# Tools to Analyse Cell Signaling Models

by

David Michael Collins

Submitted to the Department of Chemical Engineering
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Chemical Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2004

Author . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
Department of Chemical Engineering
October, 2003

Certified by . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
Paul I. Barton
Associate Professor
Thesis Supervisor

Certified by . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
Douglas A. Lauffenburger
Whittaker Professor of Bioengineering
Thesis Supervisor

Accepted by . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
Daniel Blankschtein
Chairman, Department Committee on Graduate Students

# Tools to Analyse Cell Signaling Models

by

David Michael Collins

Submitted to the Department of Chemical Engineering
on October, 2003, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Chemical Engineering

## Abstract

Diseases such as diabetes, some forms of cancer, hyper-tension, auto-immune diseases, and some viral diseases are characterized by complex interactions within the human body. Efforts to understand and treat these diseases have only been partially successful. There is currently a huge commercial and academic effort devoted to computational biology to address the shortfalls of qualitative biology. This research has become relevant due to the vast amounts of data now available from high-throughput techniques such as gene-chips, combinatorial chemistry, and fast gene sequencing.

The goal of computational biology is to use quantitative models to test complex scientific hypotheses or predict desirable interventions. Consequently, it is important that the model is built to the minimum fidelity required to meet a specific goal, otherwise valuable effort is wasted. Unlike traditional chemical engineering, computational biology does not solely depend on deterministic models of chemical behavior. There is also widespread use of many types of statistical models, stochastic models, electro-static models, and mechanical models. All of these models are inferred from noisy data. It is therefore important to develop techniques to aide the model builder in their task of verifying and using these models to make quantitative predictions.

The goal of this thesis is to develop tools for analysing the qualitative and quantitative characteristics of cell-signaling models. The qualitative behavior of deterministic models is studied in the first part of this thesis and the quantitative behavior of stochastic models is studied in the second part.

A kinetic model of cell signaling is a common example of a deterministic model used in computational biology. Usually such a model is derived from first-principles. The differential equations represent species conservation and the algebraic equations represent rate equations and equations to estimate rate constants. The researcher faces two key challenges once the model has been formulated: it is desirable to summarize a complex model by the phenomena it exhibits, and it is necessary to check whether the qualitative behavior of the model is verified by experimental observation. The key result of this research is a method to rearrange an implicit index one DAE into state-space form efficiently, amenable to standard control engineering analysis. Control engineering techniques can then be used to determine the time constants, poles, and zeros of the system, thus summarizing all the qualitative behavior of the system.

The second part of the thesis focuses on the quantitative analysis of cell migration. It is hypothesized that mammalian cell migration is driven by responses to external chemical, electrical and mechanical stimulus. It is desirable to be able to quantify cell migration (speed, frequency of turning) to correlate output to experimental conditions (ligand concentration, cell type, cell medium, etc). However, the local concentration of signaling molecules and receptors is sufficiently low that a continuum model of cell migration is inadequate, i.e., it is only possible to describe cell motion in a probabilistic fashion. Three different stochastic models of cell migration of increasing complexity were studied. Unfortunately, there is insufficient knowledge of the mechanics of cell migration to derive a first-principles stochastic model. Consequently, it is necessary to obtain estimates of the model parameters by statistical methods. Bayesian statistical methods are used to characterize the uncertainty in parameter estimates. Monte Carlo simulation is used to compare the quality of the Bayesian parameter estimates to the traditional least-squares estimates. The statistical models are also used to characterize experimental design. A surprising result is that for certain parameter values, all the estimation methods break down, i.e., for certain input conditions, observation of cell behavior will not yield useful information.

Ultimately, this thesis presents a compendium of techniques to analyze biological systems. It is demonstrated how these techniques can be used to extract useful information from quantitative models.

Thesis Supervisor: Paul I. Barton
Title: Associate Professor

Thesis Supervisor: Douglas A. Lauffenburger
Title: Whittaker Professor of Bioengineering

## Acknowledgments

I would like to acknowledge the love and care my parents have shown me over the years; without their support I would not be writing this thesis. The many friends, colleagues, and teachers have helped me over the years are too numerous to mention all by name. However, I would like to thank a few explicitly since they have a special place in my heart. My piano teacher, Mr. Holyman, was humble but had a thirst for knowledge. He taught me that perserverance is always rewarded even if it takes many years to see the benefit. I would also like to thank my chemistry teacher, Mr. Clinch. He stimulated my interest in science, provided wise counsel, and continually challenged me intellectually.

I am also grateful to my undergraduate advisor, Dr. Bogle, who helped me through a formative period and encouraged me to continue my studies. I am also indebted to a good friend, Kim Lee. She has always listened kindly and provided support over the years. I would also like to thank Dr. Cooney for encouraging me to apply to MIT.

During my time at MIT, I have been fortunate to have the wisdom of two thesis advisors. Both Paul Barton and Doug Lauffenburger have enabled me to study in the fantastic environment at MIT by supporting me academically and financially. I have learned a lot from Paul about academic rigor and computational techniques. Doug has opened my eyes to the importance of engineering in biological sciences. I have also leant a lot from my colleagues in both labs; I am grateful to all of them. The trusting environment in both labs is a credit to Paul, Doug and MIT. I would particularly like to thank John Tolsma, Wade Martinson, and Jerry Clabaugh, who over the years have devoted a lot of time to helping me learn about computers. I would also like to thank Adam Singer for helping me learn about global optimization.

Finally, I would like to thank my wife, Christiane. Her support over the last two years has been instrumental. She has taught me never to accept the status quo and to always strive to make things better. Her smiling face and confidence in other people has cheered myself and many other students, staff, and faculty at MIT.

*To my wonderful wife, Christiane.*

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

Most people are familiar with the decomposition of a mammal into biological structures at different scales (from largest to smallest): organs, tissues, cells, complex assemblies of macromolecules, and macromolecules. Many diseases exhibit symptoms at the largest length scales but the cause of the disease is found to be at a far smaller length scale. Furthermore, many diseases have a single main cause (e.g., bacteria, virus or genetic defect). Research that seeks to rationalize the mechanism of a disease to a single cause is reductionist. A simple example might be diarrhea and vomiting caused by the cholera bacteria. The symptoms of the disease have a single cause (the bacteria) and the molecular mechanism by which the bacteria causes the symptoms is well understood (see Page 868 of [144]). It is also well known that treating a patient with antibiotics will usually kill the bacteria and ultimately alleviate the symptoms.

Historically, biological research has used reductionist methods to explain disease and seek new treatments. This approach has been immensely successful. The majority of bacterial diseases can be treated with antibiotics (for example: cholera, tuberculosis, pneumonia) and a large number of serious viral diseases have an effective vaccine (hepatitis A & B, small pox). Furthermore, the cause of many hereditary diseases have been traced to a single genetic defect (for example: cystic fibrosis, Huntingdon's disease, retina-blastoma, sickle-cell anemia). However, there still remain a large number of diseases that are not so well understood and do not have an obvious single cause (for example: some forms of heart disease, some forms of cancer, and some forms of auto-immune disease). There are also many diseases that may have a single cause but are not amenable to a single treatment (for example: human immuno-deficient virus (HIV)).

We are interested in analyzing and predicting cell signaling phenomena. Understanding of cell signaling pathways is important for determining the cause of some diseases, devising treatments, or mitigating adverse consequences of treatments (e.g. chemotherapy). Examples of such diseases include: diabetes and some forms of cancer [128], and some forms of heart disease [186]. Furthermore, the cause or treatment of these diseases usually requires understanding a complex and interacting biological system. However, it is difficult to analyze complex systems without some form of mathematical model to describe the system. Consequently, computational modeling in the biological sciences has become increasingly important in recent years.

## 1.1 Modeling in Biology

The ultimate goal of modeling biological systems is to treat diseases and not to write abstract models. This objective can be stated in terms of the following desiderata for the model:

1. the model should not be too time consuming to build,
2. the model should be reliable and capable of making testable predictions, and,
3. it should be possible to extract useful information from the model.

These desiderata can often be satisfied by a hierarchical approach to modeling [64]. At one extreme, abstract models typically have moderate fidelity over a large range conditions. At the other extreme, detailed models have a far greater fidelity over a limited range of conditions. If someone devotes a fixed amount of time to building a model, they must choose an appropriate level of detail; too detailed and it will be impossible to make predictions over the full range of interest, too abstract and it will be impossible to make sufficiently accurate predictions.

A hierarchy of models can be descended as an investigation proceeds. For example, a research project might start with a geneticist, who builds an abstract statistical model that suggests a genetic cause for a disease. Data from DNA microarrays might be analyzed using a clustering technique to identify possible genes that are involved in the disease. Ultimately, a detailed model is built that describes mRNA levels, protein phosphorylation states and protein concentrations. This model can be used to predict suitable interventions to treat the disease. To build such a detailed model at the outset would be difficult and wasteful. Initially, it would not be evident which proteins and genes to include in the detailed model.

### 1.1.1 Hierarchical Modeling

A typical hierarchy of computational models is suggested in [118] and shown in Figure 1-1. Increasing amounts of *a priori* knowledge is specified as the modeling hierarchy is descended. It is therefore illogical and dishonest not to admit that some (maybe implicit) assumptions are made before formulating a model. For example, almost all scientists accept that stretches of DNA called genes contain a code for proteins. It is therefore important to analyze computational models in a system where such assumptions are made explicit and the concept of *a priori* knowledge is defined. Not only is it necessary to define knowledge, but it is also important to quantify how much we believe this knowledge, and to define some rules describing how this degree of belief is manipulated. It has been shown by [50, 51] and discussed in [122, 123] (and Chapters 1–2 of [121] for the clearest derivation) that Bayesian probability is the best known way to represent this modeling hierarchy. The amount of knowledge or information known about a system can even be quantified using the concept of entropy [200, 121]. Hence, an alternative viewpoint is that the entropy of the model description decreases as the model hierarchy is descended. Bayesian probability and the concept of entropy will be discussed in detail in Chapter 3.

The concept of probability is used in the Bayesian framework to describe uncertainty. This uncertainty is pervasive at every level in the modeling hierarchy. At the most abstract levels, probabilistic models are used to describe uncertainty of

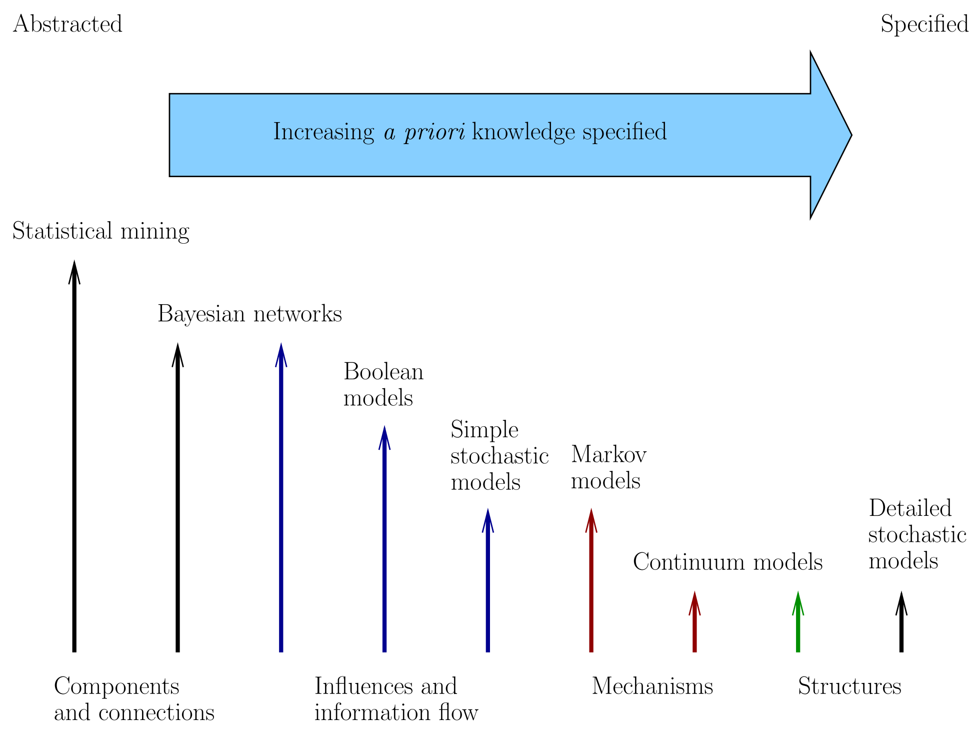


Figure 1-1: Possible hierarchy for modeling biological processes

the system structure (statistical mining and Bayesian networks). In the middle of the modeling hierarchy, Bayesian parameter estimation is combined with continuum models to cope with unspecified model parameters. At a very detailed level, the underlying physical laws governing the system can only be described in probabilistic terms (detailed stochastic models). It should be stressed that modeling uncertainty can arise even when a large amount of *a priori* knowledge is specified about a system. One such example is a detailed stochastic model (for example: the work of [12]).

### 1.1.2 Modeling at Different Levels of Abstraction

Statistical mining and Bayesian networks are abstract models and Markov chains and differential equations are more detailed models. Examples of abstract biological modeling include Bayesian networks [89, 109, 194], and examples of more detailed biological modeling include differential equation models [7, 15, 22, 198], hybrid discrete/continuous models [152, 151], and stochastic models [12]. The appropriate modeling approach is dictated by the objectives of the research. The work [89, 109, 194] used Bayesian networks to attempt to infer regulatory structure. Typically, this would be important at the beginning of an investigation. More detailed modeling work is done later in an investigation. Detailed models can be used to demonstrate a proposed signaling network structure exhibits certain dynamic behavior. The predictions from a mathematical model can be used to verify whether a proposed structure is consistent with experimental data. The more detailed models can also be used to perform *in-silico* experimentation; the model can be tested for a specific set of input conditions to predict output behavior of the system under investigation. The resulting information can be used to suggest possible interventions for a system.

In the middle of the modeling hierarchy are continuum models. These models are often used to validate hypotheses about the detailed structure of cell regulation and require a moderate to high degree of *a priori* knowledge about the system. Such models are not usually formulated in probabilistic terms. In one example, a mathematical model of interleukin-2 (IL-2) trafficking and signaling was used to maximize the long term proliferation of leukocytes by predicting the optimal binding affinities for the IL-2 ligand at different pHs [81]. Subsequently, a modified IL-2 ligand was produced from a genetically modified cell and used to verify the model predictions. The resulting ligand has the potential to reduce significantly the cost and risk associated with treating people with IL-2. Similar work has also been applied to granulate colony factor (GCF) trafficking and signaling [197]. However, sometimes the model parameters will be unknown *a priori*. In this case, the continuum model will be combined with experimental data. The continuum model will be used as an "expectation" function for Bayesian parameter estimation. For a detailed description of Bayesian parameter estimation the reader is referred to [244, 121, 122, 123, 29].

In contrast, stochastic models vary in complexity and do not lie neatly at one place in the modeling hierarchy. A stochastic process is a process where either the underlying physics are in some sense random, or the complexity of the system prevents full knowledge of the state of the system (for example: Brownian motion). A stochastic model therefore describes physical behavior in probabilistic terms. However, the term "stochastic" makes no reference to the complexity or fidelity of the model. Hence, stochastic models will be further classified into "simple" or "detailed" to describe the complexity of the model.

Both abstract and detailed models are inferred from experimental data. It is misleading to distinguish arbitrarily between "first-principles" models (or "models built on scientific/engineering fundamentals") and "statistical models". The term "first-principles" implies that there are some fundamental axioms of science that are known. However, all scientific models are subject to uncertainty and are inferred from experimental observation. What is clumsily expressed by the terms "first-principles" and "statistical" is a qualitative description of the amount of *a priori* knowledge included in the model. Additional *a priori* information will improve the fidelity of the model. A basic desiderata of Bayesian reasoning [121] dictates that the quality of the predictions made from a model will improve as more information is included, provided this *a priori* information is correct. We will always refer to "less detailed" or "more detailed" to describe the amount of *a priori* knowledge included in the model. For a detailed discussion about the scientific method the reader is referred to the preface of [122].

Another common misconception is that detailed models represent the underlying structure of the system whereas less detailed models do not, i.e., there is something *ad-hoc* about statistical mining methods. This is not true. Clustering techniques (such as Principal Component Analysis) can be interpreted in terms of hidden or latent variables [226]. The hidden variables are analogous to states in a control model. Consider principal component analysis of data obtained from DNA microarrays; the latent variables may represent mRNA and protein levels in the cell, i.e., the PCA model has a structure which has a physical interpretation. Bayesian networks are another example of a statistical technique which has a physical basis [89, 109, 194]. The resulting graph suggests connections between physically measured quantities.

### 1.1.3 Detailed Modeling of Biological Systems

Detailed models occur at one end of the modeling hierarchy. Typically, such models seek to model isolated phenomena with high fidelity. Such models are consistent with molecular-directed approaches to determining cell signaling pathways. Molecular-directed methods have been very successful at determining isolated properties of signaling networks. However, these methods are reductionist in nature. Unfortunately, it is becoming increasingly evident that trying to reduce all diseases to a single cause or treatment is a forlorn hope. The key to treating such diseases will rely on understanding complex interactions [118].

Modeling at high levels of abstraction has not been common in the biological sciences, although this is rapidly changing. Traditionally, cell signaling networks have been modeled at the highest degree of detail (consistent with a reductionist approach). Often signaling cascades are modeled as ordinary differential equations or systems of differential-algebraic equations. However, a drawback of this approach is that only a few aspects of the investigation are addressed. The broader context of the research is often not summarized by detailed models [118] and it is common to limit the scope of a detailed model to a degree where important phenomena are not modeled.

There are two possible approaches to mitigate the current shortfalls of detailed cellular modeling:

1. Build tools to make qualitative and quantitative comparisons between model behavior and experimental observation. This approach allows for the iterative refinement of detailed models based on experimental observation.

2. Model the system at a high level of abstraction, sacrificing model fidelity versus range of model applicability.

Both tactics have been used in this thesis and the goal of this work has been to develop tools that can make qualitative and quantitative comparisons between model behavior and experimental observation.

## 1.2 Tools to Analyze Models

The major goal of this thesis is to develop computational tools for analyzing cell signaling phenomena. We have chosen to investigate biological models at two different levels in the modeling hierarchy. Specifically, we have devised methods to summarize the qualitative behavior of detailed models and methods to quantify the accuracy of prediction for less detailed models.

In the first part of the thesis, a kinetic model of the EGF signaling cascade is analyzed. The model is written as a detailed system of differential algebraic equations. The next step is to be able to compare the model to experimental data. We chose to investigate how one could test qualitative agreement between model predictions and experimental data. However, it is difficult to efficiently summarize the qualitative behavior of a DAE model [119]. Yet, this problem has been broadly addressed in the control literature for systems of ordinary differential equations (ODEs) [164]. Research was done on how to rearrange a sparse index one linear time invariant DAE into explicit state-space form. Detailed control analysis can be performed on the resulting model to summarize the qualitative behavior (time constants, poles, zeros of the system).

In the second part of this thesis, stochastic models of cell migration are analyzed using Bayesian statistics. A random component to cell motion is assumed. This assumption is consistent with a model of movement dominated by signaling at low receptor number [231, 232]. Three different models of motion are analyzed representing increasing levels of model complexity. For each model, we wish to quantify several different things:

1. the quality of parameter estimates obtained from the model,
2. the error introduced by model-system mismatch,
3. the optimal experimental design for a given system, and,
4. identify parameter values where estimation was difficult or impossible.

These questions require the quantitative comparison of the computational model to experimental or simulated data. Bayesian statistics is a natural method to compare a computational model to experimental data. Bayesian statistics works by assigning a probability or "degree of belief" to every possible model outcome [122, 123, 121]. Thus, the assigned probability is a function of the hypothesis, statement, or proposition. The resulting mapping from a hypothesis to a probability is called a probability density function. Probability density functions are updated using the famous Bayes rule. While it is usually straightforward to formulate the Bayesian analysis of a computational model, these techniques can be extremely difficult to implement numerically. A particular problem is the high-dimensional integrals resulting from marginalization of unobserved variables. Work in the second part of the thesis focuses on formulating Questions 1–4 as computational problems and solving the resulting integrals.

## 1.3 Epidermal Growth Factor Signaling

It is natural to write detailed models of a cell signaling network in terms of differential and algebraic equations. The work in the first part of this thesis is focused on the analysis of DAE models of cell signaling. In particular, we are interested in developing tools to characterize the qualitative behavior of the epidermal growth factor cell signaling network.

Growth factors are essential for mitogenesis. Over recent years there has been intense experimental investigation into the epidermal growth factor (EGF) family of receptors. There is experimental evidence to suggest that over-expression of these receptors is common in some cancers [138]. Furthermore, there is increasing evidence to suggest that epidermal growth factor signaling plays a key role in cancer [145, 196]. There is active research into EGF receptor tyrosine kinase inhibitors as potential anticancer agents [36, 6]. However, there is also evidence coming to light that suggests more detailed understanding of the role of EGF will be necessary to explain clinical results [26]. To complicate matters, there is evidence to suggest that the EGF receptor is active in both mitogenic and apoptopic signaling pathways [23, 217]. There has been much work on modeling Epidermal Growth Factor Receptor signaling (EGFR signaling) [146, 216, 174, 132, 22, 113, 61, 198] to try and understand the complex behavior of the signaling network.

Epidermal growth factor receptor (alternatively called HER1) is one of a class of four Human Epidermal growth factor Receptors (HER) [75]. The HER family is characterized by a ligand-binding domain with two cysteine rich regions, a single membrane spanning region, and a catalytic domain of approximately two hundred and fifty amino acids [234]. There are a variety of ligands that bind the HER family of receptors. Typically, the ligands are either synthesized as membrane precursors that are proteolytically cleaved to release a soluble polypeptide, or else function as membrane-anchored proteins in juxtacrine signaling [185].

Ligand binding causes activation of the intrinsic kinase activity of the EGF-receptor, leading to the phosphorylation of cellular substrates at tyrosine residues [40] and autophosphorylation of receptors [65, 66]. One of the ultimate effects of ligand binding is the activation of the MAPK enzyme as shown in Figure 1-2.

While the diagram suggests a clearly understood mechanism for MAPK activation, the reality is that only part of the mechanism is fully known. For example, the role of calcium in cell signaling is poorly understood [42]. Several different models have been proposed for the regulation of calcium [157, 98]. A calcium clamp technique has been developed that yields experimental results which suggest the information content contained in the calcium signal is frequency encoded [63]. It has also been shown that ligand affinity for the EGF receptor is not the only factor defining mitogenic potency. Studies comparing the mitogenic potency of transforming growth factor $\alpha$ (TGF$\alpha$) to the potency of EGF, suggest that a lower affinity ligand does not necessarily lead to a weaker response [181]. Ligand depletion effects [180] and differential receptor down regulation [181] both play an important role in defining the response of a cell to a signaling molecule. These competing effects have been exploited by producing a genetically modified ligand for the EGF receptor with a lower affinity, which elicits a greater mitogenic response [179].

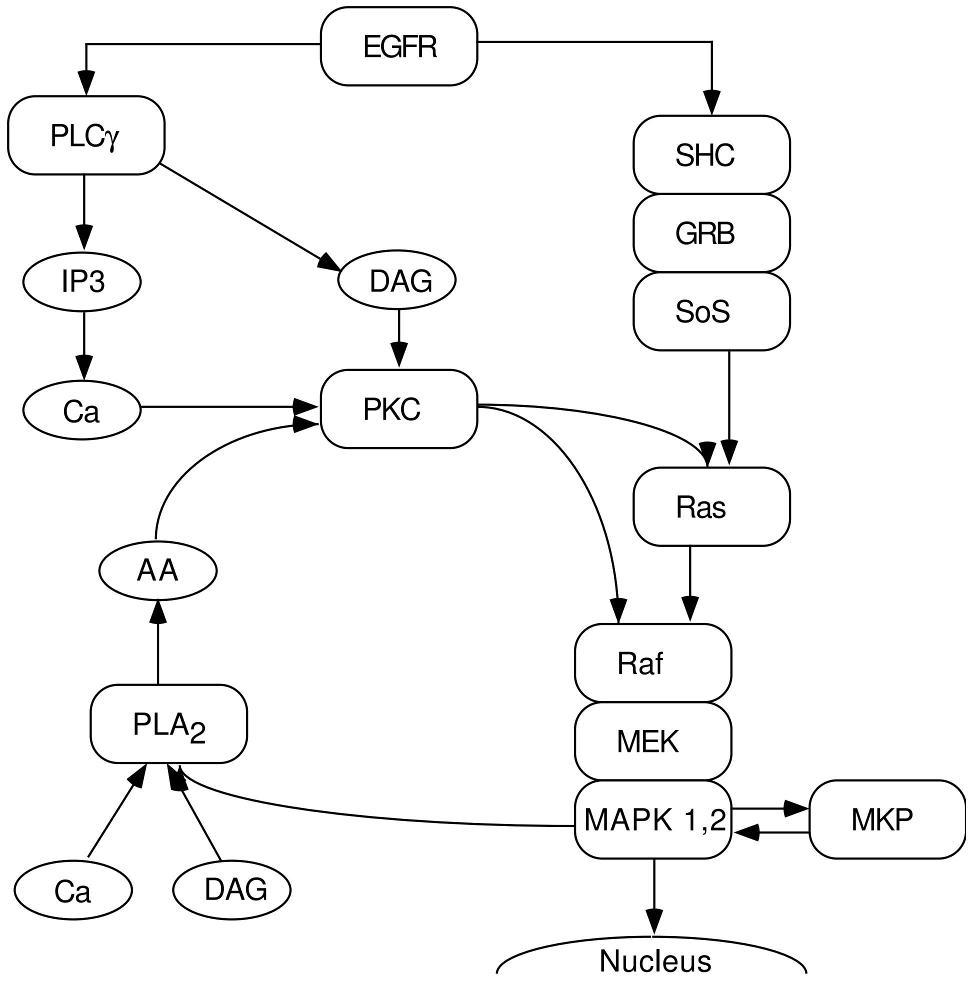


Figure 1-2: Mechanism of MAPK activation through the EGF receptor [22]

The level of complexity in the EGFR system together with competing interactions justify a hierarchical and quantitative approach to investigation [13]. Short term activation of $\mathrm{PLC}_\gamma$ and SOS by EGF has been modeled [132], although the model did not take into account trafficking effects. Mathematical modeling of cell signaling allows predictions to be made about cell behavior [141]. Typically, qualitative agreement between a mathematical model and experimental data is sought. For example, epidermal growth factor signaling has been investigated [22]. By constructing a kinetic model of the signaling network, it was possible to show that signals are integrated across multiple time scales, distinct outputs are generated depending on input strength and duration, and self-sustaining feedback loops are contained in the system. However, despite a wealth of modeling work, little attempt has been made to systematize the analysis of these signaling models. Work in the first part of this thesis is devoted to the systematic analysis of these models, exploiting control theoretic techniques.

### 1.3.1 Formulating Cell-Signaling Models

The first step in determining the qualitative behavior of a cell-signaling network model is to write a model of the system. Typically, it is important to know:

1. what level of fidelity is required of the model (abstract / statistical or detailed / mechanistic),
2. whether the physical behavior of interest is deterministic or stochastic,
3. whether the physical behavior of interest occurs at steady-state or whether the behavior is dynamic, and,
4. whether the biological system well-mixed or anisotropic.

A possible decision tree to decide what type of model is appropriate is shown in Figure 1-3.

Cell-signaling networks are often modeled in great detail and typically either a continuum or stochastic model is used. Modern physics casts doubt whether the

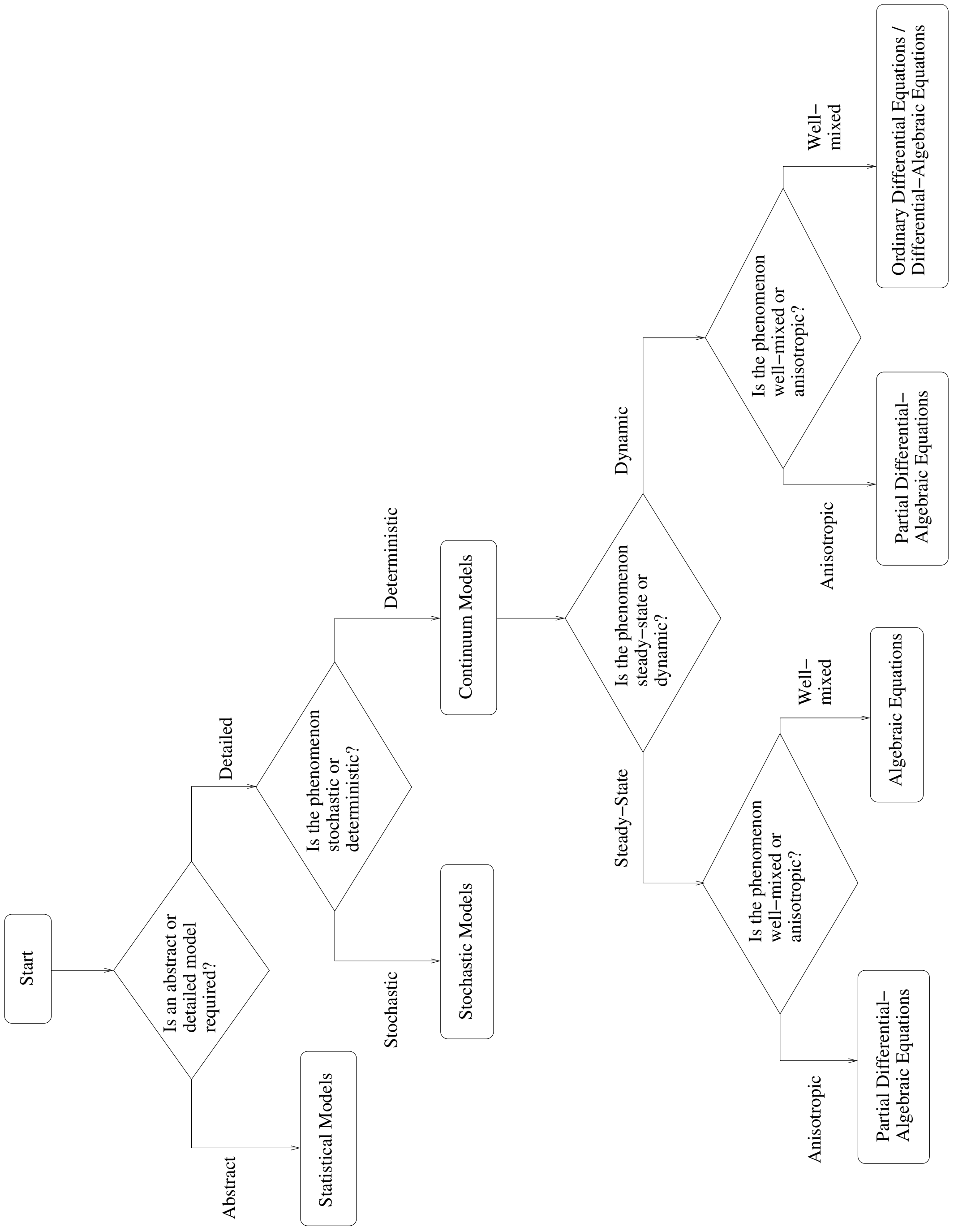

Start
Is an abstract or detailed model required?
Abstract
Statistical Models
Detailed
Is the phenomenon stochastic or deterministic?
Stochastic
Stochastic Models
Deterministic
Continuum Models
Is the phenomenon steady−state or dynamic?
Steady−State
Is the phenomenon well−mixed or anisotropic?
Anisotropic
Partial Differential−Algebraic Equations
Well−mixed
Algebraic Equations
Dynamic
Is the phenomenon well−mixed or anisotropic?
Anisotropic
Partial Differential−Algebraic Equations
Well−mixed
Ordinary Differential Equations / Differential−Algebraic Equations

Figure 1-3: Decision tree to decide appropriate model type

"laws of physics" are deterministic. However, effective deterministic behavior can be realized from a probabilistic system when average properties (concentration, temperature, etc.) are of interest. It is often far simpler and more desirable to model the effective behavior of the system rather than the small-scale detail. Depending on the situation, this can either be achieved by writing a less detailed probabilistic description of the system (a stochastic model) or else writing a deterministic continuum model.

A typical approach to modeling cell-signaling networks is to write conservation equations for each of the species of interest for a specified control volume. The conservation equations are usually an under-determined system of equations. It is therefore necessary to add additional equations and specify some variables to make a well-posed simulation. Typically, such equations would include algebraic equations specifying the rates of generation of species, algebraic relationships determining equilibrium constants from the thermodynamic state of the system, and algebraic relationships determining rate constants. Many biological systems are isothermal and little transfer of momentum occurs within the system. It is therefore common to neglect energy and momentum balances on the system of interest. Great care must be taken in selecting the appropriate control volume and accounting for any changes of volume of the system (see Chapter 2, § 2.1 for a more detailed explanation of potential errors). When a species has more than one state, (for example: the phosphorylation state of a protein) typically two or more variables are introduced to represent the quantity of molecules in each state. The abstract form of the species conservation equations can be written in terms of basic processes:

$$\text{Rate of Accumulation} = \text{Flow In} - \text{Flow Out} + \text{Rate of Generation}. \tag{1.1}$$

This abstract model can be converted into either a continuum model or a stochastic model. A crude statement of when a system is effectively deterministic is written mathematically as:

$$\frac{\sqrt{\mathrm{Var}(x)}}{|\mathrm{E}(x)|} \ll 1, \tag{1.2}$$

where Var$(x)$ is the variance of $x$ and E$(x)$ is the expected value of $x$. The ratio can interpreted as roughly the deviation of $x$ between experiments run at constant conditions divided by the average value of $x$ over all the experiments. Clearly, if the deviation of $x$ is small then one is confident in predicting the value of $x$ and the system is effectively deterministic. These systems can be safely modeled using the continuum approximation. The variables in a continuum model represent the average or expected value of a quantity (for example: concentration, velocity, etc.). The quantities of interest are represented by real numbers (rather than integer numbers) and the basic processes: accumulation, flow in, flow out, and generation occur smoothly. Clearly, the continuum model is an approximation of a cell signaling network as molecules occur in integer amounts. Hence, the continuum model is only appropriate when the quantity of molecules of interest is sufficiently high (see [56, 55] for a discussion of how stochastic effects in ligand binding are governed by cell-receptor number). If the condition in Equation (1.2) does not hold, then it is more appropriate to write a probability based model.

For systems where the quantity of species is low, it may not be appropriate to use a continuum model and instead a stochastic model of the system should be used (see [12] for an example of a stochastic simulations of a cell-signaling network). This approach is a more faithful representation of the system. However, the increased fidelity comes at some cost. Stochastic simulations are computationally more challenging to solve and the results of one simulation only represent one of many possible realizations of the system. To determine average behavior requires many simulations to be performed.

### 1.3.2 Continuum Models of Cell-Signaling

It can be seen from the decision tree shown in Figure 1-3 that continuum models represent several different classes of models. If there is no accumulation of material in the control volume (the left hand side of Equation (1.1) is identically zero), the system is at steady-state and all of the variables of interest hold a constant value with respect to time. Such a system is at equilibrium. It is quite common to manipulate the conditions of an *in vitro* experiment to try to achieve equilibrium (for example: a ligand binding assay to determine a binding affinity, $K_a$). In contrast, if there is accumulation of material within the control volume, the values of the system variables will change with respect to time (the system shows transient or dynamic behavior). This scenario is more common when examining the regulatory structure of a mammalian cell.

If the whole of a steady-state system is well-mixed, the abstract conservation equations reduce to the familiar form:

$$0 = \text{Flow In} - \text{Flow Out} + \text{Rate of Generation}. \tag{1.3}$$

Each of the terms in Equation (1.3) can be represented by algebraic terms and the resulting model equations are purely algebraic. If the system is not well-mixed, it is necessary to model the spatial variations of quantities of interest (concentration, electric field, etc.). There are two different approaches depending on the fidelity required in the model:

1. a compartment approach where the control volume is sub-divided into a finite set of well-mixed control volumes and flux equations are written to describe the flow of material between the compartments, and,

2. a partial differential-algebraic equation (PDAE) approach where the conservation equations are derived for an infinitesimal element of the control volume and boundary conditions are used to describe the flux of material to and from the control volume.

The correct approach depends on the degree of fidelity required and whether there are physically distinct regions. Typically, PDAEs are more difficult to solve and require more *a priori* knowledge. A compartment approach may be more appropriate if there are physically distinct regions within the overall control volume (for example: organelles such as the endosome and lysosome inside a cell).

However, it is more common to find that the transient behavior of a cell signaling network is of interest. Again, it is important to know whether the control volume is well-mixed. If the system is not well-mixed the choices are again to formulate a compartment model or else write a PDAE model. Biological examples of the compartment model approach to modeling cell signaling include [81, 197]. Examples of PDAE models include [112]. It is usually simpler to formulate a compartment model since the model represents a collection of well-mixed control volumes with flux equations to describe the flow of material between the control volumes.

If it is appropriate to model the control volume as a well-mixed region, the rate of accumulation term in Equation (1.1) is represented by a total derivative:

$$\text{Rate of Accumulation} = \frac{\mathrm{d}\,(\text{Quantity})}{\mathrm{d}t}. \tag{1.4}$$

It should be stressed that only *extensive* quantities such as mass, number of moles of a species, internal energy, etc, are conserved. *Intensive* quantities such as concentration are only ever conserved under very restrictive assumptions. Unfortunately, it is almost *de rigour* in the biological simulation literature to formulate models in terms of *intensive* properties. This approach has three main disadvantages: it is not clear what assumptions were used to formulate the original model, it is often an error prone task to convert a model written in terms of *extensive* quantities into a model written in terms of *intensive* quantities, and finally it is not clear to which control volume the equation applies. A common example of a conservation equation written in terms of *intensive* quantities might be:

$$\frac{\mathrm{d}C_{EGF}}{\mathrm{d}t} = k_f C_{EGF} C_{EGFR} - k_r C_{EGF-EGFR} \tag{1.5}$$

where $C_{EGF}$, $C_{EGFR}$, and $C_{EGF-EGFR}$ are the concentrations of EGF, the EGF receptor, and EGF bound to the EGF receptor. However, concentration is in general *not* a conserved quantity and it is not clear in which control volume the concentration is measured (media bulk, endosome, lysosome, cytoplasm, cell membrane, nucleus, etc.). The advantage of formulating the conservation equations in terms of *intensive* quantities is that it results in a system of ordinary differential equations. Instead, it is preferable to write the abstract species conservation directly:

$$\frac{\mathrm{d}N_{EGF}^{BULK}}{\mathrm{d}t} = Fin_{EGF}^{BULK} - Fout_{EGF}^{BULK} + R_{EGF}^{BULK} \tag{1.6}$$

where $N_{EGF}$ is the number of moles of EGF in the bulk, $Fin_{EGF}^{BULK}$ and $Fout_{EGF}^{BULK}$ are the respective flow terms of EGF into and out of the bulk and $R_{EGF}^{BULK}$ is the rate of generation of EGF in the bulk. The terms in Equation (1.6) must then be specified with additional algebraic equations. This results in a system of differential-algebraic equations. Historically, the simulation of DAEs has not been widespread and it has been perceived that it is a computationally challenging task. Consequently, much teaching has focused on ODE formulations of biological simulations. However, with modern computers and sophisticated DAE process simulators (ABACUSS II [41, 230], gPROMS [16], SPEEDUP (now Aspen Custom Modeler) [167]) it is not true that DAE simulations are unduly difficult to solve; simulations and sensitivity analyses of many thousands of equations can be computed in seconds or minutes.

Several different objectives can be achieved once the simulation has been formulated. For example, a clinician may be interested in the results of a simulation for a specific set of input conditions. In contrast, a researcher is probably more interested in characterizing the behavior of the network. To characterize the behavior of the network either requires a large number of simulations over a set of different input conditions or some other mathematical way of characterizing the system [82]. The control literature has developed sophisticated techniques to analyze systems of ordinary differential equations. Many of these approaches require the construction of an linear, time-invariant, explicit ODE approximation to the original systems of equations around an equilibrium point. The approximation is valid for sufficiently small perturbations of the system around the equilibrium point. There is a well-developed theory for describing the behavior state-space models (summarized in Chapter 2, § 2.2). Typically, the state-space approximation to the original ODE is constructed by hand. It would be a monumental task to construct such an approximation to a DAE by hand. Instead, we have developed a technique to generate such an approximation automatically (Chapter 2).

## 1.4 Mammalian Cell Migration

Work in the second part of this thesis is devoted to developing statistical techniques to analyze cell migration data. Cell migration plays a critical role in inflammation, wound healing, embryogenesis, and tumor cell metastasis [233]. Consequently, there have been numerous investigations of the *in-vitro* migration of mammalian cells (see for example [8, 56, 91, 93, 97, 143, 162, 166, 172, 184, 237, 242]). However, a key step in these studies is to quantify different types of cell migration behavior. The objective of this work is to compare cell migration tracks to see if there is a quantifiable difference between the migration of cells under different input conditions (e.g., ligand concentration, cell medium, cell substrate, cell type, etc.). This step is crucial to elucidate how cell migration occurs and what external influences control cell migration.

A mammalian cell touches the substrate at distinct areas called focal adhesions. Focal adhesions span the cell-surface membrane and contain a complex assembly of cell surface receptor proteins, internal cell signaling proteins, actin polymer fibers and other cyto-skeletal proteins [34]. The structures and mechanisms of focal adhesions are not completely understood, although experimental evidence suggests that focal adhesions are important for cell-substrate traction, mechanical properties of the cell and cell receptor signaling. Traction between the cell and the base surface is mediated through receptor-ligand interactions (often integrin receptor-fibronectin) and non-specific binding. A very simplified schematic of a receptor mediated binding at a focal adhesion is shown in Figure 1-4. The effect of integrin receptor-fibronectin interaction on cell migration has been investigated [140, 58, 166]. Experimental evidence has verified computational work suggesting that cell migration speed is a biphasic function of substrate fibronectin concentration [166].

Cell migration is often broken into four distinct phases: extension, adhesion, translocation and de-adhesion (shown in Figure 1-5) [144]. Continuous mammalian

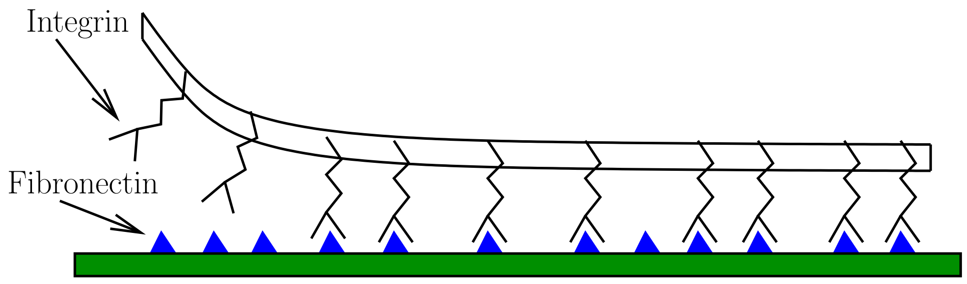


Figure 1-4: Simplified schematic of a focal adhesion [34]

cell migration is characterized by polarization of the cell and formation of a dominant leading lamella [8, 143], although it is not unusual to see growth of competing lamellae for certain cell types [156]. It is widely hypothesized that cell migration is driven by the following basic molecular processes: actin assembly, actin disassembly, cyto-skelatal reorganization, and contractile force generation. Assembly of actin filaments occurs preferentially at the tip of the lamella and disassembly occurs at the base [88, 222]. Although many of the basic molecular processes responsible for cell migration have been characterized, their physiological regulation and mechanical dynamics are not well understood [8]. For example, the following mechanisms have been proposed for controlling the orientation of actin fiber assembly: transient changes in actin polymerization [79, 209], protrusion of the cell membrane due to osmotic swelling [45, 30], Brownian movement of the cell membrane [171], detachment of membrane and actin filaments by action of myosin I [201, 83], and a combination of hydrostatic pressure and mechanical tension controlling the local dynamics of microfilament alignment [133, 134, 31].

The first phase of cell migration is polarization of the cell followed by extension of one or more lamellae through actin polymerization (extension). Ultimately, only one lamella will be stable (defining the direction of motion) if more than one lamellae are extended. A new focal adhesion forms at the tip of a stable lamella once the lamella is fully extendend (adhesion). After a stable focal adhesion has formed at the tip, the nucleus moves to the new center of the cell (translocation) by forces exerted on the nucleus through the cyto-skeleton. The final step is de-adhesion of the rear focal adhesion and adjustment of the cell membrane location (de-adhesion). Extensive work has been done to investigate regulation of some of the individual steps of cell migration: polarization and lamellae extension (for example: [175, 177, 239]), formation of focal adhesions (for example: [34, 57, 241]), translocation of cell (for example: [187, 238]), and de-adhesion of the rear focal adhesion (for example: [24, 52]). Ultimately, it is desired to characterize the effect of external conditions on physiological behavior, i.e., characterizing the response of the cell. However, there is currently insufficient information to perform detailed modeling of cell migration to the point where cell motion can be predicted. This has motivated the use of less detailed models as described in § 1.4.1.

### 1.4.1 Random-Walk Models of Cell Migration

Much of the research cited in § 1.4 focuses on determining the molecular mechanism of cell migration and regulation of migration. The ultimate goal is to be able to use *in vitro* experimental work to make predictions about *in vivo* physiological behavior. For example, it is known that over-expression of the HER2 receptor in mammary epithelial cells correlates with a poor prognosis for sufferers of breast cancer [211]. It is also known that often breast cancer cells metastasize within the body. Two questions naturally arise:

1. does over-expression of HER2 cause cancer cell migration?

2. if a chemical is found that blocks *in vitro* HER2 induced cell migration will this be an effective anti-cancer drug?

To answer either of these questions requires characterization of *in vitro* cell migration in a way that is relevant for *in vivo* predictions. To achieve this goal, it is desirable to correlate cell migration to external stimuli or cell abnormalities.

Typically, cell migration is characterized by time-lapse video microscopy. However, the researcher is then faced with the challenge of distinguishing between two different sets of time-lapse data. Ideally, it would be possible to write a mechanistic model that would predict cell motion as a function of external inputs. However, there are currently two difficulties with this approach: the regulatory structure of cell migration is not known in sufficient detail, and it is hypothesized that cell migration is driven by stochastic processes.

Instead, it has been proposed to characterize cell migration paths in terms of a small number of physical parameters that can then be correlated with external inputs. The work of [91] was one of the first to model cell migration as a random walk. In this work, the locomotion of mouse fibroblasts in tissue culture was observed at 2.5hr and 5hr time intervals. Cell migration paths were modeled as a Brownian diffusion and a correlated random walk. The Brownian diffusion model has a single parameter: diffusivity, $D$. In contrast, the correlated random walk model has two parameters: augmented diffusivity, $D^*$, and persistance tendency, $\rho$. Indeed, there are a large number of different random walk models that can be used to model cell migration. A comprehensive review of the different models is [165].

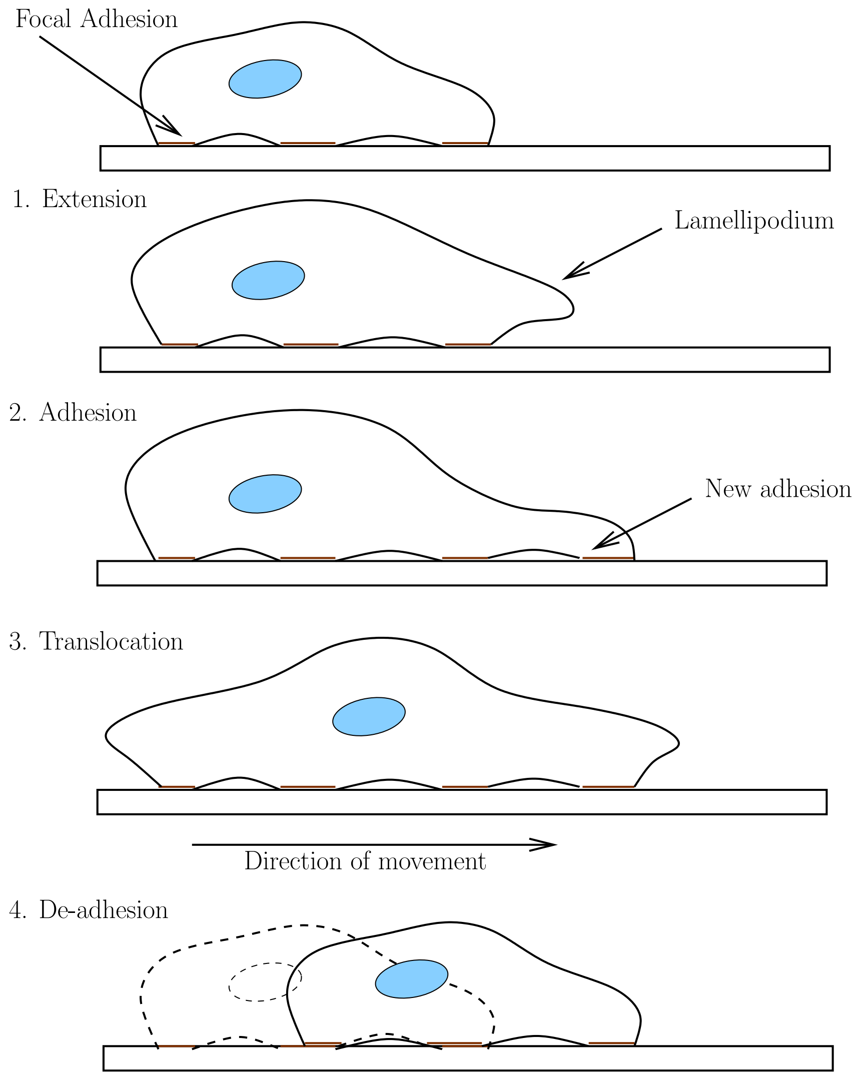


Figure 1-5: Steps in polarized keratinocyte movement (see Page 788 of [144])

# Chapter 2

# Detailed Modeling of Cell-Signaling Pathways

Frequently, cell-signaling networks are modeled by writing conservation equations for each of the signaling species, resulting in a large set of nonlinear differential-algebraic equations (DAEs) that are sparse and unstructured (as discussed in § 2.1). The goal of writing the model is to analyze the behavior of the network and predict suitable interventions. The implicit nonlinear model may be unsuitable for this task since it is difficult to analyze an implicit nonlinear model systematically. The alternative to systematic analysis of the original implicit nonlinear model is the simulation of a large number of scenarios. However, this can be quite time consuming [82].

Two methods have been advocated for constructing an explicit linear model from a set of nonlinear DAEs:

1. process identification, and

2. linearization of the original nonlinear model and rearrangement.

However, it is easy to construct an explicit linear model by process identification, which has the correct open loop behavior, but has qualitatively different closed loop behavior [119]. Consequently, the second approach of linearizing the original model appears attractive. A method for this task has been proposed by [240] and implemented in the Control Data Interface of Aspen Custom Modeler[1] and SpeedUp[2]. It has been found that this method is inadequate for larger systems [102].

For typical cell signaling models, the corresponding state-space matrices are sparse (see § 2.6.1). Sparsity of the state-space model can be exploited in further calculations; e.g., eigenvalue calculations and identifying right-half plane poles [192, 193, 73, 74, 199]. Furthermore, many of the algebraic variables that appear in the original nonlinear DAE are not of interest. For example, it is likely that an experimentalist would be interested in the concentrations of key signaling molecules, but not so interested in the value of fluxes of molecules due to trafficking and reaction. Another example might be the semi-discretization of a set of partial differential-algebraic equations (PDAEs) where algebraic equations are introduced during the discretization. It is therefore desirable to construct a smaller linear state-space model from the original large-scale nonlinear DAE. In this situation, it may be necessary to retain only a limited subset of the algebraic variables defined in the original model. Conventional stability and controllability analysis can be applied to the resulting linearized model to make qualitative statements about the original DAE [210].

## 2.1 Formulation of Cell-Signaling Models

Writing an accurate model of a cell-signaling system is the first step in performing a mathematical analysis of the system properties. Conventionally, cell-signaling models have been written as systems of ordinary differential equations (for example: [81, 197]). This approach to modeling has some potential pitfalls which will be demonstrated in this Section. In particular, it can be difficult to correctly formulate an ODE model for a system that does not have constant volume. Instead, we advocate writing such models as DAEs. The general form for such a cell-signaling model is:

$$\mathbf{f}(\mathbf{x}', \mathbf{x}, \mathbf{y}, \mathbf{u}) = \mathbf{0}, \quad \mathbf{f} : \mathbb{R}^{n_x} \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_y} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_x + n_y} \tag{2.1}$$

[1]Aspen Custom Modeler is a registered trademark of Aspen Technology, Cambridge, MA.

[2]SpeedUp was a process simulator developed at Imperial College, London and marketed by Aspen Technology.

where $\mathbf{x}(t) \in \mathbb{R}^{n_x}$ are the states of the system, $\mathbf{y}(t) \in \mathbb{R}^{n_y}$ are the algebraic variables, and $\mathbf{u}(t) \in \mathbb{R}^{n_u}$ are the inputs to the system at time $t$. It is more challenging computationally to solve systems of DAEs compared to systems of ODEs. However, with modern computers and sophisticated DAE process simulators (ABACUSS II [41, 230], gPROMS [16], SPEEDUP (now Aspen Custom Modeler) [167]) it is not true that DAE simulations are unduly difficult to solve; simulations of many thousands of equations can be computed in seconds or minutes. Indeed, some process simulators (for example: ABACUSS II [41, 230]) will even automatically generate sensitivities of the state and output vectors with respect to model parameters.

### 2.1.1 ODE Model of IL-2 Receptor Trafficking

It is instructive to analyze an example trafficking model [81] to illustrate some of the pitfalls in directly approximating the species conservation equations as an ODE. It should be stressed that the authors' model is a fairly close approximation to the DAE conservation equations for the range of conditions investigated (the error of the approximation is certainly less than the error due to parametric uncertainty). However, the error might not be so small for alternative conditions or parameter values. The goal of the work of [81] was to model the effect of molecular binding and trafficking events of interleukin-2 (IL-2) on cell proliferation. A schematic of the system is shown in Figure 2-1. The model proposed by the authors is shown in Equations (2.2)–(2.9). The notation and parameter values are summarized in Tables 2.1–2.2.

Receptor balance at cell surface:

$$\frac{\mathrm{d}R_s}{\mathrm{dt}} = V_s + k_r C_s + k_{syn} C_s - k_t R_s - k_f R_s L. \tag{2.2}$$

Ligand-receptor complex balance on cell surface:

$$\frac{\mathrm{d}C_s}{\mathrm{dt}} = k_f R_s L - k_r C_s - k_e C_s. \tag{2.3}$$

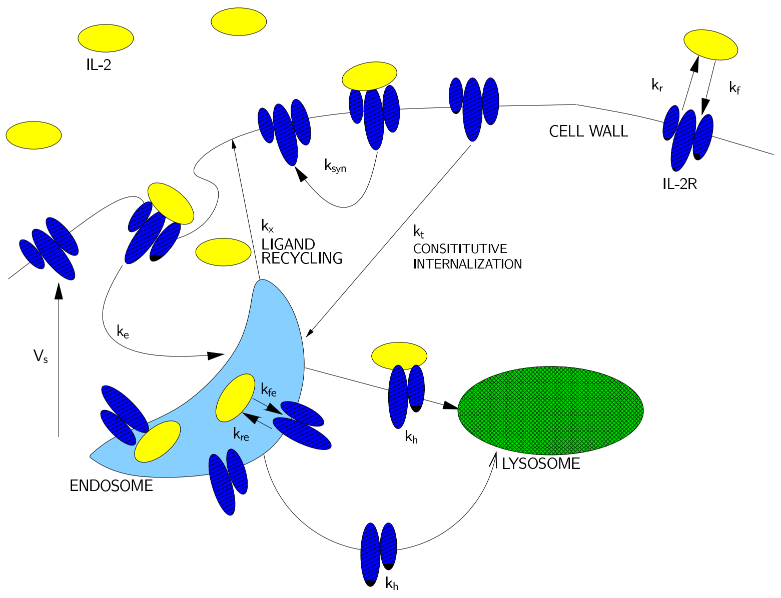


Figure 2-1: Schematic of interleukin-2 receptor-ligand trafficking

Receptor balance on endosome:

$$\frac{\mathrm{d}R_i}{\mathrm{dt}} = k_{re}C_i + k_tR_s - k_{fe}R_iL_i - k_hR_i. \tag{2.4}$$

Ligand-receptor complex balance on endosome:

$$\frac{\mathrm{d}C_i}{\mathrm{dt}} = k_eC_s + k_{fe}R_iL_i - k_{re}C_i - k_hC_i. \tag{2.5}$$

Ligand balance on endosome:

$$\frac{\mathrm{d}L_i}{\mathrm{dt}} = \frac{k_{re}C_i - k_{fe}R_iL_i}{V_eN_A} - k_xL_i. \tag{2.6}$$

Ligand balance on bulk medium:

$$\frac{\mathrm{d}L}{\mathrm{dt}} = Y\frac{k_rC_s - k_fR_sL}{N_A} + Yk_xV_eL_i. \tag{2.7}$$

Table 2.1: IL-2 trafficking parameters

| Parameter | Definition | Value |
|---|---|---|
| $k_r$ | dissociation rate constant | $0.0138\ \text{min}^{-1}$ |
| $k_f$ | association rate constant | $k_r/11.1\ \text{pM}^{-1}$ |
| $k_{re}$ | dissociation rate constant, endosome | $8k_r\ \text{min}^{-1}$ |
| $k_{fe}$ | association rate constant, endosome | $k_{re}/1000\ \text{pM}^{-1}$ |
| $k_t$ | constitutive receptor internalization rate constant | $0.007\ \text{min}^{-1}$ |
| $V_s$ | constitutive receptor synthesis rate | $11\ \#\ \text{cell}^{-1}\ \text{min}^{-1}$ |
| $k_{syn}$ | induced receptor synthesis rate | $0.0011\ \text{min}^{-1}$ |
| $k_e$ | internalization rate constant | $0.04\ \text{min}^{-1}$ |
| $k_x$ | recycling rate constant | $0.15\ \text{min}^{-1}$ |
| $k_h$ | degradation rate constant | $0.035\ \text{min}^{-1}$ |
| $V_e$ | total endosomal volume | $10^{-14}$ liter $\text{cell}^{-1}$ |
| $N_A$ | Avogadro's number | $6 \times 10^{11} \#\ (\text{pico mole})^{-1}$ |

Empirical cell growth rate relationship:

$$\frac{\mathrm{d}Y}{\mathrm{dt}} = \max\left(\frac{600C_s}{250 + C_s} - 200, 0\right) \times 10^3. \tag{2.8}$$

Concentration of ligand destroyed in lysosome:

$$\frac{\mathrm{d}L_d}{\mathrm{dt}} = \frac{k_h C_i}{V_e N_A}. \tag{2.9}$$

However, as already stated in Chapter 1, concentration is not generally a conserved quantity. The model equations are only strictly valid if the total volume of cells remains unchanged. The authors' model is an approximation to an underlying DAE model. The fact that strict conservation does not occur is illustrated by adding Equation (2.10), which tracks the total ligand concentration in bound, unbound and destroyed forms:

$$L_T = L + \frac{YC_s}{N_A} + \frac{YC_i}{N_A} + V_e Y L_i + V_e Y L_d. \tag{2.10}$$

An ABACUSS II simulation of the system is shown in Figure 2-2. In fact, it is always worth adding such an equation to a cell-signaling model as it will often reveal mistakes

Table 2.2: IL-2 trafficking nomenclature

| Variable | Definition | Units |
|---|---|---|
| $R_s$ | Number of unbound receptors on the cell surface | # cell$^{-1}$ |
| $Y$ | Cell density | # liter$^{-1}$ |
| $C_s$ | Number of ligand-receptor complexes on the cell surface | # cell$^{-1}$ |
| $L$ | Bulk concentration of unbound ligand | pM |
| $R_i$ | Number of unbound receptors in the endosome | # cell$^{-1}$ |
| $C_i$ | Number of ligand-receptor complexes in the endosome | # cell$^{-1}$ |
| $L_i$ | Concentration of unbound ligand in the endosome | pM |
| $L_d$ | Concentration of ligand destroyed in lysosome | pM |
| $L_T$ | Total ligand concentration in bound and unbound forms | pM |

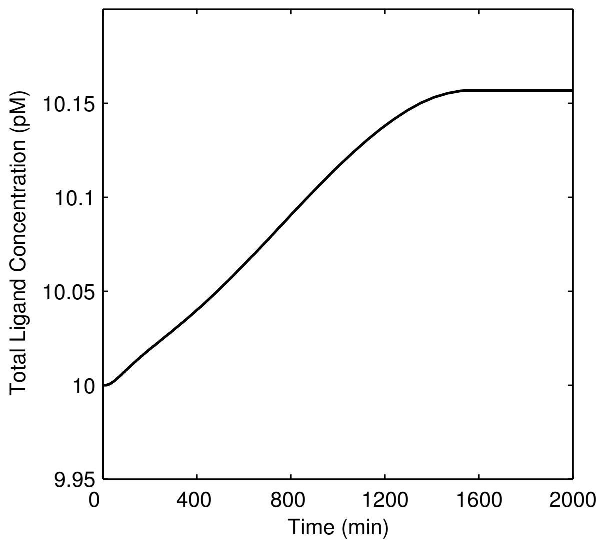


Figure 2-2: Simulation results for ODE IL-2 trafficking model

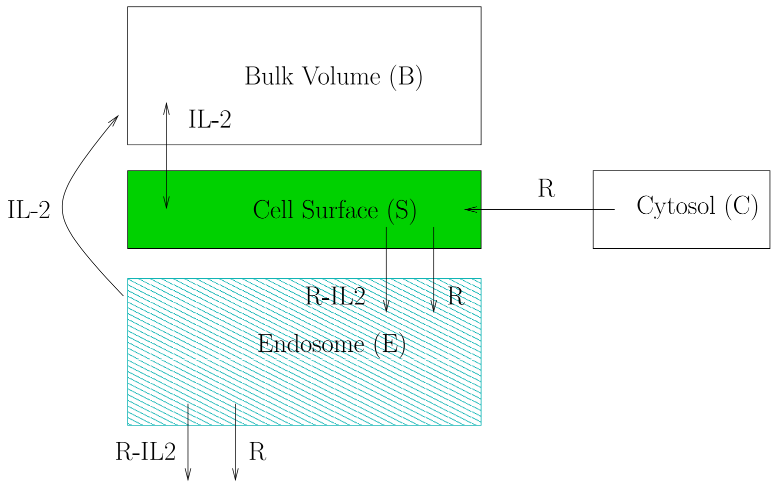


Figure 2-3: Regions of accumulation for IL-2 trafficking model

(incorrect unit conversions, mistaken assumptions, etc.). The code for the simulation is shown in Appendix B, § B.1. The total concentration of IL-2 ligand should remain constant at 10pM. However, it can be seen from the plot that there is roughly a 1.5% increase in ligand concentration; i.e., the model equations do not enforce conservation of mass. It should be emphasized that for this particular example the discrepancy in the mass balance is small so it does not alter the conclusions of the study.

### 2.1.2 Reformulated DAE Model of IL-2 Receptor Trafficking

It is preferable to formulate the model equations as a system of DAEs. The regions of accumulation of material are shown in Figure 2-3. The corresponding model is shown in Equations (2.11)–(2.36):

Empirical cell growth rate relationship:

$$\frac{\mathrm{d}Y}{\mathrm{dt}} = \max\left(\frac{600C_s}{250+C_s} - 200, 0\right) \times 10^3. \tag{2.11}$$

Ligand balance on bulk medium (constant volume):

$$\frac{\mathrm{d}L}{\mathrm{dt}} = F_L^{S\to B} - F_L^{B\to S} + F_L^{E\to B}. \tag{2.12}$$

Receptor balance on cell surface (volume is not constant):

$$\frac{\mathrm{d}N_R^S}{\mathrm{dt}} = F_R^{C\to S} - F_R^{S\to E} + r_R^S. \tag{2.13}$$

Complex balance on cell surface (volume is not constant):

$$\frac{\mathrm{d}N_C^S}{\mathrm{dt}} = r_C^S - F_C^{S\to E}. \tag{2.14}$$

Receptor balance on endosome (volume is not constant):

$$\frac{\mathrm{d}N_R^E}{\mathrm{dt}} = F_R^{S\to E} + r_R^E. \tag{2.15}$$

Complex balance on endosome (volume is not constant):

$$\frac{\mathrm{d}N_C^E}{\mathrm{dt}} = F_C^{S\to E} + r_C^E. \tag{2.16}$$

Ligand balance on endosome (volume is not constant):

$$\frac{\mathrm{d}N_L^E}{\mathrm{dt}} = -F_L^{E\to B} + r_L^E. \tag{2.17}$$

Ligand destroyed in endosome (volume is not constant):

$$\frac{\mathrm{d}N_L^D}{\mathrm{dt}} = r_L^D. \tag{2.18}$$

Ligand flux from surface to bulk:

$$F_L^{S\to B} = \frac{Yk_rC_s}{N_A}. \tag{2.19}$$

Ligand flux from bulk to surface:

$$F_L^{B\to S} = \frac{Yk_fR_sL}{N_A}. \tag{2.20}$$

Ligand flux from endosome to bulk:

$$F_L^{E\to B} = Yk_xV_eL_i. \tag{2.21}$$

Receptor flux from cytosol to surface:

$$F_R^{C\to S} = YV_s. \tag{2.22}$$

Receptor flux from surface to endosome:

$$F_R^{S\to E} = Yk_tR_s. \tag{2.23}$$

Generation of free receptors at the surface:

$$r_R^S = Y\left(k_{syn}C_s + k_rC_s - k_fR_sL\right). \tag{2.24}$$

Generation of ligand-receptor complexes at surface:

$$r_C^S = Y\left(k_fR_sL - k_rC_s\right). \tag{2.25}$$

Complex flux from surface to endosome:

$$F_C^{S\to E} = Yk_eC_s. \tag{2.26}$$

Generation of free receptors in the endosome:

$$r_R^E = Y\left(k_{re}C_i - k_{fe}R_iL_i - k_hR_i\right). \tag{2.27}$$

Generation of ligand-receptor complexes in the endosome:

$$r_C^E = Y\left(k_{fe}R_iL_i - k_{re} - k_hC_i\right). \tag{2.28}$$

Generation of free ligand in the endosome:

$$r_L^E = \frac{Y\left(k_{re}C_i - k_{fe}R_iL_i\right)}{N_A}. \tag{2.29}$$

Rate of ligand destruction in the endosome:

$$r_L^D = Y\frac{k_hC_i}{V_eN_A}. \tag{2.30}$$

Total number of receptors on cell surface:

$$N_R^S = YR_s. \tag{2.31}$$

Total number of complexes on cell surface:

$$N_C^S = YC_s. \tag{2.32}$$

Total number of receptors in the endosome:

$$N_R^E = YR_i. \tag{2.33}$$

Total number of complexes in the endosome:

$$N_C^E = YC_i. \tag{2.34}$$

Total number of ligands in the endosome:

$$N_L^E = YL_i. \tag{2.35}$$

Total number of ligands destroyed in the endosome:

$$N_L^D = YL_d. \tag{2.36}$$

The corresponding ABACUSS II simulation is shown in Appendix B, § B.2. At first glance, it may appear more cumbersome to write the model in the form proposed in Equations (2.11)–(2.36) compared with the ODE form in Equations (2.2)–(2.9). It is true that the ODE model is smaller. However, experience shows that it is easier to make mistakes (incorrect units, etc.) when the fluxes are not written out explicitly. It is certainly possible to convert the DAE model (Equations (2.11)–(2.36)) into an ODE model by substitution of equations and application of the chain rule. For example, combining Equations (2.11), (2.14), (2.32), (2.25) and (2.26) yields:

$$\frac{\mathrm{d}C_s}{\mathrm{dt}} = -\frac{C_s}{Y}\max\left(\frac{600C_s}{250+C_s}-200,0\right)\times 10^3 + k_f LR_s - \left(k_r + k_e\right)C_s.$$

It should be emphasized that little benefit is gained from this additional (and therefore potentially error prone) step since the original DAEs can be simulated with ease.

## 2.2 Properties of Explicit ODE Models

As shown in § 2.1, the original DAE cell-signaling model can be written according to Equation (2.1). However, it is desirable to summarize the qualitative behavior of such a system, thus allowing general statements to be made about the cell signaling model. To begin the discussion, the properties of ODE models will be examined in this Section.

### 2.2.1 Linear Time-Invariant ODE Models

Typically, cell-signaling models cannot be formulated as linear time-invariant ODEs since most reaction networks include bimolecular reactions of the form:

$$L + R \leftrightharpoons C.$$

The species conservation equations for such a system are bilinear. Under some conditions (for example: excess ligand) the system can be approximated by a linear pseudo first-order reaction. In general, this approximation does not hold. However, it is useful to present some results about linear time-invariant ODEs as the theory of such systems underpins the understanding of nonlinear ODE systems.

An *autonomous* linear time-invariant ODE has the form:

$$\mathbf{x}'(t) = \mathbf{A}\mathbf{x}(t)\,, \tag{2.37}$$

where $\mathbf{A}$ is a constant matrix. The system is autonomous as the independent variable, $t$, does not appear explicitly. Typically, cell signaling models are autonomous. The solution to the linear system of ODEs given in Equation (2.37) is

$$\mathbf{x}(t; \mathbf{x}_0, t_0) = \left[\exp \mathbf{A}\,(t - t_0)\right] \mathbf{x}_0. \tag{2.38}$$

Frequently, one is concerned with the response of the system to perturbations around steady-state. Steady-state occurs when there is no accumulation of material, energy, or momentum in the control volume. This corresponds to the condition:

$$\mathbf{x}'(t) = \mathbf{0}. \tag{2.39}$$

A value of $\mathbf{x}$ which causes Equation (2.39) to be satisfied is an *equilibrium* point. Clearly, $\mathbf{x} = \mathbf{0}$ is an equilibrium point for the linear system of ODEs given in Equation (2.37). It is natural to ask how the system responds to perturbations in the initial condition around the zero state. In particular, it is interesting to determine whether the states remain bounded for bounded perturbations in the initial condition. Definitions 2.2.1–2.2.2 are used to describe the solution behavior of dynamic systems formally.

**Definition 2.2.1.** (Page 370 of [243]) The zero state $\mathbf{x} = \mathbf{0}$ of is said to be *stable in the sense of Lyapunov*, if for any $t_0$ and any $\epsilon > 0$, there is a $\delta > 0$ depending on $\epsilon$ and $t_0$ such that

$$||\mathbf{x}_0|| < \delta \Rightarrow ||\mathbf{x}(t; \mathbf{x}_0, t_0)|| < \epsilon \quad \forall t \geq t_0.$$

**Definition 2.2.2.** (Page 371 of [243]) The zero state $\mathbf{x} = \mathbf{0}$ is said to be *asymptotically stable* if

1. it is Lyapunov stable, and,
2. for any $t_0$, and for any $\mathbf{x}_0$ *sufficiently close to* $\mathbf{0}$, $\mathbf{x}(t; \mathbf{x}_0, t_0) \to \mathbf{0}$ as $t \to \infty$.

It is straightforward to characterize the solution behavior of a linear time-invariant ODE as the closed-form solution is known (Equation (2.38)). It is necessary to define the minimal polynomial:

**Definition 2.2.3.** (Page 593 of [243]) Given the polynomial:

$$p(\lambda) = \sum_{k=0}^{N} a_k \lambda^k$$

the matrix $p(\mathbf{A})$ is the matrix equal to the polynomial

$$p(\mathbf{A}) = \sum_{k=0}^{N} a_k \mathbf{A}^k,$$

where $\mathbf{A}^0 = \mathbf{I}$. The minimal polynomial of the matrix $\mathbf{A}$ is the polynomial $\psi(\lambda)$ of least degree such that $\psi(\mathbf{A}) = \mathbf{0}$ and the coefficient of the highest power of $\lambda$ is unity.

The following Theorems relate the stability of the solution $\mathbf{x}(t)$ to the eigenvalues of $\mathbf{A}$.

**Theorem 2.2.1.** *[243] The system described by Equation (2.37) is Lyapunov stable* ***iff***

1. *all of the eigenvalues of* $\mathbf{A}$ *have nonpositive real parts, and,*
2. *those eigenvalues of* $\mathbf{A}$ *that lie on the imaginary axis are simple zeros of the* minimal polynomial *of* $\mathbf{A}$.

*Proof.* See Pages 375–376 of [243]. □

**Theorem 2.2.2.** *[243] The system described by Equation (2.37) is asymptotically stable* ***iff*** *all of the eigenvalues of* $\mathbf{A}$ *have negative* $(< 0)$ *real parts.*

*Proof.* See Pages 375–376 of [243]. □

### 2.2.2 Nonlinear ODE Models

A system of differential equations in which the independent variable, $t$, does not occur explicitly,

$$\mathbf{x}' = \mathbf{f}(\mathbf{x}) \, ,$$

is *autonomous.* Any system $\mathbf{x}' = \mathbf{f}(t, \mathbf{x})$ can be considered autonomous if the vector $\mathbf{x}$ is replaced by the vector $(t, \mathbf{x})$ and the system is replaced by $t' = 1$, $\mathbf{x}' = \mathbf{f}(t, \mathbf{x})$. An equilibrium point of an autonomous ODE, $\mathbf{x}_0$, is any vector that satisfies,

$$\mathbf{f}(\mathbf{x}_0) = \mathbf{0}.$$

It is difficult to make general statements about the behavior of a system of nonlinear ODEs. However, often one is interested in the behavior of the system perturbed around steady-state (an equilibrium point, since $\mathbf{x}' = \mathbf{0}$). The Hartman-Grobman Theorem can be used to make statements about perturbations of an autonomous nonlinear ODE around an equilibrium point:

**Theorem 2.2.3.** *[110, 111] In the differential equation:*

$$\boldsymbol{\xi}' = \mathbf{E}\boldsymbol{\xi} + \mathbf{F}(\boldsymbol{\xi}) \, , \tag{2.40}$$

*suppose that no eigenvalue of* $\mathbf{E}$ *has a vanishing real part and that* $\mathbf{F}(\boldsymbol{\xi})$ *is of class* $C^1$ *for small* $||\boldsymbol{\xi}||$*,* $\mathbf{F}(\mathbf{0}) = \mathbf{0}$*, and* $\partial_{\boldsymbol{\xi}}\mathbf{F}(\mathbf{0}) = \mathbf{0}$*. Consider the linear system:*

$$\boldsymbol{\zeta}' = \mathbf{E}\boldsymbol{\zeta}. \tag{2.41}$$

*Let* $T^t : \boldsymbol{\xi}_t = \boldsymbol{\eta}(t, \boldsymbol{\xi}_0)$ *and* $L^t : \boldsymbol{\zeta}_t = e^{\mathbf{E}t}\boldsymbol{\zeta}_0$ *be the general solution of Equations (2.40) and (2.41), respectively. Then there exists a continuous one-to-one map of a neighborhood of* $\boldsymbol{\xi} = \mathbf{0}$ *onto a neighborhood of* $\boldsymbol{\zeta} = \mathbf{0}$ *such that* $RT^tR^{-1} = L^t$*; in particular,* $R : \boldsymbol{\xi} \to \boldsymbol{\zeta}$ *maps solutions of Equation (2.40) near* $\boldsymbol{\xi} = \mathbf{0}$ *onto solutions of Equation (2.37) preserving parameterizations.*

*Proof.* The reader is referred to [110, 111] for full details of the proof. □

Basically, the result in Theorem 2.2.3 means that local behavior of an autonomous nonlinear ODE around an equilibrium point can be determined by studying the properties of a linearization of the ODE at the equilibrium point if there is no purely oscillatory component in the solution of the linearized ODE.

## 2.3 State-Space Approximation of DAE Models

The Theorems presented in § 2.2 allow one to characterize the behavior of an explicit ODE. However, we are advocating formulating cell-signaling models as DAEs. Naturally, the question arises whether there is an equivalent stability Theorem for systems of DAEs. We will restrict ourselves to studying a system of DAEs defined by Equation (2.1), and assume that the Jacobian matrix, $[\mathbf{f}_{\mathbf{x}'}\ \mathbf{f}_{\mathbf{y}}]$ is non-singular. This condition is satisfied for almost all cell-signaling networks and is sufficient to ensure that the DAE is index 1. Linearization of Equation (2.1) around an equilibrium solution yields the linear time invariant DAE:

$$\begin{aligned} \mathbf{0} \;=\;& \mathbf{f}_{\mathbf{x}'}(\mathbf{x}_0', \mathbf{x}_0, \mathbf{y}_0, \mathbf{u}_0)\,\Delta\mathbf{x}' + \mathbf{f}_{\mathbf{y}}(\mathbf{x}_0', \mathbf{x}_0, \mathbf{y}_0, \mathbf{u}_0)\,\Delta\mathbf{y} \\ &+ \mathbf{f}_{\mathbf{x}}(\mathbf{x}_0', \mathbf{x}_0, \mathbf{y}_0, \mathbf{u}_0)\,\Delta\mathbf{x} + \mathbf{f}_{\mathbf{u}}(\mathbf{x}_0', \mathbf{x}_0, \mathbf{y}_0, \mathbf{u}_0)\,\Delta\mathbf{u} \end{aligned} \tag{2.42}$$

where the variables $(\Delta\mathbf{x}, \Delta\mathbf{y}, \Delta\mathbf{u})$ represent perturbations from the equilibrium solution,

$$\begin{aligned} \mathbf{x} &= \mathbf{x}_0 + \Delta\mathbf{x} \\ \mathbf{y} &= \mathbf{y}_0 + \Delta\mathbf{y} \\ \mathbf{u} &= \mathbf{u}_0 + \Delta\mathbf{u}, \end{aligned}$$

and all of the Jacobian matrices are evaluated at the equilibrium solution. If the matrix $[\mathbf{f}_{\mathbf{x}'}\ \mathbf{f}_{\mathbf{y}}]$ is non-singular, then the linearization is index 1 (at least in a neighborhood of the equilibrium solution). For cell-signaling models, $[\mathbf{f}_{\mathbf{x}'}\ \mathbf{f}_{\mathbf{y}}]$ and $[\mathbf{f}_{\mathbf{x}}\ \mathbf{f}_{\mathbf{u}}]$ are unsymmetric, large, and sparse. The linearized DAE in Equation (2.42) can be rearranged to explicit ODE state-space form:

$$\begin{bmatrix} \Delta \mathbf{x}' \\ \Delta \mathbf{y} \end{bmatrix} = -\begin{bmatrix} \mathbf{f}_{\mathbf{x}'} & \mathbf{f}_{\mathbf{y}} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{f}_{\mathbf{x}} & \mathbf{f}_{\mathbf{u}} \end{bmatrix} \begin{bmatrix} \Delta \mathbf{x} \\ \Delta \mathbf{u} \end{bmatrix}. \tag{2.43}$$

It is natural to ask how the stability of the explicit state-space ODE (Equation (2.43)) is related to the stability of the original DAE. To characterize the stability of such a DAE, it is necessary to use the Implicit Function Theorem:

**Theorem 2.3.1.** *[188] Let* $\mathbf{f}$ *be a* $C^1$ *mapping of an open set* $E \subset \mathbb{R}^{n+m}$ *into* $R^n$*, such that* $\mathbf{f}(\mathbf{a}, \mathbf{b}) = \mathbf{0}$ *for some point* $(\mathbf{a}, \mathbf{b}) \in E$*. Put* $A = \mathbf{f}'(\mathbf{a}, \mathbf{b})$ *and assume that* $A_x$ *is invertible.*

*Then there exists open sets* $U \subset \mathbb{R}^{n+m}$ *and* $W \subset \mathbb{R}^m$*, with* $(\mathbf{a}, \mathbf{b}) \in U$ *and* $\mathbf{b} \in W$*, having the following property:*

*To every* $\mathbf{y} \in W$ *corresponds a unique* $\mathbf{x}$ *such that*

$$(\mathbf{x}, \mathbf{y}) \in U,$$

*and,*

$$\mathbf{f}(\mathbf{x}, \mathbf{y}) = \mathbf{0}.$$

*If this* $\mathbf{x}$ *is defined to be* $\mathbf{g}(\mathbf{y})$*, then* $\mathbf{g}$ *is a* $C^1$ *mapping of* $W$ *into* $\mathbb{R}^n$*,* $\mathbf{g}(\mathbf{b}) = \mathbf{a}$*,*

$$\mathbf{f}(\mathbf{g}(\mathbf{y}), \mathbf{y}) = \mathbf{0}, \quad \mathbf{y} \in W,$$

*and*

$$\mathbf{g}'(\mathbf{b}) = -(A_x)^{-1} A_y.$$

*Proof.* See Pages 225–227 of [188]. □

The stability of an autonomous DAE can now be described by the following Theorem:

**Theorem 2.3.2.** *Consider the autonomous DAE defined by Equation (2.44), and let*

$\mathbf{F}$ *be a* $C^1$ *mapping of an open set* $E \subset \mathbb{R}^{2n_x+n_y}$ *into* $\mathbb{R}^{n_x+n_y}$.

$$\mathbf{F}(\mathbf{x}', \mathbf{x}, \mathbf{y}) = \mathbf{0} \tag{2.44}$$

*Suppose Equation (2.45) is satisfied for some point* $(\mathbf{0}, \mathbf{x}_0, \mathbf{y}_0) \in E$*:*

$$\mathbf{F}(\mathbf{0}, \mathbf{x}, \mathbf{y}) = \mathbf{0}. \tag{2.45}$$

*If* $[\mathbf{F}_{\mathbf{x}'} \ \mathbf{F}_{\mathbf{y}}]_{(\mathbf{x}_0,\mathbf{y}_0)}$ *is non-singular and the time-invariant linearization around* $(\mathbf{x}_0, \mathbf{y}_0)$ *is asymptotically stable, then the original non-linear DAE in Equation (2.44) is asymptotically stable to sufficiently small perturbations around* $(\mathbf{x}_0, \mathbf{y}_0)$.

*Proof.* Define the variables $\Delta\mathbf{x}$ and $\Delta\mathbf{y}$ to be perturbations around the equilibrium point:

$$\begin{aligned} \mathbf{x} &= \mathbf{x}_0 + \Delta\mathbf{x} \\ \mathbf{y} &= \mathbf{y}_0 + \Delta\mathbf{y}. \end{aligned}$$

Then the following DAE can be written:

$$\mathbf{F}(\Delta\mathbf{x}', \mathbf{x}_0 + \Delta\mathbf{x}, \mathbf{y}_0 + \Delta\mathbf{y}) = \mathbf{0}, \tag{2.46}$$

where $\mathbf{F}$ is still a $C^1$ mapping and the Jacobian matrix:

$$\left[ \begin{array}{cc} \mathbf{F}_{\Delta\mathbf{x}'} & \mathbf{F}_{\Delta\mathbf{y}} \end{array} \right]_{(\mathbf{0},\mathbf{0})} = \left[ \begin{array}{cc} \mathbf{F}_{\mathbf{x}'} & \mathbf{F}_{\mathbf{y}} \end{array} \right]_{(\mathbf{x}_0,\mathbf{y}_0)}$$

is non-singular. The conditions of the Implicit Function Theorem (Theorem 2.3.1) apply to the autonomous index 1 DAE defined by Equation (2.46). Hence, it follows that locally the explicit ODE system is equivalent:

$$\Delta\mathbf{x}' = \mathbf{g}_1(\Delta\mathbf{x}) \tag{2.47}$$

$$\Delta\mathbf{y} = \mathbf{g}_2(\Delta\mathbf{x}), \tag{2.48}$$

where $\mathbf{g}_1$ and $\mathbf{g}_2$ are $C^1$ mappings. The conditions of Theorem 2.2.3 apply to Equation (2.47). In particular, asymptotic stability of the linearization of Equation (2.47) implies asymptotic stability of the explicit nonlinear ODE. Furthermore, since $\mathbf{g}_2$ is a $C^1$ mapping and $\mathbf{g}_2(\mathbf{0}) = \mathbf{0}$ by definition, it follows that $|\Delta\mathbf{y}| \to 0$ as $|\Delta\mathbf{x}| \to 0$. Hence, the original autonomous DAE given Equation (2.44) is asymptotically stable. □

It is demonstrated in Example 2.3.1 that for a non-autonomous system of ODEs, there can be a qualitative difference in the solution behavior between the time-invariant linearization and the original equations, even around an equilibrium point.

**Example 2.3.1.** Equation (2.49) has a single equilibrium point $x_0(t) = 0$.

$$x'(t) = -\frac{1}{1+t^2}x(t) \tag{2.49}$$

Consider the time-invariant linearization of the Equation (2.49) around the equilibrium point $x_0(t) = 0$. The time-invariant linearization of the equation at $t = t_0$ is given by:

$$\delta x' = -\frac{1}{1+t_0^2}\delta x.$$

Hence from the time-invariant linearization, it would be concluded that the system is asymptotically stable. The solution to the original system is given by:

$$x(t) = x(0)\exp(-\arctan(t)).$$

It is clear that the original equation is Lyapunov stable but not asymptotically stable.

