Bayesian Inference and Information Learning for Switching Nonlinear Gene Regulatory Networks

by

Nayely L. Vélez-Cruz

A Dissertation Presented in Partial Fulfillment
of the Requirement for the Degree
Doctor of Philosophy

Approved September 2023 by the
Graduate Supervisory Committee:

Antonia Papandreou-Suppappola, Chair
Bahman Moraffah
Visar Berisha
Cihan Tepedelenlioglu

ARIZONA STATE UNIVERSITY

December 2023

ABSTRACT

This dissertation centers on the development of Bayesian methods for learning different types of variation in switching nonlinear gene regulatory networks (GRNs). A new nonlinear and dynamic multivariate GRN model is introduced to account for different sources of variability in GRNs. The new model is aimed at more precisely capturing the complexity of GRN interactions through the introduction of time-varying kinetic order parameters, while allowing for variability in multiple model parameters. This model is used as the drift function in the development of several stochastic GRN models based on Langevin dynamics. Six models are introduced which capture intrinsic and extrinsic noise in GRNs, thereby providing a full characterization of a stochastic regulatory system. A Bayesian hierarchical approach is developed for learning the Langevin model which best describes the noise dynamics at each time step. The trajectory of the state, which are the gene expression values, as well as the indicator corresponding to the correct noise model are estimated via sequential Monte Carlo (SMC) with a high degree of accuracy. To address the problem of time-varying regulatory interactions, a Bayesian hierarchical model is introduced for learning variation in switching GRN architectures with unknown measurement noise covariance. The trajectory of the state and the indicator corresponding to the network configuration at each time point are estimated using SMC. This work is extended to a fully Bayesian hierarchical model to account for uncertainty in the process noise covariance associated with each network architecture. An SMC algorithm with local Gibbs sampling is developed to estimate the trajectory of the state and the indicator corresponding to the network configuration at each time point with a high degree of accuracy. The results demonstrate the efficacy of Bayesian methods for learning information in switching nonlinear GRNs.

DEDICATION

To my parents, Tyler, and my best friends. Thank you. I love you all.

ACKNOWLEDGEMENTS

I would like to thank the following individuals, without whom this work would not have been possible. My PhD advisor, Dr. Antonia Papandreou-Suppappola, for her unwavering support, patience, and dedication to my success; my other PhD advisor, Dr. Manfred Laubichler, for allowing me and encouraging me to take this detour so that I may improve my technical skills; my parents, Maria Luz Cruz-Torres and Carlos Vélez-Ibañez, for instilling a love of learning in me at a young age; my partner, Tyler, for all of the love and laughter throughout this difficult process; my best friends, Kasi, Natalie, and Tyler Q for making this the most fun and unforgettable graduate school experience; Mr. and Mrs. Demirjian, for treating me like their own daughter and for making the most delicious food; Diane Murdock, for also making delicious food; Jason Yalim at ASU Research Computing for taking plenty of time to teach me how to use the ASU super computer and ensuring that my simulations run successfully; my PhD committee members, Dr. Bahman Moraffah, Dr. Visar Berisha, and Dr. Cihan Tepedelenlioglu for their helpful feedback; Lynn Pratte, my academic advisor; and finally, the School of Electrical, Energy and Computer Engineering at ASU for accepting me into the PhD program.

TABLE OF CONTENTS

CHAPTER Page

CHAPTER Page

CHAPTER Page

LIST OF TABLES

Table Page

Table Page

Table Page

LIST OF FIGURES

Figure Page

Figure Page

Chapter 1

INTRODUCTION

1.1 Motivation: Development, Evolution, and Disease in Gene Regulatory Networks

The process of organismal development is mediated by the action of gene regulatory networks (GRNs) de Leon and Davidson (2007). These networks are complex systems consisting of genes, transcription factors (TFs) and a toolkit of regulatory elements responsible for the spatiotemporal allocation of gene expression in every cell of the organism during embryogenesis Peter and Davidson (2011). The specific topology and dynamics of GRNs determine the organization of the anatomical plan and its functional properties by creating hundreds of different cell types and integrating them into a functional whole: the organism, which is the output of the developmental process. GRNs therefore provide a mechanistic explanation for how a one-dimensional deoxyribonucleic acid (DNA) sequence is transformed into a dynamic three-dimensional structure. Their topology and dynamics serve as the mapping function between genotype, the DNA sequence, and phenotype, which broadly refers to the characters, traits, and three-dimensional morphologies of the organism.

Over evolutionary time, changes in the topology and dynamics of GRNs are consequential to the final output of the developmental process. This can occur via novel gene-transcription factor associations, gene duplication and co-option, novel interactions between non-TF and regulatory elements such as long non-coding RNAs (lncRNAs) and additional transcriptional co-factors. Changes can also be induced by the environment Sebe-Pedros *et al.* (2017). These changes in the developmental process

generate alternate developmental trajectories and may produce morphological novelties as a direct result of changes in gene expression patterns. Thus, studying how GRNs change across *evolutionary* timescales can provide a mechanistic explanation of phenotypic evolution. This also answers one of Darwin's fundamental questions from *The Origin of Species* Darwin (1909): what are the causes of phenotypic novelty? Or, in other words, how does evolution produce something new? GRNs are essential for gaining insight into the mechanisms underlying major phenotypic innovations (changes in the body plan).

On shorter timescales, changes in the topology of GRNs correspond to different stages of biological processes. These can include stages in development, the progression of diseases, such as cancer, and stages in the life cycle of an organism. Thus, understanding the structure and dynamics of such regulatory networks, as well as how they vary over time, can yield insights into diseases and developmental disorders resulting from regulatory mechanisms gone amok. Therefore, the importance of understanding GRNs spans every biological discipline.

1.2 Gene Regulatory Network Modeling and Reverse Engineering

1.2.1 GRN Modeling

The specific dynamics of gene regulation can be described by Boolean or differential equations models. Boolean models were introduced in 1969 by Stuart Kauffman to describe the discrete-time dynamics of regulatory systems Kauffman (1969). In these models, the state of a gene, X_i takes on a binary value $X_i \in \{0, 1\}$, which corresponds to an on or off state. A significant advantage of Boolean networks is that they are able to account for gene expression due to the combinatorial action of transcription factors on a target gene Akers and Murali (2021). Differential equations

models describe the rate of change of gene expression levels. The rate of change is a function of the genes themselves as well as additional parameters which encode the interactions between genes and the regulatory dynamics of other genes in the network. The primary advantage of differential equations models is that they provide a mechanistic system representation of biochemical processes by capturing the nonlinear dynamics of gene regulation. Both of these approaches can be incorporated into a state space model representation, which facilitates inference of the network using noisy microrarray data. GRN inference using a state space approach has been extensively studied Amor *et al.* (2019); Ancherbak *et al.* (2016a); Bugallo *et al.* (2015); de Luis Balaguer and Sozzani (2017); Elahi and Hasan (2018); Noor *et al.* (2012); Mercatelli *et al.* (2020); Pirgazi and Khanteymoori (2018); Sanguinetti and Huynh-Thu (2019); Santillán (2008); Shen and Vikalo (2010); Wai *et al.* (2019); Wang *et al.* (2009, 2007); Wang and Aberra (2015); Xiong and Zhou (2013); Youseph *et al.* (2015, 2019); Zhang *et al.* (2014); Zhou and Ji (2017). Specifically, the authors in Pirgazi and Khanteymoori (2018); Xiong and Zhou (2013) use Kalman filtering for inference by assuming a linear state-space model. In Bugallo *et al.* (2015); Noor *et al.* (2012); Wai *et al.* (2019); Wang *et al.* (2009); Zhou and Ji (2017); Zhang *et al.* (2014), the nonlinear sigmoid squash function is used since it can capture the switch-like behavior of regulatory systems. Alternatively, nonlinear system models based on Hill kinetics, Michaelis-Menten kinetics, or the S-system are able to more accurately capture the molecular mechanisms of gene regulation Elahi and Hasan (2018); Wang *et al.* (2007); Youseph *et al.* (2015, 2019). Nonlinear Bayesian filtering inference methods, such as extended Kalman filtering and particle filtering, were used with these nonlinear models Bugallo *et al.* (2015); Wang *et al.* (2009); Zhou and Ji (2017); Zhang *et al.* (2014). A key challenge in reverse engineering time-varying GRNs is the development of models that can capture both the direction and type of regulation (activation or

inhibition) in order to infer the gene expression trajectories. To this extent well as ODE and SDE-based methods, are able to capture both the direction and regulation type Nguyen *et al.* (2021); Lim *et al.* (2016); Moignard *et al.* (2015). An additional challenge in reverse engineering time-varying GRNs is the development of models that can capture the various types of noise that are present in GRNs. Not only are the microarray data noisy, but GRN dynamics are inherently stochastic processes. Such stochasticity is a result of varying cellular environments, varying timings of molecular events, and low copy numbers of genes inside the cell Wang and Aberra (2015). These factors give rise to stochastic fluctuations in the process of gene expression, from transcription to translation, and contribute to phenotypic variation. In general, this stochasticity can be partitioned into *intrinsic* and *extrinsic* noise and can affect production and degradation rates. Most state-space approaches for inferring GRNs exclusively assume additive Gaussian noise in both the process and measurement noise. However, it is usually unknown a priori which noise dynamics best describe the system dynamics in the process model. Furthermore, the type of noise that is present may change over time.

1.2.2 Reverse Engineering GRNs

Over the past few decades, developments in high-throughput sequencing and microarray technologies have facilitated the acquisition of tremendous amounts of -omics data, of which there are four main types: genomics, proteomics, transcriptomics, and metabolomics Kaur *et al.* (2021). Such data are used for inferring and reconstructing GRNs Delgado and Gómez-Vela (2019), a task referred to as *reverse engineering*. Reverse engineering GRNs consists of three main steps. First, gene expression in terms of mRNA or protein concentrations is measured under different experimental conditions following perturbation of the target genes. The resulting datasets encode

topological information of the underlying GRN, which is called the *target network*. Second, computational algorithms which usually make use of a model for the network are applied to the data to infer the structure of the network as well as any additional model parameters. The output of this step is the *predicted network*. Third, the predicted network is compared to the target network according to some validation criterion.

Approaches to reverse engineering GRNs can be broadly categorized into two types: static and dynamic. Static approaches use steady-state data to construct a GRN as a temporal aggregate represented by a static network. This class of approaches is largely comprised of co-expression networks, information theoretic approaches, and probabilistic graphical models. Co-expression networks use correlation metrics, such as Pearson correlation to establish regulatory relationships, whereas information[theoretic approaches use mutual information to establish nonlinear regulatory relationships Nguyen *et al.* (2021). Probabilistic graphical models, such as Bayesian networks and Gaussian graphical models (GGMs), encode conditional dependencies between genes to form the network. Static approaches do not represent information related to time and thus cannot capture time-varying regulatory relationships Lopes and Bontempi (2013). Dynamic approaches overcome this drawback by accounting for the dynamic aspects of gene regulation through the use of time series data. They are useful for inferring time-varying GRNs or GRNs under different physiological or environmental conditions Stumpf (2021). Many of these approaches are model-based and employ ordinary differential equations (ODEs), stochastic differential equations (SDEs), or Boolean models. Other such approaches include dynamic Bayesian networks and vector autoregressive models.

1.3 Contributions

As highlighted in Section 1.2.1, the main three challenges in reverse engineering GRN are as follows: (a) the design of dynamic stochastic models to capture sources of variability in GRNs while simultaneously accounting for both the direction and type of gene regulation; (b) inference methods to account for dynamic variability in the type and direction of gene regulation, which yields different GRN architectures; and (c) inference methods to account for different types of noise dynamics that best describe the behavior of the regulatory system. Our contributions in this dissertation address each of these three challenges.

1.3.1 Contribution 1: New Nonlinear Dynamic and Stochastic GRN Models

We propose a new nonlinear and dynamic multivariate GRN model that integrates both Michaelis-Menten and Hill kinetics and allows us to incorporate stochasticity to account for different sources of variability in GRNs. The model is based on Youseph *et al.* (2019) which contains parameters encoding the type and direction of regulation. The new model is multivariate extension aimed at more precisely capturing the complexity of GRN interactions through the introduction of time-varying kinetic order parameters, while allowing for variability in multiple model parameters Vélez-Cruz *et al.* (2021). This model is used as the drift function in our development of several stochastic GRN models based on Langevin dynamics. We introduce six models which capture intrinsic and extrinsic noise in GRNs, thereby providing a full characterization of a stochastic regulatory system.

1.3.2 *Contribution 2: Learning Variation in Switching Langevin Dynamics*

Gene regulation is an inherently stochastic process, and this stochasticity can be partitioned into various types depending on their source. To the authors knowledge, there is no work that focuses on Bayesian model selection of stochastic models of gene regulatory networks. We assume that the noise dynamics can change at unknown times in the GRN to account for the various sources of stochasticity. We introduce hierarchical Bayesian approach which can learn the Langevin model which best describes the noise dynamics at each time step. The unknown noise model is learned by drawing parameters from a categorical distribution with probabilities distributed according to a Dirichlet conjugate prior. The unknown measurement noise covariance is learned using an Inverse-Wishart prior with known hyerparameters. We estimate the gene expression values as well as the indicator corresponding to the noise type at each time point. Our results demonstrate the efficacy of Bayesian hierarchical modeling under varying noise types.

1.3.3 *Contribution 3: Learning Variation in Gene Regulatory Network Architecture*

In order to account for variability in the regulatory interactions, we assume that the kinetic order parameters can change at unknown times in the GRN. We introduce a hierarchical Bayesian approach that can learn the network configuration which best describes the gene expression dynamics. The unknown transition probabilities are learned by drawing parameters from a categorical distribution. The parameters of the categorical distribution are then learned using a Dirichlet distribution conjugate prior. The unknown measurement noise is learned using an Inverse-Wishart prior with known hyperparameters. We estimate the trajectory of the state, which are the gene expression values, as well as the indicator corresponding to the network

configuration at each time point. We demonstrate through simulations the efficacy of our Bayesian hierarchical modeling approach.

1.3.4 Contribution 4: Learning Variation in Gene Regulatory Network Architecture with Local Gibbs Sampling

In order to account for variability in the regulatory interactions, we assume that the kinetic order parameters can change at unknown times in the GRN. We introduce a fully hierarchical Bayesian approach that can learn the network configuration which best describes the gene expression dynamics. The unknown transition probabilities over multiple sets of kinetic order parameters are learned by drawing parameters from a categorical distribution. The parameters of the categorical distribution are then learned using a Dirichlet distribution conjugate prior. The unknown measurement and process noise covariances are learned using Inverse-Wishart priors with known hyperparameters. We develop a sequential Monte Carlo (SMC) algorithm with local Gibbs sampling to estimate the trajectory of the state and the indicator corresponding to the network configuration at each time point. We demonstrate through simulations the efficacy of our Bayesian hierarchical modeling approach.

1.4 Organization

This dissertation is organized as follows. In Chapter 2, we provide an introduction to gene regulatory networks, specifically focusing on their role in development, disease, and evolution. In Chapter 3, we review Bayesian inference methods for state-space models. In Chapter 4, we review a broad range of current approaches to processing gene regulatory networks. In Chapter 5, we introduce the new Michaelis-Menten kinetics model, which can be used to infer time-varying GRNs. We also develop several stochastic models of gene regulation aimed at more precisely capturing the complexity

of GRNs. In Chapter 6, we focus on the problem of estimating GRNs under switching noise dynamics. In Chapter 7, we introduce a Bayesian model for estimating time-varying GRNs with switching architectures. In Chapter 8, we extend the Bayesian model in Chapter 7 to a fully Bayesian hierarchical model, which incorporates uncertainty in both the process and measurement noise covariance matrices. Since the unknown state depends on the unknown process noise covariance, we implement a local Gibbs step in our SMC algorithm. In Chapter 9, we conclude our work through a discussion of next steps and future directions.

Chapter 2

CONCEPTUAL BACKGROUND

How do you build an organism? How do you go from a single cell, the egg, to an integrated collection of hundreds of different cell types? How do you build arms, legs, a head, let alone ensure that they are in the correct spots? How do complex diseases, such as cancer, arise? What is the best way to develop drug therapies that take into account the inter-individual differences in how cancers form? How did the transition from unicellular to multicellular organisms occur, and subsequently the origin of animals? Each of these questions can be answered in large by gene regulatory networks (GRNs), which are comprised of regulatory genes and their interactions with molecular entities. Such entities include transcription factors and their co-factors, long non-coding RNAs (ribonucleic acid), histones, etc. The list has grown since the dawn of molecular biology, and we are still unveiling the complexity of the genome and its regulatory elements to this day.

Regulatory genes, as their name suggests, encode proteins that regulate the expression of structural genes; these are genes that encode proteins needed for physical structures inside the cells or functional proteins Peter and Davidson (2015). The interactions between regulatory genes and the genes they regulate form vast networks inside each cell that are also able to regulate gene expression in neighboring cells. What was, for many decades, viewed as a static sequence of letters is in reality a dynamic, highly complex four-dimensional structure that exhibits a spatiotemporal hierarchical organization. It is these regulatory networks that provide a causal link, or mapping function, between genotype (DNA sequence) and phenotype (trait). The theoretical advent of gene regulatory networks, and their subsequent experimental

validation by Eric Davidson and Roy J. Britten in the 1960s, contributed to challenging the view of a direct mapping between sequence and trait, or in other words, a direct genotype-phenotype map, by providing a mechanistic explanation of the causal relationship between genome and trait [1] . Interestingly, Davidson drew inspiration from electrical engineering, specifically circuit diagrams to infer, visualize, and model the dynamics of gene regulatory networks.

2.1 GRN Structure

GRNs, and the entire process of organismal development, are encoded in the genome and specifically in cis-regulatory modules (the interactions between regulatory genes and the toolkit of molecular entities) Peter (2019). Cis-regulatory modules are sequences of non-coding DNA, where transcription factors bind to that are required for the activity of most or all genes. They are control modules responsible for the activation or repression of regulatory genes. Types of cis-regulatory elements include promoters, enhancers, and silencers which combinatorially bind transcription factors to form modules Peter and Davidson (2015). Promoters are sequences of DNA to which RNA polymerase initiates the transcription of a gene at the transcription start site (TSS). They are typically located directly upstream of the TSS **?**. Enhancers are sequences of DNA which bind clusters of transcription factors that cause increased expression of their target gene. They can be located upstream, downstream, as well as within the gene they regulate. They can regulate gene expression over very large distances by looping the transcription factors bound to them to contact the promoter; enhancers can also be found close to their target promoter Hardison and Taylor (2012).

[1] It is important to note that the idea of a direct genotype-phenotype (G-P) map was challenged early on in the history of genetics, but Davidson and Britten's work further demonstrated why the idea of a direct G-P map was not much more than a conceptual fantasy.

Silencers, on the other hand, as their name suggests, are DNA sequences that cause reduced expression of their target gene.

Each cell in the body has the same DNA. One key feature of GRNs is that using the same set of genes, GRNs are able to produce hundreds of distinct cell types. This is done by turning on and turning off different combinations of genes within each cell, which is accomplished through the combinatorial action of transcription factors and the cis-regulatory modules they form with promoters, enhancer, and silencers. Different combinations of transcription factors will regulate the genome of each cell in different ways. This is because the specific combination of those transcription factors will cause either the activation or repression of cis-regulatory modules found on certain genes in a specific cell, thereby producing different *specification states*. *Specification states* are cell type-specific *regulatory states* that define the cell identity and the differentiation genes that it expresses de Leon and Davidson (2007). The *regulatory state* is the total set of activate transcription factors in a cell nucleus at a given time and domain of the embryo de Leon and Davidson (2007).

However, in order for RNA polymerase and for transcription factors to bind to a gene to begin transcription, the DNA must be accessible. DNA is wrapped around proteins called histones to form chromatin, which can either be loosely condensed (euchromatin), or tightly condensed (heterochromatin). Typically, genes within heterochromatin are not expressed because they are not accessible. Modifications to the histones and to the DNA can be made to alter chromatin structure. One such modification is histone acetylation, in which acetyl groups added to the histones cause heterochromatin to unwind, transforming it into euchromatin and activating gene expression. When the acetyl groups are removed by certain enzymes called histone deacetylases, the DNA condenses once again to form heterochromatin, blocking transcription and gene expression. The other type of chromatin modification is DNA

methylation, which adds methyl groups directly to the DNA, not the histone proteins, and causes chromatin to condense into heterochromatin, turning off gene expression. Thus, the organization of DNA into heterochromatin or euchromatin also contributes to differential gene expression.

There are many more subtleties to the organization of the genome and the dynamics of gene regulation that are beyond the scope of this work, such as nucleosome-nucleosome interactions, chromatin looping, and the organization of the genome into topologically associating domains (TADs) and subTADs, to list a few Bonev and Cavalli (2016). For the sake of coarse graining and for developing a useful modeling framework that can aid in the reconstruction of regulatory networks using time series data, we focus on the interactions between transcription factors and regulatory genes.

2.1.1 The Role of GRN Subcircuits

A second key feature of GRNs is that they are hierarchical both spatially and temporally. In this section, we focus on the spatial hierarchy of GRNs. GRNs consist of subcircuits, which are assemblages of two to eight genes that form specific regulatory linkages among specific genes to perform biologically meaningful functions Peter and Davidson (2011). Subcircuits, not the individual regulatory genes, are the structural and functional subunits of GRNs Hinman and Cheatle Jarvela (2014). The advantage of this hierarchical organization into subcircuits is that it provides GRNs with modularity, a characteristic of both engineered systems and natural systems which have undergone evolutionary change.

From a network science perspective, modularity is an efficient feature of many complex networks. To understand why, it is important to note that there are two types of modularity: structural and functional modularity. Structural modularity refers to spatially bounded groups of highly-connected nodes that are sparsely con-

nected to other nodes Hartwell *et al.* (1999). These modules can be separated into physically independent components. The obvious advantage of structural modularity is that it reduces the cost of communication between nodes of a network through physical integration of various components Clune *et al.* (2013). Structural modularity also makes complex networks robust to attacks, since damage can be localized to specific modules without affecting the rest of the network. Functional modules, on the other hand, typically consist of different components that interact with one another to accomplish a specific function, and this function is separable from other modules Hartwell *et al.* (1999); Lacquaniti *et al.* (2013). Unlike structural modules, the components of a functional module do not have to be spatially bounded. Functional modularity also contributes to the efficiency of complex networks in that it allows for a division of labor so that not all parts of the network are required to perform every task.

In the developmental process, GRN subcircuits can exhibit both of these types of modularity. GRN subcircuits are always functionally modular since each subcircuit type is responsible for performing a particular developmental task. They are also structurally modular through the direct physical interactions between the transcription factors (TFs) and genes that comprise individual subcircuits regardless of the actual distance between a TF and the gene that it binds to since the action of distal enhancers can bring these spatially separated elements into direct physical contact (chromatin looping being one mechanism). The defining characteristic of GRN subcircuits is that their spatial organization (physical interactions between components of the subcircuits) completely determines their developmental function. As a result, their structural and functional modularities are inseparable much in the same way as electrical circuits. So far, eight canonical subcircuits have been identified in GRNs across the diversity of Metazoa Peter and Davidson (2015). We will briefly describe

each one as they are listed in Peter and Davidson's *Genomic Control Processes: Development and Evolution* Peter and Davidson (2015):

1. *Positive feedback* subcircuits consist of two to three genes which activate their own or another regulatory gene's transcription at the cis-regulatory level. Although GRNs do have positive autoregulatory feedback loops, in which a gene product activates its own transcription, multigenic positive feedback loops are much more common in GRNs than are positive autoregulatory feedback loops. One of the most interesting features of positive feedback subcircuits is that they act as filters to reduce the noise caused by the duration or intensity of upstream transient inputs, such as brief exposure to signals or brief exposure to transcription factors, converting these signals into stable regulatory states. A signal can lock a two-gene positive feedback subcircuit into two possible stable steady states. If the signal causes the protein from gene X or Y to be produced, the subcircuit locks in the ON state. This is demonstrated in Figure 2.1. The other alternative is that no activation of X or Y occur, so genes X and Y are both OFF.

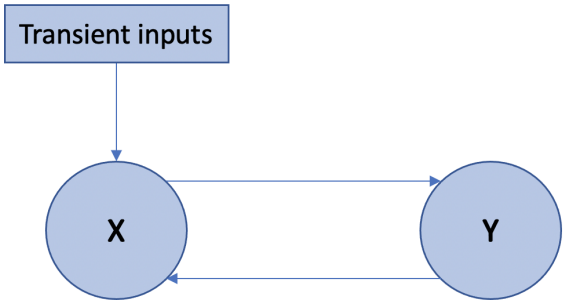


Figure 2.1: **Positive feedback subcircuit**. Gene X activates the expression of gene Y, and vise versa.

2. *Community effect* subcircuits are positive feedback subcircuits that occur be-

tween cells, rather than within a single cell as in the standard positive feedback subcircuit. The community effect subcircuit also function to reduce variability, but again, between cells. As demonstrated in Figure 2.2, the gene that encodes the ligand is activated by reception of the ligand Figure 2.2. This type of inter-cellular communication is necessary for the maintenance of regulatory states and for ensuring that each cell expresses the same downstream genes. Interestingly, analogous dynamics are found in non-molecular systems, for example, in the formation of opinions in social systems. The state (opinion) of one individual can cause an adjacent agent (friend) to adopt the same opinion.

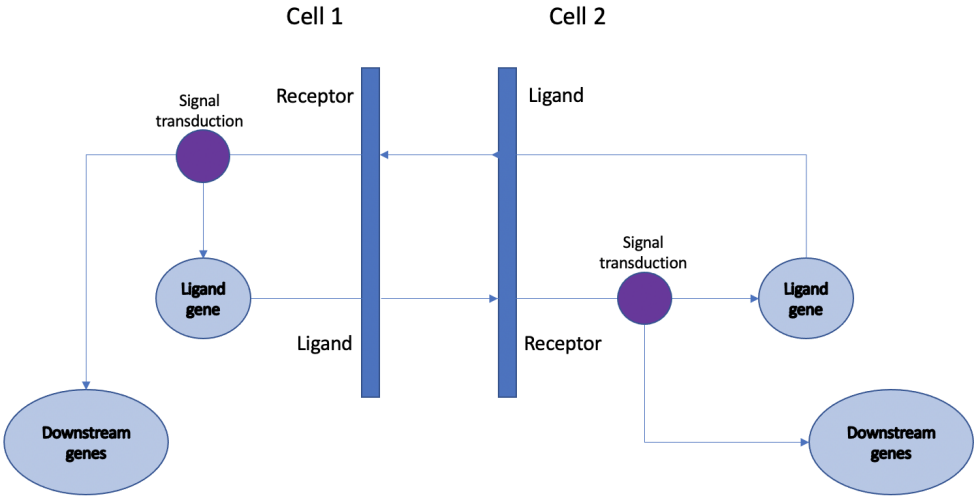


Figure 2.2: **Community effect subcircuit**. A positive feedback loop is created between Cell 1 and Cell 2. Cell 1 produces a ligand which binds to the receptor in Cell 2, resulting in signal transduction which activates the gene encoding the ligand in Cell 2. The Cell 2 ligand binds to the receptor of Cell 1, activating the signal transduction pathway responsible for activating the ligand gene in Cell 1 as well as downstream regulatory genes.

3. *Coherent feedforward* subcircuits consist of at least three genes and form a cascade. There are four possible types of coherent feedforward subcircuits. Type one is shown in Figure 2.3. This is the most basic type where X activates Y and Z, and Y activates Z, creating two parallel gene regulation paths (one from X to Z and an indirect one from X to Y to Z). Coherent feedforward subcircuits are characterized through the agreement of signs of the direct path (X to Z) and the indirect path (X to Z through Y). For example, in coherent type two, X directly represses Z, and because X represses Y, which activates Z, the indirect path from X to Z through Y also results in the repression of Z. Similarly, in coherent type three, X directly represses Z, and because Y represses Z, the indirect path from X to Z through Y also leads to the repression of Z. The sign agreement of the direct and indirect paths provide a mechanism for high levels of expression of the target genes. Furthermore, the structure of this subcircuit can also cause temporal delays in target gene expression. We demonstrate a delay using a coherent type one, under the premise that activation of Z requires inputs from X and Y. After X receives an input signal and exceeds the threshold for its activation, it binds to the promoter for Y to initiate transcription. At the same time, it binds directly to the promoter of Z. Sufficient amounts of X must be produced in order for X to activate both Y and Z. Since Z cannot be activated only by X, sufficient amounts of Y must accumulate in order to cross the activation threshold so that it can bind to the promoter of Z. Thus, Z is only activated after a delay Alon (2007).

4. *Incoherent feedforward* subcircuits are similar to coherent feedforward subcircuits, except that the sign of the direct path is opposite to that of the indirect path Alon (2007). For example, looking at incoherent type 1 as shown in Fig-

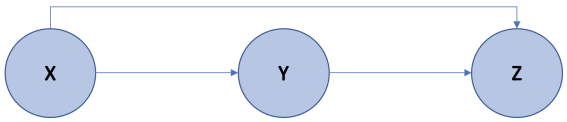


Figure 2.3: **Coherent feedforward subcircuit (type 1)**. Gene X activates the expression of gene Y and gene Z, and gene Y activates the expression of gene Z.

ure 2.4, the direct path from X to Z is positive since X activates Z, whereas the indirect path from X to Z is negative since if X activates Y, then Y represses Z. Of what biological use is this type of subcircuit? Although there are four types of incoherent feedforward subcircuits, type 1 in particular is used for subdividing spatial domains during development. This will be discussed in the next section. In summary though, gene Z can only be expressed in cells where X, not Y, is expressed; thus gene X is expressed in a domain that encompasses two subregions: one expressing gene Y and the other expressing gene Z Peter and Davidson (2015).

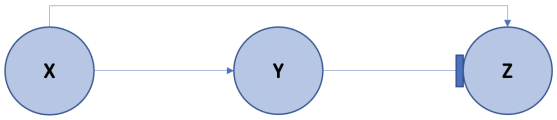


Figure 2.4: **Incoherent feedforward subcircuit (type 1)**. Gene X activates the expression of gene Y and gene Z, and gene Y inhibits the expression of gene Z.

5. *Signal mediated toggle switch* subcircuits are characterized by their ability to modify the activity of a transcription factor, in this case called an *immediate response factor*. Depending on the signal received, the immediate response factor can change its state to either positive or negative acting, as shown in Figure 2.5 which alters downstream target gene expression. Similarly to the community effect subcircuit, signal mediated toggle switch subcircuits can link the regulatory activities of GRNs in two different cells so that the GRN in one cell produces the signal (ligand) and the GRN in the second cell is activated through reception of the signal.

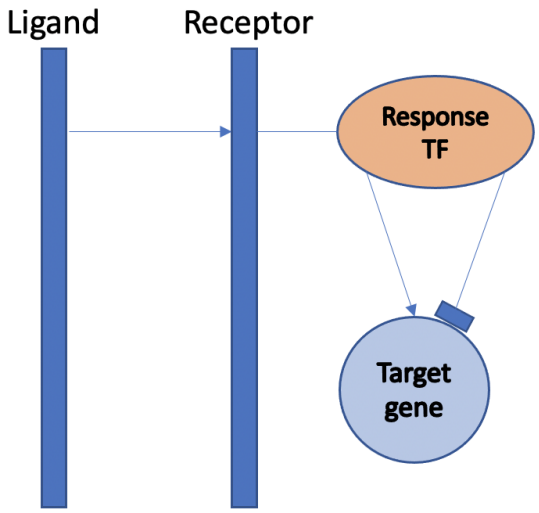


Figure 2.5: **Toggle switch subcircuit**. A gene encoding a ligand binds to a receptor. Depending on the type of ligand (the signal), the transcription factor either inhibits or activates expression of its target gene.

6. *Reciprocal repression* subcircuits consist of two genes encoding transcription factors that mutually repress each other. This type of subcircuit is used in cell fate determination. As depicted in Figure 2.6, if an input activates gene X, the transcription factor encoded by gene X represses transcription of gene Y, resulting in cell fate A and the repression of cell fate B. Conversely, if an input activates gene Y, the transcription factor encoded by gene Y represses the transcription of gene X, resulting in cell fate B and the repression of cell fate A.

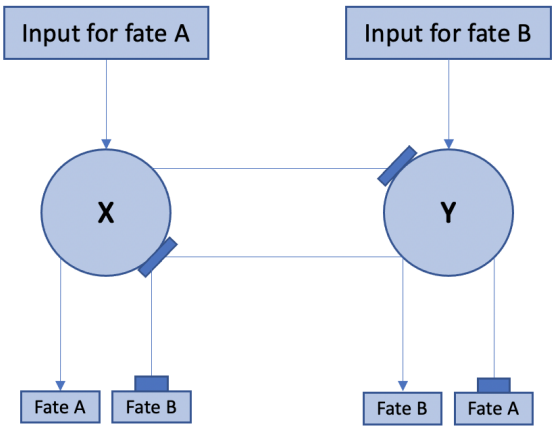


Figure 2.6: **Reciprocal repression subcircuit**. If an input favoring gene X is received, then gene X represses gene Y, resulting in cell fate A. If an input favoring gene Y is received, then gene Y represses gene X, resulting in cell fate B.

7. *Spatial exclusion* subcircuits functional similarly to reciprocal repression subcircuits in that they are used in cell fate determination, specifically in repressing alternative cell fates. This is accomplished by ensuring that regulatory genes encoding a particular cell fate are silenced. As shown in Figure 2.7, to achieve the specification state encoded by GRN 1, GRN 2 must be silenced; this occurs through the binding of a repressor to the regulatory gene of GRN 2. Similarly, to achieve the specification state encoded by GRN 2, binding of a repressor to the regulatory gene of GRN 1 silences GRN 1, thereby producing the specification state encoded by GRN 2.

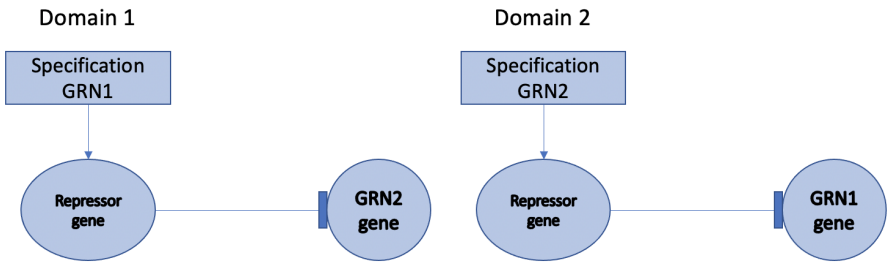


Figure 2.7: **Spatial exclusion subcircuit**. In Domain 1, a gene encoding a repressor binds to the regulatory gene responsible for the specification state encoded by GRN 2. In Domain 2, a gene encoding a repressor binds to the regulatory gene responsible for the specification state encoded by GRN 1.

8. *Double negative gate* subcircuits consist of multiple genes encoding repressors that are wired in tandem. They function in the spatial control of development by ensuring that target genes are expressed exclusively in certain subregions and repressed elsewhere regardless of the presence of their activators in those "elsewhere" subregions. In this case, the action of the repression must be dominant to the action of the activators. In Figure 2.8, there are two domains, A and D, where D $\supset$ A (D includes A). Gene X and gene Z are only expressed in

Domain A, since if gene X is activated, then it represses gene Y so that gene Y cannot repress gene Z. However, if gene X is not active, then gene Y produces the repressor binding to gene Z, so that gene Z is repressed in Domain D minus A. Therefore, this subcircuit ensures that gene Z is only expressed in domain A and repressed in domain D minus A.

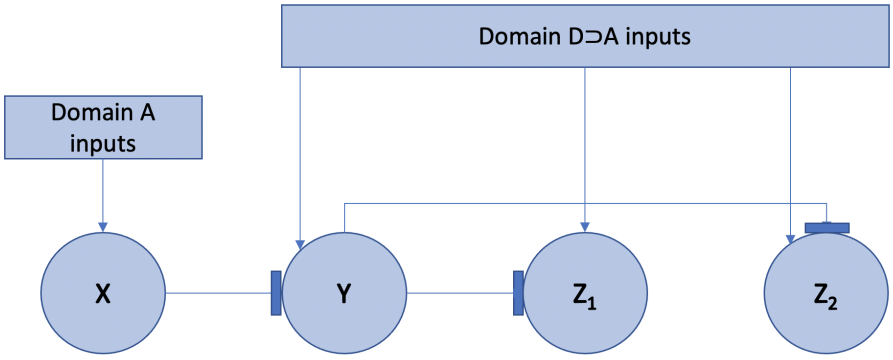


Figure 2.8: **Double negative gate subcircuit**. Given a domain A-specific input, gene X will repress gene Y, thereby allowing the domain D>A-specific inputs to activate the activity of genes Z. If gene X is not active, then gene Y will repress genes Z in domain D minus A.

As just described, one of the ways in which GRNs exhibit a hierarchical structure is through their organization into different subcircuits, where GRN subcircuit design produces specific dynamic functions due to the kinetics of transcription and translation Hinman and Cheatle Jarvela (2014). This type of hierarchy also makes GRNs modular, an efficient feature of evolved complex systems. There is a second way in which GRNs are hierarchical, and that is temporally. This is because the developmental process itself is sequential and unidirectional, in that the output of each step serves as the input to the following one. An overview of the various steps of organismal development as well as the roles of the subcircuits just discussed in the

various steps of development are discussed next.

2.2 How to Build an Organism

Referring back to the questions posed in the beginning of this section, how does one go from a single cell, the fertilized egg, to an organism with arms, legs, internal organs, and a central nervous system? And, how is it that all those parts are right where they should be (assuming that the developmental process proceeds as it should)? Now that we have an understanding of the basic biology of gene regulatory networks, we will now provide answers to these questions through an overview on how GRNs mediate the process of organismal development.

Organismal development is initiated by the localization of maternal anisotropies in the egg, which are asymmetric distributions of mRNA and other proteins needed for the activation of regulatory activity in the embryo de Leon and Davidson (2007). These initial regulatory activities set the broad coordinates and section the overall body plan which at the end of the process is organized into distinct regions with over hundreds of different cell types. The process of organismal development is one of progressive subdivisions. Step one is *axis formation and specification initiation*, and the GRNs active in this step will set the position for the future body parts. The subcircuits involved in this step act in pattern formation processes, which are processes that require coordinating the number of cells, their placement in particular regions, and placement near the correct set of neighbors, as well as their genomically encoded instructions on which cell fate the cells become, ultimately leading to the formation of a three-dimensional structure Niswander (2019). In considering the dynamics of the eight subcircuits discussed in Section 2.1.2, the double negative gate and incoherent feedforward subcircuits are most likely to be active in step one. This is because their functions involve spatial domain subdivision rather than the

stabilization of regulatory states that is accomplished by positive feedback loops. The broad domains of the endoderm, mesoderm, and ectoderm are established during this stage and constrain the position of future body parts Peter and Davidson (2011).

Within these broad domains, the GRNs in step two of organismal development establish the progenitor fields for future body parts. Progenitor fields consist of sets of progenitor cells, which are descendants of stem cells. Unlike stem cells, which have the potential to differentiate into essentially any cell type, progenitor cells correspond to specific cell types. The GRNs in the cells of these progenitor fields, as well as other GRNs that function in the formation of specific body parts, require their active regulatory genes to remain on even when the initial input signal is no longer present. As such, the subcircuits that are most likely to be found in this step are those involved in the stabilization of regulatory states, such as, for example, community effect or positive feedback subcircuits.

Once the progenitor fields are established, step three is another round of domain subdivision, in which the progenitor fields are subdivided into regions corresponding to the different sub-body parts, such as the limb zeugopod. This step requires the activity of pattern formation GRNs once again, which sets the coordinates within the progenitor fields to establish the boundaries of these sub-body parts. Therefore, the double negative gate, incoherent feedforward, and signal mediated toggle switch subcircuits are most likely to be active during this step.

Depending on the complexity of the body part, steps two and three may be re-iterated in a loop so that progenitor fields are established in more specific spatial domains. The progenitor fields are further subdivided until the cell fate specification GRNs are activated in step four. These GRNs act to specify the individual cell types that each sub-body part is composed of. Some of the GRN subcircuits that are most likely to be active during cell fate specification are the reciprocal repression,

positive feedback, and community effect subcircuits. This is because they are needed for the maintenance and stabilization of regulatory states. The output of step four serves as the input to *differentiation gene batteries*. These are functionally related sets of structural genes that encode the physical structures and functional properties of a specific cell type and are under the control of the same set of transcription factors Peter and Davidson (2011). This is the final output of the genomically encoded process of development. Hundreds of different cell types have been created, each in their particular spatial positions, correct quantities, and near neighbors of the correct cell types to form the body parts that together constitute an integrated whole: the organism.

2.2.1 GRNs and Mechanisms of Complex Diseases

The majority of diseases are complex, meaning that they are a result of the interaction between various genetic, environmental, and life style factors and not necessarily a result of a single genetic variant. Examples of complex diseases include most cancers, autism, and diabetes. Since gene regulatory networks integrate information from the environment to perform specific cellular functions, understanding diseases at the level of GRNs can yield insights on the mechanistic underpinnings of various diseases as well as aid in the development of targeted drug therapies by creating personalized GRNs Van Der Wijst *et al.* (2018).

Disease phenotypes are a result of the rewiring of gene regulatory networks. In turn, these rewirings disrupt signaling pathways that are vital for maintaining normal cellular functions. Several studies have identified specific gene regulatory network transformations involved in the manifestation of disease phenotypes ranging from breast cancer to schizophrenia Potkin *et al.* (2010). A study by Madhamshettiwar *et al.* (2012) identified a gene regulatory network involved in ovarian cancer progres-

sion. In normal cells, this network consists of 15 target genes which are regulated by the SP3 and NFκB1 transcription factors. This network is responsible for blood vessel growth in the ovaries, or angiogenesis. In cancerous cells, these 15 genes are regulated instead by the E2F1 transcription factor, causing a disruption in the angiogenesis program. From this, the resulting protein products from this gene regulatory network were identified as potential anti-cancer drug targets. In Sadeghi *et al.* (2016), the authors identified the regulatory interactions underlying the transition from primary to the metastitic state in prostate cancer. Many of the genes in the network are key components of the TGF-β signaling pathway, which is necessary for cell proliferation and differentiation. In metastitic prostate turmors, these genes were downregulated, causing a disruption in this vital signaling pathway. A particularly interesting study in Assi *et al.* (2019) found that acute myeloid leukemia (AML) is a result not of a network rewiring, but rather the formation of an entire transcriptional and signaling network. In addition, they found that specific types of AML correspond to specific regulatory networks.

These are but a few examples of how complex diseases can result from the rewiring of entire GRNs, and even the formation of novel GRNs. The conclusions of these studies also demonstrate that the topology of GRNs is intimately connected to their function, and that rewiring ultimately disrupts essential biological functions. It is the disruption of these functions that are the cause of complex disease phenotypes. Therefore, understanding the topology of regulation at the cellular level will continue to yield insights on our understanding of disease mechanisms and aid in the development of targeted drug therapies.

2.2.2 Mechanisms of GRN Evolution

Classical evolutionary theory posits that phenotypic novelties arise through small gradual changes. These changes are mutations in a DNA sequence which are random with respect to evolutionary utility. Over extremely long periods of time, such small changes produce variation in populations Erwin (2019). Adaptation is then a result of natural selection acting on this population variation, causing changes in gene frequencies. At low population sizes, neutral or slightly maladaptive traits can be fixed in a population, a phenomenon called *genetic drift*. This traditional view assumes that changes in gene frequencies are sufficient to explain all evolutionary patterns Erwin and Davidson (2009). Although some patterns of adaptive evolution can certainly follow this trajectory, classical evolutionary theory is insufficient in explaining how evolutionary novelties make their way into a population in the first place. It is also limited in its explanatory power as a mechanistic explanation of phenotypic evolution **?**. We will now describe the numerous types of modifications that can be made to developmental gene regulatory networks to produce evolutionary change. Such modifications will have specific consequences to the overall body plan depending on the type of change and where in the spatiotemporal hierarchy of the network underlying the developmental process they occur Erwin and Davidson (2009).

Modifications to GRNs can either be made through numerous types of alterations to cis-regulatory modules or through changes to the protein coding sequences of structural genes. We will primarily focus on alterations to cis-regulatory modules (CRMs) as well as entire subcircuits since these types of changes are more common. This is because if we consider the effects of a mutation to a structural gene, assuming it is pleiotropic, such a mutation can eliminate function in all stages of development. On the other hand, the effects of a change to a CRM or a subcircuit can be limited to a

single stage of development since CRMs and subcircuits modulate distinct stages in the developmental process Erwin and Davidson (2009). This is a consequence of the modularity and spatiotemporal hierarchical structure of GRNs and the developmental process itself, as previously discussed.

Alterations to *cis*-regulatory modules can result in quantitative effects, such as a change in the amount of mRNA produced or a change in the rate of transcription, or qualitative effects, which cause changes in which parts of the network are used at specific developmental stages, or changes in the overall topology of the network through rewiring of the functional linkages. Following Peter and Davidson (2011), direct alterations to *cis*-regulatory sequences will be referred to as internal changes, and alterations that affect where the CRM is physically located within the network will be referred to as external changes. The range of internal changes includes the appearance of a new target site(s), the loss of an old target site(s), changes in site spacing, and changes in site arrangement. These tend to produce quantitative effects, which may include a strengthening or weakening interaction of transcription factors and additional co-factors with the target site, or an overall change in the quantitative output so long as the linkages between the nodes are unchanged Erwin and Davidson (2009). Quantitative effects can be responsible for producing fine scale changes over evolutionary time, largely through changes in enhancer and promoter specificity Erwin (2019). However, many of such quantitative effects can have no impact on the overall output. A notable example is the lack of conservation of *eve* stripe 2 sites in *Drosophila melanogaster* Peter and Davidson (2011); Hare *et al.* (2008). Over 70 % of the specific sites are not conserved in other Drosophilidae but they produce the same output pattern Peter and Davidson (2011); Hare *et al.* (2008). Logically, this makes sense because quantitative changes occur at a higher frequency in comparison to qualitative changes.

The higher frequency of quantitative changes can also be a result of intrinsic and extrinsic noise. Intrinsic noise is the stochasticity inherent in the biochemical reactions that take place during the process of gene expression Wang and Aberra (2015). This includes noise from promoter activation and inactivation, mRNA and protein production, mRNA degradation, and delays in the overall timing of processes of gene expression Raser *et al.* (2005). Extrinsic noise refers to the differences between cells in the local environment or in the activity of any cellular component that affects gene expression Raser *et al.* (2005). These differences, to list a few examples, may be attributed to fluctuations in the amounts of RNA polymerases, the number of ribosomes in a cell and their individual growth dynamics, or the stage in the cell cycle Wang and Aberra (2015). This illustrates a point about the robustness of GRNs and the developmental process. If the system were so sensitive that any quantitative effect would have an impact on whether or not a gene would produce a certain output, then it would arguably be very rare for the developmental process to reach completion, let alone be evolutionarily conserved across a range of organisms for millions of years [2] . That is why quantitative effects tend to result in fine scale changes instead of major morphological novelties over evolutionary time.

