticities, it showed that the the Güneralp (2005) analysis can be automated using the framework presented in Chapter 6. Replacement of the finite difference method by the method described by Güneralp (2005) (Equation 7.3) is possible, feasible even. The elasticity is defined by

\[
e_{l_j,i} = \sum_{p=1}^{P} \sum_{q=1}^{Q} e_{pq,i} \frac{\sum_{s=1}^{S} g_{pq,s}}{\sum_{r=1}^{R} g_{pq,r}}
\]

(7.3)

where

- \(e_{l_j,i}\) is the elasticity of \(\lambda_i\) to link \(j\).
- \(e_{pq,i}\) is the compact elasticity of the element of the gain matrix
- \(g_{pq}\) is the gain of a pathway from state \(p\) to \(q\)
- \(S\) are the pathways the causal link is a part of
- \(R\) are all pathways from \(p\) to \(q\)

All this would require is an identification of the paths between states and calculation of their gains. The first can be done by structural analysis\(^7\) and the second is trivial once the paths are known.

And, lastly, like the system wide analysis, relating the elasticities to specific variables results in an outcome conflicting with Ford’s behavioural analysis.

7.3 Long Wave

The second model to be analysed is the Simple Long Wave model as used in Kampmann (1996b), Güneralp (2005) and Ford (1999). This is a highly nonlinear eco-

\(^7\)Identify all pathways from \(p\) to \(q\) using a treewalker and select those paths that contain \(e_{l_j,i}\). The pathways need only by found during the structural analysis part of the main procedure of the framework.
nomic model used by Sterman (1985) to explain the long term economic cycles caused by capital self-ordering in the simplest terms possible.

This resulted in a third order model of the capital sector of an economy where the stocks represent three different accumulations of capital: capital, supply and backlog. The capital sector is viewed as an aggregate, so it orders capital from itself; plants, equipment and the materials required to operate (Sterman, 1985). The capital stock represents the current level of capital in the system, which slowly degrades due to depreciation, while acquisitions form the source of new capital. The supply chain of capital is represented by the supply stock, since new capital cannot be generated instantly. Meanwhile, backlog represents the amount of outstanding orders, both from the consumer sector (goods orders) and the capital sector itself.

The version of the model used here is different from the one used in the mentioned publications. Where applicable, it uses continuous functions to approximate graph functions. This is further elaborated on in Section 7.3.3. Given the sensitive nature of the model, this does result in quantitatively slightly different behaviour. However, the basic shape of behaviour, a limit cycle with a period of approximately 40 years is the same.

A diagram of the structure of the model can be found in Figure 7.24. A detailed description of the model, its equations and its directed cycle matrix can be found in Appendix A.3. The model displays cyclic behaviour with a period of approximately 40 years (Figure 7.25). For our analysis we will use the third cycle, from $t = 80$ to $t = 120$. The model has settled into its cycle by then and there is no longer any perceivable difference between the outcomes of the fifth and fourth cycle. The behaviour of the eigenvalues and states is displayed in Figure 7.26.

The analysis will focus on the techniques used in the analysis of the Yeast model. Section 7.3.5 treats Ford’s behavioural approach, Section 7.3.2 the Loop based Eigenvalue Elasticity Analysis, Section 7.3.3 the Parameter based Eigenvalue Elasticity Analysis and Section 7.3.4 the translation from eigenvalue elasticities to the behaviour of the states. The order in which the analyses are performed is different from that in the analysis of the Yeast model. This is due to the labour-intensive nature of the Ford’s behavioural analysis, which makes the technique unsuited for applying it to a larger model in its full form (Section 7.3.5, Section 7.2.1).

7.3.1 Preliminary Analysis for the Eigenvalue Elasticity Analysis

As discussed in Section 7.2.2, the first step in performing the EEA is inspecting the behaviour of the eigenvalues and identifying any bifurcation and merger points. Based on this, the cycle of the model between $t = 80$ and $t = 120$ will be divided into four different phases. The dominant loops for each of these phases will be selected based on their influence on the eigenvalue that is perceived as the dominant one.
CHAPTER 7. PERFORMING THE ANALYSES

Figure 7.24: The Simple Long Wave model

Figure 7.25: Behaviour of the states of the Simple Long Wave model
7.3. LONG WAVE

The behaviour of the eigenvalues and the related condition numbers can be found in Figure 7.27. At \( t = 81 \) the three negative real eigenvalues change into one negative real eigenvalue and a complex pair, signalling the start of phase I. There is no significant jump in the condition number of the eigenvalues, since the switch from real eigenvalues to complex is discontinuous and the eigenvalues never approximate each other. Consequently, the matrix of right eigenvectors is not singular around this point. At the transition into phase II the complex pair bifurcates into three separate real eigenvalues which causes a singularity, but for the sake of brevity this is not shown. Moving from phase II to phase III into one real eigenvalue and a complex pair does display a clear singularity. The last transition into Phase IV, in which the model has three real negative eigenvalues, triggers a singularity as well. This is shown in the graphs zooming in on the interval \( 90 < t < 95 \).

The elasticities are expected to be visibly discontinuous around these points. In order to perform the system-wide analyses, the eigenvalue perceived as the dominant one is selected for the elasticity analysis. For the complex eigenvalues, both the overall, the real and the imaginary elasticity will be shown. For the real eigenvalues, only the real elasticity will be displayed. For this model, the dominant eigenvalue is, if all eigenvalues are negative, the one with the largest absolute value. If there is any eigenvalue with a positive real part, then that eigenvalue is selected as the dominant one ( Table 7.2 ). This arbitrary, though necessary rule for selecting a dominant eigenvalue is one of the weakness of the system wide elasticity analysis and addressed by several other forms of EEA (Güneralp, 2005; AbdelGawad et al., 2005).

The granularity - the interval between the snapshots - of the system-wide loop based analysis will be kept constant at 0.05 time unit. To demonstrate the ability of

---

Figure 7.26: Behaviour of States and Eigenvalues in the Long Wave model

---

\(^8\text{It is only visible at a granularity of } 5e^{-3} \text{ time unit} \)
CHAPTER 7. PERFORMING THE ANALYSES

(a) Eigenvalues

(b) Condition numbers

(c) Eigenvalues for $90 < t < 95$

(d) Condition numbers for $90 < t < 95$

(e) States

Figure 7.27: Phases of the longwave model based on its eigenvalues and associated condition numbers
the framework to adjust its granularity where required and to save computing time, the interval between snapshots has been set to different values according to the dynamics of each phase for the parameter based variant and the variant relating elasticities to the behaviour of specific states. When relating the elasticities to a specific state, these same granularities are used again.

<table>
<thead>
<tr>
<th>Phase</th>
<th>Dominant eigenvalue</th>
<th>Interval</th>
<th>Complex</th>
<th>Sign</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>$\lambda_2$</td>
<td>$81.15 &lt; t &lt; 84.65$</td>
<td>Yes</td>
<td>+</td>
</tr>
<tr>
<td>II</td>
<td>$\lambda_2$</td>
<td>$84.65 &lt; t &lt; 91.35$</td>
<td>No</td>
<td>+</td>
</tr>
<tr>
<td>III</td>
<td>$\lambda_2$</td>
<td>$91.35 &lt; t &lt; 93.85$</td>
<td>Yes</td>
<td>+</td>
</tr>
<tr>
<td>IV</td>
<td>$\lambda_3$</td>
<td>$93.85 &lt; t &lt; 120$</td>
<td>No</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 7.3: Adjusted granularity of the analysis for the different phases in the Long Wave model. The granularity is not adjusted for the system-wide loop-based analysis.

<table>
<thead>
<tr>
<th>Phase</th>
<th>Granularity</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>0.05</td>
</tr>
<tr>
<td>II</td>
<td>0.05</td>
</tr>
<tr>
<td>III</td>
<td>0.01</td>
</tr>
<tr>
<td>IV</td>
<td>0.5</td>
</tr>
</tbody>
</table>

### 7.3.2 Loop Eigenvalue Elasticity Analysis

**Applying The Method**

The loop set used is the one based on the SILS, not the set used by Kampmann (1996a). The details of the loop set can be found in Appendix A.3. This analysis is similar to the one performed by Kampmann (1996a). The main difference lies in the selection of the loop set and the fact that this analysis is, to all intents and purposes, continuous in time. Kampmann (1996a) selects only one point in time as representative of the dynamics in of entire phase, while this analysis is performed at a fixed interval, small enough to be considered continuous.

Accordingly, for phase I, our elasticities are displayed in Figure 7.28. $L14$ drives the exponential increase in phase I, while the $L4$ restrains the growth during the first part and $L9$ during the second part. Although $L7$ displays the largest influence on the imaginary part of the complex pair, $L14$ is still considered as the loop most responsible for current behaviour. This due to the fact that, as mentioned in Kampmann (1996a), the period of the imaginary part is too long relative to the
CHAPTER 7. PERFORMING THE ANALYSES

time constant of the real part to play much of a role in the dynamics. In addition, the overall elasticity of the fourteenth loop is still larger.

$L14$ is known as the Self Ordering loop, one of the sources of disequilibrium identified by Sterman (1985). It is also identified by Kampmann (1996a) as the main dominant loop in the first phase. The initial change from the end of the previous cycle to Phase I is caused by an increase in Desired Orders, which triggers a response in Relative Orders. As soon as relative orders becomes positive, the Backlog starts to grow causing an increase in Desired Production. The increased Desired Production requires a larger capital stock to satisfy its needs, which, via Capital Adjustment and Desired Orders results in more Relative Orders. This in turn leads to a further increase in Backlog via Backlog Orders, hence the name of the loop. See Figure 7.3.2. At the end of phase I the Self Ordering loop is no longer active since Relative Orders has reached its maximum.

The growth of capital accelerates in phase II. The exponential behaviour is supported by several loops with large real elasticities ($L3$, $L5$ and $L8$), while another group ($L1$, $L2$, $L7$ and $L6$) restrains it. See Figure 7.30. Given that most loops with a positive elasticity are countered by loops with a equally large negative elasticity, the selection of a single dominant loop is impossible. The Self Ordering Loop is no longer active in this phase.

The reinforcing loops are $L3$, $L5$ and $L8$, the strongest being $L8$, the Capital Expansion loop. The positive loops share some common structure, see Figure 7.31. The third loop drives the growth of Capital by increasing Capacity and consequently Production, so that more Capital is acquired. $L5$ does approximately the same, but instead of travelling directly from production to acquisitions, it uses the decrease in Backlog caused by the increase in production, which results in an increase in the acquisitions. In $L8$, the capital drives its own expansion by an increase in capital orders caused by more depreciation. The capital orders result in an increased supply, which eventually causes further growth of capital.

The restraining loops consist of the first order loops $L1$, $L2$ and $L7$. The first order loops, $L1$ and $L2$ need no further explanation. $L7$ works via the negative causal link from Backlog to acquisitions. An increase in backlog causes a decrease in acquisitions. This eventually results in a decrease of Capital Orders via Capital and Depreciation. The decreased orders restrain the growth of Backlog. $L6$ also has a minor balancing influence, but is not shown in the loop diagram. $L6$ is almost the same as the Capital Expansion Loop, but instead of directly going from capacity to production, it follows the path from capacity to production via Capacity utilisation, which results in a balancing role for this loop.

$L14$ is again regarded dominant in phase III. However, it now supports the rapid
7.3. LONG WAVE

7.3.1 Elasticities in Phase One

(a) Real elasticities in phase one

(b) Imaginary elasticities in phase one

(c) Overall elasticities in phase one

Figure 7.28: Elasticities for the dominant eigenvalue in phase one of the Simple Long Wave model
CHAPTER 7. PERFORMING THE ANALYSES

Figure 7.29: Influential loops in Phase 1

Figure 7.30: Real elasticities for the dominant eigenvalue in phase II
7.3. LONG WAVE

(a) Loops with positive elasticities

(b) Loops with significant negative elasticities

Figure 7.31: Influential loops in Phase 2. The first collection of loops drives the eigenvalue, while the second, balancing collection restricts the exponential behaviour.
decline of Supply and Backlog; it reinforces the decline in Relative Orders. The main stabilising loop is $L_9$, restraining the real part of the eigenvalue. The rest of the loops have very little influence, until $t = 93.4$. At that point in time, the model changes since relative orders abruptly become zero, making $L_5$ and $L_4$ the most influential loops. See Figure 7.32. This abrupt change in underlying dynamics does not have any visible effect on the behaviour of the states of the model. See Figure 7.27.

Phase IV is the most straight forward of the phases. It is a long period of relative stability where behaviour is determined by three negative real eigenvalues. The eigenvalue with the largest absolute value is considered dominant, but the elasticities to all eigenvalues are shown in Figure 7.33. The three eigenvalues are dominated by the two negative loops $L_1$, $L_2$ and $L_4$, which function as a control on backlog via Desired Production and Capacity Utilisation. See Figure 7.3.2. Phase IV lasts until the increase in desired orders triggers a response in Relative Orders, beginning a new cycle.
7.3. LONG WAVE

(a) Real elasticities

(b) Imaginary elasticities

(c) Overall elasticities

Figure 7.32: Elasticities for dominant eigenvalue in phase three of the Simple Long Wave model
Figure 7.33: Elasticities for dominant eigenvalue in phase III of the Simple Long Wave model
Figure 7.34: Balancing Loops in Phase IV
Results

The weakest point of this variant of EEA is the selection of an almost arbitrary eigenvalue as the dominant one (Kampmann and Oliva, 2005). This detracts the interpretability of this form of analysis. The larger the model, the more difficult this variant becomes to interpret. The results of the method can be made more accessible, either by extending the method with the algorithms as developed by Güneralp (2005).

7.3.3 Parameter Eigenvalue Elasticity Analysis

Modifications to the Method

In order to perform parameter based eigenvalue elasticity analysis, the model had to be modified relative to the version used by Kampmann (1996a) and Güneralp (2005). The graph functions used have been approximated by parametrised functions. The reasons for this are a) graph functions are not differentiable b) a function is far easier to translate to non-SD languages and c) the behaviour of a parametrised function can be changed by using specific parameters. This allows the modeller to control certain aspects of the involved graphs such as their maximum height and the steepness of their incline, while retaining the characteristic shape of the function (Appendix A.3). Independently, Kampmann and Oliva (2005) also encountered these problems and used the same solution as presented here.

The phases and dominant eigenvalues are the same as the ones used for the loop elasticity values (Figure 7.27). The code passes the basic consistency checks. For instance, the response to changes in the parameters related to relative orders is zero when relative orders itself is zero in Phase IV. Since the elasticity values will be scaled back to values between $-1$ and $1$ and related only to one eigenvalue, the elasticity measure for a parameter $p_j$

$$e_i = \frac{\partial \lambda_i}{\partial p_j} p_j$$

(7.4)

where, $\lambda_i$ is the eigenvalue under inspection, $p_j$ the parameter under inspection and $e_i$ the eventual elasticity. This is similar in definition to the loop influence (Kampmann and Oliva, 2005). The granularities used for each individual interval can be found in Table 7.3.

Applying The Method

In Phase I the parameters with the largest eigenvalue elasticity are Capital Output Ratio, Relative Order Height and Delivery Delay. Capital Output Ratio influences
the gain from Desired Capital to Capital Adjustment. The edge leading from Desired Production to Desired Capital is influenced by Delivery Delay. Relative Orders is influenced by Relative Orders Height. All of these directly influence $L14$, which is the dominant loop according to loop based analysis. The restraining function of $L4$ agrees with the restraining effect of Delivery Delay.

The switch of sign between the two is most likely related to the change in curvature in relative orders. The dominant loop is $L14$, and both parameters directly influence the gain of that loop. The shift in sign, which variable restricts the exponential growth and which variable drives it, occurs at almost the same time as the changing of sign of the curvature of relative orders (Figure 7.35). In addition, slightly changing the parameters also changes the Jacobian of the system. Comparing the changes in the Jacobian before and after the shift should be consistent with the role of Relative Orders. The changes in the Jacobian switch sign for the rows related to supply and Backlog, not Capital. This confirms the role of Relative Orders, since no paths lead from Relative Orders to Capital without going through either Backlog or Supply (Appendix A.3, Figure A.3).

The graph of the elasticity analysis based on the parameters suggests a change in the dynamics of the model, while the roles of the underlying loops remain the same. The seemingly conflicting results can be explained by the fact that the two parameters both influence the dominant loop and merely switch roles. This has little effect on the net dominance of the loops in the system.