From this simple example, it can be seen that quite restrictive conditions are required to guarantee qualitatively similar behavior between the linearization and the original system DAE. Linearizations around non-equilibrium solutions, and comparison of the solution behavior of the linearization with the original system, are considerably more complex, as discussed in [35, 149, 182].

### 2.3.1 Identity Elimination

It is worthwhile examining whether the system of DAEs representing a model can be simplified before calculating the state-space approximation to the original DAE. It is common when formulating a DAE model in a process modeling environment to have many assignments of the form shown in Equations (2.50)–(2.51),

$$y_i = y_j \tag{2.50}$$

$$y_i = x_j, \tag{2.51}$$

which can be safely eliminated. For example, the DAE system:

$$\begin{aligned} x_1' &= -y_1 \\ x_2' &= y_2 \\ y_1 &= x_1 \\ y_1 &= y_2 \end{aligned}$$

can be rewritten as the ODE:

$$\begin{aligned} x_1' &= -x_1 \\ x_2' &= x_1 \end{aligned}$$

by eliminating identities.

These equations result from connecting together smaller sub-models. For example, the flux of intact receptors leaving the endosome might equal the flux of intact receptors arriving at the cell surface. Such identity relationships can be automatically eliminated from a DAE model without changing the dynamics of the system. The advantages of eliminating identity equations symbolically are:

1. the resulting system of equations is smaller,
2. certain types of numerically ill-posed problems can be identified,

3. and, the elimination is exact.

It should be noted that any assignment including an element of $\mathbf{x}'$ is a differential equation and should not be eliminated. Assignments of the form $x_i = x_j$ imply a high index system, and the elimination algorithm should halt. Assignments of the form $u_i = u_j$ imply a inconsistent set of equations.

If the model is formulated in a process modeling environment (such as ABACUSS II [41, 230]), it is possible to identify these identity equations and eliminate them symbolically [17]. A modified version of DAEPACK [228] has the ability to identify simple identity equations and mark them for symbolic elimination. The algorithm proceeds by defining an identity group, which contains all variables that are equivalent. The root node of the identity group can be a differential variable, algebraic variable, or input. All subsequent variables added to the identity group must be algebraic, otherwise the problem is ill-posed and the algorithm generates an error message. The Jacobian matrices $\mathbf{W}$, $\mathbf{V}$ are compressed to reflect the eliminated equations and variables.

## 2.3.2 Construction of State-Space Approximation

It has now been established (Theorem 2.3.2) that under certain conditions the solution behavior of the state-space approximation to an autonomous DAE is equivalent to the solution behavior of the original DAE. The remaining question is whether the state-space approximation of a cell-signaling model can be efficiently and automatically constructed from the original system of DAEs.

We propose two alternative methods for calculating the state-space form of a linearized DAE and compare these methods to a modification of an existing algorithm. It is necessary to exploit model sparsity in order to construct the state-space approximation efficiently. A DAE model is sparse if each equation in the model depends only on a few variables (typically 3-5 variables). It should be stressed that model sparsity does not imply the states are uncoupled. For example, the system of nonlinear equations:

$$\begin{aligned}
f_1(x_1, x_2) &= 0 \\
f_2(x_2, x_3) &= 0 \\
\vdots &= \vdots \\
f_n(x_{n-1}, x_n) &= 0
\end{aligned}$$

is very sparse but does not block-decompose, i.e., all of the states are coupled. Most cell signaling models are sparse; each species conservation equation typically depends on a few fluxes and a small number of generation terms. In general, cell signaling models tend to be strongly connected; the mole number of each species influences the mole number of all the other species. More formally, the sparsity pattern of a matrix is defined by Definition 2.3.1.

**Definition 2.3.1.** The sparsity pattern of the matrix $\mathbf{A}$ is the set of row and column indices that correspond to a non-zero element of $\mathbf{A}$; i.e.,

$$\{\mathbf{i}, \mathbf{j} : a_{i_k, j_k} \neq 0\}.$$

An overestimate of the sparsity pattern of $\mathbf{A}$ is a set of row and column indices of $\mathbf{A}$ that contain the set of indices corresponding to the sparsity pattern of $\mathbf{A}$; i.e.,

$$\left\{\hat{\mathbf{i}}, \hat{\mathbf{j}}\right\} \supset \{\mathbf{i}, \mathbf{j}\}.$$

To construct the state-space approximation from the linearized DAE, the vector $\mathbf{y}(t)$ is partitioned into $(\mathbf{y}_1(t), \mathbf{y}_2(t))$, where $\mathbf{y}_1(t) \in \mathbb{R}^{n_{y1}}$ contains the algebraic variables desired in the state-space model (i.e., outputs that it is necessary to track), and $\mathbf{y}_2(t) \in \mathbb{R}^{n_{y2}}$ contains the variables to be eliminated (e.g., intermediate variables such as rates of reaction, fluxes, and variables due to semi-discretization of PDAEs).

The matrices, $\mathbf{W}$, $\mathbf{S}$, and $\mathbf{V}$, are defined according to Equations (2.52)–(2.54), where $\mathbf{S}$ is the state-space model to be calculated.

$$\mathbf{W} = \begin{bmatrix} \mathbf{f}_{\mathbf{x}'} & \mathbf{f}_{\mathbf{y}} \end{bmatrix} \in \mathbb{R}^{(n_x+n_y)\times(n_x+n_y)} \tag{2.52}$$

$$\mathbf{S} = \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix} \in \mathbb{R}^{\left(n_x+n_{y_1}\right)\times(n_x+n_u)} \tag{2.53}$$

$$\mathbf{V} = -\begin{bmatrix} \mathbf{f}_{\mathbf{x}} & \mathbf{f}_{\mathbf{u}} \end{bmatrix} \in \mathbb{R}^{(n_x+n_y)\times(n_x+n_u)} \tag{2.54}$$

A naïve method to calculate the state-space model would be to solve the matrix equation:

$$\mathbf{W}\mathbf{Z}_3 = \mathbf{V}, \tag{2.55}$$

using dense linear algebra and obtain $\mathbf{S}$ by eliminating unwanted rows of $\mathbf{Z}_3$. The difficulties with this approach are:

1. it is necessary to calculate rows of $\mathbf{S}$ that correspond to unwanted algebraic variables, and,

2. sparsity is not exploited in the calculation.

For a large model, the computational cost of naïvely computing $\mathbf{S}$ would be prohibitive. An even worse approach would be to calculate $\mathbf{S}$ from

$$\mathbf{S} = \mathbf{P}_1\mathbf{W}^{-1}\mathbf{V}, \tag{2.56}$$

where $\mathbf{P}_1$ is used to eliminate unwanted rows, since typically the inverse of $\mathbf{W}$ is dense even if $\mathbf{W}$ is sparse.

In the proposed approached, the non-zero structure of the state-space model is calculated *a priori*. Sparsity of the state-space model can thenbe exploited in further calculations and reduces storage requirements. The generation of the structure of the state-space model also guarantees that entries that are known to be zero are not corrupted with numerical error. The graph tracing algorithm that determines the structure of the state-space model is described in § 2.3.3. The idea of exploiting structure of $\mathbf{S}$ when solving Equation (2.55) for sparse $\mathbf{W}$ and $\mathbf{V}$ is not new [96, 95].

What is different in our approach compared to [96, 95] is that

1. direct memory addressing for the forward and back substitions is used, and,
2. in some of the proposed approaches, unwanted rows of the state-space matrix are not calculated.

While the differences are subtle the impact can be profound, leading to a far faster implementation.

## 2.3.3 Generation of State-Space Occurrence Information

It is necessary to determine the sparsity pattern of $\mathbf{S}$ from the sparsity pattern of the Jacobian matrices $\mathbf{W}$ and $\mathbf{V}$ in order to compute the state-space model efficiently. The system of DAEs is represented by an acyclic digraph which is a slight variant of the digraphs used by [183, 84]. The digraph of the DAE is determined as follows:

1. A node is generated for each strongly connected component in $\mathbf{W}$ and the matrix is permuted to block upper triangular form $\mathbf{PWQ}$.
2. The occurrence information of the rectangular system:

   $$\begin{bmatrix}\mathbf{PWQ} & \mathbf{PV}\end{bmatrix}$$

   is constructed.
3. An arc is generated from each input or state to a strongly connected component of $\mathbf{PWQ}$ if there is an entry in the column of $\mathbf{PV}$ that is in one of the rows assigned to a strongly connected component in $\mathbf{PWQ}$.
4. An arc is drawn between a strongly connected component of $\mathbf{PWQ}$ and another strongly connected component of $\mathbf{PWQ}$ if there is an entry in a column assigned to the first strongly connected component that corresponds to one of the rows assigned to the second strongly connected component.

```
/* Initialize */
1 white ← 0
2 grey ← 1
3 for each node $v \in V[G]$
4    do color $(v)$ ← white

     /* Depth-first search */
5 for each $\phi \in (\Delta\mathbf{x}, \Delta\mathbf{u})$
6    for each node $v \in Adj[\phi]$
7       do if color $(v)$ < grey
8       DFS-VISIT $(v, \phi)$
9    grey ← grey + 1

/* dfs-visit(w,u) */
  DFS-VISIT $(w, u)$
1 color $(w)$ ← grey

     /* Write occurrence information */
2 for each variable $\omega = (\Delta\mathbf{x}', \Delta\mathbf{y}) \in w$
3    write $(\omega, \phi)$

4 for each node $v \in Adj[w]$
5    do if color $(v)$ < grey
6       DFS-VISIT $(v, \phi)$
```

Figure 2-4: Generation of state-space model occurrence information

The strongly connected components of $\mathbf{W}$ are found by performing row permutations on $\mathbf{W}$ to find a maximal traversal [68]. A check is made to see whether the Jacobian $\mathbf{W}$ matrix is structurally singular, in which case the current algorithm cannot be applied. An equation assignment is made for each of the time derivatives and algebraic variables $(\Delta\mathbf{x}', \Delta\mathbf{y})$. The strongly connected components of $\mathbf{W}$ are identified from the equation assignment using an algorithm proposed by [223] and implemented by [71]. The digraph of the DAE is constructed using the depth-first search algorithm of Figure 2-4. The digraph and occurrence information corresponding to a DAE is demonstrated in Example 2.3.2.

**Example 2.3.2.** Consider the DAE defined by the system shown in Equations (2.57)–

(2.61).

$$\dot{x}_1 = -y_1 \tag{2.57}$$
$$\dot{x}_2 = y_1 - y_2 \tag{2.58}$$
$$\dot{x}_3 = y_2 \tag{2.59}$$
$$y_1 = k_1 x_1 \tag{2.60}$$
$$y_2 = k_2 x_2 \tag{2.61}$$

The occurence information for **W** permuted to block upper triangular form is:

$$\begin{array}{cccccc}
 & \dot{x}_1 & \dot{x}_2 & \dot{x}_3 & y_1 & y_2 \\
\dot{x}_1 & \otimes & 0 & 0 & \times & 0 \\
\dot{x}_2 & 0 & \otimes & 0 & \times & \times \\
\dot{x}_3 & 0 & 0 & \otimes & 0 & \times \\
y_1 & 0 & 0 & 0 & \otimes & 0 \\
y_2 & 0 & 0 & 0 & 0 & \otimes
\end{array}.$$

The occurrence information corresponding to the DAE shown in Equations (2.57)–(2.61) is:

$$\begin{bmatrix}\mathbf{PWQ} & \mathbf{PV}\end{bmatrix} = \begin{array}{ccccccccc}
 & \dot{x}_1 & \dot{x}_2 & \dot{x}_3 & y_1 & y_2 & x_1 & x_2 & x_3 \\
\dot{x}_1 & \otimes & 0 & 0 & \times & 0 & 0 & 0 & 0 \\
\dot{x}_2 & 0 & \otimes & 0 & \times & \times & 0 & 0 & 0 \\
\dot{x}_3 & 0 & 0 & \otimes & 0 & \times & 0 & 0 & 0 \\
y_1 & 0 & 0 & 0 & \otimes & 0 & \times & 0 & 0 \\
y_2 & 0 & 0 & 0 & 0 & \otimes & 0 & \times & 0
\end{array}.$$

The digraph of Equations (2.57)–(2.61) is shown in Figure 2-5.

The occurrence information for the state-space model is generated by tracing the digraph of the DAE from states (differential variables) and inputs $(\Delta\mathbf{x}, \Delta\mathbf{u})$ to time derivatives and outputs $(\Delta\mathbf{x}', \Delta y)$. From each column in **V**, the strongly connected components in **W** that depend on that input are traced. The nodes $(\Delta\mathbf{x}', \Delta\mathbf{y})$ that are

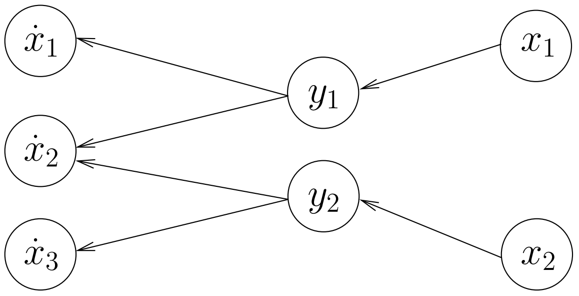


Figure 2-5: Graph of a system of DAEs

connected to the starting column $(\Delta\mathbf{x}, \Delta\mathbf{u})$ correspond to entries in the state-space matrix. To avoid reinitializing the colors of each node for every iteration in the depth-first search, a numerical value is assigned to grey and black, which is incremented on every pass of the depth-first search. The state-space occurrence information can be recorded as the digraph for the DAE is constructed (see algorithm in Figure 2-4). The proposed algorithm is a generalization of the algorithm proposed by [95]. The key difference in our approach is the recoloring of the nodes to avoid a reinitialization step. The algorithm uses a depth-first search algorithm described by [48].

**Theorem 2.3.3.** *The algorithm generates a sparsity pattern that is equal or overestimates the sparsity pattern of the state-space model $S$.*

*Proof.* The proof requires the application of Theorem 5.1 of [95] to each column of $\mathbf{V}$. □

It should be noted that if instead of augmenting the occurrence information of $\mathbf{W}$ with the occurrence information of $\mathbf{V}$, the occurrence information of $\mathbf{W}$ is augmented with the identity, $\mathbf{I}$, the structure of $\mathbf{W}^{-1}$ is obtained from this algorithm. If $\mathbf{W}$ is irreducible, the structure of $\mathbf{W}^{-1}$ will be dense [69]. Furthermore, every column of $\mathbf{V}$ with a non-zero entry, will correspond to a full column of the state-space matrix $\mathbf{S}$. However, for many problems of practical interest, $\mathbf{W}$ is reducible. Fortran code to compute the structure of $\mathbf{S}$ is shown in Appendix C, § C.1.

**Example 2.3.3.** The occurrence information for the explicit state-space form of the model in Equations (2.57)–(2.61) of Example 2.3.2 is

$$\begin{array}{c|ccc} & x_1 & x_2 & x_3 \\ \dot{x}_1 & \times & 0 & 0 \\ \dot{x}_2 & \times & \times & 0 \\ \dot{x}_3 & 0 & \times & 0 \\ y_1 & \times & 0 & 0 \\ y_2 & 0 & \times & 0 \end{array} .$$

**Complexity of Determining Occurrence Information**

The depth-first search to construct the occurrence information of the state-space model has a worst case running time of $O(\tau m)$ where $\tau$ is the number of non-zero entries in $\mathbf{W}$ and $m$ is the number of states and inputs, $(\Delta\mathbf{x}, \Delta\mathbf{u})$. This will happen when $\mathbf{W}$ block decomposes to a triangular matrix, a complete set of states and inputs are connected to every block and every block $j$ is connected to all $j-i$ blocks. However, for most reasonable applications, it is anticipated that the running time is considerably less that the worst case complexity.

### 2.3.4 Algorithms to Generate State-Space Model

Once the sparsity pattern of the state-space model has been computed, it is necessary to compute numerical values for the entries in $\mathbf{S}$. We propose three different methods, summarized by:

1. Calculate columns of $\mathbf{W}^{-1}$ and then matrix-matrix product.

$$\mathbf{W}\mathbf{Z}_1 = \mathbf{I} \tag{2.62}$$
$$\mathbf{S} = \mathbf{P}_1\mathbf{Z}_1\mathbf{V} \tag{2.63}$$

2. Calculate rows of $\mathbf{W}^{-1}$ and then matrix-matrix product.

$$\mathbf{W}^T\mathbf{Z}_2 = \mathbf{P}_2 \tag{2.64}$$

$$\mathbf{S} = \mathbf{Z}_2^T \mathbf{V} \tag{2.65}$$

3. Calculate columns of $\mathbf{S}$ directly.

$$\begin{aligned} \mathbf{W}\mathbf{Z}_3 &= \mathbf{V} && (2.66)\\ \mathbf{S} &= \mathbf{P}_1 \mathbf{Z}_3 && (2.67) \end{aligned}$$

The matrices $\mathbf{P}_1 \in \mathbb{R}^{n_x+n_{y_1} \times n_x+n_y}$ and $\mathbf{P}_2 \in \mathbb{R}^{n_x+n_y \times n_x+n_{y_1}}$ are defined by

$$p_{ij} = \begin{cases} 1 & \text{if } j = \min\psi : \quad z_\psi \in (x, y_1)^T \text{ and } p_{wj} = 0 \;\forall\; w < i, \\ 0 & \text{otherwise.} \end{cases}$$

and,

$$p_{ij} = \begin{cases} 1 & \text{if } i = \min\psi : \quad z_\psi \in (x, y_1)^T \text{ and } p_{iw} = 0 \;\forall\; w < j, \\ 0 & \text{otherwise.} \end{cases},$$

respectively. The matrices $\mathbf{P}_1$ and $\mathbf{P}_2$ are used to eliminated the unwanted algebraic variables $\mathbf{y}_2$. It is straightforward to show by simple rearrangement that Methods 1–3 are mathematically equivalent.

Method 1 is a generalization of the existing method proposed by [240] and requires the computation of a matrix inverse, $\mathbf{Z}_1$. Method (2.64) still requires the calculation of a matrix inverse, $\mathbf{Z}_2$. However, it has the computational advantage that only columns of $\mathbf{Z}_2$ corresponding to the variables $(\mathbf{x}, \mathbf{y}_1)$ to be included in the state-space model need be calculated. The final method does not require the explicit calculation of a matrix inverse, but requires the unwanted portion of the state-space model to be eliminated after it has been calculated. The general form of the proposed algorithms is shown in Figure 2-6.

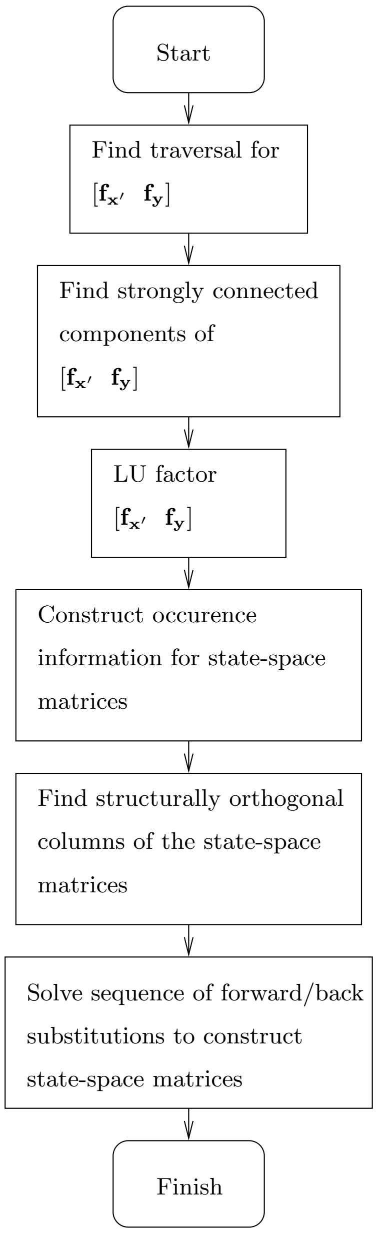

Start
Find traversal for $[\mathbf{f}_{\mathbf{x}'} \quad \mathbf{f}_{\mathbf{y}}]$
Find strongly connected components of $[\mathbf{f}_{\mathbf{x}'} \quad \mathbf{f}_{\mathbf{y}}]$
LU factor $[\mathbf{f}_{\mathbf{x}'} \quad \mathbf{f}_{\mathbf{y}}]$
Construct occurence information for state-space matrices
Find structurally orthogonal columns of the state-space matrices
Solve sequence of forward/back substitutions to construct state-space matrices
Finish

Figure 2-6: Summary of algorithm to calculate state-space model

## Algorithm I

In an existing method proposed by [240], it is assumed that the original DAE can be written as shown in Equation (2.68).

$$\begin{aligned} \mathbf{f}(\mathbf{x}', \mathbf{x}, \mathbf{y}, \mathbf{u}) &= \mathbf{0} \\ \mathbf{g}(\mathbf{x}, \mathbf{y}, \mathbf{u}) &= \mathbf{0} \end{aligned} \tag{2.68}$$

$\mathbf{f}$ is a set of $n$ differential equations and $\mathbf{g}$ is a set of $m$ algebraic equations. This system can be linearized and rearranged to give the state-space matrix, $\mathbf{S}$ as shown in Equation (2.69).

$$\mathbf{S} = -\begin{bmatrix} \mathbf{f}_{\mathbf{x}'}^{-1}\left(\mathbf{f}_{\mathbf{x}} - \mathbf{f}_{\mathbf{y}}\mathbf{g}_{\mathbf{y}}^{-1}\mathbf{g}_{\mathbf{x}}\right) & \mathbf{f}_{\mathbf{x}'}^{-1}\left(\mathbf{f}_{\mathbf{u}} - \mathbf{f}_{\mathbf{y}}\mathbf{g}_{\mathbf{y}}^{-1}\mathbf{g}_{\mathbf{u}}\right) \\ \mathbf{g}_{\mathbf{y}}^{-1}\mathbf{g}_{\mathbf{x}} & \mathbf{g}_{\mathbf{y}}^{-1}\mathbf{g}_{\mathbf{u}} \end{bmatrix} \tag{2.69}$$

It can clearly be seen that the number of differential equations must equal the number of differential variables, and that the Jacobians, $\mathbf{f}_{\mathbf{x}'}$ and $\mathbf{g}_{\mathbf{y}}$, must be invertible. The state-space form is computed directly by inversion of $\mathbf{f}_{\mathbf{x}'}$ and $\mathbf{g}_{\mathbf{y}}$. It is not clear from the description of this method whether the structure of the state-space matrix is determined.

A generalization of the method by Wong [240] can be derived for the DAEs given in Equation (2.1). The inverse of $\mathbf{W}$ is explicitly calculated by solving Equation (2.62) a column at a time. If the method is implemented naïvely, the inverse of $\mathbf{W}$ is stored in dense format. Instead, sparsity of $\mathbf{Z}_1$ and $\mathbf{V}$ should be exploited by calculating the matrix-matrix product $\mathbf{Z}_1\mathbf{V}$ according to an algorithm by [104], shown in Equation (2.70).

$$\mathbf{S} = \sum_k (\mathbf{Z}_1)_{:k}\,\mathbf{V}_{k:} \tag{2.70}$$

The state-space matrix is accumulated through intermediate calculations, and only a single column of $\mathbf{Z}_1$ is stored at any point in time. Hence, the intermediate storage required is a single vector of length $n_x + n_y$. All of the columns of $\mathbf{Z}_1$ must be calculated. Some of the entries in a full column of $\mathbf{Z}_1$ correspond to algebraic variables,

$\mathbf{y}_2$, that will be eliminated from the state-space model.

### Complexity of Algorithm I

The cost of Algorithm I is summarized below.

1. Cost of calculating $LU$ factors, $C_{LU}$.

2. Cost of $n_x + n_y$ back and forward substitutions, where the combined cost of a back and forward substitution, $C_{SOLVE}$, is $2\tau_{LU} - 3\left(n_x + n_y\right)$ where $\tau_{LU}$ is the number of entries in the factors $L$ and $U$.

3. Cost of a matrix-matrix product $C_{PRODUCT}$.

The total cost for the algorithm is is given by Equation (2.71), where $n = n_x + n_y$, $m = n_x + n_u$ and $p = n_x + n_{y_1}$. For the dense case this evaluates to the expression given by Equation (2.72).

$$\begin{aligned} C_{TOTAL} &= C_{LU} + \left(n_x + n_y\right) C_{SOLVE} + C_{PRODUCT} && (2.71)\\ &= \left(\frac{2}{3}n^3 - \frac{1}{2}n^2 - \frac{1}{6}n\right) + n\left(2n^2 - n\right) + 2nmp && (2.72)\end{aligned}$$

For the sparse case, the cost terms are problem specific.

### Algorithm II

The necessity of storing all of the inverse of $\mathbf{W}$ can also be avoided by calculating the inverse a row at a time. This forms the basis of the second algorithm, shown in Equations (2.64)–(2.65). Once a column of $\mathbf{Z}_2$ has been determined, a row of the state-space matrix can be determined directly, by forming the vector-matrix product, $\mathbf{s}_j = \mathbf{z}_i^T V$, where $\mathbf{s}_j \in \mathbb{R}^{n_x+n_u}$ is a row of the state-space matrix, and $\mathbf{z}_i \in \mathbb{R}^{n_x+n_y}$ is a column of $\mathbf{Z}_2$. The intermediate storage required is a single vector of length $n_x + n_y$. The algorithm is computationally more efficient, since unwanted rows of the inverse of $\mathbf{W}$ are not calculated. It can be concluded that there are no computational advantages in implementing the first algorithm.

## Complexity of Algorithm II

The cost of Algorithm II is summarized below.

1. Cost of calculating $LU$ factors, $C_{LU}$.

2. Cost of $n_x + n_{y_1}$ back and forward substitutions.

3. Cost of a matrix-matrix product $C_{PRODUCT}$.

The total cost for the algorithm is is given by Equation (2.73), where $n = n_x + n_y$, $m = n_x + n_u$ and $p = n_x + n_{y_1}$. For the dense case this evaluates to the expression given by Equation (2.74).

$$\begin{aligned} C_{TOTAL} &= C_{LU} + \left(n_x + n_{y_1}\right) C_{SOLVE} + C_{PRODUCT} & (2.73) \\ &= \left(\frac{2}{3}n^3 - \frac{1}{2}n^2 - \frac{1}{6}n\right) + p\left(2n^2 - n\right) + 2nmp & (2.74) \end{aligned}$$

## Algorithm III

No intermediate quantities are calculated in the third algorithm. The columns of the state-space matrix are calculated directly from Equations (2.66)–(2.67). This method has the advantage that no intermediate storage is required. Furthermore, the matrix-matrix products that are necessary in the first two methods are avoided. It should be noted that:

1. matrix-matrix products can be expensive to calculate, and

2. significant error can be introduced into the solution when the matrix-matrix products are calculated.

The third method has the disadvantage that the unwanted portion of the state-space model is calculated. However, the undesired entries can be discarded as each column of the state-space model is calculated, as shown in Equation (2.67). The additional storage workspace would be an additional $n_{y_2}$ entries.

**Complexity of Algorithm III**

The cost of Algorithm III is summarized below.

1. Cost of calculating $LU$ factors, $C_{LU}$.

2. Cost of $n_x + n_u$ back and forward substitutions.

The total cost for the algorithm is is given by Equation (2.75), where $n = n_x + n_y$ and $m = n_x + n_u$. There is no matrix-matrix product to calculate with this method. For the dense case this evaluates to the expression given by Equation (2.76).

$$\begin{aligned} C_{TOTAL} &= C_{LU} + (n_x + n_y)\, C_{SOLVE} \qquad (2.75) \\ &= \left(\frac{2}{3}n^3 - \frac{1}{2}n^2 - \frac{1}{6}n\right) + m\left(2n^2 - n\right) \qquad (2.76) \end{aligned}$$

**Comparison of Computational Cost**

The computational cost for each algorithm is summarized in Table 2.3. It is assumed that structurally orthogonal columns are not exploited in the calculation. $C_{LU}$ is the cost of $LU$ factoring $\mathbf{W}$ and is constant between all methods. $C_S$ is the cost of a back substitution. In general, the number of back substitutions required for each method will be different. However, in virtually all circumstances, $n_y >> n_u$ or $n_{y_1}$, which means that Algorithm I will be the most expensive. $C_P$ is the cost of an inner or outer product between $Z$ and $V$. It can be seen that this cost is not incurred by Algorithm III. However, if $n_{y_1} < n_u$, then Algorithm II may be the fastest.

## 2.3.5 Structurally Orthogonal Groups

The speed of all three algorithms depends on solving a linear equation of the form:

$$\mathbf{A}\mathbf{x} = \mathbf{b}. \qquad (2.77)$$

It is usually assumed that the vector $\mathbf{x}$ is dense when solving Equation (2.77) with a sparse $LU$ factorization code. This assumption allows one to use direct memory

Table 2.3: Comparison of computational costs

| Alg. | Step | Cost |
|---|---|---|
| 1 | LU Factor. | $C_{LU}$ |
| | Back subs. | $(n_x + n_y)\, C_S$ |
| | Matrix product | $(n_x + n_u)\left(n_x + n_{y_1}\right) C_P$ |
| 2 | LU Factor. | $C_{LU}$ |
| | Back subs. | $\left(n_x + n_{y_1}\right) C_S$ |
| | Matrix product | $(n_x + n_u)\left(n_x + n_{y_1}\right) C_P$ |
| 3 | LU Factor. | $C_{LU}$ |
| | Back subs. | $(n_x + n_u)\, C_S$ |

addressing when performing the forward and back substitutions. Typically, the improvement in speed of direct memory addressing compared to indirect addressing outweighs the increase in the number of operations. Clearly, if $\mathbf{x}$ is very sparse, a lot of additional work will be done. If Equation (2.77) must be solved repeatedly for many different right hand sides, it is possible to exploit the concept of structurally orthogonal columns to reduce the number of operations while still assuming the vector $\mathbf{x}$ is dense.

The concept of using structurally orthogonal columns to reduce the amount of work in evaluating a Jacobian matrix was developed by [53]. An efficient algorithm to partition a matrix into groups of structurally orthogonal columns has been developed by [44] and implemented in [43]. A pair of columns are structurally orthogonal if Definition 2.3.2 is satisfied.

**Definition 2.3.2.** A pair of columns, $(\mathbf{x}_i, \mathbf{x}_j)$, are structurally orthogonal if for all rows in $\mathbf{x}_i$ with a non-zero entry, there is not a non-zero entry in the same row of $\mathbf{x}_j$.

In all of the algorithms, a matrix equation of the form shown in Equation (2.78) is solved by repeatedly performing forward and back substitutions for each column of $\mathbf{B}$.

$$\mathbf{AX} = \mathbf{B} \tag{2.78}$$

If a set $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_k\}$ of columns of $\mathbf{X}$ are structurally orthogonal, then it is possible to solve the system shown in Equation (2.79).

$$\mathbf{A}\sum_{i=1}^{k}\mathbf{x}_i = \sum_{i=1}^{k}\mathbf{b}_i \tag{2.79}$$

The entries can then be recorded in the correct column of $\mathbf{x}_i$, since by definition each entry in $\sum_{i=1}^{k}\mathbf{x}_i$ corresponds to a unique column of $\mathbf{X}$.

The occurrence information of $\mathbf{X}$ can be generated by the Algorithm described in § 2.3.3. It is demonstrated in Example 2.3.4 that the proposed algorithm can overestimate of the number of forward and back substitutions necessary to calculate the state-space model.

**Example 2.3.4.** Consider the system:

$$\begin{aligned} x_1' &= x_1 + y_1 - y_2 \\ x_2' &= -x_1 + y_1 - y_2 \\ y_1 &= x_2 \\ y_2 &= x_2. \end{aligned}$$

The structural information for the DAE system is

$$\begin{bmatrix} \mathbf{f}_{\mathbf{x}'} & \mathbf{f}_{\mathbf{y}} | & \mathbf{f}_{\mathbf{x}} \end{bmatrix} = \left[\begin{array}{cccc|cc} \times & 0 & \times & \times & \times & 0 \\ 0 & \times & \times & \times & \times & 0 \\ 0 & 0 & \times & 0 & 0 & \times \\ 0 & 0 & 0 & \times & 0 & \times \end{array}\right].$$

The corresponding structure matrix for the state-space model is generated by the proposed depth-first search algorithm (§ 2.3.3):

$$\begin{bmatrix} A \\ C \end{bmatrix} = \begin{bmatrix} \times & \times \\ \times & \times \\ 0 & \times \\ 0 & \times \end{bmatrix}.$$

It can be seen that the columns of the state-space matrix are not structurally orthogonal. However, when the state-space matrices are calculated, as shown in Equation (2.80), it can be seen that numerical cancellation leads to structurally orthogonal columns.

$$\begin{bmatrix} A \\ C \end{bmatrix} = \begin{bmatrix} -1 & 0 \\ 1 & 0 \\ 0 & -1 \\ 0 & -1 \end{bmatrix} \tag{2.80}$$

Consequently, more forward and back substitutions would be performed than strictly required. However, the solution would be correct.

Theorem 2.3.4 can be used to bound the number of partitions of structurally orthogonal columns in $\mathbf{X}$.

**Theorem 2.3.4.** *If $\mathbf{B} \in \mathbb{R}^{n \times m}$, the inverse of $\mathbf{A} \in \mathbb{R}^{n \times n}$ exists, $m \geq n$, and $\mathbf{B}$ is structurally full row rank, then the number of partitions of structurally orthogonal columns in $\mathbf{X}$ is greater than or equal to the number of partitions of structurally orthogonal columns in $\mathbf{A}^{-1}$.*

*Proof.* Since $\mathbf{B}$ is full row rank, each column of $\mathbf{X}$, $\mathbf{x}_j$ is a linear combination of the columns of $\mathbf{Z} = \mathbf{A}^{-1}$, where at least one of the coefficients is not structurally zero as shown by Equation (2.81).

$$\mathbf{x}_j = \sum_{i=1}^{n} b_{ij} \mathbf{z}_i \tag{2.81}$$

If two columns of $\mathbf{B}$ have an entry in the same row, then by definition, two columns of $\mathbf{X}$ cannot be structurally orthogonal. If $m \geq n$ and $\mathbf{B}$ has $n$ structurally orthogonal columns, then every column of $\mathbf{Z}$ must appear as a linear combination in at least one of the columns of $\mathbf{X}$. Hence the number of orthogonal partitions must be equal to or greater than the number of orthogonal partitions of $\mathbf{A}^{-1}$. $\square$

It is tempting to try to generalize Theorem 2.3.4 to matrices, $\mathbf{B}$, of arbitrary structure and dimension. However, except for very specific structures of $\mathbf{B}$, there is no general relationship between the number of structurally orthogonal groups in $\mathbf{B}$

and $\mathbf{A}^{-1}$. If the number of columns of $\mathbf{B}$ is smaller than the number of columns of $\mathbf{A}$, then it may be possible to choose all of the columns of $\mathbf{X}$ to be orthogonal linear combinations of columns from a single orthogonal partition of $\mathbf{A}^{-1}$. This is shown in Example 2.82.

**Example 2.3.5.** Consider the matrix-matrix product $\mathbf{X} = \mathbf{A}^{-1}\mathbf{B}$ shown in Equation (2.82).

$$\begin{bmatrix} \times & \\ & \times \end{bmatrix} = \begin{bmatrix} \times & & & \\ & \times & & \\ & & \times & \times \\ & & \times & \times \end{bmatrix} \begin{bmatrix} \times & \\ & \times \\ & \\ & \end{bmatrix} \tag{2.82}$$

The number of partitions of structurally orthogonal columns in $\mathbf{A}^{-1}$ is two and the number for $\mathbf{X}$ is one.

In practical problems, the number of partitions of structurally orthogonal columns in the state-space model is smaller than the number of partitions in the inverse of $\mathbf{W}$, since usually $n_x + n_u \ll n_x + n_y$ and typically $\mathbf{V}$ contains a small number of entries per column. Hence, Algorithm I is likely to be more computationally expensive than Algorithm III even with exploitation of structurally orthogonal columns.

There is no clear relationship between the number of partitions of structurally orthogonal columns and the number of partitions of structurally orthogonal rows of a matrix. Hence, it is difficult to conclude for general problems whether the number of forward and back substitutions for Algorithm III is less than the number required for Algorithm II.

## 2.4 Error Analysis of State-Space Model

Bounds on the error in Algorithms I, II and III are derived in § 2.4.1–2.4.3, respectively. To compare the error in all three solution methods, it is necessary to bound the error made in solving a linear system of equations and the error introduced when calculating a vector-matrix product. The error introduced in calculating the solution to a set of sparse linear equations can be determined using the theory developed by [163, 208, 207, 11] and described in [219]. Error bounds are based on the notion of componentwise error. It is necessary to define the operators $|\cdot|$, and $\leqq$, as shown in Definition 2.4.1.

**Definition 2.4.1.** $|\mathbf{u}|$ is the vector of entries $|u_i|$. $|\mathbf{P}|$ is the matrix of entries $|p_{ij}|$. $\mathbf{u} \leq \mathbf{v}$ means $u_i \leqq v_i \ \forall \ i$. $\mathbf{Q} \leq \mathbf{P}$ means $q_{ij} \leqq p_{ij} \ \forall \ i, \ j$.

## 2.4.1 Algorithm I

A simplified version of Algorithm I is considered, where none of the rows of the state-space matrix are discarded. The algorithm is shown in Equations (2.83)–(2.84).

$$\mathbf{W}\mathbf{Z}_1 = \mathbf{I} \tag{2.83}$$

$$\mathbf{S} = \mathbf{Z}_1\mathbf{V} \tag{2.84}$$

The residual of Equation (2.83) is defined as $\mathbf{R}_1 = \mathbf{W}\mathbf{Z}_1 - \mathbf{I}$. The error in the solution of Equation (2.83) is given by Equation (2.85).

$$\delta\mathbf{Z}_1 = \mathbf{W}^{-1}\mathbf{R}_1 \tag{2.85}$$

The work of [208] shows, that with one step of iterative refinement and single precision residual accumulation, it is possible to bound the components of the computed value of the residual $|\mathbf{R}_1|$, as shown in Equation (2.86).

$$|\mathbf{R}_1| \leq \epsilon\,(n+1)\,|\mathbf{W}|\,|\mathbf{Z}_1| \tag{2.86}$$

Neglecting the round-off error in accumulating the necessary inner products, the component error in the state-space matrix is given by Equations (2.87)-(2.90).

$$\delta\mathbf{S} = \delta\mathbf{Z}_1\mathbf{V} \tag{2.87}$$

$$= \mathbf{W}^{-1}\mathbf{R}_1\mathbf{V} \tag{2.88}$$

$$\begin{aligned} |\delta\mathbf{S}| &\leq \left|\mathbf{W}^{-1}\mathbf{R}_1\mathbf{V}\right| && (2.89) \\ &\leq \epsilon\,(n+1)\left|\mathbf{W}^{-1}\right|\left|\mathbf{W}\right|\left|\mathbf{Z}_1\right|\left|\mathbf{V}\right| && (2.90) \end{aligned}$$

The bound shown in Equation (2.90), does not account for the fact that the computed value of $\mathbf{R}_1$ is not equal to the true value of $\mathbf{R}_1$. However, the work of [11] shows that the difference in $\mathbf{R}_1$ is a relatively small factor.

### 2.4.2 Algorithm II

A simplified version of Algorithm II is considered, where none of the rows of the state-space matrix are discarded. It is assumed again that the columns of $\mathbf{Z}_2$ are solved using iterative refinement. The algorithm is shown in Equations (2.91)–(2.92).

$$\begin{aligned} \mathbf{W}^T\mathbf{Z}_2 &= \mathbf{I} && (2.91) \\ \mathbf{S} &= \mathbf{Z}_2^T\mathbf{V} && (2.92) \end{aligned}$$

According to the analysis presented in § 2.4.1, the component error in the state-space matrix is given by Equation (2.93).

$$\begin{aligned} |\delta\mathbf{S}| &\leq \left|\mathbf{R}_2^T\mathbf{W}^{-1}\mathbf{V}\right| && (2.93) \\ &\leq \epsilon\,(n+1)\left|\mathbf{Z}_2\right|^T\left|\mathbf{W}\right|\left|\mathbf{W}^{-1}\right|\left|\mathbf{V}\right| \end{aligned}$$

If the matrix $\mathbf{W}$ is symmetric, it should be noted that $(\delta\mathbf{Z}_1)^T = \delta\mathbf{Z}_2$, and the difference in the componentwise error for both methods depends on how the error is propagated in the matrix-matrix product $\delta\mathbf{Z}\mathbf{V}$.

### 2.4.3 Algorithm III

Error is only introduced in the third algorithm from solving the linear system, shown in Equation (2.94).

$$\mathbf{W}\mathbf{S} = \mathbf{V} \qquad (2.94)$$

The corresponding bound on the component error in the state-space matrix is shown in Equation (2.95).

$$|\delta \mathbf{S}| \leq \epsilon\,(n+1)\left|\mathbf{W}^{-1}\right||\mathbf{W}|\,|\mathbf{Z}_1\mathbf{V}| \tag{2.95}$$

It can be seen that the upper bound derived in Equation (2.95) for the third algorithm is tighter than the bound derived in Equation (2.90) for the first algorithm.

## 2.5 Stability of DAE

If the requirements of Theorem 2.3.2 are met, asymptotic stability of the linearized DAE implies local asymptotic stability of the original DAE. More general comparisons of the solution behavior of the original DAE to the linearization are discussed in [35, 149]. The stability of the linearized DAE may be determined by examining the eigenvalues of the system to check for right-half-plane poles. Two possible methods to calculate the eigenvalues of the linearized DAE are:

1. examining the generalized eigenvalues of the matrix pencil, $([f_x\ f_y] + \lambda\,[f_{\dot{x}}\ 0])$,
2. and, examining the eigenvalues of the rearranged, explicit state-space model [137].

Rearrangement of the linearized DAE into state-space form eliminates the infinite eigenvalues associated with the algebraic variables. In a typical problem, there can be many thousands of eigenvalues associated with the algebraic variables and the elimination of the algebraic variables can reduce the size of the eigenvalue problem to the point where it is tractable with standard dense eigenvalue codes [10].

Robust methods exist to determine the eigenvalues of a dense matrix pencil [60, 59]. However, the size of the DAE system may often preclude the use of algorithms suitable for dense problems. Algorithms exist for the computation of a few eigenvalues of a sparse matrix pencil [189, 142]. At the core of these algorithms is the repeated *LU* factorization of the shifted coefficient matrix $([\mathbf{f_x}\ \mathbf{f_y}] + \mu\,[\mathbf{f_{x'}}\ \mathbf{0}])$. The set of indices corresponding to the sparsity pattern of the shifted coefficient matrix must be a superset of the indices corresponding to the sparsity of pattern of $[\mathbf{f_{x'}}\ \mathbf{f_y}]$.

Hence, rearrangement of the DAE into explicit state-space form followed by an eigenvalue calculation will be computationally more efficient than directly computing the eigenvalues of the matrix pencil. Furthermore, some authors suggest that infinite or almost infinite eigenvalues give rise to ill-conditioned eigenvalues that can affect otherwise well conditioned eigenvalues [125].

### 2.5.1 Eigenvalues of Explicit State-Space Model

The dimension of $\mathbf{A}$ in Equation (2.53) is equal to the number of states in the model, and may be very large. It is important that sparsity is exploited in calculation of the eigenvalues of $\mathbf{A}$. If only a small number of eigenvalues need to be calculated (such as the ones with the smallest and largest real parts), several methods exist that exploit the sparsity of $\mathbf{A}$ [192, 193]. In particular there are two classes of methods (a subspace method and Arnoldi's method) for which routines exist in the Harwell Subroutine Library [73, 74, 199] and an implementation called ARPACK [142].

Furthermore, the eigenvalue problem can be decomposed into a series of smaller eigenvalue problems for some systems, if the matrix $\mathbf{A}$ can be permuted into block upper triangular form by a series of symmetric permutations.

**Theorem 2.5.1.** *[100] If $\mathbf{A} \in \mathbb{R}^{n \times n}$ is partitioned as follows,*

$$\mathbf{Q}^T \mathbf{A} \mathbf{Q} = \mathbf{T} = \begin{bmatrix} \mathbf{T}_{11} & \mathbf{T}_{12} \\ \mathbf{0} & \mathbf{T}_{22} \end{bmatrix}$$

*then $\lambda(\mathbf{A}) = \lambda(\mathbf{T}_{11}) \cup \lambda(\mathbf{T}_{22})$.*

The permutation matrices $\mathbf{Q}$ and $\mathbf{Q}^T$ can be found by application of Tarjan's algorithm to the occurrence information of $\mathbf{A}$ [70]. It should be noted that a zero free traversal is not obtained using row permutations, since unsymmetric permutations would change the structure of the eigenvalue problem.

A code has been written to calculate a few eigenvalues of special character (i.e. the largest and smallest in magnitude) of a sparse unsymmetric matrix. The matrix is permuted to block upper triangular form, by application of Tarjan's algorithm, and the eigenvalues are determined by solving a sequence of smaller sub-problems. The code has been included in the DAEPACK libraries [228] and is based on the ARPACK eigenvalue code [142], the LAPACK eigenvalue code [10], and the HSL routines MA48 and MC13 [3, 72].

### 2.5.2 Error Analysis of Stability Calculation

The state-space matrix, $\mathbf{S}$, is defined by Equation (2.53) and the error in the state-space matrix is $\delta\mathbf{S}$. It is of particular interest to bound how much the eigenvalues of the sub-matrix, $\mathbf{A}$, are shifted by the error in $\mathbf{S}$. If the real parts of the eigenvalues of $\mathbf{A}$ change sign, the qualitative behavior of the solution of the derived state-space model will be different from the solution of the implicit linearization of the original DAE. The component of the error, $\delta\mathbf{S}$, corresponding to the sub-matrix $\mathbf{A}$ is defined as $\delta\mathbf{A}$. Theorem 2.5.2 is useful for analyzing the sensitivity of an individual eigenvalue.

**Theorem 2.5.2.** *[100]: If $\lambda(\epsilon)$ is defined by,*

$$(\mathbf{A} + \epsilon\mathbf{F})\,\mathbf{x}(\epsilon) = \lambda(\epsilon)\,\mathbf{x}(\epsilon)\,,$$

*$\lambda(0)$ is a simple eigenvalue of $\mathbf{A}$ and $||\mathbf{F}||_2 = 1$, then*

$$\left.\frac{\mathrm{d}\lambda}{\mathrm{d}\epsilon}\right|_{\epsilon=0} \leq \frac{1}{|\mathbf{y}^H\mathbf{x}|}$$

*where $\mathbf{x}$ and $\mathbf{y}$ satisfy $\mathbf{A}\mathbf{x} = \lambda\mathbf{x}$ and $\mathbf{y}^H\mathbf{A} = \lambda\mathbf{y}^H$, respectively.*

If $\epsilon$ is calculated according to $\epsilon\mathbf{F} = \delta\mathbf{A}$, then $\epsilon = ||\delta\mathbf{A}||_2$. An estimate of the upper bound in the change of $\lambda$ is given below.

$$\delta\lambda \approx \frac{||\delta\mathbf{A}||_2}{|\mathbf{y}^H\mathbf{x}|} \leq \frac{||\delta\mathbf{A}||_F}{|\mathbf{y}^H\mathbf{x}|}$$

where, $||\delta\mathbf{A}||_F$ is the Frobenius norm.

## 2.6 Results

Algorithm III is tested on a model of short-term epidermal growth factor receptor (EGF Receptor) signaling from [132] in § 2.6.1. The error in the solution and the speed of Algorithms I, II and III for randomly generated sparse matrices is shown in § 2.6.2. The superior speed of Algorithms II and III for calculating the state-space model of a semi-discretization of a PDAE is shown in § 2.6.3. Finally, Algorithms II and III are applied to a model of a distillation column in § 2.6.4.

While it was relatively straightforward to compare the three proposed algorithms in terms of computational speed, it was a more challenging task to compare the accuracy of the three algorithms. In the testing of algorithms it was assumed that a matrix-matrix product could be calculated to a far higher accuracy than the solution, $\mathbf{X}$, of $\mathbf{AX} = \mathbf{B}$. The justification was that short vector inner-products were necessary to calculate a matrix-matrix product, due to the sparsity of the matrices, i.e., $O(1)$ floating-point operations. In comparison, despite sparsity of the $LU$ factors of $\mathbf{A}$, it was common for at least $O(n)$ operations to be necessary during the forward and back-substitution phases when calculating $\mathbf{X}$ from $\mathbf{AX} = \mathbf{B}$.

### 2.6.1 Short-Term EGF Receptor Signaling Problem

Algorithm III was applied to a model of short-term EGF receptor signaling due to [132]. The original model equations are shown as ABACUSS II input code in Appendix B, § B.3. The sparsity pattern of the state-space model was automatically generated and is shown in Figure 2-7. It can be seen that the model is relatively sparse. The original model equations were simulated along with the approximate state-space model. The response of key signaling molecules in the model to a perturbation in the total epidermal growth factor concentration was compared to the output of the state-space model. It can be seen that the qualitative behavior of the model is preserved and that the discrepancy between the original model and the approximate state-space model is small. The eigenvalues of the state-space approximation were calculated. All of the eigenvalues were real and negative.

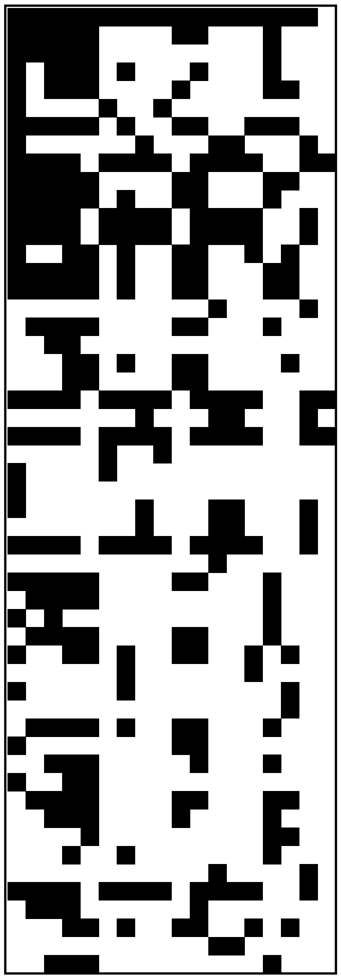
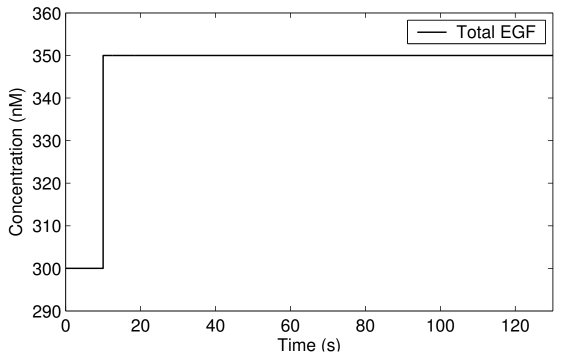
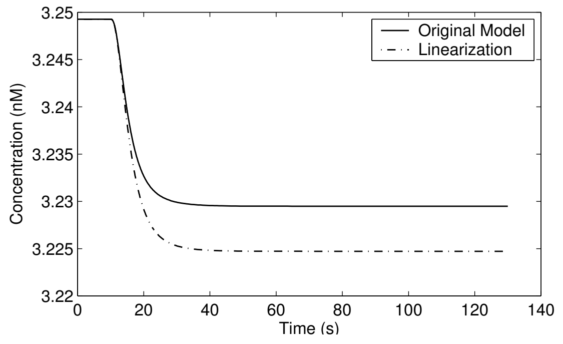


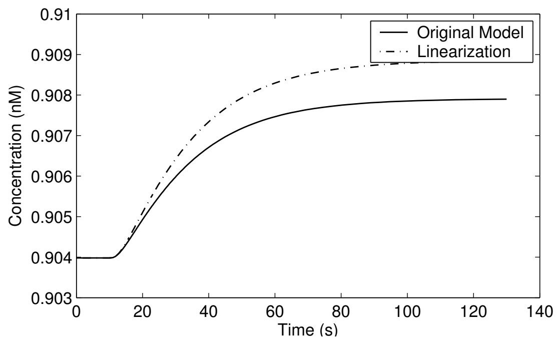
Figure 2-7: Sparsity pattern of short-term EGF signaling model [132]



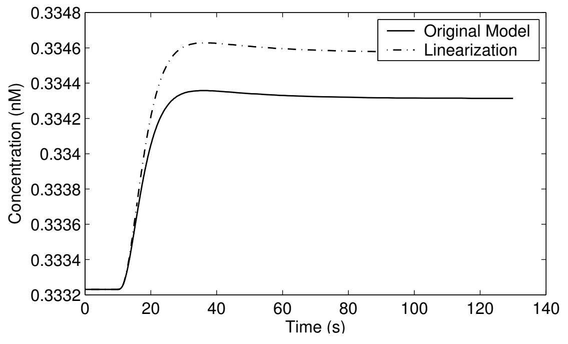
(a) Total EGF



(b) Total Phosphorylated PLC$\gamma$



(c) EGFR-Grb2-SOS Ternary Complex



(d) EGFR-Shc-Grb2-SOS Complex

Figure 2-8: Comparison of a short-term EGF signaling simulation [132] to the explicit state-space approximation

The time constant of the fastest process was 0.1s and the time constant of the slowest process was 20s. This information might be useful to an experimentalist since time-series measurements need to be taken at a rate faster than the fastest time constant to analyze some of the system dynamics.

### 2.6.2 Accuracy Testing Methods

There were two major difficulties in generating realistic test problems:

1. structural sparsity in the state-space model will only occur if the matrix $\mathbf{W}$ is reducible,

2. and, for realistic problems, it is easy to estimate tightly the sparsity of $\mathbf{X}$ in the solution of $\mathbf{AX} = \mathbf{B}$, however, it is difficult to estimate tightly the sparsity of $\mathbf{B}$ in the calculation of the matrix-matrix product $\mathbf{B} = \mathbf{AX}$.

For example, a naïve method to determine the accuracy of the proposed algorithms could be to generate random sparse matrices $\mathbf{W}$, $\mathbf{S}$, calculate the product, $\hat{\mathbf{V}}$, apply the rearrangement algorithms to the matrices, $\mathbf{W}$, $\hat{\mathbf{V}}$, and compare the solution, $\hat{\mathbf{S}}$ to the originally generated $\mathbf{S}$. If $\mathbf{W}$ was irreducible this would not be a fair test, since despite $\mathbf{S}$ was generated as a sparse matrix, the result from the rearrangement algorithm would be a dense matrix with almost all of the entries close or identically equal to zero, i.e., by construction, the matrix $\mathbf{V}$ leads to vast amounts of numerical cancellation in $\mathbf{S}$. Physically realistic problems rarely show this property.

An alternative test was to generate sparse random matrices, $\mathbf{W}$ and $\mathbf{V}$ in Matlab[3] by the function:

```
sprand(n,m,fill,1/cndno)
```

where $n$ and $m$ were the matrix dimensions, $fill$ was the fractional of non-zero entries in the matrix, and *cndno* is the condition number of the matrix. Algorithm II was applied to the matrices, $\mathbf{W}$ and $\mathbf{V}$ to calculate $\hat{\mathbf{S}}$ with a structure consistent with $\mathbf{W}$

[3]Matlab is a registered trademark of The Mathworks, Inc., Natick, MA.

and $\mathbf{V}$. The matrix $\hat{\mathbf{V}}$ was calculated from Equation (2.96) to generate a set of test matrices $\mathbf{W}$, $\hat{\mathbf{V}}$, and $\hat{\mathbf{S}}$ that were consistent.

$$\hat{\mathbf{V}} = \mathbf{W}\hat{\mathbf{S}} \tag{2.96}$$

However, this approach has the disadvantage that the recalculated $\hat{\mathbf{V}}$ has a significant number of zero or close to zero entries that were not in the original matrix $\mathbf{V}$, i.e., the structure of the matrix $\hat{\mathbf{V}}$ no longer completely represents a physically realistic problem. Many of the equations in the DAE are of the form $\mathbf{g}(\mathbf{y}) = 0$, i.e., the row in the Jacobian $\mathbf{V}$ corresponding to this equation is completely zero. However, it is impossible to preserve this row as structurally zero when calculating it from $\mathbf{W}$ and $\hat{\mathbf{S}}$, since by definition $\hat{\mathbf{S}}$ will contain entries linking entries in $\mathbf{y}$ to inputs or states in the model. A possible solution would be to eliminate entries in $\hat{\mathbf{V}}$ that did not occur in the original matrix $\mathbf{V}$. However, this would lead to a matrix $\hat{\mathbf{V}}$ that was numerically inconsistent with the matrices $\mathbf{W}$ and $\hat{\mathbf{S}}$. It was felt that the most reasonable compromise was to calculate $\hat{\mathbf{S}}$ and $\hat{\mathbf{V}}$ without eliminating entries in $\hat{\mathbf{V}}$.

### Numerical Tests with Sparse Random Test Matrices

All three algorithms were applied to the matrices $\mathbf{W}$ and $\hat{\mathbf{V}}$. The component error was calculated according to Equation (2.97)

$$\text{Error} = \max_{ij} \frac{|s_{ij} - \hat{s}_{ij}|}{|\hat{s}_{ij}|} \tag{2.97}$$

For each algorithm, $\mathbf{W}$ was sparse $LU$ factored with and without block decomposition. The algorithms were implemented in Matlab and the code is included in Appendix A, § A.2. The results for 500 test problems are summarized in Table 2.4 for $fill =$ 5 entries/row, $cndno = 10^6$.

Methods II and III require substantially fewer floating point operations in comparison to Method I. Which of Method II and III is fastest will depend on the ratio $n_u/n_{y_1}$.

Table 2.4: Comparison of error and cost without elimination of entries in $\hat{\mathbf{V}}$

| Size | Alg. | Error | | FLOPS |
|---|---|---|---|---|
| | | Mean | Std. Dev. | |
| $n_x = 50$ | I | 1.01E+19 | 2.26E+20 | 4.4E+05 |
| $n_y = 50$ | I BD | 3.12E-06 | 6.77E-05 | 3.7E+05 |
| $n_u = 20$ | II | 6.50E+01 | 1.41E+03 | 2.4E+05 |
| $n_{y_1} = 0$ | II BD | 3.08E-08 | 3.85E-07 | 2.0E+05 |
| | III | 7.68E+18 | 1.72E+20 | 1.0E+05 |
| | III BD | 3.26E-06 | 7.12E-05 | 7.4E+04 |
| $n_x = 20$ | I | 1.02E+04 | 1.92E+05 | 7.4E+05 |
| $n_y = 80$ | I BD | 3.40E-07 | 6.68E-06 | 6.5E+05 |
| $n_u = 50$ | II | 9.98E-01 | 1.77E+01 | 1.7E+05 |
| $n_{y_1} = 0$ | II BD | 5.06E-08 | 5.65E-07 | 1.4E+05 |
| | III | 5.11E+04 | 1.13E+06 | 2.1E+05 |
| | III BD | 1.07E-07 | 1.34E-06 | 1.6E+05 |

The high standard deviation in the error indicates that a few pathological matrices cause the mean error to be large. All methods perform well in terms of numerical error for most matrices. Usually, Methods II and III have less error than Method I due to the reduced number of floating point operations. Block decomposition substantially improves the error in all three methods.

Typically, the element of $\mathbf{S}$ with the largest relative error is small, and non-zero. The state-space matrices that have large amounts of error contain entries of widely varying magnitude. Error due to gross cancellation can occur when $\mathbf{W}$ has off-diagonal blocks after block triangularization that link small elements in the solution to large elements in the solution during the back-substitutions. Block decomposition reduces the error in the solution, since off-diagonal blocks that link small elements to large elements are not factored. It should be noted that spurious entries in $\hat{\mathbf{V}}$ are likely to connected to elements in $\mathbf{S}$ by off-diagonal elements in $\mathbf{W}$, i.e., the large difference in error between algorithms with block decomposition and without block decomposition may be in part attributable to the method of construction of the test matrix $\hat{\mathbf{V}}$. Results for identical tests, where the spurious entries $\hat{\mathbf{V}}$ were

Table 2.5: Comparison of error and cost with elimination of entries in $\hat{\mathbf{V}}$

| Size | Alg. | Error | | FLOPS |
|---|---|---|---|---|
| | | Mean | Std. Dev. | |
| $n_x = 50$ | I | 3.16E-06 | 5.53E-05 | 2.9E+05 |
| $n_y = 50$ | I BD | 1.99E-06 | 4.03E-05 | 2.3E+05 |
| $n_u = 20$ | II | 5.94E-06 | 8.81E-05 | 1.6E+05 |
| $n_{y_1} = 0$ | II BD | 2.00E-06 | 2.99E-05 | 1.2E+05 |
| | III | 2.07E-06 | 2.70E-05 | 9.3E+04 |
| | III BD | 3.72E-07 | 6.09E-06 | 6.7E+04 |
| $n_x = 20$ | I | 6.51E-06 | 1.39E-04 | 2.9E+05 |
| $n_y = 80$ | I BD | 2.52E-08 | 4.42E-07 | 2.4E+04 |
| $n_u = 50$ | II | 2.63E-07 | 2.82E-06 | 8.1E+04 |
| $n_{y_1} = 0$ | II BD | 2.37E-08 | 3.40E-07 | 6.0E+04 |
| | III | 3.79E-06 | 7.90E-05 | 1.8E+05 |
| | III BD | 1.84E-08 | 3.49E-07 | 1.3E+05 |

eliminated are shown in Table 2.5 These results would suggest that there would be some improvement in accuracy due to block decomposition for models that represent physical systems.

The method that provides the most accurate solution will depend on whether the elements of a column of $\mathbf{S}$ vary over a large range compared with whether the elements of a row of the inverse of $\mathbf{W}$ vary over a large range. The accuracy of each of the different methods was tested on two application problems. The applications were selected to contain a large number of algebraic variables to be eliminated from the state-space model. This feature is a characteristic of many science and engineering problems. The results are summarized in § 2.6.3–2.6.4.

### 2.6.3 Diffusion Problem

The following example looks at the discretization of a coupled PDE and ODE. The physical situation is shown in Figure 2-9. Diffusion of signaling molecules is a common process in cell signaling. There are two tanks coupled together by a porous membrane. The concentration in the tanks is given by $C_0$ and $C_{n+1}$ respectively. The

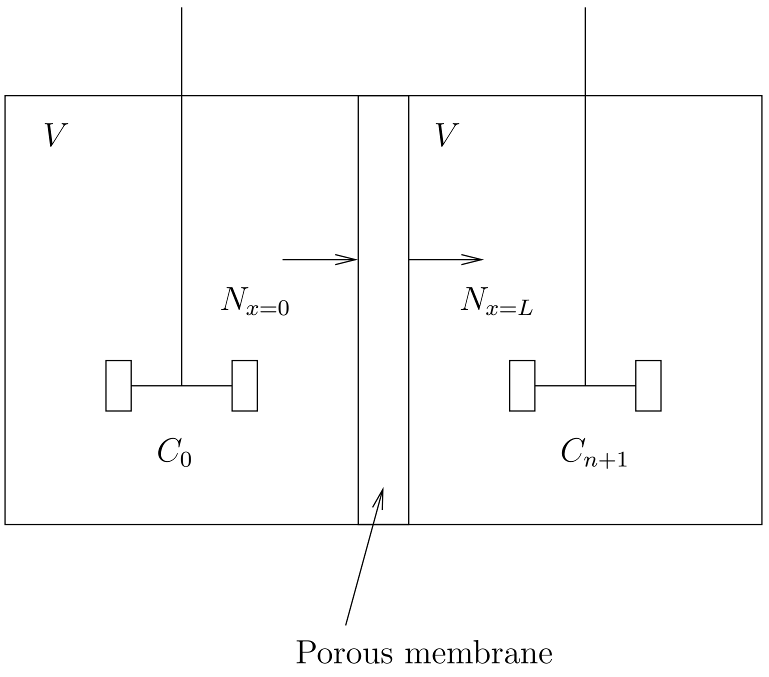


Figure 2-9: Diffusion between two well-mixed tanks

flux of material leaving the left hand tank is given by $N_{x=0}$ and the flux of material entering the right hand tank is given by $N_{x=L}$. A simple analytical solution to this problem exists, but for a more complicated geometry it would be necessary to solve this problem numerically. After sufficiently long time,

$$t >> \frac{L^2}{D},$$

and subject to the geometric constraint,

$$\frac{2ALK}{V} \ll 1,$$

where $L$ is the membrane thickness, $D$ is the diffusivity of the solute, $A$ is the surface area of the membrane, $V$ is the volume of the tanks, and $K$ is the partition coefficient of the membrane, the behavior of diffusion in the membrane can be modeled as pseudo-steady. The resulting system can be discretized into the following DAE, where $C_i$, $i = 1 \ldots n-2$ corresponds to the concentration at mesh points inside the membrane, spaced at $\Delta x$ intervals.

$$\begin{aligned} V\dot{C}_0 + AN_{x=0} &= 0 & (2.98)\\ V\dot{C}_{n+1} - AN_{x=L} &= 0 & (2.99)\\ C_1 - KC_0 &= 0 & (2.100)\\ C_n - KC_{n+1} &= 0 & (2.101)\\ N_{x=0} + D\left(\frac{C_3 - C_1}{2\Delta x}\right) &= 0 & (2.102)\\ N_{x=L} + D\left(\frac{C_n - C_{n-2}}{2\Delta x}\right) &= 0 & (2.103)\\ \frac{C_i - 2C_{i+1} + C_{i+2}}{2\Delta x^2} &= 0 & (2.104) \end{aligned}$$

The resulting DAE system was transformed into state-space form using Algorithms I, II, and III. The code was written in Matlab and is included in Appendix A, § A.3. All methods used dense linear algebra and were performed in Matlab. The analytical solution for the state variables, $C_0$ and $C_{n+1}$, are given by Equations (2.105)–(2.107).

$$\begin{aligned} C_0 &= const\left(1 + e^{-\frac{t}{\tau}}\right) & (2.105)\\ C_{n+1} &= const\left(1 - e^{-\frac{t}{\tau}}\right) & (2.106)\\ \tau &= \frac{VL}{2ADK} & (2.107) \end{aligned}$$

The eigenvalues of the system are $\left(0, -\frac{1}{\tau}\right)$. It should be noted that the discretized solution should be exact (to within roundoff error) since the flux across the membrane is constant.

The estimated number of floating point operations (FLOPS) for a system of 100 variables, were 3080740 for Algorithm I, 789412 for Algorithm II and 788580 for Algorithm III. It should be noted that the $LU$ factors of $\mathbf{W}$ are very sparse, but the inverse is dense. It can be seen that there is considerable computational advantage in Algorithms II and III compared to I.

The model was run at 100 mesh points and $\frac{1}{\tau} = 0.3030$. The state-space linearization based on all three methods solved for eigenvalues of $(0, -0.3030)$. However, if the matrix inverses required by Algorithms I and II are calculated by Gauss-Jordan elimination, rather than $LU$ factorization, the eigenvalues of the state-space linearization are $(0, -0.3333)$. It can be seen that great care must be taken when calculating the inverse of $\mathbf{W}$.

### 2.6.4 Distillation Problem

Finally, the methods were tested on a model of a benzene-toluene distillation column, written in the ABACUSS II input language (Appendix B, § B.4). This model was chosen because of its size and complexity. Currently, there are very few cell signaling models in the literature that are of a comparable size. There are many benefits to writing the model in a structured modeling environment [16]. Fortran code corresponding to the model was automatically generated by ABACUSS II. The automatic differentiation tool DAEPACK [228] was used to generate code implementing the Jacobians matrices $\mathbf{W}$, and $\mathbf{V}$, necessary for construction of the state-space approximation. Algorithms II and III were implemented as components of the DAEPACK library. An ABACUSS II input file was automatically generated corresponding to the state-space form of the model. The distillation model had $n_x = 177$ states, $n_y = 3407$ algebraic variables and $n_u = 14$ inputs. There were 929 identity equations in the model that were eliminated exactly. All of the algebraic variables were eliminated from the state-space model.

The sparsity pattern of the state-space model was generated and is shown in Figure 2-10. There are fifteen groups of structurally orthogonal columns in the state-space model. A disadvantage of Algorithm II is that it requires the generation of the sparsity pattern of the inverse $\mathbf{W}^{-1}$. There were 113347 non-zero entries in the matrix $\mathbf{W}^{-1}$ compared with 1835 entries in the state portion of the state-space model. The speed and accuracy of the algorithms for a 128Mb 450Mhz PIII computer are summarized in Table 2.6. The times were determined with the LAPACK [10] routine DSECND. Algorithm II is slower than Algorithm III for the following reasons,

1. it takes significantly more time to determine the occurrence information of $\mathbf{W}^{-1}$

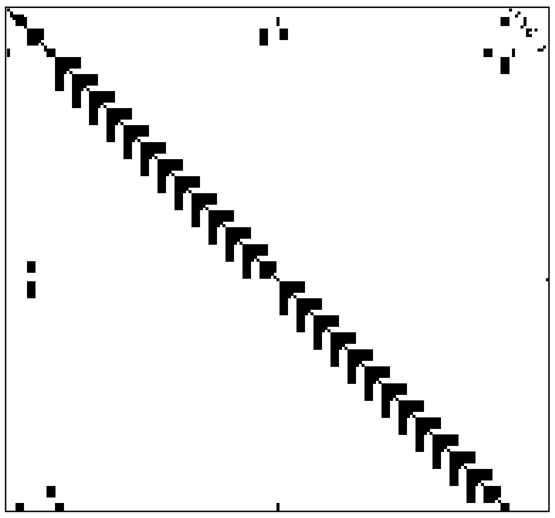


Figure 2-10: Sparsity pattern of state-space approximation of a distillation model

Table 2.6: Distillation model results

| Algorithm | Error | Time (s) |
|---|---|---|
| II (with identity elimination) | 1.8E-04 | 0.4490 |
| II (without identity elimination) | 6.6E-6 | 0.7787 |
| III (with identity elimination) | 7.4E-5 | 0.0431 |
| III (without identity elimination) | 1.2E-4 | 0.0566 |

compared with the generation of the occurrence information of **S**,

2. it takes significantly more time to determine partitions of structurally orthogonal columns of $\mathbf{W}^{-1}$ compared with determining the partitions of **S**, and,

3. more forward and back substitutions were required by Algorithm II to calculate the state-space matrix.

The accuracy of the algorithms was determined by comparing **S** to $\hat{\mathbf{S}}$, using the method outlined in § 2.6.2.

## 2.7 Summary

It was shown in this Chapter how to formulate a cell signaling model as a system of DAEs (§ 2.1.2). The advantage of this approach is that there is a smaller risk of making errors for non-constant volume systems. It was shown for an autonomous DAE, that local stability of the DAE can be determined from the stability of the state-space approximation (Theorem 2.3.2). Three new methods for transforming the linearization of an index one DAE into state-space form are demonstrated in this Chapter. One of the methods is a modification of an existing algorithm. Two of the new methods show considerable advantage in terms of computational expense. From the point of view of numerical error, if there are entries of widely varying magnitude in a row of the state-space matrix, but entries in each column do not vary too much, Algorithm III is preferred. If the entries of each row of the inverse of $\mathbf{W}$ do not vary too much, Algorithm II may be preferred.

# Chapter 3

# Bayesian Reasoning

Scientists and engineers are constantly faced with decisions based on uncertain information. Clinicians routinely make decisions based on risk: is it safer to operate and remove a tumor, or treat the tumor with an anti-cancer drug? To assess a risk, it is important to have some method of predicting outcomes and quantifying the accuracy of such predictions. To make predictions about a system requires some form of qualitative or quantitative model. As discussed in Chapter 1, qualitative modeling is often insufficient to make predictions about diseases caused by complex network interactions. In contrast, quantitative modeling of the system can yield greater insight. Work was devoted in Chapter 2 to building and analyzing such detailed models of cell-signaling networks. However, we are often faced with the situation where there is insufficient *a priori* knowledge to build a mechanistic model. Hence, we wish to find some compromise between a mechanistic model and a qualitative description of the system. It is therefore important to have some way of describing less than perfect correlations. Furthermore, the work in Chapter 2 does not provide a method to compare model predictions quantitatively with experimental data to determine the accuracy of the model. Clearly, it is important to be confident about the accuracy of a model when critical decisions are based on predictions from the model. It will be shown in this Chapter how the theory of probability can be used to address many of these shortcomings.

## 3.1 Decision Making from Models

To motivate the discussion, we shall first discuss a classic example of risk analysis: the causes of the Challenger space shuttle disaster. Many aspects of the launch decision making process are similar to decisions made in the biological sciences (drug approval, decision to operate on a person, etc.): the potential consequences of the decision were of high cost, and the decision was made based on predictions from an engineering model. On the night before the launch a decision had to be made whether there was an unacceptable risk of catastrophic failure. Initially, one might be tempted to think that any possibility of an accident resulting in a fatality is unacceptable. However, a little reflection might make one realize that this line of thought is indeed false. Even if it could be guaranteed the shuttle launch would be successful, there would be a small but finite probability that one of the engineers driving to the launch would be involved in a fatal car crash. It is almost always impossible to perform a task without risk of adverse consequences. However, one hopes that the benefits from performing such a task are sufficient compensation for the risks of an adverse consequence. For a more detailed statistical interpretation of the events that cause the disaster see [67].

**Example 3.1.1.** The engineers had to decide whether to postpone the launch due to cold weather. The space shuttle had three field joints on each of its two solid booster rockets. Each of these six joints contained two O-rings. The engineers knew that failure of one of these rings would be catastrophic. The previous lowest launch temperature was 53F. At this temperature, the engineers knew that the probability of an O-ring failure was acceptable, i.e., $\Pr(\text{O-ring fails}|T = 53\text{F})$ was vanishingly small. However, the temperature forecast for the night before the launch was an uncharacteristically cold 31F. The engineers had to evaluate $\Pr(\text{O-ring fails})$ and see whether the risk of the launch failing was acceptable. Unfortunately, the risk of launch failure was incorrectly evaluated with devastating consequences.

Several key concepts are highlighted in Example 3.1.1:

1. Decisions are always made conditionally based on some information.

2. Decisions are based on models.
3. Decision making is somehow related to risk.
4. Risk is somehow related to probability.

Let us justify these statements. In the previous example, the decision to launch the shuttle was based on the weather forecast (information) and a model of temperature dependent material failure. The engineers had to evaluate the risk associated with the launch, and this was based on the probability of O-ring failure. Furthermore, this example illustrates that it is extremely important to take into account all possibilities when making a decision. The decision whether to launch may change if it is predicted that there is a 1% chance the temperature is 31F and a 99% chance the temperature is over 60F. It is important to distinguish between the quantities Pr(O-ring fails) and Pr(O-ring fails$|T = 31$F). These two probabilities are usually not equal. Hopefully, it will become apparent that the probability can be used as a tool in making decisions.

In Example 3.1.1, the engineers needed a model of temperature dependent material failure. Clearly, an important step in making a decision is developing an accurate model. Often there is a folklore among engineers that a model with a small number of parameters will have good predictive power. This is true for models that are *linear* in the parameters. However, for nonlinear models, determining the predictive capability of the model is significantly harder, as demonstrated by Example 3.1.2. As in Chapter 2, the control-engineering convention is adopted: $x$ is a state, $y$ is a measurement or output, and $u$ is an input or manipulated variable.

**Example 3.1.2.** Consider fitting two alternative models defined by Equations (3.1) and (3.2) to the data shown in Table 3.1.

$$M_1 : \quad \hat{x} = 40\sin\left(\hat{\theta}\hat{u}\right) \tag{3.1}$$
$$M_2 : \quad \hat{x} = \hat{\theta}\hat{u}^2 \tag{3.2}$$

How does one decide which is the most appropriate model?

| $x_i$ | $y_i$ |
|---|---|
| 1 | 0.8801 |
| 2 | 3.9347 |
| 3 | 9.4853 |
| 4 | 15.4045 |
| 5 | 24.8503 |

Table 3.1: Data for Example 3.1.2

The Matlab code to perform least squares minimization on this problem is shown in Appendix A, § A.1. It can be seen from Figure 3-1a that both models can fit the data relatively well (assuming "fitting well" means a small value of the sum of the square of the residuals). However, one has the intuition that one would favor the model defined by Equation (3.2) over the model defined by Equation (3.1) given the data. This phenomenon can be explained by considering the *posterior* Probability Density Functions (PDFs) for each model,

$$\mathrm{f}_\theta(\theta|\hat{\mathbf{y}} = \mathbf{y}, M_1)\,,$$

and

$$\mathrm{f}_\theta(\theta|\hat{\mathbf{y}} = \mathbf{y}, M_2)\,.$$

A more thorough discussion of probability density functions is given in § 3.3. For the moment it should be understood that the probability that $\hat{\theta}$ lies in the range $\theta_1 \leq \hat{\theta} \leq \theta_2$ given the data, $\mathbf{y}$ and that the true model is $M_1$ can be derived from the *posterior* PDF:

$$\Pr\Big(\theta_1 \leq \hat{\theta} \leq \theta_2 | \mathbf{y}, M_1\Big) = \int_{\theta_1}^{\theta_2} \mathrm{f}_\theta(\theta|\hat{\mathbf{y}} = \mathbf{y}, M_1)\ \mathrm{d}\theta.$$

The *posterior* PDFs are shown in Figures 3-1b-c. For the model defined by Equation (3.1), it can be seen there are many possible values of $\hat{\theta}$ which are candidate "best-fit" parameter values (Figure 3-1b), hence, one is very uncertain about which value is best. This is an undesirable feature of the first model since uncertainty in

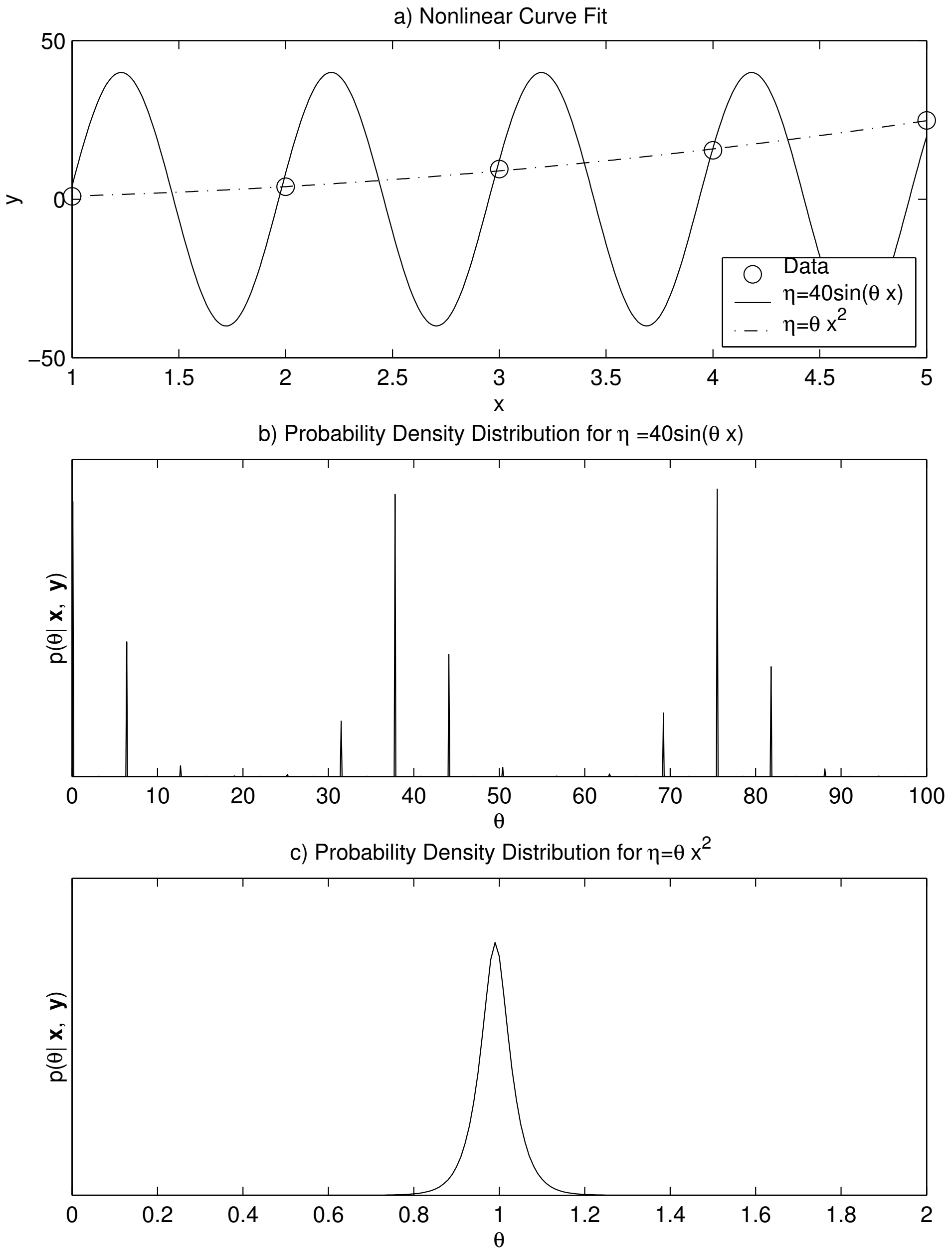


Figure 3-1: Nonlinear curve fits for Example 3.1.2

$\hat{\theta}$ causes large differences in the prediction of $\hat{x}(\hat{u})$ for values of $\hat{u}$ which did not correspond to the existing data. However, if the second model is true, one is relatively certain about the true value of $\theta$ as shown by the probability density function plotted in Figure 3-1c. In some sense, model $M_1$ has a far greater capacity to fit any data than model $M_2$. Hence at the outset, one should be more sceptical of using model $M_1$ than $M_2$ and require greater evidence that model $M_1$ is true than model $M_2$. This provides a qualitative motivation for why one would favor one model over another, even if both of the models fit the data relatively well. It is necessary to understand probability to quantify how much one model is preferred over another. For a thorough discussion of model comparison the reader is referred to Chapter 5 of [123]. In particular, the author considers the problem of determining when little additional benefit is obtained from increasing the complexity of a model.

## 3.2 Rules of Probability

Most people are familiar with probability being some measure of the frequency of an event (e.g., the fraction of times one gets a particular number when dice are rolled). However, this is quite a limited view of probability. Furthermore, it is often assumed that probability is used to describe a random or stochastic process. However, the motion of a die can be described by Newtonian mechanics. (This is a chaotic system, so it is extremely sensitive to initial conditions.) In principle, one could calculate which face the die will land on given the initial condition had been measured with sufficient accuracy. Rather than viewing probability as a frequency, it is more general to view probability as a measure of belief in a proposition.

Probabilities need not (and in some circumstances) should not be equal to frequencies. To force such a correspondence guarantees that *a priori* knowledge is worthless and that data are all one knows. To see this is absurd, consider Example 3.2.1.

**Example 3.2.1.** Suppose a completely fair coin is manufactured and it is known with certainty from the manufacturing process that there is a 50% chance of tossing the coin and obtaining heads and a 50% chance of obtaining tails. The coin is tossed five times and each time the result is heads. If the probability of throwing heads corresponds to the frequency of the result then one would conclude that there is a 100% chance of obtaining heads when the coin is tossed. However, it is already known that the chance of obtaining heads is 50%. It can also be calculated that there is a 3.1% chance of obtaining five heads in a row with a perfectly fair coin. It would be an extremely brave (or foolhardy) person to disregard the *a priori* knowledge when there is a significant probability that the results just happened by chance.

Example 3.2.1 is a characture. Most people would talk about a correspondence of probability to frequency in some limit of many experiments; i.e., one should not draw any conclusions from a small number of trials. However, scientists wish to make inferences based on limited results. Any theory that requires a large number of experiments does not seem too helpful. Furthermore, there are many quantities that are constant but cannot be measured exactly (for example: the speed of light). According to modern physics, it is incorrect to suggest that each time the speed of light is measured the speed is in fact different (even if each time a different value of the measurement is obtained).

Hence, in this thesis probability will always correspond to a degree of belief in a proposition. Rules governing the manipulations of probabilities can be obtained by extending deductive logic. This view of probability is referred to as Bayesian statistics or plausible reasoning. The development of the theory of probability as an extension of deductive logic is abbreviated from [121].

### 3.2.1 Deductive Reasoning

In deductive reasoning, the truth of a proposition is considered and deductions are based on a simple set of rules. Propositions are denoted by capital letters, $A$, $B$, etc. There are only two possible outcomes when belief in a statement is decided: either a statement is true or it is false. The notation, $A = B$ (sometimes written as $A \Leftrightarrow B$), means $A$ always has the same truth value as $B$ (not $A$ and $B$ are identical propositions), i.e., when statement $A$ is true statement $B$ is true and when statement

Table 3.2: Binary truth table for implication

| $A$ | $B$ | $A \Rightarrow B$ |
|---|---|---|
| 0 | 0 | 1 |
| 1 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 1 | 1 |

$A$ is false, statement $B$ is false.

**Definition 3.2.1.** There are three basic operations defined in deductive reasoning: negation, conjunction and disjunction.