On the other hand, qualitative changes, which are largely in the class of external changes, result in the evolutionary repatterning of entire GRNs Erwin (2019). For one, although this qualifies as an internal change, the appearance of a new target site or loss of an old target site can also result in the rewiring of GRNs if there is a complete disruption of a TF-promoter interaction or the appearance of a new functional linkage. This can result in a gain of function, as seems to be the case in the fin to limb transition in vertebrates. The emergence of a distal limb enhancer

[2]This is regardless of whether the quantitative effect were caused by either direct modifications to the CRMs or intrinsic and extrinsic noise.

in the *Hoxa11* intron during the evolution of pentadactyl limbs caused the mutually exclusive expression of *Hoxa11* and *Hoxa13* Kherdjemil *et al.* (2016). Following activation at this site by HOXA11 and HOXD13, the enhancer drives the antisense transcription of the *Hoxa11* gene and prevents its expression Kherdjemil *et al.* (2016). Thus, target site additions or losses can result in quantitative and qualitative changes to GRNs.

The range of external changes alter where the CRM is physically located within the GRN and redeploy them to new developmental domains. The insertion of transposable elements, which is one of the major causes of rapid genomic sequence change, can carry with them entire cis-regulatory modules to new genes Peter and Davidson (2011). This could result in either a gain or loss of function due to the gain of inputs to the area of the GRN where the transposable element is inserted. This may simultaneously prevent previously-existing linkages from persisting. Similarly, duplication and subfunctionalization of regulatory genes, as well as the *de novo* origination of sequences via recombination can also result in gain or loss of function due to the rewiring of pre-existing linkages.

Another important mechanism of evolutionary change is subcircuit co-option, which is the insertion of entire subcircuits to new regions of the GRN. As described in Section 2.1.1, subcircuits exhibit specific functional roles that are intimately tied to their structure. Therefore, the insertion of these subcircuits (in which the subcircuit linkages remain the same) into different regions of the network not only is largely responsible for generating the hierarchical structure of GRNs, but will result in a gain of function change so long as either the same inputs responsible for activating the subcircuit are present, or so long as alternative factors that can still activate the subcircuit are present Hinman and Cheatle Jarvela (2014). Numerous studies have demonstrated that subcircuit co-option is a mechanism for generating

phenotypic novelty. Subcircuit co-option has been a driver in the evolution of the sea urchin larval skeleton, the head Martik and Bronner (2017); Martik and McClay (2015), butterfly eyespots Monteiro and Podlaha (2009), the scaly foot snail Sun *et al.* (2020), the formation of the secondary jaw in teleost fish Fraser *et al.* (2009); Erwin and Davidson (2009), and more than likely other instances of phenotypic novelty that have yet to be identified. As well, changes in regulatory genes that are responsible for controlling entire subcircuits, called input/output (I/O) switches, can result in the activation or repression of subcircuits, as well as the intercellular signaling systems connecting regulatory subcircuits Erwin and Davidson (2009). Although much more experimental work needs to be done on exploring the role of subcircuit co-option as a driver of major phenotypic novelty, subcircuit co-option presents an interesting evolutionary phenomenon that is suitable for modeling using a state space approach. This facilitates the exploration of questions including how subcircuit topology influences the potential for evolutionary change, or how subcircuits are combined in different stages of development to generate novelty Erwin (2019).

The evolutionary consequences of the alterations to GRNs depend on the region of the network and the developmental stage in which the alteration occurr. GRNs in certain stages in development are more flexible to change than others. Reconsidering the process of development described in Section 2.2, changes that occur early on during development are more likely to alter the entire course of development. Although most of these changes are deleterious since they have downstream pleiotropic effects, those that have become fixed in a population are likely the root cause of phylum level morphological differences. For example, the GRNs acting in axis formation and specification initiation set the initial coordinates of the body plan. If these coordinates are altered through changes in the major subcircuits involved in this early step of organismal development (e.g. double negative gate or incoherent feedforward), the

overall body plan can be altered as a result. This is also the case with GRNs acting in the establishment of progenitor fields for future body parts. It is therefore hypothesized that GRNs involved in these early stages of development are more likely to be evolutionarily conserved. Davidson and Erwin Erwin and Davidson (2009) have identified the existence of *kernels*, which are evolutionarily conserved sets of few genes linked by recursively wired positive interactions and are involved in the establishment of progenitor fields. These kernels have been canalized over evolutionary time and constrain the range of accessible variation that can be produced by the developmental process. Furthermore, if one considers rewiring the signaling systems connecting GRNs involved in axis formation/specification initiation and those involved in the establishment of progenitor fields, the result is an alteration in the positions of the future body parts relative to one another, which also contributes to phylum level morphological differences.

Class level morphological differences in the body parts themselves are due to evolutionary changes in the cis-regulatory systems responsible for controlling the inductive signaling mechanisms involved in the formation of body parts Peter and Davidson (2011). This involves changes to the structure and deployment of subcircuits active in the pattern formation processes that subdivide the already established progenitor fields into sub-body parts (e.g. signal mediated toggle switch, double negative gate, and incoherent feedforward). Interestingly, we do see higher rates of evolution in inductive signaling relationships Peter and Davidson (2011). The highest rates of evolutionary change occur at the most downstream levels of the GRN hierarchy, mainly in differentiation gene batteries. These regions of GRNs are most flexible to change because they exhibit relatively minimal feedback into upstream regions of the network. As a result, changes that occur at the periphery of these networks affect less stages of the developmental process.

Chapter 3

BAYESIAN INFERENCE USING STATE SPACE MODELS

In this section, we provide the mathematical background for Bayesian state estimation. This section is based on Simon (2006); McNames (2016).

3.0.1 System Modeling

State space models are used for representing the dynamics of continuous or discrete time systems. Given a set of noisy measurements $\mathbf{y}_k \in \mathbb{R}^P$ at time step k, we are interested in estimating the unknown system parameters or state $\mathbf{x}_k \in \mathbb{R}^N$ at time step k. The state space representation is given by:

$$\mathbf{x}_k = g_{k-1}(\mathbf{x}_{k-1}, \mathbf{w}_{k-1}) \tag{3.1}$$

$$\mathbf{y}_k = h_k(\mathbf{x}_k, \mathbf{v}_k) \tag{3.2}$$

The transition equation in Equation (3.1) relates the current state at time step k and the state at the previous time step $k-1$. This is often based on a physical model of a system described by the (possibly time-varying) function $g_k(\cdot)$. The random process $\mathbf{w}_k$ is used to model possible deviations from the physical model. The measurement equation in Equation (3.2) provides the relationship between the measurement and state at time step k using the (possibly time-varying) function $h_k(\cdot)$; the measurement noise is given by the random process $\mathbf{v}_k$. The generality of this model allows for the representation of linear or nonlinear dynamics as well as varying types of random processes. It is thus suitable for representing the dynamics of dGRNs since dGRNs are characterized by highly nonlinear interactions.

3.0.2 Recursive Bayesian Estimation

Assuming the known transition and measurement functions in Equation (3.1) and Equation (3.2), and known characterization of the random processes, the goal of Bayesian estimation is to estimate the state of the system $\mathbf{x}_k$ at some time step k given a sequence of measurements $y_{1:k} = \{y_1, y_2, ..., y_k\}$ by estimating the conditional probability density function (pdf) [1] ,

$$p(\mathbf{x}_k|\mathbf{y}_{1:k}) \tag{3.3}$$

The conditional pdf in Equation (3.3) is used to compute the *a posteriori*, or *filtered*, estimate $\hat{\mathbf{x}}_k$. Given an initial pdf $p(x_0)$, we can obtain the unknown pdf in Equation (3.3) in two stages: *prediction* and *update*. The stages are computed iteratively, starting at $k = 0$. During the predicted stage at time k, the system model is used to compute the predicted marginal pdf,

$$p(\mathbf{x}_k|\mathbf{y}_{1:k-1}) \tag{3.4}$$

which is the pdf of $\mathbf{x}_k$ given all measurements up to and including time $k-1$. The predicted marginal pdf can be obtained using the Chapman-Kolmogorov equation as follows Arulampalam *et al.* (2002a):

$$p(\mathbf{x}_k|\mathbf{y}_{1:k-1}) = \int p(\mathbf{x}_k, \mathbf{x}_{k-1}|\mathbf{y}_{1:k-1}) d\mathbf{x}_{k-1} \tag{3.5}$$

$$= \int p(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{y}_{1:k-1}) p(\mathbf{x}_{k-1}|\mathbf{y}_{1:k-1}) d\mathbf{x}_{k-1} \tag{3.6}$$

$$= \int p(\mathbf{x}_k|\mathbf{x}_{k-1}) p(\mathbf{x}_{k-1}|\mathbf{y}_{1:k-1}) d\mathbf{x}_{k-1} \tag{3.7}$$

$$\tag{3.8}$$

[1] For the filtering problem, there are two posterior pdfs that one might estimate, the joint posterior pdf $p(x_{0:k}|y_{0:k})$, or the marginal posterior pdf $p(x_k|y_{0:k})$.

where $p(\mathbf{x}_k|\mathbf{x}_{k-1})$ is the prior and $p(\mathbf{x}_{k-1}|\mathbf{y}_{1:k-1})$ is the marginal posterior pdf at the previous time step. The steps of the joint posterior are given in Appendix A. Here, the common assumption of first order Markov property is used. The joint posterior pdf considers the entire state trajectory and the computational requirement grows with time. The marginal posterior pdf is therefore more efficient to calculate. Given the measurement $\mathbf{y}_k$ at time step k, it is used to update the estimate using Bayes' rule:

$$p(\mathbf{x}_k|\mathbf{y}_{0:k}) = \frac{p(\mathbf{y}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{y}_{1:k-1})}{p(\mathbf{y}_k|\mathbf{y}_{1:k-1})} \tag{3.9}$$

Thus, Equation (3.9) yields the marginal posterior pdf, where $p(\mathbf{y}_k|\mathbf{x}_k)$ is the likelihood, $p(\mathbf{x}_k|\mathbf{y}_{1:k-1})$ is the predicted marginal pdf, and $p(\mathbf{y}_k|\mathbf{y}_{1:k-1})$ is a normalization term that is obtained by,

$$\begin{aligned} p(\mathbf{y}_k|\mathbf{y}_{1:k-1}) &= \int p(\mathbf{y}_k, \mathbf{x}_k|\mathbf{y}_{1:k-1})d\mathbf{x}_k \\ &= \int p(\mathbf{y}_k|\mathbf{x}_k, \mathbf{y}_{1:k-1})p(\mathbf{x}_k|\mathbf{y}_{1:k-1})d\mathbf{x}_k \\ &= \int p(\mathbf{y}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{y}_{1:k-1})d\mathbf{x}_k \end{aligned}$$

As, in general, **??** and **??** cannot be computed analytically, they can be obtained using stochastic approximations. However, the process becomes computationally intensive and is often not feasible.

The Kalman Filter and Its Variants

When the functions $g_k(\cdot)$ and $h_k(\cdot)$ correspond to linear models and the random processes $\mathbf{w}_k$ and $\mathbf{v}_k$ are Gaussian in Equation (3.1) and Equation (3.2), then the posterior pdf in Equation (3.3) can be obtained in closed form. The closed form solution is called the Kalman filter and it results in the optimal minimum mean square error estimator of the unknown state Simon (2006). For the Kalman filter

solution, the transition equation in Equation (3.1) simplifies to

$$\mathbf{x}_k = F_k \mathbf{x}_{k-1} + \mathbf{w}_{k-1} \tag{3.10}$$

and the measurement equation in Equation (3.2) is given by,

$$\mathbf{y}_k = H_k \mathbf{x}_k + \mathbf{v}_k \tag{3.11}$$

where F_k and H_k are matrices and

$$\mathbf{w}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_k) \tag{3.12}$$

and

$$\mathbf{v}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_k) \tag{3.13}$$

are both Gaussian and have known covariance matrices $\mathbf{Q}_k$ and $\mathbf{R}_k$, respectively. The mean of the state is the Kalman filter estimate of the state and the covariance of the state is the covariance of the Kalman filter state estimate Simon (2006). The *a posteriori* state estimate can be computed as the expected value of $\mathbf{x}_k$ conditioned on all the measurements including time k:

$$\hat{\mathbf{x}}_k^+ = \mathbb{E}[\mathbf{x}_k | \mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_k] \tag{3.14}$$

The *a priori* state estimate can be computed as the expected value of $\mathbf{x}_k$ conditioned on all the measurements up to but not including time k:

$$\hat{\mathbf{x}}_k^- = \mathbb{E}[\mathbf{x}_k | \mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_{k-1}] \tag{3.15}$$

The covariances of the estimation error of $\hat{\mathbf{x}}_k^+$ and $\hat{\mathbf{x}}_k^-$ are given by,

$$P_k^+ = \mathbb{E}[(\mathbf{x}_k - \hat{\mathbf{x}}_k^+)(\mathbf{x}_k - \hat{\mathbf{x}}_k^+)^T] \tag{3.16}$$

$$P_k^- = \mathbb{E}[(\mathbf{x}_k - \hat{\mathbf{x}}_k^-)(\mathbf{x}_k - \hat{\mathbf{x}}_k^-)^T] \tag{3.17}$$

The initial estimate of the state $\mathbf{x}_0$ is given by $\hat{\mathbf{x}}_0^+ = \mathbb{E}[(\mathbf{x}_0)]$ and the initial covariance of the estimate of $\mathbf{x}_0$ is given by $P_0 = \mathbb{E}[(\mathbf{x}_0 - \bar{\mathbf{x}}_0)(\mathbf{x}_0 - \bar{\mathbf{x}}_0)^T]$. The Kalman filter consists of two steps, 1) the time update (prediction) and 2) the measurement update (filter), which are iterated. To obtain the predicted estimate for the state, we begin by substituting $\mathbf{x}_k$ into the *a priori* state estimate equation. To simplify the notation, we will use $\mathbb{E}_{k-1}[\mathbf{x}_k]$ to denote the expectation of $\mathbf{x}_k$ given all measurements up to and including $\mathbf{y}_{k-1}$.

$$\hat{\mathbf{x}}_k^- = \mathbb{E}_{k-1}[\mathbf{x}_k] \tag{3.18}$$

$$= \mathbb{E}_{k-1}[F_{k-1}\mathbf{x}_{k-1} + G_{k-1}\mathbf{u}_{k-1} + \mathbf{w}_{k-1}] \tag{3.19}$$

$$= F_{k-1}\mathbb{E}[\mathbf{x}_{k-1}] + \mathbb{E}[G_{k-1}\mathbf{u}_{k-1}] + \mathbb{E}[\mathbf{w}_{k-1}] \tag{3.20}$$

$$= F_{k-1}\mathbb{E}[\mathbf{x}_{k-1}] + G_{k-1}\mathbf{u}_{k-1} + \mathbb{E}[\mathbf{w}_{k-1}] \tag{3.21}$$

$$\hat{\mathbf{x}}_k^- = F_{k-1}\hat{\mathbf{x}}_{k-1}^+ + G_{k-1}\mathbf{u}_{k-1} \tag{3.22}$$

Next, we will derive the time update for the state error covariance P_k^-. Now that we know $\hat{\mathbf{x}}_k^-$ and $\hat{\mathbf{x}}_{k-1}^-$, we can define,

$$\begin{aligned}
\tilde{\mathbf{x}}_k^- &\triangleq \mathbf{x}_k - \hat{\mathbf{x}}_k^- \\
&= (F_{k-1}\mathbf{x}_{k-1} + G_{k-1}\mathbf{u}_{k-1} + \mathbf{w}_{k-1}) - (F_{k-1}\hat{\mathbf{x}}_{k-1}^+ + G_{k-1}\mathbf{u}_{k-1}) \\
&= F_{k-1}(\mathbf{x}_{k-1} - \hat{\mathbf{x}}_{k-1}^+) + \mathbf{w}_{k-1} \\
&= F_{k-1}\tilde{\mathbf{x}}_k^+ + \mathbf{w}_{k-1}
\end{aligned}$$

The time update for the state error covariance P_k^- is given by,

$$\begin{aligned} P_k^- &= \mathbb{E}[(F_{k-1}\tilde{\mathbf{x}}_{k-1}^+ + \mathbf{w}_{k-1})(F_{k-1}\tilde{\mathbf{x}}_{k-1}^+ + \mathbf{w}_{k-1})^T] \\ &= \mathbb{E}[F_{k-1}\tilde{\mathbf{x}}_{k-1}^+\tilde{\mathbf{x}}_{k-1}^+]^T] + \mathbb{E}[F_{k-1}\tilde{\mathbf{x}}_{k-1}^+\mathbf{w}_{k-1}^T] \\ &+ \mathbb{E}[\mathbf{w}_{k-1}\tilde{\mathbf{x}}_{k-1}^{+T}F_{k-1}] + \mathbb{E}[\mathbf{w}_{k-1}\mathbf{w}_{k-1}^T] \\ &= F_{k-1}\mathbb{E}[\tilde{\mathbf{x}}_{k-1}^+\tilde{\mathbf{x}}_{k-1}^{+T}]F_{k-1}^T + \mathbb{E}[\mathbf{w}_{k-1}\mathbf{w}_{k-1}^T] \\ &= F_{k-1}P_{k-1}^+F_{k-1}^T + Q_{k-1} \end{aligned}$$

where P_{k-1}^+ is the filtered state error covariance and Q_{k-1} is the measurement noise covariance. The next step is to calculate the measurement update for the state, which uses the current measurement $\mathbf{y}_k$ to obtain the *a posteriori* (filtered) estimate $\hat{\mathbf{x}}_k^+$. The filtered estimate is given by,

$$\hat{\mathbf{x}}_k^+ = \hat{\mathbf{x}}_k^- + K_k\mathbf{e}_k \tag{3.23}$$

where K_k is the Kalman gain and $\mathbf{e}_k$ is called the *innovation* and is given as the difference between the measurement and the predicted estimate of the measurement:

$$\mathbf{e}_k \triangleq \mathbf{y}_k - \hat{\mathbf{y}}_k^- \tag{3.24}$$

and

$$\begin{aligned} \hat{\mathbf{y}}_k^- &= \mathbb{E}_{k-1}[H_k\mathbf{x}_k + \mathbf{v}_k] \\ &= H_k\hat{\mathbf{x}}_k^- \end{aligned}$$

We can then compute the measurement update for the state error covariance P_k^+ as,

$$
\begin{aligned}
P_k^+ &= \mathbb{E}[(\tilde{\mathbf{x}}_k^+)^2] \\
&= \mathbb{E}[(\mathbf{x}_k - \hat{\mathbf{x}}_k^+)^2] \\
&= \mathbb{E}[(\mathbf{x}_k - \hat{\mathbf{x}}_k^- - K_k\mathbf{e}_k)^2] \\
&= \mathbb{E}[(\mathbf{x}_k - \hat{\mathbf{x}}_k^- - K_k(\mathbf{y}_k - H_k\hat{\mathbf{x}}_k^-))^2] \\
&= \mathbb{E}[(\tilde{\mathbf{x}}_k^- - K_k(H_k\mathbf{x}_k + \mathbf{v}_k - H_k\hat{\mathbf{x}}_k^-))^2] \\
&= \mathbb{E}[(\tilde{\mathbf{x}}_k^- - K_kH_k(\mathbf{x}_k - \hat{\mathbf{x}}_k^-) - K_k\mathbf{v}_k)^2] \\
&= \mathbb{E}[(\tilde{\mathbf{x}}_k^- - K_kH_k\tilde{\mathbf{x}}_k^- - K_k\mathbf{v}_k)^2] \\
&= \mathbb{E}[((I - K_kH_k)\tilde{\mathbf{x}}_k^- - K_k\mathbf{v}_k)^2] \\
&= \mathbb{E}[(I - K_kH_k)\tilde{\mathbf{x}}_k^-\tilde{\mathbf{x}}_k^{-T}(I - K_kH_k)^T)] - \mathbb{E}[(I - K_kH_k)\tilde{\mathbf{x}}_k^-\mathbf{v}_k^TK_k^T] \\
&- \mathbb{E}[K_k\mathbf{v}_k\tilde{\mathbf{x}}_k^{-T}(I - K_kH_k)^T] + \mathbb{E}[K_k\mathbf{v}_k\mathbf{v}_k^TK_k^T] \\
&= (I - K_kH_k)\mathbb{E}[(\tilde{\mathbf{x}}_k^-)^2](I - K_kH_k)^T + K_k\mathbb{E}[(\mathbf{v}_k)^2)]K_k^T \\
&= (I - K_kH_k)P_k^-(I - K_kH_k)^T + K_kR_kK_k^T
\end{aligned}
$$

To solve for the Kalman gain K_k, the total mean squared error (MSE) is defined as $\xi(K_k) = \text{trace}P_k^+$. We take the partial derivative of $\xi(K_k)$ with respect to K_k and set it equal to zero to find the minimum:

$$
\begin{aligned}
\frac{\partial\xi(K_k)}{\partial K_k} &= 2(I - K_kH_k)P_k^-(-H_k)^T + 2K_kR_k = 0 \\
K_kR_k &= (I - K_kH_k)P_k^-H_k^T \\
K_k(R_k + H_kP_k^-H_k^T) &= P_k^-H_k^T \\
K_k &= P_k^-H_k^T(R_k + H_kP_k^-H_k^T)^{-1}
\end{aligned}
$$

In summary, for $k = 1, ...N$, the Kalman filter recursions are given as follows,

$$
\begin{aligned}
P_k^- &= F_{k-1} P_{k-1}^- F_{k-1}^T + Q_{k-1} \\
K_k &= P_k^- H_k^T (R_k + H_k P_k^- H_k^T)^{-1} \\
\hat{\mathbf{x}}_k^- &= F_{k-1} \hat{\mathbf{x}}_{k-1}^+ \\
\hat{\mathbf{x}}_k^+ &= \hat{\mathbf{x}}_k^- + K_k(\mathbf{y}_k - H_k \hat{\mathbf{x}}_k^-) \\
P_k^+ &= (I - K_k H_k) P_k^- (I - K_k H_k)^T + K_k R_k K_k^T
\end{aligned}
$$

When Equation (3.1) and Equation (3.2) are nonlinear, then variants of the Kalman filter, the Extended Kalman Filter (EKF) and Unscented Kalman Filter (UKF) have been implemented to infer GRNs based on nonlinear state space models. The EKF assumes that the random processes in the state space representation are Gaussian. The nonlinear system dynamics in the transition and measurement equations are linearized around an operating point so that the Kalman filter can be applied. In Wang *et al.* (2009), the authors apply the EKF to three different datasets, a Malaria model consisting of 530 genes at 48 time points, a worm model consisting of 98 genes at 123 time points, and a yeast model consisting of 237 genes at 17 time points. The EKF was used for both state and parameter estimation. However, the EKF has some disadvantages as a result of the linearization step. In particular, the estimation accuracy is decreased due to the highly nonlinear dynamics and large errors are introduced in the true posterior mean that may lead to divergence Zhou and Ji (2017); Wan *et al.* (2001). Furthermore, computing the Jacobian may be computationally intensive and difficult to calculate Amor *et al.* (2019). As well, the EKF is restricted to dynamics with Gaussian noise distributions. An extended fractional Kalman filter was introduced in Zhang *et al.* (2014), which outperformed the EKF.

An alternative to the EKF that addresses some of these problems is the UKF

Wan *et al.* (2001). The UKF uses an unscented transform to pick a set of sigma sample points around the mean, assign weights to each of the points, propagates them through a nonlinear transformation function, and then calculates a weighted mean and covariance Wan *et al.* (2001). The UKF does not rely on linearization to compute the mean and covariance, and thus is more accurate than the EKF. As well, variants of the UKF, the cubature Kalman filter (CKF) Arasaratnam and Haykin (2009) and the transformed unscented Kalman filter (TUKF) Chang *et al.* (2012) have been developed to deal with high dimensional data, but these variants have yet to be applied to gene expression data. In Zhou and Ji (2017), the authors applied the UKF to a yeast model consisting of 309 genes and 24 time points. The UKF outperformed the EKF in estimating the state variables and the parameters.

Monte Carlo Integration

Monte Carlo integration uses random sampling of a function to numerically estimate the values of its integral. By choosing random points at which to evaluate the integrand, Monte Carlo methods can approximate integrals on irregular domains and in high-dimensional spaces. According to the law of the unconscious statistician (LOTUS), the expectation of a function $g(X)$ of a random variable X is given by,

$$\mathbb{E}[f(X)] = \int_{\mathcal{X}} f(x)p(x)dx \tag{3.25}$$

where $f(x)$ is the function we wish to integrate and $p(x)$ is some probability distribution. The Monte Carlo estimate is obtained by drawing samples from $p(x)$ and then calculating

$$\hat{\mu_f} = \frac{1}{N}\sum_{i=1}^{N} f(x^i) \tag{3.26}$$

where $x^{(i)} \overset{iid}{\sim} p(x)$. A quick example of Monte Carlo integration with a uniform distribution $p(x)$ is given below.

$$p(x) = \begin{cases} \frac{1}{\delta}, x_{min} \leq x \leq x_{max} \\ 0 \qquad\qquad\qquad\qquad \text{otherwise} \end{cases} \tag{3.27}$$

where $\delta = x_{max} - x_{min}$.

$$\begin{aligned} \mu_f &= \int_{x_{min}}^{x_{max}} f(x)p(x)dx \\ &= \int_{x_{min}}^{x_{max}} f(x)\frac{1}{\delta}dx \\ &= \mathbb{E}[f(x)] \\ &\approx \hat{\mu_f} = \frac{1}{N}\sum_{i=1}^{N} f(x^i) \end{aligned}$$

Reconsidering the filtering and prediction steps, we need to draw random samples from the posterior distribution, but this is difficult to do in high dimensions. Importance sampling is a Monte Carlo method which provides a way to approximate quantities of interest for a given distribution even if we are unable to directly sample from that distribution. Instead of sampling from the target distribution we instead sample from a proposal distribution $q(x)$, called the *importance density*:

$$\mu_f = \int f(x)p(x)dx = \int [\frac{p(x)}{q(x)}]q(x)dx \tag{3.28}$$

The estimate is then given as

$$\begin{aligned} \hat{\mu}_f &= \frac{1}{N}\sum_{i=1}^{N} f(x^i)\frac{p(x^i)}{q(x^i)} \\ &= \frac{1}{N}\sum_{i=1}^{N} f(x^i)\tilde{w}(x^i) \end{aligned}$$

where $\tilde{w}(x^i)$ is the unnormalized *importance weight* and is the ratio of the target distribution to the proposal distribution. It is important to note that $q(x)$ and $p(x)$

must have the same support. The normalized importance weight is given as

$$w(x^i) = \frac{\tilde{w}(x^i)}{\sum_{i=1}^{N} \tilde{w}(x^i)} \tag{3.29}$$

With these ideas in mind, we can also estimate the mean in a different way by modeling the posterior pdf as a sum of impulses. Recall that the multivariate impulse function is given as:

$$\delta(x) = \prod_i \delta(x(i)) \tag{3.30}$$

Some important properties are listed below:

1. Unit area: $\int \delta(x)dx = 1$
2. $\delta(x-a) = 0 \forall x \neq a$; $f(a) = \int f(x)\delta(x-a)dx$
3. The function is infinitely tall at $x = a$

The estimate of the mean is thus given by:

$$\hat{p}(x) = \frac{1}{N}\sum_{i=1}^{N} \delta(x - x^i) \tag{3.31}$$

where $x^{(i)} \overset{iid}{\sim} p(x)$. Substituting $\hat{p}(x)$ into the Monte Carlo integration equation we obtain

$$\begin{aligned}
\mu_f &= f(x)p(x)dx \\
\hat{\mu}_f &= \int f(x)\hat{p}(x)dx \\
&= \int f(x)\frac{1}{N}\sum_{i=1}^{N}\delta(x-x^i)dx \\
&= \frac{1}{N}\sum_{i=1}^{N}\int f(x)\delta(x-x^i)dx \\
&= \frac{1}{N}\sum_{i=1}^{N} f(x^i)
\end{aligned}$$

Combining this with the importance weight, we obtain an estimate of the posterior as a weighted sum of impulses where the weights are given by the ratio of the target distribution to the proposal distribution:

$$\hat{p}(x) = \sum_{i}^{N} w^i \delta(x - x^i) \tag{3.32}$$

This is the premise behind particle filtering.

3.0.3 Particle Filter

A Bayesian filtering method that typically out-performs the standard Kalman filter and EKF is the particle filter. In Ancherbak *et al.* (2016a), the authors use particle filtering to estimate the state and time-varying parameters for a ten-gene network with 201 time points. They also applied it to the DREAM4 dataset using a multivariate linear regression model and a Laplace prior rather than a Gaussian prior. In their model, the relationships between the genes change at each time instant, so the network describes how gene expression levels at one time are influenced by the expression levels of genes at the previous time. Each relationship between genes is between two different time steps instead of one. They found that the use of the Laplace prior leads to smoother changes in network structure and less fluctuations in regulatory coefficients Ancherbak *et al.* (2016a). In Shen and Vikalo (2010), the authors use a modified particle filter to estimate stochastic reaction rate constants in GRNs governed by chemical Langevin equations. They applied this method to a viral infection network to estimate six rate constants and evaluated the corresponding Cramér-Rao lower bound (CRLB) Shen and Vikalo (2010). For some parameters, the estimator performed well, close to the CRLB, and for the rest it significantly underperformed.

A particularly notable use of particle filtering is found in Bugallo *et al.* (2015),

where a bank of particle filters is used to jointly estimate gene expression values and GRN interaction coefficients. Each particle filter tracks one gene and its interaction coefficients, and the filters communicate estimates of the unknowns with each other. This approach increases the efficiency of estimation and also lends itself to application to higher-dimensional problems by splitting up the state space into subspaces Bugallo *et al.* (2015). The authors apply this method to an eight-gene network that used eight filters with 75 particles per filter and compared it to the standard particle filter which used 2000 particles. Their approach out-performed the standard particle filter in estimation accuracy despite the low number of particles.

Other work makes use of both Kalman filter and particle filtering, in which one is used to estimate the state and the other is used for parameter estimation. In Noor *et al.* (2012), the authors use the particle filter for estimation of gene expression values, and the Kalman filter to estimate the constant system parameters describing the nonlinear relationships between genes in a few data sets. The first is an eight-gene network with 40 time points, and the second is a drosophila dataset consisting of 530 genes and 36 time points, of which only ten genes are chosen. For both simulations, the nonlinear function is the sigmoid and the noise is Gaussian. Using 100 particles, the authors found that the particle filter out-performed the EKF and UKF. In Wang and Aberra (2016), the authors use a particle filter to study the effect of noise on synthetic GRNs. The ensemble Kalman filter samples particles generated by the particle filter Wang and Aberra (2016). Interestingly, they apply this method to a three-gene network and a five-gene network consisting of 17 time points without the use of a model. The Kalman filter is combined with Taken's method of attractor reconstruction Rand and Young (2006), in which the dynamics are propagated nonparametrically using delay-coordinate vectors vectors Wang and Aberra (2016). Their approach out-performed the particle filter in terms of the root mean-squared

error (RMSE) since the ensemble Kalman filter based particle filter takes into account the current measurement and thereby better approach the stationary posterior distribution Wang and Aberra (2016).

Recall that the goal of recursive Bayesian estimation is to estimate either the joint pdf $p(x_{0:k}|y_{0:k})$ or the marginal pdf $p(x_k|y_{0:k})$. The key idea behind particle filtering is to assume that the posterior pdf can be expressed as a weighted sum of impulses:

$$\hat{p}(\mathbf{x}_{0:k}|\mathbf{y}_{0:k}) \triangleq \sum_{i=1}^{n_p} w_k^i \delta(\mathbf{x}_{0:k} - \mathbf{x}_{0:k}^i) \tag{3.33}$$

where the importance weights w_k^i are normalized such that

$$\sum_{i=1}^{n_p} w_k^i = 1 \tag{3.34}$$

and n_p denotes the number of particles, which are state vectors $\mathbf{x}_{0:k}^i$ for $i = 1, ..., n_p$ that are generated based on the initial pdf $p(\mathbf{x}_0)$. This results in a parameterized posterior pdf estimate that is defined by the set of parameters $\{\mathbf{x}_{0:k}^i, w_k^i\}$. At each time step the particles are propagated using the system equation.

Importance Weight Calculation

The importance weights are given by the ratio of the posterior to the recursive importance density. The importance density is typically chosen such that it factors as

$$\begin{aligned} q(\mathbf{x}_{0:k}|\mathbf{y}_{0:k}) &= q(\mathbf{x}_k|\mathbf{x}_{0:k-1}, \mathbf{y}_{0:k}) q(\mathbf{x}_{0:k-1}|\mathbf{y}_{0:k}) \\ &= q(\mathbf{x}_k|\mathbf{x}_{0:k-1}, \mathbf{y}_{0:k}) q(\mathbf{x}_{0:k-1}|\mathbf{y}_{0:k-1}) \\ &= q(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{y}_k) q(\mathbf{x}_{0:k-1}|\mathbf{y}_{0:k-1}) \end{aligned}$$

This particular factorization allows samples of the trajectory $\mathbf{x}_{0:k}$ to be created sequentially by ensuring that when the particles are advanced from $k-1$ to k, $\mathbf{x}_k^i$ is not

created based on the entire past, which would be inefficient since computation would then scale with time. Instead, $\mathbf{x}_k^i$ is created using only the current measurement $\mathbf{y}_k$ and $\mathbf{x}_{k-1}^i$.

The importance weights are then given by:

$$w_k^i = \frac{p(\mathbf{x}_{0:k}^i|\mathbf{y}_{0:k})}{q(\mathbf{x}_{0:k}^i|\mathbf{y}_{0:k})} \tag{3.35}$$

To calculate the importance weights, the following equations for the posterior and the importance density are used:

$$\begin{aligned} p(\mathbf{x}_{0:k}^i|\mathbf{y}_{0:k}) &= \frac{p(\mathbf{y}_k|\mathbf{x}_k^i)p(\mathbf{x}_k^i|\mathbf{x}_{k-1}^i)}{p(\mathbf{y}_k|\mathbf{y}_{0:k-1})}p(\mathbf{x}_{0:k-1}^i|\mathbf{y}_{0:k-1}) \\ &\propto p(\mathbf{y}_k|\mathbf{x}_k^i)p(\mathbf{x}_k^i|\mathbf{x}_{k-1}^i)p(\mathbf{x}_{0:k-1}^i|\mathbf{y}_{0:k-1}) \end{aligned}$$

The importance density is given by:

$$q(\mathbf{x}_{0:k}^i|\mathbf{y}_{0:k}) = q(\mathbf{x}_k^i|\mathbf{x}_{k-1}^i, \mathbf{y}_k)q(\mathbf{x}_{0:k-1}^i|\mathbf{y}_{0:k-1}) \tag{3.36}$$

The importance weight is thus obtained by:

$$\begin{aligned} w_k^i &\propto \frac{p(\mathbf{y}_k|\mathbf{x}_k^i)p(\mathbf{x}_k^i|\mathbf{x}_{k-1}^i)p(\mathbf{x}_{0:k-1}^i|\mathbf{y}_{0:k-1})}{q(\mathbf{x}_k^i|\mathbf{x}_{k-1}^i, \mathbf{y}_k)q(\mathbf{x}_{0:k-1}^i|\mathbf{y}_{0:k-1})} \\ &= \frac{p(\mathbf{y}_k|\mathbf{x}_k^i)p(\mathbf{x}_k^i|\mathbf{x}_{k-1}^i)}{q(\mathbf{x}_k^i|\mathbf{x}_{k-1}^i, \mathbf{y}_k)} \cdot \frac{p(\mathbf{x}_{0:k-1}^i|\mathbf{y}_{0:k-1})}{q(\mathbf{x}_{0:k-1}^i|\mathbf{y}_{0:k-1})} \\ &= \frac{p(\mathbf{y}_k|\mathbf{x}_k^i)p(\mathbf{x}_k^i|\mathbf{x}_{0:k-1}^i)}{q(\mathbf{x}_k^i|\mathbf{x}_{k-1}^i, \mathbf{y}_k)}w_{k-1}^i \end{aligned}$$

Therefore, the weight of the i-th particle at time k is calculated recursively by:

$$w_k^i \propto \frac{p(\mathbf{y}_k|\mathbf{x}_k^i)p(\mathbf{x}_k^i|\mathbf{x}_{k-1}^i)}{q(\mathbf{x}_k^i|\mathbf{x}_{k-1}^i, \mathbf{y}_k)}w_{k-1}^i \tag{3.37}$$

Note that the importance density $q(\mathbf{x}_k^i|\mathbf{x}_{k-1}^i, \mathbf{y}_k)$ is chosen by the user, the likelihood $p(\mathbf{y}_k|\mathbf{x}_k^i)$ is obtained via the measurement model, and the prior $p(\mathbf{x}_k^i|\mathbf{x}_{k-1}^i)$ is obtained via the process model. The joint posterior can thus be estimated as:

$$\hat{p}(\mathbf{x}_{0:k}|\mathbf{y}_{0:k}) \triangleq \sum_{i=1}^{n_p} w_k^i \delta(\mathbf{x}_{0:k} - \mathbf{x}_{0:k}^i) \tag{3.38}$$

The marginal joint posterior $p(\mathbf{x}_k|\mathbf{y}_{0:k})$ is obtained by integrating the joint posterior pdf:

$$\begin{aligned}\hat{p}(\mathbf{x}_k|\mathbf{y}_{0:k}) &= \int \hat{p}(\mathbf{x}^i_{0:k}|\mathbf{y}_{0:k})d\mathbf{x}_{0:k-1} \\ &= \int \sum_{i=1}^{n_p} w^i_k \delta(\mathbf{x}_{0:k} - \mathbf{x}^i_{0:k})d\mathbf{x}_{0:k-1} \\ &= \sum_{i=1}^{n_p} w^i_k \int \delta(\mathbf{x}_{0:k} - \mathbf{x}^i_{0:k})d\mathbf{x}_{0:k-1} \\ &= \sum_{i=1}^{n_p} w^i_k \delta(\mathbf{x}_k - \mathbf{x}^i_k)\end{aligned}$$

This is the procedure for Sequential Importance Sampling (SIS). The steps are summarized below.

Sequential Importance Sampling (SIS) Algorithm

1. For $i = 1 : n_p$

   (a) Draw n_p particles from the prior: $\mathbf{x}^i_0 \sim p(\mathbf{x}_0)$ and set $w^i_0 = \frac{1}{n_p}$

   (b) Predict: Draw n_p new particles $\mathbf{x}^i_k$ from the importance density:

$$\mathbf{x}^i_k \sim q(\mathbf{x}_k|\mathbf{x}^i_{k-1}, \mathbf{y}_k) \tag{3.39}$$

   (c) Update: Calculate unnormalized weights:

$$\tilde{w}^i_k = \frac{p(\mathbf{y}_k|\mathbf{x}^i_k)p(\mathbf{x}^i_k|\mathbf{x}^i_{k-1})}{q(\mathbf{x}^i_k|\mathbf{x}^i_{0:k-1}, \mathbf{y}_{0:k})} w^i_{k-1} \tag{3.40}$$

   (d) Calculate normalized weights:

$$w^i_k = \frac{\tilde{w}^i_k}{\sum_{i=1}^{n_p} \tilde{w}^i_k} \tag{3.41}$$

The Degeneracy Problem and Resampling

The SIS particle filter suffers from the degeneracy problem in which after a few iterations, all but one particle will have a negligible weight and the variance of the importance weights increase over time, thereby reducing prediction accuracy Tulsyan *et al.* (2016). As well, the weights will be unevenly distributed when there is an outlier Pitt and Shephard (1999). The effective sample size N_{eff} provides a measure of degeneracy and is given as

$$\hat{N}_{eff} = \frac{n_p}{\sum_{i=1}^{n_p}(w_k^i)^2} \tag{3.42}$$

If the weights are all equal such that $w_k^i = \frac{1}{n_p}$, then the effective sample size is given as

$$\hat{N}_{eff} = \frac{1}{\sum_{i=1}^{n_p}(\frac{1}{n_p})^2} = n_p \tag{3.43}$$

Severe degeneracy occurs when $N_{eff} \leq n_p$. There are two ways to mitigate the effects of the degeneracy problem Tulsyan *et al.* (2016). The first is by implementing resampling and the second is by good choice of importance density.

Resampling the distribution eliminates the particles with low importance weights and replicates the particles with high importance weights. Whenever N_{eff} drops below some threshold. The new particles $\{\mathbf{x}_k^{i*}\}$ are drawn with a probability given by the importance weights:

$$p(\mathbf{x}_k^{i*} = \mathbf{x}_k^j) = w_k^j \tag{3.44}$$

and uniform weights are assigned to each of the resampled particles $w_k^j = \frac{1}{M}$ where $\{w_k^j\}_{j=1}^m$ are the importance weights for the resampled particles Tulsyan *et al.* (2016). There are many efficient implementations of the resampling scheme, the most popular one being systematic resampling, as described in Tulsyan *et al.* (2016).

Choice of Importance Density

Another topic of consideration with particle filtering is the choice of the importance density. A good choice of importance density will also mitigate the effects of the degeneracy problem. The most straightforward and common choice for the importance density is the prior Arulampalam *et al.* (2002a):

$$q(\mathbf{x}_k|\mathbf{x}^i_{0:k-1}, \mathbf{y}_{0:k}) = p(\mathbf{x}_k|\mathbf{x}^i_{k-1}) \tag{3.45}$$

Inserting equation (4.44) into the weight update equation yields:

$$\begin{aligned} w^i_k &= w^i_{k-1}\frac{p(\mathbf{y}_k|\mathbf{x}^i_k)p(\mathbf{x}^i_k|\mathbf{x}^i_{k-1})}{p(\mathbf{x}^i_k|\mathbf{x}^i_{k-1})} \\ &= w^i_{k-1}p(\mathbf{y}_k|\mathbf{x}^i_k) \end{aligned}$$

The optimal choice for the importance density is the one that minimizes the variance of the weights across $q(\mathbf{x}_k|\mathbf{x}^i_{0:k-1}, \mathbf{y}_{0:k})$. The importance weight mean is given by:

$$\begin{aligned} \mu_w &= \mathbb{E}[w^i_k] \\ &= \int w^i_k q(\mathbf{x}_k|\mathbf{x}^i_{0:k-1}, \mathbf{y}_{0:k})d\mathbf{x}_k \\ &= w^i_{k-1}\int \frac{p(\mathbf{y}_k|\mathbf{x}^i_k)p(\mathbf{x}_k|\mathbf{x}^i_{k-1})}{q(\mathbf{x}^i_k|\mathbf{x}^i_{0:k-1}, \mathbf{y}_{0:k})}q(\mathbf{x}_k|\mathbf{x}^i_{0:k-1})d\mathbf{x}_k \\ &= w^i_{k-1}\int p(\mathbf{y}_k|\mathbf{x}^i_k)p(\mathbf{x}_k|\mathbf{x}^i_{k-1}d\mathbf{x}_k \\ &= w^i_{k-1}\int p(\mathbf{y}_k|\mathbf{x}_k, \mathbf{x}^i_{k-1})p(\mathbf{x}_k|\mathbf{x}^i_{k-1})d\mathbf{x}_k \\ &= w^i_{k-1}\int p(\mathbf{y}_k, \mathbf{x}_k|\mathbf{x}^i_{k-1})d\mathbf{x}_k \\ &= w^i_{k-1}\int p(\mathbf{y}_k|\mathbf{x}^i_{k-1}) \end{aligned}$$

The importance weight variance is given by:

$$\begin{aligned}
\sigma_w^2 &= \mathbb{E}[(w_k^i - \mu_w^2)] \\
&= \mathbb{E}[(w_k^i)^2] - \mu_w^2 \\
&= \int (w_k^i)^2 q(\mathbf{x}_k|\mathbf{x}_{0:k-1}^i, \mathbf{y}_{0:k}) d\mathbf{x}_k - \mu_w^2 \\
&= (w_k^i)^2 \int \frac{[p(\mathbf{y}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{x}_{k-1}^i)]^2}{[q(\mathbf{x}_k|\mathbf{x}_{0:k-1}^i, \mathbf{y}_{0:k})]^2} q(\mathbf{x}_k|\mathbf{x}_{0:k-1}^i, \mathbf{y}_{0:k}) d\mathbf{x}_k - \mu_w^2 \\
&= (w_{k-1})^2 \int \frac{[p(\mathbf{y}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{x}_{k-1}^i)]^2}{q(\mathbf{x}_k|\mathbf{x}_{0:k-1}^i, \mathbf{y}_{0:k})} d\mathbf{x}_k - \mu_w^2
\end{aligned}$$

and given that $\mu_w^2 = (w_{k-1}^i)^2 p^2(\mathbf{y}_k|\mathbf{x}_{k-1}^i)$,

$$\sigma_w^2 = (w_{k-1}^i)^2 [\int \frac{[p(\mathbf{y}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{x}_{k-1}^i)]^2}{q(\mathbf{x}_k|\mathbf{x}_{0:k-1}^i, \mathbf{y}_{0:k})} d\mathbf{x}_k - p^2(\mathbf{y}_k|\mathbf{x}_{k-1}^i)] \tag{3.46}$$

The importance density that minimizes the variance of the weights is $q(\mathbf{x}_k|\mathbf{x}_{0:k-1}^i, \mathbf{y}_{0:k}) = p(\mathbf{x}_k|\mathbf{x}_{k-1}^i, \mathbf{y}_k)$. Inserting this into (**??**) yields:

$$\begin{aligned}
\sigma_w^2 &= (w_{k-1}^i)^2 [\int \frac{[p(\mathbf{y}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{x}_{k-1}^i)]^2}{q(\mathbf{x}_k|\mathbf{x}_{0:k-1}^i, \mathbf{y}_k)} d\mathbf{x}_k - p^2(\mathbf{y}_k|\mathbf{x}_{k-1}^i)] \\
&= (w_{k-1}^i)^2 [\int \frac{[p(\mathbf{y}_k|\mathbf{x}_k, \mathbf{x}_{k-1}^i), p(\mathbf{x}_k|\mathbf{x}_{k-1}^i)]^2}{p(\mathbf{x}_k|\mathbf{x}_{k-1}^i, \mathbf{y}_k)} d\mathbf{x}_k - p^2(\mathbf{y}_k, \mathbf{x}_{k-1}^i)] \\
&= (w_{k-1}^i)^2 [\int \frac{[p(\mathbf{y}_k, \mathbf{x}_k|\mathbf{x}_{k-1}^i)]^2}{p(\mathbf{x}_k|\mathbf{x}_{k-1}^i, \mathbf{y}_k)} d\mathbf{x}_k - p^2(\mathbf{y}_k, \mathbf{x}_{k-1}^i)] \\
&= (w_{k-1}^i)^2 [\int \frac{[p(\mathbf{x}_k|\mathbf{x}_{k-1}^i, \mathbf{y}_k)p(\mathbf{y}_k|\mathbf{x}_{k-1})]^2}{p(\mathbf{x}_k|\mathbf{x}_{k-1}^i, \mathbf{y}_k)} d\mathbf{x}_k - p^2(\mathbf{y}_k, \mathbf{x}_{k-1}^i)] \\
&= (w_{k-1}^i)^2 [\int p(\mathbf{x}_k|\mathbf{x}_{k-1}^i, \mathbf{y}_k)p^2(\mathbf{y}_k|\mathbf{x}_{k-1}) d\mathbf{x}_k - p^2(\mathbf{y}_k|\mathbf{x}_{k-1}^i)] \\
&= (w_{k-1}^i)^2 [p^2(\mathbf{y}_k|\mathbf{x}_{k-1}) \int p(\mathbf{x}_k|\mathbf{x}_{k-1}^i, \mathbf{y}_k) d\mathbf{x}_k - p^2(\mathbf{y}_k|\mathbf{x}_{k-1}^i)] \\
&= (w_{k-1}^i)^2 [p^2(\mathbf{y}_k|\mathbf{x}_{k-1}) - p^2(\mathbf{y}_k|\mathbf{x}_{k-1}^i)] \\
&= 0
\end{aligned}$$

Therefore, $q(\mathbf{x}_k|\mathbf{x}^i_{0:k-1}, \mathbf{y}_{0:k}) = p(\mathbf{x}_k|\mathbf{x}^i_{k-1}, \mathbf{y}_k)$ is the optimal importance density. The optimal importance weight is thus given by:

$$\begin{aligned}
w^i_k &= \frac{p(\mathbf{y}_k|\mathbf{x}^i_k)p(\mathbf{x}^i_k|\mathbf{x}^i_{k-1})}{q(\mathbf{x}^i_k|\mathbf{x}^i_{0:k-1}, \mathbf{y}_k)} w^i_{k-1} \\
&= \frac{p(\mathbf{y}_k|\mathbf{x}^i_k, \mathbf{x}^i_{k-1})p(\mathbf{x}^i_k|\mathbf{x}^i_{k-1})}{p(\mathbf{x}^i_k|\mathbf{x}^i_{k-1}, \mathbf{y}_{k-1})} w^i_{k-1} \\
&= \frac{(p(\mathbf{y}_k, \mathbf{x}^i_k|\mathbf{x}^i_{k-1})}{p(\mathbf{x}^i_k|\mathbf{x}^i_k, \mathbf{y}_k)} w^i_{k-1} \\
&= \frac{p(\mathbf{x}^i_k|\mathbf{x}^i_{k-1}, \mathbf{y}_k)p(\mathbf{y}_k|\mathbf{x}^i_{k-1})}{p(\mathbf{x}^i_k|\mathbf{x}^i_{k-1}, \mathbf{y}_k)} w^i_{k-1} \\
&= p(\mathbf{y}_k|\mathbf{x}^i_{k-1}) w^i_{k-1}
\end{aligned}$$

This optimality criterion minimizes the conditional variance so that the i-th weight is not affected by the variation in $\mathbf{x}_k$. However, there are two problems with this choice of importance density. The first concerns how to draw samples from $p(\mathbf{x}_k|\mathbf{x}^i_{k-1}, \mathbf{y}_k)$. The second is evaluating the integral $p(\mathbf{y}_k|\mathbf{x}^i_{k-1}) = \int p(\mathbf{y}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{x}^i_{k-1})d\mathbf{x}_k$ Arulampalam *et al.* (2002a); McNames (2016). Thus, the use of the optimal importance density is typically not possible, except for a few cases, so for now, we will use the prior as the importance density of choice.