![Figure 7.35: The curvature of Relative Orders versus the elasticities of Capital Output Ratio and Delivery Delay. The left y-axis is the curvature of Relative Orders, the right y-axis shows the scaled elasticity of the two parameters.](image-url)

The parameter elasticities for Phase II pose a challenge to interpret. Perturbing the parameters results in the graph for the eigenvalue elasticities as shown in Figure
CHAPTER 7. PERFORMING THE ANALYSES

Figure 7.36: Parameter Elasticities for dominant eigenvalue in Phase I of the Simple Long Wave model
7.3. LONG WAVE

The intervals close to the edges will be ignored in this analysis. As described by AbdelGawad et al. (2005) the elasticity of the different parameters can be related to the edge elasticity using the chain rule. However, the current results are obtained using small perturbations of the parameters and measuring the response in the eigenvalues.

Table 7.4: Parameters with positive and negative elasticities for phase II in the long-wave model

<table>
<thead>
<tr>
<th>Positive</th>
<th>Negative</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cap. Util. Height</td>
<td>Cap. Output Ratio</td>
</tr>
<tr>
<td>Relative Order Height</td>
<td>Cap. LifeTime</td>
</tr>
<tr>
<td>Cap. Util. Steepness</td>
<td>Delivery Delay</td>
</tr>
<tr>
<td>Relative Order Period</td>
<td></td>
</tr>
</tbody>
</table>

Since they both influence the same set of paths, the first two positive parameters that will be analysed are Cap. Util. Height and Cap. Output Ratio. Due to the nonlinearities in the model a parameter not only influences the gain of the links coming in to the variable it is directly linked to, but can influence the gains of all edges. Which edges are influenced is determined by structure and state of the model. For instance, when perturbed, both the parameters influence (amongst others) the gain of the paths included in $L_3$ and $L_5$, not just the gain of the edge leading from Capacity to Cap. Util, which is only part of $L_6$. This is in accordance with the analytical formulation of parameter eigenvalue elasticities (AbdelGawad et al., 2005). The net effect of all these paths combined results in Cap. Util. Height and Cap. Output Ratio driving the exponential growth in phase II.

Although the Self Ordering loop, $L_{14}$, is inactive in phase II, perturbation of the parameters Relative Order Period and Relative Order Height results in a strong re-
response in the dominant eigenvalue. This is due to the fact that the perturbation causes $L_{14}$ to temporarily seem active again. It remains to be seen whether the perturbation version of the calculation of the parameter eigenvalue elasticity analysis agrees with the analytical formulation.

With regard to the parameters with negative elasticities, the first one, Cap. Output Ratio restricts the gain of approximately the same paths as the Cap. Util. Height and Cap. Output Ratio. Its influence on the edge from Capital to Capacity restricts the gain of $L_3$ and $L_5$. The exact paths dependent on the parameter are unknown for now. Capital Lifetime restricts both the gain of the balancing loop $L_1$ and the exponential loop $L_8$ via the edge between Capital and Depreciation. Its net effect however, pushes the the eigenvalue towards negative infinity. An increase in Delivery Delay would result in a decrease in Desired Production, which in turn would lead to less Cap. Util, decreasing the gain of $L_3$ and $L_8$.

Since Phase III is short, yet highly dynamic, we adjust the granularity of the analysis to five times that of the one used in the previous phases; 0.01 timestep instead of 0.05. We expect the parameters associated with $L_{14}$ to have the most influence on behaviour. These should be roughly the same as in phase one (Figure 7.38). However, in contrast to the previous phase, the results pertaining to these parameters directly influencing Relative Orders are assumed to be valid again, since $L_{14}$ is active in this phase, although now supporting exponential decline.

Given the length of Phase IV and the relatively stable behaviour of the eigenvalues and the states in this phase, the interval of analysis is set to 0.5 timestep. The phase lasts until about $t = 122$, where Relative Orders is larger than zero again and the model enters phase 1 of the limit cycle again. We will identify the most influential parameter for all three eigenvalues. The results of the analysis are in Figure 7.36. Since the real part of the eigenvalue is negative, the parameters with a positive elasticity are said to decrease the speed of dampening, while those with a negative elasticity increase that speed.

The negative elasticity of Cap. LifeTime is in accordance with the influence of $L_1$ on the same eigenvalue in the loop based eigenvalue elasticity analysis. Increasing this variable would reduce the gain of that loop, pushing the eigenvalue towards positive infinity. The four parameters influencing the second and third eigenvalue do so via the same paths as described in the analysis of Phase II.

---

9The granularity used in the loop based variant of the elasticity analysis was sufficiently small to be applicable for all phases. The runtime of the entire analysis was correspondingly long.
Figure 7.38: Parameter Elasticities for dominant eigenvalue in phase III of the Simple Long Wave model.
Figure 7.39: Parameter Elasticities for dominant eigenvalue in Phase IV of the Simple Long Wave model
7.3. LONG WAVE

<table>
<thead>
<tr>
<th>Eigenvalue</th>
<th>Parameter</th>
<th>Main sign of elasticity</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\lambda_1$</td>
<td>Cap. LifeTime</td>
<td>Positive</td>
</tr>
<tr>
<td>$\lambda_2$</td>
<td>Cap. Output Ratio</td>
<td>Positive</td>
</tr>
<tr>
<td>$\lambda_2$</td>
<td>Delivery Delay</td>
<td>Positive</td>
</tr>
<tr>
<td>$\lambda_2$</td>
<td>Cap. Util. Steepness</td>
<td>Negative</td>
</tr>
<tr>
<td>$\lambda_2$</td>
<td>Capital Utilization Height</td>
<td>Negative</td>
</tr>
<tr>
<td>$\lambda_3$</td>
<td>Cap. Output Ratio</td>
<td>Positive</td>
</tr>
<tr>
<td>$\lambda_3$</td>
<td>Delivery Delay</td>
<td>Positive</td>
</tr>
<tr>
<td>$\lambda_3$</td>
<td>Cap. Util. Steepness</td>
<td>Negative</td>
</tr>
<tr>
<td>$\lambda_3$</td>
<td>Capital Utilization Height</td>
<td>Negative</td>
</tr>
</tbody>
</table>

Results

The current form of parameter based eigenvalue elasticity analysis provides insight into the direction and magnitude of the influence of the parameters in the model on the eigenvalues\textsuperscript{10}. Its results clearly show the effects of perturbing the parameters on the exponential envelope and the frequency of oscillations. However, the analysis still faces several challenges. The stated goal of the analyses performed is relating structure to behaviour, but this particular form fails to do exactly that, which sometimes leads to counterintuitive results. For instance, results can suggest changes in dynamics, while the loop based analysis shows no such change. The direction of influence is known for each model parameter, but what we are interested in is the mechanisms or pathways involved in obtaining that influence. Ideally, the analyst should be able to combine parameter based elasticity analysis with the structure of the model, showing not only what influence the parameters have, but also how this influence is obtained. This would enable the analyst to more effectively relate the parameter influence to the loop influence and the system story.

7.3.4 Relating eigenvalues to states

In contrast to other analyses (Güneralp, 2005; Ford, 1999) we will compare the results of the analysis for two state variables; Backlog and Capital. Aside from differences due to the previously identified issues with this form of the analysis, it is expected that the dominant loops will differ the most between variables in Phase IV.

For the sake of brevity the intermediary results for the elasticities will not be

\textsuperscript{10}Sterman (1985) identifies Capital Output as one of the most influential parameters in determining the behaviour of the Long Wave model. The results of this analysis, which attributes a large elasticity to Capital Output Ratio when $L14$ is active agree with this observation.
shown. The elasticities calculated use the measure as defined in Equation 7.2. The contributions of the eigenvalues to the individual states can be found in Appendix A.3.6. The results of the analyses will be compared on a per phase basis, instead of running the analyses over the entire length of the cycle and then comparing the outcomes. We are mainly interested in where outcomes differ from state variable to state variable and between the current variant of the analysis and the system wide variant of the analysis.

Applying The Method

For the results of the analysis in Phase I, the Self Ordering loop ($L_{14}$) is most influential according to the results of the system-wide variant of eigenvalue elasticity analysis. This is true for the entire interval, except for the very beginning where $L_4$ displays a strong balancing influence. Hence, $L_{14}$ is expected to be most responsible for the behaviour of both states, having the largest elasticity on the dominant eigenvalue during almost the entire phase. The results of the current analysis agree. The behaviour of both variables is predominantly determined by $L_{14}$. The main difference between the role of the loops lies with $L_2$. The change of slope of Capital is restricted by $L_2$, the Supply First Order Control loop, in the beginning of the interval. This is consistent with the temporarily large negative contribution of the third eigenvalue to the behaviour of Capital (Figure A.6) and the positive real elasticity $L_2$ has to this eigenvalue (not shown). In addition, the large elasticity of the negative loop $L_4$ to the real part of the dominant complex pair deserves mention. Although this loop shows a positive influence in this graph, its actual elasticity to that real part is negative. The use of the overall elasticity in case of complex eigenvalues hides this effect. Consequently, in the current form of the analysis the information regarding the actual elasticities is needed to interpret the influence of this loop.

In Phase II the model has three distinct real elasticities. Consequently, the overall elasticity does not need to be used. However, as in the system wide analysis, the interpretation of the results is made more challenging due to the large number of loops with a significant influence on the behaviour of the variables of interest.

Capital’s behaviour is predominantly determined by the positive $\lambda_2$. Based on the contributions, this results in the influences on Capital being almost identical to the results of the system-wide variant of the analysis, since $\lambda_2$ was chosen as the system-wide dominant eigenvalue. The most influential loop driving the behaviour of Capital is $L_8$, while $L_2$ is the most influential restricting loop. That is,

---

11Since the elasticities are defined relative to the sign of the eigenvalue, this means the loop drives the eigenvalue towards negative infinity.

12The magnitude of the complex number of which the real elasticity is the real part and the imaginary elasticity the imaginary.
Figure 7.40: Loop Influences on different state variables in Phase I of the Longwave model
Chapter 7. Performing the Analyses

L8 drives the growth of Capital since it represents the depreciation of more capital eventually leading to more Capital being acquired. The increase of Capital also leads to strong influence of the first order control loop, L2.

While Backlog is also influenced by this same set of loops, the negative contribution of $\lambda_2$ (Figure A.7) results in these influences having the opposite sign to the original elasticities and the influences on Capital. In addition, $\lambda_1$ has a large negative contribution to the behaviour of Backlog, so given the positive elasticity of L4, the Overtime Loop, to this eigenvalue, this loop has a strong negative influence on Backlog, pushing its slope towards zero. This agrees with the balancing role of this negative loop, the increase in backlog leads to a higher Desired Production, which, via the increase in the utilisation of capital leads to a higher outflow out of Backlog through Production. The common-sense explanation for the role of L8 in decreasing the growth of Backlog is it driving role in the growth of Capital. As Capital grows, so does Capacity, which leads to more Production, decreasing the stock Backlog.

When the model enters Phase III, two of the eigenvalues merge into a complex conjugate pair. The phase is referred to by Güneralp (2005), as the capital deceleration phase, while Kampmann (1996a) refers to it as the “self-order collapse” phase. According to the the system-wide analysis, L14 plays a reinforcing role again, but this time in reverse, quickly dropping to zero.

One difference of this analysis with the Güneralp (2005) analysis, is that relative orders drops below zero at the end of Phase III. The Self-Ordering loop, L14 becomes inactive just before the end of around $t = 93.4$, but this effect has not been addressed in previous publications, since the time interval between times at which the model was analysed was too high to detect the effect, although the consequences are visible\footnote{Kampmann’s analysis selects one specific point in time per interval for the analysis. Güneralp has a constant granularity for his analysis of 0.25 time unit, which amounts to only twelve points in time at which he inspects the elasticities in this interval.} (Güneralp, 2005; Kampmann, 1996a). Around this point in time Capital is higher than than Desired Capital, resulting in an almost negative capital adjustment, and, finally, a drop of Desired Orders\footnote{Desired Orders = cap adjustment + depreciation + supply adjustment} to below zero. The eigenvalues still are a complex pair until the end of the phase, so Phase IV only starts when the complex conjugate pair bifurcates.

With regards to the contributions of the eigenvalues to the behaviour of capital, when the growth of Capital starts to decrease, the contribution the complex pair switches sign, while the relative contribution of the $\lambda_3$ increases rapidly (Figure A.8). The last is probably due the contribution of the complex pair crossing zero in its continuous switch of sign. In addition, since we are dealing with a pair of complex conjugate eigenvalues, the elasticities related to that pair are based on the
Figure 7.41: Loop Influences on different state variables in Phase II of the Longwave model
magnitude of the combined imaginary and real elasticity. This necessitates using the original elasticities in order to interpret the results of the analysis, especially since the complex pair has the most significant influence on this state.

The results of the analysis show only positive elasticities before the switch of sign of the contribution of the complex eigenvalues and only negative after the switch. The peak of $L_2$ coincides with the drop of the contribution of the complex pair. Accordingly, the same loop is expected to have a large positive elasticity to $\lambda_3^{15}$ which it has. After the switch of sign, Relative Orders continues to be the most influential of the loops, but now seen as driving the decline of the growth of capital. Again, this is what was to be expected from the elasticities.

However, when comparing the overall results with the elasticities to the real part of the complex pair, $L_9$ has a large negative elasticity to the real part of the complex pair, restraining the exponential envelope, making its exact role in this form of the analysis unclear.

The behaviour of Backlog is predominantly determined by the same pair of eigenvalues. The sign of the contributions is negative, and correspondingly, all loops are given a negative influence over this variable. As with Capital, $L_{14}$ is found to be the most influential of the loops, although $L_7$ and $L_8$ play a significant role as well. The reinforcing decline of Relative Orders under the influence of $L_{14}$ first supports the decrease in growth of Backlog and then reinforces its decline. Note that when relative orders become zero, this coincides with the inflection point of Backlog$^{16}$. Of the other two loops, $L_7$ has a large negative elasticity to both the imaginary and the real part of the complex pair. $L_7$ is a negative loop, so its restraining role to the positive eigenvalue makes intuitive sense. In contrast to $L_7$ or $L_{14}$, $L_8$ does not include Backlog itself. $L_8$, the capital growth loop, sustains the growth of capital, so the (most likely) mechanism for its effect on Backlog is the continued growth of capacity while Cap. Util is near its maximum, resulting in an increase in Production.

In Phase IV, the phase Kampmann (1996a) refers to as the recovery phase, the system has three real, negative eigenvalues which have completely different results for the analysis. Also, based on previous analyses, the different states are expected to have a significantly different outcome for the contributions and eigenvalues. Güneralp (2005) identified that the eigenvalue closest to zero, $\lambda_1$, has a relative contribution of almost 1 to the behaviour of capital (Appendix A.3.6) for the majority of the phase, while Kampmann (1996a) identifies completely different sets of structure as having significant elasticities to the different eigenvalues. The phase ends when the Capital stock is depleted (Sterman, 1985) and $L_{14}$, the Self Ordering

---

$^{15}$For the current definition of elasticity and a negative real eigenvalue this means it pushes the eigenvalue towards negative infinity.

$^{16}$And that of Supply.
Figure 7.42: Loop Influences on different state variables in Phase III of the Long-wave model.
132

CHAPTER 7. PERFORMING THE ANALYSES

Loop, is triggered once again and the next cycle starts.

The results for the analysis of Capital are as in Güneralp (2005), since the change in the measure of elasticity has no effect when dealing with real eigenvalues. $L2$, the Supply First Order Control loop, is dominant for a short while, but after this $L1$ fully determines the behaviour of capital, slowly depleting the stock until the end of the phase.

The contributions of the eigenvalues to Backlog display the expected difference with those to Backlog. Whereas the behaviour of Capital is almost completely determined by $\lambda_1$, Backlog is mainly influenced by $\lambda_2$, while $\lambda_1$ plays a minor role.

With regard to the loops, we see that the $L4$ has a positive elasticity to the negative eigenvalue, increasing the speed of the balancing decline. This entails it pushing Backlog to approach equilibrium more quickly, where the derivative is zero. Hence the counterintuitive positive contribution of the loop to the change in slope of Backlog, which agrees with the negative sign of the loop. In contrast, $L1$, the Capital Decay loop, has a negative influence on Backlog, due to its positive elasticity to $\lambda_2$ which has a negative contribution to the change in slope of Backlog. The loop holds back the balancing decline of Backlog, while it is a balancing loop. Apparently, its influence on the speed of decline of Capital has the reverse effect on Backlog.

In several of the phases different sections of structure drive the behaviour of the two states in different ways. In phase I, both variables are strongly influenced by the reinforcing growth caused by $L14$, while only $Capital$ is significantly influenced by $L2$, the Supply First Order Control loop. Phase II attributes the same influence to the loops with a large elasticity to the positive eigenvalue, but loops are assigned different signs due to the contributions of that eigenvalue to the two states. Aside from this, the main difference for the two states is that, in contrast to $Capital$, Backlog is strongly influenced by $L4$, the Production Scheduling loop. The one thing that is constant for both state variables throughout both phases I and phase III is the destabilising role of the Self-Ordering loop as was suspected by Sterman (1985) in his original analysis of the model.