1. $A$ is false (negation): $\overline{A}$, $\neg A$
2. $A$ and $B$ (conjunction): $AB$, $A \wedge B$
3. $A$ or $B$ (disjunction): $A + B$, $A \vee B$

The notation, $A \Rightarrow B$, means proposition $A$ implies $B$ and obeys the truth table shown in Table 3.2. The only combination of $A$ and $B$ that is inconsistent with $A \Rightarrow B$ is that $A$ is true and $B$ is false; all other combinations of $A$ and $B$ are consistent with $A \Rightarrow B$. Given $A \Rightarrow B$ then if $A$ is true then $B$ must be true. Likewise, if $B$ is false then $A$ must be false ($\overline{B} \Rightarrow \overline{A}$). However, if $A$ is false, then $A \Rightarrow B$ does not give any information about whether $B$ is true or false. However, if one is just concerned about the plausibility of a statement, then if $A \Rightarrow B$ and $A$ is false then one would assume $B$ is less likely (since at least one reason for $B$ to be true has been removed). It is apparent that deductive reasoning is not sufficient to make scientific inferences since it is impossible to know whether a proposition or statement is completely true or completely false; one hypothesis is either more or less likely than an another. The framework to analyze the situation where propositions may be more or less likely is plausible reasoning and was first developed by [50, 51]. It is possible to derive Bayes' theorem directly from the desiderata of plausible reasoning.

### 3.2.2 Plausible Reasoning

Following the development of [121], we will consider the situation where one wants to express more than just certainty of a proposition. There are three situations where a rational human would make an inference but no conclusion can be formally deduced:

1. if $A \Rightarrow B$ and $B$ is true then one may *infer* that $A$ is more plausible,
2. if $A \Rightarrow B$ and $A$ is false then one may *infer* that $B$ is less plausible, and,
3. if $B$ is true implies $A$ is more plausible and $A$ is true then one may *infer* that $B$ is more plausible.

For example, rain ($A$) implies a cloudy sky ($B$). If the sky is cloudy ($B = 1$) then it is more likely to be raining ($A$ is more plausible). Likewise, if it is not raining $A = 0$, then it is less likely to be cloudy ($B$ is less plausible). To derive a system of plausible reasoning it is necessary to define some desiderata governing how inferences are made (see Definition 3.2.2).

**Definition 3.2.2.** $(A|B)$ denotes the plausibility of statement $A$ given statement $B$ is true. The plausibility of a statement, $(\cdot)$, obeys the following rules:

1. Degrees of plausibility are represented by a real numbers.
2. If $(A|C') > (A|C)$ then $\left(\overline{A}|C'\right) < \left(\overline{A}|C\right)$.
3. If $(A|C') > (A|C)$ and $(B|AC') = (B|AC)$ then $(AB|C') \geq (AB|C)$.
4. Conclusions about a statement that can be reasoned out via more than one route lead to the same probability of the conclusion.
5. All of the evidence must be considered when calculating the probability of a statement.
6. Equivalent states of knowledge lead to equivalent probabilities.
7. Continuity.

The probability of a proposition, $\Pr(A|B)$, is defined as a monotonically increasing function of the plausibility, $(A|B)$.

It is possible to prove the following properties from the desiderata in Definition 3.2.2:

**Theorem 3.2.1.** *By convention, $Pr(A|B) = 0$, if $A$ is false given $B$ is true. Properties 1–4 follow from the Desiderata in Definition 3.2.2:*

1. *Truth - If statement $A$ is true given $B$ is true:*

$$\Pr(A|B) = 1$$

2. *(Mutual) Exclusivity:*

$$\Pr(A|B) + \Pr\left(\overline{A}|B\right) = 1$$

3. *Bayes' Theorem:*

$$\begin{aligned} \Pr(AB|C) &= \Pr(A|C)\Pr(B|AC) \\ &= \Pr(B|C)\Pr(A|BC) \end{aligned}$$

4. *Indifference - If information $B$ is indifferent between mutually exclusive propositions $\{A_1, \ldots, A_n\}$ then:*

$$\Pr(A_i|B) = \frac{1}{n}, \quad 1 \le i \le n.$$

The proof of the properties described in Theorem 3.2.1 is quite complicated and the reader is referred to the seminal work [50, 51] or to [121] for an explanation. How the plausibility and the probability of a statement is related has not been described. All it is necessary to know is that the probability is a monotonically increasing function of the plausibility; the assignment of numerical values to probabilities is defined by Property 4 of Theorem 3.2.1.

From Exclusivity and Bayes' theorem it follows that (Chapter 2 of [121]):

$$\Pr(A+B|C) = \Pr(A|C) + \Pr(B|C) - \Pr(AB|C)\,, \tag{3.3}$$

and if the propositions, $\{A_1, \ldots, A_m\}$, are mutually exclusive:

$$\Pr(A_1 + \cdots + A_m|B) = \sum_{i=1}^{m} \Pr(A_i|B)\,. \tag{3.4}$$

### 3.2.3 Marginalization

An important corollary can be obtained from Theorem 3.2.1:

**Corollary.** *Assuming the Desiderata from Definition 3.2.2 and the propositions*

$$\{A_1, \ldots, A_n\}\,,$$

*are mutually exclusive it follows that:*

$$\Pr(C|\,(A_1 + \ldots)\,X) = \frac{\sum_{i=1}^{n} \Pr(C|A_i X)\Pr(A_i|X)}{\sum_{i=1}^{n} \Pr(A_i|X)}\,. \tag{3.5}$$

The formula in Equation (3.5) is often referred to as the marginalization formula. The corollary is useful in determining how much one believes a statement given one of many mutually exclusive statements may be true, as demonstrated by Example 3.2.2.

**Example 3.2.2.** There are three dice, one with four faces, one with five faces and another with six faces. What is the probability of rolling a five, given one of the dice were rolled?

It is necessary to define the following propositions:

1. $A_1$ a die with four faces was rolled.

2. $A_2$ a die with five faces was rolled.

3. $A_3$ a die with six faces was rolled.

4. $B$ a die was rolled. $B = A_1 + A_2 + A_3$.

5. $C$ a five was rolled.

It is assumed that each score is equally likely, since there is no additional information about the dice. From the principle of indifference it follows that: $\Pr(A_i) = 1/3, \quad i = 1 \ldots 3$, and $\Pr(C|A_1) = 0$, $\Pr(C|A_2) = 1/5$, and $\Pr(C|A_3) = 1/6$. Making the necessary substitutions it follows that:

$$\begin{aligned}
\Pr(C|BX) &= \frac{\sum_{i=1}^{3} \Pr(C|A_iX)\Pr(A_iX)}{\Pr(B|X)} \\
&= \frac{1}{3}\left(0 + \frac{1}{5} + \frac{1}{6}\right) \\
&= \frac{11}{90}.
\end{aligned}$$

Marginalization is important since it allows one to relax assumptions and make more general statements.

### 3.2.4 Independence

Often the situation arises where two or more propositions are *independent*. For example, one might reasonably expect the propositions:

A: The score on the first roll of a die is six.

B: The score on the second roll of a die is six.

to be independent (both physically and logically). Care must be taken since two propositions can be physically independent without being independent in the sense of probability. For example, it is well known that some fraction of patients who are treated with a placebo for some diseases will report an improvement in their symptoms. Hence, the propositions:

A: The patient is treated with a placebo.

B: The patient reports an improvement in their symptoms.

may appear to be physically independent, but are not independent in the sense of probability. Formally, the independence of propositions $A$ and $B$ is defined as:

$$\Pr(A|BC) = \Pr(A|C) \tag{3.6}$$

and follows directly from Desiderata 6 of Definition 3.2.2. An important corollary to Property 3 of Theorem 3.2.1 can be obtained trivially.

**Corollary.** *If two statements are independent, then Equation (3.6) holds by definition. Substituting Equation (3.6) into Bayes' theorem yields Equation (3.7).*

$$\Pr(AB|C) = \Pr(A|C)\Pr(B|C) \tag{3.7}$$

### 3.2.5 Basic Inference

How these rules can be used to solve inference problems is now demonstrated in Example 3.2.3.

**Example 3.2.3.** There are three different possible mechanisms for ligand binding and it is known with certainty that one of the mechanisms is correct. Let the statements $A$, $B$ and $C$ be defined as:

$A$: mechanism one is true,

$B$: mechanism two is true, and,

$C$: mechanism three is true.

It is assumed that each mechanism is equally likely, $\Pr(A) = \Pr(B) = \Pr(C) = \frac{1}{3}$. An experiment is performed which categorically excludes at least one of the mechanisms. What is the probability that either one of the remaining mechanisms is the correct model?

For arguments sake let mechanism three be excluded after the experiment. Let the statement $X$ be defined as:

$X$: the experiment excludes mechanism three

Applying Bayes' theorem yields Equations (3.8)–(3.9). Equation (3.11) is obtained by exclusivity (only one of the three different mechanisms is correct).

$$\begin{aligned}
\Pr(A|X) &= \frac{\Pr(X|A)\Pr(A)}{\Pr(X)} && (3.8)\\
\Pr(B|X) &= \frac{\Pr(X|B)\Pr(B)}{\Pr(X)} && (3.9)\\
\Pr(C|X) &= 0 && (3.10)\\
1 &= \Pr(A|X) + \Pr(B|X) + \Pr(C|X) && (3.11)
\end{aligned}$$

Assuming that without prior knowledge, the experiment is just as likely to exclude either one of the mechanisms that are not true, the probability of the experiment excluding mechanism three given mechanism one is the underlying mechanism is $\Pr(X|A) = \frac{1}{2}$. Similarly, $\Pr(X|B) = \frac{1}{2}$ and $\Pr(X|C) = 0$ (i.e., the experiment will not exclude the correct mechanism). Hence, the probability of mechanism one being true given the results of the experiment is given by Equation (3.12).

$$\begin{aligned}
\Pr(X) &= \Pr(X|A)\Pr(A) + \Pr(X|B)\Pr(B)\\
&= \left(\frac{1}{2}\right)\left(\frac{1}{3}\right) + \left(\frac{1}{2}\right)\left(\frac{1}{3}\right) + (0)\left(\frac{1}{3}\right) = \frac{1}{3}\\
\Pr(A|X) &= \frac{\left(\frac{1}{2}\right)\left(\frac{1}{3}\right)}{\frac{1}{3}} = \frac{1}{2} && (3.12)
\end{aligned}$$

The probability of the ligand binding occurring according to mechanism one changes from $\frac{1}{3}$ before the experiment to $\frac{1}{2}$ after the experiment. It can be seen that this corresponds with common sense; the information from an experiment either increases or decreases confidence in a hypothesis.

Bayes' theorem allows one to relate an observation to a proposition or hypothesis under investigation. For example, statement $A$ could be "The temperature in the beaker is 30$^o$C" and statement $B$ could be the statement, "The temperature *measured* in the beaker is 31$^o$C". Hence, one can relate how much one believes the temperature is 30$^o$C given the temperature is measured to be 31$^o$C.

### 3.2.6 Simple Parameter Estimation

A framework was described in § 3.2.2 for making inferences between different propositions. Once a set of propositions have been written and numerical values assigned to the probabilities of some of the propositions, it is possible to calculate some of the probabilities of the other propositions using Properties 1–4 of Theorem 3.2.1. However, scientists and engineers are not interested in just manipulating probabilities; it is necessary to connect propositions to real world problems. It is demonstrated in Example 3.2.4 how one can formulate propositions to make useful calculations.

**Example 3.2.4.** Consider the situation where a die has $p$ sides and the outcomes of $k$ rolls of the die are recorded. The largest value of the die roll is $i_{max}$. Given the measurements, how many sides does the die have?

Let $A_p$ be the statement that a die has $p$ sides and let $C_{ij}$ be the statement that the outcome of the die roll is $i$ on roll $j$. Let $D_i$ (data) be the outcome of the $i^{th}$ roll of the die. Assuming the die is unbiased, the probability of rolling $i$, given the die has $p$ sides, $\Pr(C_{ij}|A_p)$ is given by Equation (3.13).

$$\Pr(C_{ij}|A_p) = \begin{cases} \frac{1}{p} & : \quad i \leq p \\ 0 & : \quad i > p \end{cases} \tag{3.13}$$

Equation (3.13) is a model of the physical system, familiar to any scientist or engineer. The probability of a particular sequence of $k$ throws is given by Equation (3.14).

$$\Pr(C_{D_1,1} \cdots C_{D_k,k}|A_p) = \prod_{j=1}^{k} \Pr\big(C_{D_j,j}|A_p\big) \tag{3.14}$$

The probability that the die has $p$ sides given a sequence of throws is given by application of Bayes' theorem:

$$\Pr(A_p|C_{D_1,1} \cdots C_{D_k,k}) = \frac{\Pr(C_{D_1,1} \cdots C_{D_k,k}|A_p)\Pr(A_p)}{\Pr(C_{D_1,1} \cdots C_{D_k,k})}. \tag{3.15}$$

It should be stressed that the quantity,

$$\Pr(C_{D_1,1} \cdots C_{D_k,k} | A_p)\,,$$

will evaluate to either zero or $1/p^k$ depending on the data. Immediately, two difficulties arise; one needs to know

$$\Pr(C_{D_1,1} \cdots C_{D_k,k})$$

and $\Pr(A_p)$ to be able to complete the parameter estimation problem. The quantity $\Pr(A_p)$ is often referred to as the *prior*. In this example it seems reasonable to assume that with no information, it is equally likely that a die has two sides as ten. Hence,

$$\Pr(A_p) = \frac{1}{n}, \tag{3.16}$$

where $n$ is the maximum number of sides the die could possibly have. Note: $i_{max}$ is the largest value of a roll that we have seen and need not be necessarily equal to $n$, the number of sides of the die could have. Substituting Equations (3.13), (3.14) and (3.16) into Equation (3.15) yields:

$$\Pr(A_p | C_{D_1,1} \cdots C_{D_k,k}) = \begin{cases} \left(\frac{1}{p}\right)^k \frac{1}{\Pr(C_{D_1,1} \cdots C_{D_k,k})} \frac{1}{n} & : p \geq i_{max} \\ 0 & : p < i_{max} \end{cases} \tag{3.17}$$

The quantity,

$$\Pr(C_{D_1,1} \cdots C_{D_k,k})\,,$$

can be calculated from the additional requirement shown in Equation (3.18), i.e., one of the outcomes must occur and each outcome is mutually exclusive.

$$\sum_{p=1}^{n} \Pr(A_p | C_{D_1,1} \cdots C_{D_k,k}) = 1 \tag{3.18}$$

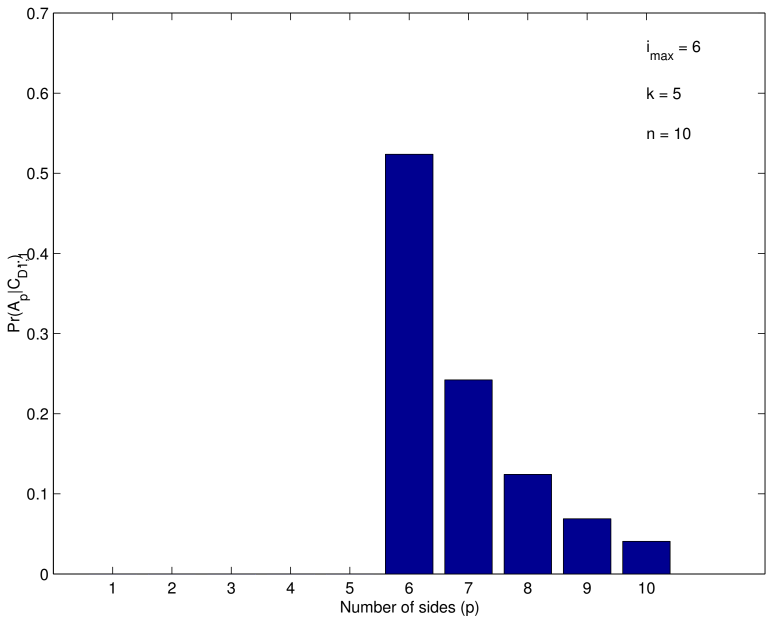


Figure 3-2: Probability density function for Example 3.2.4

Substituting Equation (3.17) into (3.18) yields:

$$\Pr(C_{D_1,1} \cdots C_{D_k,k}) = \frac{1}{n} \sum_{p=i_{max}}^{n} \left(\frac{1}{p}\right)^k . \tag{3.19}$$

Hence, the probability that a die has $p$ sides given a sequence of $k$ rolls of the die, can be made by making the necessary substitutions, as shown in Equation (3.20).

$$\Pr(A_p|C_{D_1,1} \cdots C_{D_k,k}) = \begin{cases} \left(\frac{1}{p}\right)^k \dfrac{1}{\sum_{i=i_{max}}^{n} (1/i)^k} & : p \geq i_{max} \\ 0 & : p < i_{max} \end{cases} \tag{3.20}$$

The probability density function is shown in Figure 3-2. It should be noted that the probability,

$$\Pr(A_p|C_{D_1,1} \cdots C_{D_k,k}),$$

depends on $i_{max}$ which is obtained from the data,

$$D_i, \quad i = 1 : k.$$

In fact, it can be seen that the quantity $i_{max}$ complete summarizes the data for this parameter estimation problem. The probability,

$$\Pr(A_p|C_{D_1,1} \ldots C_{D_k,k}) \,,$$

also depends on the *prior* probability, $\Pr(A_p)$. The question arises what value, $n$ to assign for the maximum possible number of sides of the die. If there is no *a priori* knowledge to determine the value of $n$, it is important that the value of $n$ is not too small since there would be a risk that $p > n$. Provided the die has been rolled more than once, ($k \geq 2$) the series:

$$\sum_{p=i_{max}}^{\infty} \left(\frac{1}{p}\right)^k$$

is convergent. Hence, the simplest solution is to assume that the number of sides, $p$, ranges $1 \leq p < \infty$.

It still remains to answer the original question, "How many sides has the die?". It is impossible to answer this question with absolute *certainty.* Instead, one can state how much one believes the statement that a die has a certain number of sides. It seems reasonable to characterize the die by the most probable statement. For this example, the most probable number of sides of the die is equal to $i_{max}$ or the maximum value in a sequence of rolls.

## 3.3 Relating Probabilities to the Real World

It is important that one can characterize problems numerically, rather than just in terms of propositions. By definition, probability is a function of a proposition; the probability of a number is *meaningless.* In Example 3.2.4 of § 3.2.6, a correspondence was defined between a proposition $A_p$ (a die has $p$ sides) and a number, $p$. In this section two important functions are defined: the cumulative density function (CDF) and the probability density function (PDF). These functions can be used to characterize the probability of certain special propositions, allowing one to relate probabilities to scientific problems. Three theorems are presented in this Section which allow one to derive new PDFs and CDFs from PDFs and CDFs that are already defined. Initially, it may seem that the information contained in these Theorems is purely theoretical and is not of use in problems of inference. However, these Theorems will be used almost constantly later in this thesis.

## 3.3.1 Cumulative Density Functions

For problems where we are interested in a real valued quantity, the continuous cumulative density function (CDF) is defined as:

**Definition 3.3.1.** Let $\hat{x} \in \mathbb{R}$ be the quantity of interest and let $x \in \mathbb{R}$ be some value that $\hat{x}$ could take. Let $A$ be the proposition:

$$A \equiv (\hat{x} \leq x)\,.$$

The continuous cumulative density function (CDF) is defined as:

$$\mathrm{F}_x(x) \equiv \Pr(A)\,.$$

The subscript on $\mathrm{F}_x(x)$ allows one to distinguish between which quantity is being compared with which value. For example, the quantity $\mathrm{F}_x(y)$ should be interpreted as:

$$\mathrm{F}_x(y) \equiv \Pr(\hat{x} \leq y)\,.$$

It is extremely important that a distinction is made between a quantity (temperature, pressure, concentration, etc.) and the value it takes (as demonstrated by Definition 3.3.1). In inference problems, the quantity under investigation is uncertain. Hence, it makes sense to compare the quantity to some fixed value. In this thesis, a variable representing a physical quantity is denoted by circumflex and variables representing possible values will not have a circumflex.

An example continuous CDF is shown in Figure 3-3a. For discrete problems where a quantity can take a countable number of values, the discrete cumulative density function is defined as:

**Definition 3.3.2.** Let $\hat{n} \in \mathbb{N}$ be the quantity of interest and let $n \in \mathbb{N}$ be some value that $\hat{n}$ could take. Let $A$ be the proposition:

$$A \equiv (\hat{n} \leq n)\,.$$

The discrete cumulative density function (CDF) is defined as:

$$\mathrm{F}_n(n) \equiv \Pr(A)\,.$$

A CDF has the properties:

1. It is always true that $\hat{x} < \infty$, hence:

$$\lim_{x\to\infty} \mathrm{F}_x(x) = 1.$$

2. It is never true that $\hat{x} < -\infty$, hence:

$$\lim_{x\to-\infty} \mathrm{F}_x(x) = 0.$$

3. If $x_1 < x_2$, it is more likely that $\hat{x} \leq x_2$ than $\hat{x} \leq x_1$, hence,

$$\mathrm{F}_x(x_1) \leq \mathrm{F}_x(x_2)\,.$$

4. From mutual exclusivity of probability it follows:

$$\Pr(\hat{x} > x) = 1 - \mathrm{F}_x(x)\,.$$

5. From mutual exclusivity of probability it follows:

$$\Pr(x_1 < \hat{x} \leq x_2) = \mathrm{F}_x(x_2) - \mathrm{F}_x(x_1)\,.$$

The discrete CDF is defined on the set of integers as shown in Figure 3-3b.

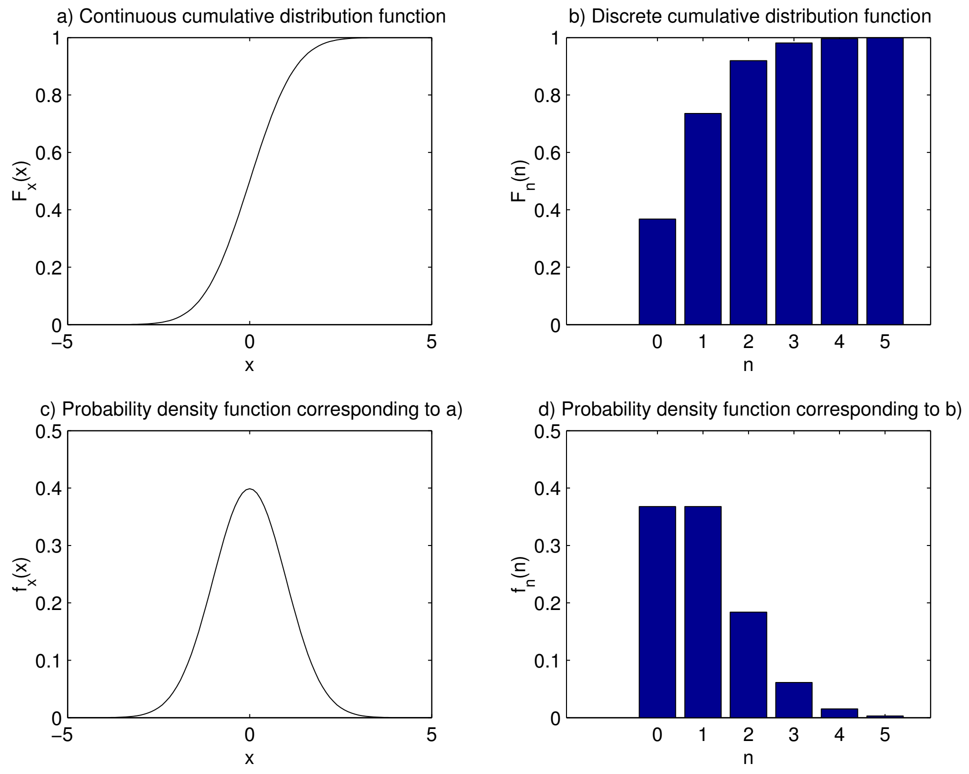


Figure 3-3: Example cumulative density functions and probability density functions

### 3.3.2 Probability Density Functions

**Definition 3.3.3.** The probability density function (PDF) corresponding to a continuous CDF is defined as a nonnegative function $\mathrm{f}_x(x)$:

$$\mathrm{F}_x(x) = \int_{-\infty}^{x} \mathrm{f}_x(t)\ \mathrm{d}t,$$

if such a function exists. An alternative definition (if this quantity is well defined) is

$$\mathrm{f}_x(x) \equiv \lim_{\delta x \to 0} \frac{\Pr(x < \hat{x} \leq x + \delta x)}{\delta x}$$

or equivalently,

$$\mathrm{f}_x(x) \equiv \frac{\mathrm{dF}_x(x)}{\mathrm{d}x}.$$

The continuous PDF has the following properties:

1. A continuous CDF is always a monotonically increasing function, hence:

$$\mathrm{f}_x(x) \geq 0.$$

2. It is always certain that $-\infty < \hat{x} < \infty$:

$$\int_{-\infty}^{\infty} \mathrm{f}_x(\tau)\ \mathrm{d}\tau = 1.$$

3. From the definition of an integral it follows:

$$\mathrm{F}_x(x_2) - \mathrm{F}_x(x_1) = \int_{x_1}^{x_2} \mathrm{f}_x(\tau)\ \mathrm{d}\tau.$$

**Definition 3.3.4.** The PDF corresponding to a discrete CDF is:

$$\mathrm{f}_n(n) \equiv \Pr(\hat{n} = n)\,.$$

The discrete PDF has the following properties:

1. A probability is never less that zero, hence:

$$\mathrm{f}_n(n) \geq 0.$$

2. It is always certain that $-\infty < \hat{n} < \infty$, hence:

$$\sum_{i=-\infty}^{\infty} \mathrm{f}_n(i) = 1.$$

3. From the definition of a PDF it follows:

$$\mathrm{F}_n(n) = \sum_{i=-\infty}^{n} \mathrm{f}_n(i)\,.$$

4. From the definition of a sum it follows (for $n_2 > n_1$):

$$\mathrm{F}_n(n_2) - \mathrm{F}_n(n_1) = \sum_{i=n_1+1}^{n_2} \mathrm{f}_n(i)\,.$$

Example probability density functions are shown in Figure 3-3.

### 3.3.3 Change of Variables

It is often necessary to map one proposition to another. For example, an input (temperature, pressure, etc.) may be measured and the output (flow rate, composition, etc.) is calculated. Scientists and engineers are used to writing models (functions) to describe these mappings:

$$\hat{y} = g(\hat{x})\,, \quad g : \mathbb{R} \rightarrow \mathbb{R}.$$

It is important to note that it is quantities that are mapped from one to another. It does not make sense to write:

$$y = g(x)$$

since $x$ is just some value with which to compare $\hat{x}$ and may (and probably will) not have any relationship to the quantity $\hat{y}$, unless,

$$\Pr(\hat{x} = x) = 1,$$

and,

$$\Pr(\hat{y} = y) = 1,$$

in which case it would seem that use of probability is unwarranted.

Correspondingly, there are many situations where one would like to calculate,

$$\Pr(\text{The quantity, } \hat{y}\text{, equals } y)\,,$$

based on information about $\hat{x}$. Theorem 3.3.1 can be used to derive the probability density function of $\hat{y}$ when the PDF of $\hat{x}$ is discrete.

**Theorem 3.3.1.** *[54] If the probability density function $f_n(n)$ for $\hat{n}$ is discrete, and the quantity $\hat{m}$, is given by $\hat{m} = g(\hat{n})$ then,*

$$f_m(m) = \Pr(\hat{m} = m) = \Pr(g(\hat{n}) = m) = \sum_{n:g(n)=m} f_n(n)\,.$$

**Example 3.3.1.** Suppose fifty male-female pairs of rabbits mate and produce two rabbits per pair. Derive the probability density function for the number of male-female pairs in the second generation. Assume that the PDF for the numbers of male rabbits born is $\mathrm{f}_{n_m}(n_m)$.

Clearly, there are a total of 100 rabbits in the second generation. Let us denote the number of male rabbits in the second generation as $\hat{n}_m$ and the number of couples in second generation as $\hat{n}_c$. Then,

$$\hat{n}_c = \begin{cases} \hat{n}_m & : \hat{n}_m \leq 50 \\ 100 - \hat{n}_m & : \hat{n}_m > 50 \end{cases}.$$

From which it follows that the PDF for the number of couples is:

$$\mathrm{f}_{n_c}(n_c) = \mathrm{f}_{n_m}(n_c) + \mathrm{f}_{n_m}(100 - n_c)\,, \quad n_c = 0, \ldots, 50.$$

If the PDF for $\hat{x}$ is continuous, then the CDF for $\hat{y}$ can be derived from Theorem 3.3.2.

**Theorem 3.3.2.** *[54] If the probability density function $f_x(x)$ for $\hat{x}$ is continuous, and the quantity $\hat{y}$ is given by $\hat{y} = g(\hat{x})$ then,*

$$\begin{aligned} F_y(y) &= \Pr(\hat{y} \leq y) = \Pr(g(\hat{x}) \leq y) \\ &= \int_{x: g(x) \leq y} f_x(x)\ \mathrm{d}x. \end{aligned}$$

**Example 3.3.2.** Calculate the PDF for $\hat{y}$ given it is related to $\hat{x}$ by

$$\hat{y} = \hat{x}^2,$$

and the PDF for $\hat{x}$ is uniform on the interval $(-1, 1)$, i.e., the PDF for $\hat{x}$ is given by:

$$\mathrm{f}_x(x) = \begin{cases} \frac{1}{2} & -1 < x < 1, \\ 0 & \text{otherwise.} \end{cases}.$$

It is clear that,

$$\hat{x} \in (-1, 1) \Rightarrow \hat{y} \in [0, 1)\,.$$

Hence, for $y < 0$, $\mathrm{F}_y(y)$ is zero. (The interval $(-\infty, y)$ does not intersect with $[0, 1)$.) For $y > 1$, $\mathrm{F}_y(y)$ is one, since all of $[0, 1)$ is contained in $(-\infty, y)$. On the interval, $0 \leq y < 1$, the CDF, $\mathrm{F}_y(y)$, is given by:

$$\mathrm{F}_y(y) = \int_{-y^{\frac{1}{2}}}^{y^{\frac{1}{2}}} \mathrm{f}_x(x)\ \mathrm{d}x = y^{\frac{1}{2}}.$$

Since $\mathrm{F}_y(y)$ is differentiable on the interval $0 < y < 1$, the PDF for $\hat{y}$ on the interval

$0 < y < 1$ is given by:

$$\mathrm{f}_y(y) = \frac{\mathrm{dF}_y(y)}{\mathrm{d}y} = \frac{1}{2y^{\frac{1}{2}}}.$$

### 3.3.4 Joint Cumulative Density Functions

It is easy to generalize the CDF to the situation where the probability of a compound statement, $\Pr(AB|C)$, is of interest rather than the probability of just a single statement, $\Pr(A|C)$.

**Definition 3.3.5.** The joint CDF is defined as:

$$\mathrm{F}_{x,y}(x, y) \equiv \Pr((\hat{x} \leq x) \wedge (\hat{y} \leq y)) .$$

It is straightforward to show the joint CDF has the following properties:

1. It is certain that $(\hat{x}, \hat{y}) \in (-\infty, \infty) \times (-\infty, \infty)$, hence:

$$\lim_{\substack{x \to \infty \\ y \to \infty}} \mathrm{F}_{x,y}(x, y) = 1.$$

2. It is never true that either $\hat{x} < -\infty$ or $\hat{y} < -\infty$, hence:

$$\lim_{x \to -\infty} \mathrm{F}_{x,y}(x, y) = 0,$$
$$\lim_{y \to -\infty} \mathrm{F}_{x,y}(x, y) = 0.$$

3. From mutual exclusivity it follows:

$$\begin{aligned}\Pr(x_1 < \hat{x} \leq x_2, y_1 < \hat{y} \leq y_2) &= \mathrm{F}_{x,y}(x_2, y_2) - \mathrm{F}_{x,y}(x_1, y_2) \\ &\quad - \mathrm{F}_{x,y}(x_2, y_1) + \mathrm{F}_{x,y}(x_1, y_1)\end{aligned}$$

4. From the definition of a joint CDF it follows:

$$\begin{aligned}\mathrm{F}_x(x) &= \lim_{y\to\infty} \mathrm{F}_{x,y}(x,y)\,,\\ \mathrm{F}_y(y) &= \lim_{x\to\infty} \mathrm{F}_{x,y}(x,y)\,.\end{aligned}$$

### 3.3.5 Joint Probability Density Functions

**Definition 3.3.6.** The joint PDF is defined if there exists an nonnegative function $\mathrm{f}_{x,y}(x,y)$:

$$\mathrm{F}_{x,y}(x,y) = \int_{-\infty}^{x}\int_{-\infty}^{y} \mathrm{f}_{x,y}(s,t)\ \mathrm{d}s\mathrm{d}t.$$

If the joint CDF is sufficiently differentiable, an alternative definition is

$$\mathrm{f}_{x,y}(x,y) = \frac{\partial^2 \mathrm{F}_{x,y}(x,y)}{\partial x \partial y}.$$

The joint continuous PDF has the following property:

1. From the definition of the joint continuous CDF:

$$\Pr(\{x,y\}\in D) = \iint\limits_{D} \mathrm{f}_{x,y}(x,y)\ \mathrm{d}x\mathrm{d}y.$$

In § 3.2.3, an important corollary was stated (called marginalization). This is a process by which one can calculate the probability that a proposition depends on the occurrence one of many mutually exclusive propositions. The continuous version of marginalization is stated as:

**Theorem 3.3.3.** *[54, 169] The marginal PDF $f_z(z)$ is related to the joint PDF $f_{z,w}(z,w)$ by*

$$f_z(z) = \int_{-\infty}^{\infty} f_{z,w}(z,w)\,\mathrm{d}w.$$

**Example 3.3.3.** The joint PDF for the weight and length of a new-born baby is given by:

$$\mathrm{f}_{l,w}(l,w) = \frac{1}{2\pi\sigma^2}\exp\left(-\frac{2.125l^2 + 3.75lw + 2.125w^2}{2\sigma^2}\right),$$

where $\hat{l}$ is the length of the baby and $\hat{w}$ is the weight of the baby. Derive the marginal PDF for the length of a baby.

The marginal PDF is obtained by direct application of Theorem 3.3.3:

$$\begin{aligned}
\mathrm{f}_l(l) &= \int_{-\infty}^{\infty} \frac{1}{2\pi\sigma^2} \exp\left(-\frac{2.125l^2 + 3.75lw + 2.125w^2}{2\sigma^2}\right) \mathrm{d}w \\
&= \frac{1}{\sigma\sqrt{4.25\pi}} \exp\left(-\frac{l^2}{4.25\sigma^2}\right).
\end{aligned}$$

Often scientists and engineers have a model which relates inputs to outputs. Sometimes the PDF for the inputs is known, and one would like to derive the PDF for the outputs. Theorem 3.3.4 relates the PDF of the outputs to the PDF of the inputs.

**Theorem 3.3.4.** *[54, 169] To find $f_{z,w}(z,w)$, the joint probability density for $\{\hat{z},\hat{w}\}$, where $\hat{z} = g(\hat{x},\hat{y})$ and $\hat{w} = h(\hat{x},\hat{y})$, solve the system:*

$$\begin{aligned}
g(x,y) &= z \\
h(x,y) &= w
\end{aligned}$$

*denoting the roots, $\{x_n, y_n\}$. Then, if the relevant Jacobians matrices are nonsingular,*

$$f_{z,w}(z,w) = \frac{f_{x,y}(x_1,y_1)}{|J(x_1,y_1)|} + \ldots + \frac{f_{x,y}(x_n,y_n)}{|J(x_n,y_n)|},$$

*where the Jacobian matrix, $J(x,y)$, is defined as*

$$J(x,y) = \begin{bmatrix} \frac{\partial z}{\partial x} & \frac{\partial z}{\partial y} \\ \frac{\partial w}{\partial x} & \frac{\partial w}{\partial y} \end{bmatrix}$$

*and $|\cdot|$ denotes the absolute value of the determinant of a matrix.*

A common example is variable rescaling (for example: a temperature is measured in Fahrenheit but is required in Celsius):

**Example 3.3.4.** The variable, $\hat{z}$, is related to the variable $\hat{x}$ by:

$$\hat{z} = \frac{\hat{x} - \mu}{\sigma}.$$

If the PDF for $\hat{z}$ is $\mathrm{f}_z(z)$, what is the PDF for $\hat{x}$?

Direct application of Theorem 3.3.4 yields:

$$\begin{aligned} \mathrm{f}_x(x) &= \frac{\mathrm{f}_z(z)}{\left|\frac{\mathrm{d}x}{\mathrm{d}z}\right|} \\ &= \frac{1}{|\sigma|}\mathrm{f}_z\left(\frac{x-\mu}{\sigma}\right). \end{aligned}$$

Frequently, a scientist or engineer faces the situation where the number of outputs of a system is less than the number of inputs (for example: the equilibrium concentration of a product may depend on the initial concentrations of two reactants and the temperature of the system). A convenient trick allows one to determine the PDF of the outputs from the PDF of the inputs (as shown in Example 3.3.5).

**Example 3.3.5.** Calculate $\mathrm{f}_z(z)$, where $\hat{z}$ is defined by $\hat{z} = \hat{x} + \hat{y}$ and the probability density function, $\mathrm{f}_{x,y}(x, y)$, is known.

The desired PDF can be obtained by the introduction of an additional variable $\hat{w} = \hat{y}$ to make the resulting system square. By Theorem 3.3.4 the PDF for the square system, $\mathrm{f}_{z,w}(z, w)$, is

$$\mathrm{f}_{z,w}(z, w) = \frac{\mathrm{f}_{x,y}(z - w, w)}{1},$$

since,

$$|J| = 1.$$

By Theorem 3.3.3, $\mathrm{f}_z(z)$ is given by:

$$\mathrm{f}_z(z) = \int_{-\infty}^{\infty} \mathrm{f}_{x,y}(z - w, w)\,\mathrm{d}w.$$

**Theorem 3.3.5.** *(Page 146 of [85]). Let $\hat{y}$ be the sum of $n$ independent variables $\hat{x}_i$:*

$$\hat{y} = \sum_{i=1}^{n} \hat{x}_i,$$

*and $f_x(x_i)$ be the probability density function for $\hat{x}_i$. The probability density function for $\hat{y}$ is given by:*

$$f_y(y) = f_x^{(n)}(y) \tag{3.21}$$

*where $f_x^{(n)}(y)$ denotes the n-fold convolution:*

$$f_x^{(n)} = f_x^{(n-1)} * f_x,$$

*and*

$$g * f = \int_{-\infty}^{\infty} g(y-x)\, f(x)\, \mathrm{d}x.$$

*Proof.* Define $\hat{y}_i$ as

$$\hat{y}_i = \hat{x}_i + \hat{y}_{i-1}$$

and the PDF for $\hat{y}_i$ as $\mathrm{g}_i(y_i)$. Introducing an additional variable, $\hat{z}_i = \hat{x}_i$, the joint PDF for $\{\hat{y}_i, \hat{z}_i\}$ is given by (Theorem 3.3.4):

$$\mathrm{h}(y_i, z_i) = \mathrm{g}_{i-1}(y_i - z_i)\, \mathrm{f}(y_i)\,.$$

Marginalization of the joint density yields the PDF for $\hat{y}_i$:

$$\mathrm{g}_i(y_i) = \int_{-\infty}^{\infty} \mathrm{h}(y_i, z_i)\ \mathrm{d}z_i. \tag{3.22}$$

Repeated application of Equation 3.22 yields the result in Equation (3.21). □

### 3.3.6 Conditional Density Functions

It is a common situation in science to know *a priori* an accurate model of the system of interest; the goal of experimentation is to determine some physical parameters. The task of inferring model parameters from experimental data is called parameter estimation. It is important to have a form of Bayes' Theorem that is suitable for this task. To obtain such a form requires the conditional CDF and conditional PDF to be defined:

**Definition 3.3.7.** The conditional CDF for $\hat{x} \in \mathbb{R}$ is given by

$$\mathrm{F}_x(x|B) = \Pr(\hat{x} \leq x|B)$$

where $B$ is some proposition. Likewise, the joint conditional CDF for $\{\hat{x}, \hat{y}\} \in \mathbb{R}^2$ is given by

$$\mathrm{F}_{x,y}(x, y|B) = \Pr((\hat{x} \leq x) \wedge (\hat{y} \leq y)\,|B)$$

where $B$ is some proposition. The corresponding PDFs are defined as the derivatives of the CDFs:

$$\begin{aligned} \mathrm{f}_x(x|B) &= \frac{\mathrm{dF}_x(x|B)}{\mathrm{d}x} \\ \mathrm{f}_{x,y}(x, y|B) &= \frac{\partial^2 \mathrm{F}_{x,y}(x, y|B)}{\partial x \partial y}. \end{aligned}$$

A difficulty arises if the proposition $B$ is defined as the real-valued quantity, $\hat{x}$, equaling a specified value $x$, $(\hat{x} = x)$, since for many situations the probability of the proposition is zero. Consequently, a conditional PDF that depends on two real-valued variables $\hat{x}$, and $\hat{y}$, $\mathrm{f}_y(y|\hat{x} = x)$, is defined as the limit:

$$\mathrm{f}_y(y|\hat{x} = x) \lim_{\delta x \to 0} \mathrm{f}_y(y|x < \hat{x} \leq x + \delta x)\,.$$

It is demonstrated in Example 3.3.6 how a typical conditional PDF can be derived.

**Example 3.3.6.** Suppose $n$ measurements are made where the PDF for the output, $\hat{y}$, is $\mathrm{f}_y(y|I)$, and $I$ is any additional information. Derive the PDF for the $k^{th}$ largest measurement, $\mathrm{f}_{y_k}(y_k|n, k, I)$.

To derive the PDF for the $k^{th}$ largest measurement, $\mathrm{f}_{y_k}(y_k|n, k, I)$ it is necessary to know the probability that the $k^{th}$ largest measurement, $\hat{y}_k$ lies in the range $y_k <$

$\hat{y}_k \leq y_k + \delta y_k$. An equivalent statement to: $\hat{y}_k$ lies in the range $y_k < \hat{y}_k \leq y_k + \delta y_k$ is there are $k-1$ measurements less than $y_k$, and there is one measurement between $y_k$ and $y_k + \delta y_k$, and there are $n-k$ measurements more than $y_k + \delta y_k$. Hence, there are three possibilities for single measurement: it is less than $y_k$, it is between $y_k$ and $y_k + \delta y_k$, or it is more than $y_k + \delta y_k$. The probabilities corresponding to the different outcomes are:

$$\begin{aligned} p_1 &= \mathrm{F}_y(y_k|I)\,, \\ p_2 &= \int_{y_k}^{y_k+\delta y_k} \mathrm{f}_y(t|I)\;\mathrm{d}t, \end{aligned}$$

and,

$$p_3 = 1 - \mathrm{F}_y(y_k + \delta y_k|I)\,,$$

respectively. Defining the following statements:

$A$: $k-1$ of the $n$ measurements are less than $y_k$,

$B$: 1 of the $n$ measurements is between $y_k$ and $y_k + \delta y_k$, and,

$C$: $n-k$ of the $n$ measurements are more than $y_k + \delta y_k$,

the probability $\Pr(ABC|n,k,I)$ is given by the multinomial density:

$$\Pr(ABC|n,k,I) = \frac{n!}{(k-1)!1!\,(n-k)!} p_1^{k-1} p_2 p_3^{n-k}.$$

By considering the limit:

$$\mathrm{f}_{y_k}(y_k|n,k,I) = \lim_{\delta y_k \to 0} \frac{\Pr(y_k < \hat{y}_k \leq y_k + \delta y_k|n,k,I)}{\delta y_k} = \lim_{\delta y_k \to 0} \frac{\Pr(ABC|n,k,I)}{\delta y_k},$$

it follows:

$$\mathrm{f}_{y_k}(y_k|n,k,I) = \frac{n!}{(n-k)!\,(k-1)!} \left(\mathrm{F}_y(y_k|I)\right)^{k-1} \left(1 - \mathrm{F}_y(y_k|I)\right)^{n-k} \mathrm{f}_y(y_k|I)\,. \tag{3.23}$$

## 3.4 Risk, Reward, and Benefit

A probability alone is insufficient to make an informed decision. To make such a decision it is also necessary to take into account the consequences of making such a decision. Expectation is a closely related concept to probability [123]. It is assumed that there is some value (reward) corresponding to the truth of a proposition, conditional on some other information. A simple example of expectation might be the game:

A: (Reward) I win \$1 if the result of a coin toss is heads,

B: (Proposition) The result of the coin toss is heads, and,

C: (Conditional Information) the coin is fair.

The expectation of reward is defined as

$$\mathrm{E}(A, B|C) = A\,\mathrm{Pr}(B|C)\,,$$

hence the expected reward of the game is 50 cents (\$1 $\times$ 0.5). The expectation is a function of the reward, the proposition, and conditional information, i.e., just like a probability, an expectation is always dependent on conditional information.

Despite an unambiguous mathematical description of expectation, the interpretation of expectation can be troublesome as demonstrated in Example 3.4.1, .

**Example 3.4.1.** (Described on Page 31 of [123].) The following game is called the Petersburg Problem. A coin is repeatedly tossed. \$1 is awarded if a heads is thrown on the first toss and \$0 is awarded if the result is tails. The reward is doubled on each successive throw of the coin. What is the value of the game?

The expected value of reward from the game is:

$$1.\frac{1}{2} + 2.\frac{1}{4} + 4.\frac{1}{8} + \cdots = \infty.$$

However, it is doubtful that there would be many people prepared to pay such a price. The difficulty with the interpretation of expectation is that the value of reward depends on how much someone already has. For example, it might be quite a catastrophe if one has \$100 and a \$500 bike is stolen. However, if the person has \$100,000 and a \$500 bike is stolen the consequences are most likely far less important.

### 3.4.1 Expectation

It is still extremely useful to use the concept of expectation for problems of inference, despite the caveat that expectation must be interpreted carefully. The expectation of a quantity where knowledge of the value of the quantity is described by a PDF is given by the following definition:

**Definition 3.4.1.** The expected value of a real-valued quantity $\hat{x}$ is defined as

$$\mathrm{E}_x(\hat{x}) = \int_{-\infty}^{\infty} x\,\mathrm{f}_x(x)\;\mathrm{d}x, \tag{3.24}$$

and is defined as

$$\mathrm{E}_n(\hat{n}) = \sum_i i\mathrm{f}_n(i)\,, \tag{3.25}$$

for a discrete-valued quantity.

Often the expected value of a function, $\hat{y} = g(\hat{x})$, is of interest. An expression for $\mathrm{E}_x(\hat{y})$ is provided by Theorem 3.4.1.

**Theorem 3.4.1.** *[54, 169] The expected value of $\hat{y} = g(\hat{x})$ is*

$$\mathrm{E}_x(g(\hat{x})) = \int_{-\infty}^{\infty} g(x)\; f_x(x)\;\mathrm{d}x, \tag{3.26}$$

*if the PDF for $\hat{x}$ is a continuous PDF, and,*

$$\mathrm{E}_n(g(\hat{n})) = \sum_i g(i)\, f_n(i)\,, \tag{3.27}$$

*if the PDF for $\hat{n}$ is a discrete PDF. For problems with two variables, where $\hat{z} = g(\hat{x},\hat{y})$*

*and $\hat{z}$, $\hat{x}$, $\hat{y}$ are real variables, the variance of $\hat{z}$ is given by:*

$$\mathrm{E}_{x,y}(g(\hat{x},\hat{y})) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(x,y)\, f_{x,y}(x,y)\ \mathrm{d}x\mathrm{d}y. \tag{3.28}$$

*The expectation operator has the following easily verified linearity properties:*

1. $\mathrm{E}_x(a\hat{x}) = a\mathrm{E}_x(\hat{x})$

2. $\mathrm{E}_{x,y}(a\hat{x} + b\hat{y}) = a\mathrm{E}_{x,y}(\hat{x}) + b\mathrm{E}_{x,y}(\hat{y})$

*where $a$ and $b$ are real-valued constants.*

## 3.4.2 Variance and Covariance

The variance of a quantity, $\hat{x}$, is defined in terms of the expectation of a function and can be used to characterize a PDF for $\hat{x}$. The variance of $\hat{x}$ is defined as follows:

**Definition 3.4.2.** The variance of a continuous variable $\hat{x}$ is defined as

$$\mathrm{Var}(\hat{x}) = \int_{-\infty}^{\infty} (x-\eta)^2\, \mathrm{f}_x(x)\,\mathrm{d}x \tag{3.29}$$

and the variance of a discrete variable is defined as

$$\mathrm{Var}(\hat{n}) = \sum_i (i-\eta)^2\, \mathrm{f}_n(i) \tag{3.30}$$

where $\eta = \mathrm{E}_x(\hat{x})$.

Applying Theorem 3.4.1 to the definition of variance, yields:

$$\mathrm{Var}(\hat{x}) = \mathrm{E}_x\big((\hat{x}-\eta)^2\big)\,.$$

From the linearity of the expectation operator, the variance of $\hat{x}$ can also be expressed as:

$$\mathrm{Var}(\hat{x}) = \mathrm{E}_x\big(\hat{x}^2\big) - (\mathrm{E}_x(\hat{x}))^2\,. \tag{3.31}$$

For a joint PDF the covariance is an important measure of correlation between two variables.

**Definition 3.4.3.** The covariance of two variables, $\hat{x}$ and $\hat{y}$, is defined as:

$$\mathrm{Cov}\left(\hat{x}\hat{y}\right) = \mathrm{E}_{x,y}((\hat{x} - \eta_x)\left(\hat{y} - \eta_y\right)) \tag{3.32}$$

where $\eta_x$ and $\eta_y$ are defined as:

$$\eta_x = \mathrm{E}_{x,y}(\hat{x})$$

and,

$$\eta_y = \mathrm{E}_{x,y}(\hat{y})\,.$$

From the properties of the expectation operator:

$$\begin{aligned} \mathrm{Cov}\left(\hat{x}\hat{y}\right) &= \mathrm{E}(\hat{x}\hat{y} - \eta_x\hat{y} - \eta_y\hat{x} + \eta_x\eta_y) && (3.33)\\ &= \mathrm{E}(\hat{x}\hat{y}) - \mathrm{E}(\hat{x})\,\mathrm{E}(\hat{y})\,. && (3.34)\end{aligned}$$

It is possible to derive the following expressions and properties:

1. $\mathrm{Var}(a\hat{x}) = a^2\mathrm{Var}(\hat{x})$, where $a$ is a real-valued constant.

2. If $\hat{x}$ and $\hat{y}$ are independent (i.e., $\mathrm{f}_{x,y}(x,y) = \mathrm{f}_x(x)\,\mathrm{f}_y(y)$), they are uncorrelated: $\mathrm{Cov}\left(\hat{x},\hat{y}\right) = 0$.

3. If $\hat{x}$ and $\hat{y}$ are uncorrelated, $\mathrm{Var}(\hat{x}+\hat{y}) = \mathrm{Var}(\hat{x}) + \mathrm{Var}(\hat{y})$.

How to calculate the expected value and variance of a quantity is demonstrated in Example 3.4.2.

**Example 3.4.2.** Calculate the expected value and variance of a Log-Normal density:

$$\mathrm{f}_x(x) = \frac{1}{x\sigma\sqrt{2\pi}}\exp\left(-\frac{\left(\log x - \mu\right)^2}{2\sigma^2}\right).$$

By definition, the expected value of the density is given by:

$$\begin{aligned}\mathrm{E}(\hat{x}) &= \int_0^\infty x\mathrm{f}_x(x)\ \mathrm{d}x \\ &= \frac{1}{\sqrt{2\pi}}\int_0^\infty \frac{1}{x\sigma}x\exp\left(-\frac{(\log x-\mu)^2}{2\sigma^2}\right)\ \mathrm{d}x.\end{aligned}$$

To evaluate the integral it is necessary to transform variables. Defining $t$ as

$$t \equiv \frac{\log x - \mu}{\sigma},$$

the expectation can be evaluated:

$$\begin{aligned}\mathrm{E}(\hat{x}) &= \frac{1}{\sqrt{2\pi}}\int_{-\infty}^\infty \exp\left(-\frac{t^2}{2}+\sigma t+\mu\right)\ \mathrm{d}t \\ &= \frac{1}{\sqrt{2\pi}}\exp\left(\mu+\frac{\sigma^2}{2}\right)\int_{-\infty}^\infty \exp\left(-\frac{(t-\sigma)^2}{2}\right)\ \mathrm{d}t \\ &= \exp\left(\mu+\frac{\sigma^2}{2}\right).\end{aligned}$$

The variance of the density is defined as:

$$\mathrm{E}\big((\hat{x}-\eta_x)^2\big) = \frac{1}{\sqrt{2\pi}}\int_0^\infty \left(x-\exp\left(\mu+\frac{\sigma^2}{2}\right)\right)^2\frac{1}{\sigma x}\exp\left(-\frac{(\log x-\mu)^2}{2\sigma^2}\right)\ \mathrm{d}x.$$

To evaluate the integral it is necessary to transform variables. Defining $t$ as

$$t \equiv \frac{\log x - \mu}{\sigma},$$

the integral can be rewritten as

$$\begin{aligned}\mathrm{Var}(\hat{x}) = &\ \frac{1}{\sqrt{2\pi}}\int_{-\infty}^\infty \left(\exp(2\sigma t+2\mu)-2\exp\left(\sigma t+2\mu+\frac{\sigma^2}{2}\right)+\exp\big(2\mu+\sigma^2\big)\right) \\ &\times\exp\left(-\frac{t^2}{2}\right)\ \mathrm{d}t,\end{aligned}$$

which can be rearranged to

$$\begin{aligned}\mathrm{Var}(\hat{x}) \;&=\; \exp\bigl(2\mu+\sigma^2\bigr)+\frac{\exp(2\mu+2\sigma^2)}{\sqrt{2\pi}}\int_{-\infty}^{\infty}\exp\left(-\frac{(t-2\sigma)^2}{2}\right)\,\mathrm{d}t\\ &\quad -2\frac{\exp(2\mu+\sigma^2)}{\sqrt{2\pi}}\int_{-\infty}^{\infty}\exp\left(-\frac{(t-\sigma)^2}{2}\right)\,\mathrm{d}t.\end{aligned}$$

Evaluating the integrals yields:

$$\mathrm{Var}(\hat{x}) = \left(\exp\left(\mu+\frac{\sigma}{2}\right)\right)^2\bigl(\exp\bigl(\sigma^2\bigr)-1\bigr)\,.$$

There are two widely used systems of inference: inference by statistics (sometimes called "frequentist approach" or "orthodox statistics") and inference by Bayes' Theorem. Inference by statistics makes use of expectation and variance to determine parameter values.

## 3.5 Systems of Parameter Inference

Two different systems of parameter inference are described in this section: inference by Bayes' Theorem, and inference by statistics (sometimes known as "the frequentist approach"). Despite the widespread adoption of the term "frequentist", we will not adopt this term as it is extremely misleading. The objective of both systems is the same; to infer the value of a parameter, $\hat{x} \in \mathbb{R}$, given data, $\hat{\mathbf{y}} \in \mathbb{R}^{n_y}$, equal to the values, $\mathbf{y} \in \mathbb{R}^{n_y}$. Historically, these two systems of inference have been seen as diametrically opposed. However, the differences between the systems of inference has been reconciled with modern theory. It is perfectly consistent to use inference by statistics and have the "Bayesian" view that a probability is a measure of belief in a proposition.

Overall, the method of inference by Bayes' theorem is preferred for two reasons:

1. the theory of Bayesian probability can be extended to more than just parameter estimation, and,

2. the method is straightforward to apply algorithmically.

In contrast, it is nearly impossible to extend the theory of inference by statistics to problems such as model selection. Furthermore, inference by statistics requires the selection of a statistic (function of the data). However, it is not straightforward to determine the correct statistic for anything other than trivial parameter estimation problems. Despite the drawbacks of inference by statistics, both systems of inference are described in § 3.5.1–3.5.2 for completeness.

### 3.5.1 Inference by Bayes' Theorem

The foundations of Bayesian inference for parameter estimation problems are described in this section. For the purposes of exposition, it is assumed that the goal is to infer the value of a single state, $\hat{x}$, given a set of independent measurements, $\mathbf{y} \in \mathbb{R}^{n_y}$, of an output, $\hat{\mathbf{y}} \in \mathbb{R}^{n_y}$. Knowledge about the value $x$ of the state, $\hat{x}$, is summarized by the conditional PDF, $\mathrm{f}_x(x|\mathbf{y})$. This conditional PDF can be obtained by a straightforward extension of Theorem 3.5.1.

**Theorem 3.5.1.** *Defining a conditional PDF according to Definition 3.3.7, application of Theorem 3.2.1 yields the following commonly used forms of Bayes' Theorem:*

$$f_{x,y}(x,y) = f_x(x|\hat{y} = y)\,\pi_y(y)\,, \tag{3.35}$$

*and,*

$$f_x(x|\hat{y} = y)\,\pi_y(y) = f_y(y|\hat{x} = x)\,\pi_x(x)\,, \tag{3.36}$$

*where $\hat{x}$, $x$, $\hat{y}$, and, $y$ are real values quantities, and $\pi_x(x)$ and $\pi_y(y)$ are the unconditional (marginal) PDFs for $\hat{x}$ and $\hat{y}$, respectively.*

*Proof.* From the definition of the conditional CDF function and application of Bayes' theorem (Theorem 3.2.1):

$$\mathrm{F}_y(y|x < \hat{x} \leq x + \delta x) \quad = \quad \frac{\Pr((\hat{y} \leq y) \wedge (x < \hat{x} \leq x + \delta x))}{\Pr(x < \hat{x} \leq x + \delta x)} \tag{3.37}$$

$$= \frac{\mathrm{F}_{x,y}(x+\delta x, y) - \mathrm{F}_{x,y}(x,y)}{\mathrm{F}_x(x+\delta x) - \mathrm{F}_x(x)}. \tag{3.38}$$

Since by definition,

$$\mathrm{F}_{x,y}(x,y) \equiv \int_{-\infty}^{x}\int_{-\infty}^{y} \mathrm{f}_{x,y}(\alpha,\beta)\ \mathrm{d}\alpha\mathrm{d}\beta$$

then,

$$\frac{\partial \mathrm{F}_{x,y}(x,y)}{\partial y} = \int_{-\infty}^{x} \mathrm{f}_{x,y}(\alpha,y)\ \mathrm{d}\alpha$$

which on differentiation of Equation (3.38) yields,

$$\mathrm{f}_y(y|x < \hat{x} \le x+\delta x) = \frac{\displaystyle\int_x^{x+\delta x} \mathrm{f}_{x,y}(\alpha,y)\,\mathrm{d}\alpha}{\displaystyle\int_x^{x+\delta x} \pi_x(\alpha)\,\mathrm{d}\alpha},$$

where $\pi_x(x)$ is the marginal PDF for $\hat{x}$. Examining the limit as $\delta x \to 0$ yields:

$$\begin{aligned}
\mathrm{f}_y(y|\hat{x}=x) &= \lim_{\delta x\to 0} \mathrm{f}_y(y|x<\hat{x}\le x+\delta x) \\
&= \lim_{\delta x\to 0} \frac{\displaystyle\int_x^{x+\delta x} \mathrm{f}_{x,y}(\alpha,y)\,\mathrm{d}\alpha}{\displaystyle\int_x^{x+\delta x} \pi_x(\alpha)\,\mathrm{d}\alpha}.
\end{aligned}$$

Applying L'Hôpital's rule [150]:

$$\begin{aligned}
\mathrm{f}_y(y|\hat{x}=x) &= \lim_{\delta x\to 0} \frac{\dfrac{\mathrm{d}}{\mathrm{d}\,(\delta x)}\displaystyle\int_x^{x+\delta x} \mathrm{f}_{x,y}(\alpha,y)\,\mathrm{d}\alpha}{\dfrac{\mathrm{d}}{\mathrm{d}\,(\delta x)}\displaystyle\int_x^{x+\delta x} \pi_x(\alpha)\,\mathrm{d}\alpha} \\
&= \frac{\mathrm{f}_{x,y}(x,y)}{\pi_x(x)}.
\end{aligned}$$

□

Another form of Bayes' Theorem can be derived by application of Theorem 3.3.3:

$$\mathrm{f}_x(x|\hat{y}=y) = \frac{\mathrm{f}_y(y|\hat{x}=x)\,\pi_x(x)}{\displaystyle\int_{-\infty}^{\infty} \mathrm{f}_y(y|\hat{x}=x)\,\pi_x(x)\,\mathrm{d}x}, \tag{3.39}$$

since,

$$\begin{aligned}\pi_y(y) &= \int_{-\infty}^{\infty} \mathrm{f}_{x,y}(x,y)\,\mathrm{d}x \\ &= \int_{-\infty}^{\infty} \mathrm{f}_y(y|\hat{x}=x)\,\pi_x(x)\,\mathrm{d}x.\end{aligned}$$

It is clear that the denominator in Equation (3.39),

$$\int_{-\infty}^{\infty} \mathrm{f}_y(y|\hat{x}=x)\,\pi_x(x)\,\mathrm{d}x,$$

is a function that does not depend on $x$. Hence, the rule in Equation (3.39) is often abbreviated to

$$\mathrm{f}_x(x|\hat{y}=y) \propto \mathrm{f}_y(y|\hat{x}=x)\,\pi_x(x)\,. \tag{3.40}$$

If more than one independent measurement of the output is made, the posterior PDF, $\mathrm{f}_x(x|\hat{\mathbf{y}}=\mathbf{y})$, can be derived by repeated application of Bayes' Theorem (Theorem 3.5.1):

$$\mathrm{f}_x(x|\hat{y}_n=y_n,\ldots,\hat{y}_1=y_1) \propto \mathrm{f}_y(y_n|\hat{x}=x)\,\mathrm{f}_x(x|\hat{y}_{n-1}=y_{n-1},\ldots,\hat{y}_1=y_1)\,. \tag{3.41}$$

Hence, the posterior PDF is updated by a factor $\mathrm{f}_y(y_i|\hat{x}=x)$ for each measurement $y_i$. Equation (3.41) can be interpreted as a rule of incremental learning. If measurements of the output are independent, the posterior PDF can also be written:

$$\mathrm{f}_x(x|\hat{\mathbf{y}}=\mathbf{y}) \propto \mathrm{f}_{\mathbf{y}}(\mathbf{y}|\hat{x}=x)\,\pi_x(x)\,, \tag{3.42}$$

where the joint likelihood function, $\mathrm{f}_{\mathbf{y}}(\mathbf{y}|\hat{x}=x)$, can be expressed as

$$\mathrm{f}_{\mathbf{y}}(\mathbf{y}|\hat{x}=x) \equiv \prod_{i=1}^{n_y} \mathrm{f}_y(y_i|\hat{x}=x)\,.$$

According to the notation of [123], the result in Equation (3.36) of Theorem 3.5.1 is called the rule of inverse probability. It is the fundamental theorem by which inferences can be made. In the simplest case, $\hat{x}$ is some quantity to be estimated (for example: temperature, concentration, pressure) and $y$ is a measurement of the output, $\hat{y}$. The PDF $\mathrm{f}_x(x|\hat{y}=y)$ summarizes the probability that the state, $\hat{x}$, will take a particular value, $x$, given the output, $\hat{y}$, is measured to be $y$; i.e., knowledge about $\hat{x}$ is summarized by the PDF, $\mathrm{f}_x(x|\hat{y}=y)$.

The quantity, $\mathrm{f}_y(y|\hat{x}=x)$ characterizes the probability of making a measurement $\hat{y}$ equal to a value $y$. According to standard statistical notation, the function $\mathrm{f}_y(y|\hat{x}=x)$ is called the *likelihood* function. In the engineering literature, $\mathrm{f}_y(y|\hat{x}=x)$ is called a process model since it maps the state of a system to a measurable output. It should be stressed that correct selection of $\mathrm{f}_y(y|\hat{x}=x)$ is made by the "art of modeling". There are no axioms from which such a model can be deduced. Accordingly, selection of the function depends on prior experience or knowledge. Some authors make this dependence explicit by denoting the likelihood function as $\mathrm{f}_y(y|\hat{x}=x, I)$, where $I$ is the information used to select the likelihood function. There are many merits to this notation, however, as a shorthand the form $\mathrm{f}_y(y|\hat{x}=x)$ is preferred.

The function $\pi_x(x)$ is referred to as the *prior* PDF. This function characterizes additional information about the quantity of interest, $\hat{x}$, that is not included in the likelihood function. The term prior is unfortunate and misleading: it does not imply any chronology to the order in which information is obtained. Occasionally, the situation may arise where there is little or no additional information about the value of the quantity of interest, in which case it is necessary to assign a prior PDF that appropriately expresses ignorance. The assignment of prior probabilities is discussed in § 3.7. The assignment of prior probabilities has been a source of controversy over the years, leading some to call the Bayesian system of inference subjective. This controversy should not be confused with the appropriateness of Bayesian/Plausible reasoning; the desiderata in Definition 3.2.2 are reasonable and the properties described in Theorem 3.2.1 are derived directly from the desiderata. There is a distinction between whether the rules of inference are objective/fair and whether assignments made to probabilities realistically represent the system of interest; it is quite possible to do Bayesian modeling badly but this does not mean that the rules of inference are incorrect. Likewise, no scientist would doubt that Newton's laws of motion cannot be used to model very high speed mechanics. However, this does not mean that the rules of calculus are incorrect.

The complete framework for inference using Bayes' Theorem has now been described in § 3.2–3.5. However, the assignment of prior PDFs and likelihood functions has not yet been discussed. This material is covered in § 3.6–3.7.

### 3.5.2 Inference by Statistics

An alternative but complementary system of inference is based upon the notion of statistics. Due to historical reasons, this approach is often described as "frequentist" or "orthodox" in the literature. The label "frequentist" refers to the interpretation of probability as the frequency with which an event occurs in the limit of many experiments. This view is not inconsistent with the Bayesian view of probability as a measure of ones belief in the probability of a proposition. Quantitative correspondence of a probability with a frequency (if such an interpretation exists) is guaranteed by Desiderata 6 of Definition 3.2.2 [121]. Consequently, the description of inference by statistics as "frequentist" is an over-simplification.

A statistic is an arbitrary function of the data. Typically, such a function maps $\mathbb{R}^{n_y} \to \mathbb{R}$. For example, a common statistic is the sample mean, $\overline{y}$, defined as:

$$\overline{y} = \frac{1}{n_y} \sum_{i=1}^{n_y} y_i.$$

The goal of this system of inference is to define a statistic in such a way that meaningful conclusions can be drawn about a parameter of interest. Traditionally, the goal has been to show that in the limit of many experiments, the value of the statistic converges to a specific value of interest (for example: the value of the state of the system). Furthermore, the rate of convergence is also characterized. Several terms are used to describe a statistic:

**Definition 3.5.1.** The following terms are useful when describing a statistic:

1. A statistic is an *unbiased* estimate if the expected value of a statistic equals the value of the parameter of interest.

2. A statistic is a *minimum variance* estimate if no other function of the data can be found with a smaller variance.

3. A set of statistics, $t_1, t_2, \ldots,$ that completely summarizes knowledge about the state is *sufficient.* By definition, for a set of sufficient statistics:

$$\mathrm{f}_x\big(x|\hat{t}_1 = t_1, \hat{t}_2 = t_2, \ldots, I\big) = \mathrm{f}_x(x|\hat{\mathbf{y}} = \mathbf{y}, \ldots, I)\,.$$

However, examining the asymptotic convergence of a statistic does not describe the behavior of a statistic based on a small number of experiments. A preferable analysis examines the conditional PDF for the statistic, $\mathrm{f}_{\overline{y}}(\overline{y}|I)$. The conditional PDF can be derived from the likelihood function $\mathrm{f}_y(y|I)$ since the statistic, $\overline{y}(\mathbf{y})$, is a function of the measurements $\mathbf{y}$. $I$ is any information that is cogent to the value of the output (for example: the value of the state of the system).

**Example 3.5.1.** The PDF for the measurement of the output of a system is given by the likelihood function,

$$\mathrm{f}_y(y_i|\hat{x} = x, \hat{\sigma} = \sigma) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(y_i - x)^2}{2\sigma^2}\right).$$

A set of independent measurements, $\mathbf{y} \in \mathbb{R}^{n_y}$ are made of the system. Derive the conditional PDF for the sample mean, $\hat{\overline{y}}$, and the sample median $\hat{y}_{1/2}$. Assume $n_y$ is odd.

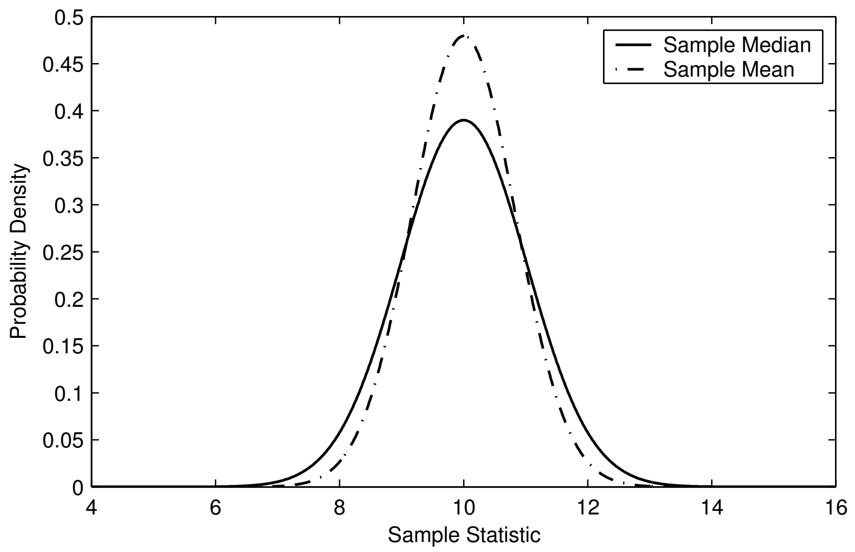


Figure 3-4: PDFs for the sample mean and median ($n = 13$, $\hat{\sigma} = 3$, $\hat{x} = 10$)

The PDF for the sample mean, $\mathrm{f}_{\overline{y}}(\overline{y}|\hat{x} = x, \hat{\sigma} = \sigma)$, is:

$$\mathrm{f}_{\overline{y}}(\overline{y}|\hat{x} = x, \hat{\sigma} = \sigma) = \frac{1}{(\sigma/\sqrt{n})\sqrt{2\pi}} \exp\left(-\frac{(\overline{y} - x)^2}{2\left(\sigma/\sqrt{n}\right)^2}\right),$$

and can be derived by direct application of Theorem 3.3.5. The PDF for the sample median was derived in Example 3.3.6 and is:

$$\mathrm{f}_{y_{1/2}}\big(y_{1/2}|\hat{n}_y = n_y, \hat{x} = x, \hat{\sigma} = \sigma\big) = \\ \frac{n_y!}{\left(\frac{n_y+1}{2}\right)!\left(\frac{n_y-3}{2}\right)!}\left(\mathrm{F}_y\big(y_{1/2}|I\big)\right)^{\frac{n_y-3}{2}}\left(1 - \mathrm{F}_y\big(y_{1/2}|I\big)\right)^{\frac{n_y+1}{2}} \mathrm{f}_y\big(y_{1/2}|I\big),$$

where,

$$\mathrm{F}_y\big(y_{1/2}|I\big) = \frac{1}{2}\left(1 + \mathrm{erf}\left(\frac{y_{1/2} - x}{\sigma\sqrt{2}}\right)\right),$$

and,

$$\mathrm{f}_y\big(y_{1/2}|I\big) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{\left(y_{1/2} - x\right)^2}{2\sigma^2}\right).$$

The PDFs for the sample mean and median are plotted in Figure 3-4.

It is straightforward to show that the mode of the PDFs:

$$\mathrm{f}_{\overline{y}}(\overline{y}|\hat{x}=x,\hat{\sigma}=\sigma)\,,$$

and,

$$\mathrm{f}_{y_{1/2}}\big(y_{1/2}|n,\hat{x}=x,\hat{\sigma}=\sigma\big)$$

occur at $\overline{y}^{*}=x$ and $y^{*}_{1/2}=x$, respectively. It can be seen from the plot in Figure 3-4 that it is fairly probable to calculate a value of the statistic that is close to the value of the state. Hence, both the sample mean and sample median can be used to estimate the value of the state. However, there is no guarantee that the sample mean (or median) will exactly equal the value of the state (in fact this is extremely unlikely). The real goal is to make inferences about the value of the state from the value of the statistic. For the sample mean this information can be obtained from the posterior PDF:

$$\mathrm{f}_{x}\big(x|\hat{\overline{y}}=y,\hat{\sigma}=\sigma\big)\,.$$

If the prior PDF is uniform, the posterior PDF is given by:

$$\mathrm{f}_{x}\big(x|\hat{\overline{y}}=y,\hat{\sigma}=\sigma\big)\propto\mathrm{f}_{\overline{y}}(\overline{y}|\hat{x}=x,\hat{\sigma}=\sigma)\,,$$

hence, inferences can be drawn directly from the sample mean. In many situations it is reasonable to assume that a uniform prior reflects ignorance about the true value of a parameter. However, sometimes an additional constraint (such as knowledge of the functional form of the likelihood function) may mean that a uniform prior does not fairly represent ignorance. This is a real drawback of inference by statistics. To emphasize the point:

$$\mathrm{f}_{x}\big(x|\hat{\overline{y}}=y,\hat{\sigma}=\sigma\big)\neq\mathrm{f}_{\overline{y}}(\overline{y}|\hat{x}=x,\hat{\sigma}=\sigma)\,,$$

in the general case. A further drawback of inference by statistics is that it is difficult to determine the functional form of a good statistic. A popular suggestion is the maximum likelihood method. In this method, the statistic is defined as the value of the state or parameter that maximizes the likelihood function (see [123] for a description of the method, together with its drawbacks). The maximum likelihood method is equivalent to maximizing the posterior PDF when the prior PDF is uniform.

**Example 3.5.2.** The likelihood function for $\hat{\mathbf{y}}$ is given by:

$$\mathrm{f}_{\mathbf{y}}(\mathbf{y}|\hat{\sigma} = \sigma) = \frac{1}{\left(\sigma\sqrt{2\pi}\right)^{n_y}} \exp\left(-\frac{\mathbf{y}^T\mathbf{y}}{2\sigma^2}\right).$$

Calculate the maximum likelihood estimate.

The value of $\sigma$ that maximizes the likelihood function is given by the solution of

$$\frac{\mathrm{d}}{\mathrm{d}\sigma}\frac{1}{\left(\sigma\sqrt{2\pi}\right)^{n_y}} \exp\left(-\frac{\mathbf{y}^T\mathbf{y}}{2\sigma^2}\right) = 0,$$

which on completing the differentiation yields:

$$\frac{1}{\left(\sigma\sqrt{2\pi}\right)^{n_y}} \exp\left(-\frac{\mathbf{y}^T\mathbf{y}}{2\sigma^2}\right)\left(\frac{\mathbf{y}^T\mathbf{y}}{\sigma^3} - \frac{n}{\sigma}\right) = 0.$$

Hence, the estimate of $\hat{\sigma}$ is

$$\sigma^* = \sqrt{\frac{\mathbf{y}^T\mathbf{y}}{n_y}}.$$

However, it has been shown that a better estimate of $\hat{\sigma}$ is in fact [29, 121, 122, 123, 244]:

$$\sigma^* = \sqrt{\frac{\mathbf{y}^T\mathbf{y}}{n_y - 1}}.$$

The maximum likelihood estimate is not optimal even for the situation where there is limited prior information. For samples that are not too small one can argue that the discrepancy between the Bayesian estimate and the maximum likelihood estimate is negligible. While this is true for the statistic derived in Example 3.5.2, in general this is not the case. The work [86] provides a catalogue of examples where the maximum likelihood estimate does not even asymptotically converge to the true parameter value.

## 3.6 Selecting a Likelihood Function

Estimation and prediction are two common problems of interest. The PDF,

$$\mathrm{f}_y(y|\hat{x} = x) \,,$$

or

$$\mathrm{f}_y(y|\hat{x} = x(u))$$

is necessary both for estimation and prediction, since the function characterizes ones belief that the output, $\hat{y}$, takes a certain value $y$ given the state (or inputs) of the system are known. It is therefore necessary to know how to select the appropriate PDF as a likelihood function. In this section, some of the more common PDFs that are used in engineering are discussed. In addition, it is shown in Examples 3.6.1–3.6.4 how the likelihood function can be used to make predictions about the output of system, $\hat{y}$, conditional on knowledge of the state, $\hat{x}$. However, knowledge of the likelihood function alone is insufficient to solve inference problems (for example: estimate the state of the system, $\hat{x}$ given a set of measurements, $\hat{\mathbf{y}} = \mathbf{y}$). For this task it is also necessary to assign a prior PDF to describe additional knowledge (or ignorance) about the value of $\hat{x}$. The assignment of prior PDFs is discussed in § 3.7. A summary of the common PDFs is included in Tables 3.3–3.5. The PDFs are classified as discrete, continuous, and derived. The discrete and continuous PDFs correspond to commonly used likelihood functions. In contrast, the derived PDFs do not correspond to commonly occurring likelihood functions, but rather to PDFs for specially defined functions of the measurements $g(\mathbf{y})$ (so called estimators), i.e., the PDF:

$$\mathrm{f}_{g(\mathbf{y})}(g(\mathbf{y})\,|\hat{x} = x) \,,$$

where $\mathbf{y} \in \mathbb{R}^{n_y}$ is a set of $n_y$ measurements (see § 3.5.2 for more details). Some of the continuous PDFs are defined in terms of the gamma function and incomplete gamma

Table 3.3: Discrete PDFs

| Density | Formula | Range |
|---|---|---|
| Binomial | $f(x\|n,p) = \binom{n}{x} p^x (1-p)^{(1-x)}$ | $0, \ldots, n$ |
| Uniform | $f(x\|N) = \frac{1}{N}$ | $1, \ldots, N$ |
| Geometric | $f(x\|p) = p(1-p)^x$ | $0, 1, \ldots$ |
| Hypergeom. | $f(x\|n, M, K) = \frac{\binom{K}{x}\binom{M-K}{n-x}}{\binom{M}{n}}$ | $0, \ldots, n$ |
| Poisson | $f(x\|\lambda, t) = \frac{(\lambda t)^x}{x!} e^{-\lambda t}$ | $0, 1, \ldots$ |

function. The gamma function and incomplete gamma functions are defined as

$$\Gamma(x) = \int_0^\infty t^{(x-1)} e^{-t}\,\mathrm{d}t, \quad 0 < x < \infty$$

and,

$$\mathrm{B}(a,b) = \frac{\Gamma(a)\,\Gamma(b)}{\Gamma(a+b)},$$

respectively.

### 3.6.1 Binomial Density

A common situation is where a trial is repeated $i = 1 : n$ times with two possible outcomes. The outcome of the $i^{\text{th}}$ trial is either $A$ or $\overline{A}$. The result of one trial does not influence subsequent trials, i.e., the result of each trial is independent from another. The PDF for the number of times $A$ occurs, $\hat{k}$, in $\hat{n} = n$ trials is referred to as the Binomial density and is given by:

$$\Pr(\text{A occurs k times in n trials}) = \mathrm{f}_k(k|\hat{n} = n) = \binom{n}{k} p^k q^{n-k} \quad q = 1 - p, \tag{3.43}$$

Table 3.4: Continuous PDFs

| Density | Formula | Range |
|---|---|---|
| Beta | $f(x\|a,b) = \frac{1}{B(a,b)} \beta^{x-1} (1-x)^{b-1}$ | $(0,1)$ |
| Exponential | $f(x\|\mu) = \frac{1}{\mu} \exp\left(-\frac{x}{\mu}\right)$ | $[0,\infty)$ |
| Log-Normal | $f(x\|\mu,\sigma) = \frac{1}{x\sigma\sqrt{2\pi}} \exp\left(-\frac{(\ln x-\mu)^2}{2\sigma^2}\right)$ | $(0,\infty)$ |
| Normal | $f(x\|\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$ | $(-\infty,\infty)$ |
| Uniform | $f(x\|a,b) = \frac{1}{a-b}$ | $[a,b]$ |

Table 3.5: Derived PDFs

| Density | Formula | Range |
|---|---|---|
| $\chi^2$ | $f(x\|\nu) = \frac{x^{\frac{(\nu-2)}{2}} \exp\left(-\frac{x}{2}\right)}{2^{\nu/2}\Gamma(\nu/2)}$ | $[0,\infty)$ |
| $F$ | $f(x\|\nu_1,\nu_2) = \frac{(\nu_1/\nu_2)^{\nu_1/2} x^{\frac{\nu_1-2}{2}}}{\mathrm{B}(\nu_1/2,\nu_2/2)} (1+\nu_1 x/\nu_2)^{-\frac{\nu_1+\nu_2}{2}}$ | $[0,\infty)$ |
| $t$ | $f(x\|\nu) = \frac{1}{\sqrt{\nu}\mathrm{B}(\nu/2,1)} \left(1+x^2/\nu\right)^{\frac{\nu+1}{2}}$ | $(-\infty,\infty)$ |

where,

$$\Pr(A) = p.$$

*Proof.* If the outcome of each trial is independent, the probability of a particular sequence (e.g., the probability of $A$ and then $\overline{A}$ occurring, $\Pr(A\overline{A})$) is the product of the probabilities $\Pr(A)$ and $\Pr(\overline{A})$.

$$\Pr(A\overline{A}) = \Pr(A)\Pr(\overline{A})$$

The number of different ways $A$ can occur $k$ times in $n$ trials is $\binom{n}{k}$, hence the binomial PDF is given by Equation (3.43). □

The appropriate use of the Binomial PDF is illustrated in Example 3.6.1.

**Example 3.6.1.** A cell has $n$ receptors, divided between the cell surface (area $A_{cell}$) and the endosome (area $A_{endosome}$). If the receptor shows no preference between the cell surface and the endosome, what is the probability of $k$ receptors occurring on the surface?

The probability of one receptor occurring on the surface is

$$p = \frac{A_{cell}}{A_{cell} + A_{endosome}}.$$

Hence the probability of $k$ of the $n$ receptors occurring on the cell surface is

$$\mathrm{f}_k(k|n) = \binom{n}{k}\left(\frac{A_{cell}}{A_{cell} + A_{endosome}}\right)^k \left(1 - \frac{A_{cell}}{A_{cell} + A_{endosome}}\right)^{n-k}.$$

### 3.6.2 Poisson Density

The Binomial PDF described in § 3.6.1, characterizes the number of times an event occurs (number of successes) in a discrete medium (number of trials). However, often one is interested in the number of times an event occurs in a continuous medium (for example: the number of photons that arrive in a fixed period of time). The Poisson

PDF characterizes this situation and is given by:

$$\Pr\Big(\text{k events}|\hat{\lambda}\hat{l} = \lambda l\Big) = \mathrm{f}_k\Big(k|\hat{\lambda}\hat{l} = \lambda l\Big) = e^{-\lambda l}\frac{(\lambda l)^k}{k!}, \tag{3.44}$$

where the frequency of events is described by the parameter, $\hat{\lambda}$, and the amount of continuous medium is $l$.

*Proof.* Consider an interval of length $L$, which is divided into two non-overlapping sections: of length $l$ and $L - l$. $n$ points are distributed at random throughout the whole interval. The probability that one point occurs in the first section is given by:

$$p = \frac{l}{L}.$$

Hence, the probability that $k$ of the $n$ points lie in section one is given by:

$$\Pr(\text{k of n points lie in section one}) = \binom{n}{k} p^k q^{n-k}.$$

If $p \ll 1$ and $k \approx np$, then $k \ll n$ and $kp \ll 1$. It follows that

$$\begin{aligned} \binom{n}{k} &= \frac{n(n-1)\dots(n-k+1)}{1.2\dots k} \\ &\approx \frac{n^k}{k!} \\ q &= 1 - p \approx e^{-p} \\ q^{n-k} &\approx e^{-(n-k)p} \approx e^{-np}. \end{aligned}$$

and,

$$\Pr(\text{k of n points lie in section one}) \approx e^{-np}\frac{(np)^k}{k!}.$$

Defining, $\hat{\lambda} = \hat{n}/L$, and assuming $\lambda$ remains constant as $L \to \infty$, then Equation (3.44) follows. □

**Example 3.6.2.** Let the rate at which photons arrive at a microscope be $0.5s^{-1}$. If a sample is viewed for $3s$, what is the probability that the detector encounters at least

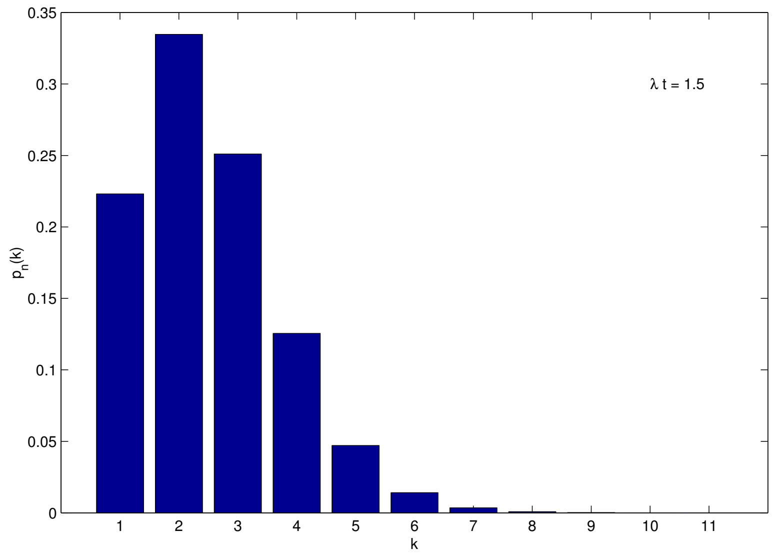


Figure 3-5: Poisson density for Example 3.6.2

2 photons?