Chapter 4

CURRENT APPROACHES TO MODELING AND PROCESSING GENE REGULATORY NETWORKS

A gene regulatory network (GRN) can be represented as a graph $G = (V, E)$. $\{V\}$ is the set of nodes, or genes, and $\{E\}$ is the set of edges, or interactions between genes. In a graph consisting of N nodes, the expression levels of each of the genes $(X_1, ..., X_N)$ are random variables. An edge $X_i \rightarrow X_j$ denotes a regulatory relationship. The primary goal of gene regulatory network inference is to learn the structure of the graph that is encoded in the microarray dataset, called the *target network* Delgado and Gómez-Vela (2019). The resulting graph can is called the *predicted network*. The predicted network can consist of *undirected*, as shown in Figure 4.1, or *directed* edges. The edges also describe the different types of regulation, which are *activating* or *inhibiting* Saint-Antoine and Singh (2020). Simple examples showing these regulation types in a directed network are shown in Figure 4.2 and Figure 4.3. As it can be seen in Figure 4.2, the arrow indicates that the expression level of gene X_1 activates the expression of gene X_2. Conversely, in Figure 4.3, the vertical bar indicates that the expression level of gene X_1 inhibits the expression level of gene X_2.

Figure 4.1: An undirected two-gene network. The line indicates a regulatory relationship between gene X_1 and gene X_2, but there is no information regarding the type nor direction of regulation.

Figure 4.2: A directed two-gene network demonstrating an activating regulatory relationship. The arrow indicates that gene X_1 activates the expression of gene X_2.

Figure 4.3: A directed two-gene network demonstrating an inhibiting regulatory relationship. The vertical bar indicates that gene X_1 inhibits the expression of gene X_2.

Undirected networks are the simplest to construct, but may not be suitable for purposes such as targeted drug design in which mechanistic or causal information is necessary. Directed networks are more difficult to construct, and reconstructing directed networks with the regulation type remains a challenge Saint-Antoine and Singh (2020). An even greater challenge is reconstructing time-varying gene regulatory networks, in which the regulatory interactions change over time. The output of this task is a temporally-indexed collection of networks, which can describe the various stages of biological processes. Such a task can of course be made more difficult by including both the direction and type of regulation.

In this chapter we present a literature survey of the various approaches for modeling and processing gene regulatory networks. We describe approaches suited for the reconstruction of undirected and directed networks, networks denoting the regulation type, as well as time-varying GRNs. We use the terms *time-varying* and *static* to describe predicted networks in which the regulatory interactions do and do not

change over time, respectively. The output of a time-varying GRN is a time series of graphs $G_1, G_2, ..., G_T$ for $t = 1, ..., T$. This is to not be confused with static methods for inferring GRNs, which employ steady-state data in GRN reconstruction, versus dynamic methods, which use time series data to reconstruct a GRN.

4.1 Modeling Gene Regulatory Networks

One of the main focuses of modern biology is reconstructing the molecular networks that encode biological processes. With the increasing availability of -omics data over the past decade, research efforts have aimed at extracting meaningful information from these data with the goal of inferring regulatory interactions from gene expression patterns. Models for reconstructing GRNs can broadly be categorized into four main types: 1) co-expression networks, 2) probabilistic models, 3) dynamical models, and 4) Bayesian nonparametric models, the latter of which integrates elements of several modeling approaches.

4.1.1 Co-expression Networks

Co-expression methods are based on the assumption that if the expression profiles of genes are fluctuating at the same time points, then there may be a regulatory relationship between those genes and those genes may be involved in performing the same biological function Pyne (2020); Bellot Pujalte (2017). In general, these methods use a linear similarity measure, such as the Pearson correlation measure $s_{ij} = |cor(i, j)|$ or one of its variants to construct a matrix of pairwise gene correlations. The resulting *similarity matrix* is denoted by $S = [s_{ij}]$. Next, a threshold is applied to select genes with a high correlation from which an adjacency matrix is constructed. This method produces a weighted, undirected static network Sanguinetti and Huynh-Thu (2019); Serin *et al.* (2016).

Co-expression methods have several advantages. First, they are simple to implement and scalable to large datasets Pyne (2020); Saint-Antoine and Singh (2020). Next, these methods enable accurate reconstruction even when the number of samples is to the order of tens of thousands less than the number of variables Pyne (2020). Third, they are accurate in reconstructing several common network motifs, such as feedforward loops, fan-ins, and fan-outs Saint-Antoine and Singh (2020). Finally, co-expression methods can provide a general overview of the network structure under investigation from which further analyses can be conducted Saint-Antoine and Singh (2020).

Despite these advantages, co-expression methods have several limitations. The main drawback is that correlation does not imply causation. As is summarized in Sanguinetti and Huynh-Thu (2019) and Saint-Antoine and Singh (2020) correlation networks cannot distinguish between direct and indirect interactions, and they are prone to false positives. They are also unable to capture the nonlinearities that are highly characteristic of GRNs due to the linearity imposed by the Pearson correlation Mercatelli *et al.* (2020); Sanguinetti and Huynh-Thu (2019). As well, these methods cannot construct directed networks due to the symmetry of the correlation measure. However, methods which address the issue of causality are reviewed in Serin *et al.* (2016), which leverage the use of time series data to construct causal co-expression networks.

4.1.2 Information Theoretic Methods

Information theoretic methods address the issues posed by the Pearson correlation through the use of the mutual information measure Margolin *et al.* (2006). The mutual

information of two random variables X_i and X_j is given by

$$I(X_i, X_j) = \sum_{x_i \in X_i} \sum_{x_j \in X_j} p(x_i, x_j) \log \frac{p(x_i, x_j)}{p(x_i)p(x_j)} \tag{4.1}$$

Similar to correlation-based methods, the mutual information is calculated for each pair of genes. If $I(X_i, X_j)$ is above a user-specified threshold, then an edge is constructed between genes i and j Bellot Pujalte (2017).

Information theoretic methods share the same advantages of correlation-based methods, including detection of common network motifs, their easy implementation, and scalability to whole-genome networks. The primary advantage of these approaches over correlation-based methods is that they can capture the non-linear relationships which are characteristic of GRNs. As such, this method tends to outperform linear correlation-based methods Pyne (2020); Saint-Antoine and Singh (2020). However, because mutual information is also a symmetric measure, these methods result in undirected networks. As well, they tend to produce a high number of false positives Saint-Antoine and Singh (2020).

4.1.3 Probabilistic Models

Probabilistic models address the issue of distinguishing between direct and indirect regulatory interactions posed by co-expression networks. Models of the data and parameters are formulated as probabilities Sanguinetti and Huynh-Thu (2019). An edge between two genes establishes conditional dependence between the genes Zhao and Duan (2019). We review three of the most widely-used probabilistic modeling approaches, which are Gaussian Graphical Models (GGMs), Bayesian Networks, and Bayesian hierarchical models.

Gaussian Graphical Models

In a Gaussian Graphical Model (GGM), the probability density for the nodes $\mathbf{X} = (X_1, ..., X_n)^T$ is a multivariate normal distribution given by

$$p(\mathbf{X}|\mathbf{m}, \Sigma) = \frac{1}{\sqrt{2\pi|\Sigma|}} \exp\Big[-\frac{1}{2}(\mathbf{X} - \mathbf{m})^T \Sigma^{-1} (\mathbf{X} - \mathbf{m}) \Big] \tag{4.2}$$

where $\mathbf{m}$ is the mean vector, Σ is the covariance matrix, and $\Sigma^{-1} = \mathbf{\Theta}$ is the precision matrix. The precision matrix encodes the conditional dependence structure of the network and contains the partial correlations between pairs of genes Sanguinetti and Huynh-Thu (2019); Tian *et al.* (2016). Two genes are conditionally independent if the corresponding entry in the precision matrix is zero Xu *et al.* (2018). The goal is to estimate the precision matrix, which usually done through maximum likelihood methods.

To ensure sparsity, which is a property that is realistic to biological networks, penalized regularization techniques, such as Lasso, are often employed. Lasso is a sparse L1 regularization technique that has been used to estimate gene regulatory networks Sanguinetti and Huynh-Thu (2019). In Zhao and Duan (2019) GGMs are used in conjunction with graphical Lasso are used to infer regulatory networks in 15 different human cancers.

In Hallac *et al.* (2017), the authors introduce a time-varying graphical Lasso to infer time-varying networks from time series data, which involves estimating the precision matrix $\Theta_i = \Sigma(t_i)^{-1}$ at each time step t_i. An advantage of this approach is that it captures different types of network dynamics, including the rewiring of a small set of edges or an entire network rewiring of its edges during a single time step Hallac *et al.* (2017). This is done by considering different types of penalty functions $\psi(\cdot)$

when solving the convex optimization problem given by

$$\underset{\Theta \in \mathbf{S}^p_{++}}{\text{minimize}} \sum_{i=1}^{T} -l_i(\Theta_i) + \lambda ||\Theta_i||_{\text{od},1} + \beta \sum_{i=2}^{T} \psi(\Theta_i - \Theta_{i-1}) \tag{4.3}$$

where $\Theta \in \mathbf{S}^p_{++}$ denotes the condition that the precision matrix Θ must be positive semi-definite. As well, $l_i(\Theta_i) = n_i(\text{logdet}\Theta_i - \text{Tr}(\mathbf{S}_i\Theta_i))$, where Tr is the trace, and $\mathbf{S}_i$ is the empirical covariance matrix at time t_i. To solve the convex optimization problem, a message-passing algorithm is used that is based on the Alternating Direction Method of Multipliers (ADMM). The algorithm is used to infer time-varying financial networks and automobile sensor data, and capture structural changes in the networks corresponding to both datasets. A similar approach, the LOcal Group Graphical Lasso Estimation, is based on the assumption that the graph changes smoothly over time Yang and Peng (2020). The smoothness is imposed through a local group-lasso penalty. The goal is to estimate the precision matrix $\mathbf{\Theta}(t) := \Sigma^{-1}(t)$ based on the observed data $\{\mathbf{x}_k\}_{k\in\mathcal{I}}$, where $\mathcal{I} = \{1, ..., N\}$ and $\mathbf{x}_k$ is a realization of $\mathbf{X}(t_k)$. $\mathbf{X}(t) = (X^1(t), ..., X^p(t))^T$ is a p-dimensional Gaussian random vector with mean $\mu(t)$ and covariance matrix $\Sigma(t)$. The precision matrix at each k-th time point $\mathbf{\Omega}(t_k)$ is estimated by minimizing a locally weighted negative log-likelihood function with a local group-Lasso penalty given by:

$$L(\lessgtr_k) := \frac{1}{\sqrt{|\mathcal{N}_{k,d}|}} \sum_{i\in\mathcal{N}_{k,d}} [\text{tr}(\mathbf{\Omega}(t_i)\hat{\Sigma}(t_i)) - \log|\mathbf{\Omega}(t_i)|] + \lambda \sum_{u\neq v} \sqrt{\sum_{i\in\mathcal{N}_{k,d}} \Omega_{uv}(t_i)^2} \tag{4.4}$$

where $\mathcal{N}_{k,d}$ are the indices of the time points centered around t_k with neighborhood width d, $\lessgtr_k$ is the set of precision matrices within this neighborhood, $\Omega_{uv}(t_i)$ is the (u, v)-th element of $\mathbf{\Omega}(t_i)$, and $\hat{\Sigma}(t)$ is the kernel estimate of the covariance matrix at time t. As in Hallac *et al.* (2017), the convex optimization problem is solved using an ADMM algorithm.

Despite the advantages of Gaussian graphical models in estimating time-varying networks with sparse measurements, the assumption of Gaussian distributed data

implies linear relationships between the genes Sanguinetti and Huynh-Thu (2019). As well, Gaussian graphical models are limited to reconstructing undirected graphs.

Bayesian Networks

Bayesian networks are also within the class of probabilistic models and are advantageous in their ability to construct directed networks. Steady-state gene expression patterns are modeled as random variables and the distribution of expression patterns is represented by their joint probability density Ho and Charleston (2011). The graph G is a directed acyclic graph (DAG) whose nodes (genes) correspond to the random variables $X_1, ..., X_n$ Ahmed *et al.* (2018). A static directed graph (from parent to child nodes) is constructed by factoring the joint probability density into a set of local conditional probability densities Sanguinetti and Huynh-Thu (2019) as

$$p(X_1, X_2, ..., X_n) = \prod_{i=1}^{n} p(X_i|\text{parents}(X_i)) \tag{4.5}$$

The conditional distributions are assumed to be Gaussian for the continuous case or multinomial for the discrete case Sanguinetti and Huynh-Thu (2019). Bayesian networks follow the Markov assumption so that each variable is independent of its non-descendants given its parents Banf and Rhee (2017). The main steps in learning Bayesian networks are 1) model selection, 2) parameter learning, and 3) model scoring Delgado and Gómez-Vela (2019) Model selection is the process of reconstructing the network topology which best suits the data Banf and Rhee (2017). Parameter learning requires estimating the best probability values for each node. Model scoring is usually done through Bayesian information criterion (BIC) or Akaike's information criterion. The higher the score, the better the model fits the data Delgado and Gómez-Vela (2019); Banf and Rhee (2017)

Some advantages of Bayesian networks is that they are particularly useful when

combining different types of data and are also less sensitive to noise Mercatelli *et al.* (2020). As well, their ability to encode prior information is suitable for correct network identification. They can also encode prior knowledge over possible network structures Werhli and Husmeier (2007). However, Bayesian networks do suffer a few drawbacks. For one, they are designed for directed acyclic graphs (DAGs), though more recent work has focused on extending Bayesian networks to graphs with cycles de Luis Balaguer and Sozzani (2017). Furthermore due to the non-uniqueness of the factorization of the joint probability distribution, multiple network configurations can encode the same probability distributions. In addition, there is a high computational cost associated with Bayesian networks, especially if one relaxes the DAG constraint Mercatelli *et al.* (2020); Sanguinetti and Huynh-Thu (2019), making them unsuitable for large datasets Saint-Antoine and Singh (2020). Local Bayesian Networks are introduced in Liu *et al.* (2016) to overcome this issue. This method first uses conditional mutual information (CMI) to construct an initial GRN which is then decomposed into several local GRNs Liu *et al.* (2016). Then the initial GRN is decomposed into a number of local networks. This method reduces the search space of possible network structures while also reducing the number of false positives Liu *et al.* (2016). However, the main disadvantage of Bayesian networks is that a directed edge only represents a probabilistic dependency, not a causal interaction de Campos *et al.* (2019).

Dynamic Bayesian Networks

Standard Bayesian networks rely on the use of steady state data, which for accurate inference, requires a large number of replicates which may not be available Michailidis and d'Alché Buc (2013). As such, it is more difficult to extract structural information from this type of data in comparison to time series data. Time series data,

though more difficult to obtain, encodes dynamical information that can aid in the reconstruction of GRNs. Dynamic Bayesian networks (DBNs) are an extension of Bayesian networks that are designed for GRN inference using time series data. The main difference between DBNs and standard Bayesian networks is that a node is a gene at a specific time slice Saint-Antoine and Singh (2020). The likelihood of the observations (gene expression levels) is given as:

$$p(\mathbf{X}_1, ..., \mathbf{X}_T) = p(\mathbf{X}_0) \prod_{t=1}^{T} p(\mathbf{X}_t|\mathbf{X}_{t-1}) = p(\mathbf{X}_0) \prod_{t=1}^{T} \prod_{i=1}^{N} p(X_{i,t}|\mathbf{X}_{\pi_i,t-1}) \tag{4.6}$$

where $\mathbf{X}_{\pi_i,t-1} := \{X_{j,t-1} : X_{j,t-1} \text{ regulates } X_{i,t}\}$ Song *et al.* (2009). DBNs reconstruct static directed networks. The edges in a DBN denote regulatory relationships between two consecutive time slices, but also allows for intra-time slice connections Chai *et al.* (2014). This allows one to capture cyclic dependencies, thereby alleviating the DAG restriction. They have been used to infer GRNs underlying cancer cell phenotypes Adabor and Acquaah-Mensah (2019); Suter *et al.* (2022). Another advantage of DBNs is that a time delay can be incorporated into the model through the use of high order DBN models, facilitating more realistic regulatory network dynamics Ahmed *et al.* (2018); Li *et al.* (2015, 2014). However, DBNs suffer from increased computational complexity because all genes are considered to be potential regulators for a given target gene making them unsuitable for large networks Li *et al.* (2014). The problem of causality posed by Bayesian networks has been addressed in Pearl (2000) by introducing the concept of a causal Bayesian network, but this requires accessibility to gene knock-out, loss/gain-of-function experimental data.

A particular notable application of DBNs is for inference of time-varying GRNs. In Song *et al.* (2009), the authors employ a kernel reweighted L1-regularized autoregressive approach, which allows for the network reconstruction problem to be decomposed into a series of simpler sub problems. Their work is based on the assumption of a

sparse network structure which smoothly varies across time. This means that each temporal snapshot of the network is not a completely different network, but rather shares common features of the network snapshot at adjacent time points. Time homogeneity is also assumed, which is unable to capture transient regulatory interactions which are present for a short duration Song *et al.* (2009); Pyne (2020), though this issue is addressed in Robinson and Hartemink (2008); Grzegorczyk and Husmeier (2009). The transition model $p(\mathbf{X}_t|\mathbf{X}_{t-1})$ becomes time dependent as $p_t(\mathbf{X}_t|\mathbf{X}_{t-1})$. The authors applied their kernel reweighted L1-regularized autoregressive procedure to inference of time-varying yeast GRNs and neural networks. The output of this method is a directed network for each time slice under consideration. An extension of this method called TESLA was introduced in Ahmed and Xing (2009). Another method is ARTIVA Lebre *et al.* (2010), which uses a DBN in conjunction with a dynamical systems model for gene regulation to infer time-varying GRNs. ARTIVA infers the change points of the regulatory inputs which describe where transcriptional changes occur in the network. Reversible jump MCMC is used for sampling from the posterior. Their approach was able to reconstruct GRNs describing the life cycle of *D. melanogaster*. Although this approach captures the time-varying nature of GRNs along with the direction of regulation, it does not capture the regulation type. The ability to reconstruct time-varying GRNs is an important contribution as it can aid our understanding of disease progression and which regulatory events underly the transition from healthy to diseased states.

Bayesian Hierarchical Models

A Bayesian hierarchical model is a statistical model that allows for the sharing of information across groups of data. This is achieved through the specification of a prior distribution from which the parameters corresponding to each group can be sampled.

This is referred to as the *population distribution* Gelman *et al.* (1995a). The hierarchy is formed by specifying hyperparameters for the prior distributions on the parameters, which themselves can be sampled from distributions with their own hyperparameters. This results in a multi-level structure that can capture the structure of the data in sufficient complexity. For example, we may be interested in grouping hospitals according to the health outcomes of their patients while simultaneously grouping the patients within each hospital. Or we may want to group schools according to the distribution of SAT scores while simultaneously grouping students according to their SAT scores. Bayesian hierarchical modeling therefore provides a flexible framework for a range of problems.

Several approaches for inferring GRNs using Bayesian hierarchical models have been introduced in the literature. In Panchal and Linder (2020), the authors propose a Bayesian hierarchical model with global-local shrinkage priors for reconstructing GRNs. The advantage of their approach is their use of the student's-t distribution for the likelihood, which enables the representation of heavy-tailed data. A linear model is used to represent the GRN dynamics and the additive noise term is modeled as a multivariate Gaussian mixture. Their approach was able to reconstruct several GRNs involved in T-cell activation. The network edges capture both the direction and type of regulation. In Cortez *et al.* (2022), the authors introduce a non-Markovian Bayesian hierarchical model for inferring the parameters of biochemical processes described by stochastic delayed birth-death processes. Gamma priors are placed over the rate parameters. They applied their framework to *E. coli* fluorescent protein synthesis data to estimate the rate parameters. In Jensen *et al.* (2007), the authors introduce a Bayesian variable selection framework which integrates three different types of biological data to reconstruct GRNs. The data include gene expression data, chromatin-immuno precipitation data (ChIP), and promoter sequence

data. They introduce a linear model for the gene regulatory network dynamics which captures the combinatorial action of transcription factors. Gaussian priors are placed on the baseline gene expression parameter, the transcription factor linear effects, and interaction effects parameters. Gibbs sampling is used for inference. They applied their framework to infer gene-transcription factor networks in yeast, resulting in a directed network which captures the regulation type. In Thompson *et al.* (2020), the authors develop a Bayesian hierarchical mixture model to infer gene expression states from replicate transcriptomic libraries. One of the advantages of their approach is that it captures complexity in both the biological process of gene regulation and the technical processes used to collect the data. Their approach was able to infer the correct expression states in both simulated and real-world datasets. Lastly, in Liang and Kelemen (2016), the authors introduce a fully Bayesian state-space approach to infer time-varying gene regulatory networks in different tissues. They use a linear state space model with time-varying state transition matrices which encode the gene-gene interactions. The hidden state distribution, which is the distribution for the gene expression states, and the measurement likelihood are Gaussian. Multivariate Gaussian priors are placed over the time-varying transition matrices with an inverse G-Wishart prior on the time-varying covariance matrix. Gibbs sampling is used for inference. Their approach was applied to both simulated and real-world data. Their results demonstrate the efficacy of Bayesian hierarchical modeling in inferring hidden variables which affect the process of gene expression. However, a disadvantage of this approach is that the model does not directly encode the direction or type of regulation. Other Bayesian approaches include Bayesian model selection, which are discussed in Chapter 6.

4.1.4 Bayesian Nonparametric Methods

More recent state-of-the-art methods leverage the statistical power of Bayesian nonparametric priors. These methods allow for the complexity of the model to grow and adapt to the data. This is achieved by placing a prior π on an infinite dimensional probability space $(\Omega, \mathcal{F}, P)$ Moraffah (2019). They often employ Dirichlet processes or Gaussian processes as priors in mixture models. The approaches discussed in this section combine elements from other methods introduced in this chapter. We review the main concepts of Bayesian nonparametric methods and their applications to GRN inference in this section. Most of the background information from this section can be found in Jordan and Teh (2015).

Dirichlet Processes

Let $(\Theta, \mathcal{A})$ be a measurable space and α be a positive real number. The Dirichlet Process is a random probability measure over the space Θ Ferguson (1973). A draw from a Dirichlet process $G(\cdot) \sim DP(H, \alpha)$ with base distribution H and concentration parameter α is a distribution over probability measures. For any partition $A_1, .., A_k$ of Θ

$$(G(A_1), ..., G(A_k)) \sim \text{Dir}(\alpha H(A_1), ..., \alpha H(A_k)) \tag{4.7}$$

where Dir is the Dirichlet distribution. For every $A \in \Theta$, the following properties hold

$$\mathbb{E}[G(A)] = H(A) \tag{4.8}$$

and

$$var[G(A)] = \frac{H(A)(1 - H(A)}{1 + \alpha} \tag{4.9}$$

As well, each realization of a Dirichlet process is discrete with probability one. In Bayesian nonparametric density estimation, the Dirichlet process is used as a prior

over the parameter space Θ. Given $G \sim DP(H, \alpha)$ and $\theta_k|G \overset{i.i.d}{\sim} G$ for $k = 1, ..., n$, it can be shown that the posterior distribution of $G|\theta_{1:n}$ is also a Dirichlet process:

$$G|\theta_{1:n} \sim DP(\frac{\alpha}{\alpha+n}H + \frac{1}{\alpha+n}\sum_{k=1}^{n}\delta_{\theta_k}, \alpha+n) \tag{4.10}$$

Blackwell and Macqueen Blackwell and MacQueen (1973) showed that by integrating out the random measure G, the predictive distribution has the following Polyá urn form

$$\theta_{n+1}|\theta_{1:n} \sim \frac{n}{\alpha+n}\sum_{k=1}^{n}\delta_{\theta_k} + \frac{\alpha}{\alpha+n}H \tag{4.11}$$

This states that with probability $\frac{n}{\alpha+n}$, the new sample θ_{n+1} is equal to one of the previous samples, and with probability $\frac{\alpha}{\alpha+n}$, draw a new sample i.i.d from H.

Stick-Breaking Construction

Sethuraman Sethuraman (1994) showed that a draw from a Dirichlet Process can be represented as

$$\beta_k \overset{i.i.d}{\sim} \text{Beta}(1, \alpha) \qquad \theta_k \overset{i.i.d}{\sim} H$$
$$\pi_k = \beta_k \prod_{l=1}^{k-1}(1-\beta_k) \qquad G = \sum_{k=1}^{\infty}\pi_k\delta_{\theta_k}$$

where $\{\pi_k\}_{k=1}^{\infty}$ are the weights and $\{\theta_k\}_{k=1}^{\infty}$ are the atom locations. This is referred to as the *stick-breaking* construction because the weights π_k are constructed in a manner analogous to breaking off a unit length stick.

Dirichlet Process Mixture Models (DPMM)

A Dirichlet Process mixture model (DPMM) is a generalization of a finite mixture model to an infinite mixture model where a Dirichlet process is used as a prior probability over the infinite-dimensional parameter space. Let $x_1, .., x_n$ be i.i.d. random

variables drawn from a distribution $F(\cdot)$ with density $f(\cdot|\theta)$. The goal is to estimate the density $f(\cdot|\theta)$. The DPMM is represented by the following hierarchy:

$$G \sim DP(H, \alpha) \tag{4.12}$$

$$\theta_k|G \sim G \tag{4.13}$$

$$x_k|\theta_k \sim f(\cdot|\theta_k) \tag{4.14}$$

$$\tag{4.15}$$

An equivalent representation is obtained by marginalizing G. The hierarchy is given by:

$$\pi|\alpha \tag{4.16}$$

$$\theta_k|H \overset{i.i.d}{\sim} H \tag{4.17}$$

$$z_k|\pi \sim \text{Categorical}(\pi) \tag{4.18}$$

$$x_k|\theta_k, z_k \sim f(\cdot|\theta_{z_k}) \tag{4.19}$$

$$\tag{4.20}$$

where z_k is an indicator variable denoting the cluster assignments.

Chinese Restaurant Process

The *Chinese Restaurant Process* (CRP) is an alternate characterization of the Dirichlet Process. It is a probability distribution over partitions. Consider a partition of N points $\boldsymbol{\pi}_{[N]}$. Each subset in the partition corresponds to a cluster. The main premise behind the CRP is as follows. Consider a Chinese restaurant with an infinite number of tables. The first customer sits at the first table with probability one. The subsequent customers can either join the first table with probability proportional to the number of customers sitting at the table or start a new table with probability

proportional to α, as:

$$\mathbb{P}(\text{customer } n+1 \text{ joins table } c) = \frac{|c|}{\alpha + N} \tag{4.21}$$

$$\mathbb{P}(\text{customer } n+1 \text{ starts a new table } c) = \frac{\alpha}{\alpha + N} \tag{4.22}$$

Note that α is the concentration parameter of the Dirichlet process. A draw from a CRP $\boldsymbol{\pi}_{[N]} \sim \text{CRP}(\alpha, N)$ defines an exchangeable partition over $[N]$. The probability of a partition is given as:

$$P(\boldsymbol{\pi}_{[N]}) = \frac{\alpha^K}{\alpha^{(N)}} \prod_{c \in \boldsymbol{\pi}_{[N]}} (|c| - 1)! \tag{4.23}$$

where K is the total number of clusters $c_1, c_2, ..., c_k$. The CRP also defines a mixture model as follows:

$$\boldsymbol{\pi}_{[N]} \sim \text{CRP}(\alpha, N) \tag{4.24}$$

$$\theta_c | \boldsymbol{\pi}_{[N]} \overset{i.i.d}{\sim} H \tag{4.25}$$

$$x_k | \theta, \boldsymbol{\pi}_{[N]} \overset{ind}{\sim} F(\theta_c) \tag{4.26}$$

$$\tag{4.27}$$

where θ_c are the parameters associated with each table and $F(\theta_c)$ is the distribution with density $f(\cdot|\theta_c)$.

Hierarchical Dirichlet Processes

The hierarchical Dirichlet process (HDP) is an extension of the Dirichlet process for clustering grouped data. Each group is associated with a mixture model and the HDP links these mixture models through statistical dependencies. Let x_{ji} denote the i-th observation in group j and assume that observations are exchangeable within and across groups. θ_{ji} is a parameter for the mixture component associated with observation x_{ji}. $F(\theta_{ji}$ is the distribution of x_{ji} given θ_{ji}. G_0 is a global random

probability measure distributed as a Dirichlet process with concentration parameter γ and base measure H. The group-specific random measures are denoted as G_j and are Dirichlet Process-distributed with base measure G_0 and concentration parameter α_0 Teh *et al.* (2004). The hierarchical model is given as:

$$G_0|\gamma, H \sim \mathrm{DP}(\gamma, H) \tag{4.28}$$

$$G_j|\alpha_0, G_0 \sim \mathrm{DP}(\alpha_0, G_0) \text{ for each } j \tag{4.29}$$

$$\theta_{ji}|G_j \sim G_j \tag{4.30}$$

$$x_{ji}|\theta_{ji} \sim F(\theta_{ji}) \tag{4.31}$$

$$\tag{4.32}$$

The set of atoms at the top level is shared by the bottom level Dirichlet processes. It is this property which enables clustering between groups through sharing of the cluster parameters.

Stick-Breaking Construction for Hierarchical Dirichlet Processes

Like the standard Dirichlet Process, hierarchical Dirichlet processes can be equivalently represented through a stick-breaking construction. The global random probability measure G_0 can be expressed as:

$$G_0 = \sum_{k=1}^{\infty} \beta_k \delta_{\phi_k} \tag{4.33}$$

where $\phi_k \sim H$ independently and $\boldsymbol{\beta} = (\beta_k)_{k=1}^{\infty} \sim$ GEM(γ). For each j-th DP:

$$G_j = \sum_{k=1}^{\infty} \pi_{jk} \delta_{\phi_k} \tag{4.34}$$

where each $\boldsymbol{\pi}_j = (\pi_{jk})_{k=1}^{\infty}$ is independently distributed according to DP$(\alpha_0, \boldsymbol{\beta})$. The ϕ_k denote that atom locations and each G_j is conditionally independent given G_0. Each factor θ_{ji} takes on value ϕ_k with probability π_{jk}. We can represent this using

an indicator variable z_{ji} such that $\theta_{ji} = \phi_{z_{ji}}$. The stick-breaking construction of the HDP is given as:

$$\boldsymbol{\beta}|\gamma \sim \text{ GEM}(\gamma) \tag{4.35}$$

$$\phi_k|H \sim H \tag{4.36}$$

$$\boldsymbol{\pi}_j|\alpha_0, \boldsymbol{\beta} \sim DP(\alpha_0, \boldsymbol{\beta}) \tag{4.37}$$

$$z_{ji}|\boldsymbol{\pi}_j \sim \boldsymbol{\pi}_j \tag{4.38}$$

$$x_{ji}|z_{ji}, (\phi_k)_{k=1}^{\infty} \sim F(\phi_{z_{ji}}) \tag{4.39}$$

$$\tag{4.40}$$

The Chinese Restaurant Franchise

Another representation of the Hierarchical Dirichlet Process (HDP) is the Chinese Restaurant Franchise (CRF). The CRF is an extension of the standard CRP. The premise is as follows: consider a franchise of Chinese restaurants where each restaurant has an infinite number of tables with a shared menu across restaurants. Each restaurant corresponds to one DP G_j. Customers at each j-th restaurant sit at the tables in the same way as the CRP Jordan and Teh (2015). At each table of each restaurant, one dish is ordered by the first customer that sits there. Each customer who sits at that table shares the dish. Multiple tables in multiple restaurants can serve the same dish due to the shared menu across restaurants, which correspond to the atom locations of the G_0. Customers at each j-th restaurant are denoted by θ_{ji}. The global menu is (ϕ_k) where each ϕ_k is independently and identically distributed according to H. The dish served at table t in restaurant j is denoted by ψ_{jt}. Each parameter is associated with one ψ_{jt}, and each ψ_{jt} is associated with one atom ϕ_k. Let t_{ji} be the index of ψ_{jt} associated with parameter θ_{ji} and let k_{jt} be the index of ϕ_k associated with ψ_{jt}. n_{jtk} denotes the number of customers in restaurant j at table t

eating dish k, where there are K unique dishes served in the entire franchise Jordan and Teh (2015). The number of tables in restaurant j serving dish k is m_{jk}. Dots denote marginals (e.g. $m_{j\cdot}$ is the number of tables in restaurant j).

To compute the predictive distribution of $\theta_{ji}|\theta_{j1}, ..., \theta_{j,i-1}, G_0, G_j$ is integrated out:

$$\theta_{ji}|\theta_{j1}, ..., \theta_{j,i-1}, \alpha_0, G_0 \sim \sum_{t=1}^{m_{j\cdot}} \frac{n_{jt\cdot}}{i-1+\alpha_0}\delta_{\psi_{jt}} + \frac{\alpha_0}{i-1+\alpha_0}G_0 \tag{4.41}$$

The above equation assigns customers to tables. The first term on the RHS means that customer sits at a table that is already occupied and second term means that customer sits at a new table with probabilities given by the mixing proportions Teh *et al.* (2004). If first term is chosen, increment n_{jt} (number of customers at table t in restaurant j), set $\theta_{ji} = \psi_{jt}$, and let $t_{ji} = t$ for whichever table is picked. If second term is chosen, increment $m_{j\cdot}$ (the number of tables in restaurant j) by one, draw a new dish for that table $\psi_{jm_j} \sim G_0$, and set $t_{ji} = m_{j\cdot}$.

To compute the predictive distribution of $\psi_{jt}|\psi_{11}, \psi_{12}, ..., \psi_{21}, ..., \psi_{jt-1}, H,\ G_0$ is integrated out as:

$$\psi_{jt}|\psi_{11}, \psi_{12}, ..., \psi_{21}, ..., \psi_{jt-1}, \gamma, H \sim \sum_{k=1}^{K} \frac{m_{\cdot k}}{m_{\cdot\cdot}+\gamma}\delta_{\phi_k} + \frac{\gamma}{m_{\cdot\cdot}+\gamma}H \tag{4.42}$$

The above equation assigns dishes to new tables. Similar to the procedure for the θ_{ji}'s, if drawing from first term, set $\psi_{jt} = \phi_k$ (dish served at restaurant j at table t is menu item ϕ_k and set $k_{jt} = k$. If drawing from second term, increment k by one and draw $\phi_K \sim H$, then set $\psi_{jt} = \phi_k$ (set the dish in restaurant j at table t to the new dish that was just drawn from H) and $k_{jt} = K$ (set atom index associated with dish ψ_{jt} to the atom index that was just drawn from H).

Dependent Dirichlet Processes

The Dependent Dirichlet Process (DDP) is an extension of the standard Dirichlet process which models collections of random distributions as related, but not identical.

The main idea underlying its construction is to replace both the stick breaking weights $\pi_k = \beta_k \prod_{l=1}^{k-1}(1-\beta_k)$ and the atom locations $\boldsymbol{\theta}$ with sample paths of a stochastic process as $\pi_{kx} = \beta_{kx} \prod_{l<k}[1-\beta_{lx}]$ and $\boldsymbol{\theta}_{\mathcal{X}}$, respectively. This approach is employed in nonparametric regression and dependency is introduced across values of the covariates $x \in \mathcal{X}$:

$$\mathbf{y}_i = f_{\boldsymbol{\theta}}(\mathbf{x}_i) + \epsilon_i \tag{4.43}$$

The DDP model is defined as

$$G_x = \sum_{k=1}^{\infty} \pi_{kx} \delta_{\boldsymbol{\theta}_{kx}} \tag{4.44}$$

DDPs can also be constructed using three fundamental operations on Poisson processes: superpositioning, subsampling (thinning), and transitioning. This construction makes a more clear connection to time series modeling as the covariates $x \in \mathcal{X}$ are replaced with time indexed by t: $G_t = \sum_{k=1}^{\infty} \pi_k \delta_{\boldsymbol{\theta}_{kt}}$. This facilitates the development of DDP mixtures for modeling dynamic phenomena. There are multiple versions of the dynamic DDP which vary depending on whether the G_t have the same weights and/or atoms, or whether the G_t each have different weights and/or atoms Zhong *et al.* (2021). We present here the dynamic DDP mixture where each G_t has different weights and atoms: $G_t = \sum_{k=1}^{\infty} \pi_{kt} \delta_{\boldsymbol{\theta}_{kt}}$. Let x_{it} denote the i-th observation at time t for $t = 1, ..., T$ Quintana *et al.* (2022). The dynamic DDP mixture model is given by:

$$\theta_{it} | G_t \sim G_t \tag{4.45}$$

$$x_{it} | \theta_{it} \sim F(\cot | \theta_{it}). \tag{4.46}$$

For more details we refer the reader to Zhong *et al.* (2021).

Gaussian Processes

Another Bayesian nonparametric approach that has been used for GRN inference is Gaussian process modeling. A Gaussian process defines a stochastic process such that

any finite subset of the random variables has a multivariate Gaussian distribution. It is used as a nonparametric prior distribution over continuous functions as

$$f(\mathbf{x}) \sim \mathcal{GP}(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')) \tag{4.47}$$

where $m(\mathbf{x}) = \mathbb{E}[f(\mathbf{x})]$ is a mean function and $k(\mathbf{x}, \mathbf{x}')$ is a covariance function that is used to construct the covariance matrix $K_{i,j}$. The covariance function is called the *kernel* of the Gaussian process Schulz *et al.* (2018). Examples of kernels include the Matérn covariance function and radial basis functions.

Applications of Bayesian Nonparametric Methods to Gene Regulatory Network Inference

Bayesian nonparametric models have been used for GRN inference. In Asif and Sanguinetti (2013) the authors use a factorial hidden Markov model (FHMM) to model the activity of transcription factors, which take on binary values. Gene expression in terms of mRNA concentration is formulated as a linear combination of these transcription factor dynamics as

$$g_i^t = \mathbf{e}_t^{iT}\theta_i + \epsilon \tag{4.48}$$

where g_i^t is the (log) mRNA expression level of gene i at time t, $\mathbf{e}_t^i$ is a vector whose entires are the binary states of the transcription factors $\{T_j^t\}_{j=1,\ldots,M}$ that bind to gene i, θ_i are the interaction strength parameters corresponding to gene i, and ϵ is zero-mean Gaussian noise. The authors infer the transcription factor dynamics and the gene-TF interaction strength parameter using gene expression data, and cluster them using a Dirichlet process mixture model. Collapsed Gibbs sampling is used for inference. The authors apply their model to the dataset describing micro-aerobic shift in *E. coli*, the yeast cell cycle dataset, and simulated data. The datasets vary from 15 to 104 transcription factors.They found that their approach performs best

on networks of intermediate size, and on data where the size of the dataset and the noise level increases.

DPMMs also facilitate integration of multiple types of data to improve inference. In Burdziak *et al.* (2019) the authors introduce *Symphony*, which combines epigenetic data with single-cell expression data to cluster cells by their cell type and their underlying GRN. They introduce a model for the epigenetic data that is later used in their model for gene expression. Their assumption is that the epigenetic data encode a network structure for each cluster of cells. The matrix encoding the network is asymmetric and is given as $R_k^{d\times d}$, where $k \in \{1, ..., K\}$ is the cluster index. Genome accessibility is represented by latent variables $\mathbf{p}_k = [p_k^1, ..., p_k^l]$ for l, where $\mathbf{p}_k \sim \text{trunc}\mathcal{N}(\boldsymbol{\eta}, \Lambda, \mathbf{0}, +\infty)$. The model for the epigenetic data is given by

$$\{\mathbf{c}\}_t^{(1,...,l)} | \mathbf{p}_k, \pi_k \sim \mathcal{N}(\sum_k \pi_k \mathbf{p}_k, \zeta I) \tag{4.49}$$

where a symmetric Dirichlet prior is placed over pi_k. The prior for the network structure is

$$R_k^{i,i'} \sim \mathcal{N}(S^{i,i'} M^{i,i'}, \mathbf{p}_k^{g(i,i')}, \lambda) \tag{4.50}$$

where the entry $R_k^{i,i'} \neq 0$ if i' directly regulates i, $S^{i,i'} = \text{sign}(\Sigma''^{i,i'})$ is a sign indicator variable denoting activation or repression and is set to the sign of the empirical covariance. $M^{i,i'}$ denotes the genomic region, and the function $g(\cdot)$ maps gene pairs to genomic regions. The model for gene expression for each cell is formulated as a Gaussian mixture model. Their approach produces a static directed network. Indirect effects up to a path length of two are taken into account when constructing directed edges, so a directed edge does not necessarily imply a direct causal interaction. The authors applied *Symphony* to simulated and real data. Variational expectation-maximization was used to sample from the posterior. They found that incorporating the epigenetic component improves clustering performance.

Other notable works use Hierarchical Dirichlet Processes to infer gene regulatory networks. In Wang and Wang (2013), a hierarchical Dirichlet process (HDP) mixture is used for both regulatory network segmentation and gene expression clustering. The authors use the Chinese restaurant franchise representation of the HDP along with Gibbs sampling for inference. Their approach produces a static directed network They apply the HDP mixture to several synthetic and real datasets, including the yeast cell cycle and human fibroblast serum data. The HDP was shown to outperform comparable methods, such as support vector machine (SVM) and Mclust.

In Thorne and Stumpf (2012), the authors use a "sticky" HDP-Hidden Markov Model (HDP-HMM), which incorporates state durations, to infer time-varying GRNs. The hierarchical model is given as follows:

$$\beta|\gamma \sim \text{GEM}(\gamma) \tag{4.51}$$

$$\boldsymbol{\pi}_k|\alpha,\beta,\kappa \sim DP\left(\alpha+\kappa, \frac{\alpha\beta+\kappa\delta_k}{\alpha+\kappa}\right) \tag{4.52}$$

$$s_j|s_{j-1},\boldsymbol{\pi} \sim \boldsymbol{\pi}_{s_{j-1}} \tag{4.53}$$

$$\theta \sim H \tag{4.54}$$

$$x_j|s_j \sim F(\theta_{s_j}) \tag{4.55}$$

where the s_j denote the hidden states. Each hidden state encodes a unique Bayesian network describing regulatory interactions indexed by time. Gibbs sampling is used to sample from the posterior. Their method is applied to several datasets, both synthetic and real, to infer the time-varying network structure. They found that although the sticky HDP-HMM performs well on synthetic data, more experimental work is needed for accurate inference of real-world regulatory networks.