To summarise the results of the analysis into a system story, as the previous cycle of the Long Wave ends, the summation of the inputs for desired orders becomes larger than zero. This triggers the Self-Ordering loop, which drives the growth of both capital and backlog as long as it’s active during the years 80 to 84. After this the effect of relative orders has reached its maximum, the growth of capital is sustained by the structure as shown in Figure 7.31. However, once capital catches up with desired capital and the sum of incoming variables for desired orders drops below zero, the decrease in backlog starts amplifying itself via the Self Ordering loop, which now works in reverse; Relative Orders rapidly drops to zero over the span...
Figure 7.43: Loop Influences on different state variables in Phase IV of the Long-wave model
of 2.5 years. Once this is zero, the system enters the recovery phase, where capital and supply slowly decline under the influence of \( L1 \) the Capital Decay and \( L2 \), the Supply First Order Control loop. This long, stable phase lasts until capital stock is depleted and desired orders rises above zero once again\(^{17}\), triggering the next cycle.

**Results**

With regards to the framework, the ability to adjust the granularity of analysis for specific analyses opens the possibility of showing detailed, short term effects in the analysis of highly dynamic intervals (Phase III) and speeding up the analysis of long, relatively stable intervals (Phase IV). In addition, the numerical calculation of edge gains circumvents the need for manually calculating these. Obtaining edge elasticities by perturbing the edges and observing the response in the eigenvalues provided some difficulties\(^{18}\), but no significant, unexpected differences\(^{19}\) between results of this analysis and previous analyses have been found. The effort involved in applying the method has been greatly reduced. However, there still is a clear need for implementing an automated version of the analytical approach, where most of the difficulty has now been reduced to the structural analysis involved.

Based on performing this analysis, several conclusions regarding the method can be drawn. First of all, although this seems a trivial observation if algorithms used in this method are inspected, the elasticity analysis is the same for different state variables in the system until the point at which the elasticities are weighted according to the eigenvalue contributions and summed together to calculate the influences on a specific state. Hence, the contributions of the eigenvalues to the behaviour of specific states will indicate exactly where the results of the analysis will differ for different states. Similarly, the results of the analysis will be comparable to those of the system wide analysis if the eigenvalue with the highest contribution to the state variable of interest is the same one as selected in the system-wide variant of the analysis.

Second, the interpretation of the role of loops on variables that are not included in the loop can be significantly harder. For instance, in phase IV of the LongWave, the role of \( L1 \), the Capital Decay loop, on Backlog is exactly the opposite from what is expected from a balancing loop. The structure of the model can be used to understand these effects, although the identification and explanation of the mechanisms involved require solid understanding of the model under inspection. Further research may be done in order to find the minimal conditions under which

\(^{17}\) Backlog is still influenced by the exogenous orders, which push the net rate of change of Backlog to slightly above zero.

\(^{18}\)MatLab sometimes switches the order of the eigenvalues in the system even if the change in the eigenvalues is minimal, so the code used had to compensate for this.

\(^{19}\)Except for those due to the different definition of an elasticity
this counterintuitive behaviour occurs.

Third, the overall elasticity gives the magnitude of the elasticity of a loop, but is unsuited for assessing the exact role of the loop. The solution proposed is to separate the contributions of the different parts of the eigenvalues and relate these to the relevant elasticities (Burak Güneralp, personal contribution). While this has been implemented for small models, it still is experimental and has yet to be implemented in a more generic form.

Finally, in my view, this form of analysis almost goes beyond the limitations set by its foundations. In order to fully understand and trust the results presented here, the analyst has to trace back the origins of these results. This is especially true in intervals where a complex pair of eigenvalues plays a significant role in determining the behaviour of the system or the state variable under inspection, since the method ignores the signs of the real and imaginary elasticities by using the overall elasticity. However, the loop influences are always based on the contributions of specific eigenvalues and the elasticities of loops to that eigenvalue. Using these two the analyst can backtrack and investigate the origins of the influence of a specific loop.

In contrast, the equivalent method proposed by AbdelGawad et al. (2005) never takes the final step of summing a loop’s elasticities weighted by the contributions of the eigenvalues. It stops at calculating the contributions of the eigenvalues and elasticities to all eigenvalues in the model.

7.3.5 Ford

Modifications To The Method

To perform the complete behavioural analysis, given that there are four phases of behaviour per cycle and 16 loops in the SILS (roughly) 16 equations would need to be rewritten and the model run performed 64 times. If we were to find shadow loops in one of the phases, all combinations of loops would have to be tested. At least, if we were to perform the complete version of the analysis. The time taken to perform this analysis would be disproportionate, so, we will focus on loops of interest based on the results from the elasticity analysis related to specific states.

The techniques at our disposal differ significantly from those that were available to Ford (1999). One of the critical differences between EEA and Ford’s behavioural approach is no longer present due to the advances in EEA. The ability to relate the results of EEA to the behaviour of a specific state allows us to directly compare the two methods. In his own analysis Ford (1999) compares the results of the system-wide variant of the Loop Eigenvalue Elasticity Analysis to the results of his own

---

20The method used is slightly different but relies on the same eigenvector-based reasoning (Saleh, 2002; AbdelGawad et al., 2005)
analysis. He selects Desired Production as his variable of interest based on the apparent match between the dominance shifts for Desired Production and the phases identified by Kampmann (1996a). Based on the fact that we only have the results for the EEA related to state variables and that we wish to compare the results of the two techniques as directly as possible, we will have to select a state variable of interest for which the intervals in which it displays a single atomic behaviour roughly match the phases used in the EEA.

Inspecting the three state variables, Backlog appears to be the most suited state variable, not only in the times of switches between the intervals, but also in the number of intervals. (Figure 7.44). Note that there is one extra interval in Phase IV. The analysis will take this as two different intervals, as per definition (Ford, 1999). The rapid switch between patterns around the transition from the previous cycle to Phase I\textsuperscript{21} will be taken as one change of behaviour pattern and the beginning of Phase I is seen as the beginning of the first interval to be analysed. It is doubtful whether the analyst would not be tempted to gloss over the rapid switch and would have just blindly taken Phase IV of the previous cycle and phase I of this cycle as one interval. However, this decision would not agree with the changes in the actual mechanisms driving the behaviour of the model.

Furthermore, in contrast to the analysis as presented in Ford (1999), in this analysis we will attempt to disable loops according to the method defined in Section 7.2.1. In contrast, Ford (1999) himself sets the independent variable (the source of the edge) to its steady state value in the dependent variable (the destination of the edge) to deactivate an edge. No explanation is given as to why this is considered a valid approach, while the difference between steady state values and current values is significant, especially in a model as non-linear as the Long Wave. Due to some unexpected results, the differences in method of deactivating edges will be tested in interval $c$.

One other caveat to the method is that, although the independent loop set offers a linearly independent subset of the loops, this does not mean that every loop has a unique edge that can be taken out to take out only that loop. However, reading the Directed Cycle Matrix (DCM) (Section A.3.4), we can find those edges that uniquely identify particular loops. One algorithm for finding the unique edges is summing the rows of the DCM and finding those edges that occur only once. Once these have been found, the loops that contain these edges can be removed from the set, as they can be eliminated without effecting other loops. The loops that remain are the loops that cannot be taken out without taking out at least one other loop. For these, we can find the edges that take out the minimal amount of other loops and take into account the effect of eliminating the additional loops.

\textsuperscript{21}The quick change is due to Relative Orders being less than zero for a very short while.
Figure 7.44: Loop Influences on different state variables in Phase IV of the Long-wave model. The roman numerals denote the different phases in this cycle as per the EEA, $a \ldots f$ denote the intervals according to Ford’s behavioural analysis.
Applying The Method

In accordance with the change in the method to deal with a larger model, possible candidate loops for dominance will be selected. Based on the results of the eigenvalue elasticities analysis related to the specific states, we will identify possible candidates for dominance for each individual interval in Table 7.6, with a maximum of three for each interval. If this analysis were based on blindly following Ford's method and the behaviour of the variable of interest, the first interval of analysis would be the interval displaying exponential decline after the maximum of Backlog. However, using the input from the eigenvalue elasticity analysis and the way of thinking in previous publications (Kampmann, 1996a; Ford, 1999; Sterman, 1985; Güneralp, 2005) we will select interval $a$ as the first interval of analysis, during which the Self-Ordering loop is active and supporting the growth of Backlog. The intermediate results of deactivating the edges are in Appendix E.

Table 7.6: Candidate Loops for Ford’s behavioural analysis.

<table>
<thead>
<tr>
<th>Interval</th>
<th>$t$ switch</th>
<th>Phase</th>
<th>Candidate Loops</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a$</td>
<td>81.15</td>
<td>I</td>
<td>$L14, L4, L7$</td>
</tr>
<tr>
<td>$b$</td>
<td>84.65</td>
<td>II</td>
<td>$L4, L8, L2$</td>
</tr>
<tr>
<td>$c$</td>
<td>92.05</td>
<td>III</td>
<td>$L14, L7, L8$</td>
</tr>
<tr>
<td>$d$</td>
<td>93.25</td>
<td>IV</td>
<td>$L4, L1$</td>
</tr>
<tr>
<td>$e$</td>
<td>100.55</td>
<td>IV</td>
<td>$L4, L1$</td>
</tr>
</tbody>
</table>

Table 7.7: The loops in the Long Wave ILS and their unique edges

<table>
<thead>
<tr>
<th>Loop</th>
<th>Unique Edge</th>
<th>From</th>
<th>To</th>
</tr>
</thead>
<tbody>
<tr>
<td>$L1$</td>
<td>Depreciation</td>
<td>Capital</td>
<td></td>
</tr>
<tr>
<td>$L2$</td>
<td>Acquisitions</td>
<td>Supply</td>
<td></td>
</tr>
<tr>
<td>$L3$</td>
<td>None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$L4$</td>
<td>None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$L5$</td>
<td>None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$L6$</td>
<td>Capacity</td>
<td>Capacity Utilization</td>
<td></td>
</tr>
<tr>
<td>$L7$</td>
<td>None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$L8$</td>
<td>None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$L9$</td>
<td>Supply</td>
<td>Supply Adjustment</td>
<td></td>
</tr>
<tr>
<td>$L10$</td>
<td>Depreciation</td>
<td>Relative Orders</td>
<td></td>
</tr>
<tr>
<td>$L11$</td>
<td>Backlog</td>
<td>Desired Supply</td>
<td></td>
</tr>
<tr>
<td>$L12$</td>
<td>Depreciation</td>
<td>Desired Orders</td>
<td></td>
</tr>
<tr>
<td>$L13$</td>
<td>Capital</td>
<td>Capital Adjustment</td>
<td></td>
</tr>
<tr>
<td>$L14$</td>
<td>Desired Production</td>
<td>Desired Capital</td>
<td></td>
</tr>
<tr>
<td>$L15$</td>
<td>Production</td>
<td>Desired Supply</td>
<td></td>
</tr>
<tr>
<td>$L16$</td>
<td>Depreciation</td>
<td>Desired Supply</td>
<td></td>
</tr>
</tbody>
</table>

With regard to Interval $a$, $L14$ has been selected since it is the overall dominant
loop in the system. Since $L4$ is attributed a large elasticity in the beginning of the phase, we will investigate its influence. $L7$ is selected as a candidate loop since it has a large imaginary elasticity. $L7$ cannot be taken out without eliminating at least one extra loop. The edge leading from Depreciation to Capital Orders eliminates only $L7$ and $L8$ however. This edge will be deactivated, but to check the results of eliminating both loops, we will take out the combination of $L8$ and $L9$ as well, using one edge; Capital Orders to Capital Orders Supply. $L4$ is considered a candidate given its strong role in the beginning of the interval, note that this also takes out $L15$.

In interval $b$, the candidate loops are $L4$, $L8$ and $L2$. According to the EEA, $L4$ should be the most influential loop in the beginning of this interval, $L8$ in the second part of the interval. $L2$ is seen as the loop which has the largest influence opposite to the current behaviour of Backlog. Since Relative Orders is at its maximum in this phase, we can safely deactivate $L8$ and $L4$ without influencing the gain of $L9$ or $L15$ respectively (Section A.3.4). Consequently, for eliminating $L8$ the edge from Capital Orders to Capital Orders Supply is deactivated and for $L4$ this is the edge from Desired Production to Capacity Utilization respectively.

Interval $c$ should be dominated by $L14$ again. It is expected to dominate over the entire interval, but it is questionable if this is true just for this interval or sooner, since the previous analyses attributed dominance to this loop before the start of the interval. $L7$ and $L8$ probably support the current behaviour of Backlog, so it might be that these share dominance or form a set of shadow loops together with $L14$.

Within both interval $d$ and $e$, $L4$ and $L1$, are the only loops that the elasticity analysis attributes a significant influence to. Hence, we will check for dominance for just these two loops.

$L14$ is deactivated by fixing the representation of Desired Production in Desired Capital to its value in the beginning of the interval. This does not change the behaviour of the interval immediately. However, it does have a strong effect on the behaviour of the variable and after about 2 time units, the behaviour of Backlog is nearly linear. In the original analysis by Ford, the same edge is deactivated, but instead of fixing the representation of Desired Production in Desired Capital to the value it has at the beginning of the interval, the representation is set to its steady state gain (1.8). Ford attributes dominance to $L14$ since for this particular method of deactivating an edge, the behavioural pattern of Backlog changes. In fact, setting the representation of Desired Production to any value larger than approximately 1 immediately changes the behavioural pattern\textsuperscript{22}.

\textsuperscript{22}The sensitivity to this value is due to the low value of Depreciation, which causes Relative Orders to reach its maximum if Desired Orders is set to 1.8. A value of Desired Orders this high sets the gain of every loop leading through Relative Orders to 0.
Table 7.8: Combinations of loops taken out in interval $a$ of the analysis of the Long Wave Model

<table>
<thead>
<tr>
<th>Combination</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$t = 81.15$</td>
</tr>
<tr>
<td>$L7 &amp; L8$</td>
<td>Slightly faster growth</td>
</tr>
<tr>
<td>$L14$</td>
<td>Nearly linear after two time units</td>
</tr>
<tr>
<td>$L4$</td>
<td>Slightly faster growth, nearly linear around 84</td>
</tr>
<tr>
<td>$L4, L14$</td>
<td>Nearly linear after two time units</td>
</tr>
<tr>
<td>$L7 &amp; L8, L14$</td>
<td>Nearly linear after two time units</td>
</tr>
<tr>
<td>$L7 &amp; L8, L4$</td>
<td>Slightly faster growth, balancing after 83.9</td>
</tr>
<tr>
<td>$L7 &amp; L8, L14, L4$</td>
<td>Nearly linear after two time units</td>
</tr>
</tbody>
</table>

If, in the same interval, $L7$ and $L8$ are eliminated, it appears that these two loops do not dominate, in spite of the fact that they are simultaneously taken out. Additionally, as a check on the role of $L7$ and $L8$, we will eliminate $L8$ and $L9$ by de-activating the edge from Capital Orders to Capital Orders Supply. The expectation is that will result in a similar effect to that when $L7$ and $L8$ are eliminated. If the behaviour pattern changes, it might be that either $L8$ or $L9$ dominates. However, this is not the case. To conclude, growth is weaker if $L7$ and $L8$ are taken out, which confirms their supporting role, but they do not dominate this interval.

With regard to $L4$, this loop again does not dominate. Eliminating this loop confirms its expected restraining role, as growth with the loop is slightly larger. The effect of $L15$ is negligible, it’s elasticities are close to zero and eliminating the loop results in almost no effect on the behaviour of the variable of interest.

Given the lack of a loop that is clearly dominant at this moment in the interval, the analysis is extended in two ways. First, to check for the existence of a shadow pair, we will eliminate combinations of candidate loops. Second, the entire analysis, including the check for the existence of shadow pairs, will be performed again, but now approximately halfway through the interval. The results of both of these activities are summarised in Table 7.8. The effects of secondary loops are not checked at this stage of the analysis.

To summarise, in the beginning of interval $a$, no single loop, nor a combination of the candidate loops is so strong that eliminating this loop results in a change in atomic behaviour pattern for Backlog. The behaviour pattern might be changed by eliminating the edges leading directly to production and those leading to either
Capital Orders or Backlog Orders, but this directly sets the rates influencing Backlog to constant. The influential role of $L14$ is confirmed by these results though; the change in behaviour pattern is not immediate, but visible.

In contrast to the beginning of the interval, the behaviour pattern of Backlog immediately changes to balancing if $L14$ is eliminated by deactivating the edge leading from Desired Production to Desired Capital according to the standard method. Accordingly, if the analysis is performed at $t = 82.8$, it leads to the conclusion that there is a clear dominant loop in the model, $L14$, the Self-Ordering loop. Every other analysis also identifies this loop as dominant.