It is necessary to evaluate, $\Pr\left(\hat{k} \geq 2\right)$. By mutual exclusivity:

$$\Pr\left(\hat{k} \geq 2\right) = 1 - \Pr\left(\hat{k} < 2\right).$$

Subsituting the Poisson PDF,

$$\Pr\left(\hat{k} \geq 2\right) = 1 - e^{-\lambda t} \sum_{k=0}^{1} \frac{(\lambda t)^k}{k!},$$

which evaluates to

$$\Pr(k \geq 2) = 0.5578.$$

The Poisson density function for $\hat{\lambda}\hat{t} = 1.5$ is shown in Figure 3-5.

### 3.6.3 Exponential Density

The exponential PDF is closely related to the Poisson PDF. This PDF is useful in characterizing the amount of medium between events that occur in a continuous medium (for example: length of time between $\alpha$-particle emissions, distance between defects in DNA, etc.). The exponential PDF is given by:

$$\mathrm{f}_t\left(t|\hat{\lambda}=\lambda\right)=\begin{cases}0 & t<0\\ \lambda e^{-\lambda t} & t\geq 0.\end{cases}. \tag{3.45}$$

where $\hat{t}$ is the quantity of the medium between events and $\hat{\lambda}$ is a parameter that characterizes the frequency of events.

*Proof.* Define the CDF $\mathrm{F}_t\left(t|\hat{\lambda}=\lambda\right)$ as,

$$\mathrm{F}_t\left(t|\hat{\lambda}=\lambda\right)\equiv\Pr\left(\hat{t}\leq t|\hat{\lambda}=\lambda\right)$$

which is equivalent to the statement,

$$\mathrm{F}_t\left(t|\hat{\lambda}=\lambda\right)\equiv\Pr\left(\text{There are at least one or more events in time } t|\hat{\lambda}=\lambda\right).$$

From mutual exclusivity it follows that

$$\mathrm{F}_t\left(t|\hat{\lambda}=\lambda\right)=1-\Pr\left(\text{There are at zero events in time } t|\hat{\lambda}=\lambda\right),$$

which on substitution of the Poisson density yields:

$$\mathrm{F}_t\left(t|\hat{\lambda}=\lambda\right)=1-e^{-\lambda t}\frac{(\lambda t)^0}{0!}=1-e^{-\lambda t}.$$

By definition, the exponential density function is the derivative of the CDF,

$$\mathrm{F}_t\left(t|\hat{\lambda}=\lambda\right),$$

yielding the PDF in Equation (3.45). □

**Example 3.6.3.** On average a migrating cell changes direction every 15 minutes. Calculate the probability that the cell turns in the first five minutes.

From the data in the question:

$$\hat{\lambda} = \frac{1}{15}\text{min}^{-1},$$

hence,

$$\begin{aligned}\Pr\left(\text{Cell turns in the first five minutes}|\hat{\lambda} = \frac{1}{15}\right) &= \int_0^5 \frac{1}{15}\exp\left(-\frac{1}{15}t\right)\,\mathrm{d}t \\ &= 1-\exp(-5/15) = 0.2835.\end{aligned}$$

### 3.6.4 Normal Density

Perhaps the most commonly used PDF in science and engineering is the Normal density and in standard form is given by:

$$\mathrm{f}_x(x) = \frac{1}{\sqrt{2\pi}}\exp\left(-\frac{x^2}{2}\right). \tag{3.46}$$

The Normal PDF naturally describes the limit of some physical processes. For example, it has been shown that the Normal density is the appropriate PDF for the position of a particle undergoing Brownian motion [76, 213, 139]. In a closely related problem, it has been shown that the PDF for a vector $\mathbf{r} \in \mathbb{R}^3$ is approximately Normal when $\mathbf{r}$ is the sum of $N \gg 1$ displacements, $\mathbf{r}_i$, and the PDF for $\mathbf{r}_i$ is arbitrary (see page 15 of [37] for a proof of this property). This is a variant of the famous Central Limit Theorem, which states the mean of $N$ samples is approximately Normal for sufficiently large $N$. This property is often invoked as a justification for using the Normal density to model an output variable, $y$, that is related to a state, $x$ by many additive errors. A useful derivation of the Normal density can be obtained by Maximum Entropy. The Entropy of a PDF characterizes the information content of the PDF (first shown by [200], see [205, 121] for discussion). Maximum ignorance (i.e., maximum Entropy, $H$),

$$H = -\int_{-\infty}^{\infty} \mathrm{f}_x(x|\alpha,\beta)\log \mathrm{f}_x(x|\alpha,\beta)\,\mathrm{d}x$$

about the value of $\hat{x}$ subject to the constraints:

$$\mathrm{E}_x(x) = \alpha, \quad \mathrm{Var}(x) = \beta^2,$$

can expressed by assigning the scaled Normal PDF:

$$\mathrm{f}_x(x|\alpha,\beta) = \frac{1}{\beta\sqrt{2\pi}} \exp\left(-\frac{(x-\alpha)^2}{2\beta^2}\right).$$

Hence, if only the first and second moments of a PDF are known then the PDF which expresses least information about the value of $\hat{x}$ is the scaled Normal PDF [205, 121].

A derivation of the joint normal PDF under relatively few assumptions was made by Herschel and Maxwell [121]. Herschel considered the errors made in measuring the position of a star $(\hat{\epsilon}_1, \hat{\epsilon}_2)$. The following assumptions were made:

1. Knowledge of $\hat{\epsilon}_1$ tells us nothing about $\hat{\epsilon}_2$:

$$\mathrm{f}_{\epsilon_1,\epsilon_2}(\epsilon_1,\epsilon_2)\,\mathrm{d}\epsilon_1\mathrm{d}\epsilon_2 = \mathrm{f}_{\epsilon}(\epsilon_1)\,\mathrm{d}\epsilon_1 \cdot \mathrm{f}_{\epsilon}(\epsilon_2)\,\mathrm{d}\epsilon_2. \tag{3.47}$$

2. The PDF can be written in polar coordinates:

$$\mathrm{f}_{\epsilon_1,\epsilon_2}(\epsilon_1,\epsilon_2)\,\mathrm{d}\epsilon_1\mathrm{d}\epsilon_2 = \mathrm{f}_{r,\theta}(r,\theta)\,r\mathrm{d}r\mathrm{d}\theta. \tag{3.48}$$

3. The PDF of the errors $(\hat{\epsilon}_1, \hat{\epsilon}_2)$ is independent of angle (invariant transformation):

$$\mathrm{f}_{r,\theta}(r,\theta) = \mathrm{f}_r(r)\,. \tag{3.49}$$

Herschel showed that these assumptions were consistent with assigning the joint Normal density:

$$\mathrm{f}_{\epsilon_1,\epsilon_2}(\epsilon_1,\epsilon_2) = \frac{\alpha}{\pi}\exp\bigl(-\alpha\left(\epsilon_1^2+\epsilon_2^2\right)\bigr)\,. \tag{3.50}$$

*Proof.* [121] Combining Equations (3.47)–(3.49) yields:

$$\mathrm{f}_r\left(\sqrt{\epsilon_1^2+\epsilon_2^2}\right) = \mathrm{f}_\epsilon(\epsilon_1)\,\mathrm{f}_\epsilon(\epsilon_2)\,. \tag{3.51}$$

Setting $\epsilon_2 = 0$ gives:

$$\mathrm{f}_r(\epsilon_1) = \mathrm{f}_\epsilon(\epsilon_1)\,\mathrm{f}_\epsilon(0)\,,$$

which implies:

$$\mathrm{f}_r\left(\sqrt{\epsilon_1^2+\epsilon_2^2}\right) = \mathrm{f}_\epsilon\left(\sqrt{\epsilon_1^2+\epsilon_2^2}\right)\mathrm{f}_\epsilon(0)\,. \tag{3.52}$$

Eliminating $\mathrm{f}_r\left(\sqrt{\epsilon_1^2+\epsilon_2^2}\right)$ from Equations (3.51)–(3.52) yields:

$$\frac{\mathrm{f}_\epsilon(\epsilon_1)\,\mathrm{f}_\epsilon(\epsilon_2)}{\left(\mathrm{f}_\epsilon(0)\right)^2} = \frac{\mathrm{f}_\epsilon\left(\sqrt{\epsilon_1^2+\epsilon_2^2}\right)}{\mathrm{f}_\epsilon(0)}.$$

Taking logarithms of both sides yields:

$$\log\frac{\mathrm{f}_\epsilon(\epsilon_1)}{\mathrm{f}_\epsilon(0)} + \log\frac{\mathrm{f}_\epsilon(\epsilon_2)}{\mathrm{f}_\epsilon(0)} = \log\frac{\mathrm{f}_\epsilon\left(\sqrt{\epsilon_1^2+\epsilon_2^2}\right)}{\mathrm{f}_\epsilon(0)}. \tag{3.53}$$

The solution of Equation (3.53) is

$$\log\frac{\mathrm{f}_\epsilon(\epsilon_1)}{\mathrm{f}_\epsilon(0)} = -\alpha\epsilon_1^2$$

which when properly normalized yields:

$$\mathrm{f}_\epsilon(\epsilon_1) = \sqrt{\frac{\alpha}{\pi}}\exp\left(-\alpha\epsilon_1^2\right).$$

Hence, the joint density for the errors in the measurement $(\hat{\epsilon}_1, \hat{\epsilon}_2)$, is given by Equation (3.50). □

Often a measurement $\hat{y}$ is related to a state $\hat{x}$ by an additive error, $\hat{\epsilon}$:

$$\hat{y} = \hat{x} + \hat{\sigma}\hat{\epsilon}, \tag{3.54}$$

where $\hat{\sigma}$ is a parameter that describes the magnitude of the error. If the PDF for the error, $\hat{\epsilon}$, is Normal (shown in Equation (3.46)), by Theorem (3.3.4) the PDF for the output $\hat{y}$ is given by the scaled Normal density:

$$\mathrm{f}_y(y|\hat{x}=x,\hat{\sigma}=\sigma)=\frac{1}{\sigma\sqrt{2\pi}}\exp\left(-\frac{(y-x)^2}{2\sigma^2}\right), \tag{3.55}$$

(see Example 3.3.4).

**Example 3.6.4.** Given a migrating cell is at position $\{\hat{x}_1,\hat{x}_2\}=\{10,10\}$ and measurements are made with a variance $\hat{\sigma}^2=1$, what is the probability of making a measurement of the output in the range $(9\le\hat{y}_1\le 11, 8\le\hat{y}_2\le 12)$?

Assuming the errors in the measurement of each coordinate are independent and the PDF for the measurement error is Normal, the joint PDF is:

$$\begin{aligned}\mathrm{f}_{y_1,y_2}(y_1,y_2|\hat{x}_1=10,\hat{x}_2=10,\hat{\sigma}=1) &= \mathrm{f}_{y_1}(y_1|\hat{x}_1=10,\hat{\sigma}=1)\,\mathrm{f}_{y_2}(y_2|\hat{x}_2=10,\hat{\sigma}=1)\\ &= \frac{1}{2\pi}\exp\left(-\frac{(y_1-10)^2+(y_2-10)^2}{2}\right)\end{aligned}$$

From the definition of a continuous PDF,

$$\Pr(9\le\hat{y}_1\le 11, 8\le\hat{y}_2\le 12)=\int_9^{11}\int_8^{12}\mathrm{f}_{y_1,y_2}(y_1,y_2|\hat{x}_1=10,\hat{x}_2=10,\hat{\sigma}=1)\ \mathrm{d}x\mathrm{d}y.$$

Making the necessary substitutions:

$$\mathrm{f}_{y_1,y_2}(y_1,y_2|\hat{x}_1=10,\hat{x}_2=10,\hat{\sigma}=1)=\mathrm{erf}\left(\frac{1}{\sqrt{2}}\right)\mathrm{erf}\left(\frac{2}{\sqrt{2}}\right)=0.6516.$$

## 3.6.5 Log-Normal Density

The Normal PDF is inappropriate for the situation where the error in the measurement scales with the magnitude of the measurement. The appropriate density to use is the log-normal density. Furthermore, the log-normal PDF is zero for negative values of the variable. The probability density is defined by Equation (3.56).

$$f_x(x|\hat{\mu} = \mu, \hat{\sigma} = \sigma) = \frac{1}{\sigma x \sqrt{2\pi}} \exp\left(-\frac{(\log x - \mu)^2}{2\sigma^2}\right) \tag{3.56}$$

## 3.7 Prior Probability Density Functions

In § 3.6 the assignment of likelihood functions was discussed; the goal was to use Equation (3.36) of Theorem 3.5.1 to make inferences about a state, $\hat{x}$, given a measurement $\hat{y}$. Frequently, the correct functional form of the likelihood function is prescribed by the underlying physics of the system of interest. However, it is necessary to assign a prior PDF before Equation (3.36) can be used for this task. Naturally, the question arises as to how one should assign *prior* PDFs. Unfortunately, this question is not as straightforward to answer as the assignment of likelihood functions.

In some situations, a *subjective* prior PDF can be assigned based on data obtained from previous experimentation or from the literature. The adjective, “subjective”, should not be taken to mean the process of Bayesian inference is unfair. Indeed, the process of updating the posterior PDF through the likelihood function allows one to change one’s mind in light of the data.

**Example 3.7.1.** Given a subjective prior PDF for $\hat{x}$:

$$\pi_x(x) = \frac{1}{\sigma_0\sqrt{2\pi}} \exp\left(-\frac{(x - \mu_0)^2}{2\sigma_0^2}\right),$$

the most probable value of $\hat{x}$ is initially close to $\mu_0$. If $n_y$ independent measurements of the output $\hat{y}$ are made, and the likelihood function for the output $\hat{y}$ is:

$$f_y(y_i|\hat{x} = x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(y_i - x)^2}{2\sigma^2}\right),$$

derive the posterior PDF and determine the most probable value of $\hat{x}$ after the measurements have been made.

Substituting the prior PDF and the likelihood function into Equation (3.42), yields:

$$\mathrm{f}_x(x|\hat{\mathbf{y}}=\mathbf{y}) \propto \frac{1}{\sigma_0\sqrt{2\pi}} \exp\left(-\frac{(x-\mu_0)^2}{2\sigma_0^2}\right) \prod_{i=1}^{n_y} \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(y_i-x)^2}{2\sigma^2}\right),$$

which on rearrangement and normalization yields:

$$\mathrm{f}_x(x|\hat{\mathbf{y}}=\mathbf{y}) = \frac{1}{\sigma_1\sqrt{2\pi}} \exp\left(-\frac{(x-\mu_1)^2}{2\sigma_1^2}\right) \tag{3.57}$$

where,

$$\mu_1 = \sigma_1^2\left(\frac{\overline{y}}{\sigma^2/n_y} + \frac{\mu_0}{\sigma_0^2}\right)$$

and,

$$\sigma_1 = \sqrt{\frac{\sigma^2\sigma_0^2/n_y}{\sigma^2/n_y+\sigma_0^2}}.$$

Therefore, the most probable value of $\hat{x}$ lies close to $\mu_1$. It can be seen that $\mu_1$ is the weighted average of $\hat{y}$ and $\mu_0$. As more data are collected, $n_y \to \infty$, $\mu_1 \to \overline{y}$, i.e., the contribution of the prior PDF to the posterior PDF becomes negligible and the data become more important.

However, often one is faced with the situation where there is little or no information pertaining to the value of a quantity to be inferred. In this situation, the assignment of such a peaked PDF is not appropriate. There are three common techniques for determining a prior PDF to express small amounts of knowledge relative to the information available from the data: the principle of indifference, the principle of invariance, and the principle of a data translated likelihood. Each of these methods seeks to minimize the impact of the prior PDF on the posterior PDF and consequently produces a non-informative prior PDF.

In all of these situations, it is assumed that it is known that the parameter lies in a certain range, $x_{min} < \hat{x} \leq x_{max}$. This is not too much of a restriction since it is extremely unusual to perform an experiment where no bounds on the states and parameters are known *a priori*. For example, a rate constant has a minimum bound of zero and an upper bound determined by the maximum rate of collisions. The prior

PDF is *proper* when *a priori* bounds are known on the parameter.

### 3.7.1 Indifferent Prior

Perhaps the simplest method for assigning a *prior* probability density is the principle of indifference. Consider $n$ propositions $A_1, A_2, ..., A_n$ which depend on the information, $B$ in the same way and are mutually exclusive. In Theorem 3.2.1 of § 3.2.2 it was established that

$$\Pr(A_i|B) = \frac{1}{n}.$$

If one is *completely* ignorant about the value of a discrete state, $\hat{x}$, (between certain bounds) then

$$\Pr(x_1 < \hat{x} \leq x_2|I) = \Pr(x_3 < \hat{x} \leq x_4|I)\,,$$

if $x_2 - x_1 = x_4 - x_3$. It follows that a uniform prior PDF should be assigned for $\hat{x}$, as defined in Equation (3.58).

$$\pi_x(x) = \frac{1}{x_{max} - x_{min} + 1} \quad x_{min} \leq \hat{x} \leq x_{max} \tag{3.58}$$

For a continuous state, the prior PDF is:

$$\pi_x(x) = \begin{cases} \frac{1}{x_{max}-x_{min}} & x_{min} \leq \hat{x} \leq x_{max}, \\ 0 & \text{otherwise.} \end{cases} \tag{3.59}$$

The difficulty with always assigning a uniform prior PDF is that it is rare that one knows absolutely nothing about the parameter to be inferred. There are two common situations:

1. It is known that the parameter should remain invariant under some transformation. For example, the units of measurement for the error between the output and state may be unknown. Clearly, the form of the prior PDF should not alter if the units of the problem are changed.

2. The form of the likelihood function is known. Hence, one may wish to assign a

prior PDF which biases the posterior PDF by the least amount.

These two situations are discussed in § 3.7.2–3.7.3.

### 3.7.2 Invariant Prior

It is often the case that one can argue the inference problem should remain the same even if the problem is reparameterized. For example, the units of measurement may not be known *a priori*. In this situation, the prior PDF can be determined by the theory of invariance groups [120]. The method relies on determining a coordinate transform under which the posterior PDF remains unchanged. Application of the method is demonstrated in Example 3.7.2.

**Example 3.7.2.** Suppose the likelihood function is defined as

$$\mathrm{f}_y(y|\hat{x}=x,\hat{\sigma}=\sigma) = h\left(\frac{y-x}{\sigma}\right), \tag{3.60}$$

but the units of the output, $\hat{y}$, are unknown (for example: $\hat{y}$ may be measured in Celsius or Fahrenheit). It is reasonable to assume that the posterior PDF remains invariant under a change of units. Use this information to determine the prior PDF for $(\hat{x},\hat{\sigma})$.

Suppose $(\hat{x},\hat{y},\hat{\sigma})$ are the problem variables in Celsius and $(\hat{x}',\hat{y}',\hat{\sigma}')$ are the problem variables in Fahrenheit. The coordinate transform between both sets of variables is

$$\begin{aligned} \hat{x}' &= a\hat{x}+b && (3.61)\\ \hat{\sigma}' &= a\hat{\sigma} && (3.62)\\ \hat{y}' &= a\hat{y}+b. && (3.63) \end{aligned}$$

If the problem remains unchanged then:

$$\mathrm{f}_{x,\sigma}(x,\sigma|\hat{y}=y) = \mathrm{f}_{x',\sigma'}(x',\sigma'|\hat{y}'=y')\,. \tag{3.64}$$

The posterior PDF for the original problem can be written as:

$$\mathrm{f}_{x,\sigma}(x,\sigma|\hat{y}=y) \propto h\left(\frac{y-x}{\sigma}\right)\pi_{x,\sigma}(x,\sigma)\,. \tag{3.65}$$

The posterior PDF for the transformed variables can be written as (Theorem 3.3.4):

$$\mathrm{f}_{x',\sigma'}(x',\sigma'|\hat{y}=y) \propto \frac{1}{a^2}h\left(\frac{y'-x'}{\sigma'}\right)\pi_{x,\sigma}\left(\frac{x'-b}{a},\frac{\sigma'}{a}\right)\,. \tag{3.66}$$

Combining Equations (3.64)–(3.66) yields the following functional relationship:

$$\pi_{x,\sigma}(x,\sigma) = \frac{1}{a^2}\pi_{x,\sigma}\left(\frac{x-b}{a},\frac{\sigma}{a}\right)\,. \tag{3.67}$$

It is straightforward to verify that the relationship in Equation (3.67) implies the prior PDF:

$$\pi_{x,\sigma}(x,\sigma) = \frac{\text{const}}{\sigma^2}. \tag{3.68}$$

The prior PDF shown in Equation (3.68) should not be used universally for a likelihood function of the form in Equation (3.60). The coordinate transform suggested in Equations (3.61)–(3.63) presupposes that knowledge about $\hat{x}$ and $\hat{\sigma}$ is not independent since the transformation to the new variables has a common parameter between $\hat{x}'$ and $\hat{\sigma}'$ (see [29, 123] for discussion). A transformation that expresses greater ignorance is therefore [120]:

$$\begin{aligned}\hat{x}' &= \hat{x}+b\\ \hat{\sigma}' &= a\hat{\sigma}\\ \hat{y}'-\hat{x}' &= a\left(\hat{y}-\hat{x}\right)\end{aligned}$$

which results in the prior PDF:

$$\pi_{x,\sigma}(x,\sigma) = \frac{\text{const}}{\sigma}, \tag{3.69}$$

which was originally proposed by [122, 123].

### 3.7.3 Data Translated Likelihood Prior

Another method for exploiting the functional form of the likelihood function for determining the prior PDF is described in [29]. It should be stressed that this method for determining a prior PDF does not necessarily yield the same PDF as the method described in § 3.7.2 since a data translated likelihood prior presupposes less *a priori* information (invariance requires knowledge of the likelihood function and a transform).

When the likelihood function takes the form:

$$g(x - f(\mathbf{y}))\,,$$

the quantity $\hat{x}$ is a *location* parameter. The data do not change the shape of the likelihood function but do change the location of the PDF. A non-informative prior for such a likelihood function is uniform [122, 123].

The goal of this method is to find a variable transform for the inferred variables such that it is reasonable to assign a uniform PDF for the transformed problem. For a single state, $\hat{x}$, the method works by finding a variable transformation such that the transformed likelihood function can be expressed as

$$\mathrm{f}_{\mathbf{y}}(\mathbf{y}|\hat{x}' = x') = g(x' - f(\mathbf{y}))\,. \tag{3.70}$$

The rationale is that if a uniform prior is assigned for $\hat{x}'$ then the shape of the posterior PDF for the transformed variable *does not* change regardless of the data collected. It is then possible to determine the prior PDF for the original likelihood function based on the variable transformation $\hat{x} \rightarrow \hat{x}'$.

**Example 3.7.3.** [29] A likelihood function has the form:

$$\mathrm{f}_{\mathbf{y}}(\mathbf{y}|\hat{\sigma} = \sigma) = h\left(\frac{\mathbf{y}^T\mathbf{y}}{\sigma}\right).$$

Determine the data translated prior PDF.

By inspection [29]:

$$\mathrm{f}_{\mathbf{y}}(\mathbf{y}|\hat{\sigma} = \sigma) = h\big(\exp\big(\log\big(\mathbf{y}^T\mathbf{y}\big) - \log\sigma\big)\big)\,,$$

hence, defining $\hat{\sigma}' = \log\hat{\sigma}$ yields the desired transformation. Given, that it is now appropriate to assign a uniform prior PDF for $\hat{\sigma}'$ this implies a prior PDF for $\hat{\sigma}$:

$$\pi_\sigma(\sigma) = \frac{\mathrm{const}}{\sigma}.$$

A difficulty arises with this method since it is not always possible to find a transformation to bring the likelihood function into data translated form (Equation (3.70)). It can be shown that the likelihood can be approximately transformed by defining the transformation (Pages 36–38 of [29]):

$$\frac{\mathrm{d}\hat{x}'}{\mathrm{d}\hat{x}} \propto H^{1/2}(t)\,,$$

where $H(\cdot)$ is the expected value of the second derivative of the log-likelihood function:

$$H(x) = -\mathrm{E}_{\mathbf{y}|x}\left(\frac{\mathrm{d}^2\log\mathrm{f}_{\mathbf{y}}(\mathbf{y}|\hat{x} = x)}{\mathrm{d}x^2}\right).$$

The corresponding prior PDF for $\hat{x}$ is

$$\pi_x(x) \propto H^{1/2}(x)\,. \tag{3.71}$$

There is a multiparameter version of the data translated likelihood rule when the goal is to infer the value of several states, $\hat{\mathbf{x}}$, given measurements, $\mathbf{y}$ of the outputs $\hat{\mathbf{y}}$ [29]. However, this rule is not necessarily very satisfactory since it is impossible to guarantee a transformation that preserves the shape of the likelihood function with respect to a change in the data $\mathbf{y}$. The best that one can hope for is that a change in data approximately preserves the volume enclosed by the likelihood function. The prior PDF for $\hat{\mathbf{x}}$ is then [29]:

$$\pi_{\mathbf{x}}(\mathbf{x}) \propto |\mathbf{H}(\mathbf{x})|^{1/2}, \tag{3.72}$$

where $\mathbf{H}(\mathbf{x})$ is the expected value of the Hessian matrix of the log-likelihood function:

$$\mathbf{H}(\mathbf{x}) = -\mathrm{E}_{\mathbf{y}|\mathbf{x}}\left( \frac{\partial^2 \log \mathrm{f}_{\mathbf{y}}(\mathbf{y}|\hat{\mathbf{x}} = \mathbf{x})}{\partial x_i x_j} \right).$$

# Chapter 4

# Bayesian Analysis of Cell Signaling Networks

The work in this Chapter focuses on applying the techniques developed in Chapter 3 to analyzing cell-signaling networks. Typically, there are three main goals when analyzing cell-signaling data:

1. determing model parameters when the structure of the model is known,
2. determing the most probable model structure supported by the data, and,
3. using the knowledge gained from the data to design more informative experiments.

It is relatively straightforward to use Bayesian statistics to develop mathematical formulations for all three goals; the challenge is in developing reliable computational techniques to solve these formulations. Some work was devoted in this thesis to developing such techniques. While only limited progress was made towards solving these problems, recent developments in optimization [203] suggest that this topic remains an interesting research question. It is necessary to solve the first two problems to develop experimental design criteria. Hence, it was decided to focus on the first two goals: parameter estimation and model selection.

## 4.1 Parameter Estimation

In the simplest formulation, it is assumed that the relationship between the inputs, $\hat{\mathbf{u}} \in \mathbb{R}^{n_u}$, and the states, $\hat{\mathbf{x}} \in \mathbb{R}^{n_x}$, of the system is deterministic; i.e.,

$$\hat{\mathbf{x}} = \mathbf{g}(\hat{\mathbf{u}}, \hat{\mathbf{p}}_1), \tag{4.1}$$

where $\hat{\mathbf{p}}_1 \in \mathbb{R}^{n_{p1}}$ is a vector of model parameters. It is also assumed that the model inputs are known with infinite precision, and the function, $\mathbf{g} : \mathbb{R}^{n_u \times n_{p1}} \to \mathbb{R}^{n_x}$, uniquely maps the inputs and parameters, $(\hat{\mathbf{u}}, \hat{\mathbf{p}}_1)$, to the states, $\hat{\mathbf{x}}$. The function defined in Equation (4.1) is sometimes called an expectation function [29]. This terminology is not unreasonable since often the outputs, $\hat{\mathbf{y}}$, are an average property of a stochastic system (for example: thermodynamic quantities such as temperature, concentrations of reacting species, etc.). However, for biological systems it is not always valid to assume the states are a deterministic function of the inputs. For example, cell migration (studied in Chapter 5) is a stochastic phenomenon. However, the deterministic approximation is often realistic for cell regulatory mechanisms. As shown in Chapter 2, cell-signaling models can often be formulated as systems of ODEs.

The outputs of the system, $\hat{\mathbf{y}}$, are related to the state of the system through a probabilistic measurement model,

$$\mathrm{f}_{\mathbf{y}}(\mathbf{y}|\hat{\mathbf{x}} = \mathbf{x}, \hat{\mathbf{p}}_2 = \mathbf{p}_2), \tag{4.2}$$

where $\hat{\mathbf{p}}_2 \in \mathbb{R}^{n_{p2}}$ is a set of parameters that characterize the measurement process. It is normally reasonable to assume that the measurement error is well characterized since the error is usually related to the precision and accuracy of the experimental apparatus. Common measurement models are discussed in § 3.6, Chapter 3. The input-output relationship (Equation (4.1)) and the measurement model (Equation (4.2)) can be combined to yield the likelihood function for a single measurement $\mathbf{y} \in \mathbb{R}^{n_y}$:

$$\mathrm{f}_{\mathbf{y}}(\mathbf{y}|\hat{\mathbf{u}} = \mathbf{u}, \hat{\mathbf{p}} = \mathbf{p}) = \mathrm{f}_{\mathbf{y}}(\mathbf{y}|\hat{\mathbf{x}} = \mathbf{g}(\hat{\mathbf{u}}, \hat{\mathbf{p}_1}), \hat{\mathbf{u}} = \mathbf{u}, \hat{\mathbf{p}} = \mathbf{p}), \tag{4.3}$$

where $\hat{\mathbf{p}} = (\hat{\mathbf{p}}_1, \hat{\mathbf{p}}_2)$. Often, it is reasonable to assume the measurements of the system output are independent between experiments. Hence, the likelihood function for $n_k$ measurements is

$$\mathrm{f}_{\mathbf{Y}}\Big(\mathbf{Y}|\hat{\mathbf{U}} = \mathbf{U}, \hat{\mathbf{p}} = \mathbf{p}\Big) = \prod_{i=1}^{n_k} \mathrm{f}_{\mathbf{y}}(\mathbf{y}_i|\hat{\mathbf{u}}_i = \mathbf{u}_i, \hat{\mathbf{p}} = \mathbf{p}) \,, \tag{4.4}$$

where $\mathbf{Y} \in \mathbb{R}^{n_y \times n_k}$ is a matrix of $n_k$ measurements, $\mathbf{y}_i$, and $\hat{\mathbf{U}} \in \mathbb{R}^{n_u \times n_k}$ is a matrix of $n_k$ input conditions. Finally, it is necessary to assign a prior PDF for the model parameters, $\pi_{\mathbf{p}}(\mathbf{p})$. Techniques for determining a suitable prior PDF are discussed in § 3.7, Chapter 3. One must take caution when assigning the prior PDF for the parameters if the problem depends on a large number of parameters [29]. By application of Bayes' Theorem, the posterior PDF for the parameters $\hat{\mathbf{p}}$ is:

$$\mathrm{f}_{\mathbf{p}}\Big(\mathbf{p}|\hat{\mathbf{Y}} = \mathbf{Y}, \hat{\mathbf{U}} = \mathbf{U}\Big) = \mathrm{f}_{\mathbf{Y}}\Big(\mathbf{Y}|\hat{\mathbf{U}} = \mathbf{U}, \hat{\mathbf{p}} = \mathbf{p}\Big)\, \pi_{\mathbf{p}}(\mathbf{p}) \,. \tag{4.5}$$

Often, the input-output relationship in Equation (4.1) is defined implicitly. For example, the relationship may be defined as the solution of a nonlinear set of equations:

$$\mathbf{g}(\hat{\mathbf{u}}, \hat{\mathbf{x}}, \hat{\mathbf{p}}) = \mathbf{0},$$

or as the solution of a system of differential equations at fixed times,

$$\hat{\mathbf{x}}' = \mathbf{g}(\hat{\mathbf{x}}, \hat{\mathbf{u}}, \hat{\mathbf{p}}) \,,$$

or as the solution of DAE/PDAE models at fixed times. It should be stressed that the posterior PDF given in Equation (4.5) is conditional on prior knowledge used to assign the expectation function, measurement model and prior PDF for $\hat{\mathbf{p}}$. To denote this dependence more clearly we will write:

$$\mathrm{f}_{\mathbf{p}}(\mathbf{p}|\hat{\mathbf{y}} = \mathbf{y}, \hat{\mathbf{u}} = \mathbf{u}, M) \,, \tag{4.6}$$

where $M$ is the statement that the assigned models (measurement, prior and expectation) are true. The consequences of not properly accounting for structural uncertainty in a model was discussed in Chapter 3 and is also discussed more fully in [67].

Estimates of the parameters and confidence intervals can be obtained directly from the posterior PDF in Equation (4.5). Typically, such estimates rely on solving a global optimization problem related to the posterior PDF. A typical estimate is the Maximum A *Posteriori* (MAP) estimate (Definition 4.1.1).

**Definition 4.1.1.** The MAP estimate is defined as the solution, $\mathbf{p}^*$, of the following problem:

$$\max_{\mathbf{p}\in P\subset\mathbb{R}^{n_p},\mathbf{X}\in\mathbb{R}^{n_x\times n_k}} \mathrm{f}_{\mathbf{Y}}\left(\mathbf{Y}|\hat{\mathbf{X}}=\mathbf{X}\right)\pi_{\mathbf{p}}(\mathbf{p})$$

subject to the following constraints:

$$\begin{aligned}\mathbf{g}(\mathbf{x}_1,\mathbf{u}_1,\mathbf{p}) &= 0\\ \vdots\quad &\quad \vdots\\ \mathbf{g}(\mathbf{x}_{n_k},\mathbf{u}_{n_k},\mathbf{p}) &= 0\end{aligned}$$

where $\mathbf{g}(\mathbf{x},\mathbf{u},\mathbf{p})$ may either be an algebraic constraint or implicitly define $\mathbf{x}(\mathbf{u},\mathbf{p})$ as the solution at fixed times to a dynamic system. The space $P$ is defined by the prior PDF, $\pi_{\mathbf{p}}(\mathbf{p})$.

The MAP parameter estimate is illustrated in Example 4.1.1.

**Example 4.1.1.** Consider the chemical reaction:

$$A \xrightarrow{p_1} B \xrightarrow{p_2} C.$$

It is desired to estimate the rate constants $\hat{p}_1$ and $\hat{p}_2$ from a single time-series experiment. The time-series concentrations of $A$, $B$, and $C$ are shown in Table 4.1. The concentrations of the reacting species are given by:

$$\begin{aligned} x_1'(t) &= -p_1 x_1(t) \\ x_2'(t) &= p_1 x_1(t) - p_2 x_2(t) \\ x_3'(t) &= p_2 x_2(t)\,, \end{aligned} \tag{4.7}$$

and the measurement model is given by:

$$\begin{gathered} \mathrm{f}_{\mathbf{Y}}(\mathbf{Y}|\hat{\mathbf{x}}(t_1) = \mathbf{x}(t_1)\,, \hat{\mathbf{x}}(t_2) = \mathbf{x}(t_2)\,, \ldots) = \\ \frac{1}{\left(\sigma\sqrt{2\pi}\right)^{n_x n_k}} \exp\left(-\frac{\sum_{i=1}^{n_k}\sum_{j=1}^{n_x}\left(y_j(t_i) - x_j(t_i)\right)^2}{2\sigma^2}\right), \end{gathered} \tag{4.8}$$

where $\sigma$ is a known parameter, $n_x = 3$, and $n_k$ equals the number of time points sampled. If a uniform prior PDF is assumed for $(\hat{p}_1, \hat{p}_2)$, the MAP estimate can be obtained by maximizing the function defined in Equation (4.8) over $\mathbf{p} \in P \subset \mathbb{R}^2$, $\mathbf{X} \in X \subset \mathbb{R}^{n_x \times n_k}$ subject to satisfying the system of ODEs defined in Equation (4.7) at fixed time points $\{t_1, \ldots t_{n_k}\}$. For measurement model given in Equation (4.8), the global optimization problem can be simplified by noting that the objective function has the form:

$$g(\mathbf{p}) = f(\alpha(\mathbf{p})) = \exp(\alpha(\mathbf{p}))\,.$$

where $\alpha$ is a scalar and $\exp(\alpha)$ is a monotonically increasing function of $\alpha$. Hence the optimization problem can be rewritten as

$$\min_{\mathbf{X}\in X, \mathbf{p}\in P} \sum_{i=1}^{n_k}\sum_{j=1}^{n_x}\left(y_j(t_i) - x_j(t_i)\right)^2 \tag{4.9}$$

subject to satisfying the system of ODEs defined in Equation (4.7) at fixed time points $\{t_1, \ldots, t_{n_k}\}$. Many aspects of this parameter estimation problem (including the selection of a suitable experimental design) are discussed in detail by [29]. In particular, it is demonstrated graphically that convexity of the resulting optimization problem depends on the chosen experimental design.

Table 4.1: Simulated Data for Example 4.1.1

| $t$ | $y_1(t)$ | $y_2(t)$ | $y_3(t)$ |
|---|---|---|---|
| 0.0 | 10.000 | 0.000 | 0.000 |
| 0.2 | 3.428 | 4.019 | 1.907 |
| 0.4 | 1.311 | 3.468 | 5.364 |
| 0.6 | 0.458 | 1.949 | 7.593 |
| 0.8 | 0.245 | 1.495 | 8.567 |
| 1.0 | -0.047 | 0.800 | 9.153 |
| 1.2 | -0.171 | 0.042 | 9.651 |
| 1.4 | -0.080 | 0.005 | 9.641 |
| 1.6 | 0.220 | 0.334 | 9.916 |
| 1.8 | 0.476 | 0.109 | 10.023 |
| 2.0 | 0.046 | 0.093 | 10.260 |

Point estimates of parameters from a posterior PDF provide little information about the confidence associated with the estimate. Confidence intervals for an estimate can be obtained from Definition 4.1.2.

**Definition 4.1.2.** [29] Let $\mathrm{f}_{\boldsymbol{\alpha}}(\boldsymbol{\alpha}|\boldsymbol{y}, \mathbf{X})$ be the posterior PDF. The volume $Z \subset \mathbb{R}^{n_\alpha}$ is a highest posterior density (HPD) region of content $1-\gamma$ **iff**

$$\Pr(\boldsymbol{\alpha} \in Z|\boldsymbol{y}, \mathbf{X}) = 1 - \gamma,$$

and $\forall \boldsymbol{\alpha}_1 \in Z$, $\forall \boldsymbol{\alpha}_2 \notin Z$

$$\mathrm{f}_{\boldsymbol{\alpha}}(\boldsymbol{\alpha}_1|\boldsymbol{y}, \mathbf{X}) \geq \mathrm{f}_{\boldsymbol{\alpha}}(\boldsymbol{\alpha}_2|\boldsymbol{y}, \mathbf{X}) .$$

It should be stressed that even confidence intervals based on the HPD region can be misleading if the posterior PDF is multimodal. In this situation, it is desirable to report all parameter values that correspond to high probability regions.

Another common scenario is that it is desired to estimate a subset of the parameters (for example, one may not be interested in the parameters characterizing the measurement model). In this situation, it is more appropriate to make inferences from the marginal PDF for a subset of the parameters, $\hat{\boldsymbol{\theta}}$:

$$\mathrm{f}_{\boldsymbol{\theta}}(\boldsymbol{\theta}|\hat{\mathbf{y}} = \mathbf{y}, \hat{\mathbf{u}} = \mathbf{u}, M) = \int_{\boldsymbol{\sigma}} \mathrm{f}_{\boldsymbol{\theta},\boldsymbol{\sigma}}(\boldsymbol{\theta}, \boldsymbol{\sigma}|\hat{\mathbf{y}} = \mathbf{y}, \hat{\mathbf{u}} = \mathbf{u}, M)\ \mathrm{d}\boldsymbol{\sigma} \tag{4.10}$$

where $\hat{\mathbf{p}} = \left(\hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\sigma}}\right)$. Care must be taken when making inferences from a marginal density if the shape of the joint posterior PDF changes dramatically as the parameters, $\boldsymbol{\sigma}$, are varied. In this situation, the change in shape serves as a warning that the resulting parameter estimate, $\boldsymbol{\theta}^*$, is very sensitive to inaccuracy in the estimate of $\hat{\boldsymbol{\sigma}}$.

Typically, the simplest parameter estimation problem that arises in cell signaling is estimation of steady-state parameters (for example, the estimation of an equilibrium dissociation constant $K_d$). The resulting expectation function is defined in terms of the solution of a set of nonlinear algebraic equations. The MAP parameter estimate from the joint posterior PDF can be obtained from the solution of a nonconvex nonlinear program (nonconvex NLP) shown in Definition 4.1.1 [29]. It is misleading to solve the nonconvex NLP with local optimization tools, since there is a risk that one might miss characterizing values of the parameters that correspond with high probability. In recent years, there have been considerable advances in the technology available to solve these types of problems to global optimality [2, 191]. Algorithms are now available that can in principle estimate up to several hundred parameters [190, 1]. Most of these techniques rely on variants of branch-and-bound algorithms.

More frequently, one is interested in the dynamic behavior of a cell signaling network. As discussed in Chapter 1, some cellular processes are controlled by the time-dependent concentration of key signaling molecules. The goal is to estimate rate constants for a series of complex enzymatic reactions (phosphorylation, dephosphorylation) that regulate the concentration of these key signaling molecules. Typically, time-series data is used to estimate model parameters. For dynamic systems, the expectation function is defined in terms of the solution at fixed times to a set of ODEs. The MAP estimate corresponds to the solution of a nonconvex dynamic embedded optimization problem. These estimation problems are not approximations of variational problems; i.e., the problems are naturally formulated as nonlinear programs.

There are two different approaches for solving these optimization problems to local optimality. The first method is a sequential algorithm. A local nonlinear program (NLP) solver is combined with an numerical integration routine. The integration routine is used to evaluate the states and sensitivities (or adjoints) at a fixed set of model parameter values (for example: [27]). The second approach is a simultaneous algorithm, which requires the approximation of the dynamic system as a set of algebraic equations (for example: [227]). The resulting large-scale NLP can then be solved with standard techniques.

Two methods have been proposed by [77, 78] for solving optimization problems with nonlinear ordinary differential equations embedded to global optimality with finite $\epsilon$ tolerance. The first method is collocation of the ODE followed by application of the $\alpha$BB algorithm [2] to the resulting NLP. Provided a sufficient number of collocation points have been chosen to control the error in the discretization of the ODE, this method will yield a close approximation to the global optimum. However, this approach has the disadvantage that many additional variables are introduced, making the resulting spatial branch-and-bound procedure intractable except for small problems.

Sufficient conditions are available for the existence of derivatives with respect to parameters of the solution of a set of ordinary differential equations [101, 173]. For problems that are twice continuously differentiable, [77] suggest a branch-and-bound strategy ($\beta$BB) which uses a convex relaxation generated in a manner similar to $\alpha$BB. However, analytic expressions for the Hessian of the solution to the ODE with respect to the parameters are generally not available. The solution space is sampled to suggest possible bounds on the elements of the Hessian. Consequently, rigorous bounds on $\beta$ are not determined by their method. i.e. Their implementation does not guarantee finite $\epsilon$ convergence to a global optimum. An advance on this technique that rigorously guarantees the global solution has been proposed [168]. In this technique, rigorous bounds on the elements of the Hessian matrix are derived which can be used to determine a value $\beta$ sufficiently large to guarantee convexity of the lower bounding problem (LBP). An alternative theory for constructing convex lower bounding problems has been developed [203]. These methods look extremely promising for solving kinetic parameter estimation problems.

### 4.1.1 Branch and Bound

A branch-and-bound [80, 214] or branch-and-reduce procedure [191, 224] is at the core of most deterministic global optimization techniques. A generic branch-and-bound algorithm is described in [154]. Let $S \subset \mathbb{R}^n$ be a nonempty compact set, and $v_*$ denote the minimum value of $f(\mathbf{x})$ subject to $\mathbf{x} \in S$. Let $\epsilon > 0$ be the acceptable amount an estimate of $v_*$ can differ from the true minimum. Set $k = 1$ and $S^{1,1} = S$. It is assumed at iteration $k$ there are $k$ regions $S^{k,l}$, each $S^{k,l} \subseteq S$ for $l = 1, \ldots, k$. For each $k$ programs, $A^{k,l}$, $v_*^{k,l}$, is the minimum of $f(\mathbf{x})$ subject to $\mathbf{x} \in S^{k,l}$ and $v_* = \min v_*^{k,l}$ $l = 1 \ldots k$ must be true.

1. Find $\mathbf{x}^{k,l} \in S^{k,l}$, an estimate of a solution point of $A^{k,l}$, for $l = 1, \ldots, k$.

2. Find $v^{k,l}$, a lower bound on $v_*^{k,l}$ for $l = 1, \ldots, k$

Let $v^k = \min\left(v^{k,l}\right)$ $l = 1, \ldots k$. If $f\left(\mathbf{x}^{k,l}\right) \leq v^k + \epsilon$ then the algorithm terminates, otherwise $k = k + 1$. It can be shown that under certain conditions, the algorithm will terminate in a finite number of steps [116]. Steps 1 and 2 can be realized by many different methods.

The lower bounding problem (Step 2) is generated by constructing a convex relaxation of the original problem via direct analysis of the functions participating in the objective function and embedded system. Convexity of a set and function is defined as follows:

**Definition 4.1.3.** The set $S$ is convex if for every $\lambda \in [0, 1]$ the point $\mathbf{x} = \lambda\mathbf{x}_1 + (1 - \lambda)\mathbf{x}_2$ lies in the set $S$ for all $\mathbf{x}_1, \mathbf{x}_2 \in S$. The function $f : S \to \mathbb{R}$ is convex on the set $S$ if:

1. the set $S$ is convex, and,

2. for every $\lambda \in (0, 1)$, $\mathbf{x}_1 \in S$ and $\mathbf{x}_2 \in S$ the following inequality holds:

$$f(\lambda\mathbf{x}_1 + (1 - \lambda)\mathbf{x}_2) \leq \lambda f(\mathbf{x}_1) + (1 - \lambda) f(\mathbf{x}_2).$$

The function $f$ is strictly convex if the inequality in 2 holds strictly.

For example, the convex envelope of a bilinear term on any rectangle in $\mathbb{R}^2$ is given by Equation (4.11) [5].

$$\begin{aligned} w &= \max(u, v) \\ u &= x_1^L \cdot x_2 + x_2^L \cdot x_1 - x_1^L \cdot x_2^L \\ v &= x_1^U \cdot x_2 + x_2^U \cdot x_1 - x_1^U \cdot x_2^U \end{aligned} \tag{4.11}$$

Any local optimum that is found for the convex relaxation is guaranteed to be the global optimum for the convex relaxation by Theorem 4.1.1. Hence, it is a valid lower bound on the original optimization problem.

**Theorem 4.1.1.** *Let $S \subset \mathbb{R}^n$ be a nonempty convex set, and let $f : S \to \mathbb{R}$ be convex on $S$. Consider the problem to minimize $f(\mathbf{x})$ subject to $\mathbf{x} \in S$. Suppose that $\mathbf{x}^* \in S$ is a local optimal solution to the problem, then, $\mathbf{x}^*$ is a global optimal solution.*

*Proof.* See [18]. □

Polynomial time, globally convergent algorithms, based on the work of [87], exist for most classes of smooth, convex NLPs [161, 235, 9]. Cutting plane methods also exist for non-smooth convex NLPs [202, 160]. An upper bounding problem (Step 1) can be constructed by solving the original embedded optimization problem to local optimality. In fact, any feasible point is a valid upper bound on the objective function.

### 4.1.2 Convexification of Nonlinear Programs

Many techniques for constructing convex relaxations of nonlinear programs depend either implicitly or explicitly on a composition theorem due to [153, 154] (shown in Theorem 4.1.2). For purely algebraic problems, the theorem is typically realized in a computationally efficient manner due to the methods of [212, 94].

**Theorem 4.1.2.** *[153, 154] Let $f(x(\mathbf{p})) : P \to \mathbb{R}$, where $P \subset \mathbb{R}^{n_p}$ is a convex set and $f(x)$ is a univariate function of $x$. Let functions $c(\mathbf{p})$ and $C(\mathbf{p})$ obey the inequality:*

$$c(\mathbf{p}) \le x(\mathbf{p}) \le C(\mathbf{p}), \quad \forall \mathbf{p} \in P,$$

*where $c(\mathbf{p})$ is convex on the set $P$ and $C(\mathbf{p})$ is concave on the set $P$. Let $x^L$ and $x^U$ be valid bounds on the state*

$$x^L \leq x(\mathbf{p}) \leq x^U,$$

*and let the function $e(\cdot)$ be a convex underestimate of the function $f$ on the interval $\left[x^L, x^U\right] \in \mathbb{R}$. Let $z_{\min}$ be defined as,*

$$z_{\min} = \arg \inf_{x^L \leq x \leq x^U} e(z)$$

*Then the lower bounding convex function on the set $\mathbf{p} \in P \cap \left\{\mathbf{p} | x^L \leq x(\mathbf{p}) \leq x^U\right\}$ is*

$$e(\text{mid}\{c(\mathbf{p}), C(\mathbf{p}), z_{\min}\})$$

*where the* mid$\{\cdot\}$ *function takes the middle value of the three values.*

*Proof.* See [154]. $\square$

Theorem 4.1.2 can be used construct a convex underestimating function of the posterior PDF, once convex underestimates $c_i(\mathbf{p})$ and concave overestimates $C_i(\mathbf{p})$ of the states, $x_i(\mathbf{p})$, and state bounds, $x_i^L$ and $x_i^U$, can be obtained.

**Example 4.1.2.** Use the McCormick relaxation shown in Theorem 4.1.2 to express the convex relaxation of the MAP objective function defined in Equation (4.9), Example 4.1.1.

From the definition of convexity, the function $h(\mathbf{x}) = f(\mathbf{x}) + g(\mathbf{x})$ is convex on the set $X$ if the functions $f(\mathbf{x})$ and $g(\mathbf{x})$ are convex on the set $X$. Hence, the problem of constructing a convex underestimate of the MAP objective function reduces to constructing convex underestimates for each of the terms in the sum defined in Equation (4.9). The function

$$f(x) = (x - a)^2,$$

is already convex and a minimum is achieved at $z_{\min} = a$. Therefore, a convex underestimate of the MAP objective function is given by:

$$\sum_{i=1}^{n_k} \sum_{j=1}^{n_x} \left(y_j(t_i) - \text{mid}\{c_j(t_i, \mathbf{p}), y_j(t_i), C_j(t_i, \mathbf{p})\}\right)^2, \tag{4.12}$$

where $c_j(t_i, \mathbf{p})$ are convex functions and $C_j(t_i, \mathbf{p})$ are concave functions that satisify:

$$c_j(t_i, \mathbf{p}) \leq x_j(t_i, \mathbf{p}) \leq C_j(t_i, \mathbf{p}), \quad \forall \mathbf{p} \in P.$$

Hence, it is necessary to construct convex functions that underestimate the solution of a system of ODEs at fixed time and concave functions that overestimate the solution of a system of ODEs at fixed time.

## 4.1.3 State Bounds for ODEs

It is necessary to bound the solution of an ODE, $\mathbf{x}(t, \mathbf{p})$, for deterministic global optimization techniques. State bounds are two functions $\mathbf{x}^L(t)$ and $\mathbf{x}^U(t)$ which satisfy the inequality:

$$\mathbf{x}^L(t) \leq \mathbf{x}(t, \mathbf{p}) \leq \mathbf{x}^U(t), \quad \forall \mathbf{p} \in P \subset \mathbb{R}^{n_p},$$

where the vector inequalites should be interpreted as holding componentwise; i.e., the functions $\mathbf{x}^L(t)$ and $\mathbf{x}^U(t)$ bound the solution of a system of ODEs for all possible parameter values $\mathbf{p} \in P$. Tight state bounds are necessary to generate convex relaxations of the fixed time solution of ODEs. Exact state bounds can be determined for a system of ODEs that is linear in the parameters [204], and linear ODEs [107, 108]. Methods to generate state bounds of a nonlinear ODE are generally based on interval Taylor methods [136, 158], interval Hermite-Obreschkoff methods [159] or differential inequalities [19, 236]. A particular problem is a phenomenon known as, "The Wrapping Effect", which causes the estimated bounds to inflate exponentially. Methods to overcome this are discussed in [220, 159]. A simple but often effective method to generate bounds due to [106] relies on differential inequalities described in Theorem 4.1.3.

**Theorem 4.1.3.** *(An extension due to [106] of a Theorem by [236].) Let* $\mathbf{x}(t, \mathbf{p})$ *be the solution of*

$$\mathbf{x}' = \mathbf{f}(\mathbf{x}, \mathbf{p}), \tag{4.13}$$

*where*

$$\begin{aligned} \mathbf{x}^L(0) \le \mathbf{x}(0) \le \mathbf{x}^U(0) \\ \mathbf{p}^L \le \mathbf{p} \le \mathbf{p}^U, \end{aligned}$$

*for some known vectors* $\mathbf{x}^L(0)$, $\mathbf{x}^U(0)$, $\mathbf{p}^L$, *and* $\mathbf{p}^U$. *Assume for all* $\mathbf{p}^L \le \mathbf{p} \le \mathbf{p}^U$, $\mathbf{f}$ *satisfies the one-sided Lipschitz condition:*

$$f_i(\mathbf{x}, \mathbf{p}) - f_i(\mathbf{z}, \mathbf{p}) \le \sum_j \lambda_{ij}(t) \left| x_j - z_j \right|, \quad \textit{when } x_i \ge z_i,$$

*where the* $\lambda_{ij}(t)$ *are continuous positive functions on* $0 \le t \le T$. *Let* $\mathbf{x}^L(t)$ *and* $\mathbf{x}^U(t)$ *satisfy*

$$\begin{aligned} x_i^{L\prime} &\le \min f_i(\mathbf{z}, \mathbf{p}), \quad \textit{where } \mathbf{p}^L \le \mathbf{p} \le \mathbf{p}^U, \mathbf{x}^L \le \mathbf{z} \le \mathbf{x}^U, z_i = x_i^L \\ x_i^{U\prime} &\ge \max f_i(\mathbf{z}, \mathbf{p}), \quad \textit{where } \mathbf{p}^L \le \mathbf{p} \le \mathbf{p}^U, \mathbf{x}^L \le \mathbf{z} \le \mathbf{x}^U, z_i = x_i^U. \end{aligned}$$

*Then* $\mathbf{x}^L(t) \le \mathbf{x}(t, \mathbf{p}) \le \mathbf{x}^U(t)$ *for all* $0 \le t \le T$.

The work of [106] advocates using interval analysis [158] to provide bounds on the derivatives, $x_i^{L\prime}$ and $x_i^{U\prime}$. This bounding technique is demonstrated in Example 4.1.3.

**Example 4.1.3.** Consider the kinetic model described in Example 4.1.1. Generate state bounds using interval evaluation of differential inequalities described in Theorem 4.1.3. Assume that $x_i^L(0) = x_i(0) = x_i^U(0)$, $p_1^L \le p_1 \le p_1^U$, and $p_2^L \le p_2 \le p_2^U$.

Consider the lower bound of the first state, $x_1^L$. According to Theorem 4.1.3, it is necessary to construct a lower bound of

$$\min\left(-p_1 z_1\right),$$

subject to $z_1 = x_1^L$, and $p_1^L \leq p_1 \leq p_1^U$ and set the lower bound equal to the derivative, $x_1^{L\prime}$. Using interval arithmetic, this evaluates to

$$x_1^{L\prime} = -\max\left(p_1^L x_1^L, p_1^U x_1^L\right). \tag{4.14}$$

Likewise, the lower bound of the second state, $x_2^L$, can be evaluated by setting the lower bound of

$$\min\left(p_1 z_1 - p_2 z_2\right)$$

subject to $x_1^L \leq z_1 \leq x_1^U$, $p_1^L \leq p_1 \leq p_1^U$, $p_2^L \leq p_2 \leq p_2^U$, and $z_2 = x_2^L$, equal to the derivative, $x_2^{L\prime}$. Again, using interval arithmetic this evaluates to

$$x_2^{L\prime} = \min\left(p_1^L x_1^L, p_1^L x_1^U, p_1^U x_1^L, p_1^U x_1^U\right) - \max\left(p_2^L x_2^L, p_2^U x_2^L\right). \tag{4.15}$$

The remaining lower and upper bounds can be evaluated analogously:

$$\begin{aligned}
x_3^{L\prime} &= \min\left(p_2^L x_2^L, p_2^L x_2^U, p_2^U x_2^L, p_2^U x_2^U\right) && (4.16)\\
x_1^{U\prime} &= -\min\left(p_1^L x_1^U, p_1^U x_1^U\right) && (4.17)\\
x_2^{U\prime} &= \max\left(p_1^L x_1^L, p_1^L x_1^U, p_1^U x_1^L, p_1^U x_1^U\right) - \min\left(p_2^L x_2^U, p_2^U x_2^U\right) && (4.18)\\
x_3^{U\prime} &= \max\left(p_2^L x_2^L, p_2^L x_2^U, p_2^U x_2^L, p_2^U x_2^U\right). && (4.19)
\end{aligned}$$

Care must taken when evaluating the ODE system corresponding to Equations (4.14)–(4.19) since the min and max functions can produce hidden discontinuities [229]. A robust algorithm to detect state events has been proposed by [170] and implemented in ABACUSS II. The ODE system shown in Equations (4.14)–(4.19) was converted into an ABACUSS II simulation (§ B.5, Appendix B) for $1 \leq p_1 \leq 3$, $1 \leq p_2 \leq 3$, and $x_1(0) = 10$, $x_2(0) = x_3(0) = 0$. The simulation results are shown in Figure 4-1. A disadvantage of this bounding method is that physical bounds on the states are not taken into account. For example, the concentration of reacting species must always be nonnegative $x_i(t) \geq 0$ for all $0 \leq t \leq T$. Furthermore, by conservation of mass, it is easy to argue that $x_1(t) \leq x_1(0)$, $x_2(t) \leq x_1(0) + x_2(0)$, and $x_3(t) \leq$

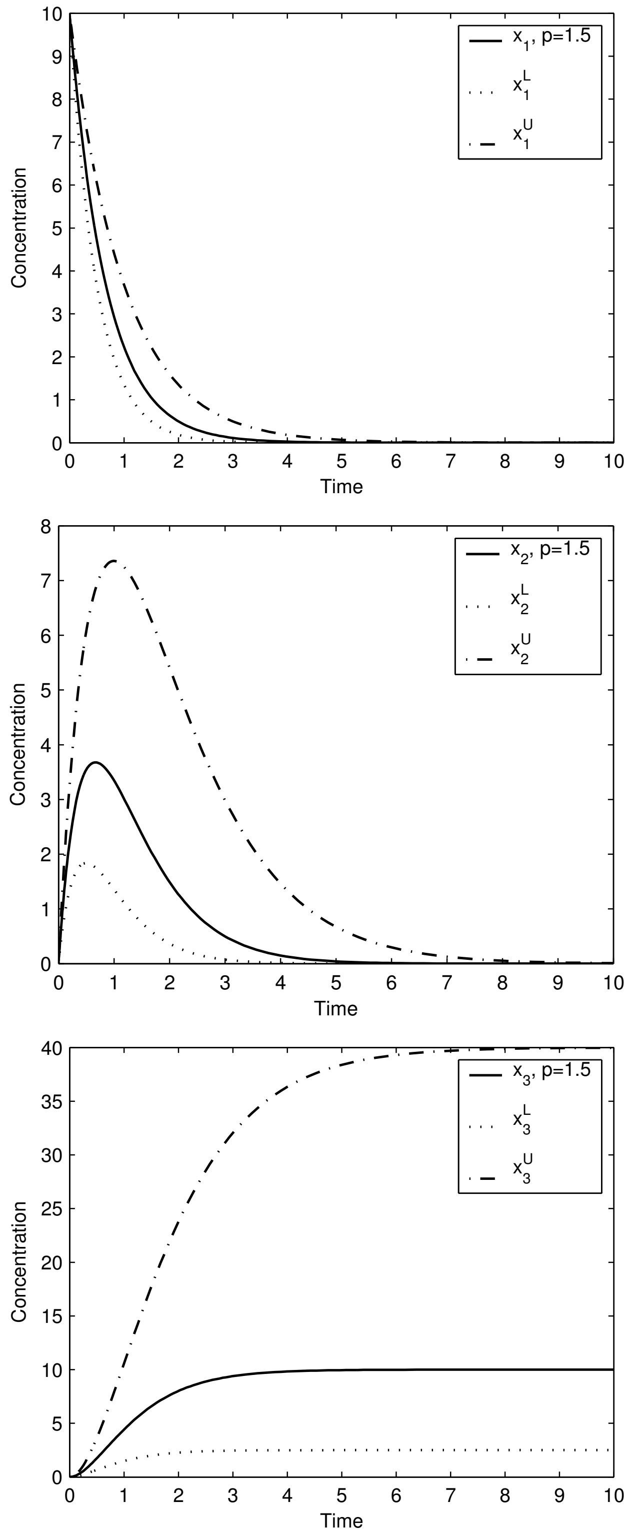


Figure 4-1: Simulation of state bounds for chemical kinetics

$x_1(0) + x_2(0) + x_3(0)$. A method based on differential inequalities that takes into account physical bounds has been proposed by [203].

### 4.1.4 Convexification of ODEs

In principle, the state bounds derived in § 4.1.3 can be used to construct a lower bound for the objective function of the MAP estimate. However, the bound on the objective function resulting from the state bounds may not be very tight. A tighter bound on the value of the objective function may often be achieved by solving a convex optimization problem constructed from the original equations. This can be achieved using Theorem 4.1.2 due to [154]. To realize this the composition in this Theorem it is necessary to generate convex functions that underestimate the solution to a system of ODEs at fixed time and concave functions that overestimate the solution to a system of ODEs at fixed time. The convex underestimate, $\mathbf{c}(\underline{t}, \mathbf{p})$, and concave overestimate, $\mathbf{C}(\underline{t}, \mathbf{p})$, must satisfy the inequality:

$$\mathbf{c}(\underline{t}, \mathbf{p}) \leq \mathbf{x}(\underline{t}, \mathbf{p}) \leq \mathbf{C}(\underline{t}, \mathbf{p}) \,,$$

for all $\mathbf{p} \in P$ and each fixed $\underline{t} \in [0, T]$. A method to construct the functions $\mathbf{c}(\underline{t}, \mathbf{p})$ and $\mathbf{C}(\underline{t}, \mathbf{p})$ has been proposed by [204, 203]. The functions are obtained by solving a system of ODEs:

$$\begin{aligned} c_i' &= u_i(\mathbf{x}^*(t)\,, \mathbf{p}^*) + \left.\frac{\partial u_i}{\partial x_i}\right|_{\mathbf{x}^*(t),\mathbf{p}^*} (c_i - x_i^*(t)) \\ &\quad + \sum_{j \neq i} \left[ \min\left\{ c_j \left.\frac{\partial u_i}{\partial x_j}\right|_{\mathbf{x}^*(t),\mathbf{p}^*}, C_j \left.\frac{\partial u_i}{\partial x_j}\right|_{\mathbf{x}^*(t),\mathbf{p}^*} \right\} - x_j^*(t) \left.\frac{\partial u_i}{\partial x_j}\right|_{\mathbf{x}^*(t),\mathbf{p}^*} \right] \\ &\quad + \sum_{j=1}^{n_p} \left(p_j - p_j^*\right) \left.\frac{\partial u_i}{\partial p_j}\right|_{\mathbf{x}^*(t),\mathbf{p}^*}, \end{aligned} \tag{4.20}$$

and,

$$C_i' = o_i(\mathbf{x}^*(t)\,, \mathbf{p}^*) + \left.\frac{\partial o_i}{\partial x_i}\right|_{\mathbf{x}^*(t),\mathbf{p}^*} (C_i - x_i^*(t)) \tag{4.21}$$

$$+\sum_{j\neq i}\left[\max\left\{c_j\,\frac{\partial o_i}{\partial x_j}\bigg|_{\mathbf{x}^*(t),\mathbf{p}^*},C_j\,\frac{\partial o_i}{\partial x_j}\bigg|_{\mathbf{x}^*(t),\mathbf{p}^*}\right\}-x_j^*(t)\,\frac{\partial o_i}{\partial x_j}\bigg|_{\mathbf{x}^*(t),\mathbf{p}^*}\right]$$
$$+\sum_{j=1}^{n_p}\left(p_j-p_j^*\right)\frac{\partial o_i}{\partial p_j}\bigg|_{\mathbf{x}^*(t),\mathbf{p}^*},$$

where $\mathbf{p}^*\in\left[\mathbf{p}^L,\mathbf{p}^U\right]$ and $x^*(t)$ is any function that lies in the set $\mathcal{X}(t)$ generated from the state bounds. The function $u_i(\mathbf{p})$ is a convex underestimate of $f_i(\mathbf{p})$ on the set $\mathcal{X}(t)\times P$ for each $t\in[0,T]$, where $f_i$ is the $i^{th}$ component of the right hand side of Equation (4.13). Likewise, the function $o_i(\mathbf{p})$ is a concave overestimate of $f_i(\mathbf{p})$ on the set $\mathcal{X}(t)\times P$ for each $t\in[0,T]$. The convex underestimates and concave overestimates can be obtained automatically through symbolic manipulation of the algebraic functions $f_i(\mathbf{x},\mathbf{p})$ using standard techniques [212, 94]. A code that automatically constructs a simulation of the necessary state-bounding ODE together with the convex lower bounding ODE and concave upper bounding ODE from the original system of ODEs is available [203].

**Example 4.1.4.** Plot $c_i(\underline{t},\mathbf{p})$ and $C_i(\underline{t},\mathbf{p})$ for each of the states corresponding to the system of ODEs defined in Equation (4.7), Example 4.1.1. Use the McCormick relaxation of the MAP objective function shown in Equation (4.12) together with the measurements in Table 4.1 to generate a plot of the corresponding convex underestimate of the objective function.

The convex relaxation of the objective function was generated on the set $\mathbf{p}\in[1,2]\times[1,2]$ around a reference trajectory obtained from the solution of Equations (4.22)–(4.24).

$$x_1^{*\prime} = -p_1^*x_1^* \tag{4.22}$$
$$x_2^{*\prime} = p_1^*x_1-p_2^*x_2^* \tag{4.23}$$
$$x_3^{*\prime} = p_2^*x_2^* \tag{4.24}$$

The initial condition was $\mathbf{x}^*(0)=(10,0,0)$ and $\mathbf{p}^*=(1.3,1.7)$. The ODEs for the convex relaxation of the states is given by the following system of Equations:

```
IF $\left(x_1^L p_1^* + p_1^U x_1^* - x_1^L p_1^U < p_1^L x_1^* + x_1^U p_1^* - p_1^L x_1^U\right)$ THEN
        $c_1' = -x_1^L p_1^* + p_1^U x_1^* - x_1^L p_1^U - p_1^U (c_1 - x_1^*) - x_1^L (p_1 - p_1^*)$
ELSE
        $c_1' = -p_1^L x_1^* + x_1^U p_1^* - p_1^L x_1^U - p_1^L (c_1 - x_1^*) - x_1^U (p_1 - p_1^*)$
ENDIF
IF $\left(x_1^U p_1^* + p_1^U x_1^* - p_1^U x_1^U > x_1^L p_1^* + p_1^L x_1^* - p_1^L x_1^L\right)$ THEN
        IF $\left(x_2^L p_2^* + p_2^U x_2^* - x_2^L p_2^U < p_2^L x_2^* + x_2^U p_2^* - p_2^L x_2^U\right)$ THEN
  $c_2' = \left(x_1^U p_1^* + p_1^U x_1^* - p_1^U x_1^U\right) - \left(x_2^L p_2^* + p_2^U x_2^* - x_2^L p_2^U\right) + \min\left(c_1 p_1^L, C_1 p_1^L\right) - p_1^L x_1^*$
        $- p_2^U (c_2 - x_2^*) + x_1^U (p_1 - p_1^*) - x_2^L (p_2 - p_2^*)$
        ELSE
  $c_2' = \left(x_1^U p_1^* + p_1^U x_1^* - p_1^U x_1^U\right) - \left(p_2^L x_2^* + x_2^U p_2^* - p_2^L x_2^U\right) + \min\left(c_1 p_1^U, C_1 p_1^U\right) - p_1^U x_1^*$
        $- p_2^L (c_2 - x_2^*) + x_1^U (p_1 - p_1^*) - x_2^U (p_2 - p_2^*)$
        ENDIF
ELSE
        IF $\left(x_2^L p_2^* + p_2^U x_2^* - x_2^L p_2^U < p_2^L x_2^* + x_2^U p_2^* - p_2^L x_2^U\right)$ THEN
  $c_2' = \left(x_1^L p_1^* + p_1^L x_1^* - p_1^L x_1^L\right) - \left(x_2^L p_2^* + p_2^U x_2^* - x_2^L p_2^U\right) + \min\left(c_1 p_1^U, C_1 p_1^U\right) - p_1^U x_1^*$
        $- p_2^U (c_2 - x_2^*) + x_1^L (p_1 - p_1^*) - x_2^L (p_2 - p_2^*)$
        ELSE
  $c_2' = \left(x_1^L p_1^* + p_1^L x_1^* - p_1^L x_1^L\right) - \left(p_2^L x_2^* + x_2^U p_2^* - p_2^L x_2^U\right) + \min\left(c_1 p_1^L, C_1 p_1^L\right) - p_1^L x_1^*$
        $- p_2^L (c_2 - x_2^*) + x_1^L (p_1 - p_1^*) - x_2^U (p_2 - p_2^*)$
        ENDIF
```

```
ENDIF
IF $\left(x_2^U p_2^* + p_2^U x_2^* - p_2^U x_2^U > x_2^L p_2^* + p_2^L x_2^* - p_2^L x_2^L\right)$ THEN
    $c_3' = x_2^U p_2^* + p_2^U x_2^* - p_2^U x_2^U + \min\left(p_2^U c_2, p_2^U C_2\right) - p_2^U x_2^* + x_2^U (p_2 - p_2^*)$
ELSE
    $c_3' = x_2^L p_2^* + p_2^L x_2^* - p_2^L x_2^L + \min\left(p_2^L c_2, p_2^L c_2\right) - p_2^L x_2^* + x_2^L (p_2 - p_2^*)$
ENDIF
```

The initial condition $\mathbf{c}(0) = (10, 0, 0)$ was used. The concave overestimates of the states are obtained in an analogous fashion:

```
IF $\left(x_1^U p_1^* + p_1^U x_1^* - p_1^U x_1^U > x_1^L p_1^* + p_1^L x_1^* - p_1^L x_1^L\right)$ THEN
    $C_1' = -\left(x_1^U p_1^* + p_1^U x_1^* - p_1^U x_1^U\right) - p_1^U (C_1 - x_1^*) - x_1^U (p_1 - p_1^*)$
ELSE
    $C_1' = -\left(x_1^L p_1^* + p_1^L x_1^* - p_1^L x_1^L\right) - p_1^L (C_1 - x_1^*) - x_1^L (p_1 - p_1^*)$
ENDIF
IF $\left(x_1^L p_1^* + p_1^U x_1^* - x_1^L p_1^U < p_1^L x_1^* + x_1^U p_1^* - p_1^L x_1^U\right)$ THEN
    IF $\left(x_2^U p_2^* + p_2^U x_2^* - p_2^U x_2^U > x_2^L p_2^* + p_2^L x_2^* - p_2^L x_2^L\right)$ THEN
        $C_2' = \left(x_1^L p_1^* + p_1^U x_1^* - x_1^L p_1^U\right) - \left(x_2^U p_2^* + p_2^U x_2^* - p_2^U x_2^U\right) + \max\left(p_1^U c_1, p_1^U C_1\right) - p_1^U x_1^* - p_2^U (C_2 - x_2^*) + x_1^L (p_1 - p_1^*) - x_2^U (p_2 - p_2^*)$
    ELSE
        $C_2' = \left(x_1^L p_1^* + p_1^U x_1^* - x_1^L p_1^U\right) - \left(x_2^L p_2^* + p_2^L x_2^* - p_2^L x_2^L\right) + \max\left(p_1^U c_1, p_1^U C_1\right) - p_1^U x_1^* - p_2^L (C_2 - x_2^*) + x_1^L (p_1 - p_1^*) - x_2^L (p_2 - p_2^*)$
    ENDIF
```

ELSE

IF $\left(x_2^U p_2^* + p_2^U x_2^* - p_2^U x_2^U > x_2^L p_2^* + p_2^L x_2^* - p_2^L x_2^L\right)$ THEN

$$\begin{aligned} C_2' &= \left(x_1^L p_1^* + p_1^U x_1^* - x_1^L p_1^U\right) - \left(x_2^U p_2^* + p_2^U x_2^* - p_2^U x_2^U\right) + \max\left(p_1^L c_1, p_1^L C_1\right) - p_1^L x_1^* \\ &\quad - p_2^U \left(C_2 - x_2^*\right) + x_1^U \left(p_1 - p_1^*\right) - x_2^U \left(p_2 - p_2^*\right) \end{aligned}$$

ELSE

$$\begin{aligned} C_2' &= \left(x_1^L p_1^* + p_1^U x_1^* - x_1^L p_1^U\right) - \left(x_2^L p_2^* + p_2^L x_2^* - p_2^L x_2^L\right) + \max\left(p_1^L c_1, p_1^L C_1\right) - p_1^L x_1^* \\ &\quad - p_2^L \left(C_2 - x_2^*\right) + x_1^U \left(p_1 - p_1^*\right) - x_2^L \left(p_2 - p_2^*\right) \end{aligned}$$

ENDIF

ENDIF

IF $\left(x_2^L p_2^* + p_2^U x_2^* - x_2^L p_2^U < p_2^L x_2^* + x_2^U p_2^* - p_2^L x_2^U\right)$ THEN

$$C_3' = \left(x_2^L p_2^* + p_2^U x_2^* - x_2^L p_2^U\right) + \max\left(p_2^L c_2, p_2^L C_2\right) - p_2^L x_2^* + x_2^L \left(p_2 - p_2^*\right)$$

ELSE

$$C_3' = \left(p_2^L x_2^* + x_2^U p_2^* - p_2^L x_2^U\right) + \max\left(p_2^U c_2, p_2^U C_2\right) - p_2^U x_2^* + x_2^U \left(p_2 - p_2^*\right)$$

ENDIF

The initial condition $\mathbf{C}(0) = (10, 0, 0)$ was used. The function $f : x \rightarrow (x-a)^2$ is convex and is minimized by $f(a)$. Hence, the convex relaxation of objective function can be obtained by application of Theorem 4.1.2:

$$\sum_{i=1}^{n_k} \sum_{j}^{n_x} \left(y_j(t_i) - \text{midfun}(c_j(t_i)\,, C_j(t_i)\,, y_j(t_i))\right)^2.$$

An automatically generated ABACUSS II simulation was generated by symbolic analysis of the original ODE system [203]. The generated code is shown in § B.6, Appendix B. Convex underestimates and concave overestimates of the states $\mathbf{x}(\underline{t}, \mathbf{p})$ at fixed time $\underline{t}$ are shown in Figure 4-2. The resulting convex relaxation of the MAP objective function together with the objective function is shown in Figure 4-3.

## 4.2 Model Selection

Model selection is a more complex problem that occurs routinely when analyzing cell signaling data. Often one wants to test several competing hypotheses (for example: species $A$ interacts with species $B$ versus species $A$ does not interact with species $B$). Therefore, it is no longer valid to assume the assigned models are accurate. Mathematically, one wishes to select the most probable model from a set of possible models $\{M^1, M^2, \ldots M^m\}$. There is a large amount of common computational infrastructure between model selection and parameter estimation since parameter estimation is a form of continuous model selection (much like an NLP relaxation of an integer problem). Techniques for Bayesian model selection have been developed and expounded by many authors [32, 206, 221], but all of these authors make many simplifying assumptions to make the problem numerically tractable.

To analyze the different criteria for model selection it is necesary to define the minimum of the sum of the square of the residuals for the $j^{th}$ model as:

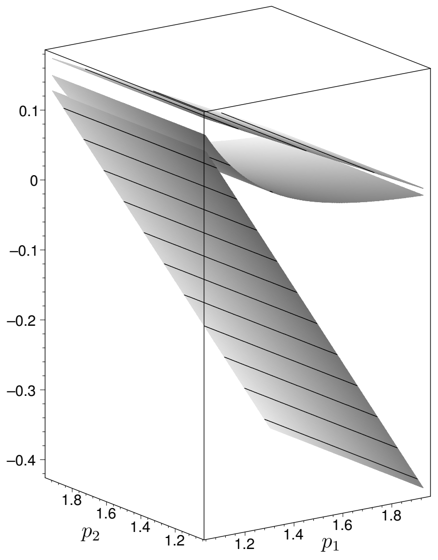
$$\hat{S}^j = \min_{\mathbf{p}\in\mathbb{R}^{n_p}} \sum_{i=1}^{n_k} \left(\mathbf{y}_i - \mathbf{g}^j(\mathbf{u}_i, \mathbf{p})\right)^T \mathbf{\Sigma}^{-1} \left(\mathbf{y}_i - \mathbf{g}^j(\mathbf{u}_i, \mathbf{p})\right),$$

where $\mathbf{\Sigma}$ is the standard covariance matrix. From the definition of the covariance matrix, the expected value of $\hat{S}^j$ for the correct model, $\mathbf{g}^j$, is

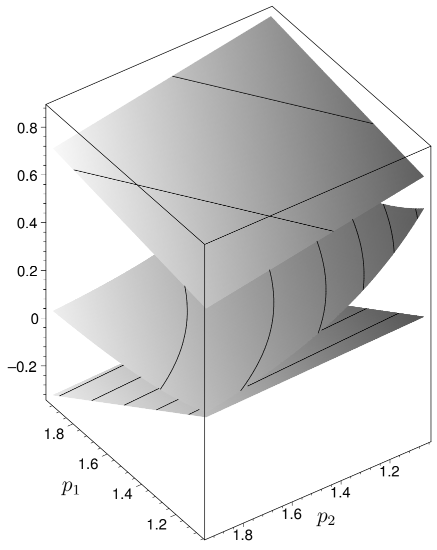
$$\mathrm{E}\left(\hat{S}^j\right) = n_x n_k, \tag{4.25}$$

or the total number of measurements made. Several approximate criteria have been developed for the purpose of model selection and experimental design:

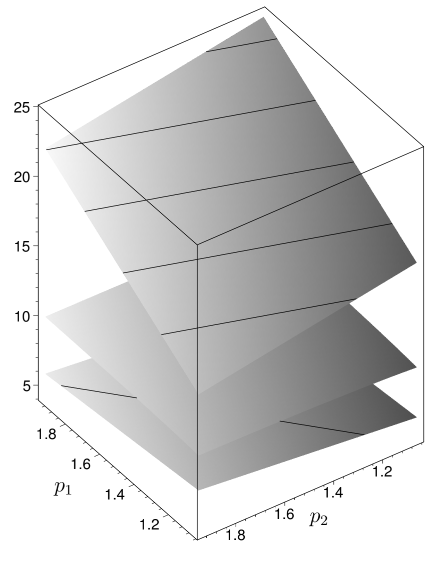
1. Akaike Information Criterion, $AIC$, for known variances in the measurement



(a) State 1



(b) State 2



(c) State 3

Figure 4-2: Convex underestimate and concave overestimate for states at $t = 4$

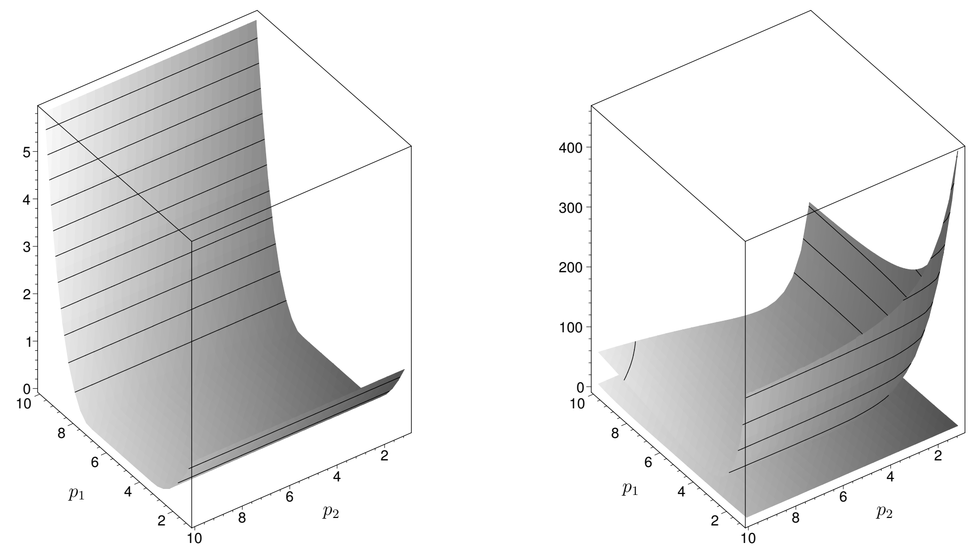


Figure 4-3: Convex underestimate (left) combined with objective function (right)

model, $\hat{\mathbf{p}}_2 = \mathbf{p}_2$, given in Equation (4.26) [4].

$$AIC = \hat{S}^j + n_p^j \tag{4.26}$$

$\hat{S}^j$ is the minimum of the sum of the square of the residuals over all data, and $n_p^j$ is the number of independent parameters in model $j$. It can be seen from Equation (4.25) that the contribution to the $AIC$ from the penalty term $n_p^j$ diminishes as more measurements are made.

2. A sequential experimental design criterion for a single output model defined by Equation (4.27) [117]:

$$\mathbf{u} = \arg \left[ \max_{\mathbf{u} \in \mathbb{R}^{n_u}} \left|\left| \tilde{x}^1 - \tilde{x}^2 \right|\right| \right]_2 \tag{4.27}$$

where after $\{y_1, \ldots, y_{n_k}\}$ measurements, $\tilde{x}^1$ is an estimate of the state based on model 1, given an estimate of the parameters $\mathbf{p}^1$, and $\tilde{x}^2$ is an estimate of the state of model based on model 2 given an estimate of the parameters $\mathbf{p}^2$.

This experimental design criterion implies that determining the most probable model does not depend on the number of parameters in the model. A good explanation of why one would expect a penalty term according to the number of parameters in a model is given in [103].

3. A criterion for a single output model based on the posterior probability of the model being true [28]:

$$\mathrm{f}_{\mathrm{M}^j}\left(\mathrm{M}^j|\mathbf{y},\mathbf{U}\right) = \frac{\mathrm{f}_{\mathrm{M}^j}\left(\mathrm{M}^j|y_1,\ldots,y_{n_k-1},\mathbf{U}\right)\mathrm{f}_y\left(y_{n_k}|\mathrm{M}^j,\mathbf{U}\right)}{\sum_{j=1}^{m}\mathrm{f}_{\mathrm{M}^j}\left(\mathrm{M}^j|y_1,\ldots,y_{n_k-1},\mathbf{U}\right)\mathrm{f}_y\left(y_{n_k}|\mathrm{M}^j,\mathbf{U}\right)} \tag{4.28}$$

where,

$$\mathrm{f}_y\left(y_{n_k}|\mathrm{M}^j,\mathbf{U}\right) = \frac{1}{\sqrt{2\pi\left(\sigma+\sigma^j\right)}}\exp\left(-\frac{1}{2\left(\sigma^2+\sigma^j\right)}\left(y_{n_k}-\tilde{y}^j_{n_k}\right)^2\right).$$

$\mathrm{f}_{\mathrm{M}^j}(\mathrm{M}^j|\mathbf{y},\mathbf{U})$ is the probability of model $j$ given a set of measurements $\mathbf{y}$ at the input conditions $\mathbf{U}$, $\mathrm{f}_y(y_{n_k}|\mathrm{M}^j,\mathbf{U})$ is the probability of making a measurement $y_{n_k}$ given model $j$, $\tilde{y}^j_{n_k}$ is the predicted value of the measurement based on the previous observations and model $j$. $\sigma$ is the variance of the measurement model:

$$\mathrm{f}_y\left(y_{n_k}|x,\mathrm{M}^j,\mathbf{U}\right) = \frac{1}{\sigma\sqrt{2\pi}}\exp\left(-\frac{\left(y_{n_k}-x\right)^2}{2\sigma^2}\right),$$

and $\sigma^j$ is the variance of the estimate $\tilde{y}^j_{n_k}$. Similarly, the estimates of the posterior PDF given in [28] do not have any penalty terms associated with the number of parameters [221].