When the gene regulation function is unknown, Gaussian processes can be used to infer $\mathbf{f}(\cdot)$. Since Gaussian process models learn the regulatory function directly from the data, known models of gene regulation can be captured within this approach Äijö

and Lähdesmäki (2009). In Äijö and Lähdesmäki (2009) the authors use Gaussian processes to infer the unknown regulatory function as well as the network structure. The Matérn covariance function as their choice for the kernel, which is given as

$$k(\mathbf{x}, \mathbf{x}') = \sigma(1 + \sqrt{3}\sqrt{\mathbf{u}^T P^{-1}\mathbf{u}})\exp(-\sqrt{3}\sqrt{\mathbf{u}^T P^{-1}}) \tag{4.56}$$

where $\mathbf{u} = \mathbf{x} - \mathbf{x}'$ and $P = \text{diag}(l^2)$. The model for gene regulation is given as

$$g(\mathbf{x}) = f(\mathbf{x}) + \mathbf{h}(\mathbf{x})^T\beta \tag{4.57}$$

where $f(\mathbf{x}) \sim \mathcal{GP}(0, k(\mathbf{x}, \mathbf{x}'))$ and β is the linear regression coefficient vector. A uniform prior is assigned over possible network structures $\mathcal{M}_j$. For inference, the authors first fit each gene to the model and then compute the posterior probabilities of the network structure given the gene expression values and explanatory variables. The method is applied on the popular IRMA dataset and was found to outperform dynamic and static Bayesian networks as well as other comparable methods.

Work by Aalto *et al.* (2018) also uses Gaussian processes to infer the unknown regulatory dynamics function. In their approach, called BINGO, they use the squared exponential covariance function as the kernel of choice. It is given by

$$k(\mathbf{x}, \mathbf{x}') = \gamma_i \exp\left(-\sum_{j=1}^{n} \beta_{i,j}(x_j - x'_j)^2 \right) \tag{4.58}$$

where $\beta_{i,j} > 0$ indicates that gene j regulates gene i. The advantage of BINGO is its scalability to large datasets via parallelized Markov chain Monte Carlo. BINGO was applied to data consisting of 2000 variables and five time series of 21 points each. The incorporation of multiple time series as well as knockout/knockdown experimental data enhanced accurate GRN inference and BINGO performed similarly to dynGENIE3. As well, BINGO performs well under high process noise and decreased sampling frequency.

Other uses of Gaussian processes for GRN inference are found in Dony *et al.* (2018); Hossain *et al.* (2021). In the former, the authors compare constrained parametric ODEs, in which the basal transcription and degradation rates are known, with unconstrained parametric ODEs, and fully non-parametric Gaussian process models. They found that constrained parametric approaches outperform the fully non-parametric Gaussian process method. However, since the constrained parametric approach requires a priori knowledge of the molecular kinetics, they suggest that a combination of parametric and non-parametric methods would perform best, which is the approach of choice for this dissertation project.

4.1.5 Dynamical Models

The specific dynamics of gene regulation can be described by ordinary differential equations or Boolean models. We review both in this section.

ODE Models

Differential equation models of gene regulation describe the rate of change of gene expression levels in terms of mRNA production. The rate of change is a function of the genes themselves as well as additional parameters which encode the regulatory dynamics between genes and the other genes in the network. Differential equation models can be linear, nonlinear, deterministic, or stochastic. The primary advantage of differential equation models is that they provide a mechanistic representation of biochemical processes and can capture the nonlinear dynamics of gene regulation. Furthermore, when formulated as a state space problem, they can be used to model GRNs as nonlinear stochastic processes, lending themselves to inference via well-developed statistical signal processing techniques, such as recursive Bayesian estimation Bugallo *et al.* (2015); Pirgazi and Khanteymoori (2018).

A popular model for gene regulation uses the sigmoid squash function to capture the switch-like behavior of GRNs (cite). The model for the state of the i-th gene at discrete time step k, $x_{i,k}$, is given by equation (Equation (4.59)).

$$x_{i,k} = \sum_{j=1}^{N} a_{ij} f(x_{j,k-1}) + w_{i,k} \tag{4.59}$$

$f(x_{i,k})$ is the sigmoid squash function given by equation (Equation (4.60)).

$$f(x_{i,k}) = \frac{1}{1 + e^{-x_{j,k-1}}} \tag{4.60}$$

In this model, N denotes the total number of genes in the system, a_{ij} models the nonlinear regulatory interactions between genes, and $w_{i,k}$ is an additive noise term typically assumed to be Gaussian. This model is used for inferring the structure of gene regulatory networks from noisy microarray data, as it has desirable properties such as continuity and monotonicity (ref 23 in quals).

A more complex model is the continuous-time S-system model Wang *et al.* (2007). The model is deterministic and is given by,

$$\frac{d}{dt} x_i = \alpha_i \prod_{j=1}^{N} x_j^{g_{ij}} - \beta_i \prod_{j=1}^{N} x_j^{h_{ij}} \tag{4.61}$$

The model shows that the rate of change in the system is described by a set of influxes minus a set of effluxes. Here, X_i is the expression level of the i-th gene, α_i and β_i are its corresponding non-negative rate constants, N is the total number of genes, h_{ij} and g_{ij} are the kinetic order parameters such that the j-th gene activates the i-th gene if $g_{ij} > 0$ and the j-th gene inhibits the expression of the i-th gene if $g_{ij} < 0$, and h_{ij} has the opposite effects of g_{ij}.

More recently, a series of stochastic S-system models have been introduced in the literature Chowdhury *et al.* (2015). These models take the same form as in equation (Equation (4.61)), but with additional noise terms. These noise terms include additive

Gaussian noise, as well as multiplicative noise and Langevin noise in either production, degradation, or both (see Chowdhury *et al.* (2015) for details). The generalized stochastic model is given by

$$\frac{d}{dt}X_i = \left[\alpha_i \prod_{j=1}^{N} X_j^{g_{ij}} - \beta_i \prod_{j=1}^{N} X_j^{h_{ij}}\right] + \mu g(X_i)\zeta_i(t) \tag{4.62}$$

where μ is the noise strength, $g(X_i)$ is the contribution of signal fluctuation, and $\zeta_i(t)$ is Gaussian white noise Chowdhury *et al.* (2015). Although these S-system models are able to capture more complex mechanisms of gene regulation, the product of power laws prevents these models from capturing any hyperbolic or sigmoidal behaviors, which are characteristic of many biological processes Youseph *et al.* (2019).

4.2 Michaelis-Menten Kinetics Model

Michaelis-Menten kinetic models can more precisely capture the nonlinear dynamics of gene regulation and have been used to generate realistic synthetic gene expression data Van den Bulcke *et al.* (2006). Michaelis-Menten and Hill kinetics have been used to reverse engineer gene regulatory networks in order to infer regulatory interactions from noisy microarray data Youseph *et al.* (2015); Elahi and Hasan (2018); Krishnan *et al.* (2020a). The Michaelis-Menten model Equation (4.63) was developed in the early 20th-century by scientists Leonor Michaelis and Maud Menten to describe the rate of enzyme reactions. In particular, the model describes how the rate of an enzyme-catalyzed reaction changes as the amount of substrate changes. Considering an enzyme that binds to a substrate S and converts it into a product P, the model provides the rate of change of the concentration of the product P as

$$\frac{d[P]}{dt} = v_{max}\frac{[S]}{K_M + [S]} \tag{4.63}$$

where $[P]$ is the concentration of the product P, $v_{\max}$ is the maximum reaction rate, $[S]$ is the concentration of the substrate which can either activate or inhibit enzyme

activity, and K_M is the Michaelis-Menten constant Youseph *et al.* (2015). At high concentrations of $[S]$, the reaction velocity asymptotes until it reaches a steady state at $v_{\max}$, so adding more substrate does not increase the rate of activity since all simple enzymes which follow Michaelis-Menten kinetics exhibit hyperbolic reaction velocities. The constant K_M represents the binding affinity between the enzyme and substrate. This is equal to the amount of substrate at which $\frac{1}{2}v_{max}$ is reached, or half of the maximum reaction rate. Large values of K_M indicate a low binding affinity since it takes more substrate to reach the maximum reaction rate. On the other hand, low levels of K_M indicate a high binding affinity, which means that it takes a lower concentration of substrate to reach the maximum reaction rate and that the substrate more effectively binds to the enzyme active site. Considering an activator A, which as the name suggests, activates the production of a product P, the Michaelis-Menten equation is given by

$$\frac{d[P]}{dt} = v_{max}\frac{[A]}{K_M + [A]} \tag{4.64}$$

and for an inhibitor I, which inhibits the formation of a product, the equation is given by

$$\frac{d[P]}{dt} = v_{\max}\frac{K_I}{K_I + [I]} \tag{4.65}$$

Thus, if both an activation and inhibitory interaction occurs, then the rate of product formation is given by multiplying equations 3.5 and 3.6

$$\frac{d[P]}{dt} = v_{max}(\frac{[A]}{K_M + [A]})(\frac{K_I}{K_I + [I]}) \tag{4.66}$$

By introducing the coefficient q to Equation (4.66), we extend the above model to include Hill kinetics.

$$\frac{d[P]}{dt} = v_{max}(\frac{[A]^q}{K_M^q + [A]^q})(\frac{K_I^q}{K_I^q + [I]^q}) \tag{4.67}$$

The Hill coefficient q is a measure of cooperativity of the substrate binding to the enzyme and reflects a behavior that is prevalent in biological networks Elahi and Hasan

(2018). In such cases, the reaction rate exhibits sigmoidal behavior where q controls the steepness of the reaction curve. A value of $q > 1$ indicates positive cooperativity, which means that the binding of a substrate to an enzyme is enhanced by the binding of another substrate to the same enzyme. A larger value of q makes the reaction curve steeper. A value of $q < 1$ indicates negative cooperativity so that once a substrate is bound to the enzyme, it decreases the affinity for subsequent binding, and thus the reaction curve is flatter. In the case of no cooperativity, where $q = 1$, Equation (4.67) simplifies to Michaelis-Menten kinetics. The Hill coefficient is somewhat reflective of the number of binding sites on an enzyme needed to produce a functional effect Santillán (2008). For example, a positive Hill coefficient would suggest that two or more binding sites are present on the enzyme. However, considering a Hill coefficient of $q = 1$, this could either suggest that there is only one binding site on the enzyme or there could be multiple binding sites whose dynamics are independent of each other. Thus, instead of serving as an estimate of the number of binding sites on an enzyme, a more accurate interpretation of the Hill coefficient is as an interaction coefficient reflecting cooperativity Santillán (2008).

A notable state-of-the-art model-based approach is dynGENIE3 Huynh-Thu and Geurts (2018), which uses non-parametric random forest models to learn the ODEs which describe the gene expression dynamics. The resulting network is a temporal aggregate of the gene expression dynamics. The efficacy of this approach is dataset-dependent, but overall outperforms its static counterpart, GENIE3. Although differential equation models can describe the biochemical processes involved in gene regulation in significant detail, one may argue that the large number of parameters needed to accurately describe a system poses a significant challenge.

Boolean Models

Boolean network models were introduced in 1969 by Stuart Kauffman to describe the discrete-time dynamics of regulatory systems Kauffman (1969). In these models, each vertex $x_i, i = 1, ..., n$ represents the state of gene i that can take on a binary value $x_i \in \{0, 1\}$ corresponding to an on or off state. An edge from gene j to i denotes a directional regulatory interaction so that gene j regulates gene i. The state of each gene at the next discrete time point $(t + 1)$ is determined by the states of the other nodes in the network at the previous time point t through the set of Boolean functions $\mathbf{f} = (f_1, ..., f_n)$ assigned to each of the nodes so that:

$$\mathbf{x}(t + 1) = \mathbf{f}(\mathbf{x}(t)) \tag{4.68}$$

where $\mathbf{x}(t) = (x_1(t), ..., x_n(t))$. The original formulation of this class of models assumes a deterministic and closed system Xiao (2009). However, from both a theoretical and practical standpoint, the assumption of a deterministic system with a single known Boolean function per gene is unrealistic. As regulatory systems are influenced by stochasticity, deterministic Boolean models are unable to capture any type of uncertainty, either due to environmental influences, interactions with other genetic networks, or intercellular variability. Probabilistic Boolean networks (PBNs) specifically address these issues.

In a PBN, rather than a single Boolean function per node, each node x_i corresponds to a set of l possible Boolean functions $F_i = \{f_j^{(i)}\}_{j=1,...,l}$. One realization of a PBN is determined by a vector of Boolean functions. Following the set up in Lähdesmäki *et al.* (2006); Shmulevich *et al.* (2002), for N possible realizations, there are N vector functions $\mathbf{f}_1, ..., \mathbf{f}_N$ where each $\mathbf{f}_j = (f_{j_1}^{(1)}, ..., f_{j_n}^{(n)})$ for $j = 1, 2, ..., N$, $1 \leq j_i \leq l(i)$ and $f_{j_i}^{(i)} \in F_i$. Given its current state at time $(t - 1)$ and a realization

$\mathbf{f}_j$, each node in the network transitions to the next discrete time point t via

$$\mathbf{x}(t) = \mathbf{f}_j(\mathbf{x}(t-1)). \tag{4.69}$$

Let $\mathbf{f} = (f^{(1)}, ..., f^{(n)})$ be a random vector which can take values in $F_1 \times F_N$. The probability that a *predictor function* $f_j^{(i)}$ is used to update the state of gene i is

$$c_j^{(i)} = \Pr\{f^{(i)} = f_j^{(i)}\} = \sum_{\mathbf{f}: f_i = f_j^{(i)}} \Pr\{\mathbf{f} = \mathbf{f}_j\} \tag{4.70}$$

Both probabilistic and deterministic Boolean models have been used to infer regulatory networks from expression data in a variety of systems Barman and Kwon (2017); Maki *et al.* (2002). More recent approaches combine Boolean models with Bayesian methods, such as Kalman filtering or particle filtering, which have shown to improve estimation accuracy Imani and Braga-Neto (2018); McClenny *et al.* (2017). In addition to their use for GRN inference, the primary advantage of Boolean models is in their ability to incorporate the logic operations (e.g. FOR, AND, NOT) and use them to combinatorially process inputs according to these rules, which are fundamental features of *cis*-regulatory function in GRNs Peter and Davidson (2015). Furthermore, these models are well suited to capture the discrete spatiotemporally bounded patterns of gene expression, which is characteristic of the developmental process. The success of the Boolean modeling approach is most notable in the reconstruction of the regulatory network describing the first 30 hours of development in the purple sea urchin Peter and Davidson (2015).

Chapter 5

STOCHASTIC MODELS OF GENE REGULATORY NETWORKS

5.1 State Space Model for GRN Estimation

An important challenge in processing gene regulatory networks (GRNs) is the development of mathematical models to aid in capturing the molecular mechanisms of gene regulation. In Youseph *et al.* (2015, 2019), the Michaelis-Menten kinetic equations are used to formulate a GRN model for estimating the Michaelis-Menten constant parameters. We propose a modified version of this model that incorporates two new important components. The new model includes the Hill kinetics to aid in more precisely capturing the complexity of GRN interactions. It also includes an additive error modeling random process to account for the inherent noise present in biological processes Wang and Aberra (2015). Our model for the expression level of the ith gene is given by

$$\frac{d}{dt}x_i(t) = v_i^{max}\left(\frac{g_{ii}x_i^{q_{ii}}(t) + h_{ii}K_{ii}^{q_{ii}}}{K_{ii}^{q_{ii}} + x_i^{q_{ii}}(t)}\right)\prod_{\substack{j=1\\ j\neq i}}^{N}\left(\frac{g_{ij}x_j^{q_{ij}}(t) + h_{ij}K_{ij}^{q_{ij}}}{K_{ij}^{q_{ij}} + x_j^{q_{ij}}(t)}\right) - v_i^d x_i(t) + w_i(t) \tag{5.1}$$

where v_i^{max} is the maximum expression rate of gene i, and h_{ij} and g_{ij} are the kinetic order parameters that indicate the type of regulation. In particular, when $h_{ij} = 1$ and $g_{ij} = 1$, $i \neq j$, then there is no regulation from gene j to gene i. $h_{ij} = 0 and g_{ij} = 1$ implies that gene j activates gene i, whereas $h_{ij} = 1, g_{ij} = 0$ implies that gene j inhibits gene i. Furthermore, the parameter K_{ij} is the Michaelis-Menten constant associated with the regulation of gene i by gene j, v_i^d is the self decay rate of mRNA expressed by gene i, q_{ij} is the Hill coefficient, and ii are subscripts associated with

autoregulation. In the context of gene regulation, the Michaelis-Menten constant K_{ij} accounts for the binding affinity between the product of a gene j and the binding site of a target gene i. Just like with enzyme kinetics, large values of K_{ij} indicate a low binding affinity and low values of K_{ij} indicate a high binding affinity. The added noise term is denoted by $w_i(t)$.

For ease of implementation, we discretize Equation (5.1) and introduce noise w_k into the system to obtain the following:

$$x_{i,k} = v_{i,k}^{max}\left(\frac{g_{ii}x_{i,k-1}^{q_{ii}} + h_{ii}K_{ii}^{q_{ii}}}{K_{ii}^{q_{ii}} + x_{i,k-1}^{q_{ii}}}\right)\prod_{\substack{j=1\\ j\neq i}}^{N}\left(\frac{g_{ij}x_{j,k-1}^{q_{ij}} + h_{ij}K_{ij}^{q_{ij}}}{K_{ij}^{q_{ij}} + x_{j,k-1}^{q_{ij}}}\right) - v_i^d x_{i,k-1} + w_{k-1} \quad (5.2)$$

where $x_{i,k} = x_i(k\Delta t)$ and $\Delta t = t_k - t_{k-1}$ and k is the discrete time value.

5.1.1 Multivariate Michaelis-Menten Kinetics Model

Extension of the model in Equation (5.2) to the multivariate case facilitates faster implementation and scalability to a larger number of genes by not having to specify each individual dynamical system equation. In state-space form, the model for gene expressions of N genes $x_{i,k}$, $n = 1, \ldots, N$

$$\mathbf{x}_k = \boldsymbol{\phi}(\mathbf{x}_{k-1}) + \mathbf{w}_{k-1}$$

$$\mathbf{y}_k = \mathbf{x}_k + \mathbf{v}_k$$

where $\mathbf{w}_k$ is the process noise vector, $\mathbf{y}_k$ is the vector of noisy measurements, $\mathbf{v}_k$ is the measurement noise vector, and

$$[\boldsymbol{\phi}(\mathbf{x}_{k-1})]_i = [\boldsymbol{\psi}(\mathbf{x}_{k-1})]_i\left(\prod_{\substack{j=1\\ j\neq i}}^{N}\mathbf{a}_{ij}\,\mathbf{b}_{ij}(\mathbf{x}_{k-1})\right) - v_i^d\; x_{i,k-1}$$

$$[\boldsymbol{\psi}(\mathbf{x}_{k-1})]_i = \frac{1}{v_i^{\max}}\left(K_{ii}^{q_{ii}} + x_{i,k-1}^{q_{ii}}\right)\prod_{\substack{j=1\\ j\neq i}}^{N}\left(K_{ij}^{q_{ij}} + x_{j,k-1}^{q_{ij}}\right)$$

where

$$\mathbf{a}_{ij} = [g_{ii}g_{ij},\ g_{ii}h_{ij},\ h_{ii}g_{ij},\ h_{ii}, h_{ij}]$$

$$\mathbf{b}_{ij}(\mathbf{x}_{k-1}) = [x_{i,k-1}^{q_{ii}} x_{j,k-1}^{q_{ij}},\ x_{i,k-1}^{q_{ii}} K_{ij}^{q_{ij}},\ x_{j,k-1}^{q_{ij}} K_{ii}^{q_{ii}},\ K_{ii}^{q_{ii}} K_{ij}^{q_{ij}}]^T$$

5.1.2 Extension to the Time-Varying Case

The extension of the model in Equation to the time-varying case is done in a straightforward manner through the addition of a time index to the kinetic order parameters $g_{ij,k}$ and $h_{ij,k}$, which encode both the direction and type of regulation. We denote the m-th set of kinetic order parameters for the $i-th$ gene by the variable $s_{ij,k}^m = (g_{ij,k}^m, h_{ij,k}^m)$, where $s_{ij,k}^m$ is referred to the m-th mode for $m = 1, ..., M$. The time-varying GRN state transition (TV-GRN) model for estimating the expression level $x_{i,k}$ of the ith gene at time step k due to the production and degradation of protein or messenger ribonucleic acid (mRNA) is given by

$$x_{i,k} = v_i^{\max} \left(\frac{g_{ii,k}\, x_{i,k-1}^{q_{ii}} + h_{ii,k}\, K_{ii}^{q_{ii}}}{K_{ii}^{q_{ii}} + x_{i,k-1}^{q_{ii}}} \right) \prod_{\substack{j=1 \\ j \neq i}}^{N} \left(\frac{g_{ij,k}\, x_{j,k-1}^{q_{ij}} + h_{ij,k}\, K_{ij}^{q_{ij}}}{K_{ij}^{q_{ij}} + x_{j,k-1}^{q_{ij}}} \right) - v_i^d\, x_{i,k-1} + u_{k-1}, \quad i = 1, \ldots, N$$

The effect of of dynamic changes in the kinetic order parameters in the state transition function $\boldsymbol{\phi}(\mathbf{x}_{k-1})$ is given as follows:

$$\mathbf{a}_{ij,k} = [g_{ii,k}g_{ij,k},\ g_{ii,k}h_{ij,k},\ h_{ii,k}g_{ij,k},\ h_{ii,k}h_{ij,k}]$$

The expression value of each gene x_i can be affected by three different types of regulatory interactions: activating, inhibiting, or no regulation. Therefore, there are a total of three possible modes for each pair of genes $j \to i \neq i \to j$. The mode $s_{ij,k}^m$ corresponding to each individual gene is indicative of the effects that the expression level of gene x_j has on the expression level of gene x_i. For a gene to transition from

mode m to mode n mode means that a new type of regulatory interaction occurs from gene j to gene i. For a system of N genes, each gene can have 3^N possible types of regulatory interactions including autoregulatory loops. These are summarized in the table below. As the number of genes in the network grows, so does the number of possible regulatory interactions.

m	Type of Regulatory Interactions (Modes)	Kinetic Order Parameters $(g^m_{ij,k}, h^m_{ij,k})$
1	$x_{j,k}$ activates $x_{i,k}$	$(1, 0)$
2	$x_{j,k}$ inhibits $x_{i,k}$	$(0, 1)$
3	no regulation from $x_{j,k}$ to $x_{i,k}$	$(1, 1)$

5.2 Stochastic Models of Gene Regulation

As the biochemical processes underlying gene regulation involve a small number of molecules, varying cellular environments, and varying timings of molecular events, gene regulation is an inherently stochastic process. Several stochastic nonlinear models have been introduced based on either S-System, the sigmoid squash function, Michaelis-Menten, or Langevin dynamics Chowdhury *et al.* (2015); Zhou and Ji (2017); Meister *et al.* (2014); Dari *et al.* (2011). Langevin dynamics are a class of stochastic differential equations (SDEs) used to model the time evolution of molecular systems as a combination of deterministic drift and random diffusion Callaham *et al.* (2021). The ubiquity of Langevin dynamics are due to their flexibility for modeling large networks Meister *et al.* (2014), and their ability to directly incorporate noise into the reaction rates and constants Shmulevich and Aitchison (2009). The Itô SDE

$$dx(t) = f(x(t), t)dt + \mu\sqrt{D(x(t), t)}dw(t) \tag{5.3}$$

follows Langevin dynamics where f and D are the drift and diffusion functions, respectively. Note that μ is a constant, and $w(t)$ is a Wiener process with Gaussian increments such that $\Delta w(t_i) = w(t_{i+1}) - w(t_i) \sim \mathcal{N}(0, \Delta t)$. Langevin dynamics have been used to model stochastic gene regulatory networks Shmulevich and Aitchison (2009). We use Langevin dynamics to develop stochastic GRN models with intrinsic and extrinsic noise. In particular, intrinsic noise is produced due to the inherent stochasticity of biochemical reactions during the process of gene expression and is typically modeled as multiplicative noise in either production or degradation Wang and Aberra (2015); Chowdhury *et al.* (2015); Raser *et al.* (2005). On the other hand, extrinsic noise is produced as the result of differences between cells in their local environments, such as the amount of mRNA polymerase or the cytoskeletal structure of a cell and is typically modeled as additive noise Wang and Aberra (2015); Hasty *et al.* (2000). In this chapter, each of these types of noise is modeled using discretized Langevin dynamics.

Given a partition of the time interval of $[0, T]$ into K equal subintervals of width Δt, the time evolution of a discrete-time stochastic process x_i can be given by the following discretized Langevin equation

$$x_{i,k+1} = x_{i,k} + f(x_{i,k}, k)\Delta t + \sqrt{D(x_{i,k}, k)}\Delta w_k \tag{5.4}$$

where for our purpose of GRN inference, $f(x_{i,k}, k)$ is the Michaelis-Menten model given in Equation (4.6). This model perturbs the deterministic model (**??**) to account for the stochastic fluctuations in molecular systems. The Langevin equation pertaining to the case of multiplicative noise in production is given by:

$$x_{i,k+1} = x_{i,k} + f(x_{i,k}, k)\Delta t + \sqrt{\rho(x_{i,k}, k)}\Delta w_k \tag{5.5}$$

where

$$f(x_{i,k},k) = v_{i,k}^{max}\left(\frac{g_{ii}x_{i,k}^{q_{ii}} + h_{ii}K_{ii}^{q_{ii}}}{K_{ii}^{q_{ii}} + x_{i,k}^{q_{ii}}}\right)\prod_{\substack{j=1\\ j\neq i}}^{N}\left(\frac{g_{ij}x_{j,k}^{q_{ij}} + h_{ij}K_{ij}^{q_{ij}}}{K_{ij}^{q_{ij}} + x_{j,k}^{q_{ij}}}\right) - v_i^d x_{i,k} \tag{5.6}$$

$$\rho(x_{i,k},k) = v_{i,k}^{max}\left(\frac{g_{ii}x_{i,k-1}^{q_{ii}} + h_{ii}K_{ii}^{q_{ii}}}{K_{ii}^{q_{ii}} + x_{i,k-1}^{q_{ii}}}\right)\prod_{\substack{j=1\\ j\neq i}}^{N}\left(\frac{g_{ij}x_{j,k-1}^{q_{ij}} + h_{ij}K_{ij}^{q_{ij}}}{K_{ij}^{q_{ij}} + x_{j,k-1}^{q_{ij}}}\right) \tag{5.7}$$

where $x_{i,k} = x_i(k\Delta t)$ and $\Delta t = t_k - t_{k-1}$ and k is the discrete time value and $\Delta w_k = w_{k+1} - w_k$. Similarly, the discrete-time Langevin equation pertaining to the case of multiplicative noise in degradation is given by the following Langevin equation:

$$x_{i,k+1} = x_{i,k} + f(x_{i,k},k)\Delta t + \sqrt{\gamma(x_{i,k},k)}\Delta w_k \tag{5.8}$$

where

$$\gamma(x_{i,k},k) = -v_i^d x_{i,k} \tag{5.9}$$

To model extrinsic noise, we use additive noise as

$$x_{i,k+1} = x_{i,k} + f(x_{i,k},k)\Delta t + \sqrt{\mu}\Delta w_k \tag{5.10}$$

where μ is a constant. Finally, combined models containing multiplicative and additive noise terms can be shown to follow Langevin dynamics Anteneodo and Tsallis (2003). The Langevin equation corresponding to multiplicative noise in production and additive noise is:

$$x_{i,k+1} = x_{i,k} + f(x_{i,k},k)\Delta t + \sqrt{\rho(x_{i,k},k)}\Delta w_k + \sqrt{\mu}\Delta w_k \tag{5.11}$$

Similarly, the Langevin equation corresponding to multiplicative noise in degradation and additive noise is:

$$x_{i,k+1} = x_{i,k} + f(x_{i,k},k)\Delta t + \sqrt{\gamma(x_{i,k},k)}\Delta w_k + \sqrt{\mu}\Delta w_k \tag{5.12}$$

Thus, we have five models accounting for each type of noise that can be present during the process of gene expression.

5.3 Conclusion

In subsequent chapters, we employ these models in the development of hierarchical Bayesian methods to tackle two main challenges. The first is inferring trajectories in time-varying GRNs by learning the network configuration which best describes the gene expression dynamics (Chapter 6). This is accomplished by learning the unknown transition probabilities describing the probability of changing to a different network configuration. The unknown process and measurement noise covariance matrices are learned using Inverse-Wishart priors with known hyperparameters. The second challenge is inferring trajectories in GRNs with unknown noise dynamics, which can change over time (Chapter 7). The unknown noise model is learned by drawing parameters from a categorical distribution with probabilities distributed according to a Dirichlet conjugate prior. We also learn the process and measurement noise covariance matrices by using Inverse-Wishart priors with known hyperparameters.

Chapter 6

SWITCHING LANGEVIN DYNAMICS IN GENE REGULATORY NETWORKS

6.1 Summary and Motivation

The process of gene regulation, from transcription to translation, is a result of the complex interactions between genes and an entire regulatory toolkit consisting of transcription factors, cis-regulatory elements, and an array of molecular constituents Peter and Davidson (2011). These complex interactions form gene regulatory networks (GRNs), which are responsible for the spatiotemporal allocation of gene expression in every cell Peter and Davidson (2011). As the biochemical processes underlying gene regulation involve a small number of molecules, varying cellular environments, and varying timings of molecular events, gene regulation is an inherently stochastic process. The stochasticity presents a challenge when reconstructing GRNs from noisy microarray data. As such, developing models that can account for the various sources of stochasticity, typically partitioned into *intrinsic* and *extrinsic* noise, can aid in the accurate reconstruction of GRNs. Furthermore, identifying these noise sources is important for the optimal design of control systems aimed at reducing noise in synthetic circuits Mundt *et al.* (2018).

Several stochastic nonlinear models have been introduced including the S-System, the sigmoid squash function, Michaelis-Menten, and Langevin dynamics Chowdhury *et al.* (2015); Zhou and Ji (2017); Meister *et al.* (2014). However, these approaches rely on two limiting assumptions: (a) the noise type is known a priori; (b) the noise type does not vary across time. Since microarray data only consists of a time series of expression values, it is not possible to know which noise type is present and it

is restricting to assume that the noise type remains the same across the whole time series. This facilitates the development of an approach that can select which noise model best fits the time series at each time step. This is ultimately a model selection problem. To this extent, multiple works have focused on Bayesian model selection in gene regulatory networks Ni *et al.* (2015); Novikov and Barillot (2009); Pan *et al.* (2016); Thorne (2016).

Despite the range of existing work on model selection for GRN inference, to the author's knowledge, there is no work that focuses on Bayesian model selection of stochastic models of GRNs. In this work, we propose a hierarchical Bayesian approach to account for switching between different types of unknown intrinsic and extrinsic noise models in the gene regulation process. We assume discretized Langevin dynamics to model the time evolution of each state, which are the gene expression values. The unknown noise model is learned by drawing parameters from a categorical distribution with probabilities distributed according to a Dirichlet conjugate prior. We also learn the unknown meausrement noise covariance matrix using an Inverse-Wishart prior, which is conjugate to the Gaussian likelihood. We employ the sequential Monte Carlo technique to do inference. In particular, we utilize particle filters to estimate the trajectory of each gene over time as well as the unknown noise model indicator,

The rest of the paper is organized as follows. In Section 6.2, we introduce several stochastic models of gene regulation using discretized Langevin dynamics. The hierarchical model and the inference algorithm are discussed in Section 6.3. We present our simulation results for a five-gene network in Section 6.4.

6.2 Stochastic Models of Gene Regulation

6.2.1 Gene Expression State

In our previous work, Vélez-Cruz *et al.* (2021), the authors developed several stochastic models of gene regulatory networks that were based on Michaelis-Menten kinetics Youseph *et al.* (2019). Our improvements to the model included an extension to the case of time-varying GRNs through the introduction of time-varying kinetic order parameters, which allows the regulatory interactions to change across time. Using this model, the expression level of the i-th gene due to mRNA production and degradation at time step k is given by

$$x_{i,k} = x_{i,k-1} + f(\mathbf{x}_{k-1})\Delta t + w_k \tag{6.1}$$

where $\mathbf{x}_{k-1} = [x_{i,k}, ..., x_{N,k}]$, Δt is the interval between time steps, and $w_{i,k}$ is a Gaussian random process used to model uncertainty, and

$$\begin{aligned} x_{i,k} &= x_{i,k-1} + \Delta t \Bigg(v_i^{max} \left(\frac{g_{ii} x_{i,k-1}^{q_{ii}} + h_{ii} K_{ii}^{q_{ii}}}{K_{ii}^{q_{ii}} + x_{i,k-1}^{q_{ii}}} \right) \\ &\prod_{\substack{j=1 \\ j \neq i}}^{N} \left(\frac{g_{ij} x_{j,k-1}^{q_{ij}} + h_{ij} K_{ij}^{q_{ij}}}{K_{ij}^{q_{ij}} + x_{j,k-1}^{q_{ij}}} \right) - v_i^d x_{i,k-1} \Bigg) + w_k \\ &= x_{i,k-1} + f(\mathbf{x}_{k-1})\Delta t + w_k \end{aligned} \tag{6.2}$$

In Equation (6.2), v_i^{max} is the maximum expression rate of gene i, and h_{ij} and g_{ij} are the kinetic order parameters that indicate the type of regulation. In particular, when $h_{ij} = 1$ and $g_{ij} = 1$, $i \neq j$, then there is no regulation from gene j to gene i. When $h_{ij} = 0$ and $g_{ij} = 1$, then gene j activates gene i, whereas $h_{ij} = 1, g_{ij} = 0$ implies that gene j inhibits gene i. The parameter K_{ij} is the Michaelis-Menten constant associated with the regulation of gene i by gene j. In the context of gene regulation, the Michaelis-Menten constant K_{ij} accounts for the binding affinity between the product

of a gene j and the binding site of a target gene i. Large values of K_{ij} indicate a low binding affinity and low values of K_{ij} indicate a high binding affinity. In Equation (6.2), v_i^d is the self decay rate of mRNA expressed by gene i, and q_{ij} is the Hill coefficient. The parameters K_{ii}, q_{ii}, h_{ii} and g_{ii} specify autoregulatory interactions.

6.2.2 Langevin Dynamics

Langevin dynamics are a class of stochastic differential equations (SDEs) used to model the time evolution of molecular systems as a combination of deterministic drift and random diffusion Callaham *et al.* (2021). The ubiquity of Langevin dynamics are due to their flexibility for modeling large networks Meister *et al.* (2014), and their ability to directly incorporate noise into the reaction rates and constants Shmulevich and Aitchison (2009). The Itô SDE

$$dx(t) = f(x(t))\ dt + \mu\sqrt{D(x(t))}dw(t) \tag{6.3}$$

follows Langevin dynamics where $f(x(t))$ and $D(x(t))$ are the drift and diffusion functions, respectively **?**. Note that μ is a constant, and $w(t)$ is a Wiener process with Gaussian increments such that $\Delta w(t_k) = w(t_{k+1}) - w(t_k) \sim \mathcal{N}(0, \Delta t)$. Langevin dynamics have been used to model stochastic gene regulatory networks Shmulevich and Aitchison (2009). We use Langevin dynamics to develop stochastic GRN models with intrinsic and extrinsic noise. In particular, intrinsic noise is produced due to the inherent stochasticity of biochemical reactions during the process of gene expression and is typically modeled as multiplicative noise in either production or degradation Wang and Aberra (2015); Chowdhury *et al.* (2015); Raser *et al.* (2005). On the other hand, extrinsic noise is produced as the result of differences between cells in their local environments, such as the amount of mRNA polymerase or the cytoskeletal structure of a cell and is typically modeled as additive noise Wang and Aberra (2015); Hasty

et al. (2000). In this paper, each of these types of noise is modeled using discretized Langevin dynamics.

Given a partition of the time interval of $[0, T]$ into K equal subintervals of width Δt, the time evolution of a discrete-time stochastic process x_i can be given by the following discretized Langevin equation

$$x_{i,k} = x_{i,k-1} + f(\mathbf{x}_{k-1})\ \Delta t + \sqrt{D(\mathbf{x}_{k-1})}\Delta w_k \tag{6.4}$$

where for our purpose of GRN inference, $f(\mathbf{x}_{k-1})$ is the Michaelis-Menten model given in Equation (6.2) and w_k is additive white Gaussian with zero mean and known covariance Σ_w. Whereas the stochastic component in Equation (6.2) is used to represent model uncertainty, in Equation (6.4), it is used to account for the stochastic fluctuations in molecular systems. The Langevin equation pertaining to the case of multiplicative noise in production is given by:

$$x_{i,k} = x_{i,k-1} + f(\mathbf{x}_{k-1})\,\Delta t + \sqrt{\rho(\mathbf{x}_{k-1})}\,\Delta w_k \tag{6.5}$$

where

$$\rho(x_{i,k-1}) = v_i^{max}\left(\frac{g_{ii}x_{i,k-1}^{q_{ii}} + h_{ii,k}K_{ii}^{q_{ii}}}{K_{ii}^{q_{ii}} + x_{i,k-1}^{q_{ii}}}\right) \prod_{\substack{j=1 \\ j\neq i}}^{N}\left(\frac{g_{ij}x_{j,k-1}^{q_{ij}} + h_{ij}K_{ij}^{q_{ij}}}{K_{ij}^{q_{ij}} + x_{j,k-1}^{q_{ij}}}\right).$$

Similarly, the discrete-time Langevin equation pertaining to the case of multiplicative noise in degradation is given by the following Langevin equation:

$$x_{i,k} = x_{i,k-1} + f(\mathbf{x}_{k-1})\ \Delta t + \sqrt{\gamma(x_{i,k-1})}\Delta w_k \tag{6.6}$$

where

$$\gamma(x_{i,k-1}) = -v_i^d x_{i,k-1}$$

A combined model of multiplicative noise in production and multiplicative noies in degradaton is given as

$$x_{i,k} = x_{i,k-1} + f(\mathbf{x}_{k-1})\,\Delta t + \sqrt{\rho(\mathbf{x}_{k-1})}\Delta w_k + \sqrt{\gamma(x_{i,k-1})}\Delta w_k \tag{6.7}$$

To model extrinsic noise, we use additive noise as

$$x_{i,k} = x_{i,k-1} + f(\mathbf{x}_{k-1})\ \Delta t + \sqrt{\mu}\Delta w_k \tag{6.8}$$

where μ is a constant. Finally, combined models containing multiplicative and additive noise terms can be shown to follow Langevin dynamics Anteneodo and Tsallis (2003). The Langevin equation corresponding to multiplicative noise in production and additive noise is:

$$x_{i,k} = x_{i,k-1} + f(\mathbf{x}_{k-1})\ \Delta t + \sqrt{\rho(\mathbf{x}_{k-1})}\Delta w_k + \sqrt{\mu}\Delta w_k \tag{6.9}$$

Similarly, the Langevin equation corresponding to multiplicative noise in degradation and additive noise is:

$$x_{i,k} = x_{i,k-1} + f(\mathbf{x}_{k-1})\ \Delta t + \sqrt{\gamma(x_{i,k-1})}\Delta w_k + \sqrt{\mu}\Delta w_k \tag{6.10}$$

The six models in Equation (6.5), Equation (6.6), Equation (6.8), Equation (6.9), and Equation (6.10) account for each type of noise that can be present during the process of gene expression.

6.3 Inference

Using the aforementioned noise models, our goal is to infer the trajectory of the states $x_{i,k},\ i = 1, ..., N$ and $k = 1, ..., K$, where N is the number of genes in the network and K is the number of time steps. It is worth mentioning that even though the number of models is known, it is not known which model is used a priori. The

state space formulation of the GRN dynamical system is given by

$$x_{i,k} = g_m(\mathbf{x}_{k-1}, \Delta w_k),\ i = 1, \ldots, N,\ m = 1, \ldots, M$$

$$\mathbf{y}_k = \mathbf{x}_k + v_k$$

where $\mathbf{y}_k$ is the noisy microarray measurement at time k and $\mathbf{v}_k$ is the measurement noise that is assumed additive white Gaussian with zero mean and unknown covariance Σ_v. The function $g_m(\mathbf{x}_{k-1}, \Delta w_k)$ represents the state function with the mth noise model, $m = 1, \ldots, M$. The modeling noise Δw_k is assumed Gaussian with known covariance Σ_w. For our case, the $M = 5$ models include the multiplicative noise in production in (Equation (6.5)), multiplicative noise in degradation in (Equation (6.6)), extrinsic noise in (Equation (6.8)), multiplicative in production and additive noise in (Equation (6.9)) and multiplicative in degradation and additive noise in (Equation (6.10)).

Note that, even though the number of noise models is assumed known, the noise model at each time step k is unknown. As a result, we need to learn the noise model at each time step in order to form the state prior probability density function (PDF). In our proposed approach, we represent the M noise model types using the indicator variables $z_{i,k} \in \{1, \ldots, M\}$ and model the indicator variables using a categorical distribution. The parameters of the categorical distribution are given by $\pi_{1,k}, \ldots, \pi_{M,k}$, where $\pi_{m,k}$ is the probability of occurrence of the mth noise model. We learn these parameters using a Dirichlet distribution prior with hyperparameter α. We also assume that the measurement noise covariance Σ_v is unknown with an Inverse-Wishart (IW) prior with scale matrix $\mathbf{\Psi_v}$ and degrees of freedom ν_v. The resulting hierarchical

Bayesian model is given by

$$\mathbf{x}_k|\mathbf{x}_{k-1}, z_k, \Sigma_w \sim G_{z_k}(\mathbf{x}_k|\mathbf{x}_{k-1}, z_k),\ i = 1, ..., N \tag{6.11}$$

$$\mathbf{y}_k|\mathbf{x}_k, \Sigma_v \sim H(\mathbf{y}_k|\mathbf{x}_k, \Sigma_v) \tag{6.12}$$

$$z_k|z_{k-1}, \pi_{m,k} \sim \text{Cat}(\pi_{m,k}),\ m = 1, ...M \tag{6.13}$$

$$\pi_{m,k}|\alpha \sim \text{Dir}(\alpha) \tag{6.14}$$

$$\Sigma_v|\mathbf{\Psi}_v, \nu_v \sim \text{IW}(\mathbf{\Psi}_v, \nu_v) \tag{6.15}$$

where z_k is the indicator corresponding to model m at time k, Cat$(\cdot)$ denotes the categorical distribution, and Dir$(\cdot)$ denotes the Dirichlet distribution. Note that the distribution $G(\cdot)$ is Gaussian since the state evolution follows a Langevin dynamic. The emission distribution $H(\cdot)$ is also assumed to be Gaussian. The posterior density needed to estimate the gene expression values $\mathbf{x}_k$ and the indicators which select the network configuration are $\{z_k^m\}_{m=1}^M$.

$$p(\{\mathbf{x}_k\}_{k=1}^K, \{z_k\}_{k=1}^K, \Sigma_v, \{\pi_{m,k}\}_{m=1,k=1}^{M,K}|\{\mathbf{y}_k\}_{k=1}^K, \mathbf{\Psi}_v, \nu_v, \alpha) \tag{6.16}$$

$$\propto p(\{\mathbf{y}_k\}_{k=1}^K, \{\mathbf{x}_k\}_{k=1}^K, \{z_k^m\}_{k=1,m=1}^{K,M}, \Sigma_v|\alpha, \mathbf{\Psi}_v, \nu_v) \tag{6.17}$$

$$p(\Sigma_v|\mathbf{\Psi}_v, \nu_v)p(\{\pi_{m,k}\}_{m=1}^M|\alpha) \tag{6.18}$$

$$= \prod_{k=1}^K p(\mathbf{y}_k|\mathbf{x}_k, \Sigma_v) \prod_{m=1}^M \left[p(\mathbf{x}_k|\mathbf{x}_{k-1}, z_k, \{\Sigma_w^m\}_{m=1}^M)^{1(z_k=m)} p(z_k|z_{k-1}, \pi_{m,k})^{1(z_k=m)} \right] \tag{6.19}$$

$$p(\Sigma_v|\mathbf{\Psi}_v, \nu_v)p(\{\pi_{m,k}\}_m^M|\alpha). \tag{6.20}$$

where z_k is the indicator corresponding to model m at time k, Σ_v is the unknown measurement noise covariance, $\mathbf{\Psi}_v$ is the $N \times N$ prior scale matrix chosen to be $0.1\mathbb{I}_N$, and ν_v is the prior degrees of freedom which is chosen to be $N + 2$. Note $mathbbI_N$ is the $N \times N$ identity matrix, that $\pi_{m,k}$ is the probability of selecting noise model m at time k, and α is the concentration parameter for the Dirichlet distribution. We set

$\alpha = 1$. The distribution $G(\cdot)$ and the likelihood $H(\cdot)$ are assumed to be Gaussian.

Sequential Monte Carlo

We develop a sequential Monte Carlo algorithm. At each time step, the joint posterior PDF from which we aim to draw samples is

$$p(\mathbf{x}_k, z_k, \pi_{m,k}, \Sigma_v | \mathbf{y}_k, \mathbf{\Psi}_v, \nu_v, \alpha) \tag{6.21}$$

$$\propto p(\mathbf{y}_k|\mathbf{x}_k, \Sigma_v) p(\mathbf{x}_k|\mathbf{x}_{k-1}, z_k) p(z_k|z_{k-1}, \pi_{m,k}) p(\Sigma_v|\mathbf{\Psi}_v, \nu_v) p(\pi_{m,k}|\alpha) \tag{6.22}$$

$$p(\mathbf{x}_{k-1}, z_{k-1}, \pi_{m,k-1}, \Sigma_v | \mathbf{y}_{k-1}, \mathbf{\Psi}_v, \nu_v, \alpha) \tag{6.23}$$

where $p(\mathbf{x}_k|\mathbf{x}_{k-1}, z_k)$ is given in Section 8.2.1 and $p(\mathbf{x}_{k-1}, z_{k-1}, \pi_{m,k-1}, \Sigma_v|\mathbf{y}_{k-1}, \mathbf{\Psi}_v, \nu_v, \alpha)$ is the posterior from the previous time step. The algorithm proceeds by sampling a model indicator from Q_1 in Equation (7.19)

$$Q_1(z_k = m|\mathbf{z}_{-k}, \mathbf{x}_k, \mathbf{y}_k, \Sigma_v \alpha) = \frac{c_{k,m}}{\sum_m c_{k-1,m} + \alpha} G(\mathbf{x}_k|\mathbf{x}_{k-1}, z_k) H(\mathbf{y}_k|\mathbf{x}_k, \Sigma_v) \tag{6.24}$$

where $c_{k-1,m}$ is the number of times model m has been chosen for times $k = 1 \ldots K-1$.