As known from the previous analyses, during Phase 2 and most of interval $b$, Relative Orders is at its maximum. Hence, all loops that contain that variable can be considered as inactive, effectively eliminating $L9$ through $L16$. Consequently, by deactivating the edge from Desired Production to Capacity Utilization this eliminates only $L4$. Similarly, deactivating the edge from Capital Orders to Capital Orders Supply, only eliminates $L8$. $L2$ already has a unique edge associated with it, the edge leading from Acquisitions to Supply.

Of the candidate loops, $L8$ and $L2$ do not appear to be dominant in interval $b$. In spite of the large role the EEA attributes to $L8$, eliminating it does not trigger a switch in behaviour pattern. The result of the current analysis does agree with the expected supporting role of the loop; without the loop, backlog balances less rapidly. Eliminating $L2$ results in Backlog approaching the equilibrium far more rapidly, consistent with a) its strong influence later in the interval b) its effect opposite to the balancing of Backlog as identified by the EEA.

In contrast, deactivating the edge from Desired Production to Capacity Utilization immediately changes the behaviour of Backlog from balancing to exponential growth. Consequently, $L4$, the Overtime Loop, surfaces as a dominant loop. The analysis concludes that $L4$ is the loop responsible for the balancing behaviour of Backlog. This partially agrees with the conclusion drawn from the EEA, since that analysis suggests a pair of shadow loops or shared dominance for $L4$ and $L8$.

With regard to interval $c$, we expect $L14$, the Self-Ordering Loop to be dominant in this phase, as it was in interval $a$. Yet, using the standard method, we cannot draw the conclusion that $L14$ is dominant. So, to find out whether the method or timing of deactivation of the unique edge play a strong role in determining the results of the method, $L14$ will be eliminated at different times and using different methods (Table 7.9). The edge will be deactivated just before the start of interval $c$, at the exact start and halfway through the interval.

Setting the unique edge for $L14$ to its steady state gain, triggers a change in behavioural pattern for all moments at which $L14$ is eliminated, although the change
Table 7.9: The results of eliminating $L_{14}$ at different times for different methods. The first point in time at which the loops is taken out is slightly before the switch of behaviour patterns. The second is at the switch of behaviour patterns. The third is halfway through the interval.

<table>
<thead>
<tr>
<th>Time</th>
<th>Ford</th>
<th>Standard</th>
</tr>
</thead>
<tbody>
<tr>
<td>91.5</td>
<td>Behaviour pattern changes</td>
<td>Behaviour pattern does not change</td>
</tr>
<tr>
<td>92.05</td>
<td>Behaviour pattern changes</td>
<td>Behaviour pattern does not change</td>
</tr>
<tr>
<td>93</td>
<td>Behaviour pattern changes, hardly visible</td>
<td>Behaviour pattern changes</td>
</tr>
</tbody>
</table>

is almost negligible for the $t = 93$. Fixing the edge to its value at the moment of analysis only triggers a switch of behaviour pattern for $t = 93$, but the response of the model to the elimination of the loop is always visible. The strength of the standard method of deactivating edges is that it is dependent on the current state of the model. However, the relatively minor difference between the original model and the model with the edge deactivated makes it less likely that this form of the method finds a dominant loop. In contrast, since the steady state method uses a larger difference between the original model run and the model with the loop taken out, it will more quickly attribute dominance. It is doubtful whether this is an advantage though. For instance, setting the unique edge associated with $L_{14}$ to its steady state gain just before beginning the interval $c$ will have the effect of moving Relative Orders from near its maximum $^{23}$ to zero. This effectively eliminates every loop that includes the Relative Orders variable. In addition, using this method we would also find $L_{14}$ to be dominant around $t = 86.5$, before it is active again.

Deactivating the edge associated with $L_7$ and $L_8$ for the same points in time as $L_{14}$ has no significant effect in this interval. Eliminating $L_8$ and $L_9$ causes no change in behavioural pattern either.

The removal of $L_1$ in interval $d$ shows no discernible difference between the original model run and the behaviour with this loop eliminated. Deactivating $L_{14}$ using the standard method and the same edge as before triggers an immediate change in behaviour pattern. Consequently, $L_4$ is identified as a dominant loop in interval $d$.

In contrast to what is expected based on the outcome of the elasticity analysis, repeating the dominance analysis from interval $d$ in interval $e$ provides different results. Both the elimination of $L_4$ and $L_2$ result in increased growth, but neither the removal of the individual loops or the pair results in an immediate switch of behavioural pattern. Consequently, in interval $e$, neither $L_4$ or $L_1$ dominates, individually or in a shadow pair.

Extending the search for dominant loops beyond what the eigenvalue elasticity

\[ ^{23}5.9182, \text{ at the beginning of the interval} \]
analysis suggests, we see that the continued increase in Backlog is a consequence of the slow decrease in Production. Due to the decay of Capital, Capacity decreases, while the increase in Capacity Utilization cannot keep up, resulting in the decrease in production. Eliminating the Capital Decay loop, however, \( L1 \), does not result in a change in behaviour pattern, setting the edge from Depreciation to Capital to its value at the beginning of the interval results in almost similar behaviour of Capital during the entire interval. Deactivating the edge leading from Capacity to Production does trigger a switch in behaviour pattern, as does deactivating the edge from Production to Backlog\(^24\). The first deactivated edge is included in \( L3 \) and \( L5 \), the second in \( L4 \) and \( L5 \). Setting the edge shared between \( L3 \) and \( L6 \) to its value at the beginning of the interval makes no noticeable difference in the behaviour of Backlog. Consequently we attribute the change in behaviour pattern to \( L5 \), however, given the above this does not contradict the role of the Capital Decay loop, \( L1 \), as identified in the EEA.

**Results**

One assumption underlies Ford’s behavioural approach. Given the method’s focus on intervals displaying a single behavioural pattern, it implicitly assumes that a change in dominance is always accompanied by a change in behavioural pattern and vice versa. However, as is evident from the results of the analysis of the long-wave model, this does not need to be the case; changes in driving structure do not need to cause a change in behaviour pattern. Before the model enters interval \( a \), our variable of interest displays the same behaviour pattern as in interval \( a \). Blindly applying the method’s reasoning the analyst would never identify two separate intervals. To find these, either input from a second method of analysis or knowledge of the system is needed.

Another consequence of this assumption is that it does not take into account the possibility that a switch in dominance occurs before the switch of behaviour pattern. For instance, \( L14 \) can be considered dominant before the switch of behaviour pattern. Again, if the analyst would blindly apply Ford’s method, the earlier switch in dominance would not be detected.

To summarise, this analysis shows that a) the model can switch dominance without switching behaviour pattern b) a switch in dominance does not need to occur at the same time as a switch in behaviour pattern. This falsifies Ford’s fundamental assumption.

An additional problem is illustrated by the analysis of interval \( a \), in the assessment of the role \( L14 \). The objective of the analysis is to eliminate the contribution of one

\(^{24}\text{After inspecting the rates influencing Backlog, this is a trivial result though.}\)
specific loop, $L14$, to the behaviour of the model. However, as seen in the analysis of interval $a$, by setting the representation of Desired Production in Desired Capital to 1.8 when Capital is near its minimum, Relative Orders is set to its maximum. This means that every loop containing the variable Relative Orders is, at that moment in time, eliminated. The method is hindered by the fact that the structural independence of the loops in the ILS does not need to be accompanied by a corresponding independence in the equations of the model.

And, finally, a section of Ford’s procedure can be supported by structural analysis. Reading the Directed Cycle Matrix (DCM), those edges that uniquely identify particular loops can be identified. One algorithm for finding the unique edges is summing the rows of the DCM and finding those edges that occur only once. Once these have been found, the loops that contain these edges can be removed from the set, as they can be eliminated without affecting other loops. The loops that remain are the loops that cannot be taken out without taking out at least one other loop by deactivating the unique edge. The remaining loops that do not have unique edges can be eliminated based on what other loops will be deactivated by removing a particular edge and the expected influence of those loops for the model in its current state. Again, this requires input from a second method of analysis or knowledge of the system.

7.4 Conclusions

7.4.1 Framework

Although this version of the analysis technique used a perturbation method for determining the link eigenvalue elasticities, it showed that the eigenvalue elasticity analysis can be automated using the framework presented in Chapter 6.

In contrast to previous forms of analyses, the ability to adjust the granularity of analysis for specific analyses opens the possibility of showing detailed, short term effects in the analysis of highly dynamic intervals of a model and speeding up the analysis of long, relatively stable intervals. For instance, the necessity for rescaling elasticities and the discontinuous elasticities around bifurcation and convergence points were found by allowing the analyst to adjust the granularity of the analysis and focus on a specific interval of interest.

Furthermore, the numerical calculation of edge gains circumvents the need for manually writing out gain equations. Obtaining edge elasticities by perturbing the edges and observing the response in the eigenvalues provided some difficulties, but no significant, unexpected differences between results of this analysis and pre-

---

25 This makes Ford’s behavioural approach at least partially structural.
vious analyses have been found. The effort involved in applying several versions of eigenvalue elasticity analysis has been greatly reduced. However, there still is a clear need for implementing an automated version of the analytical approach, where most of the difficulty lies with the structural analysis involved.

Consequently, the eigenvalue elasticity analysis in its automated form is an order of magnitude faster than Ford. Assuming that the model is in analysable form, the analysis itself is only a matter of running the appropriate functions. Thirdly, replacement of the finite difference method by the analytical version of calculating the individual edge elasticities in the model described by Guneralp (Equation 7.3) is desirable, possible and feasible. A fully automated version of this technique only leaves the modeller with the task of understanding the method and being able to interpret the results.

In my view, applying this framework emphasises again that the difficulty of automating analyses such as eigenvalue elasticity analysis lies not with the algorithms involved, but with the way different parts - model, solver and analysis tools - exchange information. If the model is built in a way that provides access to the necessary information, we end up with a much more powerful and easier to modify tool.

### 7.4.2 Eigenvalue Elasticity Analysis

#### General

First of all, large parts of eigenvalue elasticity analysis have been automated. However, it takes practice and effort to be able to analyse the results from all forms of eigenvalue elasticity analysis, especially when models become larger and display less intuitive behaviour.

Secondly, the analyses demonstrated that even if a model shows smooth behaviour for its eigenvalues, the elasticities are discontinuous on points where eigenvalues merge into conjugate pairs, or when a pair bifurcates into two real eigenvalues. This necessitates dividing the analysis into particular phases according to the eigenvalues and makes the validity of the results of the analysis around these singular points questionable. In addition, the graphs are kept readable by rescaling them to values between $-1$ and $1$.

The three versions of EEA used in analysing the models each have their own particular advantages and disadvantages. Of these, the strong points in one method can compensate for the weak points in the other. For instance, we can use the loop based elasticity analysis as a means of explaining the influence of a specific parameter in more detail. The parameter based variant compensates for the loop based variant's lack of relation to leverage points and the understanding and language of
the client.

In contrast to the loop based variant of EEA, the parameter based eigenvalue elasticity analysis aims to clarify the specific role of the parameters in the model during the model run. In its current form, the method provides insight into the direction and magnitude of the influence of the parameters in the model on the eigenvalues. Its results clearly show the effects of perturbing the parameters on the exponential envelope and the frequency of oscillations. However, the stated goal of the analyses performed is relating structure to behaviour, but this particular form fails to do exactly that. The direction of influence is known for each model parameter, but what we are interested in is the mechanisms or pathways involved in obtaining that influence. Ideally, the analyst would be able to combine parameter based elasticity analysis with the structure of the model, showing not only what influence the parameters have, but also how this influence is obtained.

With regard to the version of the elasticity analysis as defined in Kampmann (1996a) and extended in Kampmann and Oliva (2005), its weakest point is the selection of an almost arbitrary eigenvalue as the dominant one (Kampmann and Oliva, 2005). This detracts from the ease of interpretation of this form of analysis. This is especially true in more complex, larger models where the behaviour of the state variables is not linked as clearly with the eigenvalues and related EEA as the analyst would like to see. The results of the method can be made more accessible, for instance by extending the method with the algorithms as developed by Güneralp (2005).

However, the investigation into the variant of eigenvalue elasticity analysis that relates the elasticities to specific states in the model shows that there remain issues with these methods. Namely:

- There are qualitative differences in outcome depending on which elasticity measure the analyst uses. The measure chosen by Güneralp (2005) seems highly doubtful since both the inspection of the actual behaviour of the variables in the model and elasticity analysis based on parameters seem to contradict the results obtained using the measure presented in his paper.

- The interpretation of the role of loops on variables that are not included in the loop can be significantly harder as this can display counterintuitive behaviour. The structure of the model can be used to understand these effects, although the identification and explanation of the mechanisms involved require understanding of the model under inspection. Further research may be done in order to find the minimal conditions under which this behaviour occurs.

- In order to fully understand and trust the results obtained, the analyst has to trace back the origins of these results. This is especially true in intervals
where a complex pair of eigenvalues plays a significant role in determining the behaviour of the system or the state variable under inspection, since the method ignores the signs of the real and imaginary elasticities by using the overall elasticity, which only gives the magnitude of the elasticity a of loop.

In addition, although this seems a trivial observation if the algorithms used in the variant of the method that relates elasticities to the behaviour of specific states are inspected, the elasticity analysis is the same for different state variables in the system until the point at which the elasticities are weighted according the eigenvalue contributions and summed together. Hence, the contributions of the eigenvalues to the behaviour of specific states will indicate exactly where the results of the analysis will differ for different states. Similarly, the results of the analysis will be comparable to those of the system wide analysis if the eigenvalue with the highest contribution to the state variable of interest is the same one as selected in the system-wide variant of the analysis.

In conclusion, it seems that of the analysed methods, eigenvalue elasticity analysis is the most promising method. However, there still are difficulties facing the method. The most important of these is the challenge the analyst faces in mapping the outcome of an analysis back to what the model represents and and relating the results to the system story. Any strong method is only as useful as the analyst is insightful himself, unless its results can be effectively used at the appropriate stages in the modelling process and translated to the client's language. Furthermore there still are several analytical challenges facing the method. Namely:

- One of the more important results of the examination of EEA, is the identification of moments in the model where the matrix of right eigenvectors is singular. In the current form of the analysis, it is possible to identify moments in time during the model run when this is the case. This could be extended by defining that subspace in state space where the matrix of right eigenvectors is singular before running the model.

- The ease of interpretation of the variant relating elasticities to the behaviour of specific states can be extended by separating the contributions of the real and imaginary parts of a complex eigenvalue to the change in slope in the states and combining this with the real and imaginary elasticities (Appendix C.1).

- The use of the SILS in this form of the method implicitly assumes that the edges in the system are always active. However, it is possible that a loop included in the SILS set becomes inactive at some time during a simulation while another loop, excluded from the SILS, yet forming an alternative path,
remains active. This poses a problem if the original SILS is used throughout the analysis of the eigenvalue elasticities. By checking for such a condition and re-executing a structural analysis when it occurs (Güneralp and Gertner, 2006), this problem can be resolved. Current developments as mentioned in Goncalves et al. (2000) are making progress in this area, since these aim to abandon the concept of an ILS.

- Finally, one other point is the fact that linearisation still is the basis of the method. So far, there is no formal measure which indicates in how far the linearisation is appropriate. Intuitively, the analyst would expect this to be a problem where edge gains and eigenvalues are changing rapidly.

7.4.3 Ford

To facilitate the method, the analyst can use the Directed Cycle Matrix (DCM) a section of to find those edges that uniquely identify particular loops can be identified. One algorithm for finding the unique edges is summing the rows of the DCM and finding those edges that occur only once. Once these have been found, the loops that contain these edges can be removed from the set, as they can be eliminated without affecting other loops. The loops that remain are the loops that cannot be taken out without eliminating at least one other loop by deactivating the unique edge. The remaining loops that do not have unique edges can be eliminated based on what other loops will be deactivated by removing a particular edge and the expected influence of those loops for the model in its current state.

In spite of the reduced time the method now takes, the behavioural approach is still far more time consuming than an automated version of the eigenvalue elasticity analysis. In fact, in any model of reasonable size the method is so time consuming that the analyst must limit himself to assessing the role of a small, specifically selected subset of the loops in the system. For this selection he has to base himself on either the results of a different method of analysis or his own own knowledge of the system.

In addition to the time consuming nature of the method there are several weak spots in the methodology. First, the usage of the concept of shadow loops is very doubtful. The suspension of disbelief required for comparing the original model to the model with one loop taken out is small enough to still accept results. However, a model with two loops taken out is so different from the original model that its is questionable whether the conclusions based on the modified model can be validly translated back to the original.

If desirable, the same algorithm can be repeated but now looking for edges that remove 2 loops, after the loops that do have an unique edge have already been removed from the set.
Second, the restriction of the analysis to separate time intervals limits its explanatory power for an oscillating model. Its conclusions are confined to time intervals displaying one atomic behaviour pattern, causing effects spanning more than one interval to be ignored. In the case of a model with dampened oscillation, where one loop is responsible for the speed of dampening, Fords approach will have difficulty in distinguishing this effect, since it is restricted to time intervals in which the dampening cannot show.