4. A criterion based upon Bayesian arguments for model selection defined by Equation (4.29) [221]:

$$\mathrm{f}_{\mathrm{M}^j}\left(\mathrm{M}^j|\hat{\mathbf{Y}}=\mathbf{Y},\hat{\mathbf{\Sigma}}=\mathbf{\Sigma},\mathbf{U}\right) \propto 2^{-\frac{n_p^j}{2}}\exp\left(-\frac{\hat{S}^j}{2}\right)\pi_{\mathrm{M}^j}\left(\mathrm{M}^j\right) \tag{4.29}$$

where $\mathrm{f}_{\mathrm{M}^j}\left(\mathrm{M}^j|\hat{\mathbf{Y}}=\mathbf{Y},\hat{\mathbf{\Sigma}}=\mathbf{\Sigma},\mathbf{U}\right)$ is the probability of model $j$ with $n_p^j$ independent parameters, given data, $\mathbf{Y}$, and covariance matrix, $\mathbf{\Sigma}$, $\hat{S}^j$ is the minimum of the sum of the square of the residuals over all data, and $\pi_{\mathrm{M}^j}(\mathrm{M}^j)$ is the prior probability assignment for model $j$.

5. Criteria based on Maximum entropy, that account for model structural uncertainty by assuming the parameters, $\hat{\boldsymbol{\theta}}$, are distributed (as opposed to the probability associated with the parameter) [38, 39, 218]. These criteria are of little use if the information required is the model structure. Furthermore, it is implicitly assumed that some integral constraint is perfectly satisfied by potentially noisy data.

However, there are considerable conceptual difficulties with each of these model selection criteria. With an increase in the number of parameters, there is a corresponding increase in the number of potential models (hypothesis space), and a corresponding decrease in the certainty that any one of those models is correct. Therefore, models with many parameters are less likely until there are some data to support them. Both criteria 1 and 4 and provide methods for model selection with a criterion that includes a penalty term dependent on the number of parameters and the sum of the square of the residuals. However, it is easy to produce an example model that despite having a few fitted parameters, can fit any data exactly, as shown in Example 3.1.2, Chapter 3. It can be seen that the task of model selection is more complex than a including single penalty term for the number of parameters in the criterion, and it depends on the structure of the model. The methods of model selection proposed by [205, 121] are logically consistent, but difficult to solve numerically.

Typically, Bayesian model selection [205, 121] is performed by evaluating the model selection criterion for every possible model. However, the discrimination criterion is often extremely expensive to compute exactly. It is therefore wasteful to explicitly enumerate all possible models when evaluating the discrimination criterion, even if the number of possible models is small. A preferable approach is to formulate the model selection problem as either a integer nonlinear program or a mixed integer nonlinear program. The resulting selection process is an optimization problem. This approach has several advantages:

1. it may not be necessary to evaluate the selection criterion for all models, and,
2. it may be possible to generate cheap bounds on the objective function that eliminate the need for a costly objective function evaluation.

It should be stressed that this optimization approach to model selection has only recently become feasible due to advances in optimization technology. The method is still immature but represents a very interesting avenue of research.

### 4.2.1 Optimization Based Model Selection

The Bayesian model selection criteria of [205, 121] are based on the posterior PDFs for the parameters. In this Section these model selection criteria will be reformulated as integer optimization problems. The model selection criteria will be presented for single output models to simplify the exposition. The model $M_i$ is a specific combination of mechanistic model (or expectation function), measurement model and prior PDF for the parameters. The prior PDF must be proper; i.e., satisfy the constraint:

$$\int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} \pi_{\mathbf{p}}(\mathbf{p})\ \mathrm{d}\mathbf{p} = 1.$$

A set of integer variables $(z_1, z_2, \ldots, z_{n_z}) \in \{0,1\}^{n_z}$ is used to index the models $M_i$ in the hypothesis space under investigation. For example, the set of mechanistic models:

$$\begin{aligned} M^1 : x &= 0 \\ M^2 : x &= p_1 u \\ M^3 : x &= p_2 u^2 \\ M^4 : x &= p_1 u + p_2 u^2, \end{aligned}$$

could be written as

$$x = z_1 p_1 u + z_2 p_2 u^2. \tag{4.30}$$

Ideally, the mechanistic model may be defined as the solution of a set of algebraic nonlinear equations, the solution at fixed times to a set of ODEs or the solution at fixed times to a set of DAEs/PDAEs. It is assumed that each model (expectation, measurement and prior) is dependent on a subset of parameters $\mathbf{p}^j \in \mathbb{R}^{n_{p_j}} \subset \mathbf{p} \in \mathbb{R}^{n_p}$. If the $i^th$ parameter is included in the $j^{th}$ model $z_i = 1$ and if the parameter is not included in the model then $z_i = 0$. It follows that

$$n_{p_j} = \sum_{i=1}^{n_z} z_i.$$

Hence, the state of the system, $x_i \in \mathbb{R}$, is uniquely determined by the inputs, $\mathbf{u}_i \in \mathbb{R}^{n_u}$, and the parameters, $(\mathbf{p}, \mathbf{z}) \in \mathbb{R}^{n_p} \times \{0,1\}^{n_z}$. It should be stressed that different expectation models, $M_j$, may be dependent on different numbers of real parameters. Additional logic constraints must be added to the optimization problem to set the values of real parameters that do not appear in the expectation model, otherwise the solution to the optimization problem will be degenerate. The binary variables are used to change the structure of the expectation model, measurement model and prior PDF. The output of the system, $\hat{y}_i$, is related to the state of the system, by a probability model,

$$\mathrm{f}_y(y_i|\hat{x}_i = x_i, \hat{\mathbf{p}}_2 = \mathbf{p}_2, \hat{\mathbf{z}} = \mathbf{z}),$$

where $\mathbf{p}_2 \in \mathbb{R}^{n_{p2}}$ is a subset of $\mathbf{p}$ which characterize the model of uncertainty in the measurement. It is assumed that each of the measurements are independent of each other. The PDF for the vector of $n_k$ measurements is given by Equation (4.31),

$$\mathrm{f}_{\mathbf{y}}\Big(\mathbf{y}|\hat{\mathbf{X}} = \mathbf{X}(\mathbf{U}, \mathbf{p}, \mathbf{z}), \hat{\mathbf{p}} = \mathbf{p}, \hat{\mathbf{z}} = \mathbf{z}\Big) = \prod_{i=1}^{n_k} \mathrm{f}_y(y_i|\hat{\mathbf{x}}_i = \mathbf{x}(\mathbf{u}_i, \mathbf{p}, \mathbf{z}), \hat{\mathbf{p}} = \mathbf{p}, \hat{\mathbf{z}} = \mathbf{z}), \tag{4.31}$$

where $\mathbf{U} \in \mathbb{R}^{n_u \times n_k}$ is a matrix of input conditions corresponding to $n_k$ experiments and $\mathbf{y} \in \mathbb{R}^{n_k}$ is a vector of measurements. By application of Bayes' theorem the joint posterior PDF for the parameters, given the measurements, can be derived and is shown in Equation (4.32),

$$\mathrm{f}_{\mathbf{p},\mathbf{z}}(\mathbf{p}, \mathbf{z}|\hat{\mathbf{y}} = \mathbf{y}, \mathbf{U}) = \frac{\mathrm{f}_{\mathbf{y}}\Big(\mathbf{y}|\hat{\mathbf{X}} = \mathbf{X}(\mathbf{U}, \mathbf{p}, \mathbf{z}), \hat{\mathbf{p}} = \mathbf{p}, \hat{\mathbf{z}} = \mathbf{z}\Big)\, \pi_{\mathbf{p},\mathbf{z}}(\mathbf{p}, \mathbf{z})}{\pi_{\mathbf{y}}(\mathbf{y})} \tag{4.32}$$

where $\pi_{\mathbf{p},\mathbf{z}}(\mathbf{p},\mathbf{z})$ is the prior PDF for $(\mathbf{p},\mathbf{z})$. A technical difficulty, is that the structure of the prior PDF will depend on the number of real parameters that appear in the expectation function and measurement model. If $z_i = 1$, then a term corresponding to $p_i$ should be included in the prior density. If it is assumed that initially, the parameters are not correlated, the prior PDF can be expressed as the product of individual functions, as shown in Equation (4.33).

$$\pi_{\mathbf{p},\mathbf{z}}(\mathbf{p},\mathbf{z}) = \prod_{i=1}^{n_p} \left( z_i \pi_{p_i}(p_i) + (1 - z_i) \right) \tag{4.33}$$

The likelihood function and prior PDF completely define the joint posterior PDF which is used to characterize the uncertainty in the complete vector of parameters $(\mathbf{p},\mathbf{z})$ given that the values of the measurements $\mathbf{y}$ are known.

One possible model selection scheme is to estimate whole parameter vector $(\mathbf{p},\mathbf{z})$ from the joint posterior PDF. For an algebraic expectation function this would correspond to solving an Mixed Integer Nonlinear Program (MINLP). Algorithms have been developed to solve this type of problem [129, 130, 131]. However, it is more likely for cell signaling work that the expectation function is a system of ODEs. The corresponding optimization problem would then be formulated as a mixed integer dynamic optimization problem. To our knowledge these problems have not been solved using deterministic global methods. However, it is possible to use the convexity theory developed in [203] with the techniques in [129, 130, 131] to solve this problem. Unfortunately, the necessary theory has only existed for the last two months and these problem formulations were not studied in this thesis.

It is more likely that it is only required to estimate integer parameters from the marginal posterior PDF. The marginal posterior PDF is defined as:

$$\mathrm{f}_{\mathbf{z}}(\mathbf{z}|\hat{\mathbf{y}} = \mathbf{y}, \mathbf{U}) = \int_P \mathrm{f}_{\mathbf{p},\mathbf{z}}(\mathbf{p},\mathbf{z}|\hat{\mathbf{y}} = \mathbf{y}, \mathbf{U}) \; \mathrm{d}\mathbf{p}, \tag{4.34}$$

where $P$ is a space defined by the prior PDF. Rigorous optimization techniques for the objective function defined in Equation (4.34) do not currently exist. However, again it is postulated that branch-and-bound strategies may be effective. There are some interval techniques to construct rigorous bounds on the value of the multi-dimensional integral shown in Equation (4.34). These techniques are discussed in § 5.4.4, Chapter 5. However, it was found that sometimes these techniques do not work.

## 4.3 Summary

A recently developed theory of deterministic global optimization [203] was presented in this Chapter. This theory is applicable to a broad range of parameter estimation problems, including those derived from mechanistic models of cell signaling. The advantages of global optimization compared to traditional parameter estimation approaches are twofold:

1. the technique guarantees the correct parameter estimate, and,
2. it is also possible to identify other parameter values which correspond to regions of high probability density.

Therefore, it is easier to identify parameter estimation problems which are likely to lead to poor estimates (insufficient data, poor experimental design, etc.).

Model selection was discussed in the second half of this Chapter. It was found that many existing model selection criteria are based on too many restrictive assumptions (linearity of expectation model, Normal measurement model, etc.). The model selection problem was formulated in a Bayesian framework as an integer optimization problem. While techniques to solve the resulting optimization problems are poorly developed, it is postulated that this may prove an exciting new research area.

# Chapter 5

# Mammalian Cell Migration

There have been numerous investigations of *in-vitro* migration of mammalian cells (see for example [8, 56, 91, 93, 97, 143, 162, 166, 172, 184, 237, 242]). Much of the motivation for investigating cell migration has already been presented in Chapter 1, § 1.4. In particular, cell migration is a key process in inflammation, wound healing, embryogenesis, and tumor cell metastasis [233]. A lot of this research has focused on the molecular biology of cell motion (actin polymerization, assembly/disassembly of focal adhesions, microtubule dynamics, etc.). However, it is ultimately desirable to be able to correlate cell type and cell conditions to cell physiology; i.e., we wish to determine how much condition $X$ affects cell motion. The mechanistic understanding of cell migration is not yet detailed enough to be able to answer this question. It is therefore important to have experimentally verified conclusions. In particular, we wish to characterize experimentally how much external conditions affect cell motion. It is hypothesized that *in vitro* characterization of cell migration will be relevant for understanding *in vivo* cell migration. Work in this Chapter will focus on attempting to answer this problem. In particular, the aim is to characterize cell migration tracks with a few parameters (e.g., diffusivity of motion or cell speed, frequency of turning). Furthermore, posterior PDFs are derived for these parameters. The influence of experimental conditions on cell migration is revealed by the comparison of the shape and location of posterior PDFs resulting from each condition.

## 5.1 Experimental Setup

A typical use of a cell migration assay is described in [147]. The goal of their work was to determine the effect of EGF and fibronectin concentration on the speed of cell migration. In a standard setup, migrating cells are imaged using a digital video microscopy. A schematic of the microscope is shown in Figure 5-1. Images of the cells are sampled at 15min intervals. A typical image is shown in Figure 5-2[1]. The sampled images of the migrating cell is converted into cell centroid data using the proprietary DIAS[2] software. In the open literature, the clearest description of this processing step is [215]. The resulting cell centroid data is shown in Figure 5-3. The work in this Chapter focuses on converting the cell centroid data into physically relevant parameters that can be correlated with cell conditions.

## 5.2 Random Walk Models

Unfortunately, there is insufficient information to build a mechanistic model describing cell-centroid position as a function of time. The lack of knowledge is twofold: it is hypothesized that cell migration is controlled by low number cell receptor activation where stochastic effects are dominant [232], and knowledge of the regulation mechanism is incomplete. Therefore, cell migration is typically modeled as a random walk, dating from the work of [91]. The greater level of abstraction allows for many different effects to be lumped into the value of a few parameters. It is often desired to estimate the random walk parameters from experimental data to correlate these parameters either to the output from mechanistic models [58, 166] or to characterize a particular experimental intervention.

Cellular behavior in biological systems is often distributed [165] and researchers may wish to characterize inter-cellular variation. It is therefore desirable to estimate the parameters defining the motion from measurements of a single particle. The objective of the work in this Chapter is to derive Bayesian parameter estimates for

[1]Image provided by Brian Harms.

[2]Solltech, Inc., Technology Innovation Center, Oakdale, IA 52319, USA.

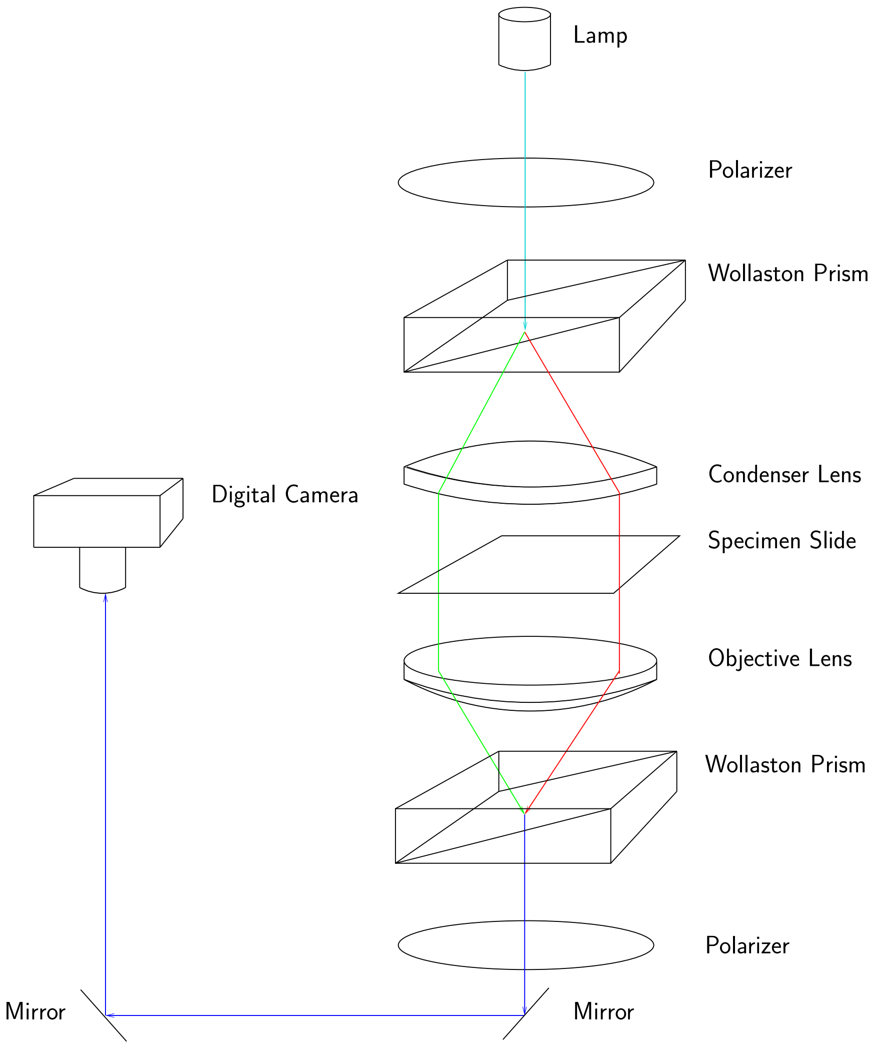


Figure 5-1: Microscope setup

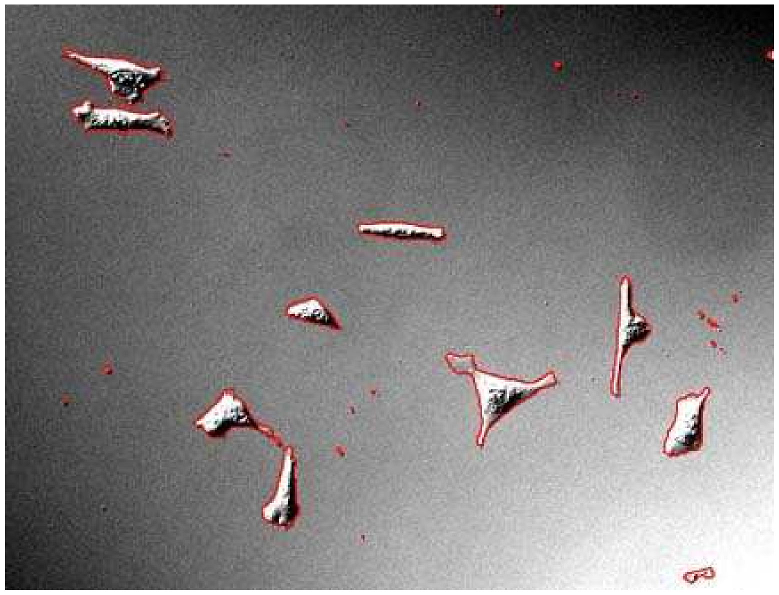


Figure 5-2: Microscope image of migrating cells

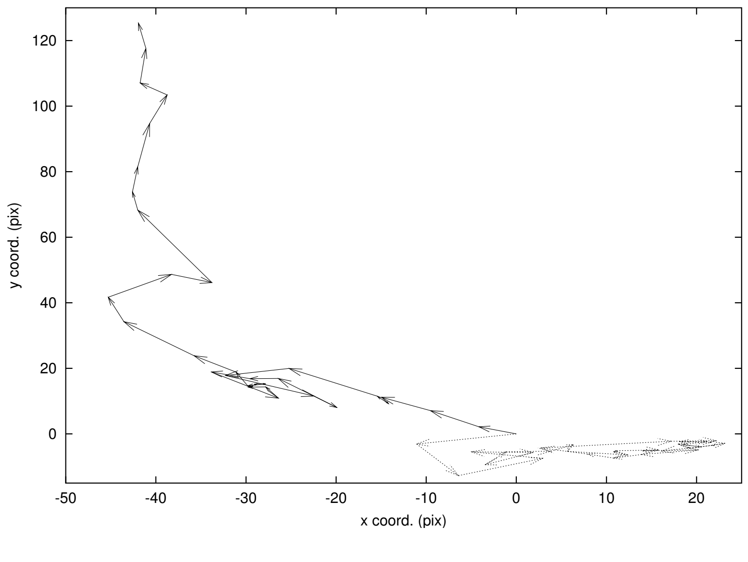


Figure 5-3: Sample cell centroid data

two different types of random walk: Brownian diffusion, and a correlated random walk. It is shown in § 5.3.3 that these models exhibit quite different behavior and can only be used interchangeably under restrictive conditions.

The simplest model is Brownian diffusion (a position-jump process) where there are $n_d$ states corresponding to the dimension of the walk. The particle displacement is completely uncorrelated between time intervals. Historically, the diffusivity of the walk is estimated from fitting the mean squared displacement. We can show that this method has several major pitfalls. In contrast, we apply a Bayesian analysis to this problem and show that the estimates of diffusivity obtained are superior to the estimate from the mean squared displacement. A weakness of the Brownian diffusion model for representing cell migration is that it admits the possibility of infinite speed signals and experimental evidence shows that cells migrate at finite speeds. However, at long sampling times this difficulty becomes less significant [91].

A more complex model for describing cell migration is a one-dimensional correlated random walk. This model is interesting as it is the simplest model where migration occurs at finite speeds. The one-dimensional model is sufficient to analyse some biologically relevant systems, including neuron cell migration, where movement is limited to the axis of the astroglial fiber [93]. The formulation of the parameter estimation problem does not change for motion in higher dimensions but solution of the problem becomes significantly more computationally expensive.

It is important to state several facts in order to appreciate the value of our contribution.

1. The particle position, $x_i$, at time $t_i$ is not the same as a measurement of the particle position, $y_i$, at time $t_i$; there is error in the measurement of the particle position.

2. A particle position at a single point in time, $x_i$ is not the same as a set of particle positions at a set of times, $\mathbf{x}$.

3. The joint density for the set of particle positions does not equal the product of

the density for the particle position at a single point in time, i.e.,

$$p(x_i|x_{i-1}) \neq p(x_i)\,.$$

There has been extensive study of Brownian diffusion [135, 33, 37, 92], starting from the original work [76, 213, 139]. Most of this work focuses on the properties of the probability density function (PDF) for the particle position, $x_i$, at a single point in time. To our knowledge no one has tried to analyze the joint density for a set of particle positions $\mathbf{x}$. The parameter estimation strategy is to derive the posterior probability density for the parameters of the walk according to Bayes theorem (see [29, 123, 244, 121] for details of Bayesian parameter estimation).

Most of the previous work on the correlated random walk has focused on the properties of the probability density function (PDF) for the particle position, $x_i$, at a single point in time. We are unaware of work that considers Bayesian parameter estimation for a one-dimensional correlated random walk. Parameter estimation for a two-dimensional correlated random walk by fitting moments has been considered [91, 62]. It has been demonstrated that parameter estimation by minimizing the square of residuals between the estimated the mean squared displacement and measured mean square displacement has the following difficulties:

1. the method only works if the magnitude of the measurement error, $\alpha$, is known *a priori*,

2. the method can lead to non-physical estimates of the model parameters, and,

3. there is a significant probability of a large discrepancy between the estimate parameters and the true value of the parameters.

The Bayesian parameter estimation framework does not suffer from these problems. To our knowledge no one has tried to analyze the joint density for a set of particle positions $\mathbf{x}$.

## 5.3 Brownian Diffusion

For sake of simplicity, we will consider diffusion in $\mathbb{R}$ and at the end of the section generalize the results to higher dimensional walks. We will drop the circumflex notation used in Chapter 3. However, it will be still understood that PDFs and CDFs are based on comparisons of the form $\hat{x} < x$. We will derive the posterior PDF of the diffusivity of a particle given a set of measurements,

$$\mathbf{y} = \left(y_1, y_2, \ldots, y_{n_y}\right) \in \mathbb{R}^{n_y},$$

of a particle obeying Brownian diffusion. Sometimes it will be useful to refer to the vector of measured particle displacements,

$$\mathbf{d} = \left(d_1, d_2, \ldots, d_{n_y}\right) \in \mathbb{R}^{n_y},$$

where $d_i = y_i - y_{i-1}$. The Brownian walk has a single state, $x(t)$, corresponding to the displacement of the particle along the axis. The vector,

$$\mathbf{x} = \left(x_0, x_1, \ldots, x_{n_y}\right) \in \mathbb{R}^{n_y+1},$$

where $x_i = x(t_i)$, represents the displacement of the particle at discrete time points, $t = t_i$. The initial measurement of particle position will be set arbitrarily $y_0 = 0$; it does not make a difference what value for $y_0$ is chosen, but $y_0 = 0$ will simplify the subsequent calculations.

It is well known that the PDF for the location of a particle diffusing according to Brownian motion in one dimension, $p(\cdot)$, is given by the solution of the parabolic partial differential equation [76, 213, 139],

$$D\frac{\partial^2 p}{\partial x^2} = \frac{\partial p}{\partial t}$$

where $x$ is the particle location, $t$ is the time elapsed, and $D$ is the diffusivity of the particle. If it is assumed that the initial location of the particle is known $(x = x_{n-1})$, the initial condition $p(x|t=0)=\delta(x-x_{n-1})$ is implied. Additionally, the constraint,

$$\int_{-\infty}^{\infty} p(x|t=0)\ \mathrm{d}x = 1$$

must be satisfied. The PDF is given by the standard result,

$$p(x|x_{n-1}, D, t) = N(x_{n-1}, 2Dt) \tag{5.1}$$

where $N(\cdot)$ is a Normal density.

The PDF for the measurement of particle positions, $f_m(\cdot)$, is given by:

$$f_m(y_i|x_i,\alpha) = N\big(x_i, \alpha^2\big)\,, \quad i = 1, \dots, n_y$$

where $\alpha$ is a parameter characterizing the magnitude of the errors in the measurement. The initial location of the particle is unknown. However, it is reasonable to assume the prior for $x_0$ is $p(x_0|\alpha) = N(0,\alpha^2)$, since we have assumed $y_0 = 0$. For a single time interval, the PDF for the measured displacement, $d$ is

$$\begin{aligned} p(d|D,\alpha,\Delta t) &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f_m(d|x_1,\alpha)\, p(x_1|x_0,D,\Delta t)\, p(x_0|\alpha)\ \mathrm{d}x_1\mathrm{d}x_0 \\ &= N\big(0, 2D\Delta t + 2\alpha^2\big)\,. \end{aligned}$$

A common literature method [91] for estimating the diffusivity is to fit $D$ by least-squares to the squared displacement:

$$D_{LS} = \arg\left[\min_D \sum_{i=1}^{n_y} \big(d_i^2 - \big(2D\Delta t_i + 2\alpha^2\big)\big)^2\right].$$

This yields the estimate:

$$D_{LS} = \frac{1}{2\sum_{i=1}^{n_y}\Delta t_i}\sum_{i=1}^{n_y}\big(d_i^2 - 2\alpha^2\big)\,, \tag{5.2}$$

although many authors forget to correct for the measurement error. There are three weaknesses to this method of estimation:

1. it is impossible to estimate accurately the confidence intervals for $D_{LS}$,

2. it is necessary to know the correct value of $\alpha^2$ *a priori*, and,

3. it does not account for correlation in the measured displacement between two adjacent time intervals.

It is not possible to estimate $\alpha$ with the least-squares method, since the measured displacements, $d_i$, are assumed to be independent. The assumption of independence of the measured displacements is false. Clearly a measurement error in $y_i$ affects both $d_i$ and $d_{i+1}$. In contrast, the Bayesian formulation explicitly accounts for the correlation in measured displacement between time intervals.

From Bayes' theorem, the joint PDF for the measurements and the particle positions at discrete times is

$$g(\mathbf{y}, \mathbf{x}|D, \alpha, \mathbf{t}) = p(x_0|\alpha) \prod_{i=1}^{n_y} f_m\big(y_i|x_i, \alpha^2\big) \prod_{i=0}^{n_y-1} p(x_{i+1}|x_i, D, \Delta t_i) \tag{5.3}$$

where $\Delta t_i = t_i - t_{i-1}$ is the sampling interval. Alternatively, Equation (5.3) can be written

$$\mathbf{z} \sim N\big(\mathbf{0}, \mathbf{V}^{-1}\big)\,,$$

where $\mathbf{z} = (\mathbf{x}, \mathbf{y})$, and

$$\mathbf{V} = \begin{bmatrix} \mathbf{V}_{11} & \mathbf{V}_{12} \\ \mathbf{V}_{21} & \mathbf{V}_{22} \end{bmatrix}.$$

The submatrices $\mathbf{V}_{11} \in \mathbb{R}^{(n_y+1)\times(n_y+1)}$, $\mathbf{V}_{21}^T = \mathbf{V}_{12} \in \mathbb{R}^{(n_y+1)\times n_y}$, and $\mathbf{V}_{22} \in \mathbb{R}^{n_y \times n_y}$

are given by:

$$\mathbf{V}_{11} = \begin{cases} v_{11} = \dfrac{1}{\alpha^2} + \dfrac{1}{2D\Delta t_1} & \\ v_{nn} = \dfrac{1}{\alpha^2} + \dfrac{1}{2D\Delta t_{n_y}} & n = n_y + 1 \\ v_{ii} = \dfrac{1}{\alpha^2} + \dfrac{1}{2D\Delta t_i} + \dfrac{1}{2D\Delta t_{i-1}} & i = 2, \dots, n_y \\ v_{ij} = -\dfrac{1}{2D\Delta t_i} & i = 1, \dots, n_y, \quad j = i+1 \\ v_{ij} = -\dfrac{1}{2D\Delta t_{i-1}} & i = 2, \dots, n_y + 1, \quad j = i-1 \\ 0 & \text{Otherwise} \end{cases},$$

$$\mathbf{V}_{21}^T = \mathbf{V}_{12} = \begin{bmatrix} 0 & \dots & 0 \\ -\frac{1}{\alpha^2} & & \\ & \ddots & \\ & & -\frac{1}{\alpha^2} \end{bmatrix},$$

and $\mathbf{V}_{22} = \alpha^{-2}\mathbf{I}$. The marginal density for the data can be obtained through integration of Equation (5.3):

$$r(\mathbf{y}|D, \alpha, \mathbf{t}) = \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} g(\mathbf{y}, \mathbf{x}|D, \alpha, \Delta t) \, \mathrm{d}\mathbf{x} \tag{5.4}$$

and is given by [244]:

$$\mathbf{y} \sim N\Big(\mathbf{0}, \left(\mathbf{V}_{22} - \mathbf{V}_{21}\mathbf{V}_{11}^{-1}\mathbf{V}_{12}\right)^{-1}\Big).$$

If the value of $\alpha$ is known, the posterior PDF for the diffusivity, $h_1(D|\mathbf{y}, \alpha, \mathbf{t})$, is obtained by application of Bayes' theorem. We will assume an improper uniform prior for the particle diffusivity. It could be argued that the prior for the particle diffusivity is proportional to $1/D$ or even $1/D^2$ [123]. However, the substance of the calculation will remain the same and we will neglect this complication.

$$h_1(D|\mathbf{y}, \alpha, \mathbf{t}) = \frac{1}{K_1} r(\mathbf{y}|D, \alpha, \mathbf{t}) \tag{5.5}$$

The constant $K_1$ can be determined from,

$$K_1 = \int_0^\infty r(\mathbf{y}|D, \alpha, \mathbf{t}) \; \mathrm{d}D.$$

and if necessary, evaluated by numerical quadrature.

However, the more likely scenario is that $\alpha$ is unknown, in which case we should decide whether we are interested in estimating the joint density, $h_2\left(D, \alpha|\mathbf{y}, \mathbf{t}\right)$, or the marginal density, $h_3\left(D|\mathbf{y}, \mathbf{t}\right)$. Remarkably, it is possible to distinguish between the contributions to the measured displacement from measurement error and particle diffusion. To derive both the joint and the marginal densities it is necessary to assume a prior for $(\alpha, D)$. We will again assume an improper uniform prior. To derive the joint density, application of Bayes' theorem yields:

$$h_2(D, \alpha|\mathbf{y}, \mathbf{t}) = \frac{1}{K_2} r(\mathbf{y}|D, \alpha, \mathbf{t}) \,, \tag{5.6}$$

where,

$$K_2 = \int_0^\infty \int_0^\infty r(\mathbf{y}|D, \alpha, \mathbf{t}) \; \mathrm{d}D \, \mathrm{d}\alpha.$$

The marginal density is obtained by integration of the joint density with respect to $\alpha$:

$$h_3(D|\mathbf{y}, \mathbf{t}) = \int_0^\infty h_2(D, \alpha|\mathbf{y}, \mathbf{t}) \, \mathrm{d}\alpha. \tag{5.7}$$

All of the necessary integrations can be achieved using simple quadrature.

*Remark.* It is possible to generalize the results in § 5.3 to a particle diffusing in higher dimensions. As discussed in [37] the PDF for the location of a particle diffusing according to isotropic Brownian motion in more than one dimension, $p(\cdot)$, is given by:

$$p\big(x_{i1}, x_{i2}, \ldots | x_{(i-1)1}, x_{(i-1)2}, \ldots, D, \Delta t\big) = \\ N\big(x_{(i-1)1}, 2n_d D \Delta t_i\big) \cdot N\big(x_{(i-1)2}, 2n_d D \Delta t_i\big) \ldots$$

and if it is assumed that the error in the measurement of each coordinate is independent and Normal, the posterior PDF is given by:

$$\frac{1}{n_d} h_1(n_d D|\mathbf{Y}, \alpha, \mathbf{t}) = \frac{1}{K_1 n_d} \prod_{i=1}^{n_y} r(n_d D|\mathbf{y}_i, \alpha, \mathbf{t})$$

where,

$$K_1 = \int_0^\infty \frac{1}{n_d} \prod_{i=1}^{n_y} r(n_d D|\mathbf{y}_i, \alpha, \mathbf{t}) \; \mathrm{d}\,(n_d D)\,,$$

and $\mathbf{Y} \in \mathbb{R}^{n_y \times n_d}$ is a matrix of $n_y$ measurements and $n_d$ is the number of dimensions. The joint density, $h_2(\cdot)$, and the marginal density, $h_3(\cdot)$, can be derived in analogous fashion.

### 5.3.1 Results

A noisy Brownian random walk was simulated (see Figure 5-4) for $\alpha^2 = 9$, $2D\Delta t = 6$, and $n_y = 30$. The parameter values were deliberately chosen to ensure that the contribution to the measured displacement from both measurement error and diffusion would be comparable. It should be realized that the simulation represents just one of an infinite number of possible realizations for $\mathbf{y}$. Each different realization of $\mathbf{y}$ will lead to a slightly different shapes for the posterior PDFs. The plots shown in this paper were selected to be qualitatively "characteristic" of a several runs. The quality of the estimates obtained from the posterior PDFs is characterized in § 5.3.2.

The joint posterior PDF, $h_2(D, \alpha|\mathbf{y}, \mathbf{t})$ was evaluated according to Equation (5.6) (shown in Figure 5-5). It can be seen from this plot that the PDF has a distinct maximum, suggesting that it is indeed possible to estimate both the diffusivity and the measurement error. The marginal posterior PDF $h_3(D|\mathbf{y}, \mathbf{t})$ and the conditional posterior PDF $h_1(D|\mathbf{y}, \mathbf{t}, \alpha)$ are shown in Figure 5-6. It can be seen that PDF $h_2(D|\mathbf{y}, \mathbf{t})$ is less peaked than $h_1(D|\mathbf{y}, \mathbf{t}, \alpha)$. This is to be expected; the conditional PDF (more peaked) represents a greater state of knowledge than the marginal PDF (less peaked). However, it is satisfying to notice that lack of knowledge of $\alpha$ does not lead to catastrophic widening of the PDF.

The Maximum A Posteriori (MAP estimate) is the value of a parameter that

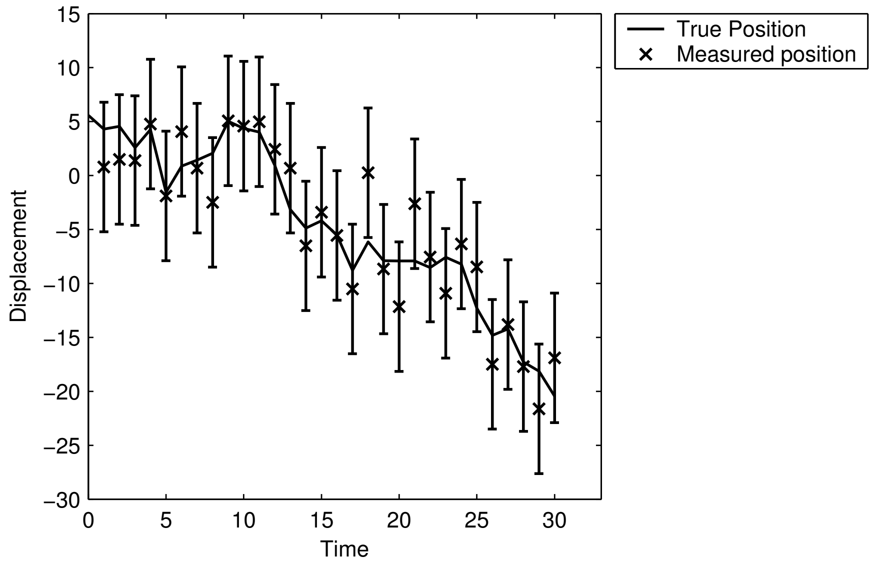


Figure 5-4: Simulated Brownian random walk for $D = 3$, $\alpha = 3$, $n_y = 30$

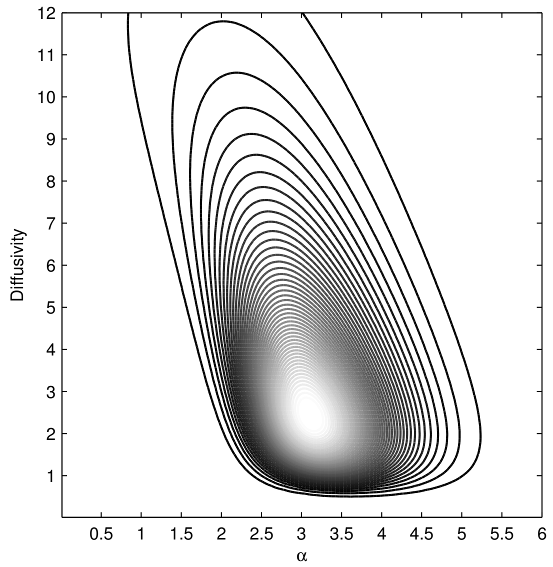


Figure 5-5: Joint posterior PDF, $h_2(D, \alpha|\mathbf{y}, \mathbf{t})$

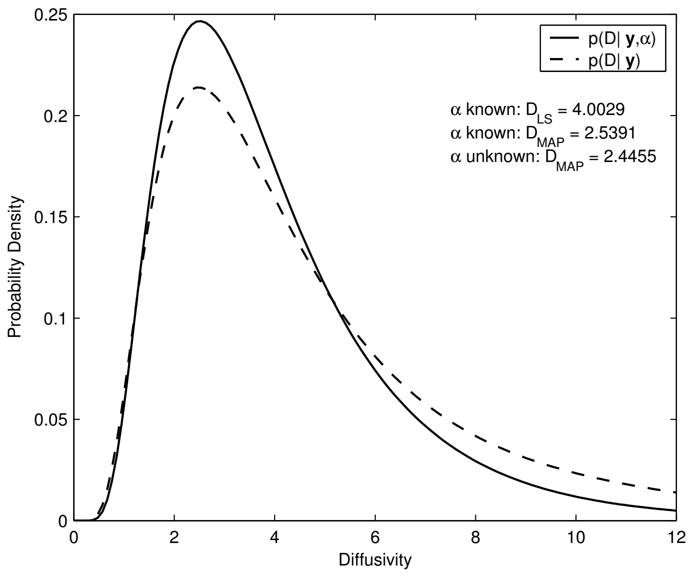


Figure 5-6: Marginal posterior and conditional PDFs for particle diffusivity

maximizes the posterior PDF [29]. It can also be shown that this is the appropriate estimate when the inference problem is framed as a decision with a "0-1" loss function [121]. We will let $D_{MAP}(\mathbf{y}, \mathbf{t}, \alpha)$ denote the MAP estimate of the diffusivity when $\alpha$ is known and $\hat{D}_{MAP}(\mathbf{y}, \mathbf{t})$ denote the MAP estimate when $\alpha$ is unknown. The least-squares estimate of the diffusivity (calculated from Equation (5.2)) is denoted $D_{LS}(\mathbf{y}, \mathbf{t}, \alpha)$. The following estimates of the diffusivity were calculated from the simulation shown in Figure 5-4: $D_{LS} = 4.0$, $D_{MAP} = 2.5$, and $\hat{D}_{MAP} = 2.4$. However, it can be seen from the long asymmetric tails of $h_1(D|\mathbf{y}, \mathbf{t}, \alpha)$ and $h_3(D|\mathbf{y}, \mathbf{t})$ (shown in Figure 5-6) that a point estimate of the diffusivity is a little misleading. In general, we will prefer the full posterior PDF (if it is available) rather than a point estimate. Furthermore, it is impossible to accurately construct confidence intervals from the least-squares problem. In contrast, it is straightforward to calculate confidence intervals directly from the posterior probability density function [29].

### 5.3.2 Comparison of MAP and Least-Squares Estimate

It has already been stated in § 5.3.1 that the plots shown in Figures 5-4–5-6 only characterize a single simulation. In general, the results from each simulation will be slightly different. In this section we will characterize in greater detail the performance of the estimates: $D_{LS}(\mathbf{y}, \mathbf{t}, \alpha)$, $D_{MAP}(\mathbf{y}, \mathbf{t}, \alpha)$, and $\hat{D}_{MAP}(\mathbf{y}, \mathbf{t})$. Even if we know the true value of the diffusivity we will not collect the same data with each experiment. This process has already been characterized by the PDF, $r(\mathbf{y}|D, \alpha, \mathbf{t})$, shown in Equation (5.4). Correspondingly, the results from each different experiment would yield a different estimate of the diffusivity. It is therefore interesting to calculate the PDFs, $p(D_{LS}|D, \mathbf{t}, n_y, \alpha)$, $p(D_{MAP}|D, \mathbf{t}, n_y, \alpha)$, and $p\left(\hat{D}_{MAP}|D, \mathbf{t}, n_y\right)$ where $D$ is the true value of the diffusivity. Ideally, this PDF will be sharply peaked and its mode will correspond to the true value of the diffusivity, i.e., it is probable that the collected data, $\mathbf{y}$, will lead to an estimate of the diffusivity that is close to the true value of the diffusivity.

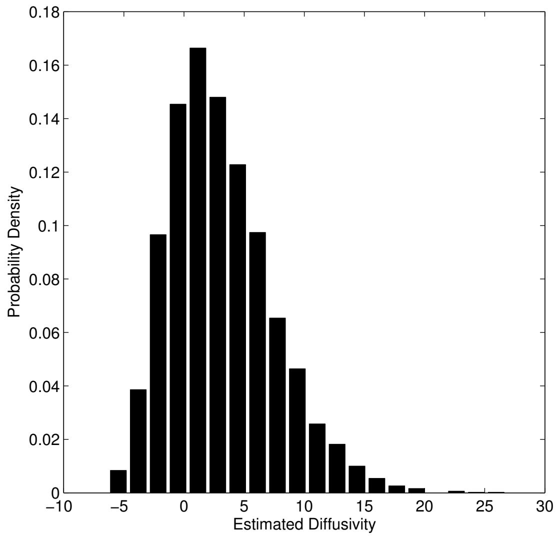
The easiest method to calculate the PDF $p(D_{EST}|D, I)$ ($I$ is the additional information) is Monte Carlo since closed-form solutions for the MAP estimates, $D_{MAP}$ and $\hat{D}_{MAP}$, do not exist. Plots for each of the estimates: $D_{LS}$, $D_{MAP}$, and $\hat{D}_{MAP}$ are shown in Figure 5-7. The Monte Carlo simulations were based on $n = 5000$ samples.

It is clear from Figure 5-7 that $D_{LS}$ is a poor estimate of $D$. The PDF,

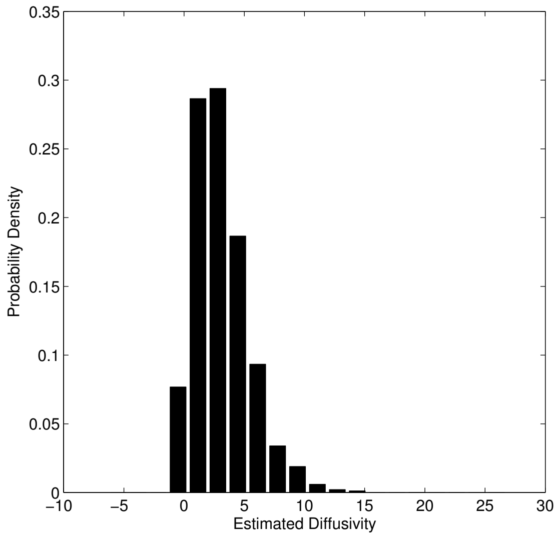
$$p(D_{LS}|D, \alpha, \Delta t, n_y)\,,$$

has wide tails indicating a high probability that the least squares estimate will not coincide with the true value of the diffusivity. Furthermore, it can be seen that there is a significant probability that the calculated value of the least-squares estimate is negative. This surprising result comes from the correction for measurement error. It is impossible to calculate a negative value for the uncorrected least-squares estimate. However, the curve for the PDF for the uncorrected estimate would be translated to the right by $n_y\alpha^2 / \sum_{i=1}^{n_y} \Delta t_i$ and the mode would no longer coincide with the true value of the diffusivity.

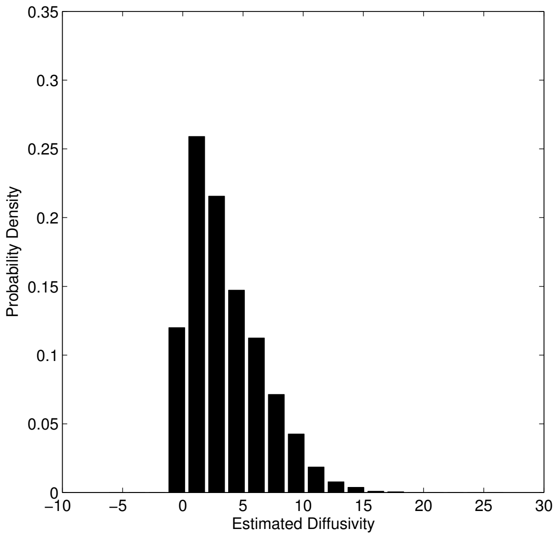
It is also interesting to compare $p(D_{MAP}|D, \alpha, \Delta t, n_y)$ with $p\left(\hat{D}_{MAP}|D, \Delta t, n_y\right)$. It can be seen from Figure 5-7 that both of these densities have a significant area



(a) $p(D_{LS}|D=3,\alpha=3,\Delta t=1,n_y=20)$



(b) $p(D_{MAP}|D=3,\alpha=3,\Delta t=1,n_y=20)$



(c) $p\left(\hat{D}_{MAP}|D=3,\Delta t=1,n_y=20\right)$

Figure 5-7: Comparison of different estimates for diffusivity ($\Delta t = 1$)

contained in the tails, i.e., there is still a significant probability that there will be large discrepancy between the calculated estimate and true value of the diffusivity. This is yet another warning that one should not rely on $D_{MAP}$ or $\hat{D}_{MAP}$ as a single point estimate characterizing the state of knowledge about $D$. It is also interesting to observe that there is not a large amount of widening between $p(D_{MAP}|D, \alpha, \Delta t, n_y)$ and $p\left(\hat{D}_{MAP}|D, \Delta t, n_y\right)$; more evidence that knowing $\alpha$ only provides a marginal improvement in the estimate of $D$.

### 5.3.3 Effect of Model-Experiment Mismatch

The effect of model-experiment mismatch was investigated. In a real cell migration data, it is likely that there is significant correlation between the displacement in adjacent time intervals due to persistance in cell motion. Perhaps a better model of cell migration is a correlated random walk, as described § 5.4. In this model, it is assumed that the cell moves in a straight line at constant speed and changes direction at time points obeying an exponential PDF. It has been shown that a correlated random walk tends to Brownian diffusion as the time interval at which the position measurements are sampled increases. The diffusion limit in this case is [165]:

$$D = \lim_{\Delta t \to \infty} \frac{C^2}{2\lambda}. \tag{5.8}$$

It is therefore interesting to generate data according to a correlated random walk and see whether the proposed Brownian diffusion estimation algorithms can extract useful information.

This simulation experiment was done for the $C = 3$, $\lambda = 0.6$ and $\Delta t = 1$. The results are shown in Figure 5-8. It can be seen that the effect of correlation for a short sampling time is to broaden the peak of the estimate. It can also be seen that there is a large discrepancy between the mode of the PDFs and the value of the diffusivity calculated from Equation (5.8) ($D = 3^2/(2 \times 0.6) = 7.5$). This is not surprising since the sampling time is small $\Delta t = 1$. In contrast, as $\Delta t$ is increased the accuracy of all the estimates improves (see Figure 5-9). However, there is still a fair degree of uncertainty in all the estimates. In fact, there is little difference between all of the estimates. Consequently, one might be tempted to use the least-squares estimate as it is simpler to compute. However, the marginal Bayesian estimate is perhaps preferable since no *a priori* knowledge of $\alpha$ is required.

## 5.4 Correlated Random Walk

Diffusion by discontinuous movements was briefly considered by G. I. Taylor [225]. A particle moves at constant speed, $C$, for an interval of time $\tau$. At each time point, $t = \tau, 2\tau, \ldots, n\tau$, the particle either continues in the same direction with probability $p$ or reverses direction with probability $q = 1 - p$. The seminal work [99] considered the limit of this discrete time random walk as $p = 1 - \tau/2A$, with $A$ constant, and $n \to \infty$, $\tau = \Delta t/n \to 0$ and showed that the probability density function describing the diffusion obeyed the telegraph equation. A continuous time model has been proposed [124] where a particle moves at constant speed, $C$, for exponentially distributed lengths of time, $\tau_i$, before reversing directions. The probability density function for $\tau_i$ is given by:

$$p(\tau_i|\lambda) = \begin{cases} \lambda \exp(-\lambda \tau_i) & \tau_i \geq 0 \\ 0 & \tau_i < 0 \end{cases} .$$

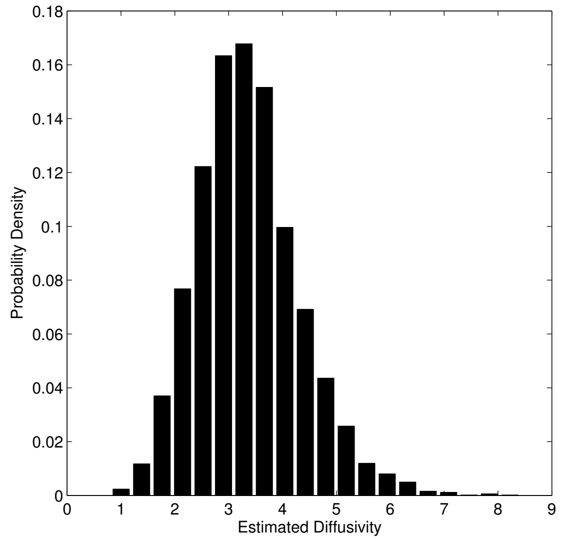
The parameter $\lambda$ characterizes the frequency at which the particle reorients. The constant $A$ is related to the turning frequency, $\lambda$, according to

$$\lambda = \frac{1}{2A}.$$

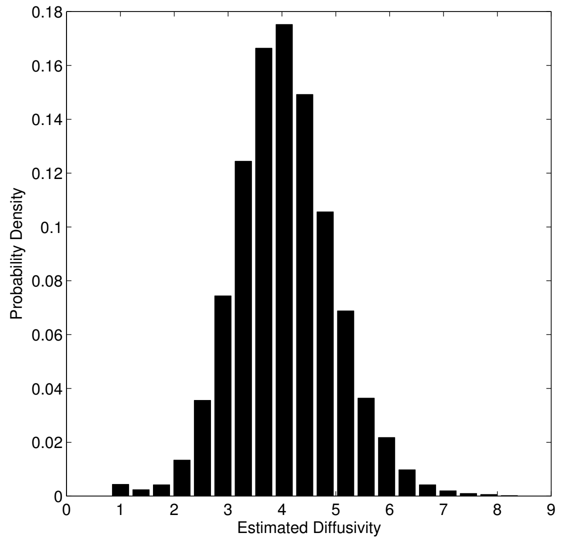
It has been shown that this model is equivalent to the limit of the discrete random walk model [124].

We will derive the posterior PDF of $(C, \lambda) \in \mathbb{R}_+^2$ for a particle given a set of measurements,

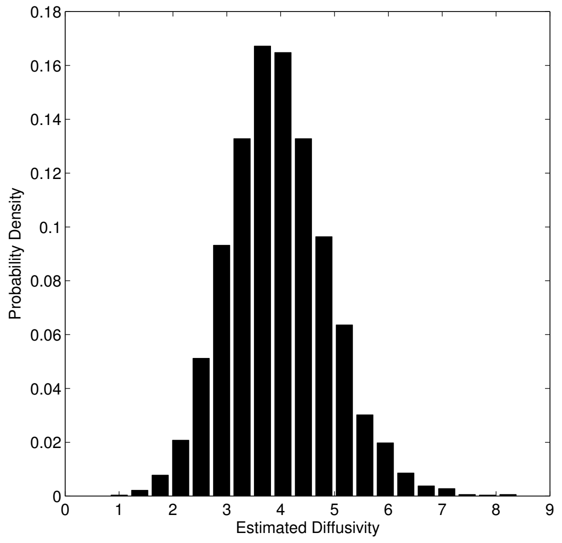
$$\mathbf{y} = \left(y_1, y_2, \ldots, y_{n_y}\right) \in \mathbb{R}^{n_y}$$



(a) $p(D_{LS}|C = 3, \lambda = 0.6, \alpha = 1, \Delta t = 1)$

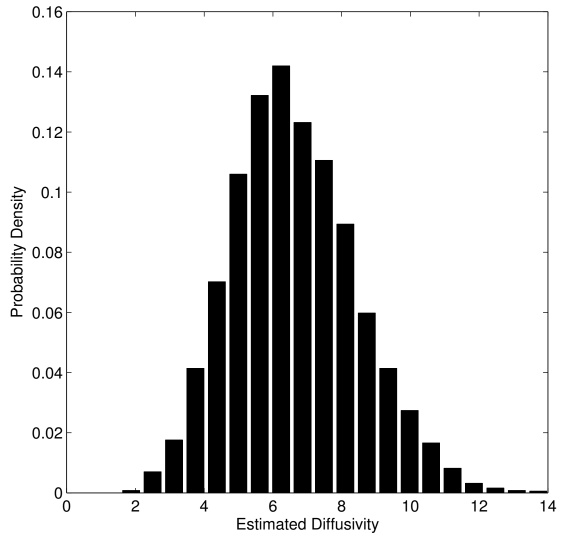


(b) $p(D_{MAP}|C = 3, \lambda = 0.6, \alpha = 1, \Delta t = 1)$

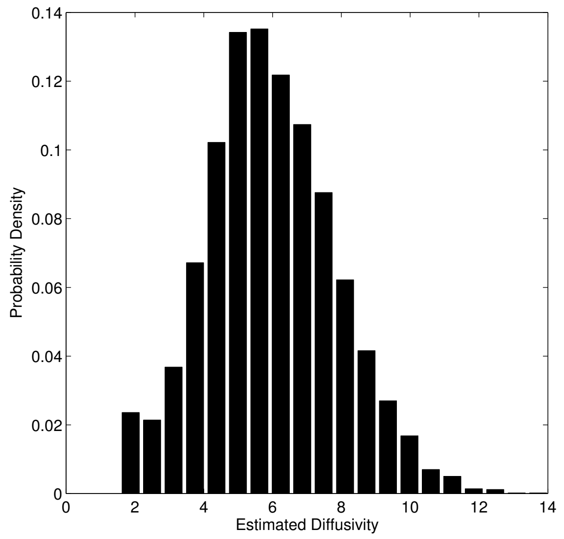


(c) $p\left(\hat{D}_{MAP}|C = 3, \lambda = 0.6, \alpha = 1, \Delta t = 1\right)$

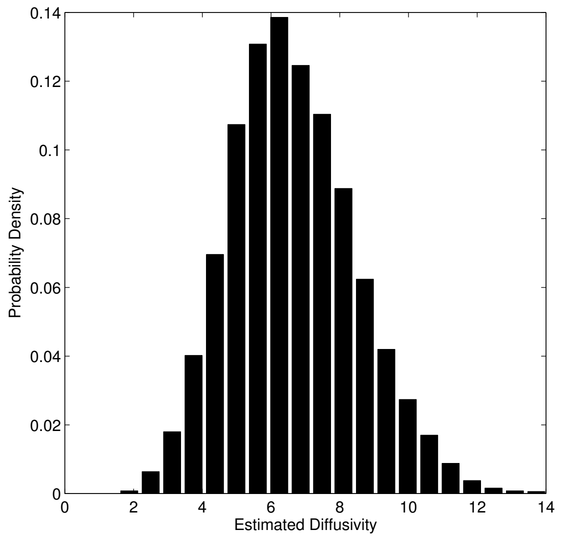
Figure 5-8: Diffusivity estimates for correlated random walk ($\Delta t = 1$, $n_y = 20$)



(a) $p(D_{LS}|C = 3, \lambda = 0.6, \alpha = 1, \Delta t = 7)$



(b) $p(D_{MAP}|C = 3, \lambda = 0.6, \alpha = 1, \Delta t = 7)$



(c) $p\left(\hat{D}_{MAP}|C = 3, \lambda = 0.6, \Delta t = 7\right)$

Figure 5-9: Diffusivity estimates for correlated random walk ($\Delta t = 7$, $n_y = 20$)

obeying the continuous correlated random walk. The initial measurement of particle position will be set arbitrarily $y_0 = 0$; it does not make a difference what value for $y_0$ is chosen, but $y_0 = 0$ will simplify the subsequent calculations. The PDF for the measurement of particle positions is given by:

$$f_m(y_i|x_i, \alpha) = N\big(x_i, \alpha^2\big)\,, \quad i = 1, \ldots, n_y \tag{5.9}$$

where $N(\cdot)$ is a Normal density and $\alpha$ is a parameter characterizing the magnitude of the errors in the measurement. The initial location of the particle is unknown. However, it is reasonable to assume the prior for $x_0$ is $p(x_0|\alpha) = N(0, \alpha^2)$ since we have assumed $y_0 = 0$. The correlated random walk has two states, $x(t)$ and $I(t)$, corresponding to the displacement along the axis and particle orientation, respectively. The vector,

$$\mathbf{x} = \big(x_0, x_1, \ldots, x_{n_y}\big) \in \mathbb{R}^{n_y+1}$$

where $x_i = x(t_i)$, represents the displacement of the particle at discrete time points, $t = t_i$. Sometimes it will be useful to refer to the vector of actual displacements

$$\mathbf{d} = \big(d_1, d_2, \ldots, d_{n_y}\big) \in \mathbb{R}^{n_y}$$

where $d_i = x_i - x_{i-1}$. The vector $\mathbf{I}$,

$$\mathbf{I} = \big(I_0, I_1, \ldots, I_{n_y}\big) \in \{0, 1\}^{n_y+1}$$

where $I_i = I(t_i)$, represents the particle orientation.

It can be shown that PDF for the particle position, $x$, is given by the solution of the telegraphers equation [99, 195, 124, 114, 126, 155] if it is assumed that it is equally probable the particle starts with a positive or negative orientation. The solution is shown in Equation (5.10),

$$\phi(x, \Delta t) = \frac{e^{-\lambda \Delta t}}{2C}\left[\delta\Big(\Delta t - \frac{x}{C}\Big) + \delta\Big(\Delta t + \frac{x}{C}\Big) + \lambda\left(I_0(\Gamma) + \frac{\lambda \Delta t}{\Gamma} I_1(\Gamma)\right)\right] \tag{5.10}$$

for $|x| \leq C\Delta t$ and $\phi(x, \Delta t) = 0$ for $|x| > C\Delta t$, where $\Gamma$ is defined as

$$\Gamma = \lambda\sqrt{(\Delta t)^2 - \frac{x^2}{C^2}},$$

and $I_0(\cdot)$ and $I_1(\cdot)$ are the modified Bessel functions of first kind of zeroth and first order, respectively.

It is important to notice that the PDF shown in Equation (5.10), $\phi(x, \Delta t)$, does not depend on the orientation of the particle at any given time. However, the probability of a specific particle orientation at the start of the time interval changes after a position measurement has been made. Consider the hypothetical situation where there is no measurement error and the particle speed is known. Suppose the initial position of the particle is $x = 0$. If the particle position at time $t = \Delta t$ is measured as $x = +C\Delta t$ (there is a probability of $1/2\exp(-\lambda\Delta t)$ of this occurring) then the particle orientation at $t = \Delta t$ is known with absolute certainty; the particle must be traveling in a positive direction. Furthermore, it now becomes significantly more probable that the particle will be found on the interval $[C\Delta t, 2C\Delta t]$ compared to the interval $[0, C\Delta t]$ at time $t = 2\Delta t$. It is clear that position measurements contain information about the orientation of the particle and bias the probability density for a subsequent measurement.

It is necessary to derive the joint density $\Phi(\mathbf{x}|\mathbf{t}, C, \lambda)$ for a set of particle positions $\mathbf{x}$. We will assume a uniform prior [123] for $(C, \lambda)$ although it is straightforward to use a more complicated function. The conditional posterior PDF, $h_1(C, \lambda|\mathbf{y}, \mathbf{t}, \alpha)$, is given by application of Bayes' Theorem:

$$\begin{aligned} h_1(C, \lambda|\mathbf{y}, \alpha, \mathbf{t}) &= \frac{1}{K_1} r(\mathbf{y}|C, \lambda, \alpha, \mathbf{t}) \\ r(\mathbf{y}|C, \lambda, \alpha, \mathbf{t}) &= \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} \Phi(\mathbf{x}|\mathbf{t}, C, \lambda) \prod_{i=1}^{n_y} f_m(y_i|x_i, \alpha)\, \mathrm{d}\mathbf{x} \end{aligned} \tag{5.11}$$

where the constant,

$$K_1 = \int_0^{\infty} \int_0^{\infty} r(\mathbf{y}|C, \lambda, \alpha, \mathbf{t})\, \mathrm{d}C\, \mathrm{d}\lambda.$$

We will assume a uniform prior for $\alpha$ if $\alpha$ is unknown. Hence, the joint posterior PDF is given by:

$$h_2(C, \lambda, \alpha|\mathbf{y}, \mathbf{t}) = \frac{1}{K_2} r(\mathbf{y}|C, \lambda, \alpha, \mathbf{t}) \,, \tag{5.12}$$

where the constant,

$$K_2 = \int_0^\infty \int_0^\infty \int_0^\infty r(\mathbf{y}|C, \lambda, \alpha, \mathbf{t}) \; \mathrm{d}\alpha \, \mathrm{d}C \, \mathrm{d}\lambda.$$

The marginal posterior PDF can be obtained by integrating the joint posterior PDF:

$$h_3(C, \lambda|\mathbf{y}, \mathbf{t}) = \int_0^\infty h_2(C, \lambda, \alpha|\mathbf{y}, \mathbf{t}) \; \mathrm{d}\alpha. \tag{5.13}$$

However, the classical work [99, 195, 124, 114, 126, 155] derives the PDF, $\phi(x, \Delta t)$, (Equation (5.10)) as the solution of the telegraph equation. Unfortunately, this work does not immediately suggest a method to derive the joint density.

A more general description of the particle motion has been considered by [115] where the motion is described by the stochastic differential equation (SDE):

$$x'(t) = C_1 - C_2 I(t) \,, \quad x(0) = 0, \tag{5.14}$$

where $I(t)$ is a dichotomous alternating renewal stochastic process [49, 127, 115]. $I(t) = 0$ corresponds to the particle moving in a positive direction and $I(t) = 1$ corresponds to the particle moving in a negative direction. The successive sojourn times of $I(t)$ in the states $\{0\}$ and $\{1\}$ are $\{\eta_i\}$ and $\{\xi_i\}$ respectively. The PDF for $\eta_i$ is $g(\eta_i|\mu)$ and the PDF for $\xi_i$ is $f(\xi_i|\lambda)$.

The joint PDF, $\Phi(\mathbf{x}|\mathbf{t}, C_1, C_2, \mu, \lambda)$, can be obtained by application of Bayes' theorem. It is necessary to define the following functions, $\beta_i(x_i, \ldots, x_1)$ and $\gamma_i(x_i, \ldots, x_1)$, which correspond to the following PDFs:

$$\beta_i(x_i, \ldots, x_1) \;=\; p(I(t_i) = 0, x_i, \ldots, x_1|\mathbf{t}, C_1, C_2, \mu, \lambda) \tag{5.15}$$

$$\gamma_i(x_i, \ldots, x_1) \;=\; p(I(t_i) = 1, x_i, \ldots, x_1|\mathbf{t}, C_1, C_2, \mu, \lambda) \,. \tag{5.16}$$

It is also necessary to define the following transition probabilities:

$$\begin{aligned}
p_{11}(d_i) &= p(x_i, I(t_i) = 0 | x_{i-1}, I(t_{i-1}) = 0, C_1, C_2, \mu, \lambda) && (5.17)\\
p_{12}(d_i) &= p(x_i, I(t_i) = 0 | x_{i-1}, I(t_{i-1}) = 1, C_1, C_2, \mu, \lambda) && (5.18)\\
p_{21}(d_i) &= p(x_i, I(t_i) = 1 | x_{i-1}, I(t_{i-1}) = 0, C_1, C_2, \mu, \lambda) && (5.19)\\
p_{22}(d_i) &= p(x_i, I(t_i) = 1 | x_{i-1}, I(t_{i-1}) = 1, C_1, C_2, \mu, \lambda)\,. && (5.20)
\end{aligned}$$

Closed-form expressions for the transition probabilities in Equations (5.17)–(5.20) are derived in § 5.4.1. Defining the transition matrix $\mathbf{P}(d_i)$ as

$$\mathbf{P}(d_i) = \begin{bmatrix} p_{11}(d_i) & p_{12}(d_i) \\ p_{21}(d_i) & p_{22}(d_i) \end{bmatrix}$$

it is possible to use Bayes theorem to write

$$\begin{bmatrix} \beta_i(x_i, \ldots, x_1) \\ \gamma_i(x_i, \ldots, x_1) \end{bmatrix} = \mathbf{P}(d_i) \begin{bmatrix} \beta_{i-1}(x_{i-1}, \ldots, x_1) \\ \gamma_{i-1}(x_{i-1}, \ldots, x_1) \end{bmatrix}.$$

It then follows that the joint PDF for a set of particle positions, $\Phi(\mathbf{x}|\mathbf{t}, C_1, C_2, \mu, \lambda)$, can be expressed by Equation (5.21):

$$\Phi(\mathbf{x}|\mathbf{t}, C_1, C_2, \mu, \lambda) = \begin{bmatrix} 1 & 1 \end{bmatrix} \prod_{i=1}^{n_y} \mathbf{P}(d_i) \begin{bmatrix} p_0(x_0) \\ p_1(x_0) \end{bmatrix} \qquad (5.21)$$

where $p_0(x_0)$ is the PDF for the initial position $x_0$ given $I(0) = 0$, and $p_1(x_0)$ is the PDF for the initial position $x_0$ given $I(0) = 1$. For an alternating renewal process at equilibrium (i.e., after sufficiently long time), the PDFs, $p_0(x_0)$ and $p_1(x_0)$, are given by Equations (5.22)–(5.23) [49]:

$$\begin{aligned}
p_0 &= \frac{\langle \xi_i \rangle}{\langle \xi_i \rangle + \langle \eta_i \rangle} p(x_0|\alpha) && (5.22)\\
p_1 &= \frac{\langle \eta_i \rangle}{\langle \xi_i \rangle + \langle \eta_i \rangle} p(x_0|\alpha) && (5.23)
\end{aligned}$$

where $\langle \xi_i \rangle$ is the mean of $f(\xi_i|\lambda)$, $\langle \eta_i \rangle$ is the mean of $g(\eta_i|\mu)$, and $p(x_0|\alpha)$ is the prior for the position of $x_0$ regardless of orientation.

### 5.4.1 Derivation of Transition PDFs

Although the work of [115] does not present the transition PDFs, the author uses an ingenious construction to derive a solution for $\phi(x, \Delta t)$ which we will use to obtain the transition PDFs. Again, it should be stressed that this model has two states: a continuous state, $x$, the particle position, and a discrete state, $I$, the particle orientation. Equation (3) of [115] states a solution for $\phi(x,t)$:

$$\begin{aligned}
\phi(x,t) \;=\;& \frac{p_0}{C_2}\Pr\left[N_\eta = 0|\hat{t}=t\right]\delta(\Omega_1) + \frac{p_1}{C_2}\Pr\left[N_\xi = 0|\hat{t}=t\right]\delta(\Omega_2) \qquad (5.24)\\
&+ \frac{p_0}{C_2}\left\{\sum_{n=1}^{\infty} g^{(n)}(\Omega_1)\Pr\left[N_\xi = n|\hat{t}=\Omega_2\right]\right.\\
&\left.+\sum_{n=1}^{\infty} f^{(n)}(\Omega_2)\Pr\left[N_\eta = n-1|\hat{t}=\Omega_1\right]\right\}\Theta(\Omega_1)\\
&+ \frac{p_1}{C_2}\left\{\sum_{n=1}^{\infty} g^{(n)}(\Omega_1)\Pr\left[N_\xi = n-1|\hat{t}=\Omega_2\right]\right.\\
&\left.+\sum_{n=1}^{\infty} f^{(n)}(\Omega_2)\Pr\left[N_\eta = n|\hat{t}=\Omega_1\right]\right\}\Theta(\Omega_2)
\end{aligned}$$

where $\Theta(\cdot)$ is the Heaviside step function,

$$\Theta(x) = \begin{cases} 1 & z \geq 0 \\ 0 & z < 0 \end{cases}.$$

$N_\xi$ counts the number of transitions from $I=0$ to $I=1$ up to time $t$, $N_\eta$ counts the number of transitions from $I=1$ to $I=0$, and, $f^n(\cdot)$ is the n-fold convolution defined in Theorem 3.3.5 of Chapter 3. We will show an alternative (but similar) derivation of Equation (5.24) which allows us to derive the transition PDFs. From the solution of Equation (5.14):

$$x = C_1\Omega_2 + (C_1 - C_2)\,\Omega_1 \qquad (5.25)$$

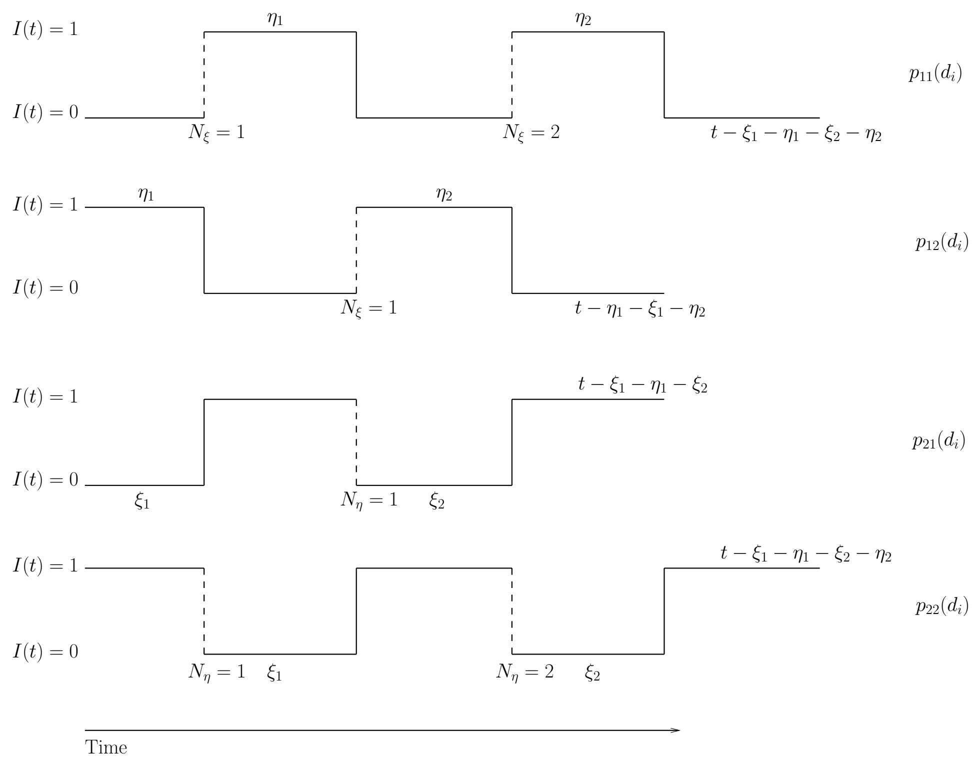


Figure 5-10: Particle orientations at start and end of time interval

where $\Omega_1$ is the total time spent in state $I=1$ and $\Omega_2$ is the total time spent in state $I=0$. By definition,

$$t=\Omega_1+\Omega_2. \tag{5.26}$$

Hence,

$$\begin{aligned}\Omega_1 &= \frac{C_1 t - x}{C_2}\\ \Omega_2 &= \frac{(C_2-C_1)\,t+x}{C_2}.\end{aligned}$$

The different possible combinations of particle orientation at the beginning and end of a time interval, corresponding to the PDFs, $p_{11}(d_i)$, $p_{12}(d_i)$, $p_{21}(d_i)$, and $p_{22}(d_i)$, are shown in Figure 5-10. It is necessary to define the following functions:

$$\begin{aligned}\theta_\eta(N_\xi) &= \eta_1+\eta_2+\ldots+\eta_{N_\xi}, \quad N_\xi=1,2,\ldots\\ \zeta_\eta(N_\xi) &= \eta_1+\eta_2+\ldots+\eta_{N_\xi+1}, \quad N_\xi=1,2,\ldots\end{aligned}$$

$$\begin{aligned}\zeta_\xi(N_\eta) &= \xi_1+\xi_2+\ldots+\xi_{N_\eta+1}, \quad N_\eta=1,2,\ldots\\ \theta_\xi(N_\eta) &= \xi_1+\xi_2+\ldots+\xi_{N_\eta}, \quad N_\eta=1,2,\ldots\end{aligned}$$

The transition probabilities can be obtained from:

$$p_{11}(x) = \delta(\Omega_1)\Pr(N_\xi=0)\left|\frac{\mathrm{d}\Omega_1}{\mathrm{d}x}\right| + \Pr(\theta_\eta=\Omega_1)\left|\frac{\mathrm{d}\Omega_1}{\mathrm{d}x}\right| \tag{5.27}$$

$$p_{12}(x) = \Pr(\zeta_\eta=\Omega_1)\left|\frac{\mathrm{d}\Omega_1}{\mathrm{d}x}\right| \tag{5.28}$$

$$p_{21}(x) = \Pr(\zeta_\xi=\Omega_2)\left|\frac{\mathrm{d}\Omega_2}{\mathrm{d}x}\right| \tag{5.29}$$

$$p_{22}(x) = \delta(\Omega_2)\Pr(N_\eta=0)\left|\frac{\mathrm{d}\Omega_2}{\mathrm{d}x}\right| + \Pr(\theta_\xi=\Omega_2)\left|\frac{\mathrm{d}\Omega_2}{\mathrm{d}x}\right|. \tag{5.30}$$

The PDFs for $\theta_\eta$, $\zeta_\xi$, $\zeta_\eta$, and $\theta_\xi$ can be obtained from Bayes theorem:

$$\Pr(\theta_\eta=\Omega_1) = \sum_{n=1}^{\infty}\Pr(\theta_\eta=\Omega_1|N_\xi=n)\Pr\big(N_\xi=n|\hat{t}=\Omega_2\big) \tag{5.31}$$

$$\Pr(\zeta_\xi=\Omega_2) = \sum_{n=1}^{\infty}\Pr(\zeta_\xi=\Omega_2|N_\eta=n-1)\Pr\big(N_\eta=n-1|\hat{t}=\Omega_2\big) \tag{5.32}$$

$$\Pr(\zeta_\eta=\Omega_1) = \sum_{n=1}^{\infty}\Pr(\zeta_\eta=\Omega_1|N_\xi=n-1)\Pr\big(N_\xi=n-1|\hat{t}=\Omega_1\big) \tag{5.33}$$

$$\Pr(\theta_\xi=\Omega_2) = \sum_{n=1}^{\infty}\Pr(\theta_\xi=\Omega_2|N_\eta=n)\Pr\big(N_\eta=n|\hat{t}=\Omega_1\big). \tag{5.34}$$

The quantities $\Pr(\theta_\eta=\Omega_1|N_\xi=n)$, $\Pr(\zeta_\xi=\Omega_2|N_\eta=n-1)$, etc., can be obtained from Theorem 3.3.5 of Chapter 3.

Making the necessary substitutions in Equations (5.27)–(5.34) yields the following expressions for $p_{11}(x)$, $p_{12}(x)$, $p_{21}(x)$ and $p_{22}(x)$:

$$p_{11}(x) = \frac{1}{C_2}\Bigg\{\Pr[N_\xi=0]\,\delta(\Omega_1) + \Theta(\Omega_1)\sum_{n=1}^{\infty}g^{(n)}(\Omega_1)\Pr\big[N_\xi=n|\hat{t}=\Omega_2\big]\Bigg\} \tag{5.35}$$

$$p_{12}(x) = \frac{\Theta(\Omega_2)}{C_2} \sum_{n=1}^{\infty} g^{(n)}(\Omega_1) \Pr\bigl[N_\xi = n-1 | \hat{t} = \Omega_2\bigr] \tag{5.36}$$

$$p_{21}(x) = \frac{\Theta(\Omega_1)}{C_2} \sum_{n=1}^{\infty} f^{(n)}(\Omega_2) \Pr\left[N_\eta = n-1 | \hat{t} = \Omega_1\right] \tag{5.37}$$

$$p_{22}(x) = \frac{1}{C_2}\Biggl\{ \Pr[N_\eta = 0]\, \delta(\Omega_2) \tag{5.38}$$

$$+ \Theta(\Omega_2) \sum_{n=1}^{\infty} f^{(n)}(\Omega_2) \Pr\bigl[N_\eta = n | \hat{t} = \Omega_1\bigr] \Biggr\}. \tag{5.39}$$

To recover the simpler motion described by [99, 195, 124, 114, 126, 155], set

$$C = C_1 = \frac{C_2}{2}$$

and

$$f(x) = g(x) = \begin{cases} \lambda \exp(-\lambda x) & x \geq 0 \\ 0 & x < 0 \end{cases}.$$

From which it follows:

$$p_{11}(d_i) = \begin{cases} \frac{\exp(-\lambda \Delta t)}{C} \left[\delta\bigl(\Delta t - \frac{d_i}{C}\bigr) + \frac{\lambda^2}{\Gamma_i} I_1(\Gamma_i) \bigl(\Delta t + \frac{d_i}{C}\bigr)\right] & |d_i| \leq C\Delta t \\ 0 & |d_i| > C\Delta t \end{cases} \tag{5.40}$$

$$p_{12}(d_i) = \begin{cases} \exp(-\lambda \Delta t)\, \frac{\lambda}{C} I_0(\Gamma_i) & |d_i| \leq C\Delta t \\ 0 & |d_i| > C\Delta t \end{cases} \tag{5.41}$$

$$p_{21}(d_i) = \begin{cases} \exp(-\lambda \Delta t)\, \frac{\lambda}{C} I_0(\Gamma_i) & |d_i| \leq C\Delta t \\ 0 & |d_i| > C\Delta t \end{cases} \tag{5.42}$$

$$p_{22}(d_i) = \begin{cases} \frac{\exp(-\lambda \Delta t)}{C} \left[\delta\bigl(\Delta t + \frac{d_i}{C}\bigr) + \frac{\lambda^2}{\Gamma_i} I_1(\Gamma_i) \bigl(\Delta t - \frac{d_i}{C}\bigr)\right] & |d_i| \leq C\Delta t \\ 0 & |d_i| > C\Delta t \end{cases} \tag{5.43}$$

where,

$$\Gamma_i = \lambda \sqrt{(\Delta t)^2 - \frac{d_i^2}{C^2}}.$$

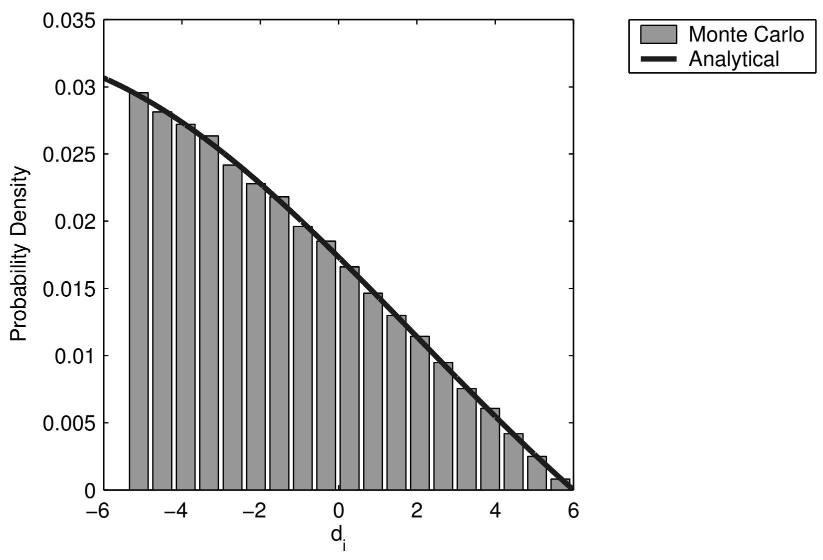


Figure 5-11: Transition PDF, $p_{22}(d_i)$, plotted against $d_i$ for $\lambda = 0.5$, $C = 3$, and $\Delta t = 2$

## 5.4.2 Comparison of Transition PDFs

The transition PDFs derived in § 5.4.1, Equations (5.40)–(5.43) were plotted and compared to Monte Carlo simulation of the correlated random walk (Figures 5-11–5-12). The term,

$$\frac{\exp(-\lambda\Delta t)}{C}\delta\left(\Delta t + \frac{d_i}{C}\right)$$

was omitted from $p_{22}(d_i)$ for clarity of the plot. The PDF, $p_{11}(d_i)$, is not shown since it is a reflection about $d_i = 0$ of the function $p_{22}(d_i)$. It can be seen that there is close agreement between the Monte Carlo simulation and the closed-form solutions for the transition PDFs.

## 5.4.3 Closed-Form Posterior PDF for $\lambda = 0$

It is interesting to consider the estimation of particle speed for the situation where $\lambda = 0$ (i.e., the particle does not turn). The transition PDFs, $p_{11}(d_i)$, $p_{12}(d_i)$, $p_{21}(d_i)$, and $p_{22}(d_i)$ simplify to

$$p_{11}(d_i) \quad = \quad \frac{1}{C}\delta\left(\Delta t - \frac{d_i}{C}\right)$$

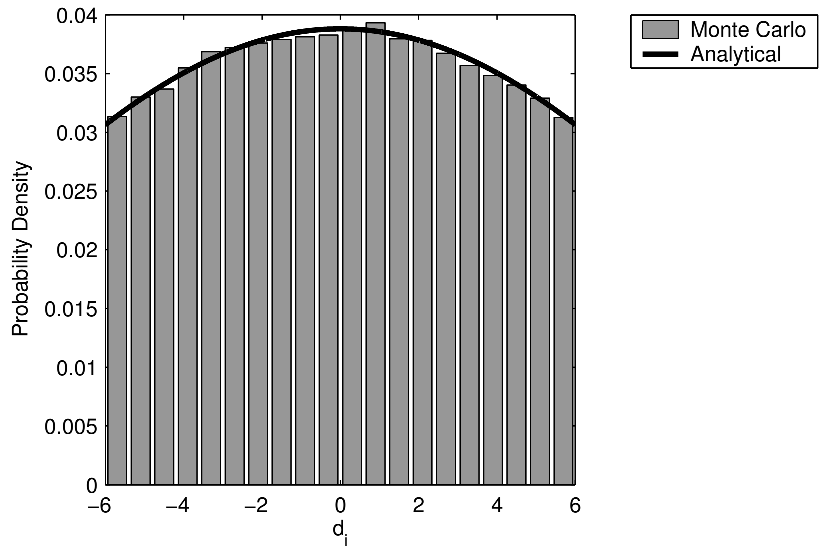


Figure 5-12: Transition PDF, $p_{21}(d_i)$, plotted against $d_i$ for $\lambda = 0.5$, $C = 3$, and $\Delta t = 2$

$$
\begin{aligned}
p_{12}(d_i) &= 0 \\
p_{21}(d_i) &= 0 \\
p_{22}(d_i) &= \frac{1}{C}\delta\left(\Delta t + \frac{d_i}{C}\right).
\end{aligned}
$$

The joint density, $\Phi(\mathbf{x}|\mathbf{t}, C\lambda)$, is therefore,

$$\Phi(\mathbf{x}|\mathbf{t}, C\lambda) = p_0(x_0|\alpha)\prod_{i=1}^{n_y}\frac{1}{C}\delta\left(\Delta t - \frac{d_i}{C}\right) + p_1(x_0|\alpha)\prod_{i=1}^{n_y}\frac{1}{C}\delta\left(\Delta t + \frac{d_i}{C}\right)$$

where,

$$p_0(x_0|\alpha) = p_1(x_0|\alpha) = \frac{1}{2\alpha\sqrt{2\pi}}\exp\left(-\frac{x_0^2}{2\alpha^2}\right).$$

If it is assumed that the PDF for the particle measurement, $f_m(\cdot)$, is given by

$$f_m\big(y_i|x_i, \alpha^2\big) = N\big(x_i, \alpha^2\big), \quad i = 1, \ldots, n_y$$

where $\alpha$ is a parameter characterizing the magnitude of the error in the measurement, the integral in Equation (5.11) can be computed in closed-form:

$$\begin{aligned} r(\mathbf{y}|C,\Delta t,\alpha) &= \frac{1}{2\left(\alpha\sqrt{2\pi}\right)^{n_y}\sqrt{n_y+1}}\left[\exp\left(\frac{\left(S_y\right)^2-\left(n_y+1\right)S_{yy}}{2\left(n_y+1\right)\alpha^2}\right)\right. \\ &\quad \left. + \exp\left(\frac{\left(\overline{S}_y\right)^2-\left(n_y+1\right)\overline{S}_{yy}}{2\left(n_y+1\right)\alpha^2}\right)\right] \end{aligned} \tag{5.44}$$

where,

$$\begin{aligned} S_y &= \sum_{i=1}^{n_y} y_i - iC\Delta t, \\ S_{yy} &= \sum_{i=1}^{n} \left(y_i - iC\Delta t\right)^2, \\ \overline{S}_y &= \sum_{i=1}^{n_y} y_i + iC\Delta t, \\ \overline{S}_{yy} &= \sum_{i=1}^{n} \left(y_i + iC\Delta t\right)^2. \end{aligned}$$

It follows that the posterior PDF is given by:

$$h_1(C|\mathbf{y},\alpha,\Delta t) = \begin{cases} \frac{1}{K_2}\left[\exp\left(-\frac{(C-\theta)^2}{2\sigma^2}\right) + \exp\left(-\frac{(C+\theta)^2}{2\sigma^2}\right)\right] & C > 0 \\ 0 & C \leq 0 \end{cases} \tag{5.45}$$

where,

$$\theta = \frac{12\left(n_y/2\sum_{i=1}^{n_y} y_i - \sum_{i=1}^{n_y} iy_i\right)}{n_y\left(n_y+1\right)\left(n_y+2\right)} \tag{5.46}$$

$$\sigma^2 = \frac{12\alpha^2}{\left(\Delta t\right)^2 n_y\left(n_y+1\right)\left(n_y+2\right)}, \tag{5.47}$$

and,

$$K_2 = \int_0^\infty \exp\left(-\frac{\left(C-\theta\right)^2}{2\sigma^2}\right) + \exp\left(-\frac{\left(C+\theta\right)^2}{2\sigma^2}\right)\, \mathrm{d}C.$$

It can be seen from Equation (5.45) that the density is unimodal and the width of the peak decreases roughly $n_y^{-3/2}$.