Then the measurement $\mathbf{y}_k$ is computed using Section 8.2.1. The measurement noise covariance hyperparameters at time step k, which are $\mathbf{\Psi}_v^{'}$ and $\nu_v^{'}$, are updated using Equations Equation (6.26) and Equation (6.27) and a new measurement noise covariance Σ_v is sampled from an inverse-Wishart with the updated hyperparameters.

$$Q_2(\Sigma_v|\mathbf{y}_k, \mathbf{\Psi}_v, \nu_v) = \text{IW}(\mathbf{\Psi}_v^{'}, \nu_v^{'}) \tag{6.25}$$

where the new hyperparameters $\mathbf{\Psi}_v^{'}$ and $\nu_v^{'}$ are incrementally updated at each time step as

$$\Psi_v^{'} = \Psi_{v,\text{prev}} + (\mathbf{x}_k - \mu_v)(\mathbf{x}_k - \mu_v)^T \tag{6.26}$$

$$\nu_v^{'} = \nu_{v,\text{prev}} + 1 \tag{6.27}$$

and we have one observation at each time step and μ_v is assumed to be $\mathbf{0}$. Next, we can compute the weights. We select the joint prior PDF as the proposal distribution since the underlying noise in our model follows Gaussian distribution. Moreover, this approach is more computationally efficient. The joint prior PDF is given as

$$q(\mathbf{x}_k^{(i)}, z_k^{(i)}, \pi_{m,k}^{(i)}, \Sigma_v^{(i)} | \mathbf{x}_{k-1}^{(i)}, z_{k-1}^{(i}, \Sigma_v^{(i)}, \mathbf{\Psi}_v^{(i)}, \nu_v^{(i)}, \alpha) \tag{6.28}$$

$$= p(\mathbf{x}_k^{(i)} | \mathbf{x}_{k-1}^{(i)}, z_k^{(i)}) p(z_k^{(i)} | z_{k-1}^{(i)}, \pi_{m,k}^{(i)}) p(\Sigma_v^{(i)} | \mathbf{\Psi}_v^{(i)}, \nu_v^{(i)}) p(\pi_{m,k}^{(i)} | \alpha) \tag{6.29}$$

The weights can then be computed as

$$w_k^{(i)} \propto w_{k-1}^{(i)} p(\mathbf{y}_k | \mathbf{x}_k^{(i)}, \Sigma_v^{(i)}) \tag{6.30}$$

where the Gaussian likelihood is given as

$$p(\mathbf{y}_k | \mathbf{x}_k^{(i)}, \Sigma_v^{(i)}) = \frac{1}{(2\pi)^{N/2} \sqrt{|\Sigma_v^{(i)}|}} \exp\left(-\frac{1}{2} (\mathbf{y}_k - \mathbf{x}_k^{(i)})^T (\Sigma_v^{(i)})^{-1} (\mathbf{y}_k - \mathbf{x}_k^{(i)}) \right) \tag{6.31}$$

See Appendix A for the full derivation.

The algorithm steps are summarized below. where $\mathbf{c} = [c_1, ..., c_M]$ and c_M is the number of occurrences of model M.

6.4 Simulation Results

In this section, we demonstrate through simulations the efficacy of our approach in estimating the trajectories of $N = 5$ genes and the unknown noise model indicator. The GRN model parameter values used in each simulation are listed in Table 6.4. A diagram of the network is shown in Figure 6.6. For each discrete time step k, one of the $m = 1, ..., M$ network configurations, which is indicated by z_k, is chosen probabilistically from a Categorical distribution with Dirichlet-distributed probabilities. We also learn the unknown measurement noise covariance matrix using Bayesian updating of the Inverse-Wishart. Our learning approach aims to concurrently estimate

Algorithm 1 Bayesian Learning of Variation in Noise Type Under Switching Langevin Dynamics

Initialization at $t = 1$

for $i = 1, ..., N$ particles **do**

  Sample $\mathbf{x}_1^{(i)} \sim p(\mathbf{x}_1)$

  Sample $z_1^{(i)} \sim \text{Cat}(\boldsymbol{\pi}_1)$

  Sample $\Sigma_v^{(i)} \sim \text{IW}(\boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)})$

end for

Sequential Updates for $t \geq 2$

Draw model probabilities from posterior of the Dirichlet distribution $\pi_{m,k} \mid \alpha + \mathbf{c} \sim \text{Dir}(\pi_{m,k} \mid \alpha + \mathbf{c})$

Sample a new cluster assignment $z_k^{(i)} \sim Q_1$

Sample $\mathbf{x}_k^{(i)} \sim p(\mathbf{x}_k \mid \mathbf{x}_{k-1}^{(i)}, z_k^{(i)})$

Update measurement $\mathbf{y}_k^{(i)} \sim p(\mathbf{y}_k \mid \mathbf{x}_k^{(i)}, \Sigma_v^{(i)})$

Update hyperparameters using Equation (6.26) and Equation (6.27) and sample $\Sigma_v^{(i)} \sim \text{IW}(\boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)})$ from Q_2

Compute particle weights $w_k^{(i)}$ using Equation Section 8.2.4

Normalize weights $w_k^{(i)} = \frac{w_k^{(i)}}{\sum_{i=1}^{N_s} w_k^{(i)}}$

Resample particles $\mathbf{x}_k^{(i)}, z_k^{(i)}, \Sigma_v^{(i)}$

the unknown state $\mathbf{x}_k$ and the model indicator variable z_k. We use 2,000 Monte Carlo runs for each simulation.

We consider switching between three noise models for $K = 250$ time points under the following scenario. Cells undergo oxidative stress when there is an imbalance of free radicals and antioxidants Lobo *et al.* (2010). When this occurs, a class of genes which encodes antioxidant enzymes, Superoxide dismutase (SOD), is activated. Under low oxidative stress conditions, cells can be subject to various stochastic processes such as fluctuating nutrient conditions or variable protein turnover rates. For this scenario, we assume that these baseline stochastic conditions manifest in multiplicative noise in degradation for $k = 1 : 100$. When oxidative stress occurs as a result of the introduction of sources of free radicals, such as cigarette smoke, radiation, or environmental pollutants, the cells increase their production of antioxidant enzymes. Signaling pathways responsible for producing the antioxidant enzymes are activated. The activation of the SOD class is not a deterministic process and can be subject to epigenetic variability or variability in transcription factor concentrations. This manifests in multiplicative noise in production for $k = 101 : 170$. After this period of oxidative stress has passed, the cell goes into a recovery mode characterized by instability in the regulatory circuits. This means that the production and degradation of enzymes can both be subjected to noise. This manifests in multiplicative noise in production and degradation for $k = 171 : 250$.

First, we demonstrate the performance of our algorithm under different numbers of particles. Table 6.1 shows the average root mean square error (RMSE) for $Ns = 50, 200$, and 1,000 particles for process noise covariance $\Sigma_w = 2e^{-3}\mathbb{I}_N$ and measurement noise covariance $\Sigma_v = 2e^{-3}\mathbb{I}_N$. We found that the root mean square error (RMSE) decreases with an increase of the number of particles. Next, we evaluate the performance of the algorithm under different measurement noise conditions. We

Table 6.1: Root mean-square error (RMSE) averaged across all time steps for $Ns = 50, 200$, and 1,000 particles for process noise covariance $\Sigma_w = 2e^{-3}\mathbb{I}_N$ and measurement noise covariance $\Sigma_v = 2e^{-3}\mathbb{I}_N$. 2,000 Monte Carlo runs were used. The dynamics are described by multiplicative noise in degradation for $k = 1 : 100$, multiplicative noise in production for $k = 101 : 170$, and multiplicative noise in production and degradation $k = 171 : 250$.

Gene	$Ns = 50$	$Ns = 200$	$Ns = 1,000$
X_1	0.075	0.069	0.058
X_2	0.083	0.075	0.065
X_3	0.091	0.083	0.072
X_4	0.080	0.074	0.064
X_5	0.038	0.036	0.032

assume $\Sigma_w = 2e^{-5}\mathbb{I}_N$ and evaluate the performance under $\Sigma_v = 2e^{-2}\mathbb{I}_N, 2e^{-3}\mathbb{I}_N$ and $2e^{-5}\mathbb{I}_N$. We compute the average RMSE across all time steps shown in Table 6.2. Our Bayesian learning algorithm is compared to the standard PF, which assumes multiplicative noise in production in Figure 6.1. As is shown, the standard particle filter fails to estimate the trajectory during the segments whose dynamics are not given by multiplicative noise in production. The true versus estimated noise model is shown in Figure 6.2. The figures in Figure 6.2, Figure 6.4, and Figure 6.3 demonstrate that an increase in the measurement noise results in a decrease in estimation accuracy. We also consider variation in the process noise covariance under $\Sigma_w = 2e^{-2}\mathbb{I}_N, 2e^{-3}\mathbb{I}_N$ and $2e^{-5}\mathbb{I}_N$ where $\Sigma_v = 2e^{-3}\mathbb{I}_N$. The RMSE values averaged across all time points are shown in Table 6.3. We show the RMSE over time for each variation in the process noise in Figure 6.5.

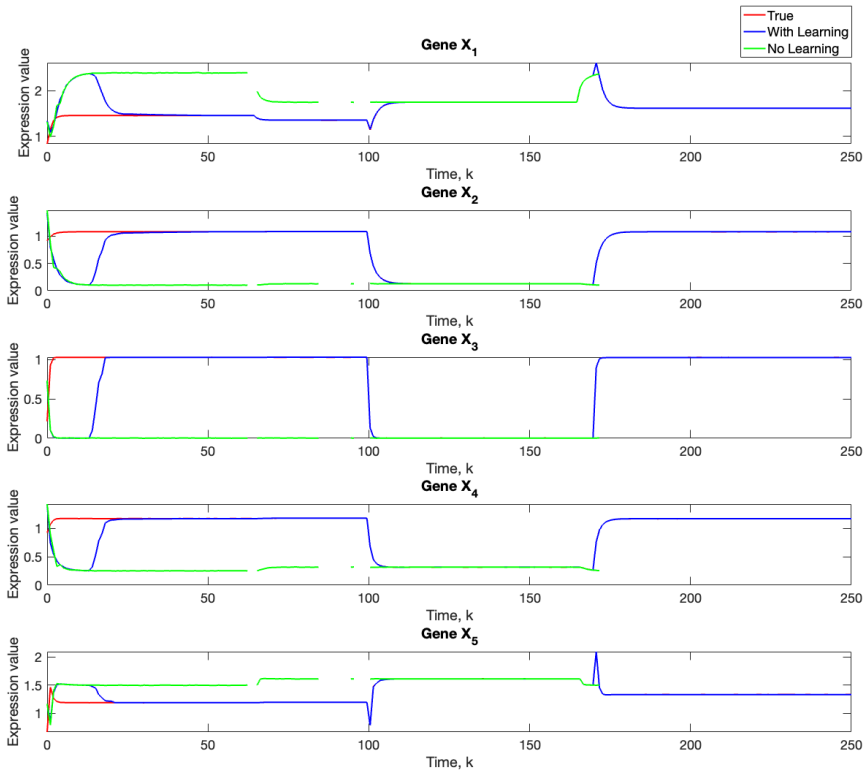


Figure 6.1: A comparison of our learning algorithm (blue) to the standard particle (green) for $\Sigma_w = 2e^{-5}\boldsymbol{I}_N$ and $\Sigma_v = 2e^{-3}\boldsymbol{I}_N$. The true trajectory is given by the red. 2,000 Monte Carlo runs and 1,000 particles were used. The dynamics are described by multiplicative noise in degradation for $k = 1 : 100$, multiplicative noise in production for $k = 101 : 170$, and multiplicative noise in production and degradation $k = 171 : 250$. Multiplicative noise in degradation is assumed.

Table 6.2: Root mean-square error (RMSE) averaged across all time steps for measurement noise covariance $\Sigma_v = 2e^{-2}\mathbb{I}_N, 2e^{-3}\mathbb{I}_N$, and $2e^{-5}\mathbb{I}_N$. We assume $\Sigma_w = 2e^{-5}\mathbb{I}_N$. 1,000 particles and 2,000 Monte Carlo runs were used. The dynamics are described by multiplicative noise in degradation for $k = 1 : 100$, multiplicative noise in production for $k = 101 : 170$, and multiplicative noise in production and degradation $k = 171 : 250$.

Gene	$\Sigma_v = 2e^{-2}\mathbb{I}_N$	$\Sigma_v = 2e^{-3}\mathbb{I}_N$	$\Sigma_v = 2e^{-5}\mathbb{I}_N$
X_1	0.073	0.069	0.047
X_2	0.074	0.073	0.051
X_3	0.081	0.079	0.056
X_4	0.070	0.068	0.049
X_5	0.034	0.030	0.022

Table 6.3: Root mean-square error (RMSE) averaged across all time steps for process noise covariance $\Sigma_w = 2e^{-2}\mathbb{I}_N, 2e^{-3}\mathbb{I}_N$, and $2e^{-5}\mathbb{I}_N$. We set $\Sigma_v = 2e^{-3}\mathbb{I}_N$. 1,000 particles and 2,000 Monte Carlo runs were used.

Gene	$\Sigma_w = 2e^{-2}\mathbb{I}_N$	$\Sigma_w = 2e^{-3}\mathbb{I}_N$	$\Sigma_w = 2e^{-5}\mathbb{I}_N$
X_1	0.063	0.054	0.052
X_2	0.068	0.061	0.059
X_3	0.077	0.068	0.066
X_4	0.070	0.058	0.056
X_5	0.043	0.026	0.023

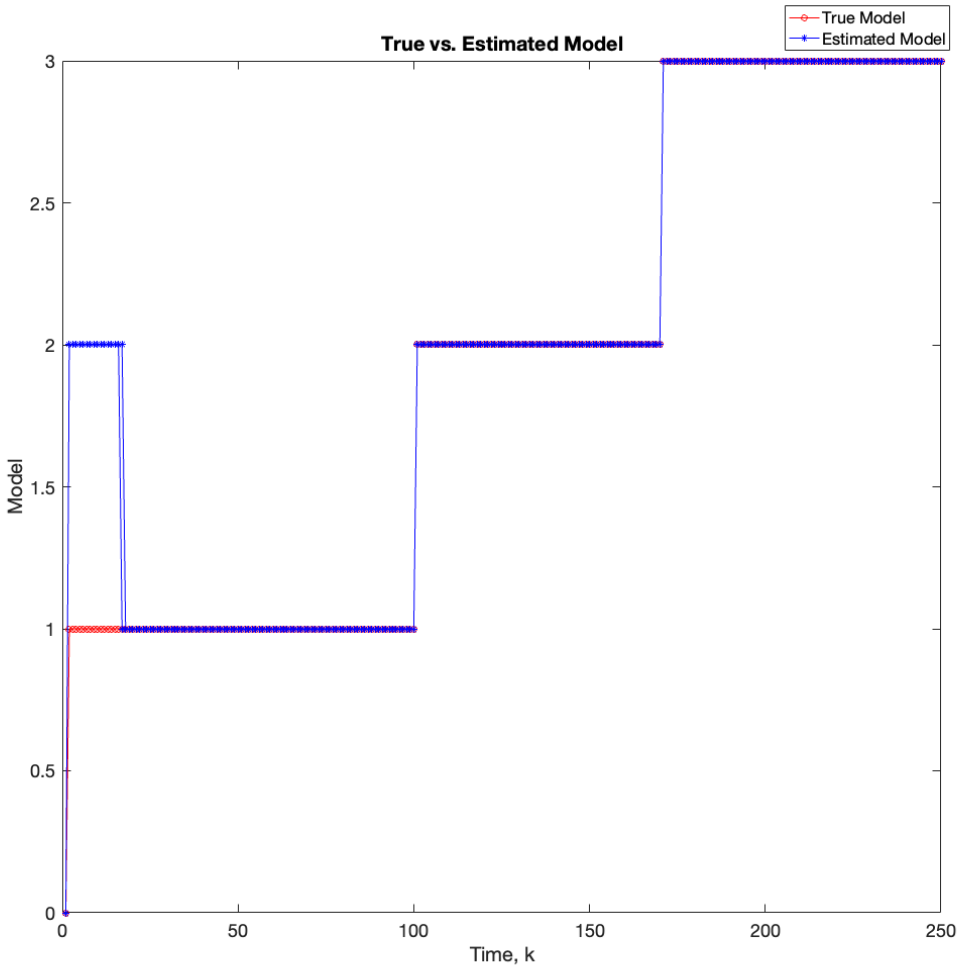


Figure 6.2: The true (red) versus estimated (blue) model or $\Sigma_w = 2e^{-5}\boldsymbol{I}_N$ and $\Sigma_v = 2e^{-5}\boldsymbol{I}_N$. 2,000 Monte Carlo runs and 1,000 particles were used. The dynamics are described by multiplicative noise in degradation for $k = 1 : 100$, multiplicative noise in production for $k = 101 : 170$, and multiplicative noise in production and degradation $k = 171 : 250$.

To evaluate the robustness of our model, we allow the dynamics to be defined by multiplicative noise in production for $k = 1 : 100$, multiplicative noise in degradation for $k = 101 : 170$, and multiplicative noise in production and degradation for $k =$

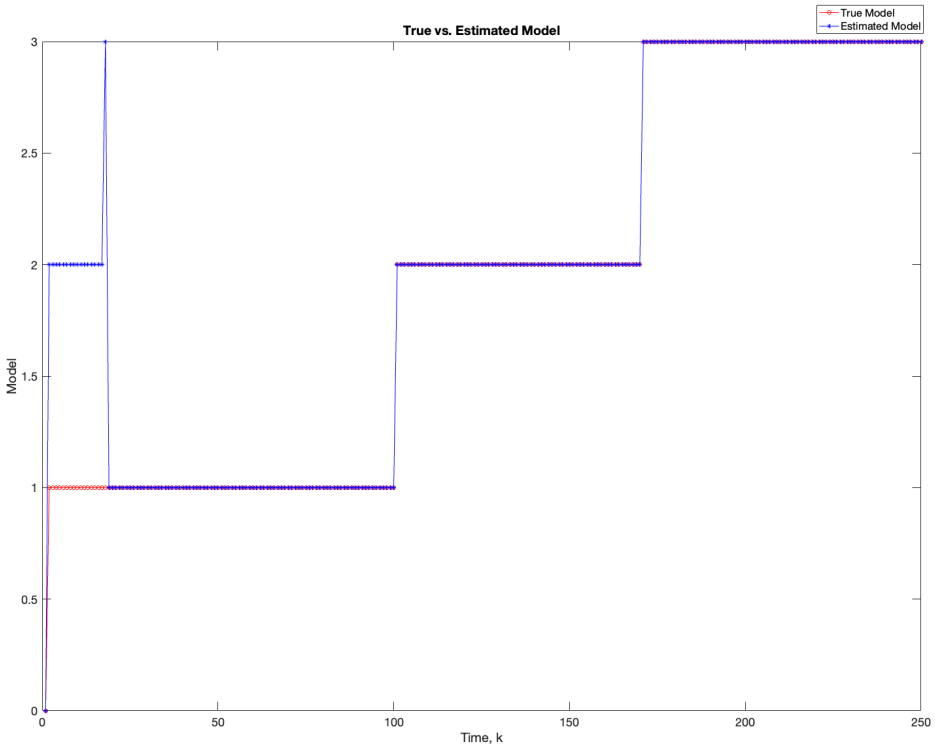


Figure 6.3: The true (red) versus estimated (blue) model or $\Sigma_w = 2e^{-5}\boldsymbol{I}_N$ and $\Sigma_v = 2e^{-3}\boldsymbol{I}_N$. 2,000 Monte Carlo runs and 1,000 particles were used. The dynamics are described by multiplicative noise in degradation for $k = 1 : 100$, multiplicative noise in production for $k = 101 : 170$, and multiplicative noise in production and degradation $k = 171 : 250$.

$171 : 250$. We evaluate the performance of the algorithm under different measurement noise conditions. We assume $\Sigma_w = 2e^{-3}\mathbb{I}_N$ and evaluate the performance under $\Sigma_v = 2e^{-2}\mathbb{I}_N$, $2e^{-3}\mathbb{I}_N$ and $2e^{-5}\mathbb{I}_N$. We compute the average RMSE across all time steps shown in Table 6.5. Our Bayesian learning algorithm is compared to the standard PF, which assumes multiplicative noise in degradation in Figure 6.7. As is shown, the standard particle filter fails to estimate the trajectory once the model switches to a

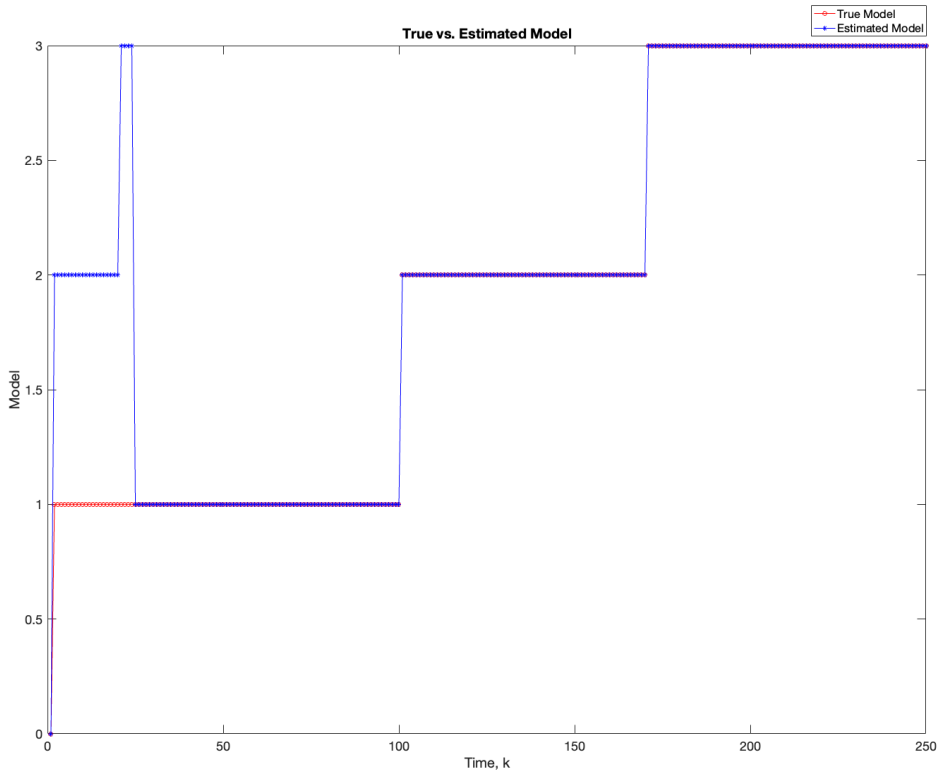


Figure 6.4: The true (red) versus estimated (blue) model or $\Sigma_w = 2e^{-5}\boldsymbol{I}_N$ and $\Sigma_v = 2e^{-2}\boldsymbol{I}_N$. 2,000 Monte Carlo runs and 1,000 particles were used. The dynamics are described by multiplicative noise in degradation for $k = 1 : 100$, multiplicative noise in production for $k = 101 : 170$, and multiplicative noise in production and degradation $k = 171 : 250$.

different noise type. For this specific scenario, our algorithm was one time point late in detecting the change in noise type that occurred at $k = 100$. This is also shown in Figure 6.8. As well, Figure 6.10 shows a peak the time change at $k = 100$ where the model was misidentified. As before, we also consider variation in the process noise covariance under $\Sigma_w = 2e^{-2}\mathbb{I}_N, 2e^{-3}\mathbb{I}_N$ and $2e^{-5}\mathbb{I}_N$ where $\Sigma_v = 2e^{-3}\mathbb{I}_N$. We show the RMSE over time for each variation in the process noise in Figure 6.10.

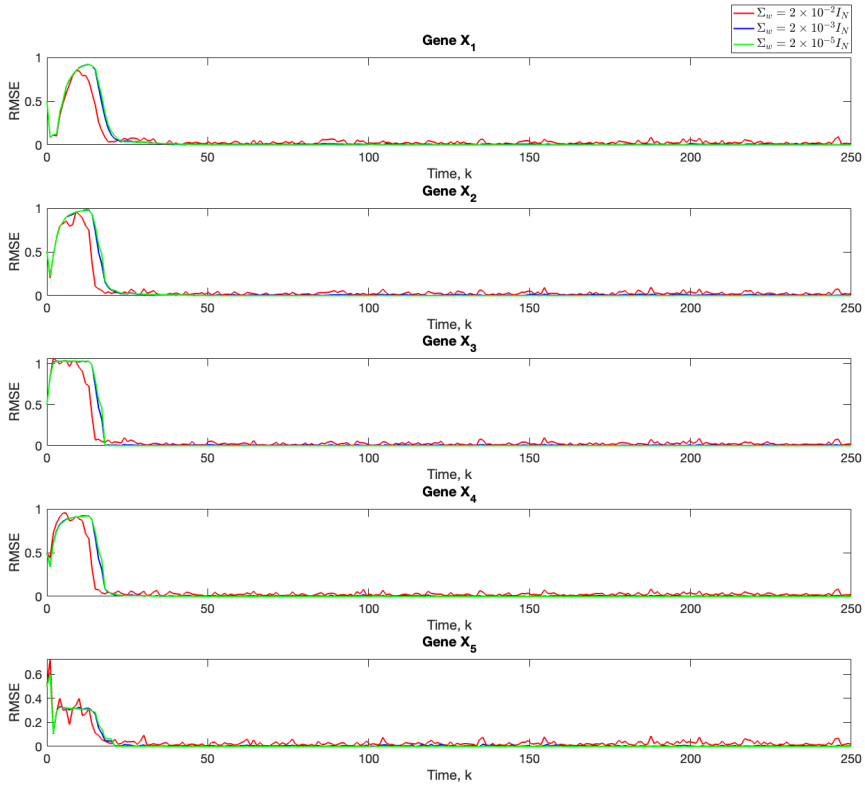


Figure 6.5: A comparison of the RMSE for different values of the process noise covariance. The dynamics are described by multiplicative noise in degradation for $k = 1 : 100$, multiplicative noise in production for $k = 101 : 170$, and multiplicative noise in production and degradation $k = 171 : 250$.

Compared to the previous scenario, we found that the RMSE was significantly lower for this sequence of models when the measurement noise covariance decreases from $\Sigma_w = 2e^{-2}\mathbb{I}_N$ to $\Sigma_w = 2e^{-3}\mathbb{I}_N$. This is also supported by Figure 6.8 and Figure 6.9.

6.5 Conclusion

In this chapter, we introduced a Bayesian hierarchical model for learning different types of stochasticity in GRNs with unknown measurement noise covariance. We

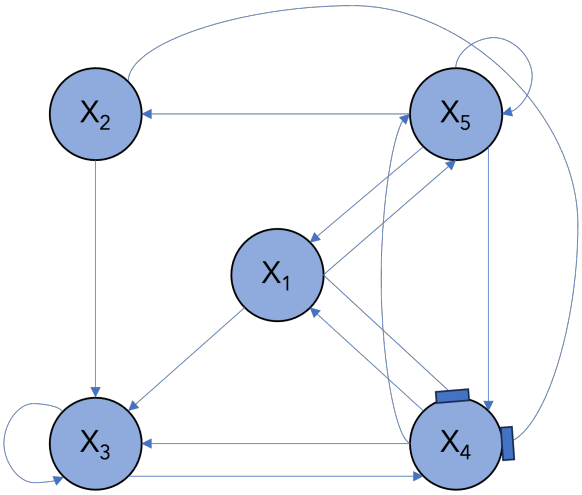


Figure 6.6: Five-gene network used for simulations. Arrows denote activating regulatory interactions and the horizontal bar denotes an inhibiting regulatory interaction.

developed an SMC algorithm for inference. Through simulations we demonstrated that our model can identify changes in the noise type with a high degree of accuracy.

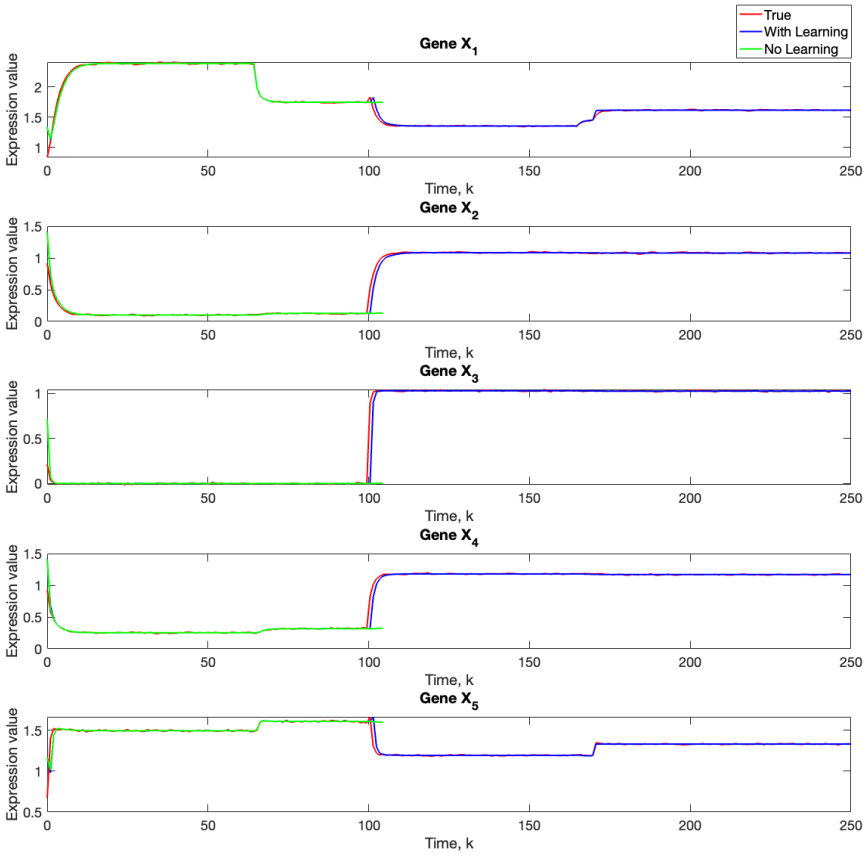


Figure 6.7: A comparison of our learning algorithm (blue) to the standard particle (green) for $\Sigma_w = 2e^{-3}\boldsymbol{I}_N$ and $\Sigma_v = 2e^{-3}\boldsymbol{I}_N$. The true trajectory is given by the red. 2,000 Monte Carlo runs and 1,000 particles were used. The dynamics are described by multiplicative noise in production for $k = 1:100$, multiplicative noise in degradation for $k = 101:170$, and multiplicative noise in production and degradation for $k = 171:250$. Multiplicative noise in production is assumed.

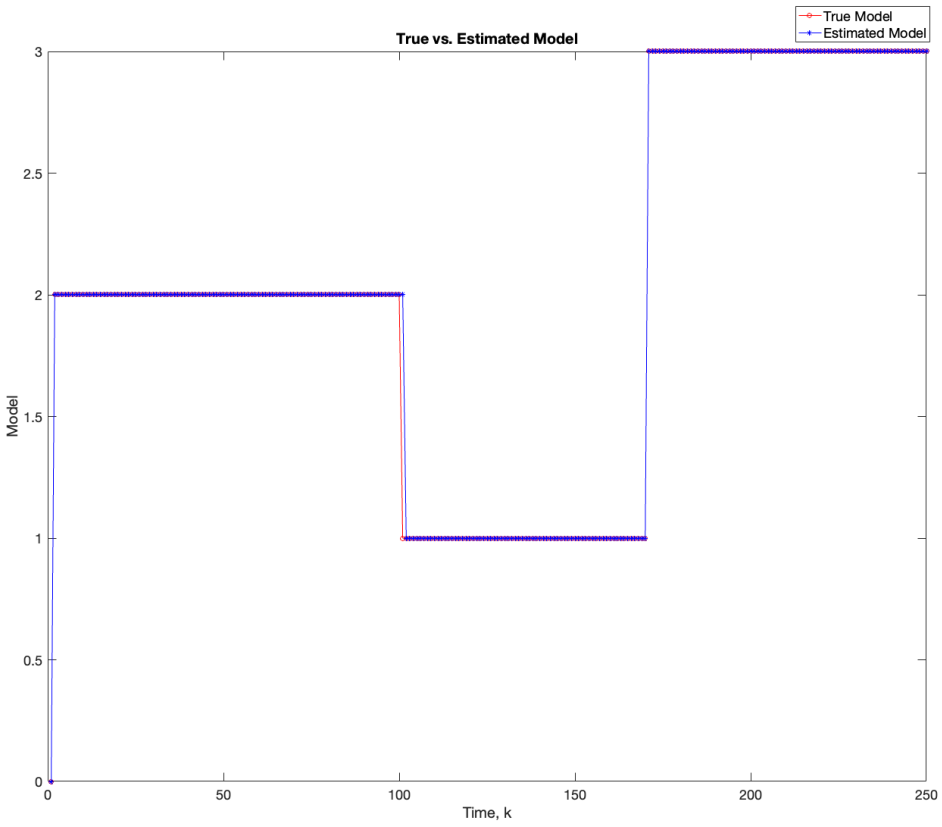


Figure 6.8: The true (red) versus estimated (blue) model or $\Sigma_w = 2e^{-3}\boldsymbol{I}_N$ and $\Sigma_v = 2e^{-3}\boldsymbol{I}_N$. We used 1,000 particles, 2,000 Monte Carlo runs and 1,000 particles were used. The dynamics are described by multiplicative noise in production for $k = 1 : 100$, multiplicative noise in degradation for $k = 101 : 170$, and multiplicative noise in production and degradation for $k = 171 : 250$.

x_i	g_{ij}, h_{ij}	K_{ij}	q_{ij}	$v_i^{\max}$	v_i^d
x_1	$g_{11}=1, h_{11}=1$ $g_{12}=1, h_{12}=1$ $g_{13}=1, h_{13}=1$ $g_{14}=1, h_{14}=0$ $g_{15}=1, h_{15}=0$	$K_{11}=1.0$ $K_{12}=2.0$ $K_{13}=4.0$ $K_{14}=1.0$ $K_{15}=1.0$	1.0	14.0	0.5
x_2	$g_{21}=1, h_{21}=1$ $g_{22}=1, h_{22}=1$ $g_{23}=1, h_{23}=1$ $g_{24}=1, h_{24}=1$ $g_{25}=1, h_{25}=0$	$K_{21}=1.0$ $K_{22}=2.0$ $K_{23}=4.0$ $K_{24}=3.0$ $K_{25}=1.0$	1.0	3.0	0.4
x_3	$g_{31}=1, h_{31}=0$ $g_{32}=1, h_{32}=0$ $g_{33}=1, h_{33}=0$ $g_{34}=1, h_{34}=0$ $g_{35}=1, h_{35}=1$	$K_{31}=1.0$ $K_{32}=2.0$ $K_{33}=4.0$ $K_{34}=3.0$ $K_{35}=1.0$	1.0	4.0	0.9
x_4	$g_{41}=0, h_{41}=1$ $g_{42}=1, h_{42}=0$ $g_{43}=1, h_{43}=0$ $g_{44}=1, h_{44}=1$ $g_{45}=1, h_{45}=0$	$K_{41}=1.0$ $K_{42}=2.0$ $K_{43}=4.0$ $K_{44}=3.0$ $K_{45}=1.0$	1.0	6.0	0.5
x_5	$g_{51}=1, h_{51}=0$ $g_{52}=0, h_{52}=1$ $g_{53}=1, h_{53}=1$ $g_{54}=1, h_{54}=0$ $g_{55}=1, h_{55}=0$	$K_{51}=1.65$ $K_{52}=2.65$ $K_{53}=35.3$ $K_{54}=3.47$ $K_{55}=15.0$	1.0	11.0	0.8

Table 6.4: Parameters of model in Equation (6.2) for the GRN in Figure 1.

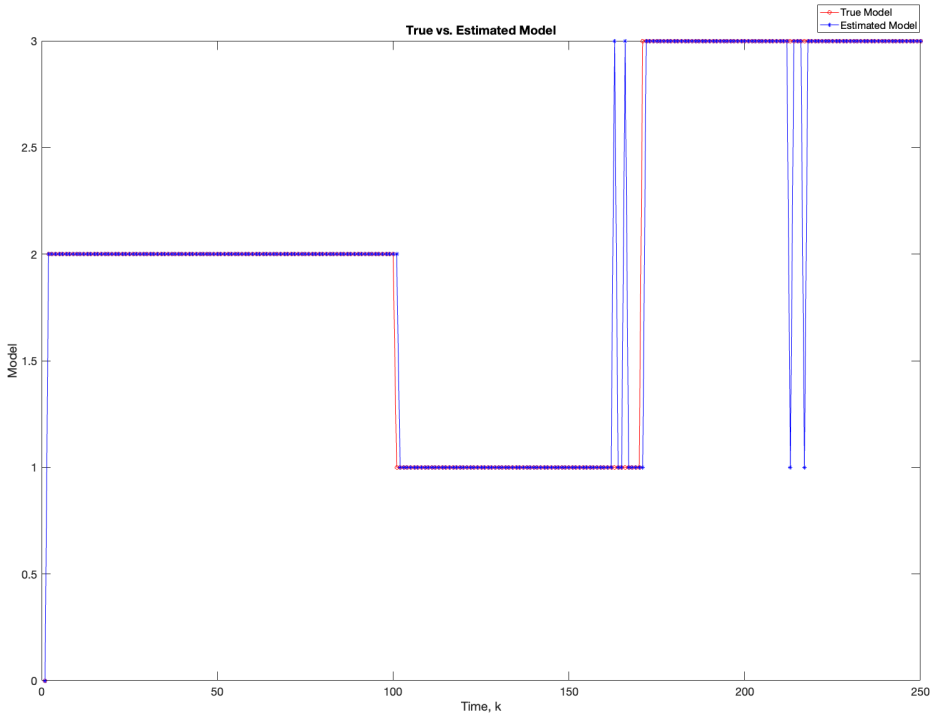


Figure 6.9: The true (red) versus estimated (blue) model or $\Sigma_w = 2e^{-3}\boldsymbol{I}_N$ and $\Sigma_v = 2e^{-2}\boldsymbol{I}_N$. We used 1,000 particles, 2,000 Monte Carlo runs and 1,000 particles were used. The dynamics are described by multiplicative noise in production for $k = 1 : 100$, multiplicative noise in degradation for $k = 101 : 170$, and multiplicative noise in production and degradation for $k = 171 : 250$.

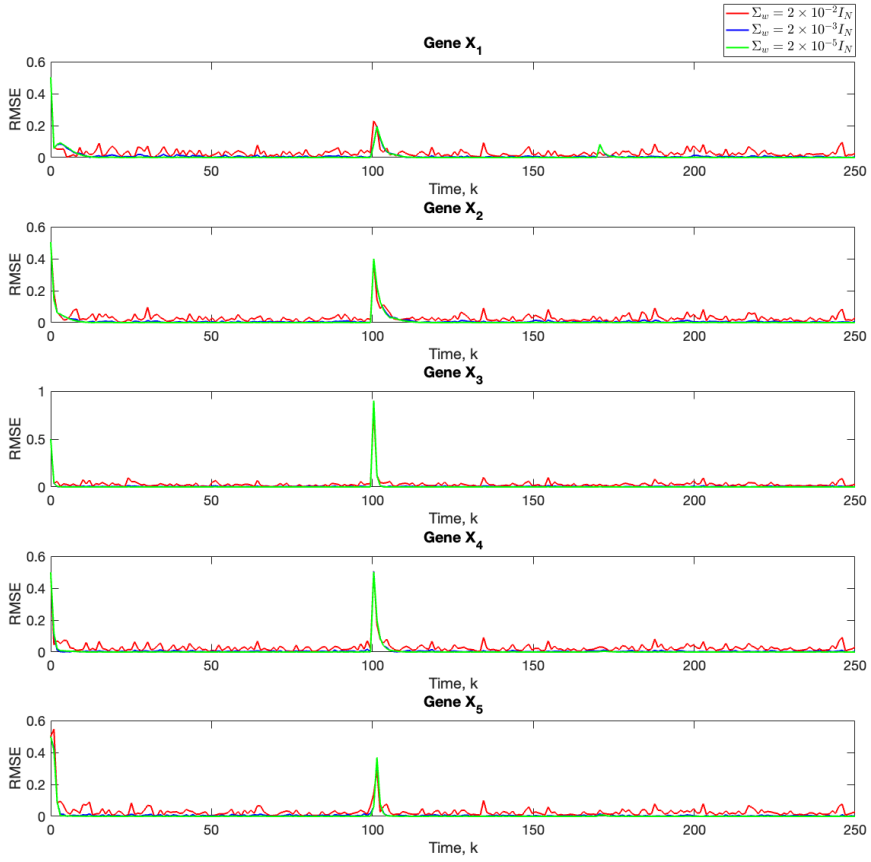


Figure 6.10: A comparison of the RMSE for different values of the process noise covariance. The dynamics are described by multiplicative noise in production for $k = 1 : 100$, multiplicative noise in degradation for $k = 101 : 170$, and multiplicative noise in production and degradation for $k = 171 : 250$.

Table 6.5: Root mean-square error (RMSE) averaged across all time steps for measurement noise covariance $\Sigma_v = 2e^{-2}\mathbb{I}_N, 2e^{3}\mathbb{I}_N$, and $2e^{-3}\mathbb{I}_N$. The process noise covariance is $\Sigma_w = 2e^{-3}\mathbb{I}_N$ 1,000 particles and 2,000 Monte Carlo runs were used. The dynamics are described by multiplicative noise in production for $k = 1 : 100$, multiplicative noise in degradation for $k = 101 : 170$, and multiplicative noise in production and degradation for $k = 171 : 250$.