Third, one implicit assumption underlies Ford's behavioural approach. Given the method's focus on intervals displaying a single behavioural pattern, it implicitly assumes that a change in dominance is always accompanied by a change in behavioural pattern and vice versa. However, this does not need to be the case; changes in driving structure do not need to cause a change in behaviour pattern. Another consequence of this assumption is that it does not take into account the possibility that a switch in dominance occurs before the switch of behaviour pattern. Blindly applying the method's reasoning the analyst would never identify two separate intervals or a switch in dominance during an interval of analysis. To find these, either input from a second method of analysis or knowledge of the system is needed.

Fourth, although the objective of the analysis is to eliminate the contribution of one specific loop to the behaviour of the model. The method is hindered by the fact that the structural independence of the loops in the ILS does not need to be accompanied by a corresponding independence in the equations of the model. If the method of deactivating an edge results in a larger 'shock' to the system it is more likely that the influences of several loops are affected. In certain cases this will make it impossible to analyse the effects of individual loops separately.

In spite of the effort involved and the above issues, checking the behaviour of the modified models with the behaviour expected due to these modifications, provides a structured method for assessing the role of the loops in the system and obtaining more insight into the causes of model behaviour. However, contrary to extreme conditions its results are not so predictable that we can use them for verification; the exact role this method of analysis could play in the modelling process is still unclear.

In conclusion, it seems that Ford's behavioural approach is not as suitable for the discovery of influential structure, but shows more potential in checking the influence of pre-selected candidate loops. Methodologically this makes the behavioural approach a weaker tool than a form of analysis that requires no knowledge of the relationship between structure and behaviour, such as eigenvalue elasticity analysis. But it can be useful in support of EEA and mitigate exactly EEA's weakest point: the abstract nature of the method.
Chapter 8

Conclusions

The aim of this research is to assess and improve the applicability of formal methods of System Dynamics (S.D.) model analysis. The methods for model analysis aim to relate model structure to behaviour by finding the specific subset of structure that is predominantly responsible for behaviour, or finding a quantitative measure for the (relative) importance of an element of structure. In order to attain this goal the following three research questions were posed:

1. **What formal methods of System Dynamics model analysis are available and which of these are most promising?**

2. **What are the strengths and weaknesses of these methods in terms of**
   a) the fundamentals of the method,
   b) results provided,
   c) the range of models they can be applied to,
   d) the resources required to perform the methods?

3. **How can we improve the applicability of these methods?**

Initially the aim was to perform the existing analyses on one or two case studies. However, this turned out to be more problematic than anticipated. The selected methods are still in their infancy and the consistent application of the methods to any real world model faces many difficulties. Consequently, the applicability of the methods had to be improved before a consistent analysis was possible, changing the order of the research questions and making the assessment question the final question of this thesis.
The answers to the research questions will be presented and discussed in this chapter, with one question discussed in each section. The third section, answering the final research question also contains a comparative analysis of the most promising methods for model analysis.

8.1 Promising Methods Available

The three methods currently available for System Dynamics model analysis include:

- The Pathway Participation Method
- Ford’s behavioural approach
- Eigenvalue Elasticity Analysis (EEA)

Of these three methods for relating structure to behaviour, the pathway participation method is considered least useful. Its form of analysis only allows for the selection of one dominant loop, not allowing for situations where multiple loops with small contributions cause the same effect. Secondly, the inability of the algorithm to deal with oscillations results in it being less able to analyse models where oscillation is caused by complex pairs of eigenvalues. All in all, this was reason enough to not include this method in the latter stages of this research. Consequently, Ford’s behavioural approach and eigenvalue elasticity analysis are identified as the most promising methods for formal model analysis.

In addition, structural analysis based on a graph-theory view of the model is proved a particularly useful technique. In this research it was used only in the sense that it forms a necessary aid for the implementation of other analysis methods.

8.1.1 Ford’s behavioural approach

Ford’s method of taking out loops during specific intervals of behaviour of a variable of interest provides the analyst with a systematic and relatively easy method of assessing loop dominance. Essentially, the method is an extended form of an extreme condition test, where the extreme condition is the sudden elimination of a specific loop. The difference between the model run without the loop and the original model is used to assess the effect of the loop on the variable and interval for which the analysis is undertaken. The intervals are determined by the behaviour pattern exhibited by the selected variable of interest. In order to implement this, we need a reduced set of loops in which each loop has at least one unique edge, enabling the analyst to take the loop out without there being consequences for the other loops.
8.1.2 Eigenvalue Elasticity Analysis

The Eigenvalue Elasticity Analysis uses the fact that the behaviour of a model is determined by eigenvalues of the linearised system. By determining how loops and parameters influence the eigenvalues, the method links model structure or parameters to behaviour. This is made feasible by linearising the model at selected timesteps in the model run; the entire analysis relies on the linearisation of the model around given points in the model run.

The concept central to the analysis is the elasticity of an eigenvalue to an edge, loop or parameter in the model. That is, the analyst measures the response of the eigenvalue to a change in the loop gain or parameter value to assess its role in the current dynamics of the model. Using the eigenvectors of the system, these elasticities can be related to the behaviour of a specific variable. Hence, the results of this method can be related either to the system as a whole, or the behaviour of a specific state variable, which gives the analyst several variants of the method.

Loop based Eigenvalue Elasticity Analysis

This variant of the method measures the influence of the individual loops in the system on the behaviour of the system. It uses a subset of the loops present in the model, the independent loopset, to determine eigenvalue elasticities. The properties of this loop set allows for a computation of individual loop elasticities. The conclusions drawn are only valid for that specific loop set, but since the set involves every edge in the model that is involved in at least one loop, it is considered representative of the model.

Parameter based Eigenvalue Elasticity Analysis

In contrast to the loop elasticity analysis, this method provides us with insight into the specific role of parameters during the model run. The advantage of this approach is that parameters are often the leverage points for the problem owner / client. This shows us what effect each leverage point has on the eigenvalues during the course of the model run.

Relating the elasticities to specific variables

The system-wide variant of the method gives the analyst no insight into how the different pieces of structure relate to the behaviour of a specific variable. As described by Güneralp (2005), we can relate the elasticities to a variable of interest via the contributions of the eigenvalues to the behaviour of that eigenvalue.

This adds to the system-wide variant of the method in several ways. First of all, if the analysis does not focus on a specific variable, the largest eigenvalue is usu-
ally selected as the dominant one and the loops or parameters that have the most influence on that eigenvalue are considered dominant. In this method, the analyst no longer selects an arbitrary dominant eigenvalue, but uses weights to relate the elasticities to the eigenvalues. This also solves the lack of relation of the results to a specific state variable.

8.2 Improvements In The Methods

Both selected methods required significant improvement. Ford’s behavioural approach has aspects that require sharper definition, while eigenvalue elasticity analysis still faces several problems with regards to implementation and the singularity of the matrix of right eigenvectors. In order to perform all variants of the eigenvalue elasticity analysis, it was necessary to design and implement a framework for model analysis. This allowed for a consistent and flexible execution of at least the eigenvalue elasticity analysis methods.

8.2.1 The Framework

The current form of representation of models in SD software limits the model analysis that can be performed. Automated methods currently in existence run into several obstacles, so it became necessary to design a framework to consistently compare the various methods even on small models. The framework uses a somewhat different perspective on a System Dynamics model than that usually taken. It relies on two central concepts to perform its tasks:

- The three parts of the framework, namely the numerical solver, the model and the analysis functions, are completely separate from one another. The information exchanged between them is the only connecting element.

- The internal structure of the model is built with the graph-theory perspective in mind from the outset. This means that the analyst can view the model as if it were a graph with nodes and edges. This simplifies structural analysis and hence facilitates further model analysis.

The framework has enabled the consistent and flexible application of the various model analysis methods on several selected small models. It allows for presentation of the results of the methods in a semi-continuous manner, tracing the influence of the different elements of structure over time. Only the variant of eigenvalue elasticity analysis that relates eigenvalue elasticities to state variables presented significant problems, but these were inherent to the method. The framework itself and the communication between the different sections performed well.
The prototype of the framework provided three concrete advantages in this research. First of all, it turned out to be a very flexible tool which enabled an easy switch between models and methods of analysis. The consistent approach to analysis made the switching between, for instance, the parameter- and loop-based variants of the eigenvalue elasticity analysis a matter of changing a single line of code. Secondly, the separation of integration and analysis provides the analyst with a tool in which he can adjust the interval that is being analysed and the granularity\(^1\) at which this analysis is performed. Over intervals where the eigenvalues are changing rapidly this is a particularly desirable feature. This feature was used to find the discontinuous eigenvalue elasticities around bifurcation and merge points of complex pairs of eigenvalues.

And, thirdly, the framework enabled the automation of parts of eigenvalue elasticity analysis which had not previously been automated. The method proposed by Güneralp (2005) has been generalised almost completely, while the version originally published required code to be written for each individual model.

The main point finding in implementing this framework was that a large part of the difficulty of automating the algorithms for analysis lies not with the algorithms themselves, but with the way different parts of the process of analysis communicate.

The major disadvantage of the prototype implemented is that we lose the current model base and knowledge of modelling software. Translating models is not easily automated and disproportionately labour intensive to do by hand. Using the framework requires working knowledge of Java and MatLab, which not all System Dynamics modellers have. It should be noted that all versions of eigenvalue elasticity analysis could be automated, but Ford still had to be done semi-manually.

### 8.2.2 Ford's Behavioural Approach

In order to apply Ford consistently and rapidly, it was necessary to adjust it at several points. This was so that the results were comparable between different loops and the time required for the analysis reduced. The following modifications were made:

- The same method for switching off loops is used for different loops and different intervals.

- The definition for a pair of shadow loops is extended. These can now only be said to exist if no loop is found to be dominant. Consequently, the search for pairs of shadow loops dominating behaviour has been restricted to those

\(^1\)Greater times between consecutive points at which the model is linearised and analysed means a lower granularity.
8.2. IMPROVEMENTS IN THE METHODS

intervals where no dominant loop is found for a certain interval. This reduces the time required for the analysis.

• In larger models, the analyst can use the Directed Cycle Matrix (DCM) to find those edges that uniquely identify particular loops. This supports and partly formalises the section of Ford's behavioural approach that uses structural analysis.

8.2.3 Eigenvalue Elasticity Analysis

As stated before, the framework made it possible to automate a large part of the Güneralp (2005) algorithms. Although the implementation currently still uses a simple finite difference method for calculating eigenvalue elasticities, the communication between the model and the analysis tools and several steps of the methodology have been automated. Furthermore, the calculation of the contribution of a specific eigenvalue to the behaviour of a specific state has been modified into a variant which can be scaled to $n$-th order models without modification.

Different elasticity measures can produce qualitatively different results. The definition of eigenvalue elasticities as used by Güneralp (2005) was shown not to relate the structure of the model to the behaviour in a satisfactory way. Consequently, an alternative measure was proposed whose results proved to be more in line with the actual mechanisms of the model.

The analyses performed also showed that although the models show continuous, relatively smooth behaviour in their eigenvalues, there are singularities on points where separate eigenvalues merge into conjugate pairs, or when a pair bifurcates into two separate eigenvalues. The fact that eigenvalue elasticity no longer performs well if the matrix of right eigenvectors is singular was already known (Kampmann and Oliva, 2005), however, this research provides the modeller with the points at which these singularities are to be expected.

In spite of the presence of these singularities, the eigenvalue elasticity analysis can still be performed by dividing up the analysis into open intervals. The intervals are determined according to the behaviour of the eigenvalues; the boundaries of the intervals are defined by the points of singularity. Rescaling the elasticities keeps the results of the analysis interpretable, even in the presence of singularities. However, the validity of the results of the analysis in the region of these points is still questionable.
8.3 Assessment of the Methods

One weakness of all the methods is the ever-present subjectivity in modelling, the effect of which is amplified in model behavioural analysis. A different structure for mathematically equal models\(^2\) results in a different output from the analysis. That is, all loop-based methods are dependent on how the analyst has built the model. Consequently, although there is no such thing as the correct model for a problem, it helps if the structure of the model reflects the structure of the real world system.

In the assessment of the methods, an item starting with a ‘+’ indicates a strength of the method in a particular area, while a ‘−’ denotes a weakness.

8.3.1 Ford’s Behavioural Approach

In spite of the improvements made to the method, the remaining procedure is still very time consuming. Further, this research also found several previously unidentified problems fundamental to the method.

Overall, the exercise of systematically taking out loops does provide qualitative insight about the role of the loop taken out. The qualitative insight into the role of a specific loop provided by the method is a valuable discovery, but taking the analysis beyond the abilities of the method is doubtful practice. Whether the method can be used to make judgements about which loop is dominant is also doubtful, given the problems with the fundamentals of the method. Furthermore, it may be that the insights generated by performing Ford have no added value for someone who is intimately familiar with the model in question.

Fundamentals of the Method

- The consistency of applying the method needed to be adjusted, but even when consistently shutting off loops, the validity of the concept of shadow loops remains doubtful.

- Given the method’s focus on intervals displaying a single behavioural pattern, it implicitly assumes that a change in dominance is always accompanied by a change in behavioural pattern and vice versa. However, this does need not the case; changes in driving structure do not need to cause a change in behaviour pattern.

- The objective of the analysis is to eliminate the contribution of one specific loop to the behaviour of the model. However, the method is marred by the fact that the structural independence of the loops in the ILS is not necessarily accompanied by a corresponding independence in the equations of the models can be mapped to each other via the substitution of auxiliary variables.
model. In certain cases this will make it impossible to analyse the effects of individual loops separately.

The Results Provided

+ The results are easily interpreted. A switch in behavioural pattern can be detected by visual inspection or by calculating the derivative and curvature of the variable of interest. If the elimination of structure results in the sign of the behavioural pattern changing, it is clearly dominant.

+ The process of eliminating loops does provide insight into the roles of the different loops at specific times in the simulation.

– The results are only qualitative in nature: a loop either is dominant or it is not. No insight or quantitative measure is given which provides the analyst with an indication of how much of an influence the loop in question has.

– In contrast to eigenvalue elasticity analysis, the conclusions of Ford's behavioural approach are only relevant to one specific variable and cannot be extended to the system as a whole.

The Range Of Models

+ The behavioural nature of the method allows for it to be applied to almost any System Dynamics model. For instance, no demands are made with regard to the continuity of the model functions.

– Its reliance on specific behaviour modes makes it far less suited for models that exhibit oscillations. While the strength of a loop remains constant, Ford may detect constantly changing behaviour patterns.

– In larger models, only a selection of the loops can be analysed, owing to the time-consuming nature of the method. This makes the method methodologically weaker when applied to these models, since it cannot test the role of loops outside of the initial selection. The selection of the loops can be based on either input from a different method of analysis, or the modeller’s knowledge of the system.

Resources Required

+ No extensive mathematical background is required to understand the method

+ The analyst can perform the analysis using only standard System Dynamics software.
The process of manually taking out loops, rerunning the model and interpreting the results is very time-consuming.

8.3.2 Eigenvalue Elasticity Analysis

The most significant contribution of eigenvalue elasticity analysis is the visualisation of the fractional influence of loops on eigenvalues and state variables over time. The results can be displayed as a (semi-)continuous graph over time, tracing the influence of structure on behaviour.

In addition, the rigorous mathematical background of the method allows it to be automated. Assuming that the model is in an analysable form, the analysis itself is only a matter of running the appropriate functions.

The several forms in which the method is available do not exclude others and they can compensate for one another's weaknesses. For instance, the loop-based elasticity analysis can be used as a means of explaining the influence of a specific parameter in more detail. The analyst can choose to perform the method from a system-wide perspective or focus his/her attention on one specific state variable.

All forms of eigenvalue elasticity analysis are still difficult to interpret. To get these accepted in the real world, the average analyst should be able to distill bitesize lessons from the analyses that are translatable to the language of the client. Currently, only those who have worked with these techniques in-depth or have been introduced to their inner workings in some other way can use them. As Kampmann and Oliva (2005) also stress, there is a need for a way to represent eigenvalue elasticity analysis results that facilitates interpretation, while not oversimplifying a complex method.

Fundamentals Of The Method

+ Its mathematically rigorous foundation results in a consistent method suited for automation.

− There are points in the analysis at which the elasticities are undefined. This forces the analysis to be split into several open intervals. These problems are due to the singularity of the right eigenvector matrix (Kampmann and Oliva, 2005).

The Results Provided

+ The method provides the analyst with a visualisation of the fractional influence of loops on eigenvalues and state variables. This fractional influence is traced semi-continuously over the entire model run.
8.3. ASSESSMENT OF THE METHODS

+ The analyst can choose to use either the form which relates structure to the system as a whole or to a specific variable.