### 5.4.4 Numerical Evaluation of Posterior PDF

A closed-form solution for $r(\mathbf{y}|C, \lambda, \alpha, \mathbf{t})$ does not exist. Hence, it is necessary to evaluate the multi-dimensional integral shown in Equation (5.11) numerically. Three different methods were examined for computing the integral:

1. integration by importance sampling,
2. integration by validated methods, and,
3. integration by iterated use of the extended Trapezoid rule.

Numerical difficulties were encountered with the first two methods for this specific problem. In comparison, the third method work fairly efficiently on this problem.

#### Integration by Importance Sampling

Importance sampling is one of several Monte Carlo methods for evaluating a complex high-dimensional integral [176]. To exploit the method it must be possible to write the multi-dimensional integral as:

$$I \equiv \int_V f(\mathbf{x})\, p(\mathbf{x})\ \mathrm{d}\mathbf{x}, \tag{5.48}$$

where,

$$\int_V p(\mathbf{x})\ \mathrm{d}\mathbf{x} = 1.$$

The function $p(\mathbf{x})$ can be interpreted as a probability density function. It follows that if probability density can be efficiently sampled: $\mathbf{x}_1, \ldots \mathbf{x}_n$, the integral can be approximated by

$$I \approx \sum_{i=1}^{n} f(\mathbf{x}_i)\,.$$

For a general integral,

$$I \equiv \int_V h(\mathbf{x})\ \mathrm{d}\mathbf{x},$$

the method can be implemented by setting $f = h/p$. Rigorous bounds on the error of integration do not exist. However, an estimate of the error in Equation (5.48) can

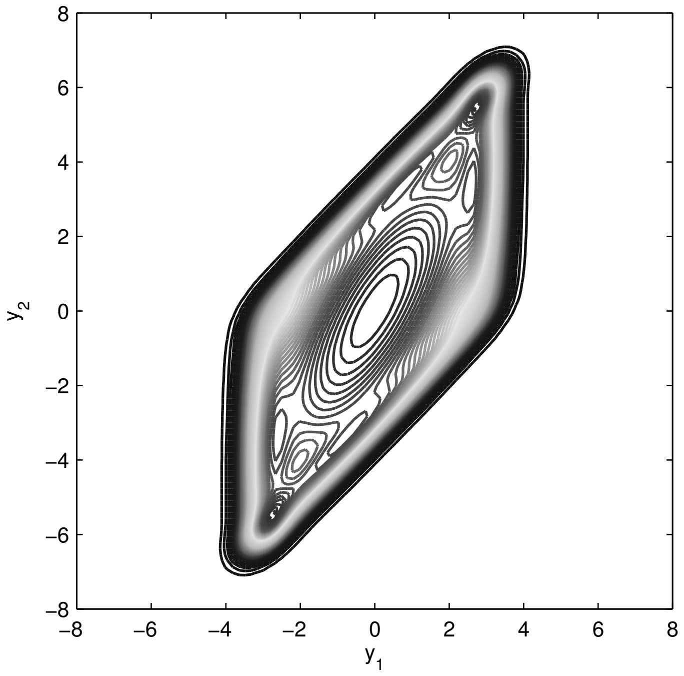


Figure 5-13: Contours of $r(y_1, y_2|C = 3, \lambda = 1.5, \Delta t = 1, \alpha = 0.3)$

be obtained from

$$\sqrt{\frac{1}{N}\left(\sum_{i=1}^{n} f^2(\mathbf{x}_i) - \left(\sum_{i=1}^{n} f(\mathbf{x}_i)\right)^2\right)}.$$

For the method to be effective, it is necessary that $f$ is relatively constant for values of $\mathbf{x}$ that correspond to high probability regions of $p$.

Samples were generated from $\Phi(\mathbf{x}|\mathbf{t}, C, \lambda)$ to calculate the integral shown in Equation (5.11). The choice of $\Phi$ is natural as it is possible to generate samples quickly. The code to generate samples from $\Phi(\cdot)$ is shown in Appendix A, § A.5. Unfortunately, the PDF $\Phi$ is a multi-modal function of $\mathbf{x}$ with many peaks. Each peak corresponds with different sequences of changes in direction. The likelihood function is effectively a blurred form of $\Phi(\cdot)$. A plot of the corresponding likelihood function for two measurements, $(y_1, y_2)$, is shown in Figure 5-13. It was found that the importance sampling integration failed to converge in a reasonable number of iterations. For a more comprehensive study of the computational difficulties associated with using a Monte Carlo integration method on a multi-modal integrand see [105].

**Integration by Validated Methods**

Another popular approach to calculating multi-dimensional integrals is verified integration, based on interval arithmetic [158, 47]. These methods are attractive since symbolic analysis yields higher order information which can be used to target values of parameters where function evaluations should occur. Furthermore, exact bounds on the error in the approximated integral are available at each iteration of the algorithm. For a comparison of Monte Carlo and verified methods see [21]. Most verified methods rely on constructing an interval Taylor polynomial (Definition 5.4.1).

**Definition 5.4.1.** [148, 20] Let $f$ be a $C^{n+1}$ mapping on $D_f \subset \mathbb{R}^\nu$, and $B = [a_1, b_1] \times \cdots \times [a_\nu, b_\nu] \subset D_f$ be an interval box containing $\mathbf{x}_0$. Let $T$ be the Taylor polynomial of $f$ around $\mathbf{x}$. An interval Taylor polynomial is defined as $(T, I)$ where,

$$f(\mathbf{x}) - T(\mathbf{x}) \in I, \quad \forall \mathbf{x} \in B.$$

A basic premise of the interval Taylor methods is that it is straightforward to calculate higher order derivatives of the elementary functions used to make $f$. It is computationally infeasible to derive these derivatives symbolically since for many elementary functions there is an explosion in the computational complexity for evaluating the derivatives. Instead, it is possible to derive recursive expressions for the high-order derivatives using Automatic Differentiation (AD) (see Pages 24–29 of [158]).

It was attempted to construct a Taylor polynomial integration scheme for the integral in Equation (5.11). Consequently, it was necessary to obtain expressions for the high-order derivatives of the modified Bessel functions of first kind, $I_0$ and $I_1$. The recursion for the $k^{\text{th}}$ derivative of $I_0$ evaluated at $x_0 \neq 0$ was derived to be

$$(f)_{k+3} = \frac{(f)_k + (f)_{k+1}\, x_0 - (f)_{k+2}\, (k+2)^2}{x_0\, (k+3)\, (k+2)}, \tag{5.49}$$

where $(f)_k$ denotes the $k^{\text{th}}$ derivative of $I_0$. Unfortunately, for $|x_0| < 1$ this recursion is unstable numerically. To demonstrate the difficulty the expression was evaluated using GNU multi-precision library (GMP 4.1). It can be seen from Table 5.4.4 that

Table 5.1: Taylor coefficients for $I_0(x)$ expanded around $x_0 = 0.001$

| 16 digit mantissa | | 256 digit mantissa | |
|---|---|---|---|
| Coefficient | Value | Coefficient | Value |
| 7 | 0.177197E-05 | 7 | 0.542535E-07 |
| 8 | -0.149622E-02 | 8 | 0.678169E-05 |
| 9 | 0.133600E+01 | 9 | 0.678169E-09 |
| 10 | -0.120240E+04 | 10 | 0.6781687E-07 |
| 11 | 0.109309E+07 | 11 | 0.565140E-11 |
| 12 | -0.100200E+10 | 12 | 0.470950E-9 |
| 13 | 0.924925E+12 | 13 | 0.3363931E-13 |

for a reasonable size of mantissa the derivatives wildly fluctuate from the true values. This difficulty can be ameliorated by moving to higher precision. However, it was decided that the additional effort required to evaluate the integrand in multi-precision arithmetic was not worthwhile. Consequently, this approach was abandoned.

### Exploiting Structure in Integrand

The final method for computing the integral in Equation (5.11) relies on special structure of the integrand. The integral can be expressed as a sequence of one-dimensional integrals:

$$\psi_1(x_1) = \int_{-\infty}^{\infty} p_{11}(x_1 - x_0)\, p_0(x_0) + p_{12}(x_1 - x_0)\, p_1(x_0)\ \mathrm{d}x_0 \tag{5.50}$$

$$\gamma_1(x_1) = \int_{-\infty}^{\infty} p_{21}(x_1 - x_0)\, p_0(x_0) + p_{22}(x_1 - x_0)\, p_1(x_0)\ \mathrm{d}x_0 \tag{5.51}$$

$$\psi_i(x_{i+1}) = \int_{-\infty}^{\infty} f_m(x_i)\, (p_{11}(x_{i+1} - x_i)\, \psi_{i-1} + p_{12}(x_i - x_i)\, \gamma_{i-1})\ \mathrm{d}x_i \tag{5.52}$$

$$\gamma_i(x_{i+1}) = \int_{-\infty}^{\infty} f_m(x_i)\, (p_{21}(x_i - x_i)\, \psi_{i-1} + p_{22}(x_{i+1} - x_i)\, \gamma_{i-1})\ \mathrm{d}x_i \tag{5.53}$$

$$r(\mathbf{y}|C, \lambda, \alpha, \mathbf{t}) = \int_{-\infty}^{\infty} f_m\big(x_{n_y}\big)\, \big(\gamma_{n_y-1} + \psi_{n_y-1}\big)\, \mathrm{d}x_{n_y} \tag{5.54}$$

where if $\lambda = \mu$, the priors for $x_0$ are

$$p_0(x_0) = p_1(x_0) = \frac{1}{2} p(x_0|\alpha)\,.$$

Each one-dimensional integral can be calculated using the extended Trapezoid approximation [46]. The Dirac-delta terms in $p_{11}(\cdot)$ and $p_{22}(\cdot)$ can be handled analytically using the identities:

$$\frac{1}{C}\int_{-\infty}^{\infty}\delta\left(\Delta t+\frac{x_{i+1}-x_i}{C}\right)f(x_i)\,\mathrm{d}x_i=f(x_{i+1}+C\Delta t) \tag{5.55}$$

and

$$\frac{1}{C}\int_{-\infty}^{\infty}\delta\left(\Delta t-\frac{x_{i+1}-x_i}{C}\right)f(x_i)\,\mathrm{d}x_i=f(x_{i+1}-C\Delta t) \tag{5.56}$$

for $C>0$.

The scheme outlined in Equations (5.50)–(5.54) requires a total of $O\big(n_y\cdot n_q^2\big)$ operations, where $n_q$ is the number of quadrature points used in evaluating a one-dimensional integral. The error in the approximation of the iterated integral by repeated application of the extended Trapezoid approximation is given by Theorem 5.4.1.

**Theorem 5.4.1.** *The error in $r(\mathbf{y}|C,\lambda,\alpha,\Delta t)$ can be bounded by the expression given in Equation (5.57):*

$$\begin{aligned}|r(\mathbf{y}|C,\lambda,\alpha,\Delta t)-s_N^r| &\leq e_T^r+e_A^r \quad (5.57)\\ &= \sum_{i=1}^{n_y}C_i w^{n_y-i}\left(1-\int_{a_i}^{b_i}f_m\big(y_i|x_i,\alpha^2\big)\,\mathrm{d}x_i\right)\\ &\quad + C_{n_y+1}w^{n_y}\left(1-\int_{a_0}^{b_0}p(x_0|\alpha)\,\mathrm{d}x_0\right) \quad (5.58)\\ &\quad + h^2\sum_{i=0}^{n_y}B_i w^{n_y-i}\end{aligned}$$

*where the integration limits are $a_i=-w/2+y_i$ and $b_i=w/2+y_i$. $s_N^f$ is the extended Trapezoid approximation to the integral:*

$$F=\int_a^b f\,\mathrm{d}x,$$

*and is defined as*

$$s_N^f = \frac{h}{2}\left[f_0 + f_N + 2\sum_{i=1}^{N-1} f_i\right],$$

*where $f_s = f(a+sh)$ with $h = (b-a)/N$.*

*Proof.* The quantities $\psi_1(x_1)$ and $\gamma_1(x_1)$ are given by:

$$\psi_1(x_1) \;=\; s_N^{\psi_1}(x_1) + e_Q^{\psi_1} + e_T^{\psi_1} + e_A^{\psi_1} \tag{5.59}$$

$$\gamma_1(x_1) \;=\; s_N^{\gamma_1}(x_1) + e_Q^{\gamma_1} + e_T^{\gamma_1} + e_A^{\gamma_1} \tag{5.60}$$

where $s_N^{\psi_1}(x_1)$ and $s_N^{\gamma_1}(x_1)$ are the extended Trapezoid approximations to the integrals shown in Equations (5.50)–(5.51), $e_A^{\psi_1}$ and $e_A^{\gamma_1}$ are the errors from approximating the integrand, and $e_Q^{\psi_1}$ and $e_Q^{\gamma_1}$ are the errors in the quadrature rule. The errors in approximating the integrands are $e_A^{\psi_1} = 0$ and $e_A^{\gamma_1} = 0$. The quadrature errors are given by [46]:

$$\begin{aligned}\left|e_Q^{\psi_1}\right| &\leq k_Q^{\psi_1} h^2\\ \left|e_Q^{\gamma_1}\right| &\leq k_Q^{\gamma_1} h^2.\end{aligned}$$

The errors from truncating limits of the integrals are given by:

$$\begin{aligned}\left|e_T^{\psi_1}\right| &\leq k_T^{\psi_1}\left(1 - \int_{a_0}^{b_0} p(x_0|\alpha)\,\mathrm{d}x_0\right)\\ \left|e_T^{\gamma_1}\right| &\leq k_T^{\gamma_1}\left(1 - \int_{a_0}^{b_0} p(x_0|\alpha)\,\mathrm{d}x_0\right),\end{aligned}$$

where $k_T^{\psi_1}$ and $k_T^{\gamma_1}$ are determined by:

$$k_T^{\psi_1} = \max_{x_0,x_1\in\mathbb{R}^2} \frac{1}{2}\left(p_{11}(x_1 - x_0) + p_{12}(x_1 - x_0)\right)$$

and

$$k_T^{\gamma_1} = \max_{x_0,x_1\in\mathbb{R}^2} \frac{1}{2}\left(p_{21}(x_1 - x_0) + p_{22}(x_1 - x_0)\right).$$

The errors in $\psi_i(x_{i+1})$ and $\gamma_i(x_{i+1})$ are given by Equations (5.61)–(5.62):

$$\begin{aligned}
\psi_i(x_{i+1}) &= s_N^{\psi_i}(x_{i+1}) + e_Q^{\psi_i} + e_T^{\psi_i} + e_A^{\psi i} && (5.61)\\
\gamma_i(x_{i+1}) &= s_N^{\gamma_i}(x_{i+1}) + e_Q^{\gamma_i} + e_T^{\gamma_i} + e_A^{\gamma i}. && (5.62)
\end{aligned}$$

The error terms $e_Q^{\psi_i}$, $e_T^{\psi_i}$, $e_A^{\psi_i}$, $e_Q^{\gamma_i}$, $e_T^{\gamma_i}$, and $e_A^{\gamma_i}$ are given by:

$$\begin{aligned}
\left|e_Q^{\psi_i}\right| &\leq k_Q^{\psi_i} h^2\\
\left|e_T^{\psi_i}\right| &\leq k_T^{\psi_i}\left(1 - \int_{a_i}^{b_i} f_m(y_i|x_i,\alpha)\,\mathrm{d}x_i\right)\\
\left|e_A^{\psi_i}\right| &\leq w\left[k_{A_1}^{\psi_i}\left(e_Q^{\psi_{i-1}} + e_T^{\psi_{i-1}} + e_A^{\psi_{i-1}}\right) + k_{A_2}^{\psi_i}\left(e_Q^{\gamma_{i-1}} + e_T^{\gamma_{i-1}} + e_A^{\gamma_{i-1}}\right)\right]\\
\left|e_Q^{\gamma_i}\right| &\leq k_Q^{\gamma_i} h^2\\
\left|e_T^{\gamma_i}\right| &\leq k_T^{\gamma_i}\left(1 - \int_{a_i}^{b_i} f_m(y_i|x_i,\alpha)\,\mathrm{d}x_i\right)\\
\left|e_A^{\gamma_i}\right| &\leq w\left[k_{A_1}^{\gamma_i}\left(e_Q^{\psi_{i-1}} + e_T^{\psi_{i-1}} + e_A^{\psi_{i-1}}\right) + k_{A_2}^{\gamma_i}\left(e_Q^{\gamma_{i-1}} + e_T^{\gamma_{i-1}} + e_A^{\gamma_{i-1}}\right)\right]
\end{aligned}$$

where,

$$\begin{aligned}
k_T^{\psi_i} &= \max_{x_i,x_{i+1}\in\mathbb{R}^2} p_{11}(x_{i+1}-x_i)\,\psi_{i-1}(x_i) + p_{12}(x_i-x_i)\,\gamma_{i-1}(x_i)\\
k_T^{\gamma_i} &= \max_{x_i,x_{i+1}\in\mathbb{R}^2} p_{21}(x_{i+1}-x_i)\,\psi_{i-1}(x_i) + p_{22}(x_i-x_i)\,\gamma_{i-1}(x_i)\\
k_{A_1}^{\psi_i} &= \max_{x_i,x_{i+1}\in\mathbb{R}^2} f_m(x_i)\,p_{11}(x_{i+1}-x_i)\\
k_{A_2}^{\psi_i} &= \max_{x_i,x_{i+1}\in\mathbb{R}^2} f_m(x_i)\,p_{12}(x_{i+1}-x_i)\\
k_{A_1}^{\gamma_i} &= \max_{x_i,x_{i+1}\in\mathbb{R}^2} f_m(x_i)\,p_{21}(x_{i+1}-x_i)\\
k_{A_2}^{\gamma_i} &= \max_{x_i,x_{i+1}\in\mathbb{R}^2} f_m(x_i)\,p_{22}(x_{i+1}-x_i)\,.
\end{aligned}$$

Finally, the error in $r(\mathbf{y}|C,\lambda,\alpha,\Delta t)$ is given by:

$$r(\mathbf{y}|C,\lambda,\alpha,\Delta t) = s_N^r\left(x_{n_y}\right) + e_Q^r + e_T^r + e_A^r$$

where,

$$\begin{aligned}
\left|e_Q^r\right| &\leq k_Q^r h^2 \\
\left|e_T^r\right| &\leq k_T^r \left(1 - \int_{a_{n_y}}^{b_{n_y}} f_m\big(y_{n_y}|x_{n_y}, \alpha^2\big)\, \mathrm{d}x_{n_y}\right) \\
\left|e_A^r\right| &\leq k_A^r w \left(e_Q^{\psi_{n_y-1}} + e_T^{\psi_{n_y-1}} + e_A^{\psi_{n_y-1}} + e_Q^{\gamma_{n_y-1}} + e_T^{\gamma_{n_y-1}} + e_A^{\gamma_{n_y-1}}\right)
\end{aligned}$$

and,

$$\begin{aligned}
k_T^r &= \max_{x_{n_y}\in\mathbb{R}} \psi_{n_y-1}\big(x_{n_y}\big) + \gamma_{n_y-1}\big(x_{n_y}\big)\,, \\
k_A^r &= \max_{x_{n_y}\in\mathbb{R}} f_m\big(x_{n_y}\big)\,.
\end{aligned}$$

The result shown in Equation (5.57) follows by making the necessary substitutions and collecting terms. □

*Remark.* If the PDFs for the measurement error and prior for $x_0$ have compact support: $f_m(y_i) : [a_i, b_i] \to \mathbb{R}_+$, and $p(x_0) : [a_0, b_0] \to \mathbb{R}_+$ the term in Equation (5.57) due to the truncation of the integration limits:

$$e_T = \sum_{i=1}^{n_y} C_i w^{n_y-i} \left(1 - \int_{a_i}^{b_i} f_m(y_i|x_i, \alpha)\, \mathrm{d}x_i\right) + C_{n_y+1} w^{n_y} \left(1 - \int_{a_0}^{b_0} p(x_0|\alpha)\, \mathrm{d}x_0\right),$$

will be precisely zero. If the PDF is defined on $\mathbb{R}$, the term can be reduced by letting $w \to \infty$ if the limits

$$\lim_{a_i\to-\infty} \int_{a_i}^{\infty} f_m(y_i|x_i, \alpha)\, \mathrm{d}x_i = 1$$

and

$$\lim_{b_i\to\infty} \int_{-\infty}^{b_i} f_m(y_i|x_i, \alpha)\, \mathrm{d}x_i = 1,$$

(and the same for $p(x_0|\alpha)$) are achieved sufficiently quickly. For example, if

$$f_m\big(y_i|x_i, \alpha^2\big) = N\big(x_i, \alpha^2\big)\,,$$

and $p(x_0|\alpha) = N(0,\alpha^2)$, the error term can be approximated by:

$$\begin{aligned} e_T &= \sum_{i=0}^{n_y} C_i w^{n_y-i}\left(1-\operatorname{erf}\left(\frac{w}{2\alpha\sqrt{2}}\right)\right) \\ &\le \sum_{i=1}^{n_y+1} 2\alpha C_i\sqrt{\frac{2}{\pi}}w^{n_y-i}\exp\left(-\frac{w^2}{8\alpha^2}\right), \end{aligned}$$

which clearly satisfies this property. The term in Equation (5.57) which comes from the approximation error:

$$e_A = h^2\sum_{i=1}^{n_y} B_i w^{n_y-1},$$

can be made small by selecting a sufficient number of quadrature points, $n_q$, where

$$h = \frac{w}{n_q-1}.$$

### 5.4.5 Results

A noisy correlated random walk was simulated (Figure 5-14) for $\lambda = 0$ and the posterior PDF was evaluated according to the closed-form solution, (Equation (5.45, § 5.4.3)). The posterior PDF was also evaluated using the numerical scheme outlined in Equations (5.50)–(5.54) of § 5.4.4. It can be seen from the results shown in Figure 5-15 that there is close agreement between the numerical solution and the closed-form solution. It can be seen from the posterior PDF that it does not require many measurements to obtain a fairly accurate estimate of speed.

A noisy correlated random walk was also simulated for $C = 3$, $\lambda = 0.6$, $\alpha = 0.1$ and $\alpha = 1$. The modified Bessel functions were calculated with FNLIB [90]. The simulations are shown in Figures 5-16 and 5-18, respectively. The posterior PDFs were calculated numerically and contour plots are shown in Figures 5-17–5-19. It can be seen for the situation where $\alpha = 0.1$, the contours of the posterior PDF (Figure 5-17) are tightly centered around the simulation values; i.e., it is relatively easy to estimate both the particle speed and turning frequency to good accuracy. In contrast, when $\alpha = 1$ (there is more error in the particle measurement), the contours

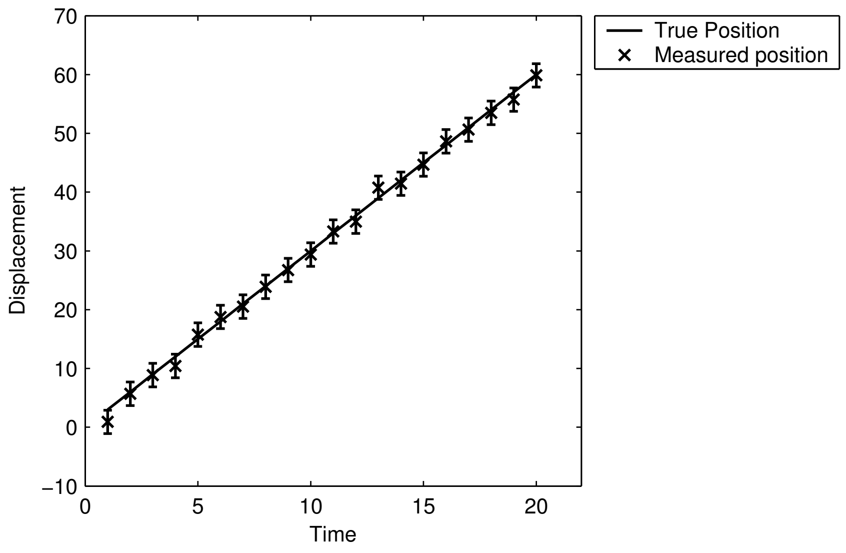


Figure 5-14: Simulated correlated random walk for $C = 3$, $\lambda = 0$, $\alpha = 1$, $n_y = 20$

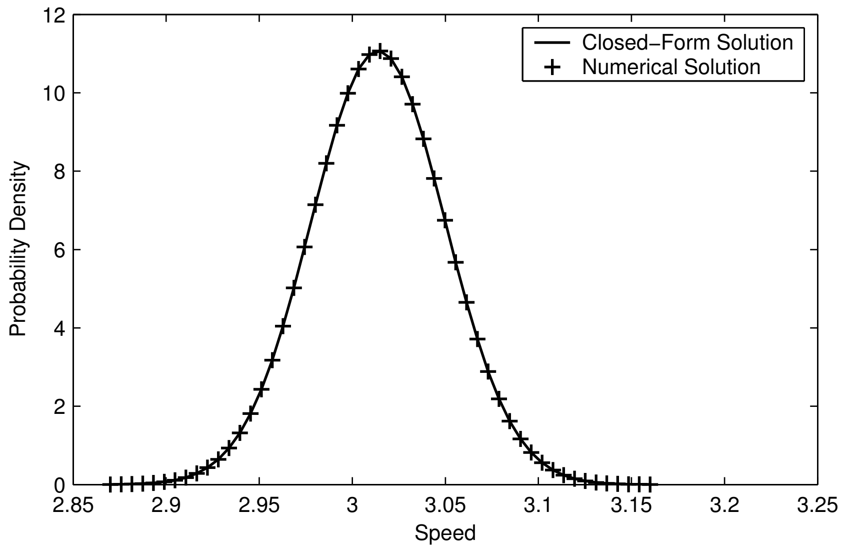


Figure 5-15: Posterior PDF for particle speed

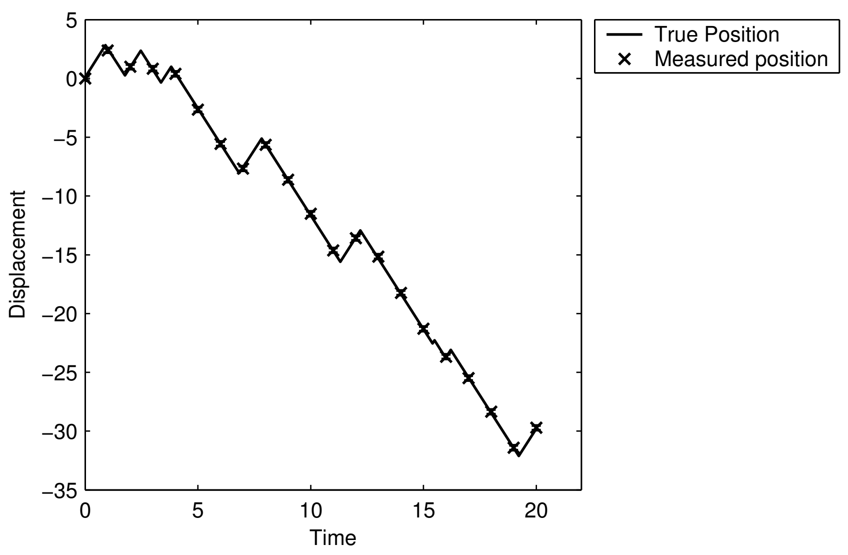


Figure 5-16: Simulated correlated random walk for $C = 3$, $\lambda = 0.6$, $\alpha = 0.1$, $n_y = 20$

of the posterior PDF (Figure 5-19) are wide and there a positive correlation between $C$ and $\lambda$. Unfortunately, this is an inescapable consequence of the problem. The Bayesian estimate is optimal and the wide contours are fundamental to the problem; it is harder to estimate speed and turning frequency with noisy data. It was attempted to reparameterize the problem in terms of $\omega = \lambda$, and $D = C^2/\lambda$ (how to make this transformation is described in Theorem 3.3.4 of Chapter 3). i.e. Derive the posterior PDF $\tilde{h}(D, \omega|\mathbf{y}, \alpha, \mathbf{t})$. The rationale was that in the limit $\Delta t \to \infty$ the parameter $D$ corresponds to a diffusion coefficient. However, this reparameterization is very misleading. The contours of $\tilde{h}(D, \omega|\mathbf{y}, \alpha, \mathbf{t})$ are fairly tight ellipses. However, the mode of the PDF is a long distance from the simulation values of $(D, \omega)$ used to generate the data $\mathbf{y}$. In general, we prefer keeping the original parameterization of the problem. The wide contours of the PDF serve to warn that the estimate will be misleading.

### 5.4.6 Experimental Design

It is particularly interesting to characterize the conditions under which the proposed parameter estimation scheme will be successful. Typically, the output of an engi-

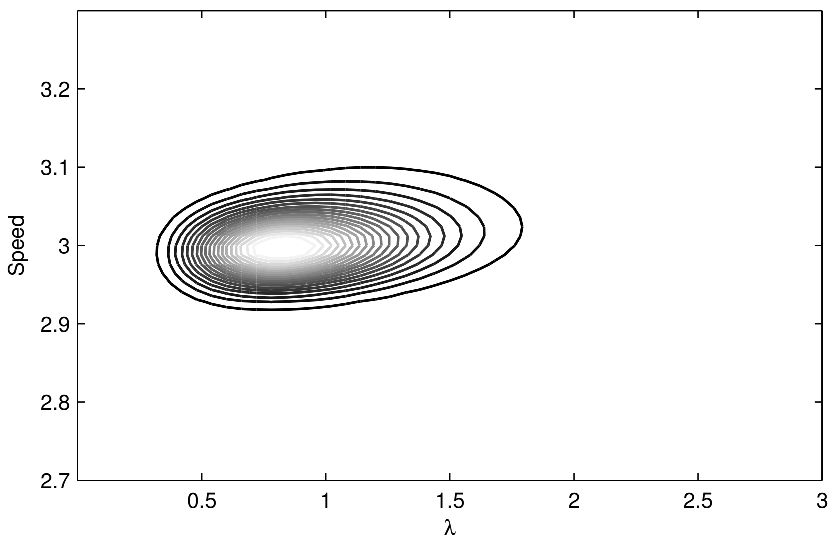


Figure 5-17: Posterior PDF for $h_1(C, \lambda | \alpha = 0.1, \mathbf{y}, \mathbf{t})$

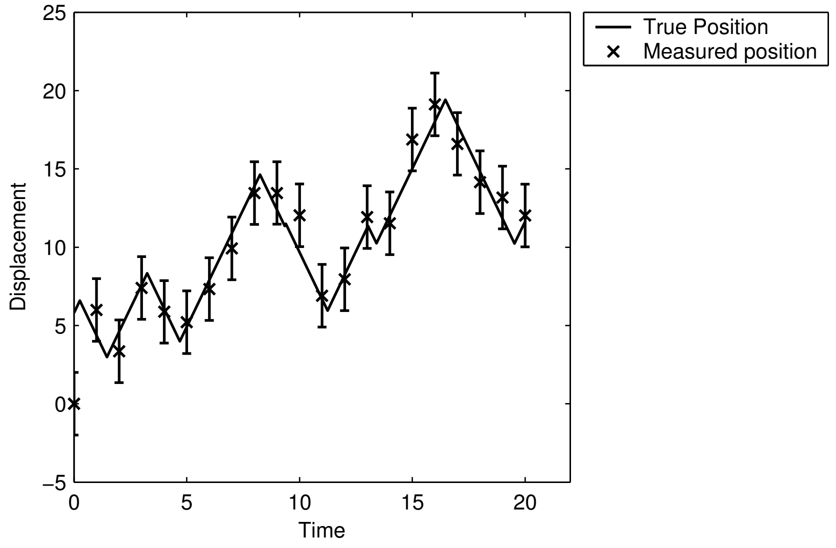


Figure 5-18: Simulated correlated random walk for $C = 3$, $\lambda = 0.6$, $\alpha = 1$, $n_y = 20$

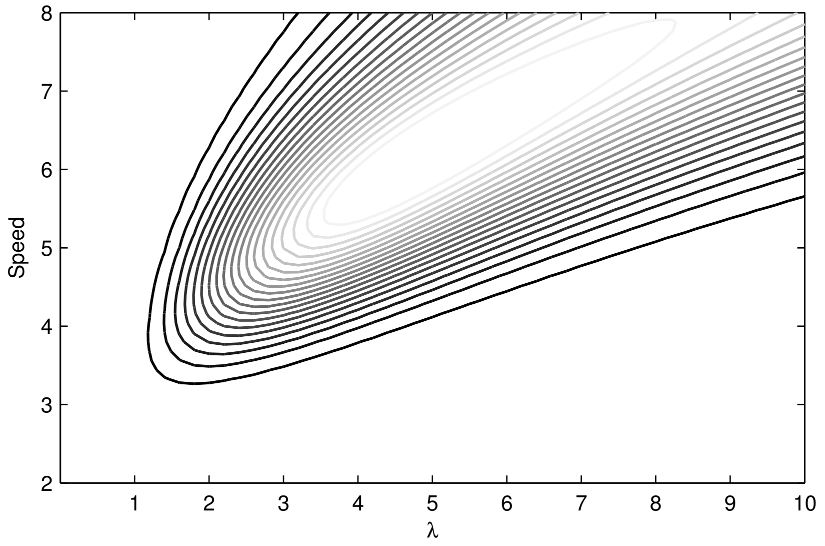


Figure 5-19: Posterior PDF for $h_1(C, \lambda | \alpha = 1, \mathbf{y}, \mathbf{t})$

neering system is a reproducible function of the inputs. Each experiment at fixed input conditions is likely to yield a similar amount of information. Hence, for many engineering experimental design problems, once the input conditions have been set, it is possible to estimate how much information is contained in a set of experimental measurements. The goal is to pick the input conditions to yield the most valuable information.

Unfortunately, the stochastic nature of cell migration changes the problem qualitatively. For fixed input conditions, the outputs are *not* a reproducible function of the inputs; i.e., it is not possible to calculate *a priori* how much information a data point $y_i$ is likely to contain. However, it is possible to describe qualitatively circumstances that will lead to accurate parameter estimates and circumstances that will not. For a given cell speed, $C$, turning frequency, $\lambda$, measurement error, $\alpha$, and sampling times, $\mathbf{t}$, there is a certain probability an experimental data set, $\mathbf{y}$, is collected that is relatively informative about the true values of speed and turning frequency and a certain probability that the data set is uninformative. We will refer to this stochastic effect as the uninformative likelihood function. The effect is described in § 5.4.7. For the correlated random walk, it is easy to determine qualitatively when there will be a high probability of collecting useful data $\mathbf{y}$. This will be explained in more detail in § 5.4.8.

### 5.4.7 Uninformative Likelihood Functions

The work of [86] gives a comprehensive review of when the likelihood function does not contain sufficient information about the values of the parameters to be estimated. The author cites an example due to [178] that is instructive. In this example, a set of independent measurements $\mathbf{y}$ are made where the individual measurement obeys the likelihood function:

$$\mathrm{f}_y(y_i|x,\sigma) = \frac{1}{2}\varphi(y_i) + \frac{1}{2\sigma}\varphi\left(\frac{y_i - x}{\sigma}\right), \tag{5.63}$$

and $\varphi(\cdot)$ is a standard Normal density $N(0,1)$. It has been shown that the maximum likelihood estimate of $(x,\sigma)$ does not converge to the true values of the parameters. A description due to [25] gives the best interpretation of why this occurs. Another way to express the likelihood function is

$$\mathrm{f}_y(y_i|x,\sigma,v_i) = \frac{1}{\sigma v_i + (1-v_i)}\varphi\left(\frac{y_i - xv_i}{\sigma v_i + (1-v_i)}\right)$$

where $v_i$ is unobserved, $v_i \in \{0,1\}$, and $\Pr(v_i = 0) = \Pr(v_i = 1) = 1/2$. Stated this way the difficulty is apparent. If $n_y$ measurements are made, there is a $(1/2)^{n_y}$ chance that $v_i = 0$ for all $i = 1,\ldots,n_y$. In this situation, the likelihood function does not depend on $(x,\sigma)$. A Bayesian problem formulation can help the situation if there is a significant amount of prior information about $(x,\sigma)$. In this formulation, if the data are uninformative, the posterior PDF will be dominated by the contribution from the prior PDF for $(x,\lambda)$. Unfortunately, there is no solution to the problem if there is no prior information; the Bayesian formulation faithfully reports ignorance. The example due to [178] is perhaps the most extreme kind of an uninformative likelihood function. We will discuss the less pathological example of the correlated random walk in § 5.4.8.

### 5.4.8 Parameter Estimation for a Correlated Random Walk

An experimental design consists of a set of values for the inputs to the system. Direct simulation can be used to verify whether a proposed experimental design will work. The steps of the procedure are outlined below:

1. The process is simulated for reasonable guesses of the model parameters and the proposed experimental design.

2. The resulting simulation data is then used to generate the posterior PDF for the parameters.

3. The posterior PDF is checked to see that an accurate estimate of the parameters can be generated.

4. The procedure is repeated several times at the same parameter values to ensure the design consistently works.

5. The procedure is repeated for slightly different parameter values to check it is insensitive to poor estimates of the model parameters.

It is straightforward to implement this scheme for the correlated random walk based on the algorithms developed in § 5.4.1–5.4.4 and it is the preferred method for testing an experimental design.

Several different experimental designs for a correlated random walk were tested. It is hard to summarize the conclusions quantitatively. However, it was found that the posterior PDF generated for some fixed parameter values would dramatically change shape between different runs; i.e., sometimes the collected data would yield useful information about the values of the parameters and sometimes the collected data would not. The two most important parameters that affected the quality of the parameter estimates were measurement error and turning frequency. This is not too surprising since it is difficult to distinguish between a cell that is turning and a cell that is ruffling or changing shape. The qualitative observations of different simulations are summarized in Table 5.2. There a three different scenario that potentially face

Table 5.2: Probability of collecting useful information

| $\alpha$ | $\lambda$ | Probability |
|---|---|---|
| Low | Low | High |
| Middle | Low | High |
| High | Low | High |
| Low | Middle | High |
| Middle | Middle | Middle |
| High | Middle | Low |
| Low | High | Low |
| Middle | High | Low |
| High | High | Low |

the experimentalist:

1. the collected data are consistently informative about the parameter values,
2. the collected data are sometimes informative about the parameter values, and,
3. the collected data are consistently uninformative about the parameter values.

Data collected from the first scenario are likely to be reported in the literature. If the third scenarios is encountered, it is likely that an experimentalist would search for a different end point to measure (for example: cell receptor phosphorylation, cell adhesion, etc.), rather than struggle with estimating parameters from poor data. The second scenario is perhaps the most worrying. In this situation it is likely that the data may point to conflicting conclusions. It is possible that such data is discarded. This raises the unsettling question of whether there is selective reporting of cell migration results in the literature.

To estimate speed and turning frequency reliably, it was observed that for moderate values of measurement error, $\alpha$, it was necessary to have at least one long run of time intervals where the cell does not change direction. Combining this observation with the result in Equation (5.47), it seems reasonable to assume that the quantity

$$\rho = \frac{\alpha^2}{\Delta t^2 k^3}, \tag{5.64}$$

must be small. $k$ is the number of time intervals in the longest run where the cell does not change direction. Clearly, $\rho$ is small if the measurement error is decreased. Unfortunately, the measurement error may be dominated by dynamic changes in cell shape over which the experimenter has no control. It is not straightforward to predict the effect of changing the sampling time, $\Delta t$, without calculation since the PDF for $k$ also depends on $\Delta t$. The PDF for $k$ is related to the geometric distribution of order $k$ and is given by [14]:

$$\Pr(k|n,p) = F(n,k,p) - F(n,k+1,p)\,, \tag{5.65}$$

where $F(n,k)$ is the recursive function:

$$F(n,k,p) = \begin{cases} F(n-1,k,p) + qp^k\left(1 - F(n-k-1,k,p)\right) & n > k \\ p^k & n = k \\ 0 & 0 \leq n < k \end{cases},$$

$n$, $p$, and $q$ are given by

$$\begin{aligned} n &= \frac{T}{\Delta t}, \\ p &= \exp(-\lambda \Delta t)\,, \\ q &= 1 - p, \end{aligned}$$

and $T$ is the total length of time over which measurements are made. The probability $p$ is the chance that a cell does not turn in a time interval of $\Delta t$. The probability of achieving a long unbroken run of measurements increases if more points are sampled, $n$, or the probability of not turning during a time interval, $p$, is increased (i.e., $\Delta t$ is reduced).

## 5.5 Summary

In the first part of the Chapter, Bayesian estimates for the diffusivity and measurement error of a Brownian random walk are presented. The Brownian random walk is the simplest model of cell migration. The Bayesian parameter estimates are compared to the common literature method of estimating diffusivity by fitting the squared displacement. It was shown that the least-squares estimate suffers from the following problems:

1. the method will only work if the magnitude of the measurement error, $\alpha$, is known *a priori*,
2. there is a relatively high probability that the estimate is significantly different from the underlying true value, and,
3. there is a significant probability that data will be collected that lead to a negative estimate.

In contrast, the Bayesian estimates are valid even when $\alpha$ is unknown. Remarkably, it is possible to distinguish between measurement error and diffusion with Bayesian methods. The Bayesian estimates have a far lower probability of differing significantly from the true value and have a zero probability of being negative. It is therefore our recommendation to use the Bayesian estimates rather than the least-squares estimates when calculating the diffusivity of a Brownian random walk. The posterior PDF for the Bayesian estimate had a significant amount of skewness and a long tail. Consequently it is more honest to plot the posterior PDF when reporting data rather than just reporting a point estimate. The effect of model mismatch was investigated. The Brownian parameter estimates for a correlated random walk did not work well for short sampling times. In contrast, for longer sampling times the Brownian parameter estimates yielded reasonably accurate estimates.

In the second part of this Chapter, a one-dimensional correlated random walk was analyzed. It was found that this model was significantly harder to use computationally due to the multi-modal nature of the likelihood function. It was found that standard

Monte Carlo techniques could not be used to analyze this model. However, a tailored integration scheme was devised to evaluate Bayesian parameter estimates. Code was written to simulate a one-dimensional correlated random walk. The proposed parameter estimation strategy was tested on simulated data. It was found that for some parameter values it was likely to collect informative data. In contrast, for some parameter values it was found unlikely to collect informative data. It is postulated that one of the key factors causing this effect is the difficulty in distinguishing between changes in cell shape and cell turning. Furthermore, it was found that data that contained a long unbroken run where the cell had not changed direction was more likely to yield information about the true parameter values.

# Chapter 6

# Conclusions and Future Work

Cell-signaling phenomena are extremely important in the physiology of disease. Over recent years there has been much interest in using mathematical modeling of cell signaling to gain insight into complex cellular behavior. This modeling effort has encompassed a broad range of mathematical formalisms: Bayesian networks and clustering of gene array data, stochastic models, and deterministic ODE/DAE/PDAE models. The recurring themes of this work are to make inferences about a complex experimental systems and to make predictions about cell physiology. In this context, it is important to analyze mathematical models of cell signaling systematically. In particular, it is important to be able to characterize the solution behavior of a model both qualitatively and quantitatively. Furthermore, it is necessary to have techniques that enable the comparison of experimental data with model predictions. Several different computational techniques have been developed in this thesis to analyze models of cell-signaling phenomenon. These techniques have ranged from the qualitative characterization of model behavior to the statistical analysis of stochastic models. The common theme is to build tools that enable one to characterize and validate complex hypotheses.

Detailed kinetic models of cell-signaling pathways were analyzed in Chapter 2. In particular, it was found that it was error prone to formulate species conservation equations as a system of ODEs. Instead, it was proposed to model such systems as index one DAEs. A drawback of formulating cell-signaling models as index one DAEs is that the systematic analysis of these models is more complex. Three methods were proposed for generating a state-space approximation to the original index one DAE system around an equilibrium solution. The idea is that there is a well-developed control theory for systematically analyzing state-space models. It was shown by the implicit function theorem that asymptotic stability of the state-space approximation implies local asymptotic stability of the original DAE system. A drawback with this analysis is that it cannot be used to make statements about the global behavior of the solution to large perturbations. Whether this is a series deficiency depends on the system under investigation. The proposed methods for generating state-space approximations exploited sparsity in the model equations and were implemented in Fortran 77. The speed and the accuracy for all three methods were characterized.

Parameter estimation for deterministic systems and model selection problems were analyzed in Chapter 4. It was found that Bayesian formulations of these problems lead to logically consistent inferences. Recent advances in deterministic global optimization make tractable kinetic parameter estimation for cell-signaling pathways. In particular, these methods rely on the generation of state bounds and convex underestimating and concave overestimating functions. The state bounds can also be used to characterize the global behavior of the solution to an ODE with respect large variations in the parameter values rigorously . Commonly, the global solution behavior is used to verify whether a model prediction is insensitive to parameter uncertainty. It was shown in this Chapter how model selection can be formulated as an integer optimization problem. It is postulated that integer optimization techniques will reduce the computational burden of model selection compared with explicit enumeration of the discriminating criteria for all possible models. Unfortunately, the optimization technology necessary to solve these problems is still too immature at the time of writing this thesis. However, it is now clear from recent advances that it is likely that this problem can be solved in the near to medium term.

In Chapter 5, stochastic models of cell migration were analyzed using Bayesian techniques. It was possible to answer a variety of questions using these techniques that are not amenable to traditional statisical analysis. For example, it was possible to estimate the diffusivity of a particle moving according to Brownian motion without a priori knowledge of the measurement error. It was found that the Bayesian parameter estimates performed better that the traditional estimates derived from expected mean-squared displacement. However, it was found that all methods of parameter estimation based on Brownian diffusion faired badly if there was significant correlation between the displacement over adjacent time intervals. A more sophisticated model cell migration based on a correlated random walk was also analyzed. Closed-form solutions for the transition PDFs were derived for this model. It was necessary to evaluate a high-dimensional integral to obtain Bayesian parameter estimates. It was found that common techniques for evaluating high-dimensional integrals such as Monte Carlo and interval methods were not suitable for this problem. A tailored integration method was developed that exploited problem structure. Bayesian parameter estimates could be obtained efficiently using this algorithm. A study was performed to characterize the accuracy of the parameter estimates for different parameter values. Unlike parameter estimation for deterministic problems, for some parameter values it was found that there was a wide variation in the information gained between runs; i.e., the variance of the posterior PDF was not constant between identical experiments. This effect is caused by the inherent stochastic nature of the problem. For given parameter values there is some probability that useful information is gained from an experiment and some probability that useful information is not gained. This observation suggests that one should be wary of over-interpreting experimental results.

## 6.1 Future Work

The work in Chapter 2 remains relatively self-containted. However, the implementation of the algorithms for constructing the state-space approximations is quite cumbersome to use. It would be nice to integrate the code for constructing the state-space approximation with high-level modeling software such as ABACUSS II to allow the automatic generation of the state-space approximation from a high-level description of the DAE system.

The work in Chapter 4 is very preliminary in nature, but quite promising. There is a large amount of work that could be done to improve parameter estimation and model selection using deterministic global optimization techniques. The convexity theory developed by [203] is very new and the automatic implementation of the technique is not yet tightly integrated to a suitable branch-and-bound code. Given the complexity of implementing this by hand, developing an integrated code should be a priority for the wide spread adoption of this parameter estimation method. The proposed parameter estimation method relies on generating tight state bounds for a system of ODEs. It seems reasonable to investigate alternative techniques for generating the bounds. The model selection formulations developed in the second part of Chapter 4 seem a promising avenue of research. It seems that the mixed integer dynamic optimization formulation could be solved by combining the convexity theory developed by [203] with the outer approximation ideas developed in [129, 130, 131]. Work could be done on realizing this in a practical implementation. In the longer term, it is necessary to develop optimization algorithms where the objective function is defined by a high dimensional integral. From previous experience it seems that a necessary step is to develop a suitable convexity theory for these problems.

In Chapter 5 it was shown how the Bayesian parameter estimation problem could be formulated for stochastic cell migration models. There are at least four avenues of research that would be interesting to follow: improvement of the integration algorithms, extension of the correlated random walk to planar motion, increasing the sophistication of stochastic models, and improvements in experimental data collection. For one-dimensional correlated migration it was found that the speed of the existing integration routine was sufficient. However, it may be necessary to optimize the integration procedure for two-dimensional cell migration parameter estimation. Perhaps the simplest improvement to the integration method would be to replace naïve quadrature for evaluation of the repeated convolution (Equation (5.11)) with Fast Fourier Transform methods, thus reducing the computational complexity of evaluating the multi-dimensional integral.

In principle, it is simply necessary to derive the transition PDFs to extend the tailored Bayesian parameter estimation methods to planar motion and more sophisticated models of cell migration. However, it does not seem likely that it will be possible to derive the requisite PDFs from alternating renewal theory. Instead, there are two obvious possibilities to construct the transition PDFs: Monte Carlo and solving Fokker-Planck equations. It seems that Monte Carlo simulation is probably the simpler alternative to implement.

Finally, there is the possibility of improving data collection. There are several different possibilities that could be tried to gain more information about cell migration. The current bottleneck in collecting cell migration data is the hand analysis of images to determine the cell outline. It therefore seems reasonable to stain the cell to see if the improvement image contrast is sufficient to allow automatic cell outline detection. Another possibility is to look at movement of the cell nucleus rather than the cell centroid. It is possible that measurement of the movement of the cell nucleus is less susceptible to error due to multiple lamella extension than the measurement of the cell centroid. Furthermore, some authors have measured the relative displacement of the cell nucleus from the cell centroid as a function of time [215]. This measurement gives an indication of the distortion of the cell as it moves. Again, it seems like this measurement might be less susceptible to error due to spurious lamella extension. Finally, the parameter estimation procedure could be improved if it was possible to measure the orientation of the cell rather than infer it from the position measurements. (Therefore, one could easily verify when a cell had turned.) It is know that certain receptors localize at the leading edge of the lamella. It might be possible to develop a antibody based marker to highlight regions on the cell membrane where these characteristic receptors have colocalized. This might give an alternative method to infer the cell orientation and hence improve the estimate of cell speed and turning frequency.

# Appendix A

# Matlab Code

## A.1 Least Squares Fit

```
%
% Simple function to fit y=40sin(ax)
% and y=ax^2.
%

function []=tt1() % Void function
clear;

%
% Set up data
%

xi=[1 2 3 4 5];
yi=[0.8801 3.9347 9.4853 15.4045 24.8503];

%
% Least Squares functions.
%

chi=inline('sum((40*sin(a*[1,2,3,4,5])-[0.8801,3.9347,9.4853,15.4045,24.8503]).^2)');
chi2=inline('sum((a*[1,2,3,4,5].^2-[0.8801,3.9347,9.4853,15.4045,24.8503]).^2)');

[P1,feval]=fminbnd(chi,0,20)
[P2,feval]=fminbnd(chi2,0,20)
return;
```

## A.2 Testing State-Space Approximation to Random Sparse DAEs

```
%
%                                                                  %
%       Code to test error in finding state-space form of a DAE    %
%       The algorithms are described in:                           %
%                                                                  %
%       Efficient Construction of Linear State-Space Models from Index One %
%       DAES, D. M. Collins, D. A. Lauffenburger and P. I. Barton, 2001. %
%                                                                  %
%       Code written by D. M. Collins                              %
%                                                                  %
%       Form problem                                               %
%       ————                                                       %
%       1.) Generate W (n+m x n+m) and V (n+m x n)                 %
%       2.) Calculate S (n+m x n) by solving W S = V.              %
%       3.) Refine V by calculating V=W*S.                         %
%                                                                  %
%       Generate State-Space Models S1, S2 and S3                  %
%       ——————————————————                                         %
%       4.) Calculate S_1 by WZ_1 = I; S_1 = Z_1 V.                %
%       5.) Calculate S_2 by W^T Z_2 = I; S_2 = Z_2^T V.           %
%       6.) Calculate S_3 by W S_3 = V.                            %
%                                                                  %
%       Compare State-Space Models S1, S2 and S3 with S            %
%       ———————————————————————                                    %
%                                                                  %
%
function test()
%       Initialize vector.
clear;
warning debug;
n=100;
m=50;
n1=20;
maxiter=500;
fill_W=5;
fill_V=5;
cndno=1e6;

error=state(n,m,n1,maxiter,fill_W,fill_V,cndno);
%
%       Syntax: error=test(n,m,n1,maxiter,fnme)
%
%       n:       Number of states+algebraic variables
%       m:       Number of states+inputs
%       n1:      Number of states and algebraic variables in state-space model
%       maxiter: Number of tests to run
%       fill_W:  Number of entries per row of W
%       fill_V:  Number of entries per row of V
%       error(1:3,1:maxiter):
```

```
%

%
%       Plot error distribution                                   %
%

hist(transpose(error),linspace(0,3*std(error(1,:)),10))
tt=strcat('Distribution of Errors for (n_{x}+n_{y})=',num2str(n),...
' (n_{x}+n_{u})=',num2str(m));
title(tt)
ylabel('Frequency')
xlabel('Relative error')
legend('Algorithm 1','Algorithm 2','Algorithm 3')
return;

function [error]=state(n,m,n1,maxiter, fill_W, fill_V, cndno)

n2=n-n1; % Number of algebraic variables eliminated.

error(1:6,1:maxiter)=0;
comperror(1:6,1:maxiter)=0;
fc1=0;
fc2=0;
fc3=0;

fc1dm=0;
fc2dm=0;
fc3dm=0;

for i=1:maxiter,

%
%               Initialize problem                                %
%

W=sprand(n,n,fill_W/n,1/cndno);
V=sprand(n,m,fill_V/m);
[iv, jv] = find(V);

% Solve for S
[S,fc]=dmss3(W,V);

% Refine V
V1=W*S;

% Delete entries that do not correspond to original V1

for j=1:length(iv),
V(iv(j),jv(j))=V1(iv(j),jv(j));
end
V=sparse(V);

%
%               Test algorithms                                   %
```

```
%

% Algorithm 1 without block decomposition.
[S_1,fc]=ss1(W, V);
fc1=fc+fc1;

% Record error.
error(1,i)=err(S,S_1,n1);
comperror(1,i)=comperr(S,S_1,n1);

% Algorithm 1 with block decomposition.
[S_1,fc]=dmss1(W, V);
fc1dm=fc+fc1dm;

% Record error.
error(4,i)=err(S,S_1,n1);
comperror(4,i)=comperr(S,S_1,n1);


% Algorithm 2 without block decomposition.
[S_2,fc]=ss2(W, V, n1);
fc2=fc+fc2;

% Record error.
error(2,i)=err(S,S_2,n1);
comperror(2,i)=comperr(S,S_2,n1);

% Algorithm 2 with block decomposition.
[S_2,fc]=dmss2(W, V, n1);
fc2dm=fc+fc2dm;

% Record error.
error(5,i)=err(S,S_2,n1);
comperror(5,i)=comperr(S,S_2,n1);

% Algorithm 3 without block decomposition.
[S_3, fc]=ss3(W, V);
fc3=fc+fc3;

% Record error.
error(3,i)=err(S,S_3,n1);
comperror(3,i)=comperr(S,S_3,n1);

% Algorithm 3 with block decomposition.
[S_3, fc]=dmss3(W, V);
fc3dm=fc+fc3dm;

% Record error.
error(6,i)=err(S,S_3,n1);
comperror(6,i)=comperr(S,S_3,n1);

end

%
```

```
%       Report statistics                                       %
%
disp(sprintf('\n%s\n\n%s\n%s\n%s%d\n%s\n%s%e',...
'Finding the Explicit form of an Implicit DAE.', ...
'Problem Statistics', ...
'------------------', ...
'Number of test problems: ', maxiter, ...
'Matrix generated by function: sprand', ...
'Mean condition number: ', cndno))
disp(sprintf('%s%d\n%s%d\n%s%d\n%s%d\n%s%d\n%s%d', ...
'Number of entries per row of W: ', fill_W, ...
'Number of entries per row of V: ', fill_V, ...
'Number of states+outputs: ',n, ...
'Number of states+inputs: ',m, ...
'Number of outputs in explicit form: ',n1, ...
'Number of outputs eliminated from model: ',n2))

disp(sprintf('\n\n%s\n%s\n%s',...
'           Max Error   Mean Error  Standard Deviation Floating Point Operations', ...
'           ---------   ----------  ------------------ -------------------------'))

disp(sprintf('\n%s\n%s%e%s%e%s%e%s%d\n%s%e%s%e%s%e%s%d\n%s%e%s%e%s%e%s%d',...
'Norm error: ',...
'A1          ', max(error(1,:)),' ',mean(error(1,:)),' ',...
std(error(1,:)),'          ',fc1/maxiter, ...
'A2          ', max(error(2,:)),' ',mean(error(2,:)),' ',...
std(error(2,:)),'          ',fc2/maxiter, ...
'A3          ', max(error(3,:)),' ',mean(error(3,:)),' ',...
std(error(3,:)),'          ',fc3/maxiter))

disp(sprintf('\n%s%e%s%e%s%e%s%d\n%s%e%s%e%s%e%s%d\n%s%e%s%e%s%e%s%d',...
'A1 DM       ', max(error(4,:)),' ',mean(error(4,:)),' ',...
std(error(4,:)),'          ',fc1dm/maxiter, ...
'A2 DM       ', max(error(5,:)),' ',mean(error(5,:)),' ',...
std(error(5,:)),'          ',fc2dm/maxiter, ...
'A3 DM       ', max(error(6,:)),' ',mean(error(6,:)),' ',...
std(error(6,:)),'          ',fc3dm/maxiter))

disp(sprintf('\n%s\n%s%e%s%e%s%e%s%d\n%s%e%s%e%s%e%s%d\n%s%e%s%e%s%e%s%d',...
'Component error: ',...
'A1          ', max(comperror(1,:)),' ',mean(comperror(1,:)),' ',...
std(comperror(1,:)),'          ',fc1/maxiter, ...
'A2          ', max(comperror(2,:)),' ',mean(comperror(2,:)),' ',...
std(comperror(2,:)),'          ',fc2/maxiter, ...
'A3          ', max(comperror(3,:)),' ',mean(comperror(3,:)),' ',...
std(comperror(3,:)),'          ',fc3/maxiter))

disp(sprintf('\n%s%e%s%e%s%e%s%d\n%s%e%s%e%s%e%s%d\n%s%e%s%e%s%e%s%d',...
'A1 DM       ', max(comperror(4,:)),' ',mean(comperror(4,:)),' ',...
std(comperror(4,:)),'          ',fc1dm/maxiter, ...
'A2 DM       ', max(comperror(5,:)),' ',mean(comperror(5,:)),' ',...
std(comperror(5,:)),'          ',fc2dm/maxiter, ...
'A3 DM       ', max(comperror(6,:)),' ',mean(comperror(6,:)),' ',...
std(comperror(6,:)),'          ',fc3dm/maxiter))
```

```
return;

function [x]=err(S,Serr,n1)
% Function to generate error for non-zero components
x=normest(S(1:n1,:)−Serr(1:n1,:))/normest(S(1:n1,:));
return;

function [x]=comperr(S,Serr,n1)
S=sparse(S);
Serr=sparse(Serr);
[i,j]=find(S(1:n1,:));
nz=length(i);
x=0;
for k=1:nz,
        t=abs((S(i(k),j(k))−Serr(i(k),j(k)))/S(i(k),j(k)));
        x=max(x,t);
end
return;

function [S,fc]=ss1(W, V)
%
% Algorithm 1
%
flops(0);
I=speye(size(W));
Z_1=W\I;
S=Z_1*V;
fc=flops;
return;

function [S,fc]=dmss1(W, V)
%
% Algorithm 1
%
flops(0);
I=speye(size(W));
Z_1=dmsolve(W,I);
S=Z_1*V;
fc=flops;
return;

function [S,fc]=ss2(W, V, n1)
%
% Algorithm 2
%
flops(0);
I=speye(size(W));
Z_2=W'\I(:,1:n1);
S=Z_2'*V;
fc=flops;
return;

function [S,fc]=dmss2(W, V, n1)
%
```

```
% Algorithm 2
%
flops(0);
I=speye(size(W));
Z_2=dmsolve(W',I(:,1:n1));
S=Z_2'*V;
fc=flops;
return;


function [S,fc]=ss3(W, V)
%
% Algorithm 3
%
flops(0);
S=W\V;
fc=flops;
return;

function [S,fc]=dmss3(W, V)
%
% Algorithm 3
%
flops(0);
S=dmsolve(W,V);
fc=flops;
return;


function x = dmsolve(A,b)
%
%       Solve Ax=b by permuting to block                          %
%       upper triangular form and then performing                 %
%       block back substition.                                    %
%                                                                 %
%       Adapted from pseudo-code in:                              %
%       Sparse Matrices in Matlab: Design and Implementation,     %
%       John R. Gilbert, Cleve Moler, and Robert Schreiber.       %
%                                                                 %
%       By: David M. Collins 02/12/01.                            %
%

        % Check for a square matrix.

        [n,m]=size(A);

        if (n~=m)
        error('Matrix is not square.')
        end

        % Check that b is long enough.
        m=length(b);

        if (n~=m)
        error('Vector is different length to order of matrix')
        end

        % Permute A to block form.
        [p,q,r]= dmperm(A);
        nblocks=length(r)−1;
        A=A(p,q);
        x=b(p,:);

        % Block backsolve
        for k=nblocks:−1:2
                % Indices above the kth block
                i=1:r(k)−1;

                % Indices of the kth block.
                j=r(k) : r(k+1)−1;
                x(j,:) = A(j,j)\x(j,:);
                x(i,:) = x(i,:) − A(i,j)*x(j,:);

       end;

       j=r(1):r(2)−1;
       x(j,:)=A(j,j)\x(j,:);
       % Undo the permutation of x.
       x(q,:) = x;

return;
```

## A.3 Generation of State-Space Approximation to Coupled-Tanks Problem

```
%
%       Example 1 for IFAC paper. Two tanks coupled by a membrane.
%
%       Equations:
%       V xdot1 + Ayn+1 = 0
%       V xdot2 - Ayn+2 = 0
%       y1 - Kx1 = 0
%       Kyn - x2 = 0
%       yn+1 + D(y3-y1)/2delta = 0
%       yn+2 + D(yn-yn-2)/2delta = 0
%       (yi-2yi+1+yi+2)/2delta^2 = 0 i=1..n-2
%
clear;
more on;
% Initialize the Jacobian.
N=100;
V=10;
D=1;
A=3;
K=0.5;
delta=0.01;

Jac =zeros(N+4,N+4);

Jac(1,1)=V;
Jac(1,N+3)=+A;
Jac(2,2)=V;
Jac(2,N+4)=-A;
Jac(3,3)=1;
Jac(4,N+2)=1;
Jac(5,N+3)=1;
Jac(5,3)=-D/2/delta;
Jac(5,5)=D/2/delta;
Jac(6,N+4)=1;
Jac(6,N+2)=D/2/delta;
Jac(6,N)=-D/2/delta;

for i=1:N-2,
        Jac(6+i, 3+i)=-2;
        Jac(6+i, 2+i)=1;
        Jac(6+i, 4+i)=1;
end
JacX=zeros(N+4,2);
JacX(3,1)=K;
JacX(4,2)=K;

% Initialize the identity.
I=eye(N+4);
```

```
% Setup problem:
flops(0);
[L, U]=lu(Jac);
fclu=flops;

flops(0);
[LT, UT]=lu(transpose(Jac));
fclut=flops;

% Method 1
flops(0);
Z_1=U\(L\I);
S_1=Z_1*JacX;
fc1=flops+fclu;

% Method 2
flops(0);
Z_2=UT\(LT\I(:,1:2));
S_2=transpose(Z_2)*JacX;
fc2=flops+fclut;

% Method 3
flops(0);
S_3=U\(L\JacX);
fc3=fclu+flops;

l=(N-1)*delta
t=l^2/D
[fc1,fc2,fc3]

geom=2*A*l*K/V
[eig(S_1(1:2,1:2)), eig(S_2(1:2,1:2)), eig(S_3(1:2,1:2))]
tau= 1/(V*l/2/A/D/K)
```

## A.4 Bayesian Parameter Estimation for Brownian Diffusion

```
function []=brownian();
%==============================================
%
% Function to simulate Brownian Random walk
% and then estimate D and alpha from simulation
% data. Written by D. M. Collins 08/12/03.
%
%==============================================
  close;
  Df_true=3;
  alpha_true=3;
  ndata=20;
  deltat(1:ndata,1)=ones(ndata,1);
  beta_true=sqrt(2*Df_true.*deltat);
  K=sqrt(2*pi);
%
% Generate Data
%

  x=zeros(ndata+1,1);
  x(1)=alpha_true*randn(1,1);
  x(2:ndata+1)=beta_true.*randn(ndata,1);
  x=cumsum(x);
  y(1:ndata,1)=alpha_true*randn(ndata,1)+x(2:ndata+1,1);
  t=[0;cumsum(deltat)];

  d=diff([0;y]);
  dsl=1/(2*sum(deltat))*sum(d.*d−2*alpha_true^2);


%
% Plot the data!!
%
  plotl=plot(t,x,'k-');
  hold on;
  e=2*alpha_true*ones(1,ndata);
  errl=errorbar(t(2:ndata+1),y,e,'xk');
  legend([plotl,errl(2)],'True Position','Measured position',−1)
  v=axis;
  axis([0,1.1*max(t),v(3:4)])
  xlabel('Time')
  ylabel('Displacement')
  hold off;
  exportfig(gcf,'simdata','FontMode','Fixed','FontSize','8','Color','gray',...
  'Height','3','Width','5','LineMode','Fixed','LineWidth','1')
%==============================================
%
% Run Estimation
%
```

```
%=============================================


%
% Set up diffusivity grid
%
 minD=0.01;
 maxD=12;
 norder=8;
 [df,wb]=qsimp(minD,maxD,norder);
 nD=length(df);
 beta=sqrt(2*df*deltat');


 minalpha=0.01;
 maxalpha=6;
 [alpha,wa]=qsimp(minalpha,maxalpha,norder);
 nalpha=length(alpha);
 pdf=zeros(nD,nalpha);


 for j=1:nalpha,
 for i=1:nD,
   V11=zeros(ndata+1,ndata+1);
   V11(2:ndata+1,1:ndata)=V11(2:ndata+1,1:ndata)-diag(1./beta(i,:).^2);
   V11(1:ndata,2:ndata+1)=V11(1:ndata,2:ndata+1)-diag(1./beta(i,:).^2);
   V11=V11+1/alpha(j)^2*eye(ndata+1);
   V11(1:ndata,1:ndata)=V11(1:ndata,1:ndata)+diag(1./beta(i,:).^2);
   V11(2:ndata+1,2:ndata+1)=V11(2:ndata+1,2:ndata+1)+diag(1./beta(i,:).^2);
   V22=1/(alpha(j))^2*eye(ndata);
   V12=zeros(ndata+1,ndata);
   V12(2:ndata+1,1:ndata)=-1/alpha(j)^2*eye(ndata);
   Q=V22-V12'*inv(V11)*V12;
   pdf(i,j)=sqrt(det(Q))*exp(-y'*Q*y/2)/(K^ndata);
 end
 end
%=============================================
%
% Plot Results
%
%=============================================


 [alph,nal]=min((alpha-alpha_true).*(alpha-alpha_true));
 colormap('gray')
 contour(alpha,df,pdf,50)
 xlabel('\alpha')
 ylabel('Diffusivity')
 exportfig(gcf,'contplot','FontMode','Fixed','FontSize','8','Color','gray',...
 'Height','4','Width','4','LineMode','Fixed','LineWidth','1')

 pdfal=pdf(:,nal)/(wb'*pdf(:,nal));
 pdf2=pdf*wa;
 pdf2=pdf2/(wb'*pdf2);
```

```
  plot(df,pdfal,'k',df,pdf2,'k--')
  legend('p(D|{\bf y},\alpha)','p(D|{\bf y})')
  xlabel('Diffusivity')
  ylabel('Probability Density')
  dmapa=df(find(pdfal==max(pdfal)));
  dmap=df(find(pdf2==max(pdf2)));

  info=['\alpha known: D_{LS} = ',num2str(dsl)];
  info=strvcat(info,['\alpha known: D_{MAP} = ',num2str(dmapa)]);
  info=strvcat(info,['\alpha unknown: D_{MAP} = ',num2str(dmap)]);
  h=axis;
% '\n D_{MAP}')
  text(7.5,0.75*h(4),info)

  exportfig(gcf,'brownian','FontMode','Fixed','FontSize','8','Color','gray',...
  'Height','4','Width','5','LineMode','Fixed','LineWidth','1')



return;

%
%       Simpson integration routine.
%

function [xi,wi]=qsimp(a,b,n);
    ngap=2^(n-1);
    xi=(a:(b-a)/ngap:b)';
    wi=zeros(ngap+1,1);
    wi(1)=1/3*(b-a)/ngap;
    for i=(2:2:ngap),
        wi(i)=4/3*(b-a)/ngap;
    end
    for i=(3:2:ngap-1),
        wi(i)=2/3*(b-a)/ngap;
    end
    wi(ngap+1)=1/3*(b-a)/ngap;
return
```

## A.5 Generation of Correlated Random Walk Data

```
function []=rwalk();
sigma=1;
lambda=0.6;
ndata=21;
C=3;
deltat=1;
[tturn,ytrue,tsample,ymeasured]=rwalk(sigma,lambda,ndata,C,deltat);
close
plotl=plot(tturn,ytrue,'k-');
hold on;
e=2*sigma*ones(ndata,1);
errl=errorbar(tsample,ymeasured,e,'xk');
legend([plotl,errl(2)],'True Position','Measured position',−1)
v=axis;
 axis([0,1.1*max(tsample),v(3:4)])
 xlabel('Time')
 ylabel('Displacement')
 hold off;
 exportfig(gcf,'simdata2','FontMode','Fixed','FontSize','8','Color','gray',...
  'Height','3','Width','5','LineMode','Fixed','LineWidth','1')

fid=fopen('corr.dat','w');
  fprintf('% Data and results for Correlated Random Walk\n');
  fprintf('% Generated by rwalk.m \n');
  fprintf(fid,'ndata = %i;\n',ndata-1);
  fprintf(fid,'y = [');
  fprintf(fid,'%d, ',ymeasured(2:ndata−1));
  fprintf(fid,'%d ];\n',ymeasured(ndata));
  fclose(fid);

return;

function [tturn,ytrue,tsample,ymeasured]=rwalk(sigma,lambda,ndata,C,deltat)
%
% Function to simulate noisy random walk data
% ——————————————————-
% tturn     Turning times
% ytrue     True positions of random walk
% tsample  Sampling times
% ymeasured Noisy measurements of particle position
%
    np=6*ceil(lambda*ndata);
    isign=2*(rand(1)<0.5)−1;
    x=exprnd(1/lambda,np,1);
    t=zeros(np+1,1);
    disp=zeros(np+1,1);
    t(2:np+1)=cumsum(x);
    isign=1−2*(rand(1)>0.5);
    disp(3:2:np+1)=isign*x((2:2:np));
    disp(2:2:np)=−isign*x((1:2:np−1));
    disp(1)=sigma*randn(1,1);
    disp=C*cumsum(disp);

    tsample=(0:deltat:(ndata-1)*deltat);
    ymeasured(1)=0;
    ymeasured(2:ndata)=interp1(t,disp,tsample(2:ndata))'+sigma*randn(ndata-1,1);
    tmax=tsample(ndata);
    imax=min(find(tmax<t));
    tturn=t(1:imax);
    ytrue=disp(1:imax);


ytrue(imax)=(ytrue(imax)-ytrue(imax-1))/(tturn(imax)-tturn(imax-1))*...
            (tsample(ndata)-tturn(imax-1))+ytrue(imax-1);
    tturn(imax)=tsample(ndata);

return;
```

# Appendix B

# ABACUSS II Code

## B.1 Interleukin-2 Trafficking Simulation [81]

```
#==============================================
#       Simple model of IL-2 trafficking. Adapted from
#       Fallon, EM, Lauffenburger DA, Computational Model for
#       Effects of Ligand/Receptor Binding Properties on
#       Interleukin-2 Trafficking Dynamics and T Cell Proliferation
#       Response, Biotechnol. Prog, 16:905-916, 2000.
#
#       Written by David M. Collins 03/27/02.
#       Copyright MIT 2002.
#==============================================

DECLARE
TYPE
Concentration = 1000 :−1E−4 :1E20 UNIT = "pM"
Moleculescell = 1E3         :−1E−4 :1E20 UNIT = "molecules/cell"
Moleculesliter = 1E3  :−1E−4 :1E20 UNIT = "molecules/liter"
CellDensity  = 1E8          :0            :1E10   UNIT = "cells/litre"
END #declare

MODEL InterleukinTrafficking

PARAMETER
  # Surface dissociation rate constant (min^-1)
  kr     AS     REAL
  # Surface association rate constant (pM^-1 min^-1)
  kf     AS     REAL
  # Constitutive receptor internalization rate constant (min^-1)
  kt     AS     REAL
  # Constitutive receptor synthesis rate (# cell^-1 min^-1)
  Vs     AS     REAL
  # Induced receptor synthesis rate (min^-1)
  ksyn  AS     REAL
  # Internalization rate constant (min^-1)
  ke     AS     REAL
  # Avogadro's number (#/pico mole)
  Na    AS     REAL
  # Endosome dissociation rate constant (min^-1)
  kre   AS     REAL
  # Endosome association rate constant (pM^-1 min^-1)
  kfe   AS     REAL
  # Recycling rate constant (min^-1)
  kx    AS     REAL
  # Degradation rate constant (min^-1)
  kh    AS     REAL
  # Endosomal volume (liter/cell)
  Ve    AS     REAL

VARIABLE
  # Number of unbound receptors at cell surface (#/cell)
  Rs    AS     Moleculescell
  # Number of ligand-receptor complexes at cell surface (#/cell)
  Cs    As     Moleculescell
  # Ligand concentration in bulk (pM)
  L     AS     Concentration
  # Number of unbound receptors in endosome (#/cell)
  Ri    AS     Moleculescell
  # Number of ligand-receptor complexes in endosome (#/cell)
  Ci    AS     Moleculescell
  # Ligand concentration in endosome (pM)
  Li    AS     Concentration
  # Ligand destroyed in endosome (pM)
  Ld    AS     Concentration
  # Number of cells per unit volume (#/litre)
  Y     AS     CellDensity
  # Total ligand concentration in all forms (pM)
  LT    AS     Concentration

SET
  kt:=0.007;
  Vs:=11;
  ksyn:=0.0011;
  ke:=0.04;
  kx:=0.15;
  kh:=0.035;
  Ve:=1E−14;
  Na:=6E11;


EQUATION

# Warning: The number of cells in the medium is not constant.
# Each time a cell divides, the number of receptors at the
# surface halves.

# Receptor balance at surface:

$Rs = Vs + kr*Cs + ksyn*Cs − kt*Rs − kf*Rs*L;
```

```
# Ligand-receptor complex balance at surface:

$Cs = kf*Rs*L − kr*Cs − ke*Cs;

# Receptor balance in endosome:

$Ri = kre*Ci + kt*Rs − kfe*Ri*Li − kh*Ri;

# Ligand-receptor complex balance in endosome:


$Ci = ke*Cs + kfe*Ri*Li − kre*Ci − kh*Ci;

# Ligand balance in endosome:

$Li = (kre*Ci−kfe*Ri*Li)/(Ve*Na) − kx*Li;

# Ligand balance on bulk medium:

$L = (Y*kr*Cs/Na + Y*kx*Ve*Li − Y*kf*Rs*L/Na);

# Empirical cell growth relationship
$Y = MAX(600*Cs/(250+Cs)−200,0)*1E3;

# Concentration of ligand destroyed in endosome (pM/min)
$Ld = kh*Ci/(Ve*Na);

# Track total ligand concentration in bound/unbound forms (pM)
LT = L + (Y*Cs/Na +Y*Ci/Na + Ve*Y*Li+ Ve*Y*Ld);

END #model

SIMULATION EXAMPLE
OPTIONS
  CSVOUTPUT := TRUE ;
UNIT
  CellProliferation AS InterleukinTrafficking
REPORT
  CellProliferation.LT
SET
WITHIN CellProliferation DO
   kr:=0.0138;
   kf:=kr/11.1;
   kre:=8*0.0138;
   kfe:=CellProliferation.kre/1000;
END

INITIAL
  WITHIN CellProliferation DO
    L = 0;
    Y = 2.5E8;
    $Rs = 0;
    $Cs = 0;
    $Ri = 0;
    $Ci = 0;
    $Li = 0;
    Ld = 0;
 END

SCHEDULE
  SEQUENCE
    CONTINUE FOR 1
    REINITIAL
      CellProliferation.L
    WITH
      CellProliferation.L=10;
    END
    CONTINUE FOR 5*24*60
  END
END #simulation
```

## B.2 Reformulated Interleukin-2 Trafficking Simulation

```
#==========================================
#       Simple model of IL-2 trafficking. Adapted from
#       Fallon, EM, Lauffenburger DA, Computational Model for
#       Effects of Ligand/Receptor Binding Properties on
#       Interleukin-2 Trafficking Dynamics and T Cell Proliferation
#       Response, Biotechnol. Prog, 16:905-916, 2000.
#
#       Written by David M. Collins 03/27/02.
#       Copyright MIT 2002.
#==========================================

DECLARE
TYPE
Concentration = 1000 :-1E-4 :1E20 UNIT = "pM"
Moleculescell = 1E3         :-1E-4 :1E20 UNIT = "molecules/cell"
Moleculesliter = 1E3  :-1E-4 :1E20 UNIT = "molecules/liter"
CellDensity     = 1E8  :0            :1E10   UNIT = "cells/litre"
Flux            = 1E3  :-1E20 :1E20  UNIT = "pM/min"
END #declare

MODEL InterleukinTrafficking

PARAMETER
  # Surface dissociation rate constant (min^-1)
  kr    AS    REAL
  # Surface association rate constant (pM^-1 min^-1)
  kf    AS    REAL
  # Constitutive receptor internalization rate constant (min^-1)
  kt    AS    REAL
  # Constitutive receptor synthesis rate (# cell^-1 min^-1)
  Vs    AS    REAL
  # Induced receptor synthesis rate (min^-1)
  ksyn  AS    REAL
  # Internalization rate constant (min^-1)
  ke    AS    REAL
  # Avogadro's number (#/pico mole)
  Na    AS    REAL
  # Endosome dissociation rate constant (min^-1)
  kre   AS    REAL
  # Endosome association rate constant (pM^-1 min^-1)
  kfe   AS    REAL
  # Recycling rate constant (min^-1)
  kx    AS    REAL
  # Degradation rate constant (min^-1)
  kh    AS    REAL
  # Endosomal volume (liter/cell)
  Ve    AS    REAL

VARIABLE
```

```
# Number of unbound receptors at cell surface (#/cell)
Rs    AS    Moleculescell
# Number of ligand-receptor complexes at cell surface (#/cell)
Cs    As    Moleculescell
# Ligand concentration in bulk (pM)
L     AS    Concentration
# Number of unbound receptors in endosome (#/cell)
Ri    AS    Moleculescell
# Number of ligand-receptor complexes in endosome (#/cell)
Ci    AS    Moleculescell
# Ligand concentration in endosome (pM)
Li    AS    Concentration
# Ligand destroyed in endosome (pM)
Ld    AS    Concentration
# Number of cells per unit volume (#/litre)
Y     AS    CellDensity
# Total ligand concentration in all forms (pM)
LT    AS    Concentration
# Total number of receptors
Nrs   AS    Moleculesliter
# Total number of complexes
Ncs   AS    Moleculesliter
# Total number of internalized receptors
Nri   AS    Moleculesliter
# Total number of internalized compexes
Nci   AS    Moleculesliter
# Overall concentration of ligand in endosme
Nli   AS    Concentration
# Overall concentration of ligand destroyed
Nld   AS    Concentration
# Flux of ligand from surface to bulk
FLsb  AS    Flux
# Flux of ligand from bulk to surface
FLbs  AS    Flux
# Flux of ligand from endosome to bulk
FLeb  AS    Flux
# Flux of receptor from cytosol to surface
FRcs  AS    Flux
# Flux of receptor from surface to endosome
FRse  AS    Flux
# Rate of generation of free receptors at surface
rRs   AS    Flux
# Rate of generation of ligand-receptor complexes at surface
rCs   AS    Flux
# Flux of complexes from surface to endosome
FCse  AS    Flux
# Rate of generation of receptors in endosome
rRe   AS    Flux
# Rate of generation of complexes in endosome
rCe   AS    Flux
# Rate of generation of ligands in endosome
rLe   AS    Flux

SET
```

```
  kt:=0.007;
  Vs:=11;
  ksyn:=0.0011;
  ke:=0.04;
  kx:=0.15;
  kh:=0.035;
  Ve:=1E−14;
  Na:=6E11;


EQUATION

# The number of cells in the medium is not constant. Each time a
# cell divides, the number of receptors at the surface halves.
# Hence we must perform the balance around the total cell volume.

# Empirical cell growth relationship
$Y = MAX(600*Cs/(250+Cs)−200,0)*1E3;

# Ligand balance on bulk medium:

# Accumulation                              $L              (pM/min)
# Dissociation of ligand-receptor           Y*kr*Cs         (#/litre/min)
# Ligand recycling                          Y*kx*Ve*Li      (pM/min)
# Association of ligand and receptor        Y*kf*Rs*Ls      (#/litre/min)

$L = FLsb−FLbs+FLeb;
FLsb=Y*kr*Cs/Na;
FLbs=Y*kf*Rs*L/Na;
FLeb=Y*kx*Ve*Li;

# Receptor balance at surface:

# Accumulation                              $(YRs)          (#/litre/min)
# Bulk synthesis                            Y*Vs            (#/litre/min)
# Dissociation of ligand-receptor complex   Y*kr*Cs         (#/litre/min)
# Induced receptor synthesis                Y*ksyn*Cs       (#/litre/min)
# Constitutive internalization              Y*kt*Rs         (#/litre/min)
# Association of ligand and receptor        Y*kf*Rs*L       (#/liter/min)

$Nrs =FRcs−FRse+rRs;
FRcs=Y*Vs;
FRse=Y*kt*Rs;
rRs=Y*(ksyn*Cs+kr*Cs−kf*Rs*L);
Nrs = Y*Rs;

# Ligand-receptor complex balance at surface:

# Accumulation                              $(YCs)          (#/litre/min)
# Association of ligand and receptor        Y*kf*Rs*L       (#/litre/min)
# Dissociation of ligand-receptor complex   Y*kr*Cs         (#/litre/min)
# Internalization of complex from surface   Y*ke*Cs         (#/litre/min)

$Ncs = rCs−FCse;
```

```
rCs=Y*(kf*Rs*L − kr*Cs);
FCse=Y*ke*Cs;
Ncs = Y*Cs;

# Receptor balance in endosome:

# Accumulation                                  $(YRi)         (#/litre/min)
# Dissociation of ligand-receptor complex       Y*kre*Ci       (#/litre/min)
# Constitutive internalization                  Y*kt*Rs        (#/litre/min)
# Association of ligand and receptor            Y*kfe*Ri*Li    (#/litre/min)
# Receptor destruction by lysosome              Y*kh*Ri        (#/litre/min)

$Nri = FRse+rRe;
rRe=Y*(kre*Ci − kfe*Ri*Li − kh*Ri);
Nri=Y*Ri;

# Ligand-receptor complex balance in endosome:

# Accumulation                                  $(YCi)         (#/litre/min)
# Internalization of complex from surface       Y*ke*Cs        (#/litre/min)
# Association of ligand and receptor            Y*kfe*Ri*Li    (#/litre/min)
# Dissociation of ligand-receptor complex       Y*kre*Ci       (#/litre/min)
# Complex destruction by lysosome               Y*kh*Ri        (#/litre/min)

$Nci = FCse+rCe;
rCe=Y*(kfe*Ri*Li − kre*Ci − kh*Ci);
NCi=Y*Ci;

# Ligand balance in endosome:

# Accumulation                                  $(Y*Li*Ve)     (pM/min)
# Dissociation of ligand-receptor complex       Y*kre*Ci       (#/litre/min)
# Association of ligand and receptor            Y*kfe*Ri*Li    (#/litre/min)
# Ligand recycling                              Y*kx*Li        (pM/min)

$Nℓi = −FLeb+rLe;
rLe=Y*(kre*Ci−kfe*Ri*Li)/Na;
Nli=Y*Li*Ve;

# Concentration of ligand destroyed in endosome (pM/min)
$NLd = Y*(kh*Ci/(Ve*Na));
Nld=Y*Ld;

# Track total ligand concentration in bound/unbound forms (pM)
LT = L + (Y*Cs/Na +Y*Ci/Na + Ve*Y*Li+ Ve*Y*Ld);

END #model


SIMULATION EXAMPLE
OPTIONS
  CSVOUTPUT:=TRUE;
UNIT
  CellProliferation AS InterleukinTrafficking
```

```
REPORT
  CellProliferation.LT
SET
WITHIN CellProliferation DO
   kr:=0.0138;
   kf:=kr/11.1;
   kre:=8*0.0138;
   kfe:=CellProliferation.kre/1000;
END


INITIAL
  WITHIN CellProliferation DO
    Y = 2.5E8;
    $NRs = 0;
    $NCs = 0;
    $NRi = 0;
    $NCi = 0;
    $NLi = 0;
    NLd = 0;
    L = 0;
  END

SCHEDULE
  SEQUENCE
  CONTINUE FOR 1
  REINITIAL
    CellProliferation.L
  WITH
    CellProliferation.L = 10;
  END
  CONTINUE FOR 5*24*60
  END
END #simulation
```