Gene	$\Sigma_v = 2e^{-2}\mathbb{I}_N$	$\Sigma_v = 2e^{-3}\mathbb{I}_N$	$\Sigma_v = 2e^{-5}\mathbb{I}_N$
X_1	0.073	0.017	0.015
X_2	0.074	0.019	0.018
X_3	0.081	0.015	0.013
X_4	0.019	0.016	0.014
X_5	0.024	0.017	0.017

6.6 Particle Filter Derivation

Here, we derive the steps for the particle filter. At each time step, the joint posterior PDF from which we aim to draw samples is

$$p(\mathbf{x}_{1:k}^{(i)}, \mathbf{s}_{1:k}^{(i)}, \Sigma_v^{(i)}, \boldsymbol{\pi}_{m,1:k}^{(i)} | \mathbf{y}_{1:k}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}, \alpha) \tag{6.32}$$

The importance weights are given by

$$w_k^{(i)} = \frac{p(\mathbf{x}_{1:k}^{(i)}, \mathbf{z}_{1:k}^{(i)}, \Sigma_v^{(i)}, \boldsymbol{\pi}_{m,1:k}^{(i)} | \mathbf{y}_{1:k}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}, \alpha)}{q(\mathbf{x}_{1:k}^{(i)}, \mathbf{z}_{1:k}^{(i)}, \Sigma_v^{(i)}, \boldsymbol{\pi}_{m,1:k}^{(i)} | \mathbf{y}_{1:k}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}, \alpha)} \tag{6.33}$$

The numerator can be factored as

$$p(\mathbf{x}_{1:k}^{(i)}, \mathbf{z}_{1:k}^{(i)}, \Sigma_v^{(i)}, \boldsymbol{\pi}_{m,1:k}^{(i)} | \mathbf{y}_{1:k}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}, \alpha) \tag{6.34}$$

$$= \frac{(\mathbf{x}_{1:k}^{(i)}, \mathbf{y}_{1:k}, \mathbf{z}_{1:k}^{(i)}, \Sigma_v^{(i)}, \boldsymbol{\pi}_{m,1:k}^{(i)}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}, \alpha)}{p(\mathbf{y}_{1:k}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}, \alpha)} \tag{6.35}$$

$$= p(\mathbf{y}_k | \mathbf{y}_{1:k-1}, \mathbf{x}_{1:k}^{(i)}, \Sigma_v^{(i)}, \boldsymbol{\Psi}_v, \nu_v^{(i)}) \frac{p(\mathbf{x}_{1:k}^{(i)}, \mathbf{z}_{1:k}^{(i)}, \boldsymbol{\pi}_{m,1:k}^{(i)} | \mathbf{y}_{1:k-1}, \alpha)}{p(\mathbf{y}_{1:k}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}, \alpha)} \tag{6.36}$$

$$\propto p(\mathbf{y}_k | \mathbf{y}_{1:k-1}, \mathbf{x}_{1:k}^{(i)}, \Sigma_v^{(i)}, \boldsymbol{\Psi}_v, \nu_v^{(i)}) p(\mathbf{x}_{1:k}^{(i)}, \mathbf{z}_{1:k}^{(i)}, \boldsymbol{\pi}_{m,1:k}^{(i)} | \mathbf{y}_{1:k-1}, \alpha) \tag{6.37}$$

$$\propto p(\mathbf{y}_k | \mathbf{x}_k^{(i)}, \Sigma_v^{(i)}, \boldsymbol{\Psi}_v, \nu_v^{(i)}) \tag{6.38}$$

$$\cdot\, p(\mathbf{x}_k^{(i)}, z_k^{(i)}, \pi_{m,k}^{(i)} | \mathbf{y}_{1:k-1}, \mathbf{x}_{1:k-1}^{(i)}, \mathbf{z}_{1:k-1}^{(i)}, \boldsymbol{\pi}_{m,1:k-1}^{(i)}, \alpha) \tag{6.39}$$

$$\cdot\, p(\mathbf{x}_{1:k-1}^{(i)}, \mathbf{z}_{1:k-1}^{(i)}, \boldsymbol{\pi}_{m,1:k-1}^{(i)} | \mathbf{y}_{1:k-1}, \alpha) \tag{6.40}$$

$$\propto p(\mathbf{y}_k | \mathbf{x}_k^{(i)}, \Sigma_v^{(i)}, \boldsymbol{\Psi}_v, \nu_v^{(i)}) p(\mathbf{x}_k^{(i)}, z_k^{(i)}, \pi_{m,k}^{(i)} | \mathbf{x}_{k-1}^{(i)}, z_{k-1}^{(i)}, \boldsymbol{\pi}_{m,1:k-1}^{(i)}, \alpha) \tag{6.41}$$

$$\cdot\, (\mathbf{x}_{1:k-1}^{(i)}, \mathbf{z}_{1:k-1}^{(i)}, \Sigma_v^{(i)}, \boldsymbol{\pi}_{m,1:k-1}^{(i)} | \mathbf{y}_{1:k-1}, \boldsymbol{\Psi}_v, \nu_v^{(i)}, \alpha) \tag{6.42}$$

$$\propto p(\mathbf{y}_k | \mathbf{x}_k^{(i)}, \Sigma_v^{(i)}, \boldsymbol{\Psi}_v, \nu_v^{(i)}) p(\mathbf{x}_k^{(i)} | \mathbf{x}_{k-1}^{(i)}, z_k^{(i)}) \tag{6.43}$$

$$\cdot\, p(\Sigma_v^{(i)} | \boldsymbol{\Psi}_v, \nu_v^{(i)}) p(z_k^{(i)} | z_{k-1}^{(i)}, \pi_{m,k}) p(\pi_{m,k} | \alpha) \tag{6.44}$$

$$\cdot\, p(\mathbf{x}_{1:k-1}^{(i)}, \mathbf{z}_{1:k-1}^{(i)}, \boldsymbol{\pi}_{m,1:k-1}^{(i)}, \Sigma_v^{(i)}, | \mathbf{y}_{1:k-1}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}, \alpha) \tag{6.45}$$

$$\tag{6.46}$$

where due to the Markov property and because $\pi_{m,k}^{(i)}$ does not depend on its previous values

$$p(\mathbf{x}_k^{(i)}, z_k^{(i)}, \pi_{m,k}^{(i)} | \mathbf{y}_{1:k-1}, \mathbf{x}_{1:k-1}^{(i)}, \mathbf{z}_{1:k-1}^{(i)}, \boldsymbol{\pi}_{m,1:k-1}^{(i)}, \alpha) \tag{6.47}$$

$$= p(\mathbf{x}_k^{(i)}, z_k^{(i)}, \pi_{m,k}^{(i)} | \mathbf{x}_{k-1}^{(i)}, z_{k-1}^{(i)}, \alpha) \tag{6.48}$$

$$= p(\mathbf{x}_k^{(i)} | \mathbf{x}_{k-1}^{(i)}, z_k^{(i)}) p(s_k^{(i)} | z_{k-1}^{(i)}, \pi_{m,k}^{(i)}) p(\pi_{m,k}^{(i)} | \alpha) \tag{6.49}$$

Next, we want to select the importance density so that if factorizes as

$$q(\mathbf{x}_{1:k}^{(i)}, \mathbf{z}_{1:k}^{(i)}, \boldsymbol{\pi}_{m,1:k}^{(i)}, \Sigma_v^{(i)} | \mathbf{y}_{1:k}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}, \alpha) \tag{6.50}$$

$$= q(\mathbf{x}_k^{(i)}, z_k^{(i)}, \Sigma_v^{(i)}, \pi_{m,k}^{(i)} | \mathbf{x}_{1:k-1}^{(i)}, \mathbf{z}_{1:k-1}^{(i)}, \boldsymbol{\pi}_{m,1:k-1}^{(i)}, \mathbf{y}_{1:k}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}, \alpha) \tag{6.51}$$

$$\cdot\, q(\mathbf{x}_{1:k-1}^{(i)}, \mathbf{z}_{1:k-1}^{(i)}, \boldsymbol{\pi}_{m,1:k-1}^{(i)}, \Sigma_v^{(i)} | \mathbf{y}_{1:k-1}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}, \alpha) \tag{6.52}$$

For the weights, we have the following

$$w_k^{(i)} = \frac{\left[p(\mathbf{y}_k | \mathbf{x}_k^{(i)}, \Sigma_v^{(i)}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}) p(\mathbf{x}_k^{(i)} | \mathbf{x}_{k-1}^{(i)}, z_k^{(i)}) p(\Sigma_v^{(i)} | \boldsymbol{\Psi}_v, \nu_v^{(i)}) \right.}{\left. \cdot\, p(z_k^{(i)} | z_{k-1}^{(i)}, \pi_{m,k}^{(i)}) p(\pi_{m,k}^{(i)} | \alpha) p(\mathbf{x}_{1:k-1}^{(i)}, \mathbf{z}_{1:k-1}^{(i)}, \boldsymbol{\pi}_{m,1:k-1}^{(i)}, \Sigma_v^{(i)} | \mathbf{y}_{1:k-1}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}, \alpha) \right]}$$
$$\frac{\left[q(\mathbf{x}_k^{(i)}, z_k^{(i)}, \pi_{m,k}^{(i)}, \Sigma_v^{(i)} | \mathbf{x}_{1:k-1}^{(i)}, \mathbf{z}_{1:k-1}^{(i)}, \boldsymbol{\pi}_{m,1:k-1}^{(i)}, \mathbf{y}_{1:k}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}, \alpha) \right.}{\left. \cdot\, q(\mathbf{x}_{1:k-1}^{(i)}, \mathbf{z}_{1:k-1}^{(i)}, \boldsymbol{\pi}_{m,1:k-1}^{(i)}, \Sigma_v^{(i)} | \mathbf{y}_{1:k-1}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)} \alpha) \right]} \tag{6.53}$$

$$= \frac{p(\mathbf{x}_{1:k-1}^{(i)}, \mathbf{z}_{1:k-1}^{(i)}, \boldsymbol{\pi}_{m,1:k-1}^{(i)}, \Sigma_v^{(i)} | \mathbf{y}_{1:k-1}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}, \alpha)}{q(\mathbf{x}_{1:k-1}^{(i)}, \mathbf{z}_{1:k-1}^{(i)}, \boldsymbol{\pi}_{m,1:k-1}^{(i)}, \Sigma_v^{(i)} | \mathbf{y}_{1:k-1}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}, \alpha)} \tag{6.54}$$

$$\cdot \left[p(\mathbf{y}_k | \mathbf{x}_k^{(i)}, \Sigma_v^{(i)}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}) p(\mathbf{x}_k^{(i)} | \mathbf{x}_{k-1}^{(i)}, z_k^{(i)}) \right.$$
$$\left. \cdot\, p(\Sigma_v^{(i)} | \boldsymbol{\Psi}_v, \nu_v^{(i)}) p(z_k^{(i)} | z_{k-1}^{(i)}, \pi_{m,k}^{(i)}) p(\pi_{m,k}^{(i)} | \alpha) \right]$$
$$\cdot \frac{}{q(\mathbf{x}_k^{(i)}, z_k^{(i)}, \pi_{m,k}^{(i)}, \Sigma_v^{(i)} | \mathbf{x}_{1:k-1}^{(i)}, \mathbf{z}_{1:k-1}^{(i)}, \mathbf{y}_{1:k}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}, \alpha)} \tag{6.55}$$

Choosing the importance density to be the prior PDF of the unknown parameters

$$q(\mathbf{x}_k^{(i)}, z_k^{(i)}, \pi_{m,k}^{(i)}, \Sigma_v^{(i)} | \mathbf{x}_{1:k-1}^{(i)}, \mathbf{z}_{1:k-1}^{(i)}, , \boldsymbol{\pi}_{m,1:k-1}^{(i)}, \mathbf{y}_{1:k}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}, \alpha) \tag{6.56}$$

$$= p(\mathbf{x}_k^{(i)}, z_k^{(i)}, \pi_{m,k}^{(i)}, \Sigma_v^{(i)} | \mathbf{x}_{k-1}^{(i)}, z_{k-1}^{(i)}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}, \alpha) \tag{6.57}$$

$$= p(\mathbf{x}_k^{(i)} | \mathbf{x}_{k-1}^{(i)}, z_k^{(i)}) p(z_k^{(i)} | z_{k-1}^{(i)}, \pi_{m,k}^{(i)}) p(\pi_{m,k}^{(i)} | \alpha) p(\Sigma_v^{(i)} | \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}) \tag{6.58}$$

we obtain

$$w_k^{(i)} = w_{k-1}^{(i)} \frac{\begin{bmatrix} p(\mathbf{y}_k | \mathbf{x}_k^{(i)}, \Sigma_v^{(i)}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}) p(\mathbf{x}_k^{(i)} | \mathbf{x}_{k-1}^{(i)}, z_k^{(i)}) p(z_k^{(i)} | z_{k-1}^{(i)}, \pi_{m,k}^{(i)}) \\ \cdot p(\Sigma_v^{(i)} | \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}) p(\pi_{m,k}^{(i)} | \alpha) \end{bmatrix}}{\begin{bmatrix} p(\mathbf{x}_k^{(i)} | \mathbf{x}_{k-1}^{(i)}, z_k^{(i)}) p(z_k^{(i)} | z_{k-1}^{(i)}, \pi_{m,k}^{(i)}) \\ \cdot p(\Sigma_v^{(i)} | \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}) p(\pi_{m,k}^{(i)} | \alpha) \end{bmatrix}} \tag{6.59}$$

$$= w_{k-1}^{(i)} p(\mathbf{y}_k | \mathbf{x}_k^{(i)}, \Sigma_v^{(i)}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}) \tag{6.60}$$

Chapter 7

ESTIMATING TIME-VARYING GENE REGULATORY NETWORKS

7.1 Introduction

Gene regulatory networks (GRNs) are complex systems consisting of genes and a toolkit of molecular elements responsible for coordinating the spatiotemporal allocation of gene expression in every cell of the body. They are involved in the execution of essential biological processes, such as development, metabolism, and responses to environmental signals. One of their key characteristics is dynamicity. The regulatory interactions constituting these networks are not static; rather, they evolve, giving rise to time-dependent GRN architectures. These temporal alterations in regulatory structures have consequences for the biological processes which are encoded by GRNs across both short and long time scales and can affect all major biological processes, including development, disease, and phenotypic evolution. GRNs also exhibit a complex hierarchical architecture comprised of interlinked subunits called "subcircuits." Ranging from two to eight genes, these subcircuits consist of specific regulatory interactions that execute specific biological functions Peter and Davidson (2011). Subcircuits thus intimately link structure and function GRNs Hinman and Cheatle Jarvela (2014). Research shows that these subcircuits are evolutionarily conserved across Metazoa, reinforcing their critical role Peter and Davidson (2015). Various subcircuit types, such as positive-feedback loops, feed-forward connections, and reciprocal repression mechanisms, have been identified Alon (2019). During development, different subcircuits are involved in different temporal stages of the developmental process, such as pattern formation, state stabilization, and cellular differentiation Peter and

Davidson (2015). As development proceeds, the sets of active genes and their associated regulatory interactions change. In regard to disease, the rewiring of regulatory networks can cause disruptions in essential biological functions, giving rise to diseases such as cancer or disorders such as schizophrenia Potkin *et al.* (2010). These rewirings can manifest at the level of subcircuits, which emphasizes their importance in understanding disease mechanisms Saunders and McClay (2014). Such alterations may involve transitions from one subcircuit type to another—for example, a shift from a positive-feedback loop to a feed forward cascade Saunders and McClay (2014). Similarly, evolutionary changes in GRN configurations contribute to the emergence of novel phenotypes within populations Ha *et al.* (2022). Identifying these shifts in GRN interactions, especially within subcircuits, is thus an imperative task for understanding essential biological processes. This is fundamentally a change point detection problem.

To this extent, state-space models provide a mathematical framework that captures the dynamical behavior of GRNs, including their subcircuits, over time. GRN estimation using a state-space approach has been extensively studied Amor *et al.* (2019); Ancherbak *et al.* (2016a); Bugallo *et al.* (2015); Noor *et al.* (2012); Pirgazi and Khanteymoori (2018). However, these approaches assume that the network structure is static across all time. To this extent, several works have considered the problem of estimating GRNs with time-varying structures using linear models. Specifically, the authors in Pirgazi and Khanteymoori (2018); Xiong and Zhou (2013) use Kalman filtering for inference by assuming a linear state-space model. In Dondelinger *et al.* (2013), the network structure assumed to change slowly across time. However, environmental stressors, disease, or mutations may substantially alter the network structure in a more abrupt manner. Furthermore, gene regulation is a nonlinear process due to the feedback loops present in the architectures, threshold effects, and combina-

torial binding of transcription factors on DNA. Linear models may therefore obscure these phenomena. Alternatively, nonlinear system models based on Hill kinetics, Michaelis-Menten kinetics, or the S-system are able to more accurately capture the molecular mechanisms of gene regulation Elahi and Hasan (2018); Wang *et al.* (2007); Youseph *et al.* (2015, 2019). Nonlinear Bayesian filtering inference methods, such as extended Kalman filtering and particle filtering, were used with these nonlinear models Bugallo *et al.* (2015); Wang *et al.* (2009); Zhang *et al.* (2014). In Lee *et al.* (2013), a nonlinear chemical kinetics model is considered, but the transition probabilities for switching between various network structures (or modes) are assumed to be known, which is usually not the case in practice.

Given the limitations of these works, we introduce a Bayesian hierarchical model to account for sudden changes in the network architecture as well as unknown measurement noise covariance. In our model, the regulatory network interactions are encoded by kinetic order parameters, which we assume change at unknown times. The parameters of the categorical distribution are learned using a Dirichlet distribution conjugate prior. The unknown measurement noise covariance is learned using an Inverse-Wishart prior whose hyperparameters are sequentially updated. We employ sequential Monte Carlo (SMC) to choose among the network configurations and infer the model and trajectories of the states, which are the gene expression values. This chapter is organized as follows. In Section 7.2, we introduce our nonlinear state space model for time-varying GRNs and the hierarchical Bayesian model and algorithm that we use for inference. We present simulations demonstrating the efficacy of Bayesian hierarchical modeling in Section 7.3.

7.2 Materials and Methods

7.2.1 Modeling Gene Regulatory Networks

Our model for the expression $n = 1, \ldots, N$ genes, $\mathbf{x}_k$, at discrete time k is

$$\mathbf{x}_k = \boldsymbol{\phi}(\mathbf{x}_{k-1}) + \mathbf{w}_{k-1} \Rightarrow p(\mathbf{x}_k|\mathbf{x}_{k-1}) \tag{7.1}$$

$$\mathbf{y}_k = \mathbf{x}_k + \mathbf{v}_k \Rightarrow p(\mathbf{y}_k|\mathbf{x}_k) \tag{7.2}$$

where $\mathbf{w}_k$ is the Gaussian process noise vector, $\mathbf{v}_k$ is the Gaussian measurement noise vector, $\mathbf{y}_k$ is the vector of noisy measurements, and the state transition function $\boldsymbol{\phi}(\cdot)$ has the following form

$$[\boldsymbol{\phi}(\mathbf{x}_{k-1})]_i = [\boldsymbol{\psi}(\mathbf{x}_{k-1})]_i \left(\prod_{\substack{j=1 \\ j \neq i}}^{N} \mathbf{a}_{ij}\, \mathbf{b}_{ij}(\mathbf{x}_{k-1}) \right) - v_i^d\, x_{i,k-1}$$

$$[\boldsymbol{\psi}(\mathbf{x}_{k-1})]_i = \frac{1}{v_i^{\max}} \left(K_{ii}^{q_{ii}} + x_{i,k-1}^{q_{ii}}\right) \prod_{\substack{j=1 \\ j \neq i}}^{N} (K_{ij}^{q_{ij}} + x_{j,k-1}^{q_{ij}})$$

where

$$\mathbf{a}_{ij} = [g_{ii}g_{ij},\ g_{ii}h_{ij},\ h_{ii}g_{ij},\ h_{ii}, h_{ij}] \tag{7.3}$$

$$\mathbf{b}_{ij}(\mathbf{x}_{k-1}) = [x_{i,k-1}^{q_{ii}} x_{j,k-1}^{q_{ij}},\ x_{i,k-1}^{q_{ii}} K_{ij}^{q_{ij}},\ x_{j,k-1}^{q_{ij}} K_{ii}^{q_{ii}},\ K_{ii}^{q_{ii}} K_{ij}^{q_{ij}}]^T$$

The term v_i^{max} is the maximum expression rate of gene i, and h_{ij} and g_{ij} are the kinetic order parameters that indicate the type of regulation. In particular, when $h_{ij} = 1$ and $g_{ij} = 1$, $i \neq j$, then there is no regulation from gene j to gene i. When $h_{ij} = 0$ and $g_{ij} = 1$, then gene j activates gene i, whereas $h_{ij} = 1, g_{ij} = 0$ implies that gene j inhibits gene i. The parameter K_{ij} is the Michaelis-Menten constant associated with the regulation of gene i by gene j. In the context of gene regulation, the Michaelis-Menten constant K_{ij} accounts for the binding affinity between the product of a gene j and the binding site of a target gene i. Large values of K_{ij} indicate a low

Parameters	Description
$x_{i,k}$	Expression level of Gene i at time step k
	$\mathbf{x}_k = [x_{1,k}\ x_{2,k}\ \ldots\ x_{N,k}]^T$
K_{ij}	Michaelis-Menten constant associated with the regulation of
	Gene i by Gene j
q_{ij}	Hill coefficient
$v_i^{\max}$	Maximum expression rate of the ith gene
v_i^d	Self-decay rate of protein or mRNA expressed by Gene i
h_{ij}, g_{ij}	Kinetic order parameters indicating regulation type
$\mathbf{w}_k$	Modeling error random variable
$y_{i,k}$	Microarray measurement relating to the ith gene at time step k
	$\mathbf{y}_t = [y_{1,k}\ x_{2,k}\ \ldots\ y_{N,k}]^T$
$\mathbf{v}_k$	Measurement noise

Table 7.1: GRN state-space model parameters

binding affinity and low values of K_{ij} indicate a high binding affinity. The term v_i^d is the self decay rate of mRNA expressed by gene i, and q_{ij} is the Hill coefficient. The parameters K_{ii}, q_{ii}, h_{ii} and g_{ii} specify autoregulatory interactions. These are summarized in Table 8.1. The extension of the model in Equation (7.3) to the time-varying case is done in a straightforward manner through the addition of a time index to the kinetic order parameters $g_{ij,k}$ and $h_{ij,k}$, which encode both the direction and type of regulation. The effect of of dynamic changes in the kinetic order parameters in the state transition function $\boldsymbol{\phi}(\mathbf{x}_{k-1})$ is given as follows:

$$\mathbf{a}_{ij,k} = [g_{ii,k}g_{ij,k},\ g_{ii,k}h_{ij,k},\ h_{ii,k}g_{ij,k},\ h_{ii,k}h_{ij,k}] \tag{7.4}$$

7.2.2 Bayesian Hierarchical Modeling

One of the primary advantages of hierarchical Bayesian modeling lies in its ability to integrate prior domain knowledge about specific phenomena, including the dynamic transitions between different subcircuit types within GRNs, aiming to capture their dynamic behaviors and transitions between different types. Although we note that this model can be generalized to any switching nonlinear system. As well, sequential Monte Carlo (SMC) methods can be integrated into this framework, offering the ability to handle non-linear and non-Gaussian processes, which are often the case in biological systems. For these reasons, we employ Bayesian hierarchical modeling as a robust framework for estimating the dynamics and transitions of subcircuits within gene regulatory networks. This approach not only allows us to build upon existing domain knowledge but also to develop models with the complexity necessary for identifying temporal changes in the GRN interactions. In light of these capabilities, we employ Bayesian hierarchical modeling specifically for estimating the types of subcircuits and the gene expression trajectories $\{\mathbf{x}_k\}_{k=1}^K$ in GRNs. The hierarchy specific to our problem is given as follows:

$$\mathbf{x}_k|\mathbf{x}_{k-1}, s_k \sim G_{s_k}(\mathbf{x}_k|\mathbf{x}_{k-1}),\ i = 1, ..., N \tag{7.5}$$

$$\mathbf{y}_k|\mathbf{x}_k, \Sigma_v \sim H(\mathbf{y}_k|\mathbf{x}_k, \Sigma_v) \tag{7.6}$$

$$s_k|s_{k-1}, \pi_{m,k} \sim \text{Cat}(\pi_{m,k}),\ m = 1, ...M \tag{7.7}$$

$$\pi_{m,k}|\alpha \sim \text{Dir}(\alpha) \tag{7.8}$$

$$\Sigma_v|\mathbf{\Psi}_v, \nu_v \sim \text{IW}(\mathbf{\Psi}_v, \nu_v) \tag{7.9}$$

where Σ_v is the measurement noise covariance, $\mathbf{\Psi}_v$ is the $N \times N$ prior scale matrix, and ν_v is the prior degrees of freedom which is chosen to be $N + 2$. Note that $\pi_{m,k}$ is the probability of selecting model m at time k, and α is the concentration parameter

for the Dirichlet distribution. We set $\alpha = 1$. The distribution $G(\cdot)$ and the likelihood $H(\cdot)$ are assumed to be Gaussian. The posterior density needed to estimate the gene expression values $\mathbf{x}_k$ and the indicators which select the network configuration, $\{s_k^m\}_{m=1}^M$, is

$$p(\{\mathbf{x}_k\}_{k=1}^K, \{s_k\}_{k=1}^K, \Sigma_v, \{\pi_{m,k}\}_{m=1,k=1}^{M,K} | \{\mathbf{y}_k\}_{k=1}^K, \mathbf{\Psi}_v, \nu_v, \alpha) \tag{7.10}$$

$$\propto p(\{\mathbf{y}_k\}_{k=1}^K, \{\mathbf{x}_k\}_{k=1}^K, \{s_k^m\}_{k=1,m=1}^{K,M}, \Sigma_v | \alpha, \mathbf{\Psi}_v, \nu_v) \tag{7.11}$$

$$p(\Sigma_v | \mathbf{\Psi}_v, \nu_v) p(\{\pi_{m,k}\}_{m=1}^M | \alpha) \tag{7.12}$$

$$= \prod_{k=1}^K p(\mathbf{y}_k | \mathbf{x}_k, \Sigma_v) \prod_{m=1}^M \left[p(\mathbf{x}_k | \mathbf{x}_{k-1}, s_k, \{\Sigma_w^m\}_{m=1}^M)^{1(s_k=m)} p(s_k | s_{k-1}, \pi_{m,k})^{1(s_k=m)} \right] \tag{7.13}$$

$$p(\Sigma_v | \mathbf{\Psi}_v, \nu_v) p(\{\pi_{m,k}\}_m^M | \alpha). \tag{7.14}$$

Available Parameters at Time $(k-1)$

- Let $\mathbf{c}_{k-1}$ be the vector of size M containing the counts for all models $\mathbf{s}_{1:k-1}$ at time $(k-1)$.
- Let $s_{k-1} = m$ be the indicator for model m at time $k-1$.
- Let Σ_v be the measurement noise covariance.

Available parameters at time k

- Let $\mathbf{c}_k$ be the counts for all models at time k.
- Let $s_k = m$ be the indicator for model m at time k.
- Let Σ_v be the measurement noise covariance.
- Let $\boldsymbol{\pi}_k$ be the vector of transition probabilities associated with each model at time k.

At time k, we draw a new model indicator s_k from a Categorial distribution with Dirichlet-distributed probabilities. The prior probability that model m is selected is

$$p(s_k = k|s_{k-1}) = \frac{c_{m,k-1}}{\sum_m c_{m,k-1} + \alpha} \tag{7.15}$$

Sequential Monte Carlo

We develop a sequential Monte Carlo algorithm. At each time step, the joint posterior PDF from which we aim to draw samples is

$$p(\mathbf{x}_k, s_k, \pi_{m,k}, \Sigma_v|\mathbf{y}_k, \mathbf{\Psi}_v, \nu_v, \alpha) \tag{7.16}$$

$$\propto p(\mathbf{y}_k|\mathbf{x}_k, \Sigma_v)p(\mathbf{x}_k|\mathbf{x}_{k-1}, s_k)p(s_k|s_{k-1}, \pi_{m,k})p(\Sigma_v|\mathbf{\Psi}_v, \nu_v)p(\pi_{m,k}|\alpha) \tag{7.17}$$

$$p(\mathbf{x}_{k-1}, s_{k-1}, \pi_{m,k-1}, \Sigma_v|\mathbf{y}_{k-1}, \mathbf{\Psi}_v, \nu_v, \alpha) \tag{7.18}$$

where $p(\mathbf{x}_k|\mathbf{x}_{k-1}, s_k)$ is given in Section 8.2.1 and $p(\mathbf{x}_{k-1}, s_{k-1}, \pi_{m,k-1}, \Sigma_v|\mathbf{y}_{k-1}, \mathbf{\Psi}_v, \nu_v, \alpha)$ is the posterior from the previous time step. The algorithm proceeds by sampling a model indicator from Q_1 in Equation (7.19)

$$Q_1(s_k = m|\mathbf{s}_{-k}, \mathbf{x}_k, \mathbf{y}_k, \Sigma_v\alpha) = \frac{c_{m,k-1}}{\sum_m c_{m,k-1}} G(\mathbf{x}_k|\mathbf{x}_{k-1}, s_k)H(\mathbf{y}_k|\mathbf{x}_k, \Sigma_v) \tag{7.19}$$

where $\boldsymbol{\pi}_k$ is the vector of transition probabilities at time k for models $m = 1, ..., M$.

Then the measurement $\mathbf{y}_k$ is computed using Equation (7.2). The measurement noise covariance hyperparameters at time step k, which are $\mathbf{\Psi}_v^{'}$ and $\nu_v^{'}$ are computed using Equation (7.21) and Equation (7.22) and a new measurement noise covariance Σ_v is sampled from an inverse-Wishart with the updated hyperparameters.

$$Q_2(\Sigma_v|\mathbf{y}_k, \mathbf{\Psi}_v, \nu_v) = \text{IW}(\mathbf{\Psi}_v^{'}, \nu_v^{'}) \tag{7.20}$$

where the new hyperparameters $\mathbf{\Psi}_v^{'}$ and $\nu_v^{'}$ are incrementally updated at each time

step as

$$\Psi_v^{'} = \Psi_{v,\text{prev}} + (\mathbf{x}_k - \mu_v)(\mathbf{x}_k - \mu_v)^T \tag{7.21}$$

$$\nu_v^{'} = \nu_{v,\text{prev}} + 1 \tag{7.22}$$

where we have one observation at each time step and μ_v is assumed to be $\mathbf{0}$.

Next, we can compute the weights. We select the joint prior PDF as the proposal distribution since the underlying noise in our model follows a Gaussian distribution. Moreover, this approach is more computationally efficient. The joint prior PDF is given as

$$q(\mathbf{x}_k^{(i)}, s_k^{(i)}, \pi_{m,k}^{(i)}, \Sigma_v^{(i)} | \mathbf{x}_{k-1}^{(i)}, s_{k-1}^{(i}, \Sigma_v^{(i)}, \mathbf{\Psi}_v^{(i)}, \nu_v^{(i)}, \alpha) \tag{7.23}$$

$$= p(\mathbf{x}_k^{(i)} | \mathbf{x}_{k-1}^{(i)}, s_k^{(i)}) p(s_k^{(i)} | s_{k-1}^{(i)}, \pi_{m,k}^{(i)}) p(\Sigma_v^{(i)} | \mathbf{\Psi}_v^{(i)}, \nu_v^{(i)}) p(\pi_{m,k}^{(i)} | \alpha) \tag{7.24}$$

The weights can then be computed as

$$w_k^{(i)} \propto w_{k-1}^{(i)} p(\mathbf{y}_k | \mathbf{x}_k^{(i)}, \Sigma_v^{(i)}) \tag{7.25}$$

where the Gaussian likelihood is given as

$$p(\mathbf{y}_k | \mathbf{x}_k^{(i)}, \Sigma_v^{(i)}) = \frac{1}{(2\pi)^{N/2}\sqrt{|\Sigma_v^{(i)}|}} \exp\left(-\frac{1}{2}(\mathbf{y}_k - \mathbf{x}_k^{(i)})^T (\Sigma_v^{(i)})^{-1} (\mathbf{y}_k - \mathbf{x}_k^{(i)}) \right) \tag{7.26}$$

See Appendix A for the full derivation.

7.3 Results and Discussion

7.3.1 Simulation Settings

In this section, we apply our Bayesian learning approach to a gene regulatory network (GRN) consisting of $N = 5$ genes. For each discrete time step k, one of the $m = 1, ..., M$ network configurations, which is indicated by s_k, is chosen probabilistically from a Categorical distribution with Dirichlet-distributed probabilities. We also

Algorithm 2 Sequential Monte Carlo for Subcircuit Detection

Initialization at $t = 1$

for $i = 1, ..., N$ particles **do**

  Sample $\mathbf{x}_1^{(i)} \sim p(\mathbf{x}_1)$

  Sample $s_1^{(i)} \sim \text{Cat}(\boldsymbol{\pi}_1)$

  Sample $\Sigma_v^{(i)} \sim \text{IW}(\boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)})$

end for

Sequential Updates for $t \geq 2$

Sample a new cluster assignment $s_k^{(i)} \sim Q_1$

Sample $\mathbf{x}_k^{(i)} \sim p(\mathbf{x}_k \mid \mathbf{x}_{k-1}^{(i)}, s_k^{(i)})$

Update measurement $\mathbf{y}_k^{(i)} \sim p(\mathbf{y}_k \mid \mathbf{x}_k^{(i)}, \Sigma_v^{(i)})$

Update hyperparameters using Equation (6.26) and Equation (6.27) and sample $\Sigma_v^{(i)} \sim \text{IW}(\boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)})$ from Q_2

Compute particle weights $w_k^{(i)}$ using Section 8.2.4

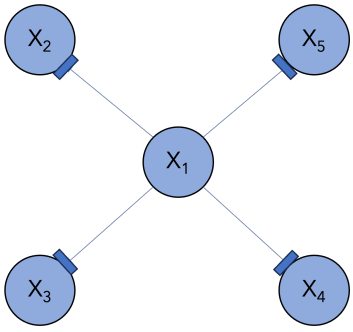
Normalize weights $w_k^{(i)} = \frac{w_k^{(i)}}{\sum_{i=1}^{Ns} w_k^{(i)}}$

Resample particles $\mathbf{x}_k^{(i)}, s_k^{(i)}, \Sigma_v^{(i)}$

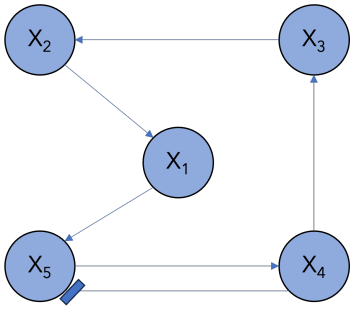
learn the measurement noise covariance. Note that we do not explicitly estimate Σ_v, but rather learn this information associated with each particle in order to estimate the unknown state $\mathbf{x}_k$ and the unknown network configuration indicator s_k. The parameters for the state-space model in Equation (7.1) are given in Table . The kinetic order parameters corresponding to each of the network configurations is given in Table Table 7.5. We evaluate the efficacy of our algorithm across two different scenarios which involve transitions between three network architectures. Each of the networks is depicted in Figure 1. The networks consist of architectures of varying degrees of complexity, the simplest one being a single-input module (SIM) and the more com-

plex ones being a composition of various types of canonical subcircuits. SIMs consist of a master regulator which regulates several genes and are present across a range of systems. For example, Gcn4 is a transcriptional activator of several amino acid biosynthesis genes in *Saccharomyces cerevisiae*, or yeast Mittal *et al.* (2017). In bacteria, LexA is a transcription factor responsible for repressing several genes involved in DNA repair McKenzie *et al.* (2000). Under stressful conditions such as nutrient deprivation, the network may switch to a state which requires the coordination between several genes such as in the feed forward loop depicted in Figure 7.1b. For example, when yeast are subject to osmotic stress, the high-osmolarity glycerol (HOG) pathway is activated which involves the subsequent activation of several genes involved in metabolism regulation, temporary arrest of cell cycle progression, and a milieu of processes required for cellular adaptation in a feed forward architecture Nadal and Posas (2022). Gene regulatory networks can often contain several different types of subcircuits/motifs. An example of such a complex subcircuit is depicted in Figure 7.1c. It consists of several different architectures, including autoinhibitory interactions, in which a gene product negatively regulates its own expression (genes X_1 and X_2), mutual repression loops (genes X_1 and X_3), and feed forward architectures. These types of complex architectures are found for example the hypothalamic-pituitary-adrenal (HPA) axis, which is responsible for regulating the stress response in humans Tsigos and Chrousos (2002). We have selected these specific network structures as they showcase the adaptive capacities of our Bayesian learning approach in identifying architectural changes between networks with varying degrees of complexity, ranging from simple networks like the SIM or feed forward loops, to more intricate architectures consisting of a composition between different subcircuit types. This highlights the versatility of our model.

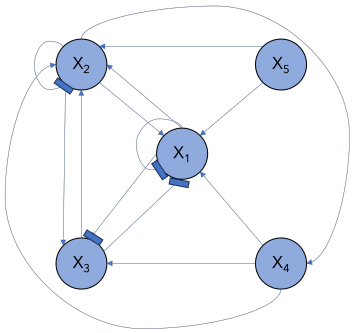
In the first scenario, we consider transitions between $M = 3$ models: a single-



(a) A single-input module (SIM) subcircuit. Gene X_1 inhibits the expression of genes X_2, X_3, X_4, and X_5.



(b) A feed forward loop with an inhibitory interaction. Gene X_1 activates a cascade of activating regulatory interactions proliferating throughout the network. Gene X_5 activates gene X_4, which in turn inhibits the expression of gene X_5.



(c) A complex subcircuit architecture consisting of negative autoinhibitory interactions on genes X_1 and X_2. A feed forward loop structure is also incorporated as X_2 activates the expression of X_4, which activates the expression of X_3 and X_2. Gene X_5 activates the expression of gene X_1, which inhibits the expression of X_3. Gene X_3 inhibits the expression of X_1, forming a mutually repressive loop.

Figure 7.1: Visual depiction of each of the subcircuit models used in this work with varying degrees of complexity.

input module in Figure 7.1a, a feedforward loop with an inhibitory interaction in Figure 7.1b and the complex subcircuit in Figure 7.1c. We consider a time series

consisting of $K = 250$ time steps. For each simulation, we assume that the process and measurement noise are Gaussian as $\mathbf{w}_k \sim \mathcal{N}(\mathbf{0}, \Sigma_w)$ and $\mathbf{v}_k \sim \mathcal{N}(\mathbf{0}, \Sigma_v)$. We use 5,000 Monte Carlo runs and 1,000 particles. We set $\Sigma_w = 3e^{-2}\mathbb{I}_N$ where $\mathbb{I}_N$ is the $N \times N$ identity matrix. For time steps $k = 1 : 65$, the dynamics are described by Figure 7.1a; for time steps $k = 66 : 165$, the dynamics are described by the model in Figure 7.1b; and for time steps $k = 166 : 250$, the dynamics are described by model Figure 7.1c. The prior scale matrix for the Inverse-Wishart is $\mathbf{\Psi}_v = 0.1\mathbb{I}_N$ and the prior degrees of freedom is $\nu_v = N + 2$. A comparison of our learning algorithm to the standard particle filter (no learning) is shown in Figure 7.2. The true versus estimated model is shown in Figure 7.3. Although our algorithm outperforms the standard PF, there is still a delay in identifying the correct model as is shown in the Figure 7.3. The averaged root mean square error (RMSE) values across all time points for different measurement noise intensities are shown in Table 7.2. Overall, the RMSE decreases as the measurement noise decreases.

In the second scenario, for time steps $k = 1 : 65$, the dynamics are described by Figure 7.1b; for time steps $k = 66 : 165$, the dynamics are described by the model in Figure 7.1a; and for time steps $k = 166 : 250$, the dynamics are described by model Figure 7.1c. A comparison of our learning algorithm to the standard particle filter (no learning) is shown in Figure 7.5. The standard particle filter assumes that the dynamics are given by Figure 7.1c. As the figure shows, the standard PF can only estimate the trajectory during the time segment where the true dynamics are given by Figure 7.1c. The true versus estimated model is shown in Figure 7.7. Again, the algorithm exhibits a delay in detecting the time changes. The averaged root mean square error (RMSE) values across all time points for different measurement noise intensities are shown in Table 7.3. As before, the RMSE decreased as the measurement noise intensity decreased. This is also supported by the figures in Figure 7.6, Figure 7.7

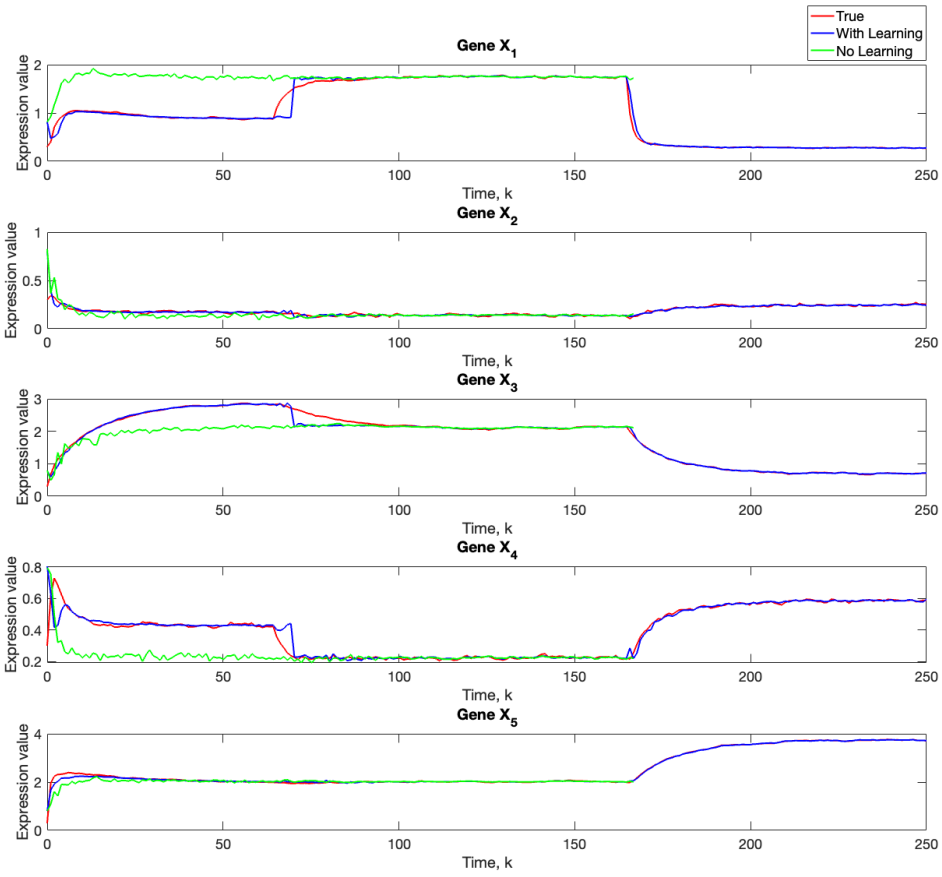


Figure 7.2: A comparison of our learning algorithm (blue) to the standard particle (green) for $\Sigma_w = 3e^{-2}\boldsymbol{I}_N$ and $\Sigma_v = 3e^{-3}\boldsymbol{I}_N$. The true trajectory is given by the red. We used 1,000 particles, 5,000 Monte Carlo runs. For time steps $k = 1 : 65$, the dynamics are described by Figure 7.1a; for time steps $k = 66 : 165$, the dynamics are described by the model in Figure 7.1b; and for time steps $k = 166 : 250$, the dynamics are described by model Figure 7.1c. The dynamics given by Figure 7.1a are assumed.

and Figure 7.8. As is shown, detection performance improves as the measurement noise decreases.

Table 7.2: Root mean-square error (RMSE) averaged across all time steps for measurement noise covariance $\Sigma_v = 3e^{-2}\mathbb{I}_N, 3e^{-3}\mathbb{I}_N$, and $3e^{-5}\mathbb{I}_N$. The process noise covariance is $\Sigma_w = 3e^{-2}\mathbb{I}_N$ 1,000 particles and 5,000 Monte Carlo runs were used. For time steps $k = 1 : 65$, the dynamics are described by Figure 7.1a; for time steps $k = 66 : 165$, the dynamics are described by the model in Figure 7.1b; and for time steps $k = 166 : 250$, the dynamics are described by model Figure 7.1c.

Gene	$\Sigma_v = 3e^{-2}\mathbb{I}_N$	$\Sigma_v = 3e^{-3}\mathbb{I}_N$	$\Sigma_v = 3e^{-5}\mathbb{I}_N$
X_1	0.040	0.034	0.033
X_2	0.017	0.016	0.013
X_3	0.045	0.041	0.036
X_4	0.021	0.019	0.010
X_5	0.035	0.028	0.027

Table 7.3: Root mean-square error (RMSE) averaged across all time steps for measurement noise covariance $\Sigma_v = 3e^{-2}\mathbb{I}_N, 3e^{-3}\mathbb{I}_N$, and $3e^{-5}\mathbb{I}_N$. The process noise covariance is $\Sigma_w = 3e^{-2}\mathbb{I}_N$ 1,000 particles and 5,000 Monte Carlo runs were used. For time steps $k = 1 : 65$, the dynamics are described by Figure 7.1b; for time steps $k = 66 : 165$, the dynamics are described by the model in Figure 7.1a; and for time steps $k = 166 : 250$, the dynamics are described by model Figure 7.1c.

Gene	$\Sigma_v = 3e^{-2}\mathbb{I}_N$	$\Sigma_v = 3e^{-3}\mathbb{I}_N$	$\Sigma_v = 3e^{-5}\mathbb{I}_N$
X_1	0.048	0.044	0.040
X_2	0.042	0.041	0.038
X_3	0.043	0.040	0.036
X_4	0.045	0.042	0.041
X_5	0.046	0.042	0.039

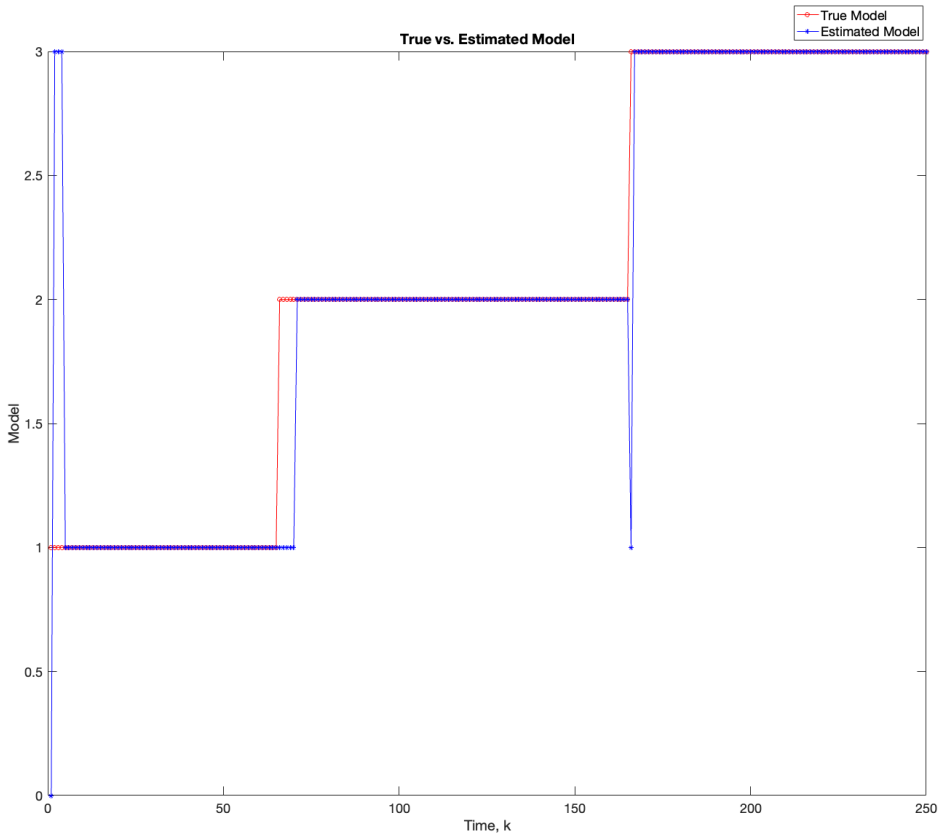


Figure 7.3: The true (red) versus estimated (blue) model for $\Sigma_w = 3e^{-2}\boldsymbol{I}_N$ and $\Sigma_v = 3e^{-3}\boldsymbol{I}_N$. We used 1,000 particles and 5,000 Monte Carlo runs. For time steps $k = 1 : 65$, the dynamics are described by Figure 7.1a; for time steps $k = 66 : 165$, the dynamics are described by the model in Figure 7.1b; and for time steps $k = 166 : 250$, the dynamics are described by model Figure 7.1c.

7.3.2 Tables

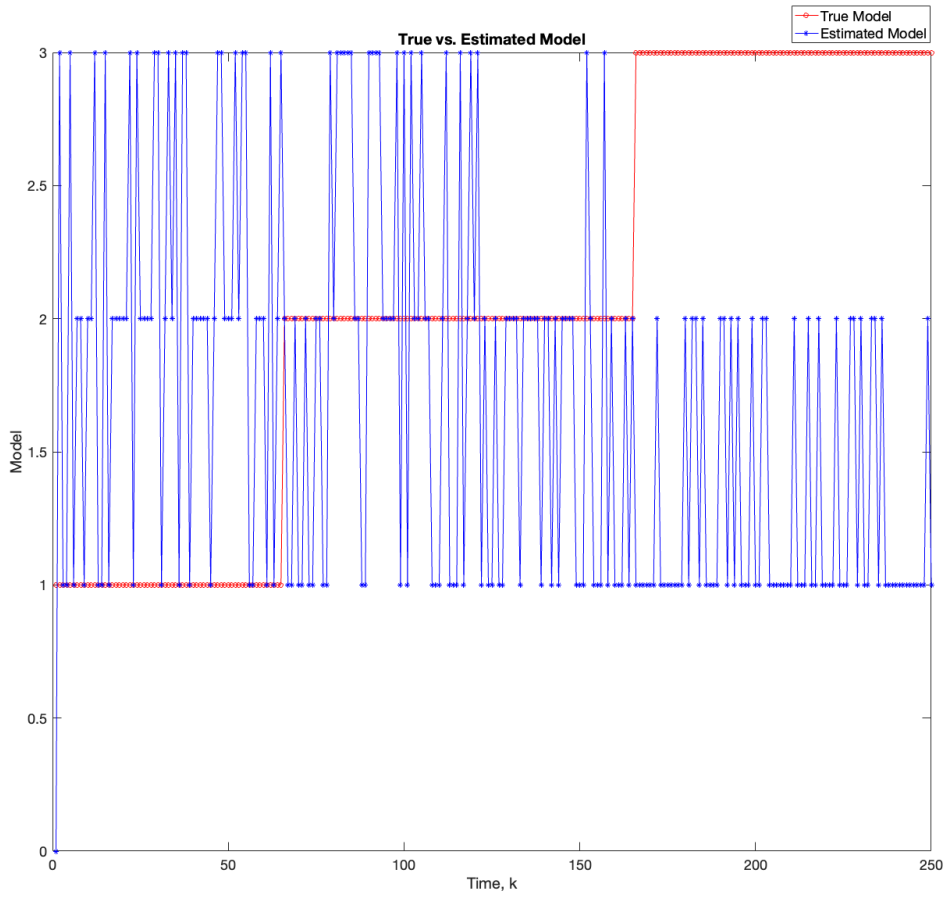


Figure 7.4: The true (red) versus estimated (blue) model for $\Sigma_w = 3e^{-2}\boldsymbol{I}_N$ and $\Sigma_v = 3e^{-3}\boldsymbol{I}_N$ using the standard particle filter (no learning). We used 1,000 particles and 5,000 Monte Carlo runs. For time steps $k = 1 : 65$, the dynamics are described by Figure 7.1a; for time steps $k = 66 : 165$, the dynamics are described by the model in Figure 7.1b; and for time steps $k = 166 : 250$, the dynamics are described by model Figure 7.1c.