+ By using a parameter-based approach or a loop-based approach, the analyst can relate either model parameters or model structure to behaviour.

− Due to the abstract nature of the method, there is a steep learning curve in interpreting the results of the analyses.

− The weakness of the parameter-based variant is that it does not actually relate structure to behaviour. The mechanisms (loops, pathways) via which the parameters influence the eigenvalues are not addressed.

− The variant of the analysis that relates the elasticities to the behaviour of specific state variables is less interpretable when a complex conjugate pair of eigenvalues is present. This forces the analyst to trace back the origins of the results and inspect the original elasticities.

Range Of Models

+ Although the method may suffer from numerical instability when dealing with larger models, the range of models to which it is applicable is not limited by the effort the modeller has to invest. However, this is under condition that the models are in a form that can be readily analysed by automated tools.

− The form of the analysis that associates loops or parameters to system wide behaviour is less scalable. The higher the order of the model, the less interpretable the system-wide variant becomes (Kampmann and Oliva, 2005).

− Although it is not apparent from this research, the methods based on the shortest independent loops set have difficulties with analysing models that contain edges that can be turned off completely (Güneralp and Gertner, 2006).

Resources Required

+ Compared to Ford’s behavioural approach, the eigenvalue elasticity analysis is particularly fast when automated.

− The analyst needs to be familiar not only with the method itself, but also with the underlying concepts such as linearisation, eigenvalues and eigenvectors.

− The method requires software such as MatLab, Java or Mathematica (Kampmann and Oliva, 2005) in order to perform the required calculations.

− The model needs to be in a form that is compatible with the analysis tools.
8.3.3 Comparative Analysis

Disparities In Results

The eigenvalue elasticity analysis provides us with a mathematical criterion for dividing the continuous time span of the analysis into specific phases. These phases are determined on the basis of overall model behaviour rather than the behaviour of a single variable. Ford’s method, however, focuses on the behaviour of a single variable, and on this basis determines its time intervals for analysis. The two methods generate different intervals to divide the analysis into: while the division used for the eigenvalue elasticity has a mathematical foundation, the division used by Ford is based on the immediately apparent behaviour of the variable of interest. In contrast to eigenvalue elasticity analysis, the Ford method assumes that these intervals signal changes in the structure driving behaviour. The validity of this assumption is questionable.

In addition, there are discrepancies in the outcomes of the two methods, especially where the Ford Method requires the elimination of more than one loop in the model. We find the difference between the original model and the model with one loop taken out small enough to be acceptable. In contrast, a model with two loops eliminated is so different from the original model that it is doubtful whether the conclusions based on the modified model can be translated directly to the original model.

Finally, both methods have problems with analysing larger models. Eigenvalue elasticity analysis has a greater chance of displaying singularities if models become larger. However, due to several reasons, Ford’s behavioural approach runs into severe problems when model size increases. First of all, the larger the model, the higher the chance of shadow loops, which does not improve the validity of Ford’s method. Second, the larger the model, the lower the percentage of the loops present in the model that can be analysed within a reasonable timeframe. And, third, larger models are more vulnerable to the lack of independence between loops due to nonlinear effects.

Implementation Requirements

In spite of improvements in the consistency of Ford’s method, it has not yet been fully automated. Full automation is considered feasible, if user input is still allowed to compensate for the shortcomings in the foundations of Ford’s method. An additional argument for this automation is that it will compensate for some of the methodological weaknesses of the method.

It was necessary to automate model behavioural analysis to obtain the results presented here, since no standard readily available software exists for implement-
8.4. **Final Remarks**

Although there has been significant progress in model analysis in recent years, methods are still too underdeveloped and too difficult to understand to put into everyday practice. However, the further automation of the available methods can help in speeding up research and spreading the application of the methods.

In my opinion, eigenvalue elasticity analysis appears to be the most powerful method currently available, due to its ability to describe either system behaviour or the behaviour of a single variable, to effectively deal with oscillations and to estimate the elasticity of either system parameters, loops or edges. However, it is still very much in the experimental phase, as shown by the issues still remaining, such as its inability to deal with discrete edges in the model and the difficulty in interpreting results.
Chapter 9

Recommendations

As mentioned in the conclusions, the available methods are still in the exploratory stage of development. However, they have the potential to move beyond this stage, so that they are more useful in aiding the analyst in achieving full understanding of the model. In this chapter, several challenges are identified; solving these challenges would bring the methods significantly closer to fulfilling their potential.

The challenges facing the field of model behavioural analysis lie in several areas and will be categorised accordingly. The categories represent research that can be done to further solidify the fundamentals of the methods, mainly eigenvalue elasticity analysis.

Theory
Challenges concerning the further improvement of the fundamentals of the methods.

Implementation
Challenges at the level of implementation of analysis techniques that do not require further theoretical research.

Application
Challenges concerning the application of model behavioural analysis to the modelling process.

Research Process
These recommendations concern steps to facilitate future research.

9.1 Theory

- As mentioned in the assessment of eigenvalue elasticity analysis, the results of an eigenvalue elasticity analysis are undefined at points where the right
9.2. IMPLEMENTATION

The eigenvectors of the gain matrix are undefined. The question is whether it is possible to analytically determine at which points (or subspaces) in state space the singularities will occur. This will provide the analyst with a tool that can predict when these singularities will occur, without running the model.

- In their 2005 paper, Kampmann and Oliva (2005) state that eigenvalue elasticity analysis is not suited for the analysis of models that exhibit chaotic behaviour. They also conclude that the technique is most suited for the analysis of linear and quasi-linear oscillation. However, the remaining question is whether a qualitative or quantitative criterion can be formulated for exactly how suited eigenvalue elasticity is for a particular model.

- Gonçalves et al. (2000) identified the possibility of working with approaches to analysis that are not based on an independent loop set. The formulation of these methods would mean the emergence of another variant of eigenvalue elasticity analysis, which possibly mitigates several of the weaknesses of an independent-loop-set-based approach.

- The algorithms used in model analysis only provide results valid for a specific representation of the model in question. The question as to whether the methods can be made independent of the current representation of the model, or whether a heuristic can be found for determining a single “correct” representation which still retains enough information, remains.

9.2 Implementation

- The current implementation of eigenvalue elasticity analysis uses perturbation to calculate the elasticity of an edge to an eigenvalue. However, it is feasible (and recommended) to implement the analytical variant of calculating the edge elasticities (Güneralp, 2005). The most difficult section of this form of the method would be the structural analysis required, since it relies on the identification of pathways in the model.

- Not all variants of eigenvalue elasticity analysis currently available have been implemented for this research. For instance, the variant as presented by AbdelGawad et al. (2005) has not been implemented. The implementation of the remaining variants is recommended, if only to compare their results with the currently analysed variants.

- As noted by Güneralp and Gertner (2006) the current implementations of independent-loop-set-based eigenvalue elasticity methods can, in certain cases,
not cope with edges in the model that can be shut off completely\(^1\). In these cases, the independent loops set built at the beginning of the simulation may no longer be valid for analysis. It would be advisable to build an implementation of the eigenvalue elasticity analysis methods that can cope with this situation, either by no longer relying on an independent loop set, or by recalculating the set when edges are turned off.

- The eigenvalue elasticities are hardest to interpret when there is a complex conjugate pair of eigenvalues. Currently the real and imaginary parts of the elasticities of the pair of eigenvalues are separated, but the relative contributions of the real and imaginary parts of the eigenvalues to the behaviour of the state variable of interest are not. This limits the insights derived. By quantifying the separate contributions of real and imaginary parts of the complex eigenvalues, this problem can also be addressed.

- The proposal of Diker and Allen (2006) to develop a more standard, formal way of representing implemented System Dynamics models based on XML seems particularly promising. In my opinion, this would facilitate research into quantitative methods for System Dynamics by circumventing proprietary formats and using the power of an accepted format such as XML. With regards to the research into model behavioural analysis, would be advisable to implement a prototype of the proposed format and use this to circumvent the labour-intensive process of implementing a model currently needed to perform eigenvalue elasticity analysis.

- In order to avoid at least part of the method’s methodological weakness, it would be advisable to automate the Ford (1999) method. Given the weaknesses of the method, even an automated version of this method would still require input from the analyst, especially at points such as the selection of the method of deactivating edges.

### 9.3 Application

- As was stated in the limitations of this research, a thorough investigation into how to link the available methods for model analysis to the modelling process is required. However, if the methods are to be applied in the real world it is necessary to further define how exactly they can be used and what tangible benefit they can provide to analyst and client. Ways in which the methods

\(^1\)By using, for instance, an IF-function.
can support the modelling process include:\(^2\,3^3:\)

- **Testing System Boundaries.** If the behaviour of the model is strongly dependent on an exogenous explanation, the system boundary we started out with might not be suitable for the purpose of the model. Assuming that the analyst is looking for an endogenous explanation of behaviour, if he finds that that behaviour is for a large part dependent on an assumed relation, this may compromise the utility of the model. In general, however, model analysis can help in identifying those assumed relations that play a large role in the dynamics of the system, possibly in support of sensitivity analysis.

- **Model behavioural analysis** can help in identifying leverage points in the model, along with their associated effects (Richardson, 1986). In addition, insight into the dynamics underlying behaviour can aid the modeller in designing more reactive, intelligent strategies. The understanding of dynamics could provide insight into which options create interference and which synergy.

- **The dynamic hypothesis** is an attempt to explain observed behaviour in terms of the model. Model analysis can be used to test whether the implemented model corroborates or refutes the hypothesis.

- If the modeller is developing the model incrementally, understanding model behaviour can help in guiding the development into the regions most influential to behaviour (Thissen, 1978).

- **Model analysis** can aid in identifying those parts of a model that can be left out without having much effect on model dynamics and thereby aid in reducing the model to a smaller form.

- The communication of the insights derived from an analysis method to the client and the interpretability of the results of eigenvalue elasticity analysis are both underexplored areas of research. Since both relate to the communication of results, they are considered to be part of the application phase.

- First of all, as identified by the examination of the eigenvalue elasticity analysis, the results of the method are very hard to interpret as they stand now. A smart visualisation of loop, edge or parameter elasticities would provide the analyst with a tool which can more directly communicate the results of the analysis. However, it is a challenge to find a form of

\(^2\)Also mentioned in Section 2.3.1

\(^3\)In my opinion, the techniques for model analysis appear to be similar in their application to sensitivity analysis. However, where sensitivity analysis takes a relatively “black box” approach, these techniques aim for more insight into the mechanics of the model.
visualisation that can convey the complex results of an eigenvalue elasticity analysis, while still maintaining accessibility. Given the difficulty and the possible benefits of this visualisation, research into this area is recommended.

- Given the difficulty of interpreting eigenvalue elasticity, the technique can be made more powerful by developing a set of heuristics for the interpretation of its results and building a system story out of the numbers. For instance, one of the practices that can help in getting to a system story is to clearly name the loops in the system, forcing the analyst to map results back to conceptual terms. Naming the phases of the model would have similar benefits. Hence, developing a set of guidelines or heuristics for the translation back to conceptual language is recommended.

- A consistent and more insightful terminology for describing the relationships between model behaviour and structure would be useful for explaining the method and its results. For instance, the term “loop dominance” assumes an almost binary concept of dominance; either a loop is part of the dominant set or not. However, given the insights derived from eigenvalue elasticity, this is not necessarily the perspective that is the most accurate representation of how model dynamics are generated. The fractional contribution to behaviour would be a more accurate term for describing the role of an element of structure. Hence, research into a more communicative terminology is necessary.

9.4 Research Process

- A possible approach to further research into the application of the methods would be to build a set of “standard” test models for methods for model behavioural analysis. What is needed is the selection of these test models, and criteria to base the selection and categorisation of these models on. For instance, criteria for categorisation could be size or the presence of discrete edges.

- Given the presence of several equivalent methods (Kampmann and Oliva, 2005; Güneralp, 2005; AbdelGawad et al., 2005; Goncalves et al., 2000), several different implementations of similar algorithms and the challenges facing these methods, a lot can be gained from organising a research workshop. The participants in this workshop would ideally be the researchers currently active in the field of model behavioural analysis. The workshop can focus on
the directions for further research and ideas for tackling the problems currently facing the available methods for model analysis.

- In this research, the assessment of the methods for model behavioural analysis was done based on the advantages and disadvantages of the methods in certain areas. However, it might be feasible to formulate more exact criteria for the performance of a method for model analysis. Establishing these criteria would involve input from both the active modelling community and those researchers that work on the methods for model analysis. Formulating these criteria might provide more focus to the research effort into model behavioural analysis.


Richardson, G. P. and Pugh, A. L. (1981). *Introduction to System Dynamics Modelling with DYNAMO*. Productivity, Cambridge, MA. 1, 2, 6, 7, 8, 9, 22


Appendix A

The Example Models

A.1 Predator - Prey

This section contains a short introduction to the example model used throughout Section 4.1 and Chapter 5.

The model is a classic predator-prey model. It displays quasi-linear oscillation.
A.1.1 System Diagram

Figure A.1: Lotka-Volterra model
A.1.2 Equations

The equations are set in the form of a matlab function.

```matlab
function varargout = lotkevolts(t, x)

    % Constants
    birth_rate_prey = 0.15;
    standard_death_rate = -0.01;
    starvation_factor = -0.00012;
    kill_factor = -0.004;
    birth_factor_predator = 0.0005;
    predator_death_rate = -0.2;

    % states
    prey = x(1, :);
    predator = x(2, :);

    % Auxes
    prey_birth = prey * birth_rate_prey;
    standard_deaths = prey * standard_death_rate;
    starvation_deaths = starvation_factor * prey.^2;
    encounter_kills = kill_factor*predator.*prey;
    total_prey_deaths = standard_deaths + starvation_deaths +
    + encounter_kills;
    predator_births = birth_factor_predator.*prey.*predator;
    predator_deaths = predator_death_rate.*predator;

    % dys
    dx(1,:) = prey_birth + total_prey_deaths;
    dx(2,:) = predator_deaths + predator_births;

    % Output has to be a column. Odd matlab behavior.
    varargout{1} = [dx(1,:); dx(2,:)];

    % Output auxiliaries if requested
    if nargout == 2
        varargout{1} = [dx(1,:); dx(2,:)];
        varargout{2} = [
            prey_birth;
            standard_deaths;
            starvation_deaths;
            encounter_kills;
            total_prey_deaths;
            predator_births;
            predator_deaths];
    end
end
```
A.1.3 Directed Cycle Matrix

Table A.1: Directed Cycle Matrix of the SILS of the Lotka Volterra Model

<table>
<thead>
<tr>
<th>From</th>
<th>To</th>
<th>L1</th>
<th>L2</th>
<th>L3</th>
<th>L4</th>
<th>L5</th>
<th>L6</th>
<th>L7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Predator</td>
<td>PredatorBirth</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>PredatorBirth</td>
<td>Predator</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Predator</td>
<td>PredatorDeath</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>PredatorDeath</td>
<td>Predator</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Prey</td>
<td>PreyBirth</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>PreyBirth</td>
<td>Prey</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Prey</td>
<td>Kills</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Kills</td>
<td>PreyDeath</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>PreyDeath</td>
<td>Prey</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Prey</td>
<td>Starve</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Starve</td>
<td>PreyDeath</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Prey</td>
<td>StandardD</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>StandardD</td>
<td>PreyDeath</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Predator</td>
<td>Kills</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Prey</td>
<td>PredatorBirth</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

A.2 Yeast

The model is a representation of the growth of Yeast cells in a vat. The cells multiply and eventually die out as a result of the alcohol they produce. In the initial stages of the model, there are very little Cells present and virtually no alcohol. This results in an almost exponential increase of the amount of Cells. As the number of cells grows, so does the Alcohol concentration. At a certain point the influence of alcohol on the deaths of cells is so large, the growth of cells begins to decrease until the amount of cells reaches a maximum at $t \approx 65$. From there on the effect of Alcohol on the Cells is so large this decrease in number until the amount of Cells eventually approximates zero.
A.2.1 System Diagram

Figure A.2: The Yeast model
A.2.2 Equations

Table A.2: Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Symbol</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lifetime</td>
<td>( l )</td>
<td>30</td>
</tr>
<tr>
<td>Division Time</td>
<td>( k )</td>
<td>15</td>
</tr>
<tr>
<td>Alcohol Per Cell</td>
<td>( g_c )</td>
<td>( 1e - 2 )</td>
</tr>
<tr>
<td>Birth Mult</td>
<td>( m )</td>
<td>(-0.1)</td>
</tr>
<tr>
<td>Birth Added</td>
<td>( a_b )</td>
<td>1.1</td>
</tr>
<tr>
<td>Exponential effect on death</td>
<td>( z_d )</td>
<td>(-11)</td>
</tr>
</tbody>
</table>

Alcohol: \( A \)

\[
\frac{dA}{dt} = g
\]

Cells: \( C \)

\[
\frac{dC}{dt} = b - d
\]

Alcohol Generation: \( g \)

\[
g = g_c C
\]

Effect Alcohol on Deaths: \( a_d \)

\[
a_d = e^{A + z_d}
\]

Effect Alcohol on Births: \( a_b \)

\[
a_b = Am + a_b
\]

Cell Births: \( b \)

\[
b = (C/k) a_b
\]

Cell Deaths: \( d \)

\[
d = (C/l) a_d
\]
A.3. **LONGWAVE**

### A.2.3 The Directed Cycle Matrix

<table>
<thead>
<tr>
<th>From</th>
<th>To</th>
<th>$L_1$</th>
<th>$L_2$</th>
<th>$L_3$</th>
<th>$L_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cells</td>
<td>Births</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Births</td>
<td>Cells</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Cells</td>
<td>Deaths</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Deaths</td>
<td>Cells</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Cells</td>
<td>AlcoholGeneration</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>AlcoholGeneration</td>
<td>Alcohol</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Alcohol</td>
<td>EffAlcBirth</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>EffAlcBirth</td>
<td>Births</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Alcohol</td>
<td>EffAlcDeath</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>EffAlcDeath</td>
<td>Deaths</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

### A.3 LongWave

This model is a highly nonlinear economic model used by Sterman (1985) to explain the long term economic cycles caused by capital self-ordering in the simplest terms possible. This resulted in a third order model where the stocks represent three different sectors; capital, supply and backlog. Despite its small size, the model has 16 loops in its SIFS. Its behaviour is a limit cycle.