## B.3 Short Term Epidermal Growth Factor Signaling Model

```
#======================================
#
#       A kinetic model of short term EGF activation provided from
#       paper:
#
#       Kholodenko B. N., Demin O. V., Moehren G., and Hoek J. B.,
#       Quantification of Short Term Signaling by the Epidermal
#       Growth Factor Receptor, Journal of Biological Chemistry,
#       274(42), pp 30169-30181, 1999.
#
#       Model written by D. M. Collins, 11/25/2000.
#
#======================================

DECLARE

TYPE
        # Identifier          # Default     # Lower      # Upper
Concentration  =0             : −1E−7       : 10000 UNIT="nM"
Rate           =1             : −1E9        : 1E9   UNIT="nM/s"

END

MODEL EGF

PARAMETER
        NFORWD AS INTEGER # Number of forward reactions
        NREVRS AS  INTEGER # Number of reverse reactions
        NMICHL AS  INTEGER # Number of M-M reactions
        NREACS AS  INTEGER # Total number of reactions

        kforwd AS     ARRAY(NFORWD) OF REAL #(nM/s or s^-1)
        krevrs AS     ARRAY(NREVRS) OF REAL #(nM/s or s^-1)
        K      AS     ARRAY(NMICHL) OF REAL #nM
        V      AS     ARRAY(NMICHL) OF REAL #nM/s

#
#       Mass balance constraints
#

        EGFRT,
        PLCgT,
        GrbT,
        ShcT,
        SOST          AS REAL

VARIABLE

#
```

```
#       States
#
        EGF,
        R,
        Ract,
        Rdimer,
        RP,
        R_PL,
        R_PLP,
        R_G,
        R_G_S,
        R_Sh,
        R_ShP,
        R_Sh_G,
        R_Sh_G_S,
        G_S,
        ShP,
        Sh_G,
        Sh_G_S,
        PLCg,
        PLCgP,
        PLCgP_I,
        Grb,
        Shc,
        SOS     AS      Concentration

#
#       Calculated quantities
#
        R_BOUND_SOS,
        TOTAL_P_PLCg,
        TOTAL_P_Shc,
        TOTAL_R_Grb,
        TOTAL_Grb_Shc AS Concentration

#
#       Rates
#
        u       AS      ARRAY(NREACS) OF Rate

#
#       Inputs
#
        EGFT  AS Concentration

SET
        NFORWD:=22; # Number of forward reactions
        NREVRS:=22; # Number of reverse reactions
        NMICHL:=3;  # Number of M-M reactions
        NREACS:=25; # Total number of reactions

#
#       Mass balance constraints
#
```

```
        EGFRT := 100;
        PLCgT := 105;
        GrbT := 85;
        ShcT := 150;
        SOST := 34;

#
#       Elementary reaction parameters
#
        kforwd(1):=0.003;
        krevrs(1):=0.06;

        kforwd(2):=0.01;
        krevrs(2):=0.1;

        kforwd(3):=1;
        krevrs(3):=0.01;

        kforwd(4):=0.06;
        krevrs(4):=0.2;

        kforwd(5):=1;
        krevrs(5):=0.05;

        kforwd(6):=0.3;
        krevrs(6):=0.006;

        kforwd(7):=0.003;
        krevrs(7):=0.05;

        kforwd(8):=0.01;
        krevrs(8):=0.06;

        kforwd(9):=0.03;
        krevrs(9):=4.5E−3;

        kforwd(10):=1.5E−3;
        krevrs(10):=1E−4;

        kforwd(11):=0.09;
        krevrs(11):=0.6;

        kforwd(12):=6;
        krevrs(12):=0.06;

        kforwd(13):=0.3;
        krevrs(13):=9E−4;

        kforwd(14):=0.003;
        krevrs(14):=0.1;

        kforwd(15):=0.3;
        krevrs(15):=9E−4;
        kforwd(16):=0.01;
        krevrs(16):=2.14E−2;

        kforwd(17):=0.12;
        krevrs(17):=2.4E−4;

        kforwd(18):=0.003;
        krevrs(18):=0.1;

        kforwd(19):=0.03;
        krevrs(19):=0.064;

        kforwd(20):=0.1;
        krevrs(20):=0.021;

        kforwd(21):=0.009;
        krevrs(21):=4.29E−2;

        kforwd(22):=1;
        krevrs(22):=0.03;
#
#       Michaelis-Menton parameters
#
        V(1):=450;
        K(1):=50;

        V(2):=1;
        K(2):=100;

        V(3):=1.7;
        K(3):=340;

EQUATION

#       $EGF = -u(1);
#       $R = -u(1);
        $Ract = u(1)−2*u(2);
        $Rdimer = u(2)+u(4)−u(3);
        $RP = u(3)+u(7)+u(11)+u(15)+u(18)+u(20)−u(4)−u(5)−u(9)−u(13);
#       $R_PL = u(5)-u(6);
        $R_PLP = u(6)−u(7);
        $R_G = u(9)−u(10);
        $R_G_S = u(10)−u(11);
        $R_Sh = u(13)−u(14);
        $R_ShP = u(14)−u(24)−u(15)−u(17);
        $R_Sh_G = u(17)−u(18)−u(19);
        $R_Sh_G_S = u(19)−u(20)+u(24);
        $G_S = u(11)+u(23)−u(12)−u(24);
        $ShP = u(15)+u(23)−U(21)−u(16);
        $Sh_G = u(18)+u(21)−u(22);
        $PLCg = u(8)−u(5);
        $PLCgP = u(7)−u(8)−u(25);
        $PLCgP_I = u(25);
#       $Grb = u(12)-u(9)-u(17)-u(21);
#      $Shc = u(16)-u(13);
#      $SOS = u(12)-u(10)-u(19)-u(22);
        $Sh_G_S = u(20)+u(22)-u(23);


#
#      Elementary and MM reactions
#
        u(1) = kforwd(1)*R*EGF - krevrs(1)*Ract;
        u(2) = kforwd(2)*Ract*Ract - krevrs(2)*Rdimer;
        u(3) = kforwd(3)*Rdimer - krevrs(3)*RP;
        u(4) = V(1)*RP/(K(1)+RP);
        u(5) = kforwd(4)*RP*PLCg - krevrs(4)*R_PL;

        u(6) = kforwd(5)*R_PL - krevrs(5)*R_PLP;
        u(7) = kforwd(6)*R_PLP - krevrs(6)*R*PLCgP;
        u(8) = V(2)*PLCgP/(K(2)+PLCgP);
        u(9) = kforwd(7)*RP*Grb - krevrs(7)*R_G;
        u(10) = kforwd(8)*R_G*SOS - krevrs(8)*R_G_S;

        u(11) = kforwd(9)*R_G_S - krevrs(9)*RP*G_S;
        u(12) = kforwd(10)*G_S - krevrs(10)*Grb*SOS;
        u(13) = kforwd(11)*RP*Shc - krevrs(11)*R_Sh;
        u(14) = kforwd(12)*R_Sh - krevrs(12)*R_ShP;
        u(15) = kforwd(13)*R_ShP - krevrs(13)*ShP*RP;

        u(16) = V(3)*ShP/(K(3)+ShP);
        u(17) = kforwd(14)*R_ShP*Grb - krevrs(14)*R_Sh_G;
        u(18) = kforwd(15)*R_Sh_G - krevrs(15)*RP*Sh_G;
        u(19) = kforwd(16)*R_Sh_G*SOS - krevrs(16)*R_Sh_G_S;
        u(20) = kforwd(17)*R_Sh_G_S - krevrs(17)*Sh_G_S*RP;

        u(21) = kforwd(18)*ShP*Grb - krevrs(18)*Sh_G;
        u(22) = kforwd(19)*Sh_G*SOS - krevrs(19)*Sh_G_S;
        u(23) = kforwd(20)*Sh_G_S - krevrs(20)*ShP*G_S;
        u(24) = kforwd(21)*R_ShP*G_S - krevrs(21)*R_Sh_G_S;
        u(25) = kforwd(22)*PLCgP - krevrs(22)*PLCgP_I;

#
#      Mass balance constraints
#

        EGFRT = R + Ract + 2*(Rdimer + RP + R_PL + R_PLP + R_G + R_G_S
                + R_Sh + R_ShP + R_Sh_G + R_Sh_G_S);

        EGFT = EGF + Ract + 2*(Rdimer + RP + R_PLP + R_PL + R_Sh + R_ShP
                + R_G + R_G_S + R_Sh_G + R_Sh_G_S);

        PLCgT = R_PL + R_PLP + PLCg + PLCgP + PLCgP_I;

        GrbT = Grb + G_S + Sh_G + Sh_G_S + R_G + R_G_S + R_Sh_G + R_Sh_G_S;

        ShcT = Shc + ShP + Sh_G + Sh_G_S + R_Sh + R_ShP + R_Sh_G + R_Sh_G_S;
        SOST = SOS + G_S + Sh_G_S + R_G_S + R_Sh_G_S;

#
#       Calculated Quantities
#
      R_BOUND_SOS = R_G_S + R_Sh_G_S;
      TOTAL_P_PLCg = R_PLP + PLCgP;
      TOTAL_P_Shc = R_ShP + R_Sh_G + R_Sh_G_S + ShP + Sh_G + Sh_G_S;
      TOTAL_R_Grb = R_G + R_G_S + R_Sh_G + R_Sh_G_S;
      TOTAL_Grb_Shc = R_Sh_G + Sh_G + R_Sh_G_S + Sh_G_S;
END

SIMULATION SHORT_TERM_EGF
OPTIONS
ALGPRINTLEVEL:=1;
ALGRTOLERANCE:=1E−9;
ALGATOLERANCE:=1E−9;
ALGMAXITERATIONS:=100;
DYNPRINTLEVEL:=0;
DYNRTOLERANCE:=1E−9;
DYNATOLERANCE:=1E−9;
CSVOUTPUT := TRUE;
UNIT EGF_Kinetics AS EGF

REPORT
 EGF_Kinetics.TOTAL_P_PLCg, EGF_Kinetics.R_G_S,
 EGF_Kinetics.EGFT, EGF_Kinetics.R_Sh_G_S

INPUT

#
#       Mass balance constraints
#
        WITHIN EGF_Kinetics DO
        EGFT :=300;
        END
INITIAL
STEADY_STATE

SCHEDULE
SEQUENCE
SAVE PRESETS TEST
CONTINUE FOR 10
RESET
        EGF_Kinetics.EGFT := 350;
END
CONTINUE FOR 120
END
END
```

## B.4 Distillation Model

```
#=======================================
#
#        Distillation Model (Final Column of HDA Distillation train)
#
#        Based on distillation model written for 10.551 Systems
#        Engineering Class
#
#=======================================

DECLARE

TYPE
# Identifier          # Default     # Lower      #Upper
  NoType              =0.9          : −1E9       : 1E9   UNIT="−"
  MoleComposition     =0.5          : 0          : 1     UNIT="kmol/kmol"
  Temperature         =373          : 100        : 473   UNIT="K"
  MoleFlow            =100          : 0          : 1E3   UNIT="kmol/hr"
  Pressure            =1.0135       : 1E−9       : 100   UNIT="Bar"
  Energy              =50           : −1E3       : 1E3   UNIT="MJ/kmol"
  EnergyHoldup        =20           : −1E6       : 1E6   UNIT="MJ"
  MoleHoldup          =10           : 1e−9       : 1000  UNIT="kmol"
  MolecularWeight     =20           : 1e−9       : 1000  UNIT="kg/kmol"
  Density             =800          : 1e−9       : 1000  UNIT="kg/m^3"
  Percent             =50           : 0          : 100   UNIT="%"
  Control_Signal      =50           : −1E9       : 1E9   UNIT="−"
  Heat                =1e4          : −1E7       : 1E7   UNIT="MJ/hr"
  Length              =1            : 1e−9       : 20    UNIT="m"
  Area                =1            : 0          : 100   UNIT="m^2"
  SpecificVolume      =1            : 1e−9       : 1500  UNIT="m^3/kmol"
  Velocity            =1            : 0          : 100   UNIT="m/s"
  SpecificArea        =1            : 0          : 1E2   UNIT="10E−1m^2/mmol"
  SurfaceTension      =1            : 0          : 1000  UNIT="dyne/cm"

STREAM

  Process_stream      IS     MoleFlow,
                             MoleComposition,
                             Temperature,
                             Pressure,
                             Energy,
                             SpecificVolume,
                             Density

END

#
#        End of declare section
#

MODEL LiquidProperties
#=======================================
#
#       Simple thermodynamic model for:     Liquid enthalpy
#                                           Liquid molecular weight
#                                           Liquid density
#                                           Liquid surface tension
#
#       Physical properties taken from:
#       Reid R. C., Prausnitz J. M., Poling B. E.,
#       The Properties of Gases and Liquids, 4th Ed, McGraw Hill, 1987.
#
#       Parameter       Description                         Units
#       ———             ————                                —–
#       R               Gas constant                        (MJ/kmol K)
#       NC              Number of components
#       alpha           Index for Watson equation
#       no              Avogadros Number
#       CPA, CPB, ..    Idea heat capacity coeffs
#       TC              Critical Temperatures               (K)
#       TBR             Reduced boiling temperature         (K)
#       PC              Critical pressure                   (bar)
#       MW              Pure component molecular weight     (kg/kmol)
#       DHf             Pure component heat of formation    (MJ/kmol)
#       ZRA             Rackett compressibility factor      (-)
#
#       Variable        Description                         Units
#       ——–             ————                                —–
#
#       x               Array of liquid mole fractions      (kmol/kmol)
#       TR              Array of reduced temperatures       (K/K)
#       DHvb            Pure comp. heat of vapor. at b.p.   (MJ/kmol)
#       DHv             Pure comp. heat of vapor.           (MJ/kmol)
#       Hvi             Pure comp. vapor enthalpy           (MJ/kmol)
#       Hli             Pure comp. liquid enthalpy          (MJ/kmol)
#       Ai              Pure comp. specific area            (1E4 m^2/mol)
#       Vs              Pure comp. liquid specific volume   (m^3/kmol)
#       sigi            Pure comp. liquid surface tension   (dyne/cm)
#       Q               Intermediate in surface tension calc
#       P               Pressure                            (bar)
#       T               Temperature                         (K)
#       mwl             Molecular weight of liquid mixture  (kg/kmol)
#       rhol            Density of liquid mixture           (kg/m^3)
#       hl              Liquid mixture enthalpy             (MJ/kmol)
#       A               Specific area of mixture            (1E4 m^2/mol)
#       voll            Liquid mixture specific volume      (m^3/kmol)
#       sigl            Liquid mixture surface tension      (dyne/cm)
#
#       Modifications:
#========================================

PARAMETER
        R               AS      REAL
        NC              AS      INTEGER
        alpha           AS      REAL
        no              AS      REAL
```

```
    CPA, CPB, CPC,
    CPD                 AS ARRAY(NC) OF REAL
    TC                  AS ARRAY(NC) OF REAL
    TBR                 AS ARRAY(NC) OF REAL
    PC                  AS ARRAY(NC) OF REAL
    MW                  AS ARRAY(NC) OF REAL
    DHf                 AS ARRAY(NC) OF REAL
    ZRA                 AS ARRAY(NC) OF REAL

VARIABLE

    x           AS      ARRAY(NC) OF MoleComposition
    TR          AS      ARRAY(NC) OF NoType
    DHvb, DHv,
    Hvi, Hli    AS      ARRAY(NC) OF Energy
    Ai          AS      ARRAY(NC) OF SpecificArea
    Vs          AS      ARRAY(NC) OF SpecificVolume
    sigi        AS      ARRAY(NC) OF SurfaceTension
    Q           AS      ARRAY(NC) OF NoType

    P           AS      Pressure
    T           AS      Temperature
    mwl         AS      MolecularWeight
    rhol        AS      Density
    hl          AS      Energy
    A           AS      SpecificArea
    voll        AS      SpecificVolume
    sigl        AS      SurfaceTension

SET

#   Component properties are set here.
#   Assumes two components: Component 1 is Toluene and component 2 is
#   Benzene

#   Gas constant
    R           := 0.0083144; # MJ/(kmol K)

#   Avogadro's number
    no          := 6.023E5;

#   Watson index
    alpha       := 0.38;

#   Molecular weights
    MW(1)       := 92.141;
    MW(2)       := 78.114;

#   Critical Temperatures
    TC(1)       := 591.8;
    TC(2)       := 562.2;

#   Reduced boiling temperature
```

```
        TBR(1)        := 383.8/591.8;
        TBR(2)        := 353.2/562.2;

#       Critical Pressures
        PC(1)         := 41.0;
        PC(2)         := 48.9;

#       Enthalpies of formation
        DHf(1)        := 5.003E+1;
        DHf(2)        := 8.298E+1;

#       Ideal Heat Capacity coeffs.
        CPA(1)        := -2.435E-2;
        CPB(1)        := 5.125E-4;
        CPC(1)        := -2.765E-7;
        CPD(1)        := 4.911E-11;

        CPA(2)        := -3.392E-2;
        CPB(2)        := 4.739E-4;
        CPC(2)        := -3.017E-7;
        CPD(2)        := 7.130E-11;

#       Rackett parameters for liquid molar volume
        ZRA(1)        := 0.2644;
        ZRA(2)        := 0.2698;

EQUATION

#       Reduced temperature
        TR*TC = T;

#       Giacalone Equation
        DHvb = R*TC*TBR*LOG(PC/1.01325)/(1-TBR);

#       Watson Equation
        DHv = DHvb*((1-TR)/(1-TBR))^alpha;

#       Pure component vapor enthalpy
        Hvi = CPA*(T-298.2)+CPB/2*(T^2-298.2^2)+CPC/3*(T^3-298.2^3)+
        CPD/4*(T^4-298.2^4)+DHf;

#       Pure component liquid enthalpy
        Hli = Hvi-DHv;

#       Liquid mixture enthalpy
        hl = SIGMA(Hli*x);

#       Average liquid molecular weight
        mwl = SIGMA(MW*x);

#       Pure component liquid molar volume
        Vs = 10*R*TC/PC*ZRA^(1+(1-TR)^(2.0/7.0));

#       Liquid mixture specific volume
```

```
          voll = SIGMA(Vs*x);

#         Liquid density
          rhol = mwl / voll;

#         Sum of liquid mole fractions
          SIGMA(x) = 1;

#         Pure component surface tension (Corresponding states)
          sigi = PC^(2.0/3)*TC^(1.0/3)*Q*(1-TR)^(11.0/9);
          Q = 0.1196*(1+TBR*LOG(PC/1.01325)/(1-TBR))-0.279;

#         Liquid mixture surface tension for binary mixture
#         (assumes ideality)
          Ai = Vs^(2.0/3)*no^(1.0/3);
          A = 0.5*SIGMA(Ai);
          sigl = SIGMA(x*sigi) - A/(200*R*T)*(sigi(1)-sigi(2))^2*x(1)*x(2);

END

MODEL PhysicalProperties INHERITS LiquidProperties
#=======================================CE
#
#         Simple thermodynamic model for:   K Values
#                                           Vapor enthalpy
#                                           Vapor density
#
#         Physical properties taken from:
#         Reid R. C., Prausnitz J. M., Poling B. E.,
#         The Properties of Gases and Liquids, 4th Ed, McGraw Hill, 1987.
#
#         Parameter      Description                           Units
#         ---------      -----------                           -----
#         VPA, VPB..     Modified Antoine coefficients
#
#         Variable       Description                           Units
#         --------       -----------                           -----
#
#         y              Array of vapor mole fractions         (kmol/kmol)
#         logPvap        log of pure component vapor pressure  (-)
#         logK           log of K value                        (-)
#         mwv            Molecular weight of vapor             (kg/kmol)
#         rhov           Vapor mixture density                 (kg/m^3)
#         hv             Vapor mixture enthalpy                (MJ/kmol)
#         volv           Vapor mixture specific volume         (m^3/kmol)
#
#         Modifications:
#=======================================
PARAMETER

  VPA, VPB, VPC,
  VPD                 AS ARRAY(NC) OF REAL

VARIABLE
```

```
  y            AS     ARRAY(NC) OF MoleComposition
  logPvap,
  logK         AS     ARRAY(NC) OF NoType
  mwv  AS     MolecularWeight
  rhov AS     Density
  hv   AS     Energy
  volv AS     SpecificVolume

SET

#       Component properties are set here.
#       Assumes two components: Component 1 is Toluene and component 2 is
#       Benzene

#       Extended Antoine coeffs.
        VPA(1)         := -7.28607;
        VPB(1)         := 1.38091;
        VPC(1)         := -2.83433;
        VPD(1)         := -2.79168;

        VPA(2)         := -6.98273;
        VPB(2)         := 1.33213;
        VPC(2)         := -2.62863;
        VPD(2)         := -3.33399;

EQUATION

#       Extended Antoine Vapor pressure of each component
        (logPvap - LOG(PC))*TR = (VPA*(1-TR)+VPB*(1-TR)^1.5+VPC*(1-TR)^3+
                                  VPD*(1-TR)^6);

#       Vapor mixture enthalpy
        hv = SIGMA(Hvi*y);

#       K-value
        logK = logPvap - LOG(P);

#       Average vapor molecular weight
        mwv = SIGMA(MW*y);

#       Vapor mixture specific volume
        volv = 10*R*T/P;

#       Vapor density
        rhov = mwl / volv;

#       Sum of vapor mole fractions
        SIGMA(y) = 1;
END

MODEL Flash INHERITS PhysicalProperties
#========================================
#
```

```
#    Generic dynamic flash model
#
#    Parameter    Description                         Units
#    ———          ————                                —–
#    Vtot         Volume of flash tank                (m^3)
#    AT           Cross-sectional area of tank        (m^2)
#    g            Gravitational constant              (m/s^2)
#
#    Variable     Description                         Units
#    ——–          ————                                —–
#
#    hlin         Specific enthalpy of liquid feed    (MJ/kmol)
#    vollin       Specific volume of liquid feed      (m^3/kmol)
#    rholin       Density of liquid feed              (kg/m^3)
#    Level        Liquid level in tank                (m)
#    Tin          Temperature of liquid feed          (K)
#    Pin          Pressure of liquid feed             (bar)
#    Pout         Outlet pressure of liquid           (bar)
#    z            Array of mole fraction of feed      (kmol/kmol)
#    F            Feed flow rate                      (kmol/hr)
#    L            Liquid outlet flowrate              (kmol/hr)
#    V            Vapor outlet flowrate               (kmol/hr)
#    N            Array of comp. total mole holdups   (kmol)
#    Nv           Vapor mole holdup                   (kmol)
#    Nl           Liquid mole holdup                  (kmol)
#    U            Internal energy of contents         (MJ)
#    Qh           Heat supplied to vessel             (MJ/hr)
#
#    Modifications:
#======================================

PARAMETER
    Vtot    AS    REAL
    AT      AS    REAL
    g       AS    REAL

VARIABLE

    hlin    AS    Energy
    vollin  AS    SpecificVolume
    rholin  AS    Density
    Level   AS    Length
    Tin     AS    Temperature
    Pin,
    Pout    AS    Pressure
    z       AS    ARRAY(NC) OF MoleComposition
    F, L,
    V       AS    MoleFlow
    N       AS    ARRAY(NC) OF MoleHoldup
    Nv,
    Nl      AS    MoleHoldup
    U       AS    EnergyHoldup
    Qh      AS    Heat
```

```
STREAM
  Feed:       F, z, Tin, Pin, hlin, vollin, rholin    AS Process_stream
  Vapor:      V, y, T, P, hv, volv, rhov              AS Process_Stream
  Liquid:     L, x, T, Pout, hl, voll, rhol           AS Process_Stream

EQUATION

#       Species Balance
        $N = F*z−L*x − V*y;

#       Energy Balance
        $U = F*hlin − V*hv − L*hl + Qh;

#       Equilibrium
        LOG(y) = logK + LOG(x);

#       Definition of molar holdups
        N = Nv*y + Nl*x;

#       Definition of energy holdups
        U + 0.1*P*VTot = Nv*hv + Nl*hl;

#       Volume constraint
        Vtot = Nv*volv + Nl*voll;

#       Outlet liquid pressure based on static head
        Level*AT = Nl*voll;
        Pout = P + 1E−5*Level*rhol*g;

END

MODEL Downcomer INHERITS LiquidProperties
#========================================
#
#       Simple mass and energy balance model of downcomer:
#       Assumes negligible dP/dt term
#
#       Parameter     Description                         Units
#       ———           ————                                ——
#       Ad            Cross-sectional area of downcomer   (m^2)
#       g             Gravitational constant              (m/s^2)
#
#       Variable      Description                         Units
#       ——            ————                                ——
#
#       Nl            Liquid mole holdup in downcomer     (kmol)
#       Tin           Inlet liquid temperature            (K)
#       Pin           Pressure                            (bar)
#       hlin          Specific enthalpy of inlet liquid   (MJ/kmol)
#       xin           Inlet liquid mole composition       (kmol/kmol)
#       vollin        Specific volume of inlet liquid     (m^3/kmol)
#       rholin        Density of inlet liquid             (kg/m^3)
#       Lin           Inlet liquid flowrate               (kmol/hr)
#       Lout          Outlet liquid flowrate              (kmol/hr)
#       Level          Liquid level in downcomer             (m)
#
#       Modifications:
#=========================================

PARAMETER
        Ad             AS     REAL
        g              AS     REAL

VARIABLE
        Nl             AS     MoleHoldup
        Tin            AS     Temperature
        Pin            AS     Pressure
        hlin           AS     Energy
        xin            AS     ARRAY(NC) OF MoleComposition
        vollin         AS     SpecificVolume
        rholin         AS     Density
        Lin, Lout      AS     MoleFlow
        Level          AS     Length

STREAM
  Liqin:  Lin, xin, Tin, Pin, hlin, vollin, rholin       AS Process_stream
  Liqout: Lout, x, T, P, hl, voll, rhol                  AS Process_Stream

EQUATION
#       Overall mass balance
        $Nℓ = Lin − Lout;

#       Component balance
        FOR I:=1 TO NC−1 DO
                Nl*$x(I) = Lin*(xin(I)−x(I));
        END

#       Energy balance - Neglects pressure term
        Nl*$hℓ = Lin*(hlin − hl);

#       Outlet pressure
        Level*Ad = Nl*voll;
        P = Pin + 1E−5*rhol*g*Level;

END

MODEL Tray INHERITS PhysicalProperties
#=========================================
#
#       Model of distillation tray and downcomer:
#
#               Equilibrium stage model
#               Full hyrdrodynamics
#               Negligible liquid and vapor entrainment
#               Downcomer is sealed
#
#               Assumes that dV/dt term is negligible in energy balance
#               on tray
```

```
#
#      Lin                                   Tray p
#    |    |    |
#    |    V    |
#    |         |
#    |_______|——————————————- Pp-1
#    |         |
#    |         |        Vout
#    |         |         ^
#    |         |_______|_________ ————- Pp
#    |         |                    \
#    |LDout ->              |     |     |
#    |_______________________| | |       |
#                        ^      | | |       | — Pp+1
#                        |          V
#                       Vin       Lout
#
#    Parameter    Description                           Units
#    ————         —————                                 ——
#
#    g            Gravitational constant                (m/s^2)
#    PI
#    dh           Diameter of sieve holes               (m)
#    tt           Tray thickness                        (m)
#    Ad           Cross-sectional area of downcomer     (m^2)
#    Ac           Cross-sectional area of column        (m)
#    phi          Fraction of hole to bubbling area     (m^2/m^2)
#    k            Dry plate pressure drop coeff.
#    hw           Weir height                           (m)
#    hd           Clearance under downcomer             (m)
#    Cdd          Discharge coefficient for downcomer
#
#    Variable     Description                           Units
#    ———          —————                                 ——
#
#    Lin          Liquid flowrate into downcomer        (kmol/hr)
#    xin          Liquid inlet mole composition         (kmol/kmol)
#    Tlin         Inlet liquid temperature              (K)
#    Plin         Pressure on plate p-1                 (bar)
#    hlin         Specific enthalpy of inlet liquid     (MJ/kmol)
#    vollin       Specific volume of inlet liquid       (m^3/kmol)
#    rholin       Density of inlet liquid               (kg/m^3)
#    LDout        Liquid flowrate out of downcomer      (kmol/hr)
#    xD           Downcomer outlet composition          (kmol/kmol)
#    TD           Temperature of downcomer outlet       (K)
#    PD           Pressure at base of downcomer         (bar)
#    hlD          Specific enthalpy of downcomer outlet (MJ/kmol)
#    vollD        Specific volume of downcomer outlet   (m^3/kmol)
#    rholD        Density of downcomer outlet           (kg/m^3)
#    Vin          Vapor flowrate onto plate             (kmol/hr)
#    yin          Inlet vapor composition               (kmol/kmol)
#    Tvin         Inlet vapor temperature               (K)
#    Pvin         Inlet vapor pressure                  (bar)
#    hvin         Inlet vapor specific enthalpy         (MJ/kmol)
#       volvin          Inlet vapor specific volume             (m^3/kmol)
#       rhovin          Inlet vapor density                     (kg/m^3)
#       Lout            Liquid outlet flowrate                  (kmol/hr)
#       Vout            Vapor flowrate off plate                (kmol/hr)
#       Nl              Liquid mole holdup on plate             (kmol)
#       U               Internal energy of liquid on plate      (MJ)
#       DeltaPr         Pressure drop due to surface tension    (bar)
#       DeltaPdt        Pressure drop due to dry plate          (bar)
#       DeltaPcl        Pressure drop due to clear liquid       (bar)
#       DeltaPcli       Pressure drop due to clear liquid
#                       at entrance to plate                    (bar)
#       DeltaPudc       Pressure drop due to flow out of
#                       downcomer                               (bar)
#       Ab              Bubbling area                           (m^2)
#       Ah              Area covered by sieve holes             (m^2)
#       uh              Super. vel. based on hole area          (m/s)
#       us              Super. vel. based on bubbling area      (m/s)
#       psi             Discharge coefficient for plate
#       Fr              Froude number                           (-)
#       FrP             Froude number                           (-)
#       eps             Aeration factor                         (-)
#       Cd              Discharge coefficient over weir         (-)
#       how             Height of liquid over weir              (m)
#       hcl             Height of clear liquid                  (m)
#       theta           Angle subtended by downcomer            (rads)
#       W               Length of weir                          (m)
#
#       Modifications:
#========================================

PARAMETER
        g       AS REAL
        PI      AS REAL

        dh      AS REAL
        tt      AS REAL
        Ad      AS REAL
        Ac      AS REAL
        phi     AS REAL
        k       AS REAL
        hw      AS REAL
        hd      AS REAL
        Cdd     AS REAL

UNIT
        Downcomer AS Downcomer

VARIABLE
#       Liquid in
        Lin     AS MoleFlow
        xin     AS ARRAY(NC) OF MoleComposition
        Tlin    AS Temperature
        Plin    AS Pressure
        hlin    AS Energy
        vollin  AS SpecificVolume
        rholin  AS Density

#       Liqiud out of downcomer
        LDout   AS MoleFlow
        xD      AS ARRAY(NC) OF MoleComposition
        TD      AS Temperature
        PD      AS Pressure
        hlD     AS Energy
        vollD   AS SpecificVolume
        rholD   AS Density

#       Vapor in
        Vin     AS MoleFlow
        yin     AS ARRAY(NC) OF MoleComposition
        Tvin    AS Temperature
        Pvin    AS Pressure
        hvin    AS Energy
        volvin  AS SpecificVolume
        rhovin  AS Density

#       Liquid flowrate out
        Lout    AS MoleFlow

#       Vapor flowrate out
        Vout    AS MoleFlow

#       Tray holdup
        Nl      AS MoleHoldup
        U       AS EnergyHoldup

#       Hydrodynamics
        DeltaPr,
        DeltaPdt,
        DeltaPcl,
        DeltaPcli,
        DeltaPudc       AS Pressure
        Ab, Ah          AS Area
        uh, us          AS Velocity
        psi, Fr, FrP,
        eps, Cd         AS NoType
        how, hcl        AS Length

#       Tray geometry
        theta           AS NoType
        W               AS Length

STREAM
#       Downcomer
  Liqin:   Lin, xin, Tlin, Plin, Hlin, vollin, rholin   AS    Process_stream
  LiqDout: LDout, xD, TD, PD, hlD, vollD, rholD         AS    Process_stream

#       Tray
  Liqout:  Lout, x, T, P, hl, voll, rhol                AS    Process_stream
  Vapin:  Vin, yin, Tvin, Pvin, hvin, volvin, rhovin AS    Process_Stream
  Vapout: Vout, Y, T, P, hv, volv, rhov              AS    Process_Stream

SET
        dh := 0.005; # Lockett Recommendation
        tt := 0.0025; # Tray thickness
        Ad := 0.25; # Cross-sectional area of downcomer
        Ac := 1.5;  # Cross-sectional area of column
        phi := 0.1;  # Fraction of hole area to bubbling area
        k := 0.94;   # Assumes triangular pitch
        hw := 0.05; # Weir height
        hd := 0.025; # Clearance under downcomer
        Cdd := 0.56; # Discharge coefficient for downcomer (Koch)

EQUATION
#       Overall Mass Balance
        $Nℓ = LDout − Lout + Vin − Vout;

#       Component Balance
        FOR  I:=1 TO NC−1 DO
        Nl * $x(I) = LDout*(xD(I)−x(I)) + Vin*(yin(I)−x(I))+ Vout*(x(I)−y(I));
        END

#       Energy Balance
        U + 0.1*P*Nl*voll = Nl*hl;

        $U = LDout*hlD − Lout*hl + Vin*hvin − Vout*hv;

#       Equilibrium
        LOG(y) = logK + LOG(x);

#       Connect Downcomer to tray
        Liqin = Downcomer.Liqin;
        LiqDout = Downcomer.Liqout;

#
#       Hydrodynamics: All correlations from Lockett M. J., Distillation tray
#       fundamentals, Cambridge University Press, 1986. Reported original
#       references for completeness
#


#
#       Calculate weir length
#

        2*Ad/Ac * PI = theta − SIN(theta);
        W = (4*Ac/PI)^0.5*sin(theta/2);


#
#       Vapor flow onto plate
#


#
#       Residual pressure drop:
```

```
#       Van Winkle M., Distillation, McGraw-Hill, 1967
#       Fair J. R., (In Smith B. D.) Design of Equilibrium Stage Processes,
#       Chp 15, McGraw-Hill, 1963.
#
        DeltaPr = 4E−8*sigl/dh;

#
#       Dry plate pressure drop:
#       Cervenka J. and Kolar V. Hyrdodynamics of plate columns VIII,
#       Czech. Cem. Comm. 38, pp 2891, 1973.
#

        phi*Ab = Ah;
        Ab = Ac−2*Ad;
        uh = Vin * volvin/(3600*Ah);
        psi = k*(1−phi^2)/(phi*tt/dh)^0.2;
        DeltaPdt = 1E−5*psi*rhovin*uh^2/2;

#
#       Clear liquid pressure drop from mass balance
#

        DeltaPcl = 1E−5*Nl*mwl*g/Ab;

#
#       Pressure drop across plate (defines vapor flowrate)
#

        Pvin − P = DeltaPdt + DeltaPcl + DeltaPr;

#
#       Liquid flowrate off plate
#

#
#       Clear liquid pressure drop:
#       Colwell C. J., Clear liquid height and froth density on sieve trays,
#       Ind. Eng. Chem. Proc. Des. Dev., 20(2), pp 298, 1979.
#

        DeltaPcl = 1E−5*rhol*g*hcl;
        us = Vin * volvin/(3600*Ab);

        eps = 12.6*(1−eps)*FrP^0.4*phi^(−0.25);
        FrP*(rhol−rhovin) = Fr*rhovin;
        Fr * hcl= us^2/g;
        hcl = (1−eps)*(hw+0.7301*(Lout*voll/(3600*W*Cd*(1−eps)))^0.67);
        how *(1−eps)= hcl−hw*(1−eps);

        IF how/hw > 8.14 THEN
                Cd*how^1.5 = 1.06*(how+hw)^1.5;
        ELSE
                Cd*hw = 0.61*hw+0.08*how;
        END
```

```
#
#       Liquid flowrate onto plate from downcomer: Momentum balance
#

        DeltaPcli = 1E-5*rhold*g*(2./g*(LDout*vollD/(3600*W))^2*(1./hcl-1./hd) +
                    2./3*hcl^2/(1-eps))^0.5;


        DeltaPudc = 1E-5*rhold/2*(LDout*vollD/(3600*W*hd*Cdd))^2;

        Pd - P = DeltaPcli + DeltaPudc;

END

MODEL TopTray INHERITS PhysicalProperties
#========================================
#       Top tray model:
#               Top tray does not have a downcomer associated with it!!!
#               Equilibrium stage model
#               Full hyrdrodynamics
#               Negligible liquid and vapor entrainment
#               Assumes that dV/dt term is negligible in energy balance
#               on tray
#
#          Lin          Tray p
#           |
#           V
#               Vout
#                ^
#        --------|------------------  ---- Pp
#       |                            \
#       |LDout ->                |    |   |
#       |------------------------| | |    |
#                       ^        | | |    | -- Pp+1
#                       |          V
#                      Vin       Lout
#
#       Parameter       Description                             Units
#       ---------       -----------                             -----
#
#       g               Gravitational constant                  (m/s^2)
#       PI
#       dh              Diameter of sieve holes                 (m)
#       tt              Tray thickness                          (m)
#       Ad              Cross-sectional area of downcomer       (m^2)
#       Ac              Cross-sectional area of column          (m)
#       phi             Fraction of hole to bubbling area       (m^2/m^2)
#       k               Dry plate pressure drop coeff.
#       hw              Weir height                             (m)
#
#       Variable        Description                             Units
#       --------        -----------                             -----
#
```

```
#       Lin          Liquid flowrate into downcomer        (kmol/hr)
#       xin          Liquid inlet mole composition         (kmol/kmol)
#       Tlin         Inlet liquid temperature              (K)
#       Plin         Pressure on plate p-1                 (bar)
#       hlin         Specific enthalpy of inlet liquid     (MJ/kmol)
#       vollin       Specific volume of inlet liquid       (m^3/kmol)
#       rholin  Density of inlet liquid                    (kg/m^3)
#       Vin          Vapor flowrate onto plate             (kmol/hr)
#       yin          Inlet vapor composition               (kmol/kmol)
#       Tvin         Inlet vapor temperature               (K)
#       Pvin         Inlet vapor pressure                  (bar)
#       hvin         Inlet vapor specific enthalpy         (MJ/kmol)
#       volvin       Inlet vapor specific volume           (m^3/kmol)
#       rhovin       Inlet vapor density                   (kg/m^3)
#       Lout         Liquid outlet flowrate                (kmol/hr)
#       Vout         Vapor flowrate off plate              (kmol/hr)
#       Nl           Liquid mole holdup on plate           (kmol)
#       U            Internal energy of liquid on plate    (MJ)
#       DeltaPr      Pressure drop due to surface tension  (bar)
#       DeltaPdt     Pressure drop due to dry plate        (bar)
#       DeltaPcl     Pressure drop due to clear liquid     (bar)
#       Ab           Bubbling area                         (m^2)
#       Ah           Area covered by sieve holes           (m^2)
#       uh           Super. vel. based on hole area        (m/s)
#       us           Super. vel. based on bubbling area    (m/s)
#       psi          Discharge coefficient for plate
#       Fr           Froude number                         (-)
#       FrP          Froude number                         (-)
#       eps          Aeration factor                       (-)
#       Cd           Discharge coefficient over weir       (-)
#       how          Height of liquid over weir            (m)
#       hcl          Height of clear liquid                (m)
#       theta        Angle subtended by downcomer          (rads)
#       W            Length of weir                        (m)
#
#       Modifications:
#=========================================
PARAMETER
        g       AS REAL
        PI      AS REAL

        dh      AS REAL
        tt      AS REAL
        Ad      AS REAL
        Ac      AS REAL
        phi     AS REAL
        k       AS REAL
        hw      AS REAL


VARIABLE
#       Liquid in
        Lin     AS MoleFlow
        xin     AS ARRAY(NC) OF MoleComposition
        Tlin    AS Temperature
        Plin    AS Pressure
        hlin    AS Energy
        vollin  AS SpecificVolume
        rholin  AS Density

#       Vapor in
        Vin     AS MoleFlow
        yin     AS ARRAY(NC) OF MoleComposition
        Tvin    AS Temperature
        Pvin    AS Pressure
        hvin    AS Energy
        volvin  AS SpecificVolume
        rhovin  AS Density

#       Liquid flowrate out
        Lout    AS MoleFlow

#       Vapor flowrate out
        Vout    AS MoleFlow

#       Tray holdup
        Nl      AS MoleHoldup
        U       AS EnergyHoldup

#       Hydrodynamics
        DeltaPr,
        DeltaPdt,
        DeltaPcl,

        Ab, Ah          AS Area
        uh, us          AS Velocity
        psi, Fr, FrP,
        eps, Cd         AS NoType
        how, hcl        AS Length

#       Tray geometry
        theta           AS NoType
        W               AS Length

STREAM

#       Tray
  Liqin:   Lin, xin, Tlin, Plin, hlin, vollin, rholin    AS   Process_stream
  Liqout:  Lout, x, T, P, hl, voll, rhol                 AS   Process_stream
  Vapin:   Vin, yin, Tvin, Pvin, hvin, volvin, rhovin    AS   Process_Stream
  Vapout:  Vout, Y, T, P, hv, volv, rhov                 AS   Process_Stream

SET
        dh := 0.005; # Lockett Recommendation
        tt := 0.0025; # Tray thickness
        Ad := 0.25; # Cross-sectional area of downcomer
        Ac := 1.5;  # Cross-sectional area of column
        phi := 0.1;  # Fraction of hole area to bubbling area
        k := 0.94;   # Assumes triangular pitch
        hw := 0.05; # Weir height

EQUATION
#       Overall Mass Balance
        $Nℓ = Lin − Lout + Vin − Vout;

#       Component Balance
        FOR  I:=1 TO NC−1 DO
        Nl * $x(I) = Lin*(xin(I)−x(I)) + Vin*(yin(I)−x(I))+ Vout*(x(I)−y(I));
        END

#       Energy Balance
        U + 0.1*P*Nl*voll = Nl*hl;

        $U = Lin*hlin − Lout*hl + Vin*hvin − Vout*hv;

#       Equilibrium
        LOG(y) = logK + LOG(x);

#
#       Hydrodynamics: All correlations from Lockett M. J., Distillation tray
#       fundamentals, Cambridge University Press, 1986. Reported original
#       references for completeness
#

#
#       Calculate weir length
#

        2*Ad/Ac * PI = theta − SIN(theta);
        W = (4*Ac/PI)^0.5*sin(theta/2);

#
#       Vapor flow onto plate
#

#
#       Residual pressure drop:
#       Van Winkle M., Distillation, McGraw-Hill, 1967
#       Fair J. R., (In Smith B. D.) Design of Equilibrium Stage Processes,
#       Chp 15, McGraw-Hill, 1963.
#
        DeltaPr = 4E−8*sigl/dh;

#
#       Dry plate pressure drop:
#       Cervenka J. and Kolar V. Hyrdodynamics of plate columns VIII,
#       Czech. Cem. Comm. 38, pp 2891, 1973.
#

        phi*Ab = Ah;
        Ab = Ac−2*Ad;
        uh = Vin * volvin/(3600*Ah);
```

```
          psi = k*(1−phi^2)/(phi*tt/dh)^0.2;
          DeltaPdt = 1E−5*psi*rhovin*uh^2/2;

#
#         Clear liquid pressure drop from mass balance
#

          DeltaPcl = 1E−5*Nl*mwl*g/Ab;

#
#         Pressure drop across plate (defines vapor flowrate)
#

          Pvin − P = DeltaPdt + DeltaPcl + DeltaPr;

#
#         Liquid flowrate off plate
#

#
#         Clear liquid pressure drop:
#         Colwell C. J., Clear liquid height and froth density on sieve trays,
#         Ind. Eng. Chem. Proc. Des. Dev., 20(2), pp 298, 1979.
#

          DeltaPcl=1E−5*rhol*g*hcl;
          us = Vin * volvin/(3600*Ab);

          eps = 12.6*(1−eps)*FrP^0.4*phi^(−0.25);
          FrP*(rhol−rhovin) = Fr*rhovin;
          Fr * hcl= us^2/g;
          hcl = (1−eps)*(hw+0.7301*(Lout*voll/(3600*W*Cd*(1−eps)))^0.67);
          how *(1−eps)= hcl−hw*(1−eps);

          IF how/hw > 8.14 THEN
                    Cd*how^1.5 = 1.06*(how+hw)^1.5;
          ELSE
                    Cd*hw = 0.61*hw+0.08*how;
          END

END

MODEL FeedTray INHERITS PhysicalProperties
#======================================
#
#         Model of distillation tray and downcomer:
#
#                   Equilibrium stage model
#                   Full hyrdrodynamics
#                   Negligible liquid and vapor entrainment
#                   Downcomer is sealed
#
#                   Assumes that dV/dt term is negligible in energy balance
#                   on tray
```

```
#
#         Lin              Tray p
#      |     |    |
#      |     V    |
#      |          |
#      |______|—————————————- Pp-1
#      |          |
#      |          |        Vout      <- F
#      |          |        ˆ
#      |          |______|__________ ———- Pp
#      |          |                       \
#      |LDout ->                  |     |     |
#      |_____________________| | |       |
#                           ˆ       | | |       | — Pp+1
#                           |           V
#                          Vin       Lout
#
#      Parameter       Description                             Units
#      ———             ————                                    —–
#
#      g               Gravitational constant                  (m/sˆ2)
#      PI
#      dh              Diameter of sieve holes                 (m)
#      tt              Tray thickness                          (m)
#      Ad              Cross-sectional area of downcomer       (mˆ2)
#      Ac              Cross-sectional area of column          (m)
#      phi             Fraction of hole to bubbling area       (mˆ2/mˆ2)
#      k               Dry plate pressure drop coeff.
#      hw              Weir height                             (m)
#      hd              Clearance under downcomer               (m)
#      Cdd             Discharge coefficient for downcomer
#
#      Variable        Description                             Units
#      ——–             ————                                    —–
#
#      F               Feed flowrate onto plate                (kmol/hr)
#      z               Feed composition                        (kmol/kmol)
#      Tf              Feed temperature                        (K)
#      Pf              Feed pressure                           (bar)
#      Hf              Specific enthalpy of feed               (MJ/kmol)
#      vollf           Specific volume of feed                 (mˆ3/kmol)
#      rholf           Density of feed                         (kg/mˆ3)
#      Lin             Liquid flowrate into downcomer          (kmol/hr)
#      xin             Liquid inlet mole composition           (kmol/kmol)
#      Tlin            Inlet liquid temperature                (K)
#      Plin            Pressure on plate p-1                   (bar)
#      hlin            Specific enthalpy of inlet liquid       (MJ/kmol)
#      vollin          Specific volume of inlet liquid         (mˆ3/kmol)
#      rholin          Density of inlet liquid                 (kg/mˆ3)
#      LDout           Liquid flowrate out of downcomer        (kmol/hr)
#      xD              Downcomer outlet composition            (kmol/kmol)
#      TD              Temperature of downcomer outlet         (K)
#      PD              Pressure at base of downcomer           (bar)
#      hlD             Specific enthalpy of downcomer outlet   (MJ/kmol)
#       vollD           Specific volume of downcomer outlet     (m^3/kmol)
#       rholD           Density of downcomer outlet             (kg/m^3)
#       Vin             Vapor flowrate onto plate               (kmol/hr)
#       yin             Inlet vapor composition                 (kmol/kmol)
#       Tvin            Inlet vapor temperature                 (K)
#       Pvin            Inlet vapor pressure                    (bar)
#       hvin            Inlet vapor specific enthalpy           (MJ/kmol)
#       volvin          Inlet vapor specific volume             (m^3/kmol)
#       rhovin          Inlet vapor density                     (kg/m^3)
#       Lout            Liquid outlet flowrate                  (kmol/hr)
#       Vout            Vapor flowrate off plate                (kmol/hr)
#       Nl              Liquid mole holdup on plate             (kmol)
#       U               Internal energy of liquid on plate      (MJ)
#       DeltaPr         Pressure drop due to surface tension    (bar)
#       DeltaPdt        Pressure drop due to dry plate          (bar)
#       DeltaPcl        Pressure drop due to clear liquid       (bar)
#       DeltaPcli       Pressure drop due to clear liquid
#                       at entrance to plate                    (bar)
#       DeltaPudc       Pressure drop due to flow out of
#                       downcomer                               (bar)
#       Ab              Bubbling area                           (m^2)
#       Ah              Area covered by sieve holes             (m^2)
#       uh              Super. vel. based on hole area          (m/s)
#       us              Super. vel. based on bubbling area      (m/s)
#       psi             Discharge coefficient for plate
#       Fr              Froude number                           (-)
#       FrP             Froude number                           (-)
#       eps             Aeration factor                         (-)
#       Cd              Discharge coefficient over weir         (-)
#       how             Height of liquid over weir              (m)
#       hcl             Height of clear liquid                  (m)
#       theta           Angle subtended by downcomer            (rads)
#       W               Length of weir                          (m)
#
#       Modifications:
#======================================

PARAMETER
        g       AS REAL
        PI      AS REAL

        dh      AS REAL
        tt      AS REAL
        Ad      AS REAL
        Ac      AS REAL
        phi     AS REAL
        k       AS REAL
        hw      AS REAL
        hd      AS REAL
        Cdd     AS REAL

UNIT
        Downcomer AS Downcomer
```

```
VARIABLE
#       Feed
        F       AS MoleFlow
        z       AS ARRAY(NC) OF MoleComposition
        Tf      AS Temperature
        Pf      AS Pressure
        Hf      AS Energy
        vollf   AS SpecificVolume
        rholf   AS Density

#       Liquid in
        Lin     AS MoleFlow
        xin     AS ARRAY(NC) OF MoleComposition
        Tlin    AS Temperature
        Plin    AS Pressure
        hlin    AS Energy
        vollin  AS SpecificVolume
        rholin  AS Density

#       Liqiud out of downcomer
        LDout   AS MoleFlow
        xD      AS ARRAY(NC) OF MoleComposition
        TD      AS Temperature
        PD      AS Pressure
        hlD     AS Energy
        vollD   AS SpecificVolume
        rholD   AS Density

#       Vapor in
        Vin     AS MoleFlow
        yin     AS ARRAY(NC) OF MoleComposition
        Tvin    AS Temperature
        Pvin    AS Pressure
        hvin    AS Energy
        volvin  AS SpecificVolume
        rhovin  AS Density

#       Liquid flowrate out
        Lout            AS MoleFlow

#       Vapor flowrate out
        Vout            AS MoleFlow

#       Tray holdup
        Nl              AS MoleHoldup
        U               AS EnergyHoldup

#       Hydrodynamics
        DeltaPr,
        DeltaPdt,
        DeltaPcl,
        DeltaPcli,
        DeltaPudc       AS Pressure
        Ab, Ah          AS Area
        uh, us          AS Velocity
        psi, Fr, FrP,
        eps, Cd         AS NoType
        how, hcl        AS Length

#       Tray geometry
        theta           AS NoType
        W               AS Length

STREAM
#       Feed
  Feed: F, Z, Tf, Pf, Hf, vollf, rholf                AS    Process_stream


#       Downcomer
  Liqin:   Lin, xin, Tlin, Plin, Hlin, vollin, rholin AS    Process_stream
  LiqDout: LDout, xD, TD, PD, hlD, vollD, rholD       AS    Process_stream

#       Tray
  Liqout:  Lout, x, T, P, hl, voll, rhol              AS    Process_stream
  Vapin:   Vin, yin, Tvin, Pvin, hvin, volvin, rhovin AS    Process_Stream
  Vapout:  Vout, Y, T, P, hv, volv, rhov              AS    Process_Stream

SET
        dh := 0.005; # Lockett Recommendation
        tt := 0.0025; # Tray thickness
        Ad := 0.25; # Cross-sectional area of downcomer
        Ac := 1.5;  # Cross-sectional area of column
        phi := 0.1; # Fraction of hole area to bubbling area
        k := 0.94;   # Assumes triangular pitch
        hw := 0.05; # Weir height
        hd := 0.025; # Clearance under downcomer
        Cdd := 0.56; # Discharge coefficient for downcomer (Koch)

EQUATION
#       Overall Mass Balance
        $Nℓ = F + LDout − Lout + Vin − Vout;

#       Component Balance
        FOR  I:=1 TO NC−1 DO
        Nl * $x(I) = F*(z(I)−x(I)) + LDout*(xD(I)−x(I)) + Vin*(yin(I)
        −x(I))+ Vout*(x(I)−y(I));
        END

#       Energy Balance
        U + 0.1*P*Nl*voll = Nl*hl;

        $U = F*hf + LDout*hlD − Lout*hl + Vin*hvin − Vout*hv;

#       Equilibrium
        LOG(y) = logK + LOG(x);

#       Connect Downcomer to tray
        Liqin = Downcomer.Liqin;
        LiqDout = Downcomer.Liqout;

#
#       Hydrodynamics: All correlations from Lockett M. J., Distillation tray
#       fundamentals, Cambridge University Press, 1986. Reported original
#       references for completeness
#

#
#       Calculate weir length
#

        2*Ad/Ac * PI = theta - SIN(theta);
        W = (4*Ac/PI)^0.5*sin(theta/2);

#
#       Vapor flow onto plate
#

#
#       Residual pressure drop:
#       Van Winkle M., Distillation, McGraw-Hill, 1967
#       Fair J. R., (In Smith B. D.) Design of Equilibrium Stage Processes,
#       Chp 15, McGraw-Hill, 1963.
#
        DeltaPr = 4E-8*sigl/dh;

#
#       Dry plate pressure drop:
#       Cervenka J. and Kolar V. Hyrdodynamics of plate columns VIII,
#       Czech. Cem. Comm. 38, pp 2891, 1973.
#

        phi*Ab = Ah;
        Ab = Ac-2*Ad;
        uh = Vin * volvin/(3600*Ah);
        psi = k*(1-phi^2)/(phi*tt/dh)^0.2;
        DeltaPdt = 1E-5*psi*rhovin*uh^2/2;

#
#       Clear liquid pressure drop from mass balance
#

        DeltaPcl = 1E-5*Nl*mwl*g/Ab;

#
#       Pressure drop across plate (defines vapor flowrate)
#

        Pvin - P = DeltaPdt + DeltaPcl + DeltaPr;

#
#       Liquid flowrate off plate
#
```

```
#
#        Clear liquid pressure drop:
#        Colwell C. J., Clear liquid height and froth density on sieve trays,
#        Ind. Eng. Chem. Proc. Des. Dev., 20(2), pp 298, 1979.
#

         DeltaPcl=1E−5*rhol*g*hcl;
         us = Vin * volvin/(3600*Ab);

         eps = 12.6*(1−eps)*FrP^0.4*phi^(−0.25);
         FrP*(rhol−rhovin) = Fr*rhovin;
         Fr * hcl= us^2/g;
         hcl = (1−eps)*(hw+0.7301*(Lout*voll/(3600*W*Cd*(1−eps)))^0.67);
         how *(1−eps)= hcl−hw*(1−eps);

         IF how/hw > 8.14 THEN
                  Cd*how^1.5 = 1.06*(how+hw)^1.5;
         ELSE
                  Cd*hw = 0.61*hw+0.08*how;
         END

#
#        Liquid flowrate onto plate from downcomer: Momentum balance
#

         DeltaPcli = 1E−5*rhold*g*(2./g*(LDout*vollD/(3600*W))^2*(1./hcl−1./hd) +
                           2./3*hcl^2/(1−eps))^0.5;


         DeltaPudc = 1E−5*rhold/2*(LDout*vollD/(3600*W*hd*Cdd))^2;

         Pd − P = DeltaPcli + DeltaPudc;

END

MODEL ValveLiquid
#=======================================
#
#        Algebraic model of a valve
#
#
#        Date: 26th June 2000
#
#        Model Assumptions:   Linear model of non-flashing liquid valve
#                             No enthalpy balance
#
#        Parameter       Description                         Units
#        ———             ————                                ——
#        NC              Number of components
#        Cv              Valve constant                      m^-2
#        Tau_p           Valve time constant
#
#        Variable        Description                         Units
#         ———-          ————-                         —-
#         L             Liquid flowrate               kmol/hr
#         X             Liquid composition            kmol/kmol
#         T             Temperature                   K
#         Plin          Inlet pressure                bar
#         hl            Enthalpy of liquid            kJ/kmol
#         voll          Specific volume of liquid     m^3/kmol
#         rhol          Density of liquid             kg/m^3
#         Plout         Pressure at outlet            bar
#         I_in          Control signal
#         P_drop        Pressure drop across valve    bar
#         Stem_pos      Valve stem position
#
#         Modifications:
#         Included valve dynamics
#========================================
PARAMETER
          NC            AS     INTEGER
          Cv,
          Tau_p         AS     REAL

VARIABLE
#         Input:
          L             AS MoleFlow
          x             AS ARRAY(NC) OF MoleComposition
          T             AS Temperature
          Plin          AS Pressure
          Hl            AS Energy
          voll          AS SpecificVolume
          rhol          AS Density

#         Output
          Plout         AS Pressure

#         Connection
          I_in          AS     NoType

#         Internal
          P_drop        AS     Pressure
          Stem_pos      AS     Percent

STREAM

  Liqin:   L, x, T, Plin, Hl, voll, rhol       AS     Process_stream
  Liqout:  L, x, T, Plout, Hl, voll, rhol      AS     Process_stream

#         Connections required for the controllers.

  Manipulated : I_in                           AS     CONNECTION

SET
          Cv            :=     1;
          Tau_p         :=     0.006;
```

```
EQUATION
#       Pressure relationship

        Plout   =       Plin − P_drop;

#       Valve dynamics
        Tau_p * $Stem_pos + Stem_pos = I_in;

#       Flow equation for non-flashing liquids
        L * voll = Cv * Stem_pos *
                        SIGN(P_drop)* SQRT(ABSOLUTE(P_drop)/(rhol/1000));

END

MODEL Reboiler INHERITS Flash
#========================================
#
#       Model of reboiler.  Inherits model of flash
#
#
#       Date: 26th June 2000
#
#       Model Assumptions:   Simple model of steam line. Instantaneous
#                            heat transfer
#
#       Parameter       Description                             Units
#       ———             ————                                    ——
#       Tau_p           Time constant for valve
#       VPAW, VPBW, .. Vapor pressure constants for water
#       PCW             Critical pressure of water              (bar)
#       TCW             Critical temperature of water           (K)
#       UA              Heat transfer coefficient for reboiler  (MJ/hr K)
#
#       Variable        Description                             Units
#       ———             ————                                    ——
#       TS              Temperature of steam                    (K)
#       TRS             Reduced temperature of steam
#       P_Reboiler_in   Pressure of steam at reboiler inlet     (bar)
#       I_in            Control signal to valve
#       I_in_c          Clipped control signal
#
#       Modifications:
#       Included valve dynamics
#========================================
PARAMETER
        Tau_p           AS      REAL
        VPAW, VPBW,
        VPCW, VPDW   AS REAL
        PCW, TCW        AS      REAL
        UA              AS      REAL

VARIABLE
        TS              AS      Temperature
        TRS             AS      NoType
        P_Boiler        AS    Pressure
        P_Reboiler_in   AS    Pressure
        P_Reboiler_out  AS    Pressure
        I_in            AS    NoType
        Stem_Pos        AS    NoType
SET
        Vtot   := 2;
        AT     := 1;
        UA := 129.294433;

#       Water Vapor Pressure Coefficients from Reid Prausnitz and Poling

        PCW := 221.2;
        TCW := 647.3;
        VPAW := −7.76451;
        VPBW := 1.45838;
        VPCW := −2.77580;
        VPDW := −1.23303;
        Tau_p := 0.006;

EQUATION

#       Model of steam line

        TRS*TCW = TS;
        (LOG(P_Reboiler_in) − LOG(PCW))*TRS = (VPAW*(1−TRS)+VPBW*(1−TRS)^1.5
                                +VPCW*(1−TRS)^3 + VPDW*(1−TRS)^6);
#       Valve dynamics
        Tau_p*$Stem_pos + Stem_pos = I_in;

#       Heat transfer

        Stem_pos*SQRT(ABSOLUTE((P_Boiler−P_Reboiler_in)*P_Boiler))
          = 150*SQRT(ABSOLUTE((P_Reboiler_in−P_Reboiler_out)*P_Reboiler_out));

        Qh = UA*(TS − T);

END

MODEL Condenser INHERITS Flash
#=========================================
#
#       Model of condenser. Inherits model of flash
#
#
#       Date: 26th June 2000
#
#       Model Assumptions:  Includes additional equation to calculate
#                           inlet vapor flow
#
#       Parameter       Description                         Units
#       ———             ————                                ——
#       K_Valve         Valve constant for inlet vapor flow
#       Tau_p           Time constant for valve
#       CPW           Specific heat capacity of water       (MJ/kg K)
#       M             Mass of water in condenser            (kg)
#       UA            Heat transfer coefficient             (MJ/hr K)
#
#       Variable      Description                           Units
#       ————          —————                                 ——
#       D             Distillate flowrate                   (kmol/hr)
#       LT            Reflux flowrate                       (kmol/hr)
#       I_in          Control signal
#       Stem_pos      Stem position
#       T_Water_in    Temperature of inlet water            (K)
#       T_Water_out   Temperature of outlet water           (K)
#
#       Modifications:
#       Included valve dynamics
#========================================
PARAMETER
        K_Valve AS REAL
        Tau_p   AS REAL
        CPW     AS REAL
        M       AS REAL
        UA      AS REAL
        Height  AS REAL

VARIABLE
        D       AS MoleFlow
        LT      AS MoleFlow
        Plout   AS Pressure

#       Cooling Water
        I_in          AS NoType
        Stem_pos      AS Percent
        T_Water_in    AS Temperature
        T_Water_out   AS Temperature

STREAM
  Reflux:       LT, x, T, Plout, hl, voll, rhol   AS Process_Stream
  Distillate:   D, x, T, Plout, hl, voll, rhol    AS Process_Stream

SET
        Vtot    := 2;
        AT      := 1;
        K_Valve := 2454;
        Tau_p   := 0.006;
        CPW     := 0.0042;
        M       := 200;
        UA      := 121.81;
        Height  := 0.63;

EQUATION
#       Vapor Flowrate
        F = K_Valve*(Pin − P)/(1E−4+SQRT(ABSOLUTE(Pin −P)));

#       Reflux splitter
```

```
        L = D+LT;

#       Total Condenser
        V = 0;

#       Pressure drop due to static head between accumulator and return
        Plout = Pout+1E-5*Height*g*rhol;

#       Calculate cooling from the cooling water flow

        3600*M*CPW*$T_Water_Out =
        4.9911272727*Stem_pos*(T_Water_In - T_Water_out) - Qh;

        Qh = UA*(T_Water_Out - T);

#       Cooling water valve dynamics

        Tau_p* $Stem_pos + Stem_pos = I_in;

END

MODEL PI_Cont
#========================================
#
#       Model of PI Controller
#
#
#       Date: 26th June 2000
#
#       Model Assumptions:
#
#       Variable        Description                             Units
#       --------        -----------                             -----
#       I_in            Control signal
#       SP              Controller setpoint
#       I_out           Actuator signal
#       Bias            Controller bias
#       Error           Difference between control signal and SP
#       Gain            Controller gain
#       I_error         Integral error
#       C_reset         Integral time
#       Value           Unclipped actuator signal
#       I_max           Maximum actuator signal
#       I_min           Minimum actuator signal
#
#       Modifications:
#       Included valve dynamics
#========================================

VARIABLE

#       Connections

        I_in    AS      Control_Signal
        SP      AS      Control_Signal
        I_out   AS      Control_Signal

#       Internal

        Bias            AS      Notype
        Error           AS      Notype
        Gain            AS      Notype
        I_error         AS      Notype
        C_reset         AS      Notype
        Value           AS      NoType
        I_max           AS      NoType
        I_min           AS      NoType

EQUATION

        Error = SP − I_in;
        $I_error = Error;
        Value = Bias + Gain * (Error + I_error / C_reset );

#       Ensure signal is clipped.  Pick sensible values of I_max and
#       I_min for the valves

        IF Value > I_max      THEN

                I_out = I_max;

        ELSE IF Value < I_min THEN

                I_out = I_min;

        ELSE

                I_out = Value;

        END
        END

END

MODEL LiqFeed INHERITS LiquidProperties
#=======================================
#       Model to set feed condition to column
#=======================================
VARIABLE
        F       AS MoleFlow

STREAM
        Feed:   F, x, T, P, hl, voll, rhol AS Process_stream

END

MODEL Column
#=======================================
#
#       Model of distillation column
#
#       Parameter       Description                             Units
#       ———             ————                                    ——
#       NC              Number of components in mixture
#       NT              Number of stages + reboiler + condenser
#       NF              Location of feed tray
#       g               Gravitational constant                  (m/s^2)
#       PI
#
#       Variable        Description                             Units
#       ———             ————                                    ——
#
#       Xdist           Scaled distillate composition
#       Xbot            Scaled bottoms composition
#       DistillatePurity        Distillate purity
#       BottomsPurity           Bottoms purity
#
#       Modifications:
#========================================

PARAMETER
        NC      AS INTEGER
        NT      AS INTEGER # Number of stages + reboiler + condenser
        NF      AS INTEGER # Location of feed tray
        g       AS REAL
        PI      AS REAL
UNIT
        Liquid          AS LiqFeed
        Rectifier       AS ARRAY(NF−2) OF Tray
        TopTray         AS TopTray
        FeedTray        AS FeedTray
        Stripper        AS ARRAY(NT−NF−2) OF Tray
        Reboiler        AS Reboiler
        RefluxValve,
        DistillateValve,
        BottomsValve    AS ValveLiquid
        Condenser       AS Condenser

VARIABLE
        XDist,
        XBot                    AS NoType
        DistillatePurity,
        BottomsPurity           AS Percent
SET
        NT := 30;
        NF := 15;

EQUATION
        Liquid.Feed = FeedTray.Feed;

        TopTray.LiqOut = Rectifier(1).LiqIn;
        TopTray.VapIn = Rectifier(1).VapOut;
        Rectifier(1:NF−3).LiqOut = Rectifier(2:NF−2).LiqIn;
        Rectifier(1:NF−3).VapIn = Rectifier(2:NF−2).VapOut;

#       Connections to feed tray

        Rectifier(NF−2).VapIn = FeedTray.VapOut;
        Rectifier(NF−2).LiqOut = FeedTray.LiqIn;
        Stripper(1).LiqIn = FeedTray.LiqOut;
        Stripper(1).VapOut = FeedTray.VapIn;

        Stripper(1:NT−NF−3).Liqout = Stripper(2:NT−NF−2).Liqin;
        Stripper(1:NT−NF−3).Vapin = Stripper(2:NT−NF−2).Vapout;

#       Connections to reboiler

        Reboiler.Feed = Stripper(NT−NF−2).Liqout;
        Reboiler.Vapor = Stripper(NT−NF−2).Vapin;
        Reboiler.Liquid = BottomsValve.LiqIn;

#       Connections to condenser

        TopTray.VapOut = Condenser.Feed;
        Condenser.Reflux = RefluxValve.LiqIn;
        Condenser.Distillate = DistillateValve.LiqIn;
        RefluxValve.LiqOut = TopTray.LiqIn;

        TopTray.Plin = TopTray.P;

#       Calculate scaled composition and product purity

        Xdist = LOG(TopTray.X(2)/TopTray.X(1));
        Xbot = LOG(Stripper(13).X(2)/Stripper(13).X(1));
        DistillatePurity = 100*DistillateValve.X(2);
        BottomsPurity = 100*BottomsValve.X(1);


END

#=========================================
SIMULATION Distillation
OPTIONS
ALGPRINTLEVEL:=0;
ALGRTOLERANCE:=1E−7;
ALGATOLERANCE:=1E−7;
ALGMAXITERATIONS:=500;
DYNPRINTLEVEL:=0;
DYNRTOLERANCE:=1E−7;
DYNATOLERANCE:=1E−7;

UNIT
        Plant   AS      Column

SET
```

```
    WITHIN Plant DO
        NC :=2;
        g := 9.81;
        PI := 3.141592654;
    END

INPUT
    WITHIN Plant.Liquid DO
        F := 100;
        x(1) := 0.25;
        T := 353;
        P := 1.17077;
    END

    WITHIN Plant.BottomsValve DO
        Plout := 1.435;
        I_In := 42.41;
    END

    WITHIN Plant.DistillateValve DO
        Plout := 1.31;
        I_in := 54.43;
    END

    WITHIN Plant.RefluxValve DO
        I_in := 27.197;
    END

    WITHIN Plant.Condenser DO
        T_Water_in := 291;
        I_in := 50;
    END

    WITHIN Plant.Reboiler DO
        I_in := 61.8267;
        P_Boiler := 10;
        P_Reboiler_out := 1;
    END

PRESET INCLUDE DistPresets
INITIAL
STEADY_STATE

SCHEDULE
    SEQUENCE
    CONTINUE FOR 10
    RESET
    Plant.Condenser.I_in:=60;
    END
    CONTINUE FOR 10
    END
END
```

## B.5 State Bounds for Reaction Kinetics

```
#===============================
#
# Simulation of state bounds
# for kinetics A->B->C.
# D. M. Collins 08/22/03.
#
#===============================
DECLARE
  TYPE
  NoType = 0 :−1E5 :1E5  UNIT = "−"
END

MODEL Series
  PARAMETER
    p  AS ARRAY(2) OF REAL
    pL AS ARRAY(2) OF REAL
    pU AS ARRAY(2) OF REAL
  VARIABLE
    XL AS ARRAY(3) OF NoType
    XU AS ARRAY(3) OF NoType
    X  AS ARRAY(3) OF NoType
  EQUATION
# Original ODE
    $X(1)=−p(1)*X(1);
    $X(2)=p(1)*X(1)−p(2)*X(2);
    $X(3)=p(2)*X(2);
# Bounding system
    $XL(1)=−MAX(PL(1)*XL(1),PU(1)*XL(1));
    $XL(2)=MIN(PL(1)*XL(1),PU(1)*XL(1),PL(1)*XU(1),PU(1)*XU(1))
    −MAX(PL(2)*XL(2),PU(2)*XL(2));
    $XL(3)=MIN(PL(2)*XL(2),PU(2)*XL(2),PL(2)*XU(2),PU(2)*XU(2));
    $XU(1)=−MIN(PL(1)*XU(1),PU(1)*XU(1));
    $XU(2)=MAX(PL(1)*XL(1),PU(1)*XL(1),PL(1)*XU(1),PU(1)*XU(1))
    −MIN(PL(2)*XU(2),PU(2)*XU(2));
    $XU(3)=MAX(PL(2)*XL(2),PU(2)*XL(2),PL(2)*XU(2),PU(2)*XU(2));
END

SIMULATION BoundReactions
  OPTIONS
  CSVOUTPUT:=TRUE;
  UNIT SeriesSimulation AS Series
  SET
    WITHIN SeriesSimulation DO
      pL(1):=1;
      pU(1):=2;
      pL(2):=1;
      pU(2):=2;
      p(1):=1.5;
      p(2):=1.5;
    END
  INITIAL
```

```
    WITHIN SeriesSimulation DO
      XL(1)=10;
      XU(1)=10;
      X(1)=10;
      XL(2)=0;
      XU(2)=0;
      X(2)=0;
      XL(3)=0;
      XU(3)=0;
      X(3)=0;
    END

  SCHEDULE
    CONTINUE FOR 10
END
```

## B.6 Convex Underestimates and Concave Overestimates of States

```
# This file automatically generated by ./oa.exe on Thu Aug 21 11:31:07 2003


DECLARE
  TYPE
    STATE = 0.0 : −1E9 : 1E9
END #declare

MODEL OAmodel

  PARAMETER
    p AS ARRAY(2) OF REAL
    p_L AS ARRAY(2) OF REAL
    p_U AS ARRAY(2) OF REAL
    p_ref AS ARRAY(2) OF REAL

  VARIABLE
    x AS ARRAY(3) OF STATE
    x_ref AS ARRAY(3) OF STATE
    x_L AS ARRAY(3) OF STATE
    x_U AS ARRAY(3) OF STATE
    c AS ARRAY(3) OF STATE
    CC AS ARRAY(3) OF STATE
    ifVar AS ARRAY(16) OF STATE

  EQUATION
# original equation(s)
$x(1)=−p(1)*x(1);

$x(2)=p(1)*x(1)−p(2)*x(2);

$x(3)=p(2)*x(2);


# original equation(s) lower bound(s)
$x_L(1)=−max(p_L(1)*x_L(1), p_U(1)*x_L(1));

$x_L(2)=min(p_L(1)*x_L(1), p_L(1)*x_U(1), p_U(1)*x_L(1), p_U(1)*x_U(1))
−max(p_L(2)*x_L(2), p_U(2)*x_L(2));

$x_L(3)=min(p_L(2)*x_L(2), p_L(2)*x_U(2), p_U(2)*x_L(2), p_U(2)*x_U(2));


# original equation(s) upper bound(s)
$x_U(1)=−min(p_L(1)*x_U(1), p_U(1)*x_U(1));

$x_U(2)=max(p_L(1)*x_L(1), p_L(1)*x_U(1), p_U(1)*x_L(1), p_U(1)*x_U(1))
−min(p_L(2)*x_U(2), p_U(2)*x_U(2));
```

```
$x_U(3)=max(p_L(2)*x_L(2), p_L(2)*x_U(2), p_U(2)*x_L(2), p_U(2)*x_U(2));


# convex OA term(s)
$c(1)=−min(x_L(1)*p_ref(1)+p_U(1)*x_ref(1)−x_L(1)*p_U(1),
p_L(1)*x_ref(1)+x_U(1)*p_ref(1)−p_L(1)*x_U(1))
+(−ifVar(1))*(c(1)−x_ref(1))+(−ifVar(2))*(p(1)−p_ref(1));

$c(2)=max(x_U(1)*p_ref(1)+p_U(1)*x_ref(1)−p_U(1)*x_U(1),
x_L(1)*p_ref(1)+p_L(1)*x_ref(1)−p_L(1)*x_L(1))
−min(x_L(2)*p_ref(2)+p_U(2)*x_ref(2)−x_L(2)*p_U(2),
p_L(2)*x_ref(2)+x_U(2)*p_ref(2)−p_L(2)*x_U(2))+
min(ifVar(5)*c(1), ifVar(5)*CC(1))−ifVar(5)*x_ref(1)
+(−ifVar(6))*(c(2)−x_ref(2))+ifVar(7)*(p(1)−p_ref(1))
+(−ifVar(8))*(p(2)−p_ref(2));

$c(3)=max(x_U(2)*p_ref(2)+p_U(2)*x_ref(2)−p_U(2)*x_U(2),
x_L(2)*p_ref(2)+p_L(2)*x_ref(2)−p_L(2)*x_L(2))
+min(ifVar(13)*c(2), ifVar(13)*CC(2))
−ifVar(13)*x_ref(2)+ifVar(14)*(p(2)−p_ref(2));


# concave OA term(s):
$CC(1)=−max(x_U(1)*p_ref(1)+p_U(1)*x_ref(1)−p_U(1)*x_U(1),
x_L(1)*p_ref(1)+p_L(1)*x_ref(1)−p_L(1)*x_L(1))
+(−ifVar(3))*(CC(1)−x_ref(1))+(−ifVar(4))*(p(1)−p_ref(1));

$CC(2)=min(x_L(1)*p_ref(1)+p_U(1)*x_ref(1)−x_L(1)*p_U(1),
p_L(1)*x_ref(1)+x_U(1)*p_ref(1)−p_L(1)*x_U(1))
−max(x_U(2)*p_ref(2)+p_U(2)*x_ref(2)−p_U(2)*x_U(2),
x_L(2)*p_ref(2)+p_L(2)*x_ref(2)−p_L(2)*x_L(2))
+max(ifVar(9)*c(1), ifVar(9)*CC(1))−ifVar(9)*x_ref(1)
+(−ifVar(10))*(CC(2)−x_ref(2))+ifVar(11)*(p(1)−p_ref(1))
+(−ifVar(12))*(p(2)−p_ref(2));

$CC(3)=min(x_L(2)*p_ref(2)+p_U(2)*x_ref(2)−x_L(2)*p_U(2),
p_L(2)*x_ref(2)+x_U(2)*p_ref(2)−p_L(2)*x_U(2))
+max(ifVar(15)*c(2), ifVar(15)*CC(2))−ifVar(15)*x_ref(2)
+ifVar(16)*(p(2)−p_ref(2));


# define the if variable(s):
IF x_L(1)*p_ref(1)+p_U(1)*x_ref(1)−x_L(1)*p_U(1)
< p_L(1)*x_ref(1)+x_U(1)*p_ref(1)−p_L(1)*x_U(1) THEN
  ifVar(1) = +p_U(1);
ELSE
  ifVar(1) = p_L(1);
END #if

IF x_L(1)*p_ref(1)+p_U(1)*x_ref(1)−x_L(1)*p_U(1)
< p_L(1)*x_ref(1)+x_U(1)*p_ref(1)−p_L(1)*x_U(1) THEN
  ifVar(2) = x_L(1);
ELSE
  ifVar(2) = +x_U(1);
```

```
END #if

IF x_U(1)*p_ref(1)+p_U(1)*x_ref(1)−p_U(1)*x_U(1)
> x_L(1)*p_ref(1)+p_L(1)*x_ref(1)−p_L(1)*x_L(1) THEN
  ifVar(3) = +p_U(1);
ELSE
  ifVar(3) = +p_L(1);
END #if

IF x_U(1)*p_ref(1)+p_U(1)*x_ref(1)−p_U(1)*x_U(1)
> x_L(1)*p_ref(1)+p_L(1)*x_ref(1)−p_L(1)*x_L(1) THEN
  ifVar(4) = x_U(1);
ELSE
  ifVar(4) = x_L(1);
END #if

IF x_U(1)*p_ref(1)+p_U(1)*x_ref(1)−p_U(1)*x_U(1)
> x_L(1)*p_ref(1)+p_L(1)*x_ref(1)−p_L(1)*x_L(1) THEN
  ifVar(5) = +p_U(1);
ELSE
  ifVar(5) = +p_L(1);
END #if

IF x_L(2)*p_ref(2)+p_U(2)*x_ref(2)−x_L(2)*p_U(2)
< p_L(2)*x_ref(2)+x_U(2)*p_ref(2)−p_L(2)*x_U(2) THEN
  ifVar(6) = +p_U(2);
ELSE
  ifVar(6) = p_L(2);
END #if

IF x_U(1)*p_ref(1)+p_U(1)*x_ref(1)−p_U(1)*x_U(1)
> x_L(1)*p_ref(1)+p_L(1)*x_ref(1)−p_L(1)*x_L(1) THEN
  ifVar(7) = x_U(1);
ELSE
  ifVar(7) = x_L(1);
END #if

IF x_L(2)*p_ref(2)+p_U(2)*x_ref(2)−x_L(2)*p_U(2)
< p_L(2)*x_ref(2)+x_U(2)*p_ref(2)−p_L(2)*x_U(2) THEN
  ifVar(8) = x_L(2);
ELSE
  ifVar(8) = +x_U(2);
END #if

IF x_L(1)*p_ref(1)+p_U(1)*x_ref(1)−x_L(1)*p_U(1)
< p_L(1)*x_ref(1)+x_U(1)*p_ref(1)−p_L(1)*x_U(1) THEN
  ifVar(9) = +p_U(1);
ELSE
  ifVar(9) = p_L(1);
END #if

IF x_U(2)*p_ref(2)+p_U(2)*x_ref(2)−p_U(2)*x_U(2)
> x_L(2)*p_ref(2)+p_L(2)*x_ref(2)−p_L(2)*x_L(2) THEN
  ifVar(10) = +p_U(2);
```

```
ELSE
  ifVar(10) = +p_L(2);
END #if

IF x_L(1)*p_ref(1)+p_U(1)*x_ref(1)−x_L(1)*p_U(1)
< p_L(1)*x_ref(1)+x_U(1)*p_ref(1)−p_L(1)*x_U(1) THEN
  ifVar(11) = x_L(1);
ELSE
  ifVar(11) = +x_U(1);
END #if

IF x_U(2)*p_ref(2)+p_U(2)*x_ref(2)−p_U(2)*x_U(2)
> x_L(2)*p_ref(2)+p_L(2)*x_ref(2)−p_L(2)*x_L(2) THEN
  ifVar(12) = x_U(2);
ELSE
  ifVar(12) = x_L(2);
END #if

IF x_U(2)*p_ref(2)+p_U(2)*x_ref(2)−p_U(2)*x_U(2)
> x_L(2)*p_ref(2)+p_L(2)*x_ref(2)−p_L(2)*x_L(2) THEN
  ifVar(13) = +p_U(2);
ELSE
  ifVar(13) = +p_L(2);
END #if

IF x_U(2)*p_ref(2)+p_U(2)*x_ref(2)−p_U(2)*x_U(2)
> x_L(2)*p_ref(2)+p_L(2)*x_ref(2)−p_L(2)*x_L(2) THEN
  ifVar(14) = x_U(2);
ELSE
  ifVar(14) = x_L(2);
END #if

IF x_L(2)*p_ref(2)+p_U(2)*x_ref(2)−x_L(2)*p_U(2)
< p_L(2)*x_ref(2)+x_U(2)*p_ref(2)−p_L(2)*x_U(2) THEN
  ifVar(15) = +p_U(2);
ELSE
  ifVar(15) = p_L(2);
END #if

IF x_L(2)*p_ref(2)+p_U(2)*x_ref(2)−x_L(2)*p_U(2)
< p_L(2)*x_ref(2)+x_U(2)*p_ref(2)−p_L(2)*x_U(2) THEN
  ifVar(16) = x_L(2);
ELSE
  ifVar(16) = +x_U(2);
END #if


# Enter the user defined x_ref equation(s)
  $x_ref(1)=−p_ref(1)*x_ref(1);

$x_ref(2)=p_ref(1)*x_ref(1)−p_ref(2)*x_ref(2);

$x_ref(3)=p_ref(2)*x_ref(2);
```

```
END #OAmodel


SIMULATION mySim

  UNIT OA AS OAmodel

  # Enter parameter values here
  SET
    WITHIN OA DO
      p(1) := 1.75;
      p(2) := 1.55;
      p_ref(1) := 1.3;
      p_ref(2) := 1.7;
      p_L(1) := 1;
      p_L(2) := 1;
      p_U(1) := 2;
      p_U(2) := 2;
    END # within

  # Enter initial conditions here
  INITIAL
    WITHIN OA DO
      x(1) = 10;
      x(2) = 0;
      x(3) = 0;
      x_L(1) = 10;
      x_L(2) = 0;
      x_L(3) = 0;
      x_U(1) = 10;
      x_U(2) = 0;
      x_U(3) = 0;
      c(1) = 10;
      c(2) = 0;
      c(3) = 0;
      CC(1) = 10;
      CC(2) = 0;
      CC(3) = 0;
      x_ref(1) = 10;
      x_ref(2) = 0;
      x_ref(3) = 0;
    END # within

  SCHEDULE
    SEQUENCE
     # Enter the simulation length
     CONTINUE FOR 10
    END # sequence

END # simulation
```