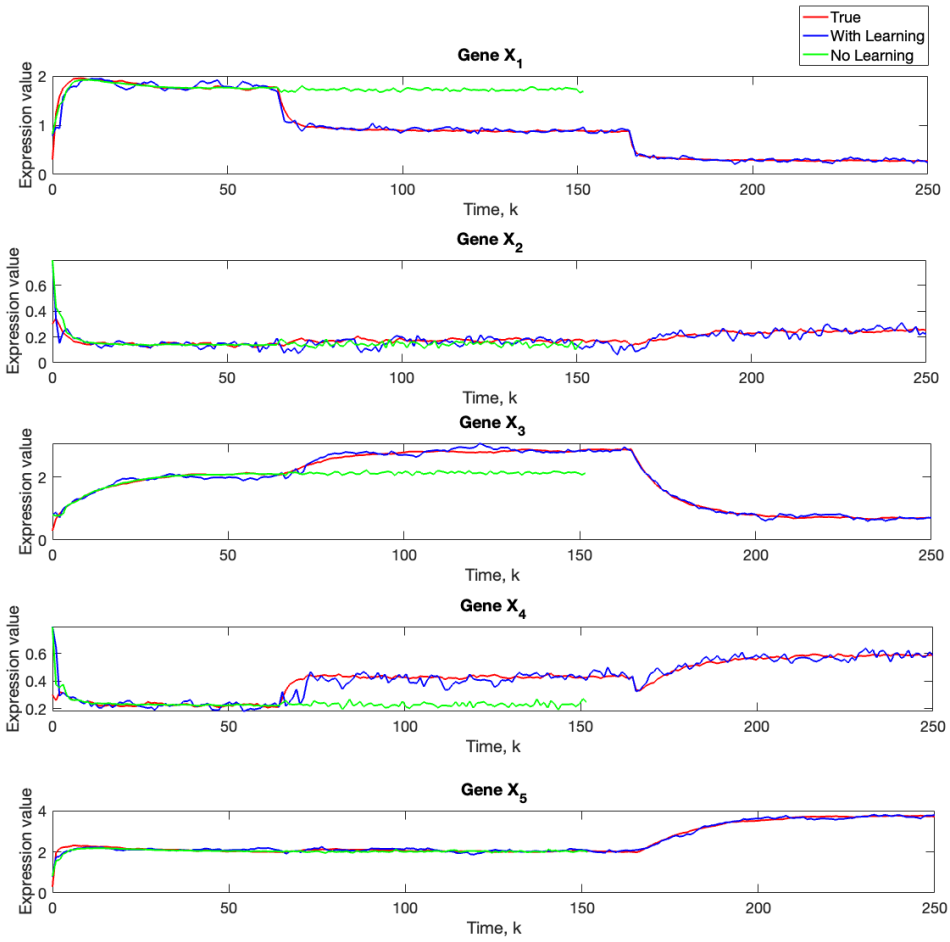


Figure 7.5: A comparison of our learning algorithm (blue) to the standard particle (green) for $\Sigma_w = 3e^{-2}\boldsymbol{I}_N$ and $\Sigma_v = 3e^{-3}\boldsymbol{I}_N$. The true trajectory is given by the red. We used 1,000 particles and 5,000 Monte Carlo runs. For time steps $k = 1 : 65$, the dynamics are described by Figure 7.1b; for time steps $k = 66 : 165$, the dynamics are described by the model in Figure 7.1a; and for time steps $k = 166 : 250$, the dynamics are described by model Figure 7.1c. The dynamics given by Figure 7.1c are assumed.

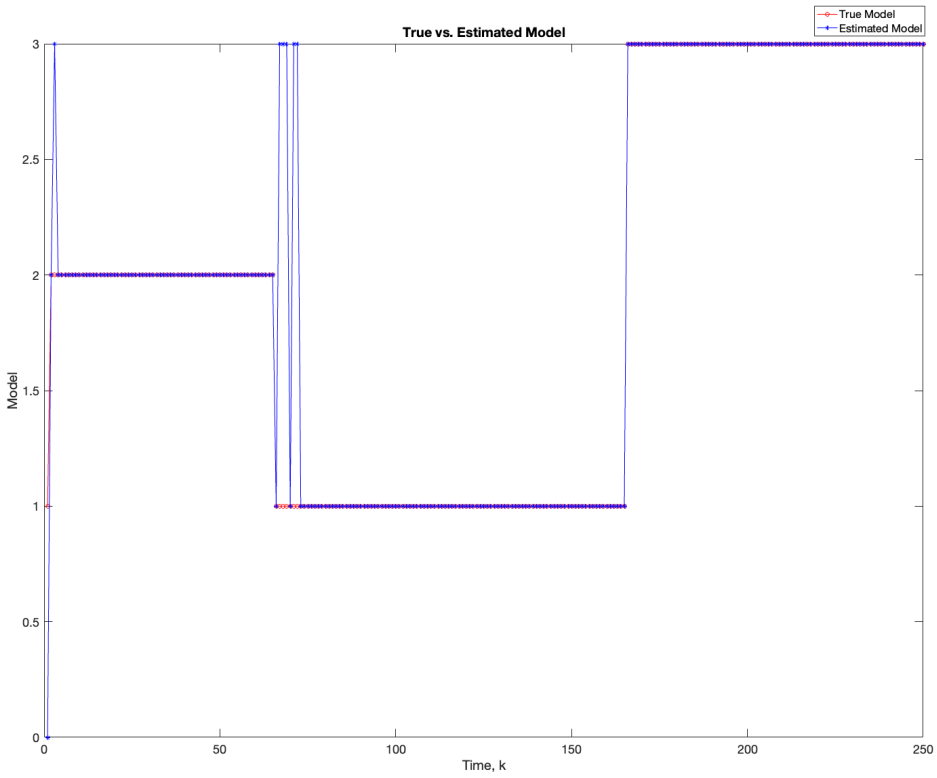


Figure 7.6: The true (red) versus estimated (blue) model for $\Sigma_w = 3e^{-2}\boldsymbol{I}_N$ and $\Sigma_v = 3e^{-2}\boldsymbol{I}_N$. We used 1,000 particles and 5,000 Monte Carlo runs. For time steps $k = 1 : 65$, the dynamics are described by Figure 7.1b; for time steps $k = 66 : 165$, the dynamics are described by the model in Figure 7.1a; and for time steps $k = 166 : 250$, the dynamics are described by model Figure 7.1c.

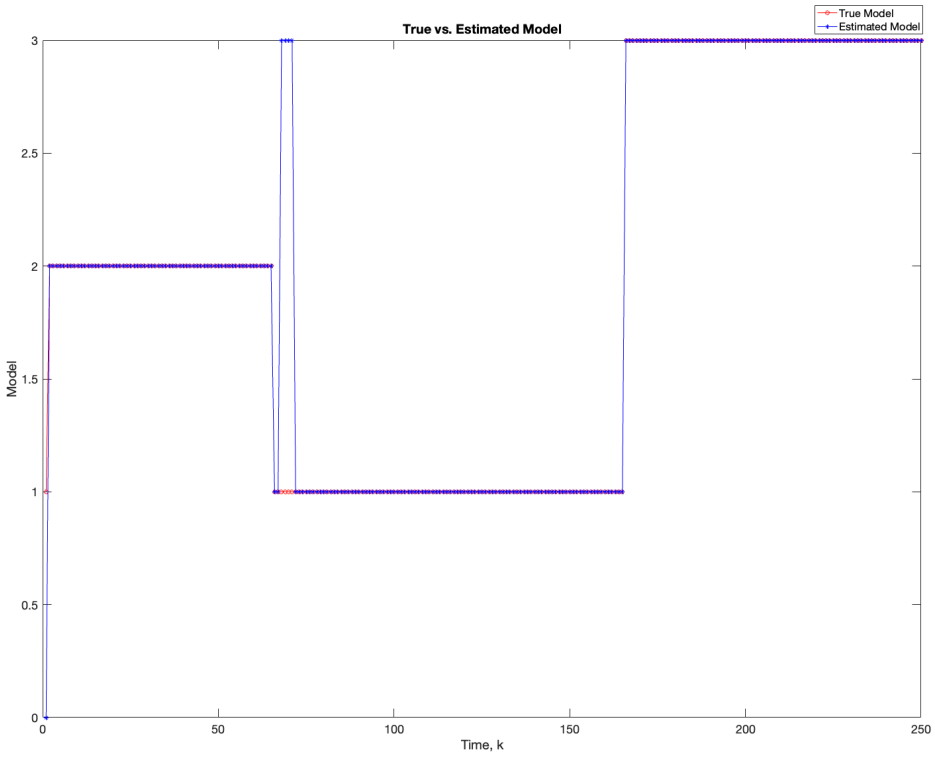


Figure 7.7: The true (red) versus estimated (blue) model for $\Sigma_w = 3e^{-2}\boldsymbol{I}_N$ and $\Sigma_v = 3e^{-3}\boldsymbol{I}_N$.We used 1,000 particles and 5,000 Monte Carlo runs. For time steps $k = 1 : 65$, the dynamics are described by Figure 7.1b; for time steps $k = 66 : 165$, the dynamics are described by the model in Figure 7.1a; and for time steps $k = 166 : 250$, the dynamics are described by model Figure 7.1c.

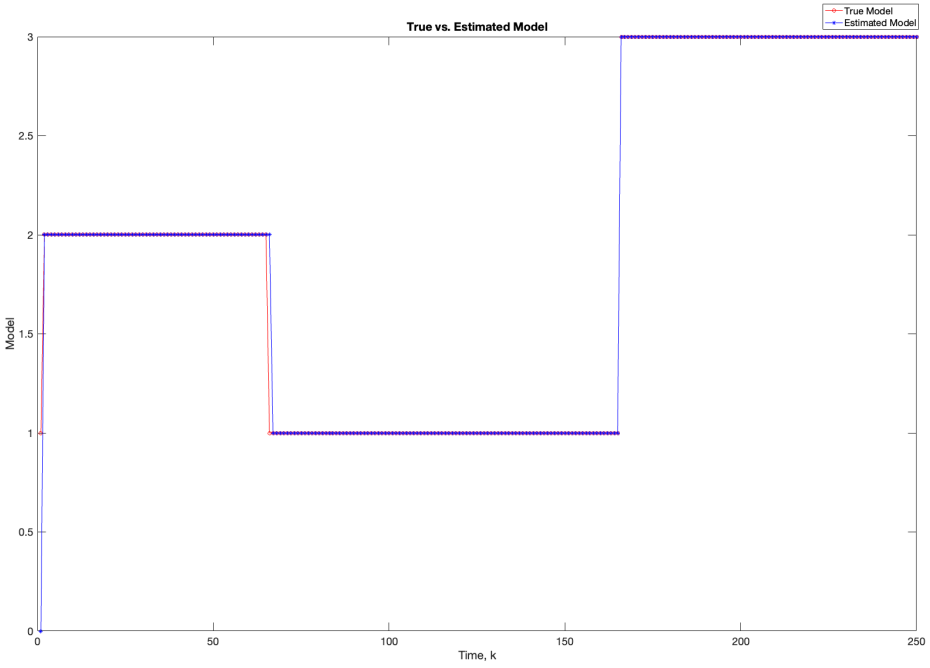


Figure 7.8: The true (red) versus estimated (blue) model for $\Sigma_w = 3e^{-2}\boldsymbol{I}_N$ and $\Sigma_v = 3e^{-5}\boldsymbol{I}_N$.We used 1,000 particles and 5,000 Monte Carlo runs. For time steps $k = 1 : 65$, the dynamics are described by Figure 7.1b; for time steps $k = 66 : 165$, the dynamics are described by the model in Figure 7.1a; and for time steps $k = 166 : 250$, the dynamics are described by model Figure 7.1c.

Table 7.4: Parameter Values for the Five-Gene Network in Figure 1.

Gene ID	**MM Constant** K_{ij}	**Hill Coefficient** q_{ij}	**Max Expression Rate** $V_{\max}$	**Deg. Rate** V_d
X_1	$K_{11} = 1.0$ $K_{12} = 2.0$ $K_{13} = 4.0$ $K_{14} = 1.0$ $K_{15} = 1.0$	1.0	14.0	0.50
X_2	$K_{21} = 1.0$ $K_{22} = 2.0$ $K_{23} = 4.0$ $K_{24} = 3.0$ $K_{25} = 1.0$	1.0	3.0	0.40
X_3	$K_{31} = 1.0$ $K_{32} = 2.0$ $K_{33} = 4.0$ $K_{34} = 3.0$ $K_{35} = 1.0$	1.0	4.0	0.10
X_4	$K_{41} = 1.0$ $K_{42} = 2.0$ $K_{43} = 4.0$ $K_{44} = 3.0$ $K_{45} = 1.0$	1.0	6.0	0.50
X_5	$K_{51} = 1.0$ $K_{52} = 2.0$ $K_{53} = 1.0$ $K_{54} = 1.0$ $K_{55} = 3.0$	1.0	11.0	0.80

Table 7.5: Kinetic order parameters for each of the five models in Figure 1.

Gene	**Kinetic Order Parameters**		
ID	$m = 1$	$m = 2$	$m = 3$
X_1	$g_{11} = 1, h_{11} = 1$	$g_{11} = 1, h_{11} = 1$	$g_{11} = 0, h_{11} = 1$
	$g_{12} = 1, h_{12} = 1$	$g_{12} = 1, h_{12} = 0$	$g_{12} = 1, h_{12} = 0$
	$g_{13} = 1, h_{13} = 1$	$g_{13} = 1, h_{13} = 1$	$g_{13} = 0, h_{13} = 1$
	$g_{14} = 1, h_{14} = 1$	$g_{14} = 1, h_{14} = 1$	$g_{14} = 1, h_{14} = 0$
	$g_{15} = 1, h_{15} = 1$	$g_{15} = 1, h_{15} = 1$	$g_{15} = 0, h_{15} = 1$
X_2	$g_{21} = 0, h_{21} = 1$	$g_{21} = 1, h_{21} = 1$	$g_{21} = 1, h_{21} = 0$
	$g_{22} = 1, h_{22} = 1$	$g_{22} = 1, h_{22} = 1$	$g_{22} = 0, h_{22} = 1$
	$g_{23} = 1, h_{23} = 1$	$g_{23} = 1, h_{23} = 0$	$g_{23} = 1, h_{23} = 0$
	$g_{24} = 1, h_{24} = 1$	$g_{24} = 1, h_{24} = 1$	$g_{24} = 0, h_{24} = 1$
	$g_{25} = 1, h_{25} = 1$	$g_{25} = 1, h_{25} = 1$	$g_{25} = 1, h_{25} = 0$
X_3	$g_{31} = 0, h_{31} = 1$	$g_{31} = 1, h_{31} = 1$	$g_{31} = 0, h_{31} = 1$
	$g_{32} = 1, h_{32} = 1$	$g_{32} = 1, h_{32} = 1$	$g_{32} = 1, h_{32} = 0$
	$g_{33} = 1, h_{33} = 1$	$g_{33} = 1, h_{33} = 1$	$g_{33} = 1, h_{33} = 0$
	$g_{34} = 1, h_{34} = 1$	$g_{34} = 1, h_{34} = 0$	$g_{34} = 1, h_{34} = 0$
	$g_{35} = 1, h_{35} = 1$	$g_{35} = 1, h_{35} = 1$	$g_{35} = 1, h_{35} = 1$
X_4	$g_{41} = 0, h_{41} = 1$	$g_{41} = 1, h_{41} = 1$	$g_{41} = 1, h_{41} = 1$
	$g_{42} = 1, h_{42} = 1$	$g_{42} = 1, h_{42} = 1$	$g_{42} = 0, h_{42} = 1$
	$g_{43} = 1, h_{43} = 1$	$g_{43} = 1, h_{43} = 1$	$g_{43} = 1, h_{43} = 1$
	$g_{44} = 1, h_{44} = 1$	$g_{44} = 1, h_{44} = 1$	$g_{44} = 1, h_{44} = 1$
	$g_{45} = 1, h_{45} = 1$	$g_{45} = 1, h_{45} = 0$	$g_{45} = 1, h_{45} = 1$

Table 7.5 – continued from previous page

Gene	**Kinetic Order Parameters**		
ID	$m = 1$	$m = 2$	$m = 3$
X_5	$g_{51} = 0, h_{51} = 1$	$g_{51} = 1, h_{51} = 0$	$g_{51} = 1, h_{51} = 1$
	$g_{52} = 1, h_{52} = 1$	$g_{52} = 1, h_{52} = 1$	$g_{52} = 1, h_{52} = 1$
	$g_{53} = 1, h_{53} = 1$	$g_{53} = 1, h_{53} = 1$	$g_{53} = 1, h_{53} = 1$
	$g_{54} = 1, h_{54} = 1$	$g_{54} = 0, h_{54} = 1$	$g_{54} = 1, h_{54} = 1$
	$g_{55} = 1, h_{55} = 1$	$g_{55} = 1, h_{55} = 1$	$g_{55} = 1, h_{55} = 1$

7.4 Particle Filter Derivation

Here, we derive the steps for the particle filter. At each time step, the joint posterior PDF from which we aim to draw samples is

$$p(\mathbf{x}_{1:k}^{(i)}, \mathbf{s}_{1:k}^{(i)}, \Sigma_v^{(i)}, \boldsymbol{\pi}_{m,1:k}^{(i)} | \mathbf{y}_{1:k}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}, \alpha) \tag{7.27}$$

The importance weights are given by

$$w_k^{(i)} = \frac{p(\mathbf{x}_{1:k}^{(i)}, \mathbf{s}_{1:k}^{(i)}, \Sigma_v^{(i)}, \boldsymbol{\pi}_{m,1:k}^{(i)} | \mathbf{y}_{1:k}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}, \alpha)}{q(\mathbf{x}_{1:k}^{(i)}, s_{1:k}^{(i)}, \Sigma_v^{(i)}, \boldsymbol{\pi}_{m,1:k}^{(i)} | \mathbf{y}_{1:k}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}, \alpha)} \tag{7.28}$$

The numerator can be factored as

$$p(\mathbf{x}_{1:k}^{(i)}, \mathbf{s}_{1:k}^{(i)}, \Sigma_v^{(i)}, \boldsymbol{\pi}_{m,1:k}^{(i)} | \mathbf{y}_{1:k}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}, \alpha) \tag{7.29}$$

$$= \frac{(\mathbf{x}_{1:k}^{(i)}, \mathbf{y}_{1:k}, \mathbf{s}_{1:k}^{(i)}, \Sigma_v^{(i)}, \boldsymbol{\pi}_{m,1:k}^{(i)}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}, \alpha)}{p(\mathbf{y}_{1:k}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}, \alpha)} \tag{7.30}$$

$$= p(\mathbf{y}_k | \mathbf{y}_{1:k-1}, \mathbf{x}_{1:k}^{(i)}, \Sigma_v^{(i)}, \boldsymbol{\Psi}_v, \nu_v^{(i)}) \frac{p(\mathbf{x}_{1:k}^{(i)}, \mathbf{s}_{1:k}^{(i)}, \boldsymbol{\pi}_{m,1:k}^{(i)} | \mathbf{y}_{1:k-1}, \alpha)}{p(\mathbf{y}_{1:k}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}, \alpha)} \tag{7.31}$$

$$\propto p(\mathbf{y}_k | \mathbf{y}_{1:k-1}, \mathbf{x}_{1:k}^{(i)}, \Sigma_v^{(i)}, \boldsymbol{\Psi}_v, \nu_v^{(i)}) p(\mathbf{x}_{1:k}^{(i)}, \mathbf{s}_{1:k}^{(i)}, \boldsymbol{\pi}_{m,1:k}^{(i)} | \mathbf{y}_{1:k-1}, \alpha) \tag{7.32}$$

$$\propto p(\mathbf{y}_k | \mathbf{x}_k^{(i)}, \Sigma_v^{(i)}, \boldsymbol{\Psi}_v, \nu_v^{(i)}) \tag{7.33}$$

$$\cdot\, p(\mathbf{x}_k^{(i)}, s_k^{(i)}, \pi_{m,k}^{(i)} | \mathbf{y}_{1:k-1}, \mathbf{x}_{1:k-1}^{(i)}, \mathbf{s}_{1:k-1}^{(i)}, \boldsymbol{\pi}_{m,1:k-1}^{(i)}, \alpha) \tag{7.34}$$

$$\cdot\, p(\mathbf{x}_{1:k-1}^{(i)}, \mathbf{s}_{1:k-1}^{(i)}, \boldsymbol{\pi}_{m,1:k-1}^{(i)} | \mathbf{y}_{1:k-1}, \alpha) \tag{7.35}$$

$$\propto p(\mathbf{y}_k | \mathbf{x}_k^{(i)}, \Sigma_v^{(i)}, \boldsymbol{\Psi}_v, \nu_v^{(i)}) p(\mathbf{x}_k^{(i)}, s_k^{(i)}, \pi_{m,k}^{(i)} | \mathbf{x}_{k-1}^{(i)}, \mathbf{s}_{k-1}^{(i)}, \boldsymbol{\pi}_{m,1:k-1}^{(i)}, \alpha) \tag{7.36}$$

$$\cdot\, (\mathbf{x}_{1:k-1}^{(i)}, \mathbf{s}_{1:k-1}^{(i)}, \Sigma_v^{(i)}, \boldsymbol{\pi}_{m,1:k-1}^{(i)} | \mathbf{y}_{1:k-1}, \boldsymbol{\Psi}_v, \nu_v^{(i)}, \alpha) \tag{7.37}$$

$$\propto p(\mathbf{y}_k | \mathbf{x}_k^{(i)}, \Sigma_v^{(i)}, \boldsymbol{\Psi}_v, \nu_v^{(i)}) p(\mathbf{x}_k^{(i)} | \mathbf{x}_{k-1}^{(i)}, s_k^{(i)}) \tag{7.38}$$

$$\cdot\, p(\Sigma_v^{(i)} | \boldsymbol{\Psi}_v, \nu_v^{(i)}) p(s_k^{(i)} | s_{k-1}^{(i)}, \pi_{m,k}) p(\pi_{m,k} | \alpha) \tag{7.39}$$

$$\cdot\, p(\mathbf{x}_{1:k-1}^{(i)}, \mathbf{s}_{1:k-1}^{(i)}, \boldsymbol{\pi}_{m,1:k-1}^{(i)}, \Sigma_v^{(i)}, | \mathbf{y}_{1:k-1}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}, \alpha) \tag{7.40}$$

$$\tag{7.41}$$

where due to the Markov property and because $\pi_{m,k}^{(i)}$ does not depend on its previous values

$$p(\mathbf{x}_k^{(i)}, s_k^{(i)}, \pi_{m,k}^{(i)} | \mathbf{y}_{1:k-1}, \mathbf{x}_{1:k-1}^{(i)}, \mathbf{s}_{1:k-1}^{(i)}, \boldsymbol{\pi}_{m,1:k-1}^{(i)}, \alpha) \tag{7.42}$$

$$= p(\mathbf{x}_k^{(i)}, s_k^{(i)}, \pi_{m,k}^{(i)} | \mathbf{x}_{k-1}^{(i)}, s_{k-1}^{(i)}, \alpha) \tag{7.43}$$

$$= p(\mathbf{x}_k^{(i)} | \mathbf{x}_{k-1}^{(i)}, s_k^{(i)}) p(s_k^{(i)} | s_{k-1}^{(i)}, \pi_{m,k}^{(i)}) p(\pi_{m,k}^{(i)} | \alpha) \tag{7.44}$$

Next, we want to select the importance density so that if factorizes as

$$q(\mathbf{x}_{1:k}^{(i)}, s_{1:k}^{(i)}, \boldsymbol{\pi}_{m,1:k}^{(i)}, \Sigma_v^{(i)} | \mathbf{y}_{1:k}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}, \alpha) \tag{7.45}$$

$$= q(\mathbf{x}_k^{(i)}, s_k^{(i)}, \Sigma_v^{(i)}, \pi_{m,k}^{(i)} | \mathbf{x}_{1:k-1}^{(i)}, \mathbf{s}_{1:k-1}^{(i)}, \boldsymbol{\pi}_{m,1:k-1}^{(i)}, \mathbf{y}_{1:k}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}, \alpha) \tag{7.46}$$

$$\cdot\, q(\mathbf{x}_{1:k-1}^{(i)}, \mathbf{s}_{1:k-1}^{(i)}, \boldsymbol{\pi}_{m,1:k-1}^{(i)}, \Sigma_v^{(i)} | \mathbf{y}_{1:k-1}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}, \alpha) \tag{7.47}$$

For the weights, we have the following

$$w_k^{(i)} = \frac{\left[p(\mathbf{y}_k | \mathbf{x}_k^{(i)}, \Sigma_v^{(i)}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}) p(\mathbf{x}_k^{(i)} | \mathbf{x}_{k-1}^{(i)}, s_k^{(i)}) p(\Sigma_v^{(i)} | \boldsymbol{\Psi}_v, \nu_v^{(i)}) \right.}{\left. \cdot\, p(s_k^{(i)} | s_{k-1}^{(i)}, \pi_{m,k}^{(i)}) p(\pi_{m,k}^{(i)} | \alpha) p(\mathbf{x}_{1:k-1}^{(i)}, \mathbf{s}_{1:k-1}^{(i)}, \boldsymbol{\pi}_{m,1:k-1}^{(i)}, \Sigma_v^{(i)} | \mathbf{y}_{1:k-1}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}, \alpha) \right]}{\left[q(\mathbf{x}_k^{(i)}, s_k^{(i)}, \pi_{m,k}^{(i)}, \Sigma_v^{(i)} | \mathbf{x}_{1:k-1}^{(i)}, \mathbf{s}_{1:k-1}^{(i)}, \boldsymbol{\pi}_{m,1:k-1}^{(i)}, \mathbf{y}_{1:k}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}, \alpha) \right.} \tag{7.48}$$

$$\left. \cdot\, q(\mathbf{x}_{1:k-1}^{(i)}, \mathbf{s}_{1:k-1}^{(i)}, \boldsymbol{\pi}_{m,1:k-1}^{(i)}, \Sigma_v^{(i)} | \mathbf{y}_{1:k-1}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)} \alpha) \right]$$

$$= \frac{p(\mathbf{x}_{1:k-1}^{(i)}, \mathbf{s}_{1:k-1}^{(i)}, \boldsymbol{\pi}_{m,1:k-1}^{(i)}, \Sigma_v^{(i)} | \mathbf{y}_{1:k-1}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}, \alpha)}{q(\mathbf{x}_{1:k-1}^{(i)}, \mathbf{s}_{1:k-1}^{(i)}, \boldsymbol{\pi}_{m,1:k-1}^{(i)}, \Sigma_v^{(i)} | \mathbf{y}_{1:k-1}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}, \alpha)} \tag{7.49}$$

$$\cdot \left[p(\mathbf{y}_k | \mathbf{x}_k^{(i)}, \Sigma_v^{(i)}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}) p(\mathbf{x}_k^{(i)} | \mathbf{x}_{k-1}^{(i)}, s_k^{(i)}) \right.$$

$$\left. \cdot\, p(\Sigma_v^{(i)} | \boldsymbol{\Psi}_v, \nu_v^{(i)}) p(s_k^{(i)} | s_{k-1}^{(i)}, \pi_{m,k}^{(i)}) p(\pi_{m,k}^{(i)} | \alpha) \right]$$

$$\cdot \frac{}{q(\mathbf{x}_k^{(i)}, s_k^{(i)}, \pi_{m,k}^{(i)}, \Sigma_v^{(i)} | \mathbf{x}_{1:k-1}^{(i)}, \mathbf{s}_{1:k-1}^{(i)}, \mathbf{y}_{1:k}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}, \alpha)} \tag{7.50}$$

Choosing the importance density to be the prior PDF of the unknown parameters

$$q(\mathbf{x}_k^{(i)}, s_k^{(i)}, \pi_{m,k}^{(i)}, \Sigma_v^{(i)} | \mathbf{x}_{1:k-1}^{(i)}, \mathbf{s}_{1:k-1}^{(i)}, , \boldsymbol{\pi}_{m,1:k-1}^{(i)}, \mathbf{y}_{1:k}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}, \alpha) \tag{7.51}$$

$$= p(\mathbf{x}_k^{(i)}, s_k^{(i)}, \pi_{m,k}^{(i)}, \Sigma_v^{(i)} | \mathbf{x}_{k-1}^{(i)}, s_{k-1}^{(i)}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}, \alpha) \tag{7.52}$$

$$= p(\mathbf{x}_k^{(i)} | \mathbf{x}_{k-1}^{(i)}, s_k^{(i)}) p(s_k^{(i)} | s_{k-1}^{(i)}, \pi_{m,k}^{(i)}) p(\pi_{m,k}^{(i)} | \alpha) p(\Sigma_v^{(i)} | \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}) \tag{7.53}$$

we obtain

$$w_k^{(i)} = w_{k-1}^{(i)} \frac{\begin{bmatrix} p(\mathbf{y}_k | \mathbf{x}_k^{(i)}, \Sigma_v^{(i)}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}) p(\mathbf{x}_k^{(i)} | \mathbf{x}_{k-1}^{(i)}, s_k^{(i)}) p(s_k^{(i)} | s_{k-1}^{(i)}, \pi_{m,k}^{(i)}) \\ \cdot p(\Sigma_v^{(i)} | \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}) p(\pi_{m,k}^{(i)} | \alpha) \end{bmatrix}}{\begin{bmatrix} p(\mathbf{x}_k^{(i)} | \mathbf{x}_{k-1}^{(i)}, s_k^{(i)}) p(s_k^{(i)} | s_{k-1}^{(i)}, \pi_{m,k}^{(i)}) \\ \cdot p(\Sigma_v^{(i)} | \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}) p(\pi_{m,k}^{(i)} | \alpha) \end{bmatrix}} \tag{7.54}$$

$$= w_{k-1}^{(i)} p(\mathbf{y}_k | \mathbf{x}_k^{(i)}, \Sigma_v^{(i)}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}) \tag{7.55}$$

7.5 Conclusion

Overall, our algorithm can identify the correct model with a high degree of accuracy under varying measurement noise conditions. In general, the RMSE increases as the measurement noise increases.

Chapter 8

BAYESIAN LEARNING OF NONLINEAR GRNS WITH SWITCHING ARCHITECTURES

8.1 Introduction

The previous chapter introduced a Bayesian hierarchical model for identifying structural changes in gene regulatory networks (GRNs). However, the process noise is assumed to be known and the measurement noise is assumed to be known and constant as in Bugallo *et al.* (2015); Noor *et al.* (2012); Ancherbak *et al.* (2016b), where particle filtering is used to infer the dynamic network. This may not capture changes in the noise statistics that can arise when a change in the regulatory network structure occurs. Given this limitation, we extend the work of the previous chapter to account for changing noise statistics that arise when an architectural shift occurs. We introduce a fully Bayesian hierarchical model to account for the following: 1) sudden changes in the network architecture, 2) unknown process noise covariance which may change along with the network structure, and 3) unknown measurement noise covariance. Fully Bayesian hierarchical models have several advantages including their ability to account for uncertainty at all levels, flexibility, incorporation of prior information, and are less prone to overfitting. Furthermore, the use of conjugate priors enables more tractable and efficient inference. In our model, the regulatory network interactions are encoded by kinetic order parameters, which we assume change at unknown times. The parameters of the categorical distribution are learned using a Dirichlet distribution conjugate prior. The unknown measurement and process noise covariances are learned using Inverse-Wishart priors whose hyperparameters are

sequentially updated. We employ sequential Monte Carlo (SMC) with local Gibbs sampling to choose among the network configurations and infer the model and trajectories of the states, which are the gene expression values. This paper is organized as follows. In Section 2, we introduce our nonlinear state space model for time-varying GRNs and the fully hierarchical Bayesian model and algorithm that we use for inference. We present simulations demonstrating the efficacy of Bayesian hierarchical modeling in Section 3.

8.2 Materials and Methods

8.2.1 Modeling Gene Regulatory Networks

In Vélez-Cruz *et al.* (2021), we developed a stochastic nonlinear GRN model to capture the molecular mechanisms of gene regulation and account for inherent variability in biological processes. The model includes information on binding affinity, expression and self-decay rates. It is based on Michaelis-Menten kinetics that describe different types of regulation processes between a gene and multiple other genes (Youseph *et al.*, 2015, 2019; Krishnan *et al.*, 2020b). It also incorporates the Hill coefficient that represents the effect of binding affinity between genes (Alon, 2019). We represent the GRN model by a dynamics state space formulation for N genes X_i, $i=1,\ldots,N$ as

$$\mathbf{x}_k = \boldsymbol{\phi}(\mathbf{x}_{k-1}) + \mathbf{w}_{k-1} \tag{8.1}$$

$$\mathbf{y}_k = \mathbf{x}_k + \mathbf{v}_k \tag{8.2}$$

where $\mathbf{x}_k = [x_{1,k} \ \ldots \ x_{N,k}]^T$, $x_{i,k}$, $i=1,\ldots,N$, is the expression level of Gene i at time step k, $\mathbf{y}_k = [y_{1,k} \ \ldots \ y_{N,k}]^T$, $y_{i,k}$ is the microarray measurement for Gene X_i, $\mathbf{w}_k$ is measurement noise, and $\mathbf{v}_k$ is a random process used to represent modeling error uncertainty. In Equation 8.1, the ith element of the $N\times 1$ transition vector $\boldsymbol{\phi}(\mathbf{x}_{k-1})$

is given by

$$[\boldsymbol{\phi}(\mathbf{x}_{k-1})]_i = [\boldsymbol{\rho}(\mathbf{x}_{k-1})]_i - \sigma(x_{i,k-1}) = [\boldsymbol{\psi}(\mathbf{x}_{k-1})]_i \prod_{\substack{j=1 \\ j\neq i}}^{N} \mathbf{a}_{ij}\, \mathbf{b}_{ij}(\mathbf{x}_{k-1}) - v_i^d\, x_{i,k-1} \quad (8.3)$$

where the terms $[\boldsymbol{\rho}(\mathbf{x}_{k-1})]_i$ and $\sigma(x_{i,k-1})$ denote production and degradation, respectively, v_i^d is the self-decay rate of protein or mRNA expressed by Gene X_i,

$$[\boldsymbol{\psi}(\mathbf{x}_{k-1})]_i = \frac{v_i^{\max}}{(K_{ii}^{q_{ii}} + x_{i,k-1}^{q_{ii}}) \prod_{\substack{j=1 \\ j\neq i}}^{N} (K_{ij}^{q_{ij}} + x_{j,k-1}^{q_{ij}})} \quad (8.4)$$

$$\mathbf{a}_{ij} = \begin{bmatrix} g_{ii}\, g_{ij} & g_{ii}\, h_{ij} & h_{ii}\, g_{ij} & h_{ii}\, h_{ij} \end{bmatrix} \quad (8.5)$$

$$\mathbf{b}_{ij}(\mathbf{x}_{k-1}) = \begin{bmatrix} x_{i,k-1}^{q_{ii}}\, x_{j,k-1}^{q_{ij}} & x_{i,k-1}^{q_{ii}}\, K_{ij}^{q_{ij}} & x_{j,k-1}^{q_{12}}\, K_{ii}^{q_{ii}} & K_{ii}^{q_{ii}}\, K_{ij}^{q_{ij}} \end{bmatrix}^T, \quad (8.6)$$

and $v_i^{\max}$ is the maximum expression rate of Gene X_i.

The terms h_{ij} and g_{ij} are the kinetic order parameters that indicate the type of regulation. In particular, when $h_{ij} = 1$ and $g_{ij} = 1$, $i \neq j$, then there is no regulation from gene j to gene i. When $h_{ij} = 0$ and $g_{ij} = 1$, then gene j activates gene i, whereas $h_{ij} = 1, g_{ij} = 0$ implies that gene j inhibits gene i. The parameter K_{ij} is the Michaelis-Menten constant associated with the regulation of gene i by gene j. In the context of gene regulation, the Michaelis-Menten constant K_{ij} accounts for the binding affinity between the product of a gene j and the binding site of a target gene i. Large values of K_{ij} indicate a low binding affinity and low values of K_{ij} indicate a high binding affinity. The term v_i^d is the self decay rate of mRNA expressed by gene i, and q_{ij} is the Hill coefficient. The parameters K_{ii}, q_{ii}, h_{ii} and g_{ii} specify autoregulatory interactions. These are summarized in Table 8.1.

Parameters	Description
$x_{i,k}$	Expression level of Gene i at time step k
	$\mathbf{x}_k = [x_{1,k}\ x_{2,k}\ \ldots\ x_{N,k}]^T$
K_{ij}	Michaelis-Menten constant associated with the regulation of
	Gene i by Gene j
q_{ij}	Hill coefficient
$v_i^{\max}$	Maximum expression rate of the ith gene
v_i^d	Self-decay rate of protein or mRNA expressed by Gene i
h_{ij}, g_{ij}	Kinetic order parameters indicating regulation type
$\mathbf{w}_k$	Modeling error random variable
$y_{i,k}$	Microarray measurement relating to the ith gene at time step k
	$\mathbf{y}_t = [y_{1,k}\ x_{2,k}\ \ldots\ y_{N,k}]^T$
$\mathbf{v}_k$	Measurement noise

Table 8.1: GRN state-space model parameters

8.2.2 Formulation of Time-Varying GRN Model

We consider a TV GRN model where both the direction and type of regulation vary with time. This time variation is reflected in the kinetic order parameters in Equation 8.5.

$$\mathbf{a}_{ij,k} = [g_{ii,k}\ g_{ij,k} \quad g_{ii,k}\ h_{ij,k} \quad h_{ii,k}\ g_{ij,k} \quad h_{ii,k}\ h_{ij,k}]\,, \tag{8.7}$$

and thus in the transition function $\boldsymbol{\phi}_k(\cdot)$ in Equations 8.1 and 8.2. With this TV model, we can describe interactions between any number of genes and capture any changes in the GRN configuration with varying degrees of complexity.

As a simple illustration, we consider the transition from time step $k-1$ to k of the

simple GRN in Figure 8.1 to the one in Figure 8.2. In Figure 8.1, Genes X_1 and X_2 activate each other's expression in a positive feedback loop at $k-1$. From Table 8.2, activation of Gene X_1 by Gene X_2 is indicated by $h_{12}=0$ and $g_{12}=1$ and activation of Gene X_2 by Gene X_1 is indicated by $h_{21}=0$ and $g_{21}=1$. Thus, the kinetic order parameter vector at time step $k-1$ is given by

$$\begin{aligned}\mathbf{a}_{12,k-1} = \mathbf{a}_{X_1 \leftarrow X_2,k-1} &= [g_{11,k-1}\, g_{12,k-1} \;\; g_{11,k-1}\, h_{12,k-1} \;\; h_{11,k-1}\, g_{12,k-1} \;\; h_{11,k-1}\, h_{12,k-1}] \\ &= [0\,1 \;\; 0\,0 \;\; 0\,1 \;\; 0\,0] = \mathbf{a}_{X_2 \leftarrow X_1,k-1} = \mathbf{a}_{21,k-1}\,.\end{aligned}$$

where notation $X_1 \leftarrow X_2$ denotes the interaction from Gene X_2 to Gene X_1. In Figure 8.2(b), the interaction from Gene X_2 to Gene X_1 switches from activation to inhibitory at time k. Whereas $\mathbf{a}_{21,k} = \mathbf{a}_{21,k-1}$, since inhibition of Gene X_1 by Gene X_2 is indicated by $h_{12}=1$ and $g_{12}=0$, then

$$\mathbf{a}_{X_1 \leftarrow X_2,k} = [g_{11,k}\, g_{12,k} \;\; g_{11,k}\, h_{12,k} \;\; h_{11,k}\, g_{12,k} \;\; h_{11,k}\, h_{12,k}] = [0\,0 \;\; 0\,1 \;\; 0\,0 \;\; 0\,1]$$

The effect of of dynamic changes in the kinetic order parameters in the state transition function $\boldsymbol{\phi}(\mathbf{x}_{k-1})$ is given as follows:

$$\mathbf{a}_{ij,k} = [g_{ii,k}g_{ij,k},\;\; g_{ii,k}h_{ij,k},\;\; h_{ii,k}g_{ij,k},\;\; h_{ii,k}h_{ij,k}] \tag{8.8}$$

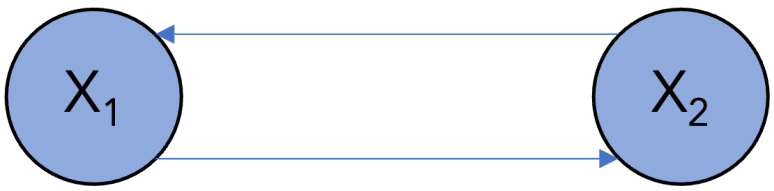


Figure 8.1: Gene X_1 and gene X_2 mutually activate each other's expression.

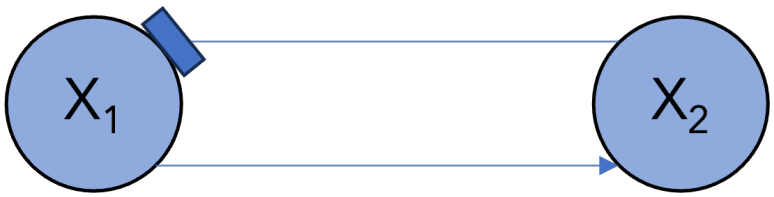


Figure 8.2: Gene X_1 activates the expression of gene X_2 whereas gene X_2 inhibits the expression of gene X_1

Parameter g_{ij}	Parameter h_{ij}	Regulation type from X_j to X_i
1	1	no regulation
1	0	activation
0	1	inhibition

Table 8.2: GRN kinetic order parameters indicating type of regulation from Gene X_j to Gene X_i, $i=j$; autoregulation is indicated by $g_{ii}=h_{ii}=1$.

8.2.3 Bayesian Learning for Tracking

In this paper, we propose a new method for estimating the TV gene expressions under the realistic conditions of sudden changes in the GRN architecture and unknown statistics in the state space formulation. In particular, we assume that the GRN circuit features on developmental function can be selected from M subcircuit models. The mth model at time step k, denoted by the indicator $s_k=m$, $m=1,\ldots,M$, is associated with the mth TV kinematic order parameter vector $\mathbf{a}_{ij,k}^{(m)}=\mathbf{a}_{ij,k}$ in Equation 8.8. The state space formulation for estimating $\mathbf{x}_k$ is now given by

$$\mathbf{x}_k = \boldsymbol{\phi}(\mathbf{x}_{k-1}, s_k = m) + \mathbf{w}_{k-1}^{(m)} \tag{8.9}$$

$$\mathbf{y}_k = \mathbf{x}_k + \mathbf{v}_k\,, \tag{8.10}$$

where the modeling error process $\mathbf{w}_{k-1}^{(m)} \sim \mathcal{N}(\mathbf{0}, \Sigma_{\mathbf{w}}^{(m)})$ for the mth model and the measurement noise $\mathbf{v}_k \sim \mathcal{N}(\mathbf{0}, \Sigma_{\mathbf{v}})$ are both assumed zero-mean Gaussian with constant but unknown covariance matrices $\Sigma_{\mathbf{w}}^{(m)}$ and $\Sigma_{\mathbf{v}}$, respectively.

The new method, Bayesian Learning for Tracking or BLT, sequentially estimates the gene expressions while learning the probability of selecting one of the M models at each time step k as well as the unknown covariance matrices. From Equation 8.9, the model indicator is drawn from a categorical distribution with parameter vector $\boldsymbol{\pi}_k = [\pi_k^{(1)} \ \ldots \ \pi_k^{(M)}]$; specifically, $s_k \mid \boldsymbol{\pi}_k, M \sim \text{Cat}(s_k \mid \boldsymbol{\pi}_k, M)$, where $\pi_k^{(m)} = \Pr(s_k = m)$ is the probability of selecting the mth model at time k and $\boldsymbol{\pi} \mid \boldsymbol{\alpha} \sim \text{Dir}(\boldsymbol{\alpha})$ is obtained from a Dirichlet distribution prior with concentration hyperparameter $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_M]$ (Sudderth, 2016). The prior probability that the existing model m is selected at time step k is

$$p(s_k = m \,|\, \mathcal{S}_{k-1}, \pi_k^{(m)}, \alpha_m) = \frac{c_{1:k-1}^{(m)}}{\sum_{l=1}^{m} c_{1:k-1}^{(l)} + \alpha_m} \tag{8.11}$$

where $\mathcal{S}_{k-1} = \{s_1, \ldots, s_{k-1}\}$ and $c_{1:k-1}^{(m)}$ is the number of times the mth model was selected up to the previous time step $k-1$. The covariance matrix $\Sigma_{\mathbf{w}}^{(m)} \,|\, \boldsymbol{\Psi}_{\mathbf{w}}^{(m)} \sim \text{IWD}(\boldsymbol{\Psi}_{\mathbf{w}}^{(m)})$ is modeled using an inverse Wishart distribution (IWD) prior with hyperparameter $\boldsymbol{\Psi}_{\mathbf{w}}^{(m)} = \{\Lambda_{\mathbf{w}}^{(m)}, \nu_{\mathbf{w}}^{(m)}\}$. Here, $\Lambda_{\mathbf{w}}^{(m)}$ is the scale matrix and $\nu_{\mathbf{w}}^{(m)}$ is the number of degrees of freedom (Gelman *et al.*, 1995b).