#### A.3.1 System Diagram

#### A.3.2 Equations

The equations are presented in the form of the matlab function used to test the model.

```matlab
function varargout = longwavess(t, x)

% This function is used as a version of the longwave model.
% Calculations are order sensitive and everything is calculated from
% the states down. This function cannot be used to calculate edge
% gains.

% constants
avg_capital_life = 20;
capital_out_rate = 3;
capital_adjust = 1.5;
normal_delivery = 1.5;
supply_adjust_time = 1.5;
```

```matlab
```
Figure A.3: The Simple Long Wave model
A.3. LONGWAVE

15 % states
16 backlog = x(1,:);
17 capital = x(2,:);
18 supply = x(3,:);
19
20 % aux
21 % 1
22 capacity = capital./capital_out_rat;
23 % 2
depreciation = capital./av_capital_life;
24 % 3
desired_production = backlog./normal_delivery;
25 % 4
desired_capital = desired_production.*capital_out_rat;
26 % 5
capital_adjustment = (desired_capital - capital)./capital_adjust;
27 % 6
cap_util = capacityUtil(desired_production./capacity);
28 % 7
production = capacity.*cap_util;
29 % 8
desired_supply = depreciation.*(backlog./production);
30 % 9
supply_adjustment = (desired_supply - supply)./supply_adjust_time;
31 % 10
desired_orders = depreciation + capital_adjustment + supply_adjustment;
32 % 11
relative_orders = relativeOrders(desired_orders./depreciation);
33 % 12
capital_orders = depreciation.*relative_orders;
34 % 13
acquisitions = supply.* (production./backlog);
35 % 14
% SET TO ONES
36 goods_orders = ones(size(capital_orders));
37
38 % dys
39 dbacklog = capital_orders + goods_orders - production;
40 dcapital = acquisitions - depreciation;
41 dsupply = capital_orders - acquisitions;
42
43 varargout{1} = [dbacklog ; dcapital ; dsupply];
44
45 % output auxiliary values if requested
46 if nargout == 2
47 varargout{2} = [
48     capacity;
49     depreciation;
50     desired_production;
51     desired_capital;
52 ];
53 end
APPENDIX A. THE EXAMPLE MODELS

A.3.3 Graph functions

The functions used as replacements for the graph functions. Their behaviour is compared with that of the original graph functions in graph...

```matlab
function ro = relativeOrders(x)
    seven = ones(size(x))*7.5;
    x = min([seven;x], [], 1);
    zer = x > -1;
    % pi/8.5 controls steepness, the 3.02 controls height
    % not parametrized here, see java function
    func = (sin((pi/8.5)*(x-3)) + 1)*3.02 - 0.1;
    ro = func.*zer;
end

function cu = capacityUtil(x)
    %Input is ratio of orders by depreciation
    % 1.8 rate of increase towards maximum, 1.13 height of maximum
    % not parametrized here, see java function
    func = (((-exp(-x*1.8) + 1)*1.13));
    zer = zeros(size(func));
    onon = ones(size(func))*1.11;
    cu = max([zer; func], [], 1);
end
```
Figure A.4: Graph functions and their replacements in the long wave model
A.3.4 The Directed Cycle Matrix

Table A.4: Directed Cycle Matrix of the Long Wave Model

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Capital</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Depreciation</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Acquisitions</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Supply</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Capacity</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Production</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Depreciation</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Acquisitions</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Supply</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Capacity</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Production</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Depreciation</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Acquisitions</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Supply</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Capacity</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Production</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Depreciation</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Acquisitions</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Supply</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Capacity</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Production</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

From Table A.4:
A.3. **LONGWAVE**

### A.3.5 Behavior of auxiliaries

![Diagram showing the behavior of auxilary variables during a cycle of the Long Wave model](image)

Figure A.5: Behavior of auxilary variables during a cycle of the Long Wave model
A.3.6 Contributions of the Eigenvalues

Figure A.6: Contributions of the eigenvalues and the behavior of the state variable of interest in Phase I of the LongWave model

Figure A.7: Contributions of the eigenvalues and the behavior of the state variable of interest in Phase II of the LongWave model
Figure A.8: Contributions of the eigenvalues and the behavior of the state variable of interest in Phase III of the LongWave model

Figure A.9: Contributions of the eigenvalues and the behavior of the state variable of interest in Phase IV of the LongWave model
Appendix B

Structural Analysis Supporting

B.1 Calculating the Reachability Matrix

This appendix demonstrates the reasoning used to obtain the reachability matrix of a model. All calculations are done using Boolean matrix algebra. If we multiply the adjacency matrix by itself, we obtain a matrix that shows the 2-step paths between variables. That is, for an adjacency matrix \( A \), if \( A_{i,j}^2 = 1 \), there is a path of length 2 from vertex \( i \) to \( j \). A path from vertex \( i \) to vertex \( j \) is always along a third vertex \( m \), the total path being \( i \rightarrow m \rightarrow j \). Hence, both \( A_{im} \) and \( A_{mj} \) are 1. Similarly, if there is a path \( i \rightarrow m \rightarrow i \), both \( A_{im} \) and \( A_{mi} \) are 1. Consequently, using the rules of matrix algebra and multiplying \( A \) with its \( j \)-th column to obtain the \( j \)-th column of \( A^2 \), the \( i \)-th element in the resulting column will be one (Aldous and Wilson, 2000).

In the example above there is a path \( b \rightarrow c \rightarrow a \) (Figure B.1).

---

Figure B.1: Calculation of \( A^2 \)

\[ A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \]

\[ A^2 = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \]

---

\(^{1}\)Boolean matrix algebra is defined on the Boolean addition \( 1+1=1, 0+0 = 0, 0+1=1 \) and \( 1+0=1 \). (Oliva, 2004)
In parallel fashion, multiplying $A^2$ with $A$ results in the matrix displaying all paths in the graph with a length of 3. In general, multiplying $A^{n-1}$ with $A$ gives the existence of paths of length $n$.

Given the matrix $R$ constructed by $R = \sum_{i=1}^{n} A^i$, $R_{ij}$ shows whether variable $i$ can reach $j$ within $n$ steps or less. There is a value of $n$ for which $R_{n-1} \neq R_n = R_{n+1}$. Lengthening the possible paths then no longer results in new vertices being reachable by other vertices. The matrix thus obtained is the reachability matrix. The reachability matrix for the example can be found in Figure B.1.

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>B</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>C</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>E</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>F</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Figure B.2: Reachability matrix

The reachability matrix shows that there exist paths from vertices $a$, $b$ and $c$ to all vertices, while from vertices $d$, $e$ and $f$ only $d$, $e$ and $f$ can be reached.

Oliva (Oliva, 2004) uses the same reasoning to obtain a reachability matrix, but first makes the graph reflexive - each variable is considered to be part of its own direct predecessor and successor set - before determining $R$. In matrix form, to make the graph reflexive we set $B = I + A$ and determine the reachability matrix by raising the power of $B$ instead of $A$, i.e. $R = B^n$, until $R_{n-1} \neq R_n = R_{n+1}$.

**B.2 Example of model partitioning**

To illustrate partitioning we will use the following SD model and its graph representation. This is a Lotka-Volterra model with a small extension. The predators and their prey inhabit a nature reserve that relies on eco-tourism for its income. If game viewing is entertaining enough, tourists will spread word of the reserve, increasing its popularity. However, too many visitors will result in overcrowding, negatively influencing the reserve's popularity. The number of tourist visits determines the revenue of the reserve. The diagram of the model has been slightly modified around prey deaths to prevent it from becoming too large. See Figure B.3.

The example can be divided into five partition levels according to the algorithm de-
Figure B.3: Systems Diagram of Lotka-Volterra with tourism

<table>
<thead>
<tr>
<th>prey birthrate</th>
<th>prey deaths</th>
<th>starvation factor</th>
<th>standard death rate</th>
<th>encounter kills</th>
<th>kill factor</th>
<th>income per visit</th>
<th>park revenue</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>prey births</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>prey</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>starvation factor</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>standard death rate</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>encounter kills</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>predator</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>predator birth factor</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>predator death rate</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>predator deaths</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>entertainment factor</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>entertainment</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>mouth to mouth rate</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>mouth to mouth</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>tourist visits</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>crowdedness factor</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>crowdedness effect</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>income per visit</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>parc revenue</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Figure B.4: Adjacency Matrix of Lotka-Volterra with tourism
Figure B.5: Sparsity pattern of the reachability Matrix Lotka-Volterra with tourism (variable names and zeros not shown, ones are dots)
scribed above. All the variables in a feedback loop are situated in the same partition level, but not all variables in a partition level are part of the loop. A higher level is exogenous to a lower level; there exists a relation from the higher level to lower, but never the other way around. Even stronger, part A of a model that is exogenous to part B will always be of a higher level. For example, all variables in level 5 in Figure B.6 are exogenous to those in level 4. See Table B.1 and Figure B.6.
B.3. MATLAB CODE LOOP DETECTION

Table B.1: Level partitioning of LV with tourism

<table>
<thead>
<tr>
<th>Level 1</th>
<th>Level 2</th>
<th>Level 3</th>
<th>Level 4</th>
<th>Level 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>parc revenue</td>
<td>mouth to mouth</td>
<td>entertainment factor</td>
<td>prey births</td>
<td>prey birth rate</td>
</tr>
<tr>
<td>tourist visits</td>
<td>mouth to mouth rate</td>
<td>prey</td>
<td>starvation factor</td>
<td></td>
</tr>
<tr>
<td>crowdedness effect</td>
<td>crowdedness factor</td>
<td>prey deaths</td>
<td>standard death rate</td>
<td></td>
</tr>
<tr>
<td>income per visit</td>
<td>encounter kills</td>
<td>kill factor</td>
<td></td>
<td></td>
</tr>
<tr>
<td>predator</td>
<td>predator birth factor</td>
<td>predator deaths</td>
<td></td>
<td></td>
</tr>
<tr>
<td>predator births</td>
<td>predator death rate</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>predator deaths</td>
<td>entertainment factor</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure B.6: Partition levels in LV with tourism

In the example, cycle partitions in the example can be found in level 2 and level 4. They are:

B.3 MatLab Code Loop Detection

```matlab
function c = detectloops(adjacencyMatrix, rowIndex, influencesSoFar, c)
% It’s recursive, hence not suitable for exorbitantly large matrices
% However, maximum path length between two vertices determines depth of
% recursion, hence recursion not that much of a problem.
```
Table B.2: Cycle Partitions in LV with tourism

<table>
<thead>
<tr>
<th>Cycle Partition 1</th>
<th>Cycle Partition 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>mouth to mouth</td>
<td>prey births</td>
</tr>
<tr>
<td>tourist visits</td>
<td>prey</td>
</tr>
<tr>
<td>crowdedness</td>
<td>prey deaths</td>
</tr>
<tr>
<td></td>
<td>encounter kills</td>
</tr>
<tr>
<td></td>
<td>predator</td>
</tr>
<tr>
<td></td>
<td>predator births.</td>
</tr>
<tr>
<td></td>
<td>predator deaths</td>
</tr>
</tbody>
</table>

% Argument checking
the_size = size(adjacencyMatrix); % square
if the_size(1) ~= the_size(2)
    error('matrix_not_square');
end

if iscell(c) ~= true
    error('must_have_a_cell_as_fourth_argument');
end

if rowIndex < 1 || rowIndex > the_size(2)
    error('Invalid_rowindex');
end
% End of Argument checking

% variable init
ended = false;
loop = false;

% Add current rowIndex to path
influencesSoFar = [influencesSoFar; rowIndex];

% Check for endconditions
% First for sink
if sum(adjacencyMatrix(rowIndex,:)) == 0
    ended = true;
end

% Then for loop
sizeOfFind = size(find(influencesSoFar == rowIndex));
% influences are in column vector form, hence, first entry in size of find
if sizeOfFind(1) > 1
    loop = true;
end

% If neither, recur function on successors
if ended ~= true && loop ~= true
    for n = 1:the_size
        if adjacencyMatrix(rowIndex,n) == 1
            % pass by value
            c = detectloops(adjacencyMatrix, n, influencesSoFar, c);
        end
    end
else %if loop == true
    % Unless this is the end of path or a loop
    % If its a loop, add to path cell
    sizeofc = size(c);
    c(1, sizeofc(2)+1) = {influencesSoFar};
end
Appendix C

Expert Consultations

C.1 Burak Güneralp

This section contains the relevant sections of the consultation of Burak Güneralp, the developer of the most recent addition to model analysis methods. There was a detailed discussion on some of the implementations and interpretations in Güneralp (2005), but this is not considered relevant.

My initial two questions

Dear Burak,

In the past few months I’ve been working on a master thesis regarding the application of loop dominance analysis in System Dynamics. Your 2005 paper on the subject has been a tremendously useful reference so far. However, I’ve been struggling with a few points and would very much appreciate your opinion on these.

1) In your paper you chose to write out gain equations for nearly all model variables. In contrast, I’m running a method that does a fairly quick (and dirty) finite difference method to calculate edge gains at certain points in the simulation. Would you consider the numeric approximation of edge gains valid?

2) You took the approach of performing a loop dominance analysis for every timestep in the simulation. It seemed to me that the analyst could be provided with the option of reducing / increasing the granularity of the analysis. So, basically, I decided to build the option of taking snapshots of the model at preselected times and perform the analysis. The change of slope is still calculated over a single timestep. Do you see any danger in taking this approach?
So far, I’ve been working on a few functions and Java Classes that are intended to help the automation of model analysis. If you’re interested in my progress I’d be happy to send you more information. All work is still very much “under construction” though :-).  

kind regards, thanks in advance,  

Willem Geert Phaff  

_response from Guneralp_  

Dear Willem,  

It is very nice to know that there are people like you who are interested in furthering the formal loop dominance analysis. I am also happy that my conference paper has been useful for your studies.  

1) I do not think that, in most cases, it would pose a problem. But sometimes in some models the edge gains change vary fast. In those situations it might be a better idea to keep the analysis interval as fine as possible. The best option would be to couple this approach with the flexibility of being able to change the analysis time step. Now I guess this brings me to your second question.  

2) This is a very useful extension to the methodology. As you mentioned there may be certain time windows in which the analyst is more interested. I do not see any danger in this approach. To the contrary, it provides flexibility which may be very much needed in dealing with complex models. I have a question though: Does this approach also encompass the ability to change the analysis time step if desired?  

I am certainly interested in your work and would like to know more about your progress. If you have any other questions (or if there is anything unclear in my above response) just let me know...  

Good luck in your studies and keep in touch!  

BurakG  

_my response_  

Dear Burak,  

Thanks for the reply, this is all getting more and more interesting. I tried to answer your question regarding the analysis timesteps, but this has turned out to be a far
lengthier e-mail than I expected. Also, I took the opportunity to ask yet another question.

With regards to your question about the ability to change the analysis timestep, I think the answer is yes.

For now, I just pass the analysis function a vector of timesteps for the integrator and a separate vector of times at which to perform the analysis. For clarity’s sake, if I refer to timesteps, this means the timesteps for the integration of the model, not the analysis.

Also note that Vensim (or any other standard SD software package for that matter) is not being used. Personally, I found them to be a bit too restrictive to perform the analysis the way I wanted to.