# Appendix C

# Fortran Code

## C.1 Generation of State-Space Occurrence Information

```
      SUBROUTINE GRAPH(NB,NY,NX,NSTATE, MINPUT, NEINPUT, NINDEX,
     $  NEINDEX, NESTATE, IRINPUT, JCINPUT, IPERM, IFLAGY,
     $  IBLOCK, IRPRM, JCPRM, ISTATE, JSTATE, IWORK,
     $  LOCIWORKOLD, LIWORK, LSTATE,IERROR,INFO)

      IMPLICIT NONE
      INTEGER NB,LIWORK,NY,NX,NSTATE, MINPUT, NEINPUT
      INTEGER NINDEX, NEINDEX, NESTATE, LSTATE
      INTEGER IRINPUT(NEINPUT), JCINPUT(NEINPUT)
      INTEGER IPERM(2*NINDEX)
      INTEGER IFLAGY(NY)
      INTEGER IBLOCK(NINDEX+1)
      INTEGER IRPRM(NEINDEX), JCPRM(NEINDEX)
      INTEGER ISTATE(LSTATE), JSTATE(LSTATE)
      INTEGER IWORK(LIWORK)

C====================================
C     INPUTS
C     ——
C     NB:     Number of blocks in block decomosition.
C     NX :    Number of states (XDOTs).
C     NY :    Number of algebraic variables.
C     LIWORK: Length of integer workspace.
C     LSTATE: Length of ISTATE, JSTATE and FSTATE. May need up to
C             NSTATE*MINPUT but may be a lot less.
C     NSTATE: Number of states+number of algebraic variables to be
C             included in the state-space model. (Some of the Y's
C             may be eliminated.)
C     MINPUT: Number of inputs+ number of states.
C     NINDEX: NX+NY.
C     NEINPUT: Number of entries in IRINPUT, JCINPUT
C     NEINDEX: Number of entries in IRPRM, JCPRM.
```

```
C       LIWORK: Length of integer array IWORK.
C       LSTATE: Length of integer arrays ISTATE, JSTATE and double array
C                FSTATE.
C       IRINPUT: An integer array of length NEINPUT which holds the row
C                indices of [f_x f_u]. The list must be column sorted.
C       JCINPUT: An integer array of length NEINPUT which holds the column
C                indices of [f_x f_u]. The list must be column sorted.
C       IFLAGY: An array of length NY which indicates whether an algebraic
C                variable, Y, is to be included in the state-space model.
C                IFLAGY(I)=0 indicates the I'th algebraic variable is to be
C                kept. IFLAGY(I)=-1 indicates the I'th algebraic variable is
C                to be eliminated.
C       IPERM:  An integer array of length 2*NINDEX. The first NINDEX
C                 entries hold the column permutations and the next NINDEX
C                 entries hold the row permutations
C       IBLOCK: Integer array I=1.NB+1 which points to the row which starts
C                 the I'th block.
C       IRPRM: Integer array of length NEINDEX which holds IRINDEX in
C                 block upper triangular form. It is not established until
C                 after the call to FACTOR.
C       JCPRM: Integer array of length NEINDEX which holds JCINDEX in
C                 block upper triangular form. It is not established until
C                 after the call to FACTOR.
C       IWORK: Integer workspace. See error checks for length.
C
C       OUTPUTS
C       ——-
C       NESTATE: Number of entries in ISTATE, JSTATE.
C       LSTATE: Length of ISTATE, JSTATE, FSTATE
C       ISTATE: An integer array of length LSTATE which holds the row
C                indices of the state-space model.
C       JSTATE: An integer array of length LSTATE which holds the column
C                indices of the state-space model.
C       IERROR: An integer holding the error return code.
C       INFO:   An integer holding additional information about an error
C                return.
C
C       ERROR RETURN CODES
C       ————————
C
C       IERROR: -1 Insufficient integer workspace. INFO=Required memory
C       IERROR: -11 Error return from GRAPH, insufficient memory to accumulate
C                  ISTATE, JSTATE.
C
C==================================
        INTEGER LOCISTBLOCK, LOCIPLIST
        INTEGER LOCICOLOUR, LOCIBLKNO, LOCICOLNO, LOCIWORK
        INTEGER LOCIRINPUTC, LOCJCINPUTC,LOCIWORKEND
        INTEGER I, LWRK

        INTEGER LOCIWORKOLD, IERROR,INFO

        LWRK=MAX(MINPUT+3*NEINPUT+1,NY+NINDEX,2*NEINDEX+NINDEX)
        LOCIRINPUTC=1
```

```
      LOCJCINPUTC=LOCIRINPUTC+NEINPUT
      LOCISTBLOCK=LOCJCINPUTC+NEINPUT
      LOCIPLIST=LOCISTBLOCK+NINDEX+1
      LOCICOLOUR=LOCIPLIST+NINDEX
      LOCIBLKNO=LOCICOLOUR+NINDEX
      LOCICOLNO=LOCIBLKNO+NINDEX
      LOCIWORK=LOCICOLNO+NINDEX
      LOCIWORKEND=LOCIWORK+LWRK

C
C     Check workspace requirements again now we have repartitioned
C
      IF ((LOCIWORKOLD+LOCIWORKEND).GT.LIWORK) THEN
            IERROR=-1
            INFO=LOCIWORKOLD+LOCIWORKEND
      RETURN
      ENDIF

      DO 100 I=1,NEINPUT
            IWORK(I)=IRINPUT(I)
            IWORK(I+NEINPUT)=JCINPUT(I)
 100  CONTINUE

      CALL GRAPH2(NB,NY,NX,NSTATE, MINPUT, NEINPUT,NINDEX,
     $ NEINDEX, NESTATE, LSTATE, LWRK,IWORK(LOCIRINPUTC),
     $ IWORK(LOCJCINPUTC),
     $ IPERM, IFLAGY,IBLOCK, IRPRM, JCPRM, ISTATE, JSTATE,
     $ IWORK(LOCISTBLOCK),
     $ IWORK(LOCIPLIST), IWORK(LOCICOLOUR),
     $ IWORK(LOCIBLKNO), IWORK(LOCICOLNO),
     $ IWORK(LOCIWORK), IERROR)

      RETURN
      END
C==================================

      SUBROUTINE GRAPH2(NB,NY,NX,NSTATE, MINPUT, NEINPUT,NINDEX,
     $ NEINDEX, IPSTATE, LSTATE, LWRK, IRINPUTC, JCINPUTC,
     $ IPERM, IFLAGY,
     $ IBLOCK, IRPRM, JCPRM,ISTATE, JSTATE,ISTBLOCK, IPLIST,
     $ ICOLOUR, IBLKNO, ICOLNO,IWORK, IERROR)

      IMPLICIT NONE
      INTEGER NX, NY

      INTEGER NSTATE, MINPUT, NEINPUT
      INTEGER NINDEX, NEINDEX, LSTATE, IERROR
      INTEGER NB, LWRK

      INTEGER IRINPUTC(NEINPUT), JCINPUTC(NEINPUT)
      INTEGER IPERM(2*NINDEX)
      INTEGER IFLAGY(NY)
      INTEGER IBLOCK(NINDEX+1),IWORK(LWRK)
      INTEGER ISTBLOCK(NINDEX+1)
      INTEGER ISTATE(LSTATE), JSTATE(LSTATE)
      INTEGER IRPRM(NEINDEX), JCPRM(NEINDEX)
      INTEGER IPLIST(NINDEX)

      INTEGER I, J, IFINDBK
      INTEGER IYDELETE,ISP, IGREY, IPSTATE

C
C     Variables for depth first search:
C     IBLKNO(ISP) contains block on stack.
C     ICOLNO(ISP) contains pointer to IRLIST.
C     ICOLOUR(I) is the colour of the i'th block.
C
      INTEGER ICOLOUR(NINDEX), IBLKNO(NINDEX), ICOLNO(NINDEX)

C====================================
C     Now we need to permute the rows of INPUT so that they correspond
C     with the block triangularized form of [f_xdot f_y]
C

      DO 100 I=1,NEINPUT
         IRINPUTC(I)=IPERM(IRINPUTC(I)+NINDEX)
 100  CONTINUE

C
C     Row and column sort the data
C
      CALL DECCOUNTSORT(NEINDEX,NINDEX,IRPRM,JCPRM,IWORK,
     $ IWORK(NEINDEX+1),IWORK(2*NEINDEX+1))

      CALL COUNTSORT(NEINDEX,NINDEX,IWORK(NEINDEX+1),IWORK,
     $ JCPRM, IRPRM,IWORK(2*NEINDEX+1))

C====================================
C     Assemble [f_x f_u] into row pointer form.
C     Entries are marked for every row associated with IBLOCK(I)
C     Establish pointer for start of column into JCINPUT. Last pointer must
C     be NEINPUT+1 i.e. the last row finishes at the end of the data.
C     For empty rows the pointer is not incremented. Duplicate entries in
C     JCINPUT for each block are removed.
C

C====================================
C     Construct ISTBLOCK(I)=row number I=1,NB a pointer into the new system
C     Some of the blocks may be empty!!
C     First we'll permute IFLAGY into IWORK.
C

      DO 800 I=1,NX
         IWORK(IPERM(I))=0
 800  CONTINUE

      DO 900 I=1,NY
         IWORK(IPERM(I+NX))=IFLAGY(I)
 900   CONTINUE

       ISTBLOCK(1)=1
       IYDELETE=0

       DO 1000 I=2,NB+1
               DO 1100 J=IBLOCK(I-1),IBLOCK(I)-1
                       IYDELETE=IYDELETE+IWORK(J)
 1100          CONTINUE
               ISTBLOCK(I)=IBLOCK(I)+IYDELETE
 1000  CONTINUE


C==================================
C      Need to initialize pointer IPLIST(I) = BLOCK NUMBER.
C
C

       J=1
       IPLIST(1)=1

       DO 1200 I=2, NINDEX
               IF (I.GE.IBLOCK(J+1)) THEN
                 J=J+1
               ENDIF
               IPLIST(I)=J
 1200  CONTINUE

C
C      Change IBLOCK so it points to a position in IRPRM rather than the
C      column number.
C

       J=1

       DO 1400 I=2,NB
 1500          IF (JCPRM(J).LT.IBLOCK(I)) THEN
                       J=J+1
                       GOTO 1500
               ENDIF
               IBLOCK(I)=J
 1400  CONTINUE

       IBLOCK(NB+1)=NEINDEX+1

C==================================
C      At this point we have all the pointers set and we can begin
C      the depth first search.  Remember to go up the matrix!!!
C      since we are block upper triangular.
C

C      Initialize block colours.

       DO 1600 I=1,NB
```

```
            ICOLOUR(I) = 0
 1600   CONTINUE

C
C       Initialize pointer to ISTATE, JSTATE
C
        IPSTATE=0


C==================================
C       Start depth first search
C

        DO 8000 I=1,NEINPUT
            IGREY=JCINPUTC(I)

C
C       Starting block for dfs
C
        IFINDBK = IPLIST(IRINPUTC(I))

        IF (ICOLOUR(IFINDBK).NE.IGREY) THEN
C
C       Put a single block on the stack.
C

C
C       We have to change the colours on each iteration of the depth
C       first search otherwise we will be n^2!!! with the reinitializations.
C       Hence ICOLOUR(I).NE.IGREY means unvisited on this round.
C
            ISP=1

C
C       Initialize stack inputs to zero
C

            IBLKNO(1)=IFINDBK
            ICOLNO(1)=IBLOCK(IFINDBK)+1

C
C       Mark block as grey
C
            ICOLOUR(IFINDBK)=IGREY

C
C       Do while stack is not empty!!
C

 9000   IF (ISP.NE.0) THEN

C       Check to see if there are any remaining blocks to be searched
C
```

```
          IFINDBK=IBLKNO(ISP)
          IF (ICOLNO(ISP).GE.IBLOCK(IFINDBK+1)) THEN
C
C      Pop a block from the stack.
C

C==================================
C
C      Occurence information is written as stack is popped.
C
C      Write occurence information to ISTATE in sparse format. Now we
C      need to be careful about the entries we are going to delete from y.
C


      DO 5000 J=ISTBLOCK(IFINDBK),ISTBLOCK(IFINDBK+1)−1
              IPSTATE=IPSTATE+1
              IF (IPSTATE.GT.LSTATE) THEN
              IERROR=−11
              RETURN
              ENDIF
              ISTATE(IPSTATE)=J
              JSTATE(IPSTATE)=JCINPUTC(I)

 5000 CONTINUE
C==================================
          ISP=ISP−1

          ELSE
            IFINDBK=IPLIST(IRPRM(ICOLNO(ISP)))

              IF (ICOLOUR(IFINDBK).NE.IGREY) THEN

C
C      Put connected blocks on the stack
C
                  ICOLNO(ISP)=ICOLNO(ISP)+1
                  ISP=ISP+1
                  ICOLNO(ISP)=IBLOCK(IFINDBK)+1
                  IBLKNO(ISP)=IFINDBK
                  ICOLOUR(IFINDBK)=IGREY

              ELSE

C
C      Skip over entry
C
                  ICOLNO(ISP)=ICOLNO(ISP)+1
              ENDIF

          ENDIF

      GOTO 9000
      ENDIF
      ENDIF
 8000 CONTINUE

C=====================================
C     Permute the graph back so that it is consistent with the
C     original inputs.
C
C     This bit is a little confusing as we have to remember all
C     those Y's we've deleted.
C
C
C
C

      IYDELETE=0

      DO 9500 I=1,NY
         IF (IFLAGY(I).EQ.0) THEN
            IYDELETE=IYDELETE+1
            IWORK(IYDELETE)=I
         ENDIF
 9500 CONTINUE

      DO 9600 I=1,NINDEX
         IWORK(NY+I)=0
 9600 CONTINUE

      DO 9700 I=1,NX
         IWORK(NY+IPERM(I))=I
 9700 CONTINUE

      DO 9800 I=1,IYDELETE
         IWORK(NY+IPERM(IWORK(I)+NX))=I+NX
 9800 CONTINUE

      IYDELETE=0
      DO 9900 I=1,NINDEX
         IF (IWORK(I+NY).NE.0) THEN
            IYDELETE=IYDELETE+1
            IWORK(IYDELETE+NY)=IWORK(I+NY)
         ENDIF
 9900 CONTINUE

C
C     Permute ISTATE according to IWORK.
C

      DO 9950 I=1,IPSTATE
         ISTATE(I)=IWORK(ISTATE(I)+NY)
 9950 CONTINUE


C
```

```
C===================================
      RETURN
      END
```

---

```
C     Fortran code to perform counting sort
C     assumes the data is in the form of pairs (i,j)
C     where the data is to be sorted on i.
C     By David M. Collins 09/12/00

      SUBROUTINE countsort(ne, n, irow, jcol,irowsort,jcolsort,irwork)
      IMPLICIT NONE
      INTEGER ne, n, i, j
      INTEGER irow(ne), jcol(ne), irowsort(ne), jcolsort(ne)
      INTEGER irwork(n)

C     Initialize workspace

      DO 10 i=1,n
         irwork(i) = 0
 10   CONTINUE

      DO 20 j=1,ne
         irwork(irow(j)) = irwork(irow(j))+1
 20   CONTINUE

C     irwork(i) now contains # elements equal to i

      DO 30 i=2,n
         irwork(i) = irwork(i)+irwork(i-1)
 30   CONTINUE

C     irwork(i) now contains # elements less than or equal to i

      DO 40 j=ne,1, -1
         irowsort(irwork(irow(j))) = irow(j)
         jcolsort(irwork(irow(j))) = jcol(j)
         irwork(irow(j)) = irwork(irow(j)) - 1
 40   CONTINUE

      RETURN
      END


      SUBROUTINE deccountsort(ne, n, irow, jcol,irowsort,jcolsort,
     $  irwork)
      IMPLICIT NONE
      INTEGER ne, n, i, j
      INTEGER irow(ne), jcol(ne), irowsort(ne), jcolsort(ne)
      INTEGER irwork(n)

C     Initialize workspace
```

```
      DO 10 i=1,n
         irwork(i) = 0
 10   CONTINUE

      DO 20 j=1,ne
         irwork(irow(j)) = irwork(irow(j))+1
 20   CONTINUE

C     irwork(i) now contains # elements equal to i

      DO 30 i=2,n
         irwork(i) = irwork(i)+irwork(i-1)
 30   CONTINUE

C     irwork(i) now contains # elements less than or equal to i

      DO 40 j=ne,1, -1
         irowsort(ne+1-irwork(irow(j))) = irow(j)
         jcolsort(ne+1-irwork(irow(j))) = jcol(j)
         irwork(irow(j)) = irwork(irow(j)) - 1
 40   CONTINUE

      RETURN
      END
C=====================================
      SUBROUTINE countsortd(ne, n, irow, jcol,f,irowsort,jcolsort,fsort,
     $  irwork)
      IMPLICIT NONE
      INTEGER ne, n, i, j
      INTEGER irow(ne), jcol(ne), irowsort(ne), jcolsort(ne)
      DOUBLE PRECISION f(ne), fsort(ne)
      INTEGER irwork(n)

C     Initialize workspace

      DO 10 i=1,n
         irwork(i) = 0
 10   CONTINUE

      DO 20 j=1,ne
         irwork(irow(j)) = irwork(irow(j))+1
 20   CONTINUE

C     irwork(i) now contains # elements equal to i

      DO 30 i=2,n
         irwork(i) = irwork(i)+irwork(i-1)
 30   CONTINUE

C     irwork(i) now contains # elements less than or equal to i

      DO 40 j=ne,1, -1
         irowsort(irwork(irow(j))) = irow(j)
```

```
            jcolsort(irwork(irow(j))) = jcol(j)
            fsort(irwork(irow(j))) = f(j)
            irwork(irow(j)) = irwork(irow(j)) - 1
 40     CONTINUE

        RETURN
        END



C==================================
        SUBROUTINE heapsort(ne,irow,jcol)
C
C       Code adapted from Numerical Recipes in Fortran
C
        IMPLICIT NONE
        INTEGER ne
        INTEGER irow(ne), jcol(ne)
        INTEGER i, ir, j, l
        INTEGER itemprow, jtempcol

C
C       Check if we are called with only one thing to be sorted
C
        IF (ne.LT.2) RETURN

        l=ne/2+1
        ir=ne

 10     CONTINUE
            IF (l.GT.1) THEN
                l=l-1
                itemprow=irow(l)
                jtempcol=jcol(l)
            ELSE
                itemprow=irow(ir)
                jtempcol=jcol(ir)
                irow(ir)=irow(1)
                jcol(ir)=jcol(1)
                ir=ir-1

                IF (ir.EQ.1) THEN
                    irow(1)=itemprow
                    jcol(1)=jtempcol
                    RETURN
                ENDIF
            ENDIF
            i=l
            j=l+l
 20         IF (j.LE.ir) THEN
                IF (j.LT.ir) THEN
                    IF(irow(j).LT.irow(j+1)) j=j+1
                ENDIF
                IF (itemprow.LT.irow(j)) THEN
                    irow(i)=irow(j)
                    jcol(i)=jcol(j)
                    i=j
                    j=j+j
                ELSE
                    j=ir+1
                ENDIF
            GOTO 20
            ENDIF
            irow(i)=itemprow
            jcol(i)=jtempcol
        GOTO 10
        END
C==================================
        SUBROUTINE heapsortd(ne,irow,jcol,f)
C
C       Code adapted from Numerical Recipes in Fortran
C
        IMPLICIT NONE
        INTEGER ne
        INTEGER irow(ne), jcol(ne)
        INTEGER i, ir, j, l
        INTEGER itemprow, jtempcol
        DOUBLE PRECISION f(ne)
        DOUBLE PRECISION ftemp

C
C       Check if we are called with only one thing to be sorted
C
        IF (ne.LT.2) RETURN

        l=ne/2+1
        ir=ne

 10     CONTINUE
            IF (l.GT.1) THEN
                l=l-1
                itemprow=irow(l)
                jtempcol=jcol(l)
                ftemp=f(l)
            ELSE
                itemprow=irow(ir)
                jtempcol=jcol(ir)
                ftemp=f(ir)
                irow(ir)=irow(1)
                jcol(ir)=jcol(1)
                f(ir)=f(1)
                ir=ir-1

                IF (ir.EQ.1) THEN
                    irow(1)=itemprow
                    jcol(1)=jtempcol
                    f(1)=ftemp
                    RETURN
                ENDIF
            ENDIF
            i=l
            j=l+l
20          IF (j.LE.ir) THEN
                IF (j.LT.ir) THEN
                    IF(irow(j).LT.irow(j+1)) j=j+1
                ENDIF
                IF (itemprow.LT.irow(j)) THEN
                    irow(i)=irow(j)
                    jcol(i)=jcol(j)
                    f(i)=f(j)
                    i=j
                    j=j+j
                ELSE
                    j=ir+1
                ENDIF
            GOTO 20
            ENDIF
            irow(i)=itemprow
            jcol(i)=jtempcol
            f(i)=ftemp
        GOTO 10
        END
```

## C.2 Bayesian Parameter Estimation for a Correlated Random Walk

```
      program main
      implicit none
c###  Expermental observations (generated from a simulation)
      integer NDATA
      parameter (NDATA=21)
      double precision YOBS(NDATA)
      data YOBS /0D0, 3.672119e-02, 4.827638e+00, 4.637363e+00,
     $ 4.560976e+00, 8.747609e+00, 6.471495e+00, 5.676686e+00,
     $ 9.041612e+00, 1.188896e+01, 1.452761e+01, 1.844011e+01,
     $ 2.012600e+01, 2.589751e+01, 2.461056e+01, 2.639762e+01,
     $ 2.668319e+01, 2.571101e+01, 2.255608e+01, 1.881938e+01,
     $ 2.001533e+01/
      double precision DELTAT, SIGMA
      parameter (DELTAT=1D0, SIGMA=1D0)
c###  Lambda, Speed grid
      integer NDIST, NSPEED
      parameter (NDIST=51, NSPEED=51)
      double precision MINDIST, MAXDIST, MINSPEED, MAXSPEED
      parameter (MINDIST=0.5D0, MAXDIST=10D0)
      parameter (MINSPEED=2D0, MAXSPEED=8D0)
      double precision DIST(NDIST), SPEED(NSPEED)
c###  Solution
      double precision z(NSPEED,NDIST)
c###  Intermediate variables
c###  Upper a lower integration limits are y_i-NWIDTH*alpha, and
c###  y_i+NWIDTH*alpha
      integer NWIDTH
      parameter (NWIDTH=6)
c###  Number of quadrature points (must be an odd number!!)
      integer NINTP, NFMAX
      parameter (NINTP=201)
      parameter (NFMAX=1000000)
      integer nf, ixmin, ixmax
      integer iceil
      integer iptrst, iptrend, iptrphis, iptrphie
      integer istart, igstart,iend
      double precision x(NFMAX), f(NFMAX), w(NFMAX)
      double precision phi(NFMAX), gamma(NFMAX)
      double precision phitemp(NFMAX), gammatemp(NFMAX)
      double precision p1(NFMAX), p2(NFMAX), p3(NFMAX), p4(NFMAX)
      double precision xn(NINTP), r(NINTP), s(NINTP), t(NINTP)
      double precision width1, width2, w1(NDIST), w2(NSPEED)
      double precision temp1, temp2
      double precision width, xmin, xmax, hmin, hmax
      double precision time1, time2
      integer i, j, k, l

      if (mod(NINTP,2).ne.1) then
        write(*,*) "NINTP is even"
```

```
          stop
        endif

        call cputime(time1)
c###    Generate lambda speed grid
        width=(MAXDIST−MINDIST)/(NDIST−1)
        do i=1,NDIST
          DIST(i)= (i−1)*width+MINDIST
        enddo

        width=(MAXSPEED−MINSPEED)/(NSPEED−1)
        do i=1,NSPEED
          SPEED(i) = (i−1)*width+MINSPEED
        enddo

        do j=1,NDIST
          do i=1,NSPEED
c###    Set PDF = 0 and begin accumulations
            z(i,j)=0D0
c###    Set initial x grid for evaluation of measurement PDF.
            width=2*SPEED(i)*DELTAT/(NINTP−1)
            ixmin=−iceil(NWIDTH*SIGMA/width)
            ixmax=+iceil(NWIDTH*SIGMA/width)
            nf=2*ixmax+1
            if (nf.gt.NFMAX) then
              write(*,*) "Insufficient memory. nf =", nf
              stop
            endif
            xmin=width*ixmin
            xmax=width*ixmax
            do l=ixmin,ixmax
              x(l−ixmin+1)=l*width+YOBS(1)
            enddo
c###    Calculate transition probabilities
            do k=1,NINTP
              xn(k) = (k−1)*width−SPEED(i)*DELTAT
            enddo
            call cond11(r,xn,DIST(j),DELTAT,SPEED(i),NINTP)
            call cond21(s,xn,DIST(j),DELTAT,SPEED(i),NINTP)
            call cond22(t,xn,DIST(j),DELTAT,SPEED(i),NINTP)

            do k=1,nf
              phi(k)=1D0
              gamma(k)=1D0
            enddo
c###    Loop over data performing convolutions with transition PDFs.
            do k=1,NDATA−1
               call gauss(f,x,YOBS(k),SIGMA,nf)
               do l=1,nf
c###    Store integrand for Dirac-delta function
                 phitemp(l)=phi(l)*f(l)
                 phi(l)=width*phitemp(l)
                 gammatemp(l)=gamma(l)*f(l)
                 gamma(l)=width*gammatemp(l)
```

```
         enddo


c###  Calculate pointers into convolution that
c###  correspond to the NWIDTH*ALPHA limits on the
c###  measurement PDF and recalculate grid.
         hmin=xmin-SPEED(I)*DELTAT
         hmax=xmax+SPEED(I)*DELTAT
         xmin=YOBS(k+1)-NWIDTH*SIGMA
         xmin=nint((xmin-YOBS(1))/width)*width+YOBS(1)
         do l=1,nf
           x(l)=(l-1)*width+xmin
         enddo
         xmax=x(nf)
         if ((xmax.lt.hmin).or.(xmin.gt.hmax)) then
c###  Intersection is empty we know that z=0 !!
           z(i,j)=0D0
           goto 100
         endif
         iptrst=max(1,nint((xmin-hmin)/width)+1)
         iptrend=min(nf+NINTP-1,nf+NINTP-1-
     $                 nint((hmax-xmax)/width))

c###  Calculate pointers into phi and gamma
         iptrphis=max(1,nint((hmin-xmin)/width)+1)
         iptrphie=iptrphis+iptrend-iptrst

c###  Call convolution code

         call partconv (phi,nf,r,NINTP,p1,iptrst,iptrend)
         call partconv (gamma,nf,s,NINTP,p2,iptrst,iptrend)
         call partconv (gamma,nf,t,NINTP,p3,iptrst,iptrend)
         call partconv (phi,nf,s,NINTP,p4,iptrst,iptrend)
c###  Zero out parts that don't intersect.
         do l=1,iptrphis-1
           phi(l)=0D0
           gamma(l)=0D0
         enddo

         do l=1,iptrend-iptrst+1
          phi(l+iptrphis-1)=p1(l)+p2(l)
         enddo

         istart=max(iptrst,1)
         iend=min(nf,iptrend)

         do l=istart,iend
          phi(l+iptrphis-istart)=exp(-SPEED(i)/DIST(j)*DELTAT)*
     $            phitemp(l+iptrst-istart)+phi(l+iptrphis-istart)
         enddo

         do l=1,iptrend-iptrst+1
          gamma(l+iptrphis-1)=p3(l)+p4(l)
         enddo
```

```
            istart=max(NINTP,iptrst)
            iend=min(nf+NINTP,iptrend)
            igstart=max(1,iptrst-NINTP+1)
            iptrphis=iptrphis+max(0,NINTP-iptrst)
            do l=istart,iend
              gamma(l+iptrphis-istart)=exp(-SPEED(i)/DIST(j)*DELTAT)*
     $             gammatemp(l-istart+igstart)+
     $             gamma(l+iptrphis-istart)
            enddo

            do l=iptrphie+1,nf
              phi(l)=0D0
              gamma(l)=0D0
            enddo

c###    enddo k
          enddo
          call gauss(f,x,YOBS(NDATA),SIGMA,nf)
          do l=1,nf
            phi(l)=width*phi(l)*f(l)
            gamma(l)=width*gamma(l)*f(l)
            z(i,j)=z(i,j)+phi(l)+gamma(l)
          enddo
 100    continue
c###    enddo i
          enddo
c###    enddo j
        enddo
c###    Normalize z(i,j) assuming a uniform prior for (C,lambda)

c       call qsimp(w1,NLAMBDA)
c         call qsimp(w2,NSPEED)
c       width1=(MAXLAMBDA-MINLAMBDA)/(NLAMBDA-1)
c       width2=(MAXSPEED-MINSPEED)/(NSPEED-1)
c
c       temp2=0D0
c       do j=1,NLAMBDA
c         temp1=0D0
c         do i=1,NSPEED
c           temp1=temp1+z(i,j)*w2(i)
c         enddo
c         temp2=temp2+w1(j)*temp1
c       enddo
c       do j=1,NLAMBDA
c         do i=1,NSPEED
c         z(i,j)=z(i,j)/temp2
c         enddo
c         enddo

        call cputime(time2)
        write(*,*) "Total elapsed CPU time:", time2-time1
        open(10,file='results.m')
          write(10,*) "% Matlab results file for posterior PDF"
          write(10,*) "% written by correlated.f"
          write(10,*) "% Total elapsed CPU time: ", time2-time1, "s"
          write(10,*)
          write(10,*) "ndata = ", NDATA-1, ";"
          write(10,*) "y =[ "
          do i=2,NDATA
          write(10,*) YOBS(i)
          enddo
          write(10,*) "];"
          write(10,*) "DeltaT =", DELTAT, ";"
          write(10,*) "alpha =", SIGMA, ";"
          write(10,*) "i=(1:1:ndata)';"
          write(10,*) "theta = 12/(DeltaT*(ndata*(ndata+2)*(ndata+1)))
     $ *(ndata/2*sum(y)-sum(i.*y));"
          write(10,*) "sigma=sqrt(12*alpha^2/
     $ (DeltaT^2*ndata*(ndata+1)*(ndata+2)));"
          write(10,*) "speed =[ "
          do i=1,NSPEED
          write(10,*) SPEED(i)
          enddo
          write(10,*) "];"
          write(10,*)
          write(10,*) "dist =[ "
          do i=1,NDIST
          write(10,*) DIST(i)
          enddo
          write(10,*) "];"
          write(10,*) "z =["
          do i=1,NSPEED
          write(10,1000) (z(i,j), j=1,NDIST)
          enddo
          write(10,*) "];"
          write(10,*) "z=z./repmat(speed,1,length(dist));"
          write(10,*) "z=z./repmat(dist',length(speed),1);"
          write(10,*) "contour(dist,speed,z,20)"
          write(10,*) "pause"
          write(10,*) "width=speed(2)-speed(1);"
          write(10,*) "z(:,1)=z(:,1)/(width*sum(z(:,1)));"
          write(10,*) "z2=normpdf(speed,theta,sigma)
     $ +normpdf(speed,-theta,sigma);"
          write(10,*) "z2=z2/(width*sum(z2));"
          write(10,*) "plot(speed,z2,speed,z(:,1),'+')"
          write(10,*) "% End of file"
        close(10)
1000  format(1000(D16.9, 1X))
        end

        subroutine cond11(r,x,DIST,DELTAT,SPEED,NINT)
C###  Subroutine to evaluate the transition probability r.
        implicit none
C###  Inputs: x(NINT), LAMBDA, DELTAT, SPEED
        integer NINT
        double precision x(NINT), DIST,LAMBDA, DELTAT, SPEED
C###  Outputs: r(NINT)
```

```
      double precision r(NINT)
C###  Intermediate variables
      integer i
      double precision gamma
      double precision dbesi1
      double precision EPS
      parameter (EPS=1D-60)

      LAMBDA=SPEED/DIST

      do i=1,NINT
        gamma=DELTAT * DELTAT - x(i) * x(i)/(SPEED*SPEED)

        if (gamma.le.0) then
          gamma=EPS
        else
          gamma=LAMBDA * sqrt (gamma)
        endif
          r(i)=exp (-LAMBDA * DELTAT) * LAMBDA * LAMBDA *
     $      dbesi1 (gamma) / (gamma*SPEED) * (DELTAT - x(i)/SPEED)
      enddo
      return
      end

      subroutine cond21(s, x,DIST,DELTAT,SPEED,NINT)
C###  Subroutine to evaluate the transition probability, s.
      implicit none
C###  Inputs: x(NINT), LAMBDA, DELTAT, SPEED
      integer NINT
      double precision x(NINT), DIST,LAMBDA, DELTAT, SPEED
C###  Outputs: s(NINT)
      double precision s(NINT)
C###  Intermediate variables
      integer i
      double precision gamma
      double precision dbesi0
      double precision EPS
      parameter (EPS=1D-60)

      LAMBDA=SPEED/DIST
      do i=1,NINT
        gamma=DELTAT * DELTAT - x(i) * x(i)/(SPEED*SPEED)
        if (gamma.le.0) then
          gamma=EPS
        else
          gamma=LAMBDA * sqrt (gamma)
        endif
        s(i)=exp (-LAMBDA * DELTAT) * LAMBDA * dbesi0 (gamma)
     $                /SPEED
      enddo

      return
      end
      subroutine cond22(t, x,DIST,DELTAT,SPEED,NINT)
C###  Subroutine to evaluate the transition probability, t.
      implicit none
C###  Inputs: x(NINT), LAMBDA, DELTAT, SPEED
      integer NINT
      double precision x(NINT), DIST,LAMBDA, DELTAT, SPEED
C###  Outputs: tnorm(NINT)
      double precision t(NINT)
C###  Intermediate variables
      integer i
      double precision gamma
      double precision dbesi1
      double precision EPS
      parameter (EPS=1D−60)
      LAMBDA=SPEED/DIST
      do i=1,NINT
        gamma=DELTAT * DELTAT − x(i) * x(i)/(SPEED*SPEED)

        if (gamma.le.0) then
          gamma=EPS
        else
          gamma=LAMBDA * sqrt (gamma)
        endif
          t(i)=exp (−LAMBDA * DELTAT) * LAMBDA * LAMBDA *
     $     dbesi1 (gamma) / (gamma*SPEED) * (DELTAT + x(i)/SPEED)
      enddo

      return
      end

      subroutine gauss (z,x,mu,sigma,N)
C###  Subroutine to calculate Gaussian density with mean, mu, and variance
C###  sigma.
      implicit none
C###  Inputs:  x(N), mu, sigma
      integer N
      double precision x(N)
      double precision mu, sigma
C###  Outputs: z(N)
      double precision z(N)
C###  Intermediate variables
      integer i
      double precision k
      parameter(k=0.3989422804014327D0)

      do i=1,N
        z(i)=k/sigma*exp(−(x(i)−mu)*(x(i)−mu)/(2D0*sigma*sigma))
      enddo

      return
      end

      subroutine partconv (f,nf,g,ng,h,iptrst,iptrend)
c###  Subroutine to calculate partial numerical convolution.
```

```
c###    We are only interested in the part of the convolution
c###    that overlaps the non-zero section of the measurement
c###    PDF. hfull(x)=int f(x)*g(x-u) du
c###    where nhfull=nf+ng.
c###    h=hfull(iptrh:iptrend)
        implicit none
c###    Inputs: f(nf), g(ng), nf, ng, iptrst, iptrend
        integer nf, ng, iptrst, iptrend
        double precision f(nf), g(ng)
c###    Outputs: h(nf)
        double precision h(*)
c###    Intermediate variables
        integer i, j, istart, iend, jstart, jend

        do i=iptrst,iptrend
          h(i−iptrst+1)=0D0
c###    Band index
          istart=min(i,ng)
          iend=max(1,i−nf+1)
c###    Column index
          jstart=max(1,i−ng+1)
          jend=jstart+istart−iend
c###    Quadrature uses 1/2 endpoints
              h(i−iptrst+1)=h(i−iptrst+1)+0.5D0*f(jstart)*g(istart)
              do j=jstart+1,jend−1
                h(i−iptrst+1)=h(i−iptrst+1)+f(j)*g(istart−j+jstart)
              enddo
              h(i−iptrst+1)=h(i−iptrst+1)+0.5D0*f(jend)*g(iend)
        enddo

c###    If there is only one entry in the row the integral is zero
        if (iptrst.eq.1) then
          h(1)=0D0
        endif

        if (iptrend.eq.(nf+ng−1)) then
          h(nf+ng−1)=0D0
        endif
        return
        end


        subroutine partconv2 (f,nf,g,ng,h,iptrst,iptrend)
c###    Subroutine to calculate partial numerical convolution.
c###    We are only interested in the part of the convolution
c###    that overlaps the non-zero section of the measurement
c###    PDF. hfull(x)=int f(x)*g(x-u) du
c###    where nhfull=nf+ng.
c###    h=hfull(iptrh:iptrend)

        implicit none
c###    Inputs: f(nf), g(ng), nf, ng, iptrst, iptrend
        integer nf, ng, iptrst, iptrend
        double precision f(nf), g(ng)
```

```
c###    Outputs: h(nf)
        double precision h(*)
c###    Intermediate variables
        integer i, j, istart, iend, jstart, jend

        do i=1,iptrend−iptrst+1
          h(i)=0D0
        enddo

        istart=max(1,iptrst+1−nf)
        iend=min(ng,iptrend)

        do i=istart,iend
          jstart=max(iptrst+1−i,1)
          jend=min(nf,iptrend+1−i)
          do j=jstart,jend
            h(i+j−iptrst)=f(j)*g(i)+h(i+j−iptrst)
          enddo
        enddo

        return
        end

        function iceil(x)
c###    Function to round towards infinity
        implicit none
        double precision x
        integer iceil

        if (((x−int(x)).eq.0D0).or.(int(x).lt.0D0)) then
          iceil=int(x)
        else
          iceil=int(x)+1
        endif
        return
        end

        subroutine qsimp(w,nf)
c###    Subroutine to calculate vector of Simpson weights
        implicit none
        integer nf, n,i
        double precision w(nf)

        if (mod(nf,2).ne.1) then
          write(*,*) "NF is even"
          stop
        endif
        n=(nf−1)/2
        w(1)=1D0/3D0
        w(nf)=1D0/3D0
        do i=1,n−1
          w(1+2*i)=2D0/3D0
        enddo
```

```
do i=1,n
  w(2*i)=4D0/3D0
enddo

return
end
```

# Bibliography

[1] C. S. Adjiman, I. P. Androulakis, and C. A. Floudas. A global optimization method, $\alpha$BB, for general twice-differentiable constrained NLPs - II. Implementation and computational results. *Computers and Chemical Engineering*, 22(9): 1159–1179, 1998.

[2] C. S. Adjiman, S. Dallwig, C. A. Floudas, and A. Neumaier. A global optimization method, $\alpha$BB, for general twice-differentiable constrained NLPs - I. Theoretical advances. *Computers Chem. Engng*, 22(9):1137–1158, 1998.

[3] *Harwell Subroutine Library Specifications: Release 11.* AEA and SERC, July 1993.

[4] H. Akaike. A new look at the statistical model identification. *IEEE Trans. Automatic Control*, 19:716–723, 1974.

[5] F. A. Al-Khayyal and J. E. Falk. Jointly constrained biconvex programming. *Mathematics of Operations Research*, 8(2):273–286, 1983.

[6] J. Albanell, F. Rojo, S. Averbuch, A. Feyereislova, J. M. Mascaro, R. Herbst, P. LoRusso, D. Rischin, S. Sauleda, J. Gee, R. I. Nicholson, and J. Baselga. Pharmacodynamic studies of the epidermal growth factor receptor inhibitor ZD1839 in skin from cancer patients: Histopathologic and molecular consequences of receptor inhibition. *J. Clin. Oncol.*, 20(1):110–124, 2002.

[7] U. Alon, M. G. Surette, N. Barkai, and S. Leibler. Robustness in bacterial chemotaxis. *Nature*, 397:168–171, 1999.

[8] W. Alt, O. Brosteanu, B. Hinz, and H. W. Kaiser. Patterns of spontaneous motility in videomicrographs of human epidermal keratinocytes. *Biochem. Cell Biol.*, 73:441–459, 1995.

[9] E. D. Andersen and Y. Ye. A computational study of the homogeneous algorithm for large-scale convex optimization. *Computational Optimization and Applications*, 10:243–269, 1998.

[10] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen. LAPACK users' guide. Software, Environments, Tools. SIAM, Philadelphia, 3rd edition, 1999.

[11] M. Arioli, J. W. Demmel, and I. S. Duff. Solving sparse linear systems with sparse backward error. *SIAM J. Matrix Anal. Appl.*, 10(2):165–190, 1989.

[12] A. Arkin, J. Ross, and H. H. McAdams. Stochastic kinetic analysis of developmental pathway bifurcation in phage $\lambda$-infected *escherichia coli* cells. *Genetics*, 149:1633–1648, 1998.

[13] A. R. Asthagiri and D. A. Lauffenburger. Bioengineering models of cell signaling. *Annu. Rev. Biomed. Eng.*, 2:31–53, 2000.

[14] N. Balakrishnan and M. V. Koutras. *Runs with Scans and Applications.* John Wiley & Sons, 2002.

[15] N. Barkai and S. Leibler. Robustness in simple biochemical networks. *Nature*, 387:913–917, 1997.

[16] P. I. Barton. *The modeling and simulation of combined discrete/continuous processes.* PhD thesis, University of London, 1992.

[17] P. I. Barton. Automated identity elimination. ABACUSS Project Report, Massachusetts Institute of Technology, 1994.

[18] M. S. Bazaraa, H. D. Sherali, and C. M. Shetty. *Nonlinear Programming: Theory and Algorithms*. John Wiley & Sons, New York, 2nd edition, 1993.

[19] E. F. Beckenbach and R. Bellman. *Inequalities*. Springer-Verlag, Berlin, 1961.

[20] M. Berz and G. Hoffstätter. Computation and application of Taylor polynomials with interval remainder bounds. *Reliable Computing*, 4:83–97, 1998.

[21] M. Berz and K. Makino. New methods for high-dimensional verified quadrature. *Reliable Computing*, 5:13–22, 1999.

[22] U. S. Bhalla and R. Iyengar. Emergent properties of networks of biological signaling pathways. *Science*, 283:381–387, 1999.

[23] N. Bhatia, C. Agarwal, and R. Agarwal. Differential responses of skin cancer-chemopreventive agents silibinin, quercetin, and epigallocatechin 3-gallate on mitogenic signaling and cell cycle regulators in human epidermoid carcinoma a431 cells. *Nutr. Cancer*, 39(2):292–299, 2001.

[24] A. Bhatt, I. Kaverina, C. Otey, and A. Huttenlocher. Regulation of focal complex composition and disassembly by the calcium-dependent protease calpain. *J. Cell. Sci.*, 115(17):3415–3425, 2002.

[25] D. A. Binder. Comment on estimating mixtures of normal distributions and switching regressions. *J. of the American Statistical Association*, 73(364):746–747, 1978.

[26] P. C. Bishop, T. Myers, R. Robey, D. W. Fry, E. T. Liu, M. V. Blagosklonny, and S. E. Bates. Differential sensitivity of cancer cells to inhibitors of the epidermal growth factor receptor family. *Oncogene*, 21(1):119–27, 2002.

[27] H. G. Bock. *Numerical Treatment of Inverse Problems in Chemical Reaction Kinetics*, volume 18 of *Springer Series in Chemical Physics*, chapter 8. Springer Verlag, 1981.

[28] G. E. P. Box and W. J. Hill. Discrimination among mechanistic models. *Technometrics*, 9(1):57–71, 1967.

[29] G. E. P. Box and G. C. Tiao. *Bayesian Inference in Statistical Analysis.* Wiley Classics Library. John Wiley & Sons, New York, 1992.

[30] D. Bray, N. Money, F. Harold, and J. Bamburg. Responses of growth cones to changes in osmolarity of the surrounding medium. *J. Cell Sci.*, 98:507–515, 1991.

[31] D. Bray and J. G. White. Cortical flow in animal cells. *Science*, 239:883–888, 1988.

[32] L. G. Bretthorst. Fundamental theories of physics. In G. J. Erickson and C. R. Smith, editors, *Maximum-Entropy and Bayesian Methods in Science and Engineering*, volume 1, chapter Excerpts from Bayesian Spectrum Analysis and Parameter Estimation, pages 75–145. Kluwer Academic Publishers, 1988.

[33] H. C. Brinkman. Brownian motion in a field of force and the diffusion theory of chemical reactions. *Physica*, 22:29–34, 1956.

[34] K. Burridge and M. Chrzanowska-Wodnicka. Focal adhesions, contractility, and signaling. *Annu. Rev. Cell Dev. Biol.*, 12:463–519, 1996.

[35] S. L. Campbell. Linearizations of DAEs along trajectories. *ZAMP*, 46:70–84, 1995.

[36] K. C. Chan, W. F. Knox, J. M. Gee, J. Morris, R. I. Nicholson, C. S. Potten, and N. J. Bundred. Effect of epidermal growth factor receptor tyrosine kinase inhibition on epithelial proliferation in normal and premalignant breast. *Cancer Res.*, 62(1):122–128, 2002.

[37] S. Chandrasekhar. Stochastic problems in physics and astronomy. *Reviews of Modern Physics*, 15(1):1–89, 1943.

[38] M. K. Charter and S. F. Gull. Maximum entropy and its application to the calculation of drug absorption rates. *J. of Pharmacokinetics and Biopharmaceutics*, 15(6):645–655, 1987.

[39] M. K. Charter and S. F. Gull. Maximum entropy and drug absorption. *J. of Pharmacokinetics and Biopharmaceutics*, 19(5):497–520, 1991.

[40] W. S. Chen, C. S. Lazar, M. Poenie, R. Y. Tsein, G. N. Gill, and M. G. Rosenfeld. Requirement for intrinsic protein tyrosine kinase in the immediate and late actions of the EGF receptor. *Nature*, 328:820–823, 1987.

[41] J. A. Clabaugh, J. E. Tolsma, and P. I. Barton. ABACUSS II: Advanced modeling environment and embedded simulator. Technical report, Massachusetts Institute of Technology, 2000. http://yoric.mit.edu/abacuss2/abacuss2.html.

[42] D. E. Clapham. Calcium signaling. *Cell*, 80:259–268, 1995.

[43] T. F. Coleman, B. S. Garbow, and J. J. Moré. Software for estimating sparse Jacobian matrices. *ACM Transactions on Mathematical Software*, 10(3):329–345, 1984.

[44] T. F. Coleman and J. J. Moré. Estimation of sparse Jacobian matrices and graph coloring problems. *SIAM J. Numer. Anal.*, 20(1):187–209, 1983.

[45] J. Condeelis, M. D. Bresnick, S. Dharmawardhane, R. Eddy, A. L. Hall, R. Saurerer, and V. Warren. Mechanics of amoeboid chemotaxis: an evaluation of the cortical expandion model. *Dev. Genet.*, 11:333–340, 1990.

[46] S. D. Conte and C. de Boor. *Elementary Numerical Analysis: An Algorithmic Approach*. International Series in Pure and Applied Mathematics. McGraw-Hill, New York, 3rd edition, 1980.

[47] G. F. Corliss and L. B. Rall. Adaptive, self-validating numerical quadrature. *SIAM J. Sci. Stat. Comput.*, 8(5):831–847, 1987.

[48] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms*. The MIT Press, Cambridge, 1994.

[49] D. R. Cox. *Renewal Theory*. Methuen, London, 1962.

[50] R. T. Cox. Probability, frequency and reasonable expectation. *American Journal of Physics*, 14(1):1–13, 1946.

[51] R. T. Cox. *The Algebra of Probable Inference*. John Hopkins University Press, Baltimore, MD, 1961.

[52] B. D. Cuevas, A. N. Abell, J. A. Witowsky, T. Yujiri, J. A. Witowsky, T. Yujuri, N. L. Johnson, K. Kesavan, M. Ware, P. L. Jones, S. A. Weed, R. L. DeBiasi, Y. Oka, K. L. Tyler, and G. L. Johnson. MEKK1 regulates calpain-dependent proteolysis of focal adhesion proteins for rear-end detachment of migrating fibroblasts. *EMBO*, 22(13):3346–3355, 2003.

[53] A. R. Curtis, M. J. D. Powell, and J. K. Reid. On the estimation of sparse Jacobian matrices. *J. Inst. Math. App.*, 13(1):117–119, 1974.

[54] M. H. DeGroot. *Probability and Statistics*. Addison-Welsey, Reading, MA, 1975.

[55] C. DeLisi and F. Marchetti. A theory of measurement error and its implications for spatial and temporal gradient sensing during chemotaxis. *Cell Biophys.*, 5: 237–253, 1983.

[56] C. DeLisi and F. W. Wiegel. Effect of nonspecific forces and finite receptor number on rate constants of ligand-cell bound receptor interactions. *Proc. Natl. Acad. Sci.*, 78(9):5569–5572, 1981.

[57] K. A. DeMali and K. Burridge. Coupling membrane protrusion and cell adhesion. *J. Cell Sci.*, 116(12):2389–2397, 2003.

[58] P. A. DeMilla, K. Barbee, and D. A. Lauffenburger. Mathematical model for the effects of adhesion and mechanics on cell migration speed. *Biophys. J.*, 60: 15–37, 1991.

[59] J. Demmel and B. Kågström. The generalized Schur decomposition of an arbitrary pencil $A - \lambda B$: Robust software with error bounds and applications. Part II: Software and applications. *ACM Transactions on Mathematical Software*, 19(2):175–201, 1993.

[60] J. Demmel and B. Kågström. The generalized Schur decompotisision of an arbitrary pencil $A - \lambda B$: Robust software with error bounds and applications. Part I: Theory and algorithms. *ACM Transactions on Mathematical Software*, 19(2):160–174, 1993.

[61] A. E. DeWitt, J. Y. Dong, H. S. Wiley, and D. A. Lauffenburger. Quantitative analysis of the EGF receptor autocrine system reveals cryptic regulation of cell response by ligand capture. *J. Cell Sci.*, 114:2301–2313, 2001.

[62] R. B. Dickinson and R. T. Tranquillo. Optimal estimation of cell movement indices from the statistical analysis of cell tracking data. *AIChE J.*, 39(12): 1995–2010, 1993.

[63] R. E. Dolmetsch, K. Xu, and R. S. Lewis. Calcium oscillations increase the efficiency and specificity of gene expression. *Nature*, 392(30):933–936, 1998.

[64] J. M. Douglas. *Conceptual Design of Chemical Processes.* McGraw-Hill Book Company, New York, 1988.

[65] J. Downward, P. Parker, and M. D. Waterfield. Autophosphorylation sites on the epidermal growth factor receptor. *Nature*, 311:483–485, 1984.

[66] J. Downward, M. D. Waterfield, and P. J. Parker. Autophosphorylation and protein kinase C phosphorylation of the epidermal growth factor receptor. *J. Biol. Chem.*, 260(27):14538–14546, 1985.

[67] D. Draper. Assessment and propagation of model uncertainty. *J. R. Statist. Soc. B*, 57(1):45–97, 1995.

[68] I. S. Duff. On algorithms for obtaining a maximum transversal. *ACM Transactions on Mathematical Software*, 7:315–330, 1981.

[69] I. S. Duff, A. M. Erisman, C. W. Gear, and J. K. Reid. Sparsity structure and Gaussian elimination. *SIGNUM Newsletter*, 23(1):2–8, 1988.

[70] I. S. Duff, A. M. Erisman, and J. K. Reid. *Direct Methods for Sparse Matrices.* Clarendon Press, Oxford, 1992.

[71] I. S. Duff and J. K. Reid. An implementation of Tarjan's algorithm for the block triangularization of a matrix. *ACM Transactions on Mathematical Software*, 4 (2):137–147, 1978.

[72] I. S. Duff and J. K. Reid. MA48, a Fortran code for direct solution of sparse unsymmetric linear systems of equations. Technical report, Rutherford Appleton Laboratory, 1993. RAL-93-072.

[73] I. S. Duff and J. A. Scott. Computing selected eigenvalues of sparse unsymmetric matrices using subspace iteration. *ACM Transactions on Mathematical Software*, 19(2):137–159, 1993.

[74] I. S. Duff and J. A. Scott. Corrigendum. *ACM Transactions on Mathematical Software*, 21(4):490, 1995.

[75] H. S. Earp, T. L. Dawson, X. Li, and H. Yu. Heterodimerization and functional interaction between EGF receptor family members: A new signaling paradign with implications for breast cancer research. *Breast Cancer Research and Treatment*, 35:115–132, 1995.

[76] A. Einstein. Über die von der molekular-kinetischen Theorie der Wärme geforderte Bewegung von in ruhenden Flüssigkeiten suspendierten teilchen. *Ann. Phys.*, 17:549–560, 1905.

[77] W. R. Esposito and C. A. Floudas. Global optimization in parameter estimation of nonlinear algebraic models via the error-in-variables approach. *Ind. Eng. Chem. Res.*, 37:1841–1858, 1998.

[78] W. R. Esposito and C. A. Floudas. Deterministic global optimization in nonlinear optimal control problems. *Journal of Global Optimization*, 17:97–126, 2000.

[79] E. Evans. New physical concepts for cell amoeboid motion. *Biophys. J.*, 64: 1306–1322, 1993.

[80] J. E. Falk and R. M. Soland. An algorithm for separable nonconvex programming problems. *Management Science*, 15(9):550–569, 1969.

[81] E. M. Fallon and D. A. Lauffenburger. Computational model for effects of ligand/recepter binding properties on interleukin-2 trafficking dynamics and T cell proliferation response. *Biotechnol. Prog.*, 16:905–916, 2000.

[82] S. Fararooy, J. D. Perkins, T. I. Malik, M. J. Oglesby, and S. Williams. Process controllability toolbox (PCTB). *Computers Chem. Engng.*, 17(5-6):617–625, 1993.

[83] K. R. Fath and D. R. Burgess. Membrane motility mediated by unconventional myosin. *Curr. Opin. Cell Biol.*, 6:131–135, 1994.

[84] W. F. Feehery and P. I. Barton. Dynamic optimization with equality path constraints. *Ind. Eng. Chem. Res.*, 38(6):2350–2363, 1999.

[85] W. Feller. *An Introduction to Probability Theory and its Applications*, volume II, page 146. John Wiley & Sons, New York, 2nd edition, 1971.

[86] T. S. Ferguson. An inconsistent maximum likelihood estimate. *J. of the American Statistical Association*, 77(380):831–834, 1982.

[87] A. V. Fiacco and G. P. McCormick. *Nonlinear Programming: sequential unconstrained minimization techniques*. Classics in Applied Mathematics. SIAM, Philadelphia, 1990. Reprint.

[88] P. Forscher and S. J. Smith. Actions of cytochalasins on the organization of actin filaments and microtubules in neuronal growth cone. *J. Cell Biol.*, 124: 971–983, 1988.

[89] N. Friedman, M. Linial, I. Nachman, and D. Pe'er. Using Bayesian networks to analyze expression data. *J. Comput. Biol.*, 7:601–620, 2000.

[90] L. W. Fullerton. Portable special function routines. In W. Cowell, editor, *Portability of Numerical Software*, volume 57 of *Lecture Notes in Computer Science*, New York, 1976. Springer-Verlag.

[91] M. H. Gail and C. W. Boone. The locomotion of mouse fibroblasts in tissue culture. *Biophy. J.*, 10:980–993, 1970.

[92] C. W. Gardiner. *Handbook of Stochastic Methods for Physics, Chemistry and the Natural Sciences.* Springer Series in Synergetics. Springer-Verlag, New York, 1983.

[93] U. E. Gasser and M. E. Hatten. Central nervous system neurons migrate on astroglial fibers from heterotypic brain regions *in vitro*. *Proc. Natl. Acad. Sci.*, 87:4543–4547, 1990.

[94] E. P. Gatzke, J. E. Tolsma, and P. I. Barton. Construction of convex function relaxations using automated code generation techniques. *Optimization and Engineering*, 3(3):305–326, 2002.

[95] J. R. Gilbert. Predicting structure in sparse matrix computations. *SIAM J. Matrix Anal. Appl.*, 15(1):62–79, 1994.

[96] J. R. Gilbert, C. Moler, and R. Schreiber. Sparse matrices in MATLAB: Design and implementation. *SIAM J. Matrix Anal. Appl.*, 13(1):333–356, 1992.

[97] A. Glading, P. Chang, D. A. Lauffenburger, and A. Wells. Epidermal growth factor receptor activation of calpain is required for fibroblast motility and occurs

via an ERK/MAP kinase signaling pathway. *J. Biol. Chem.*, 275(4):2390–2398, 2000.

[98] A. Goldbeter, G. Dupont, and M. J. Berridge. Minimal model for signal-induced $Ca^{2+}$ oscillations and for their frequency encoding through protein phosphorylation. *Proc. Nat. Acad. Sci.*, 87:1461–1465, 1990.

[99] S. Goldstein. On diffusion by discontinuous movements, and on the telegraph equation. *Quart. J. Mech. and Applied Math.*, 4(2):129–156, 1951.

[100] G. H. Golub and C. F. Van Loan. *Matrix Computations.* The John Hopkins University Press, Baltimore, 3rd edition, 1996.

[101] T. H. Gronwall. Note on the derivatives with respect to a parameter of the solutions of a system of differential equations. *Annals of Mathematics*, pages 292–296, 1919.

[102] F. Gross. *Energy-efficient Design and Operation of Complex Distillation Processes.* PhD thesis, Swiss Federal Institute of Technology, Zürich, Switzerland, 1995.

[103] S. F. Gull. Bayesian inductive inference and maximum entropy. In G. J. Erickson and C. R. Smith, editors, *Maximum-Entropy and Bayesian Methods in Science and Engineering*, volume 1, pages 53–73. Kluwer Academic Publishers, 1988.

[104] F. G. Gustavson. Two fast algorithms for sparse matrices: Multiplication and permuted transposition. *ACM Transactions on Mathematical Software*, 4(3): 250–269, 1978.

[105] C. Han and B. P. Carlin. Markov chain Monte Carlo methods for computing Bayes factors: A comparative review. *J. of the American Statistical Association*, 96(455):1122–1132, 2001.

[106] G. W. Harrison. Dynamic models with uncertain parameters. In X. J. R. Avula, editor, *Proceedings of the First International Conference on Mathematical Modeling*, volume 1, pages 295–303, University of Missouri, Rolla, 1977.

[107] G. W. Harrison. Compartmental models with uncertain flow rates. *Mathematical Biosciences*, 43:131–139, 1979.

[108] G. W. Harrison. A stream pollution model with intervals for the rate coefficients. *Mathematical Biosciences*, 49:111–120, 1980.

[109] A. J. Hartemink, D. K. Gifford, and T. S. Jaakola. Using graphical models and genomic expression data to statistically validate models of genetic regulatory networks. *Pac. Symp. Biocomput.*, 1:422–433, 2001.

[110] P. Hartman. On the local linearization of differential equations. *Proceedings of the American Mathematical Society*, 14(4):568–573, 1963.

[111] P. Hartman. *Ordinary Differential Equations*. Birkhäuser, Boston, 2nd edition, 1982.

[112] J. M. Haugh and D. A. Lauffenburger. Analysis of receptor internalization as a mechanism for modulating signal transduction. *J. Theor. Biol.*, 195:187–218, 1998.

[113] J. M. Haugh, A. Wells, and D. A. Lauffenburger. Mathematical modeling of epidermal growth factor signaling through the phospholipase C pathway: Mechanistic insights and predictions for molecular interventions. *Biotech. Bioeng.*, 70:225–238, 2000.

[114] P. C. Hemmer. On a generalization of Smoluchowski's diffusion equation. *Physica*, 27:79–82, 1961.

[115] M. O. Hongler. On the diffusion induced by alternating renewal processes. *Physica A*, 188:597–606, 1992.

[116] R. Horst and H. Tuy. *Global Optimization: Deterministic Approaches*. Springer-Verlag, Berlin, 3rd edition, 1996.

[117] W. G. Hunter and A. M. Reiner. Designs for discriminating between two rival models. *Technometrics*, 7(3):307–323, 1965.

[118] T. Ideker and D. Lauffenburger. Building with a scaffold: emerging strategies for high- to low-level cellular modeling. *TRENDS in Biotechnology*, 21(6):255–262, 2003.

[119] E. W. Jacobsen and S. Skogestad. Inconsistencies in dynamic models for ill-conditioned plants: Application to low-order models of distillation columns. *Ind. Eng. Chem. Res.*, 33:631–640, 1994.

[120] E. T. Jaynes. Prior probabilities. *IEEE Trans. on System Science and Cybernetics*, SSC-4(3):227–241, 1968.

[121] E. T. Jaynes. *Probability Theory: The Logic Of Science*. Cambridge University Press, UK, 2003. Edited by G. Larry Bretthorst.

[122] H. Jeffreys. *Scientific Inference*. Cambridge University Press, Cambridge, third edition, 1973.

[123] H. Jeffreys. *Theory of Probability*. Oxford University Press, Oxford, third edition, 1998.

[124] M. Kac. A stochastic model related to the telegrapher's equation. *Rocky Mountain J. Math.*, 4(3):497–509, 1974. Reprinted from 1959.

[125] B. Kågström. RGSVD - an algorithm for computing the Kronecker structure and reducing subspaces of singular $A - \lambda B$ pencils. *SIAM J. Sci. Comput.*, 7 (1):185–211, 1986.

[126] S. Kaplan. Differential equations in which the Poisson process plays a role. *Bull. Amer. Math. Soc.*, 70:264–268, 1964.

[127] S. Karlin and H. M. Taylor. *A First Course in Stochastic Processes.* Academic Press, New York, 2nd edition, 1975.

[128] R. Katso, K. Okkenhaug, K. Ahmadi, S. White, J. Timms, and M. D. Waterfield. Cellular function of phosphoinositide 3-kinases: Implications for development, immunity, homeostasis, and cancer. *Annu. Rev. Cell Dev. Biol.*, 17: 615–675, 2001.

[129] P. Kesavan and P. I. Barton. Decomposition algorithms for nonconvex mixed-integer nonlinear programs. In *Fifth International Conference on Foundations of Computer-Aided Process Design*, volume 96(323) of *AIChE Symposium Series*, pages 458–461, Breckenridge, Colorado, 2000. URL `http://yoric.mit.edu/cgi-bin/bartongpub`.

[130] P. Kesavan and P. I. Barton. Generalized branch-and-cut framework for mixed-integer nonlinear optimization problems. In *7th International Symposium on Process Systems Engineering*, volume 24 of *Computers and Chemical Engineering*, pages 1361–1366, Keystone, Colorado, 2000. URL `http://yoric.mit.edu/cgi-bin/bartongpub`.

[131] P. Kesavan, E. P. G. R. J. Allgor, and P. I. Barton. Outer approximation algorithms for separable nonconvex mixed-integer nonlinear programs. URL `http://yoric.mit.edu/cgi-bin/bartongpub`. Submitted to Mathematical Programming (in revision), 2001.

[132] B. N. Kholodenko, O. V. Demin, G. Moehren, and J. B. Hoek. Quantification of short term signaling by the epidermal growth factor receptor. *J. Biol. Chem.*, 274(42):30169–30181, 1999.

[133] J. Kolega. Effects of mechanical tension on protrusive activity and microfilament and intermediate filament organization in an epidermal epitheliam moving in culture. *J. Cell. Biol.*, 102:1400–1411, 1986.

[134] W. Korohoda, M. Vöth, and J. Bereiter-Hahn. Biphasic response of human polymorphonuclear leucocytes and keratinocytes (epitheliocytes) from Xenopus *laevis* to mechanical stimulation. *Protoplasma*, 167:169–174, 1992.

[135] H. A. Kramers. Brownian motion in a field of force and the diffusion model of chemical reactions. *Physica*, 7(4):284–304, 1940.

[136] F. Krückeberg. Ordinary differential equations. In E. Hansen, editor, *Topics in Interval Analysis*, Oxford, 1969. Clarendon Press.

[137] H. Kwakernaak and R. Sivan. *Linear Optimal Control Systems.* John Wiley & Sons, New York, 1972.

[138] W. W. Lai, F. F. Chen, M. H. Wu, N. H. Chow, W. C. Su, M. C. Ma, P. F. Su, H. Chen, M. Y. Lin, and Y. L. Tseng. Immunohistochemical analysis of epidermal growth factor receptor family members in stage I non-small cell lung cancer. *Ann. Thorac Surg.*, 72(6):1868–1876, 2001.

[139] P. Langevin. Sur la theorie du mouvement Brownien. *Comptes Rendus*, 146: 530, 1908.

[140] D. A. Lauffenburger. A simple model for the effects of receptor-mediated cell-substratum adhesion on cell migration. *Chem. Eng. Sci.*, 44(9):1903–1914, 1999.

[141] D. A. Lauffenburger and J. J. Linderman. *Receptors: Models for Binding, Trafficking, and Signaling.* Oxford University Press, New York, 1993.

[142] R. B. Lehoucq, D. C. Sorensen, and C. Yang. ARPACK users' guide: Solution of large-scale eigenvalue problems with implicitly restarted Arnoldi methods. Software, Environments, Tools. SIAM, Philadelphia, 1998.

[143] T. Libotte, H.-W. Kaiser, W. Alt, and T. Bretschneider. Polarity, protrusion and retraction dynamics and their interplay during keratinocyte cell migration. *Exp. Cell Res.*, 270:129–137, 2001.

[144] H. Lodish, A. Berk, S. L. Zipursky, P. Matsudaira, D. Baltimore, and J. Darnell. *Molecular Cell Biology*. W. H. Freeman and Company, New York, 4th edition, 2000.

[145] F. Lonardo, K. H. Dragnev, S. J. Freemantle, Y. Ma, N. Memoli, D. Sekula, E. A. Knauth, J. S. Beebe, and E. Dmitrovsky. Evidence for the epidermal growth factor receptor as a target for lung cancer prevention. *Clin. Cancer Res.*, 8(1):54–60, 2002.

[146] K. A. Lund, L. K. Opresko, C. Starbuck, B. J. Walsh, and H. S. Wiley. Quantitative analysis of the endocytic system involved in hormone-induced receptor internalization. *J. Biol. Chem.*, 265(26):15713–15723, 1990.

[147] G. Maheshwari, A. Wells, L. G. Griffith, and D. A. Lauffenburger. Biophysical integration of effects of epidermal growth factor and fibronectin on fibroblast migration. *Biophysical Journal*, 76:2814–2823, 1999.

[148] K. Makino and M. Berz. Remainder differential algebras and their applications. Computational Differentiation: Techniques, Applications, and Tools, pages 63–74, Philadelphia, 1996. SIAM.

[149] R. März. On linear differential-algebraic equations and linearizations. *Applied Numerical Mathematics*, 18:267–292, 1995.

[150] A. Mattuck. *Introduction to Analysis*. Prentice-Hall, Inc., Upper Saddle River, NJ, 1999.

[151] H. H. McAdams and A. Arkin. Simulation of prokaryotic genetic circuits. *Annu. Rev. Biophys. Biomol. Struct.*, 27:199–224, 1998.

[152] H. H. McAdams and L. Shapiro. Circuit simulation of genetic networks. *Science*, 269:650–656, 1995.

[153] G. P. McCormick. Computability of global solutions to factorable nonconvex programs: Part 1 - Convex underestimating problems. *Mathematical Programming*, 10:147–175, 1976.

[154] G. P. McCormick. *Nonlinear programming: Theory, Algorithms and Applications.* John Wiley & Sons, 1983.

[155] H. P. McKean. Chapman–Enskog–Hilbert expansion for a class of solutions of the telegraph equation. *J. Math. Physics*, 8(3):547–551, 1967.

[156] H. Meinhardt. Orientation of chemotactic cells and growth cones: models and mechanisms. *J. Cell Sci.*, 112:2867–2874, 1999.

[157] T. Meyer and L. Stryer. Molecular model for receptor-stimulated calcium spiking. *Proc. Nat. Acad. Sci.*, 85:5051–5055, 1988.

[158] R. E. Moore. *Methods and Applications of Interval Analysis.* SIAM Studies in Applied Mathematics. SIAM, Philadelphia, 1979.

[159] N. S. Nedialkov. *Computing Rigorous Bounds on the Solution of an Initial Value Problem for an Ordinary Differential Equation.* PhD thesis, University of Toronto, 1999.

[160] Y. Nesterov. Complexity estimates of some cutting plane methods based on the analytic barrier. *Mathematical Programming*, 69:149–176, 1995.

[161] Y. Nesterov and A. Nemirovskii. *Interior-Point Polynomial Algorithms in Convex Programming.* SIAM, Philadelphia, 1994.

[162] C. D. Nobes and A. Hall. Rho GTPases control polarity, protrusion, and adhesion during cell movement. *J. Cell Biol.*, 144:1235–1244, 1999.

[163] W. Oettli and W. Prager. Compatibility of approximate solution of linear equations with given error bounds for coefficients and right-hand sides. *Numerische Mathematik*, 6:405–409, 1964.

[164] B. A. Ogunnaike and W. H. Ray. *Process Dynamics, Modeling and Control.* Oxford University Press, Oxford, 1994.

[165] H. G. Othmer, S. R. Dunbar, and W. Alt. Models of dispersal in biological systems. *J. Math. Biol.*, 26:263–298, 1988.

[166] S. P. Palecek, J. C. Loftus, M. H. Ginsberg, D. A. Lauffenburger, and A. F. Horwitz. Integrin-ligand binding properties govern cell migration speed through cell-substratum adhesiveness. *Nature*, 385:537–540, 1997.

[167] C. C. Pantelides. SpeedUp: recent advances in process simulation. *Computers Chem. Engng.*, 12(7), 1988.

[168] I. Papamichail and C. S. Adjiman. A global optimization algorithm for systems described by ordinary differential equations. Presented at AIChE Annual Meeting, Reno, Nevada, 2001.

[169] A. Papoulis. *Probability, Random Variables, and Stochastic Processes*. Electrical Engineering. Communications and Signal Processing. McGraw-Hill, Boston, Massachusetts, third edition, 1991.

[170] T. Park and P. I. Barton. State event location in differential-algebraic models. *ACM Transactions on Modelling and Computer Simulation*, 6(2):137–165, 1996.

[171] C. Peskin, G. Odell, and G. Oster. Cellular motion and thermal fluctuations: the brownian ratchet. *Biophys. J.*, 65:316–324, 1993.

[172] E. Piccolo, P. Innominato, M. A. Mariggio, T. Maffucci, S. Iacobelli, and M. Falasca. The mechanism involved in the regulation of phospholipase c$\gamma$1 activity in cell migration. *Oncogene*, 21:6520–6529, 2002.

[173] L. S. Pontryagin. *Ordinary Differential Equations*, pages 170–180. Addison-Wesley, Reading, MA, 1962. Translated from the Russian by Leonas Kacinskas and Walter B. Counts.

[174] I. Posner, M. Engel, and A. Levitzki. Kinetic model of the epidermal growth factor (EGF) receptor tyrosine kinase and a possible mechanism of its activation by EGF*. *J. Biol. Chem.*, 267(29):20638–20647, 1992.

[175] D. A. Potter, J. S. Tirnauer, R. Janssen, D. E. Croall, C. N. Hughes, K. A. Fiacco, J. W. Mier, M. Maki, and I. M. Herman. Calpan regulates actin remodeling during cell spreading. *J. Cell Biol.*, 141(3), 647–662.

[176] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in Fortran: The Art of Scientific Computing.* Cambridge University Press, 2nd edition, 1992.

[177] L. A. Puto, K. Pestonjamasp, C. C. King, and G. M. Bokoch. p21-activated kinases 1 (PAK1) interacts with the Grb2 adapter protein to couple to growth factor signaling. *J. Biol. Chem.*, 278(11):9388–9393, 2003.

[178] R. E. Quandt and J. B. Ramsey. Estimating mixtures of normal distributions and switching regressions. *J. of the American Statistical Association*, 73(364): 730–738, 1978.

[179] C. C. Reddy, S. K. Niyogi, A. Wells, H. S. Wiley, and D. A. Lauffenburger. Engineering epidermal growth factor for enhanced mitogenic potency. *Nature Biotech.*, 14:1696–1699, 1996.

[180] C. C. Reddy, A. Wells, and D. A. Lauffenburger. Receptor-mediated effects on ligand availability influence relative mitogenic potencies of epidermal growth factor and transforming growth factor $\alpha$. *J. Cell. Physiol.*, 166:512–522, 1996.

[181] C. C. Reddy, A. Wells, and D. A. Lauffenburger. Comparitive mitogenic potencies of EGF and TGF$\alpha$ and their dependence on receptor-limitation versus ligand-limitation. *Med. Biol. Eng. Comp.*, 36:499–507, 1998.

[182] S. Reich. On the local qualitative behavior of differential-algebraic equations. *Circuit Systems Signal Process*, 14(4):427–443, 1995.

[183] K. J. Reinschke. Multivariable control: A graph-theoretic approach. volume 108 of *Lecture Notes in Control and Information Sciences.* Springer Verlag, New York, 1988.

[184] A. J. Ridley, H. F. Paterson, C. L. Johnston, D. Diekmann, and A. Hall. The small GTP-binding protein rac regulates growth factor-induced membrane ruffling. *Cell*, 70:401–410, 1992.

[185] D. J. Riese II and D. F. Stern. Specificity within the EGF family/erbB receptor family signaling network. *BioEssays*, 20:41–48, 1998.

[186] J. Rossant and L. Howard. Signaling pathways in vascular development. *Annu. Rev. Cell Dev. Biol.*, 18:541–573, 2002.

[187] P. Roy, W. M. Petroll, C. J. Chuong, H. D. Cavanagh, and J. V. Jester. Effect of cell migration on the maintenance of tension on a collagen matrix. *Ann. Biomed. Eng.*, 27:721–730, 1999.

[188] W. Rudin. *Principles of Mathematical Analysis.* McGraw-Hill, New York, 3rd edition, 1976.

[189] A. Ruhe. Rational Krylov: A practical algorithm for large sparse nonsymmetric matrix pencils. *SIAM J. Sci. Comput.*, 19(5):1535–1551, 1998.

[190] H. S. Ryoo and N. V. Sahinidis. Global optimization of nonconvex NLPs and MINLPs with applications in process design. *Computers & Chemical Engineering*, 19(5):551–566, 1995.

[191] H. S. Ryoo and N. V. Sahinidis. A branch-and-reduce approach to global optimization. *Journal of Global Optimization*, 8(2):107–139, 1996.

[192] Y. Saad. Numerical solution of large nonsymmetric eigenvalue problems. *Computer Physics Communications*, 53:71–90, 1989.

[193] Y. Saad. *Numerical Methods for Large Eigenvalue Problems.* Manchester University Press, Manchester, England, 1992.

[194] K. Sachs, D. Gifford, T. Jaakkola, P. Sorger, and D. A. Lauffenburger. Bayesian network approach to cell signaling pathway modeling. *Sci. STKE*, 148:pe38, 2002.

[195] R. A. Sack. A modification of Smoluchowski's diffusion equation. *Physica*, 22: 917–918, 1956.

[196] J. F. Sah, R. L. Eckert, R. A. Chandraratna, and E. A. Rorke. Retinoids suppress epidermal growth factor-associated cell proliferation by inhibiting epidermal growth factor receptor-dependent ERK1/2 activation. *J. Biol. Chem.*, 277(12):9728–9735, 2002.

[197] C. A. Sarkar and D. A. Lauffenburger. Cell-level pharmacokinetic model of granulocyte colony-stimulating factor: Implications for ligand lifetime and potency in vivo. *Mol. Pharmacol.*, 63:147–158, 2003.

[198] B. Schoeberl, C. Eichler-Jonsson, E. D. Gilles, and G. Müller. Computational modeling of the dynamics of the MAP kinase cascade activated by surface and internalized EGF receptors. *Nat. Biotechnol.*, 20:370–375, 2002.

[199] J. A. Scott. An Arnoldi code for computing selected eigenvalues of sparse, real, unsymmetric matrices. *ACM Transactions on Mathematical Software*, 21(4): 432–475, 1995.

[200] C. E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27:379–423, 623–656, 1948.

[201] M. P. Sheetz, D. B. Wayne, and A. L. Pearlman. Extension of filopodia by motor-dependent actin assembly. *Cell Motil. Cytoskeleton*, 22:160–169, 1992.

[202] N. Z. Shor. *Minimization Methods for Non-Differentiable Functions*, volume 3 of *Springer Series in Computational Mathematics*. Springer-Verlag, 1985.

[203] A. B. Singer and P. I. Barton. Global optimization with nonlinear ordinary differential equations. Manuscript in preparation, 2003.

[204] A. B. Singer and P. I. Barton. Global solution of linear dynamic embedded optimization problems. In press, *J. Optimization Theory and Applications*, 2003.

[205] D. S. Sivia. *Data Analysis: A Bayesian Tutorial*. Clarendon Press, Oxford, 1996.

[206] D. S. Sivia and C. J. Carlile. Molecular spectroscopy and Bayesian spectral analysis - how many lines are there? *J. Chem. Phys.*, 96(1):170–178, 1992.

[207] R. D. Skeel. Scaling for numerical stability in Gaussian elimination. *J. Assoc. Comp. Mach.*, 26(3):494–526, 1979.

[208] R. D. Skeel. Iterative refinement implies numerical stability for Gaussian elimination. *Mathematics of Computation*, 35(151):817–832, 1980.

[209] B. A. Skierczynski, S. Usami, and R. Skalak. A model of leukocyte migration through solid tissue. In *Cell Biol.*, volume 84 of *ser. H*, pages 285–328. NATO ASI (Adv. Sci. Inst.), 1994.

[210] S. Skogestad and I. Postlethwaite. *Multivariable Feedback Control.* John Wiley & Sons, New York, 1997.

[211] D. J. Slamon, G. M. Clark, S. G. Wong, W. J. Levin, A. Ullrich, and W. L. McGuire. Human breast cancer: Correlation of relapse and survival with amplification of the HER-2/*neu* oncogene. *Science*, 235:177–182, 1987.

[212] E. M. d. B. Smith. *On the Optimal Design of Continuous Processes.* PhD thesis, Imperial College of Science, Technology, and Medicine, 1996.

[213] M. v. Smoluchowski. Zur kinetischen Theorie der Brownschen molekularbewegung und der suspensionen. *Ann. Phys.*, 21:756–780, 1906.

[214] R. M. Soland. An algorithm for separable nonconvex programming problems II: Nonconvex constraints. *Management Science*, 17(11):759–773, 1971.

[215] D. R. Soll. The use of computers in understanding how animal cells crawl. *International Review of Cytology*, 163:43–104, 1995.

[216] C. Starbuck, H. S. Wiley, and D. A. Lauffenburger. Epidermal growth factor binding and trafficking dynamics in fibroblasts: Relationship to cell proliferation. *Chem. Eng. Sci.*, 45(8):2367–2373, 1990.

[217] J. P. Steinbach, P. Supra, H. J. Huang, W. K. Cavenee, and M. Weller. CD95-mediated apoptosis of human glioma cells: modulation by epidermal growth factor receptor activity. *Brain Pathol.*, 12(1):12–20, 2002.

[218] P. J. Steinbach, K. Chu, H. Frauenfelder, J. B. Johnson, D. C. Lamb, G. U. Nienhaus, T. B. Sauke, and R. D. Young. Determination of rate distributions from kinetic experiments. *Biophys. J.*, 61:235–245, 1992.

[219] G. W. Stewart and J. Sun. Matrix perturbation theory. Computer Science and Scientific Computing. Academic Press, San Diego, 1990.

[220] N. F. Stewart. A heuristic to reduce the wrapping effect in the numerical solution of $x' = f(t, x)$. *BIT*, 11:328–337, 1971.

[221] W. E. Stewart, Y. Shon, and G. E. P. Box. Discrimination and goodness of fit of multiresponse mechanistic models. *AIChE J.*, 44(6):1404–1412, 1998.

[222] M. H. Symons and T. J. Mitchinson. Control of actin polymerization in live and permeabilized fibroblasts. *J. Cell Biol.*, 114:503–513, 1991.

[223] R. Tarjan. Depth-first search and linear graph algorithms. *SIAM J. Comput.*, 1(2):146–160, 1972.

[224] M. Tawarmalani and N. V. Sahinidis. Global optimization of mixed integer nonlinear programs: A theoretical and computational study. URL `http://archimedes.scs.uiuc.edu/group/publications.html`. Submitted to Mathematical Programming, 1999.

[225] G. I. Taylor. Diffusion by continuous movements. *Proc. London Math. Soc.*, 20: 196–212, 1921–22.

[226] M. E. Tipping and C. M. Bishop. Probabilistic principal component analysis. *J. R. Statist. Soc. B*, 61(3):611–622, 1999.

[227] I. B. Tjoa and L. T. Biegler. Simultaneous solution and optimization strategies for parameter-estimation of differential-algebraic equation systems. *Ind. Eng. Chem. Res.*, 30(2):376–385, 1991.

[228] J. Tolsma and P. I. Barton. DAEPACK: An open modeling environment for legacy models. *Ind. Eng. Chem. Res.*, 39(6):1826–1839, 2000.

[229] J. E. Tolsma and P. I. Barton. Hidden discontinuities and parametric sensitivity calculations. *SIAM J. Sci. Comput.*, 23(6):1862–1875, 2002.

[230] J. E. Tolsma, J. A. Clabaugh, and P. I. Barton. Symbolic incorporation of external procedures into process modeling environments. *Ind. Eng. Chem. Res.*, 41(16):3867–3876, 2002.

[231] R. T. Tranquillo and D. A. Lauffenburger. Stochastic model of leukocyte chemosensory movement. *J. Math. Biol.*, 25:229–262, 1987.

[232] R. T. Tranquillo, D. A. Lauffenburger, and S. H. Zigmond. A stochastic model of leukocyte random motility and chemotaxis based on receptor binding fluctuations. *J. Cell Biol.*, 106:303–309, 1988.

[233] J. P. Trinkhaus. *Cells Into Organs: The Forces that Shape the Embryo*. Prentice-Hall, Englewood Cliffs, NJ, 1984.

[234] P. van der Greer, T. Hunter, and R. A. Lindberg. Receptor protein-tyrosine kinases and their signal transduction pathways. *Annu. Rev. Cell Biol.*, 10: 251–337, 1994.

[235] L. Vandenberghe and S. Boyd. Semidefinite programming. *SIAM Review*, 38 (1):49–95, 1996.

[236] W. Walter. *Differential and Integral Inequalities*. Springer-Verlag, Berlin, 1970.

[237] M. F. Ware, A. Wells, and D. A. Lauffenburger. Epidermal growth factors alters fibroblast migration speed and directional persistence reciprocally and in a matrix-dependent manner. *J. Cell Sci.*, 111:2423–2432, 1998.

[238] B. Wehrle-Haller and B. A. Imhof. Actin, microtubules and focal adhesion dynamics during cell migration. *Int. J. Biochem. and Cell Biol.*, 35:39–50, 2003.

[239] A. Wells, M. F. Ware, F. D. Allen, and D. A. Lauffenburger. Shaping up for shipping out: PLC$\gamma$ signaling of morphology changes in EGF-stimulated fibroblast migration. *Cell Motil. and Cytoskeleton*, 44:227–233, 1999.

[240] M. P. F. Wong. *Assessment of controllability of chemical processes.* PhD thesis, University of London, 1985.

[241] M. A. Woodrow, D. Woods, H. M. Cherwinski, D. Stokoe, and M. McMahon. Ras-induced serine phosphorylation of the focal adhesion protein paxillin is mediated by the Raf$\rightarrow$MEK$\rightarrow$ERK pathway. *Exper. Cell Res.*, 287:325–338, 2003.

[242] H. Xie, M. A. Pallero, K. Gupta, P. Chang, M. F. Ware, W. Witke, D. J. Kwiatkowski, D. A. Lauffenburger, J. E. Murphy-Ullrich, and A. Wells. EGF receptor regulation of cell motility: EGF induces disassembly of focal adhesions independently of the motility-associated PLC$\gamma$ signaling pathway. *J. Cell Sci.*, 111:615–624, 1998.

[243] L. A. Zadeh and C. A. Desoer. *Linear System Theory: The State Space Approach.* Robert E. Krieger, Huntington, New York, 1979.

[244] A. Zellner. *An Introduction to Bayesian Inference in Econometrics.* John Wiley & Sons, New York, 1971.