After performing the necessary structural analysis*, the function runs the model until the first time for loop dominance analysis. It stops running at that point, performs the analysis, stores the results and continues to run until the next time for analysis. This goes on until either the last time for analysis or the last timestep has been reached. During intervals of rapidly changing loop gains, the times for analysis could be put closer together.

The intention is to provide the modeller with maximum flexibility for both integration and analysis by separating the integrator, the model and the analysis functions. So, both the integration timesteps and the times at which the analysis is performed are completely up to the analyst. The intervals of interest can quite easily be determined by tracing loop gains and checking how variable they are, but this is not included in the function and still up to the modeller.

To make things a bit less abstract I included a prototype of the main MatLab function. Supporting functions and Java classes are not included, that would be overkill. The function unfortunately doesn’t run without these. For now this only performs the analyses as described by Kampmann, but the idea is there. The same framework could be used on any form of analysis. I hope my code and answer are clear enough, if not, I’d be happy to provide further explanation.

... continues on discussion regarding implementation of calculating elasticities.

* The assumption is that model structure is constant for a single model run, so we only have to perform structural analysis once. The Java representation features a method that returns the adjacency matrix for the model. Obtaining the Directed Cycle Matrix requires some matrix manipulation.

** In my opinion, separation and modularity are the way to go with loop dominance analysis. Especially since it’s still very much under development.

Burak’s final response

Dear Willem,
Thanks for your response and I apologise for my late reply. However, I had to take my time to give you a proper answer.

First, based on what you wrote in your mail, I think the approach you have taken is the way to go in formal loop analysis. The basic foundation of the methodology has now been properly laid out. At this point, what we need is the speed and flexibility of an automation of the methodology in its present form -such as yours- in order to further the analytical foundation (there are issues such as repeated eigenvalues, application in case of chaotic behaviour etc.) and improve the applicability.

... continues with discussion regarding implementation of calculating elasticities.

C.2 Alexander Verbraeck

These are the main points obtained from the conversation with Alexander Verbraeck on Monday the 3rd of October.

1. Reduce the current interfaces to their absolute minimum required to get things running.
2. Give the methods clearer names than what’s currently present.
3. Rethink the Vertex to variable inheritance so that it’s clearer what part is vertex and what part is System Dynamics.
4. See if the specification of the model can be reduced. The current way of formulating a model is clumsy at best.
5. See if the current implementation can be linked to the Jung Library for the Graph perspective. It is unnecessary to do work that has already been done.

Recommendations 1 through 3 have been (partially) implemented. Recommendations 4 and 5 will be taken into account for future development.
Appendix D

Prototype Framework

As a proof of the above concept, a small prototype has been built. This appendix will describe its properties and the choices made in more detail. The prototype is a combination of Java classes for the representation of the model and MatLab functions for the overall procedure, integration and analysis.

D.1 Choice of Languages

Java is an object oriented programming language. The language centres on building classes of objects, where a program runs by passing messages among these. One the one hand, message passing is not well suited for use in relatively procedural like activities, like numerically solving differential equations. On the other hand, the Java practice off strictly defining classes of objects, inheritance between them and polymorphism is a feature that is desirable in any project that requires strict conceptual definitions. Consequently, Java has been used for the definition of the model and the internal model structure.

At the core, MatLab is a high level language for technical computing specifically oriented at matrices. The basic data element in the language is a weakly typed array that does not require dimensioning, which - combined with the extensive function library - allows for quick and easy simulation and computation. The entire MatLab system features a development environment, an extensive function library and support for graphics, both 3D and 2D. It the ability to interface with external programs, including Java. Combined with Java, we have the languages for the implementation of the prototype.

The stateless nature of matlab functions make the language unsuited for implementation of some of the more advanced techniques that are used by the framework. A model run using only matlab functions can not be stopped in order to
D.2. THE OVERALL PROCEDURE

inspect model state and linearisations easily. The object oriented way of working within MatLab itself was not considered to be conceptually clear enough.

D.2 The Overall Procedure

function [realRes, imagRes] = y.demo(theModel, timespan, ... snapShots, integrator)

% Model made global for wrapper function when integrating
global model

% Java imports
import nl.tudelft.thm.pa.rubberband.util.*;
import nl.tudelft.thm.pa.rubberband.*;

% Model reference
model = theModel;

% Assuming constant structure, do the structural analysis first.
% Deal with discrete edges in lower functions, do not change structure
% Do not include model parameters.
adjac = model.getAdjacencyMatrix(...
    VertexInterface.EXCLUDEPARAMETERS);
[edges DCM] = getSilsDCM(adjac);

% Determine the end of time
endOfTime = timespan(numel(timespan));

% Initialize vectors for results of integration
y = zeros(1, model.getStates().length);
t = 0;

% Initialize controls for time and snapshots
snapIndex = 1;
tLast = timespan(1);
snapTime = snapShots(snapIndex);
remainingTime = timespan;
finished = 0;

% Run while taking snapshots for analyses.
% Analyse model snapshots in separate function.
while ~finished
    % Get the snapshot, else if (time < snapshot) run till snapshot
    if tLast == snapTime
        % Analyze and store results in a 3d result matrix
        % Perform LEEA based on SILS
```matlab
[realR, imagR] = analyzeK(model, edges, DCM);
% This should be better
realRes(:, :, snapIndex) = realR;
imagRes(:, :, snapIndex) = imagR;

snapIndex = snapIndex + 1;
if snapIndex > length(snapShots) | snapTime == endOfTime
    finished = 1;
else
    snapTime = snapShots(snapIndex);
end
else
    % Run till snapshot
    % Determine the vector of the timespan until the next snapshot first.
    idx = find(remainingTime >= snapTime, 1, 'first');
    tidx = remainingTime(idx);
    % If snaptime falls exactly on a timestep
    if tidx == snapTime;
        integVector = remainingTime(1:idx);
        remainingTime = remainingTime(idx:numel(remainingTime));
    % Else if between timesteps
    elseif tidx > snapTime
        integVector = [remainingTime(1:idx-1) snapTime];
        remainingTime = [snapTime, remainingTime(idx:numel(...
            remainingTime)];
    end
    % Very unmatlab, should find proper way of doing this
    states = model.getStates();
    for statesIdx = 1:length(states)
        inits(1, statesIdx) = states(statesIdx).getValue();
    end
    % Integrate. Integrator is a function handle, which
    % should cover the independency of the analysis, the model and
    % the integrator.
    [ts ys] = integrator(@simpleInteg, integVector, inits);
    % Vertcat the solution, do not include redundant steps.
    % Include model solution as part of possible function output?
    t = vertcat(t, ts(2:length(ys)));
    y = vertcat(y, ys(2:size(ys,1),:));
    % Determine the current time of the model
    tLast = integVector(numel(integVector));
end
end

% Remove dummy rows from result matrices
t(1,:) = [];
y(1,:) = [];
```
D.3. ANALYSIS FUNCTIONS

D.3 Analysis Functions

This section of the appendix contains the functions used for the analyses. Some of these have been written into a more generic form. Others required some translation between different formats.

D.3.1 Structural Analysis

This is based on implementations in Oliva (2004). It calls several of the functions described in that paper and implemented by Oliva. As an input it takes the adjacency matrix of a model, from which the SILS is obtained, which is then translated to a DCM. See also R. Oliva’s resource page.

The main wrapper function.

```matlab
function [edgesDCM] = getSilsDCM(adjacency)
% This is the major function for translating the SILS to a DCM
% Get the boolean adjacency matrix from the Java Utility Class
actualMatrix = adjacency.matrix;
% Run the matrix through Oliva’s structural analysis
theStructure = structure(actualMatrix);

% Obtain the loopset from the structural analysis
loopset = theStructure.fbl;
% Reformat the loopset
loopset = crunchILS(loopset);

% Map the crunched ILS to a Directed Cycle matrix, while logging the
[edgesDCM] = edgesInILS2(loopset);
variableArray = adjacency.vertices.toArray();
sizeEdges = size(edges);

% Translate the edges to the appropriate Java objects.
edgesJ = javaArray(‘nl.tudelft.tbm.pa.rubberband.VariableInterface’, sizeEdges);
for i = 1:sizeEdges(1)
    for j = 1:sizeEdges(2)
        edgesJ(i,j) = variableArray(edges(i,j));
    end
end
```

Supportive function. This transforms the matrix as returned by the Oliva functions into a directed cycle matrix.

```matlab
function [edgesDCM] = edgesInILS2(crunchILS)
% This maps the ILS as by Oliva to a directed cycle matrix
sizeILS = size(crunchILS);
```
edges = [];
for idx = 1:sizeILS(1)
   % Remove all trailing zeros from a loop description. Remember that
   % find returns the indices of non-zero elements.
   loop = crunchedILS(idx, find(crunchedILS(idx, :)));

   % For loop (a, b, c) a circshift gives the edges in the loop rather
   % quickly. loop = a b c, circshift(loop, 1) = b c a, vertcat to
   % | a b c |
   % | b c a |
   % Read the edges in the column, up to down. Since SD doesn’t use
   % multigraphs, edges are defined by the vertices they connect.
   loopEdges = (vertcat(circshift(loop, 1)', loop));
   loopEdges = circshift(loopEdges, -1);
   sizeLoopEdges = size(loopEdges);

   % The first loop can just be added, all edges are new.
   if isempty(edges)
      edges = loopEdges;
      DCM = ones(size(edges(:, 1)));
   else
      sizeEdges = size(edges);
      edgesToAdd = [];
      % This is used to track both the DCM and the associated cycles
      for i = 1:sizeLoopEdges(1)
         % Start off with the assumption the edge can be added.
         addEdge = 1;
         % Test if the edge is present in the current edgeset
         for j = 1:sizeEdges(1)
            if (sum(edges(:, :) == loopEdges(i, :) , 2) == 2)
               % Don’t add the edge if it is present.
               addEdge = 0;
               % However, do set the relative element in the DCM to
               % 1.
               DCM(j, idx) = 1;
            end
         end
      end
      % If the edge isn’t present in the current DCM
      if addEdge == 1
         % Add it to the collection of edges to be added for
         % this loop.
         if isempty(edgesToAdd)
            edgesToAdd = loopEdges(i, :);
         else
            edgesToAdd = vertcat(edgesToAdd, loopEdges(i, :));
         end
      end
      % And adjust the DCM accordingly.
      sizeDCM = size(DCM);
      DCM(sizeDCM(1)+1, idx) = 1;
   end
end
D.3. ANALYSIS FUNCTIONS

D.3.2 Kampmann

These algorithms are mainly based on the calculations in Kampmann (1996b). Main function that return only two real valued elasticities.

```matlab
function [realr, imagr] = analyzeKampmann(anyModel, edge, dm)
% Inputs are a reference to the model, the edges of interest and the
% Directed cycle matrix.
% Just Kampmann for now, nothing more.
descriptor = anyModel.getDescriptor(1);
% Just the version that return two real elasticities. Full,
% non-prototype version should have both this and the single complex
% value as possible outputs.
[rlg img] = edgeVectorElasticities(edge, descriptor);
% Calculate the inverse of the Directed Cycle Matrix
theInverse = pinv(dm);
realr = theInverse*rlg;
imagr = theInverse*img;
end
```

D.3.3 Guneralp

These are the generic versions of the pseudo-code published in Güneralp (2005).

```matlab
function varargout = slopesR(model, deltat, t)
% model : A reference to the model,
% deltat: the deltat used to calculate the change in slope
% (independent from integration and analysis times)
% t : current time in the model

% Get the current rates of change (slopevector) from the model
slps = model.getNetRates(t);
% Get the gain matrix from the model
gainMatrix = desc2jacob(model.getDescriptor(0));
% Calculate linearized slopes
slopes_zero = slopesAtT(slps, gainMatrix, 0);
slopes_delta = slopesAtT(slps, gainMatrix, deltat);
```

end
```
% Get the difference between t(0) and t_delta. Relative to deltat
slopechange = (slopes_delta - slopes_zero)/deltat;

% Reduction to real seems to work, the imaginary part always occurs in
% conjugates, so this drops out anyway.
slopechange = real(slopechange);

function scs = slopesAtT(slp, gains, dt)
% Valid to keep in complex form, scs always result in complex
% conjugates.
[vectors lambdas] = eig(gainMatrix);

% The fundamental matrix.
% (boyce & diPrima, paragraph 7.8, 6th edition)
psi = vectors*(exp(lambdas*dt).*eye(size(lambdas)));

% Calculate alphas based on current slopes.
alphas = inv(vectors)*slp;
% Diagonalize alphas
alphas = diag(alphas);
% calculate slope components, retain in 2d form to conserve
% information about different lambdas
scs = psi*alphas;

end

varargout{1} = slopechange;
if nargout == 2
    varargout{2} = diag(lambdas);
end

D.4 Internal Model Structure

This section describes the implementation of the internal model structure as detailed in Section 6.3.4. The inheritance diagram for the model elements can be found in

D.4.1 VertexInterface

As said before, one of the ideas central to the framework is the ability to view an SD model as a graph. Hence, the VertexInterface is the interface from which all other model elements inherit. Every element of the model (vertex in the model graph) implements this interface.
Figure D.1: Inheritance diagram of model elements
D.4.2 SDModelElement

The model element directly inherits from vertex and adds the two methods that all elements in an SD model have. Namely, a name and a value.

D.4.3 Variable

The variable interface has been defined as the common way all variables communicate with the rest of the model. The only extra public method added relative to SDModelElement is that a variable can return the gains of the connections with its predecessors.

Internally, the abstractVariable implements the methods used to calculate values and gains. For the implementation of the internal model structure there were two requirements.

Order Independence

The modeller should not be bothered with setting equations in the correct order. The way the model is formulated should take care of this.

One equation only

To avoid inconsistency in implementation, all variables should determine its value in one place only.

To satisfy the first criterium, each variable directly asks its predecessor for its value by using the method, as described in Section 6.3.4. Equations are defined without order, but executed correctly order by the model asking the states for their net rate of change, which then ask their rates for their value, triggering the chain of requests described in Section 6.3.4.

The problem was that this conflicted the calculation of edge gains and the requirement that values should only be calculated in one place for a given variable. The prototype used a very simple finite difference to calculate edge gains. It now implements a method that accepts a vector of predecessor values and used this for both the calculations of gains and for normal requests of the variable value. The contents of the vector are determined by requesting the value of the predecessors. Gains are linearised for a certain model state.

The weak point in the current prototype is that variables still have to have their predecessors manually set, which results in a cumbersome construction of models. This is done at the construction of the variable or at a later point.
D.4.4 State

This extends the VariableInterface and adds the setValue() and the getTotalRate() method. The first accepts feedback from the integrator and the second provides the net rate of change of the state variables to the integrator. This bases its value on the rates associated with the state, triggering the getValue() recursion.

The associated abstract class extends the the abstract class for all variables. Two methods are overridden, namely the getGains() method and the getValue() one. The getGains() method becomes trivial since all predecessors are rates and all associated gains are 1 or −1. The getValue() method returns the current value of the state, which has been set (indirectly) by the integrator, via the model.

D.4.5 Rate

This extends the general VariableInterface, but adds nothing to it for now. It functions as a type check for the states. Levels should only depend on rates, not auxiliary variables.

D.4.6 Parameter

An interface and associated class used for all model parameters. If these are specified explicitly, they can be used for certain forms of elasticity analyses and, perhaps, sensitivity analysis. It inherits directly from SDModelElement. If asked for its predecessors it will return the empty set.
Appendix E

Intermediate Results Of Ford’s Behavioural Approach

Interval A
Figure E.1: Single edges deactivated at $t = 81.15$. Subfigures are named after the eliminated loops.
Figure E.2: Single edges deactivated at $t = 82.8$. Subfigures are named after the eliminated loops.
Figure E.3: Multiple edges deactivated at $t = 81.15$. Subfigures are named after the eliminated loops.
Figure E.4: Multiple edges deactivated at $t = 82.8$. Subfigures are named after the eliminated loops.
Interval B

Figure E.5: Single edges deactivated at $t = 84.65$. Subfigures are named after the eliminated loops.
Interval C

Figure E.6: $L14$, the Self-Ordering loop, deactivated at $t = 91.5$, $t = 92.05$ and $t = 93$ by setting the unique edge to either its value at the beginning of the interval or to its steady state value. Subfigures are named after the method and moment of elimination.
Figure E.7: Elimination of other candidate loops in interval $c$, with edges deactivated at $t = 91.5$, $t = 92.05$ and $t = 93$ by setting the unique edge to its value at the beginning of the interval. Subfigures are named after the loops and moment of elimination.
Interval D

Figure E.8: Single edges deactivated at $t = 93.25$. Subfigures are named after the eliminated loops.
Interval E

Figure E.9: Single edges deactivated at $t = 100.55$. Subfigures are named after the eliminated loops.
Figure E.10: Multiple edges deactivated at $t = 100.55$. Subfigures are named after the eliminated loops.