The overall learning model is summarized by the following hierarchy

$$\begin{aligned}
&\mathbf{x}_k \,|\, \mathbf{x}_{k-1}, s_k = m, \Sigma_{\mathbf{w}}^{(m)} \sim \mathcal{N}(\cdot \,|\, \mathbf{x}_{k-1}, s_k = m, \Sigma_{\mathbf{w}}^{(m)}) \\
&\mathbf{y}_k \,|\, \mathbf{x}_k, \Sigma_{\mathbf{v}} \sim \mathcal{N}(\cdot \,|\, \mathbf{x}_k, \Sigma_{\mathbf{v}}) \\
&s_k \,|\, \mathcal{S}_{k-1}, \pi_k^{(m)}, \alpha_m \sim \mathrm{Cat}(\boldsymbol{\pi}_k, M) \\
&\pi_k^{(m)} \,|\, \alpha_m \sim \mathrm{Dir}(\boldsymbol{\alpha}) \\
&\Sigma_{\mathbf{w}}^{(m)} \,|\, \boldsymbol{\Psi}_{\mathbf{w}}^{(m)} \sim \mathrm{IWD}(\boldsymbol{\Psi}_{\mathbf{w}}^{(m)}) \\
&\Sigma_{\mathbf{v}} \,|\, \boldsymbol{\Psi}_{\mathbf{v}} \sim \mathrm{IWD}(\boldsymbol{\Psi}_{\mathbf{v}}) \,.
\end{aligned}$$

Following a sequential Bayesian Monte Carlo approach, the model first predicts $\mathbf{x}_k$ using

$$\begin{aligned}
p\Big(\mathbf{x}_k \,|\, \mathbf{x}_{k-1}, s_k = m, \mathcal{S}_{k-1}, \pi_k^{(m)}, \alpha_m, \Sigma_{\mathbf{w}}^{(m)}, \boldsymbol{\Psi}_{\mathbf{w}}^{(m)}\Big) \quad \propto \\
p(\mathbf{x}_k \,|\, \mathbf{x}_{k-1}, s_k = m, \Sigma_{\mathbf{w}}^{(m)}) \; p(s_k = m \,|\, \mathcal{S}_{k-1}, \pi_k^{(m)}, \alpha_m) \; p(\pi_k^{(m)} \,|\, \alpha_m) \\
p(\Sigma_{\mathbf{w}}^{(m)} \,|\, \boldsymbol{\Psi}_{\mathbf{w}}^{(m)})
\end{aligned} \tag{8.12}$$

where $p(\mathbf{x}_k \,|\, \mathbf{x}_{k-1}, s_k = m, \Sigma_{\mathbf{w}}^{(m)})$ is obtained from Equation 8.9, $p(s_k = m \,|\, \mathcal{S}_{k-1}, \pi_k^{(m)}, \alpha_m)$ is the categorical distribution in Equation 8.11, $p(\pi_k^{(m)} \,|\, \alpha_m)$ is the Dirichlet prior, and $p(\Sigma_{\mathbf{w}}^{(m)} \,|\, \boldsymbol{\Psi}_{\mathbf{w}}^{(m)})$ is the IWD prior. Before using measurement $\mathbf{y}_k$ to update the state $\mathbf{x}_k$ at time step k, the noise covariance matrix $\Sigma_{\mathbf{v}} \sim \mathrm{IWD}(\boldsymbol{\Psi}_{\mathbf{v}})$ is learned using an IWD prior with hyperparameter $\boldsymbol{\Psi}_{\mathbf{v}} = \{\Lambda_{\mathbf{v}}, \nu_{\mathbf{v}}\}$. This results in

$$p(\mathbf{y}_k \,|\, \mathbf{x}_k, \Sigma_{\mathbf{v}}, \boldsymbol{\Psi}_{\mathbf{v}}) \;\propto\; p(\mathbf{y}_k \,|\, \mathbf{x}_k, \Sigma_{\mathbf{v}}) \; p(\Sigma_{\mathbf{v}} \,|\, \boldsymbol{\Psi}_{\mathbf{v}}) \tag{8.13}$$

where $p(\mathbf{y}_k \,|\, \mathbf{x}_k, \Sigma_{\mathbf{v}})$ is given in Equation 8.10. and $p(\Sigma_{\mathbf{v}} \,|\, \boldsymbol{\Psi}_{\mathbf{v}})$ is the IWD prior. Using Equations 8.12 and 8.13, the joint posterior distribution at each time step k is given

by

$$p\Big(\mathbf{x}_k, s_k = m, \pi_k^{(m)}, \Sigma_{\mathbf{w}}^{(m)}, \Sigma_{\mathbf{v}} \,|\, \mathbf{y}_k, \mathcal{S}_{k-1}, \alpha_m, \mathbf{\Psi}_{\mathbf{w}}^{(m)}, \mathbf{\Psi}_{\mathbf{v}}\Big) \quad \propto$$
$$p\Big(\mathbf{x}_k \,|\, \mathbf{x}_{k-1}, s_k = m, \mathcal{S}_{k-1}, \pi_k^{(m)}, \alpha_m, \Sigma_{\mathbf{w}}^{(m)}, \mathbf{\Psi}_{\mathbf{w}}^{(m)}\Big)\; p(\mathbf{y}_k \,|\, \mathbf{x}_k, \Sigma_{\mathbf{v}}, \mathbf{\Psi}_{\mathbf{v}})$$
$$p\Big(\mathbf{x}_{k-1}, s_{k-1}, \pi_{k-1}^{s_{k-1}}, \Sigma_{\mathbf{w}}^{(s_{k-1})}, \Sigma_{\mathbf{v}} \,|\, \mathbf{y}_{k-1}, \alpha_{s_{k-1}}, \mathbf{\Psi}_{\mathbf{w},\text{prev}}^{(s_{k-1})}, \mathbf{\Psi}_{\mathbf{v},\text{prev}}\Big). \qquad (8.14)$$

The last distribution in Equation 8.14 is the posterior from the previous time step, obtained using model s_{k-1}, where $\mathbf{\Psi}_{\mathbf{w},\text{prev}}^{(s_{k-1})}$ and $\mathbf{\Psi}_{\mathbf{v},\text{prev}}$ are the IWD hyperparameters used at time step $k-1$.

Note that the derivation of the BLT is provided in Appendix A.

8.2.4 BLT Implementation using Particle Filtering and Local Gibbs Sampling

The dynamic state space model in Equations 8.9 and 8.4 is highly nonlinear, so we implement the BLT using particle filtering (Arulampalam *et al.*, 2002b). As Equation 8.12 is not explicitly known, the PF also estimates both the unknown TV model indicator s_k and unknown constant covariance matrix $\Sigma_{\mathbf{w}}^{(s_k)}$. However, the PF performs poorly when used to estimate parameters that do not change with time (Chopin, 2002). One approach to improve the PF performance for constant parameters, MCMC can be used together with a rejuvenation test based on the Kullback–Leibler divergence measure (Lee and Chia, 2002; Li and Papandreou-Suppappola, 2006). For the BLT, we instead use local Gibbs sampling (Gelfand *et al.*, 1990; Gelman *et al.*, 1995b) at each time step k in order to sample from available multivariate conditional distributions and sequentially update the constant IWD hyperparameters of the covariance matrix.

The BLT implementation steps are summarized in Algorithm 3. The algorithm uses the joint prior in Equation 8.12 as the PF proposal distribution and assumes N_s state particles. At each time step and particle, we perform L Gibbs sampling

iterations. In what follows, $\mathbf{x}_{i,k}^{(\ell)}$ denotes the ith particle, $i=1,\ldots,N_s$, at the ℓth iteration, $\ell=0,\ldots L$, at time step k; similarly, $s_{i,k}^{(\ell)}$ denotes the model indicator, $\pi_{i,k}^{(m,\ell)}$ denotes the probability of the mth model, $m=1,\ldots,M$, $c_{i,1:k-1}^{(m,\ell)}$ denotes the number of times the mth model is selected up to time step $k-1$, and $\Sigma_{\mathbf{w},i}^{(m,\ell)}$ denotes the mth modeling error covariance matrix. At $\ell=0$, the iterations are initialized using the estimates obtained from the previous time step, $\mathbf{x}_{i,k}^{(0)}=\mathbf{x}_{i,k-1}$ and $\Sigma_{\mathbf{w},i}^{(m,0)}=\Sigma_{\mathbf{w},i}^{(s_{i,k-1})}$. At the ℓth iteration, $\ell=1,\ldots,L$, the algorithm (i) samples probability $\pi_{i,k}^{(m,\ell)}$, $m=1,\ldots,M$, from

$$\pi_{i,k}^{(m,\ell)}\,|\,\alpha_m \sim \mathrm{Dir}\Big(\alpha_1+c_{i,1:k-1}^{(1,\ell-1)},\ldots,\alpha_M+c_{i,1:k-1}^{(M,\ell-1)}\Big) \tag{8.15}$$

(ii) uses $\pi_{i,k}^{(m,\ell)}$ and samples model indicator $s_{i,k}^{(\ell)}$ from

$$p\Big(s_{i,k}^{(\ell)}=m\,|\,\mathcal{S}_{i,k-1}^{(\ell-1)},\ \pi_{i,k}^{(m,\ell)},\ \mathbf{x}_{i,k}^{(\ell-1)},\ \mathbf{y}_{k-1},\ \Sigma_{\mathbf{w},i}^{(m,\ell-1)},\Sigma_{\mathbf{v},\mathrm{prev}}\Big)$$

$$=p\Big(s_{i,k}^{(\ell)}=m\,|\,\mathcal{S}_{i,k-1}^{(\ell-1)},\pi_{i,k}^{(m,\ell)},\Big)p\Big(\mathbf{x}_{i,k}^{(\ell-1)}\,|\,\mathbf{x}_{i,k-1},s_{i,k}^{(\ell)}=m,\Sigma_{\mathbf{w},i}^{(m,\ell-1)}\Big)\ p(\mathbf{y}_{k-1}\,|\,\mathbf{x}_{i,k}^{(\ell-1)},\Sigma_{\mathbf{v},\mathrm{prev}}\Big) \tag{8.16}$$

where $\Sigma_{\mathbf{v},\mathrm{prev}}$ is the modeling error covariance from the previous time step.

(iii) samples state $\mathbf{x}_{i,k}^{(\ell)}$ from $p\Big(\mathbf{x}_{k,i}^{(\ell)}\,|\,\mathbf{x}_{i,k-1},s_{i,k}^{(\ell)}=m,\Sigma_{\mathbf{w},i}^{(m,\ell-1)}\Big)$ in Equation 8.9.

(iv) samples $\Sigma_{\mathbf{w},i}^{(m,\ell)}$ from $p\Big(\Sigma_{\mathbf{w},i}^{(m,\ell)}\,|\,\mathbf{x}_{i,k}^{(\ell)},s_{i,k}^{(\ell)}=m,\boldsymbol{\Psi}_{\mathbf{w},i}^{(m,\ell)}\Big)=\mathrm{IWD}(\boldsymbol{\Psi}_{\mathbf{w},i}^{(m,\ell)})$ using incrementally

updated IWD parameter set $\boldsymbol{\Psi}_{\mathbf{w},i}^{(m,\ell)}=\{\Lambda_{\mathbf{w},i}^{(m,\ell)},\nu_{\mathbf{w},i}^{(m,\ell)}\}$, where

$$\Lambda_{\mathbf{w},i}^{(m,\ell)}=\Lambda_{\mathbf{w},i}^{(m,\ell-1)}+(\mathbf{x}_{i,k}^{(\ell)})(\mathbf{x}_{i,k}^{(\ell)})^T,\qquad \nu_{\mathbf{w},i}^{(m,\ell)}=\nu_{\mathbf{w},i}^{(m,\ell-1)}+1\,. \tag{8.17}$$

At the end of the L iterations, the predicted state particle is given by $\mathbf{x}_{i,k}=\mathbf{x}_{i,k}^{(L)}$. The corresponding weight is computed using

$$\omega_{i,k}\ \propto\ \omega_{i,k-1}\,p(\mathbf{y}_k\,|\,\mathbf{x}_{i,k},\Sigma_{i,\mathbf{v}}) \tag{8.18}$$

where the Gaussian likelihood, using N measurements, is given as

$$p(\mathbf{y}_k\,|\,\mathbf{x}_{i,k},\Sigma_{\mathbf{v},i})=\frac{1}{(2\pi)^{N/2}\,|\Sigma_{\mathbf{v},i}|^{1/2}}\exp\Big(-\frac{1}{2}(\mathbf{y}_k-\mathbf{x}_{i,k})^T\,\Sigma_{\mathbf{v},i}^{-1}\,(\mathbf{y}_k-\mathbf{x}_{i,k})\Big)\,.$$

The covariance matrix $\Sigma_{\mathbf{v},i}$ is estimated using the IWD hyperparameters that are updated from $\mathbf{\Psi}_{\mathbf{v},\text{prev}} = \{\Lambda_{\mathbf{v},\text{prev}},\ \nu_{\mathbf{v},\text{prev}}\}$ using

$$\Sigma_{\mathbf{v},i} \,|\, \mathbf{y}_k, \mathbf{x}_{i,k} \sim \text{IWD}(\Lambda_{\mathbf{v},\text{prev}} + (\mathbf{y}_k - \mathbf{x}_{i,k})(\mathbf{y}_k - \mathbf{x}_{i,k})^T,\ \nu_{\mathbf{v},\text{prev}} + 1)\,. \tag{8.19}$$

Algorithm 3 Sequential Monte Carlo with Local Gibbs Sampling for Subcircuit Detection

Initialization at $t = 1$

for $i = 1, ..., N$ particles **do**

  Sample $\mathbf{x}_1^{(i)} \sim p(\mathbf{x}_1)$

  Sample $s_1^{(i)} \sim \text{Cat}(\boldsymbol{\pi}_1)$

  Sample $\{\Sigma_w^{m,(i)}\}_{m=1}^M \sim \text{IW}(\{\mathbf{\Psi}_w^{m,(i)}, \nu_w^{m,(i)}\}_{m=1}^M)$

  Sample $\Sigma_v^{(i)} \sim \text{IW}(\mathbf{\Psi}_v^{(i)}, \nu_v^{(i)})$

end for

Sequential Updates for $t \geq 2$

for $l = 1, ..., L$ Gibbs iterations **do**

  Sample a new model indicator $s_k^{(i)} \sim Q_1$

  Sample $\mathbf{x}_k^{(i)} \sim p(\mathbf{x}_k \mid \mathbf{x}_{k-1}^{(i)}, s_k^{(i)}, \Sigma_w^{s_k,(i)})$

  Sample $\Sigma_w^{s_k,(i)} \sim Q_2$

end for

Update measurement $\mathbf{y}_k^{(i)} \sim p(\mathbf{y}_k \mid \mathbf{x}_k^{(i)}, \Sigma_v^{(i)})$

Update hyperparameters and sample $\Sigma_v^{(i)} \sim \text{IW}(\mathbf{\Psi}_v^{(i)}, \nu_v^{(i)})$

Compute particle weights $w_k^{(i)}$

Normalize weights $w_k^{(i)} = \frac{w_k^{(i)}}{\sum_{i=1}^{N_s} w_k^{(i)}}$

Resample particles $\mathbf{x}_k^{(i)}, s_k^{(i)}, \Sigma_w^{s_k,(i)}, \Sigma_v^{(i)}$

8.3 Results and Discussion

8.3.1 Simulation Settings

We demonstrate the efficacy of our proposed Bayesian learning approach using $N = 5$ genes that transition between multiple GRN configurations over time. We consider $M = 5$ subcircuits, of varying degrees of complexity, as shown in Figure 8.1. Their configurations vary from a simple single-input module (SIM) configuration, which regulates several genes and is present across a range of systems, to more complex ones; these include compositions of different canonical subcircuits, such as feedforward and mutual repression loops. Switching between these specific network structures highlights the versatility of our model as it demonstrates the adaptive capacities of our Bayesian learning approach in identifying architectural changes. The configurations are described as follows.

M_1 SIM subcircuit (Figure 8.1(a)): Gene X_1 inhibits the expression of genes X_2, X_3, X_4, and X_5.

M_2 Subcircuit with feed forward loop and an inhibitory interaction (Figure 8.1(b)): Gene X_1 activates a cascade of activating regulatory interactions proliferating throughout the network; Gene X_5 activates Gene X_4, which in turn inhibits the expression of Gene X_5.

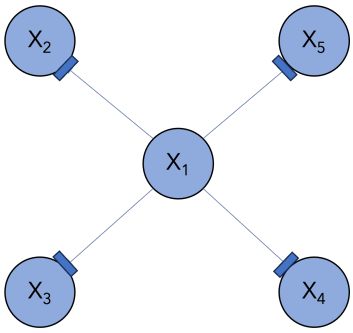
M_3 Complex subcircuit with negative autoinhibitory interactions on genes X_1 and X_2 and a feed forward loop (Figure 8.1(c)): X_2 activates the expression of X_4, which then activates the expression of X_3 and X_2; Gene X_5 activates the expression of Gene X_1, which inhibits the expression of X_3; and Gene X_3 inhibits the expression of X_1, forming a mutually repressive loop.

M_4 Complex subcircuit (Figure 8.1(d)): Gene X_1 is activated by genes X_2 and X_5; when X_3 is activated via an autoregulatory interaction, it inhibits Gene X_1 and activates genes X_4 and X_2; in turn, Gene X_2 activates X_1 and inhibits X_5, which activates the expression of X_1.

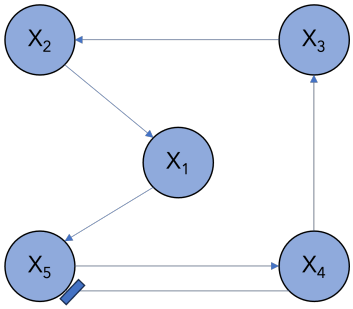
M_5 Complex subcircuit characterized by positive feedback loops (Figure 8.1(e)): each gene activates its own expression; Genes X_2 and X_5 form a positive feedback loop; Genes X_2 and X_3 inhibit X_4; Genes X_1 and X_3 activate X_5.

All configurations use the same constant model parameters provided in Table 8.3. The TV kinetic order parameters for each configuration are given in Table 8.4.

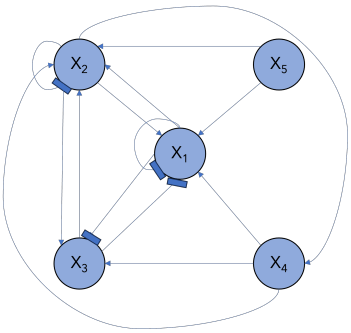
The BLT implementation using the PF and Gibbs sampling follows Algorithm 3. In the simulations, the modeling error process and measurement noise process are both assumed Gaussian with zero-mean and uncorrelated samples; their corresponding unknown covariance matrices are $\Sigma_{\mathbf{w}}^{(m)} = \text{var}_{\mathbf{w},m} I_N$ and $\Sigma_{\mathbf{v}} = \text{var}_{\mathbf{v}} I_N$. Here, $\text{var}_{\mathbf{w},m}$ and $\text{var}_{\mathbf{v}}$ denote their respective variance values and I_N is the $N \times N$ identity matrix. At each time step k, $k = 1, \ldots, K$, our learning approach concurrently estimates the unknown expression levels and the configuration model. We use the categorical distribution over M values whose vector parameter is learned using the Dirichlet distribution conjugate prior with hyperparameter $\boldsymbol{\alpha}$; in the simulations, we use $\alpha_m = 1$, $m = 1, \ldots, M$. We place IWD conjugate priors over the unknown covariance matrices with initial hyperparamers $\{\Psi_0, \nu_0\} = \{0.1 I_N,\ N+2\}$. We demonstrate the robustness of our approach across different sequences of gene regulation configurations. Unless otherwise specified, the simulations use 1,000 particles, 50,000 Monte Carlo runs and 10,000 Gibbs iterations.



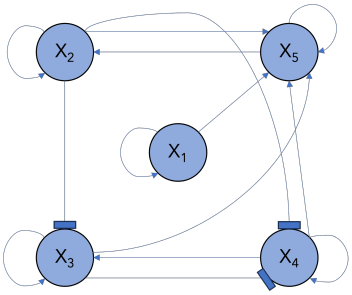
(a) A single-input module (SIM) subcircuit. Gene X_1 inhibits the expression of genes X_2, X_3, X_4, and X_5.



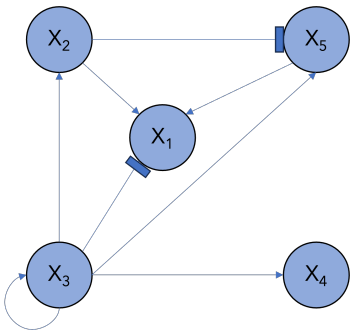
(b) A feed forward loop with an inhibitory interaction.



(c) A complex subcircuit architecture consisting of negative autoinhibitory interactions.



(e) A complex subcircuit architecture characterized by positive feedback loops. Each gene activates its own expression. Genes X_2 and X_5 form a positive feedback loop. Genes X_2 and X_3 inhibit X_4. Genes X_1 and X_3 activate X_5.



(d) A complex subcircuit architecture.

Figure 8.1: Visual depiction of each of the subcircuit models used in this work with varying degrees of complexity.

Simulation Results

We consider three different simulation scenarios with a varying number of configuration models during different time step segments, as summarized in Table **??**. In the table, $\mathcal{T}_{l,m}$ denotes the lth time segment during which the mth model is used. The description of each model is provided in Section **??**. We selected similar modeling error and measurement noise variances as in (Noor *et al.*, 2012). We compare the performance of our proposed BLT approach with the PF that performs tracking without any learning. The PF assumes that the GRN dynamics are only described by one of the M models in each scenario and uses 0.00003 for both the modeling error and measurement noise variances. These low variance values ensure improved estimation performance for the PF if the model is known, even though the actual variances are not learned.

Scenario 1: Three-model transition

The BLT considers an unknown number of TV transitions between $M = 3$ models: M_1 in Figure 8.1(a) during $\mathcal{T}_{1,1} = 1{:}65$, M_2 in Figure 8.1(b) during $\mathcal{T}_{2,2} = 66{:}165$, and M_3 in Figure 8.1(c) during $\mathcal{T}_{3,3} = 66 : 165$. The PF assumes the M_1 GRN configuration over all time steps. We simulate five different measurement noise variance values, $\text{var}_{\mathbf{v}} = \{2, 0.2, 0.03, 0.003, 0.00003\}$, and we use $\text{var}_{\mathbf{w},1} = 0.00002$, $\text{var}_{\mathbf{w},2} = 0.0005$ and $\text{var}_{\mathbf{w},3} = 0.00004$ for the actual values of the modeling error variances. The resulting root mean-square error (RMSE), averaged over all time steps, for estimating the gene expression levels using the BLT is provided in Table 8.5. As expected, the RMSE decreases as $\text{var}_{\mathbf{v}}$ increases. Using $\text{var}_{\mathbf{v}} = 0.003$, Figure 8.2 and Figure 8.3 further demonstrate the performance of the proposed BLT approach. Figure 8.2 compares the true expression levels of each of the 5 genes to those estimated by the BLT and PF methods. As demonstrated, the PF performed well during the first time segment as

it assumed the correct model M_1. However, the estimation performance deteriorated when the network architecture changed. In contrast, the BLT accurately estimated the gene expressions by learning the model transition probabilities and unknown noise variances. Figure 8.3 compares the actual model labels with those learned by the BLT. Note that after only two initial incorrect estimates, the BLT learned the correct model at each change in model transition. For the same modeling error variances, Figure 8.4 compares the true and estimated model labels when the measurement noise variance is $\text{var}_{\mathbf{v}} = 0.2$. As it can be seen from Figure 8.4 and Figure 8.5, as the noise in the measurements increases, the number of incorrect labels also increases. In Figure 8.6 and Figure 8.7, we increased the modeling error variances to $\text{var}_{\mathbf{w},1} = 0.02$, $\text{var}_{\mathbf{w},2} = 0.004$ and $\text{var}_{\mathbf{w},3} = 0.05$, and $\text{var}_{\mathbf{w},1} = 0.2$, $\text{var}_{\mathbf{w},2} = 0.05$ and $\text{var}_{\mathbf{w},3} = 0.4$ and kept the measurement noise variance to $\text{var}_{\mathbf{v}} = 0.003$. We observed that the model estimation accuracy is more sensitive to increased modeling error variances.

Next, for time steps $k = 1 : 65$, the dynamics are described by model 2; for time steps $k = 66 : 165$, the dynamics are described by model 1; and for time steps $k = 166 : 270$, the dynamics are described by model 3. The tracking results for $\Sigma_v = 3e^{-2}\boldsymbol{I}_N$ are shown in Figure 8.8. The true versus estimated model is shown in Figure 8.9. Again, after a few initial incorrect estimates, the algorithm estimates the true model at each of the change points. The RMSE across different measurement noise intensity values is shown in Table 8.7. The RMSE across different modeling error intensity values is shown in Table 8.8. Note that changing the time segments of the different models produced similar RMSE results, thus demonstrating the robustness of our Bayesian learning algorithm for the three-model case.

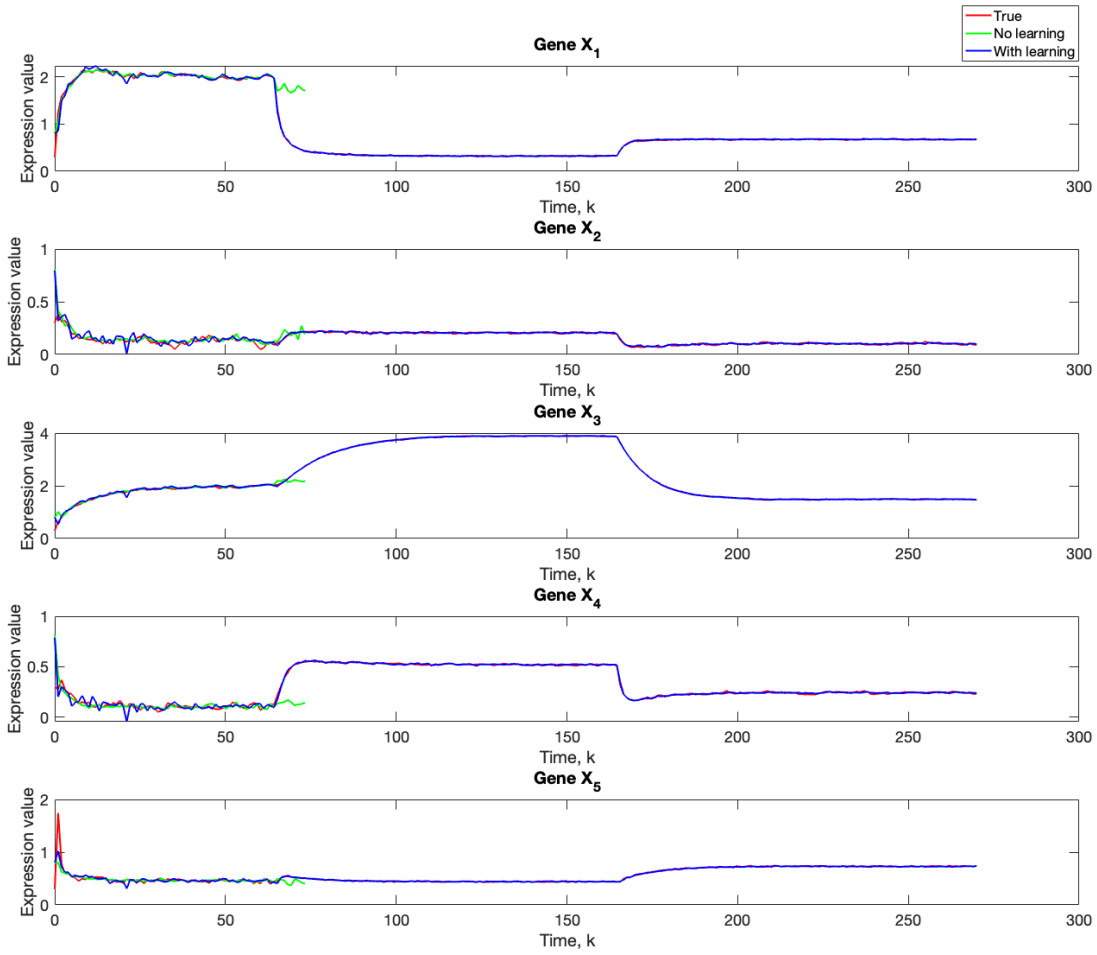


Figure 8.2: Scenario 1: Comparison of true, PF estimated and BLT estimated expression level $x_{i,k}$ of Gene X_i, $i=1,\ldots,5$ using modeling error variances $\text{var}_{\mathbf{w},1}=0.00002$, $\text{var}_{\mathbf{w},2}=0.0005$, $\text{var}_{\mathbf{w},3}=0.00004$ and measurement noise variance $\text{var}_{\mathbf{v}}=0.003$.

Scenario 2: Transitions Among Four Models

Scenario 2: Four-model transition

In this scenario, the BLT considers transitions between $M=4$ models: M_3 in Figure 8.1(c) during $\mathcal{T}_{1,3}=1:65$, M_2 in Figure 8.1(b) during $\mathcal{T}_{2,2}=66:165$, M_1 in Figure 8.1(c) during $\mathcal{T}_{3,1}=166:265$, and M_4 in Figure 8.1(d) during $\mathcal{T}_{4,4}=266:370$. The PF assumes that model M_3 is used for all time steps. We simulate five different

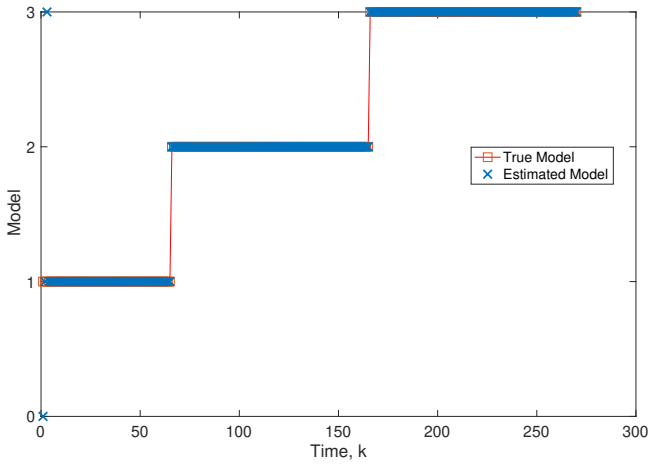


Figure 8.3: Comparison of true and BLT estimated labels of the configuration models in Scenario 1.

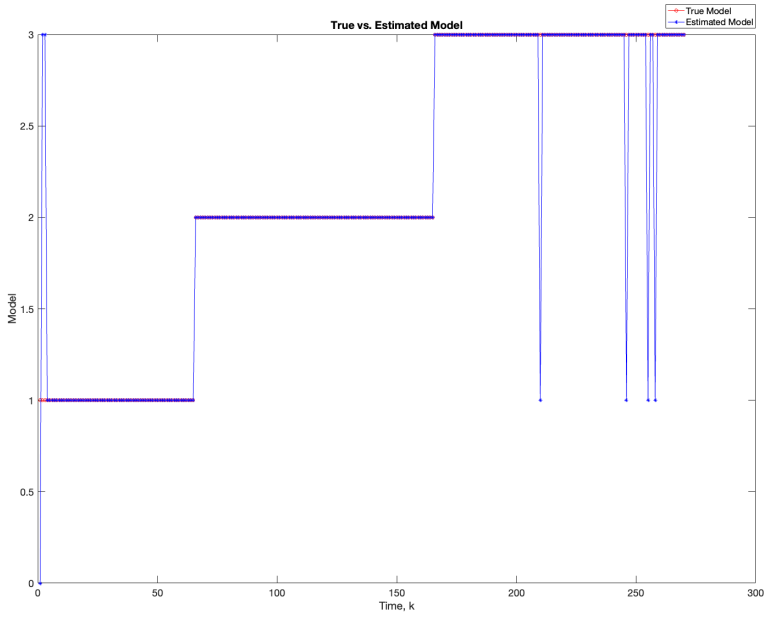


Figure 8.4: Comparison of true and BLT estimated labels of the configuration models in Scenario 1. The measurement noise intensity is 0.2.

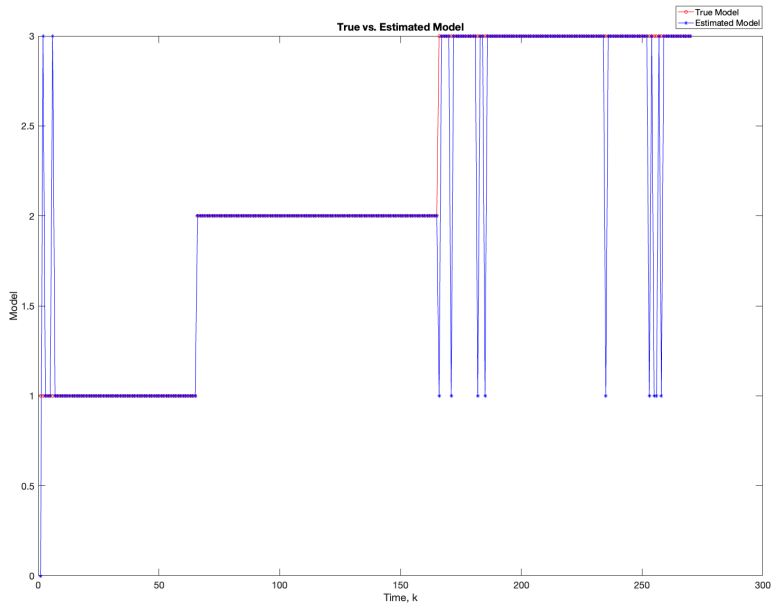


Figure 8.5: Comparison of true and BLT estimated labels of the configuration models in Scenario 1. The measurement noise intensity is 2.0.

measurement noise variance values, $\text{var}_{\mathbf{v}} = \{2, 0.2, 0.03, 0.003, 0.00003\}$, and we use $\text{var}_{\mathbf{w},1} = 0.00002$, $\text{var}_{\mathbf{w},2} = 0.0005$, $\text{var}_{\mathbf{w},3} = 0.00004$ and $\text{var}_{\mathbf{w},4} = 0.0002$ for the actual values of the modeling error variances. The gene expression averaged RMSE using the BLT are provided in Table 8.9 and Table 8.10. As expected, the RMSE increases as $\text{var}_{\mathbf{v}}$ increases.

A comparison of our learning algorithm to the standard PF (no learning) is shown in Figure 8.10 for $\text{var}_{\mathbf{v}} = 0.003$. The standard PF (no learning method) assumes that the dynamics are only described by M_3. As in scenario 1, the standard particle filter (no learning) can only provide estimates of the gene expressions whose trajectories are described by M_3. until the network architecture switches. then it is no longer able to estimate the trajectory. In contrast, the BLT is able to estimate the trajectory at each of the change points. The true versus estimated model is shown in Figure 8.11,

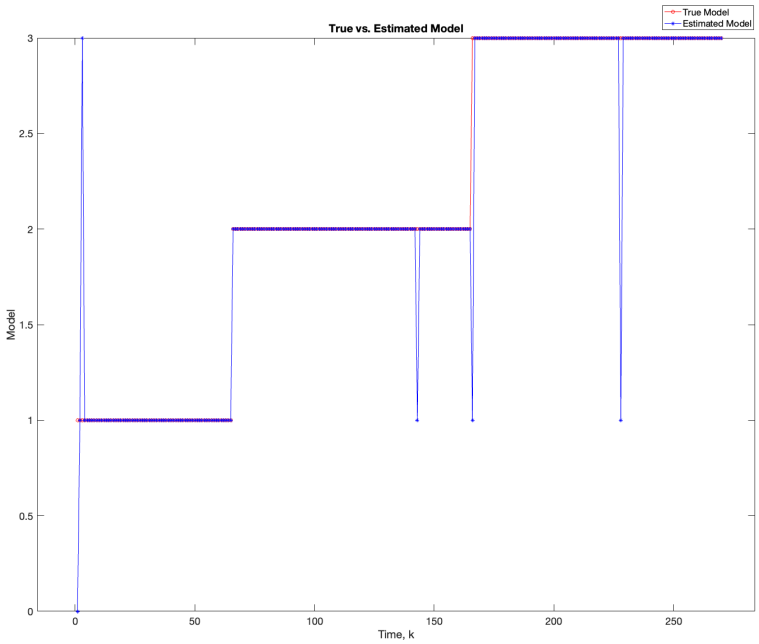


Figure 8.6: Comparison of true and BLT estimated labels of the configuration models in Scenario 1. The process noise intensity for segment 1 is 0.02, for segment 2 is 0.004, and for segment 3 is 0.05. The measurement noise intensity is 0.003.

which shows that the BLT effectively estimates the correct model. The RMSE across different measurement noise intensity values is shown in Table 8.9. We also vary the modeling error variance and the RMSE values are shown in Table 8.10. Figure 8.13 further demonstrates the performance of the proposed BLT approach under high measurement noise conditions. Figure 8.14 demonstrates the BLT performance under increased modeling error variance.

Scenario 3: Transitions Among Five Models

Scenario 3: Transitions Among Five Models

In this scenario, the BLT considers transitions between $M = 5$ models: M_3 in Figure 8.1(c) during $\mathcal{T}_{1,5} = 1 : 65$, M_2 in Figure 8.1(b) during $\mathcal{T}_{2,3} = 66 : 165$, M_1

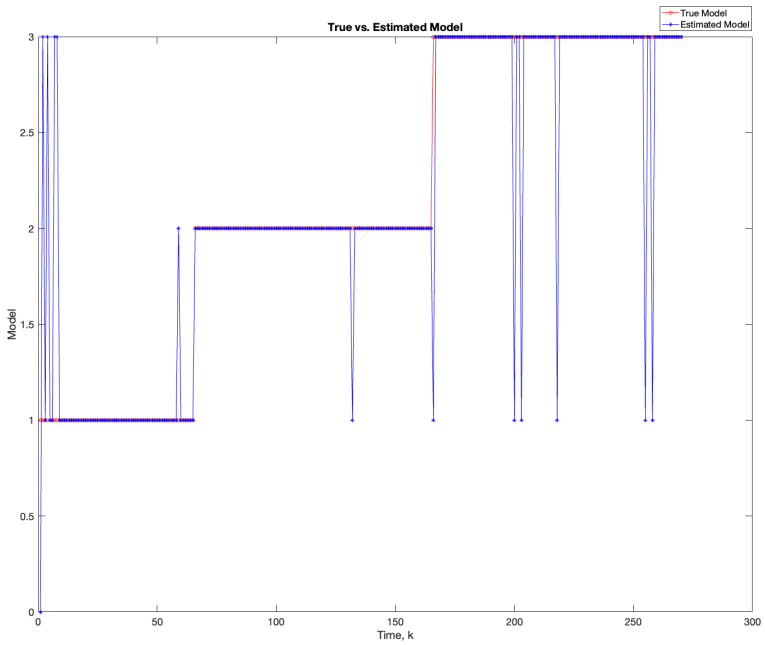


Figure 8.7: Comparison of true and BLT estimated labels of the configuration models in Scenario 1. The process noise intensity for segment 1 is 0.2, for segment 2 is 0.05, and for segment 3 is 0.4. The measurement noise intensity is 0.003.

in Figure 8.1(c) during $\mathcal{T}_{3,1} = 166 : 265$, M_4 in Figure 8.1(d) during $\mathcal{T}_{4,4} = 266 : 360$. and M_5 in Figure 8.1(e) during $\mathcal{T}_{5,2} = 361 : 460$. The time series consists of $K = 460$ time steps. We simulate five different measurement noise variance values, $\text{var}_{\mathbf{v}} = \{2, 0.2, 0.03, 0.003, 0.00003\}$, For each simulation where the measurement noise is varied, the true values of $\text{var}_{\mathbf{w},1} = 0.00002$, $\text{var}_{\mathbf{w},2} = 0.0005$, $\text{var}_{\mathbf{w},3} = 0.00004$, $\text{var}_{\mathbf{w},4} = 0.0002$, $\text{var}_{\mathbf{w},5} = 0.003$.

Using $\text{var}_{\mathbf{v}} = 0.02$, Figure 8.18 and Figure 8.19 further demonstrate the performance of the proposed BLT approach. Figure 8.18 compares the true expression levels of each of the 5 genes to those estimated by the BLT and PF methods. As in the previous scenarios, the standard particle filter (no learning) is only able to estimate the trajectory during the third segment where model M_1 is assumed. Then

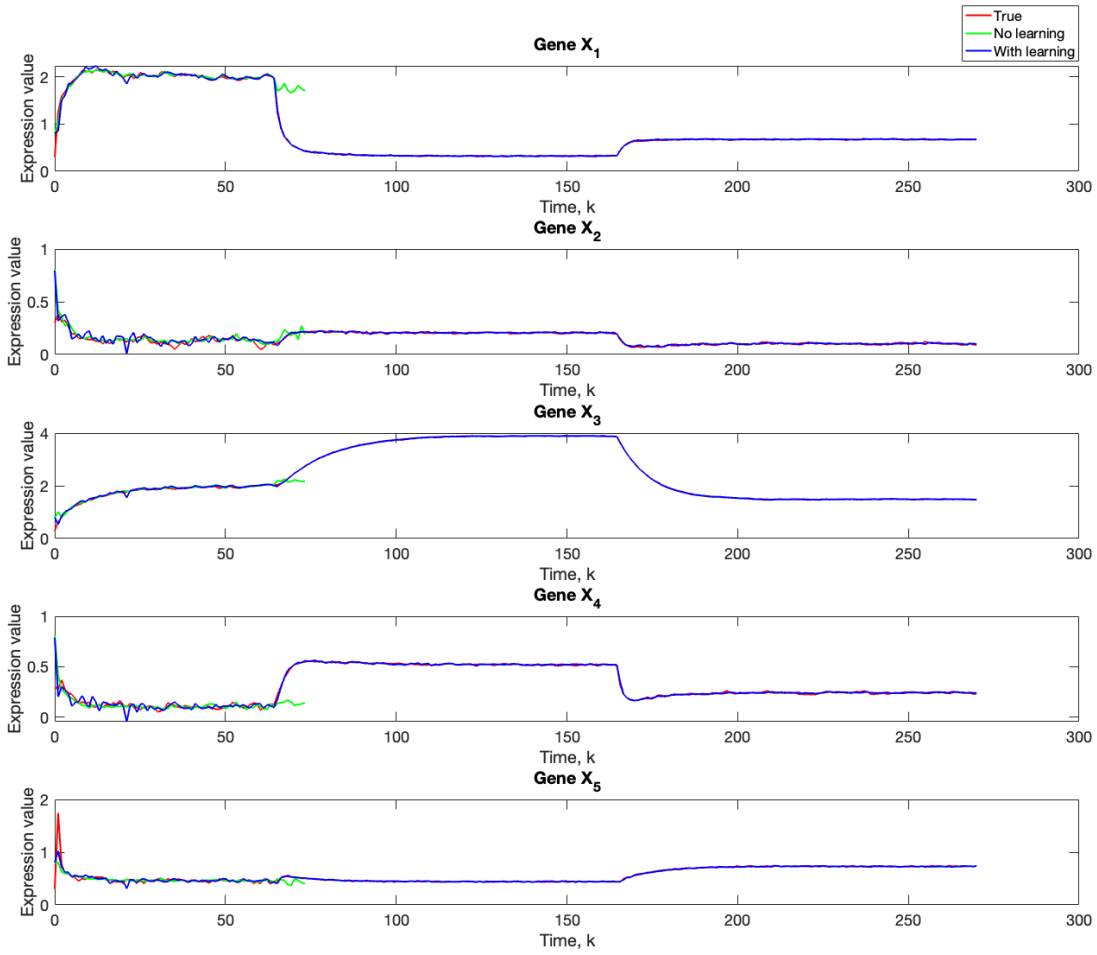


Figure 8.8: A comparison of our learning algorithm (blue) to the standard particle filter (green) for $\Sigma_v = 3e^{-2}\boldsymbol{I}_N$. The true trajectory is given by the red. We used 1,000 particles, 5,000 Monte Carlo runs, and 10,000 Gibbs iterations. For time steps $k = 1 : 65$, the dynamics are described by model 2; for time steps $k = 66 : 165$, the dynamics are described by model 1; and for time steps $k = 166 : 270$, the dynamics are described by model 3.

it is no longer able to estimate the trajectory. In contrast, the BLT method is able to estimate the trajectory at each of the change points. Figure 8.19 compares the actual model labels with those learned by the BLT. The RMSE across different measure-

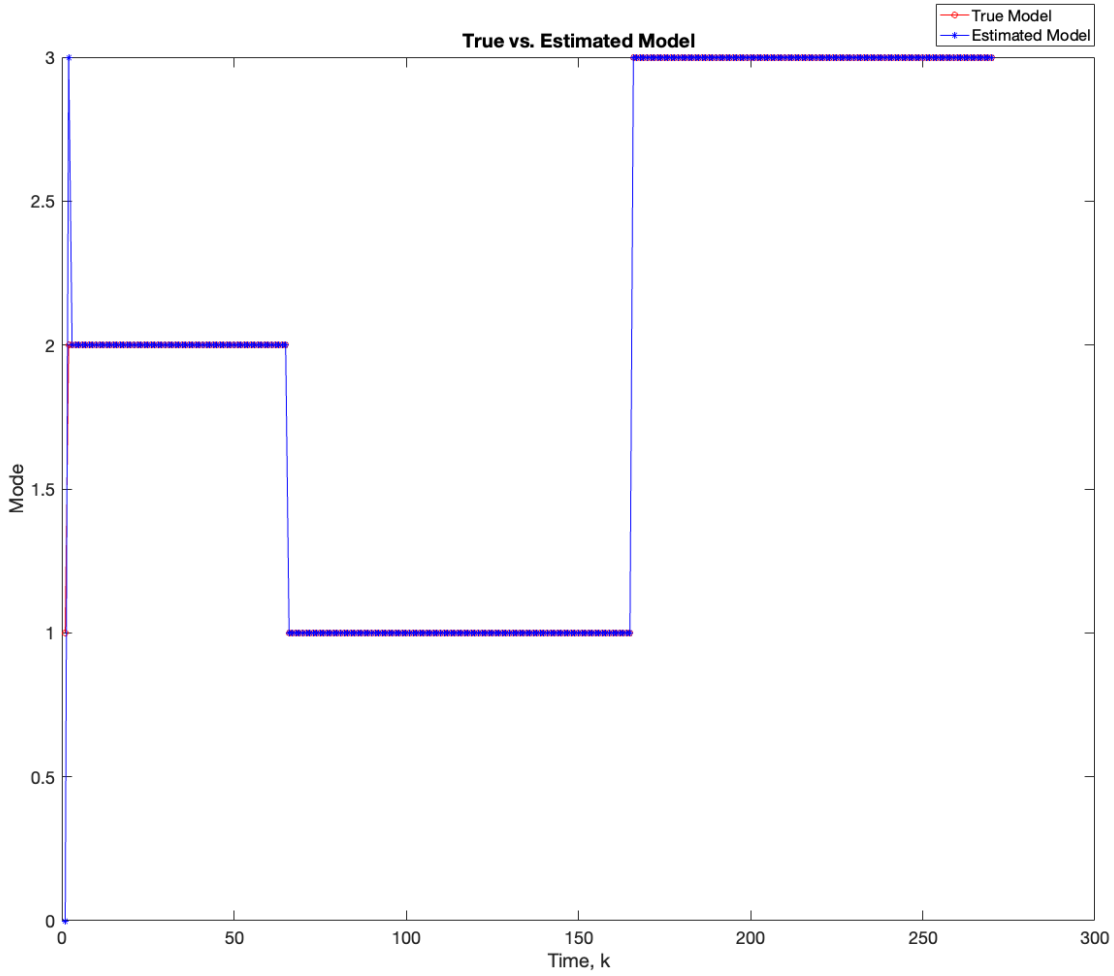


Figure 8.9: The true (red) versus estimated (blue) model for $\Sigma_v = 3e^{-2}\boldsymbol{I}_N$. We used 1,000 particles, 5,000 Monte Carlo runs, and 10,000 Gibbs iterations. For time steps $k = 1 : 65$, the dynamics are described by model 2; for time steps $k = 66 : 165$, the dynamics are described by model 1; and for time steps $k = 166 : 270$, the dynamics are described by model 3.

ment noise intensity values is shown in Table 8.11. The true versus estimated model for different values of the measurement noise intensity is shown in Figure 8.20 and Figure 8.21. The RMSE across different process noise intensity values is shown in Table 8.12. The true versus estimated model for different values of the measurement

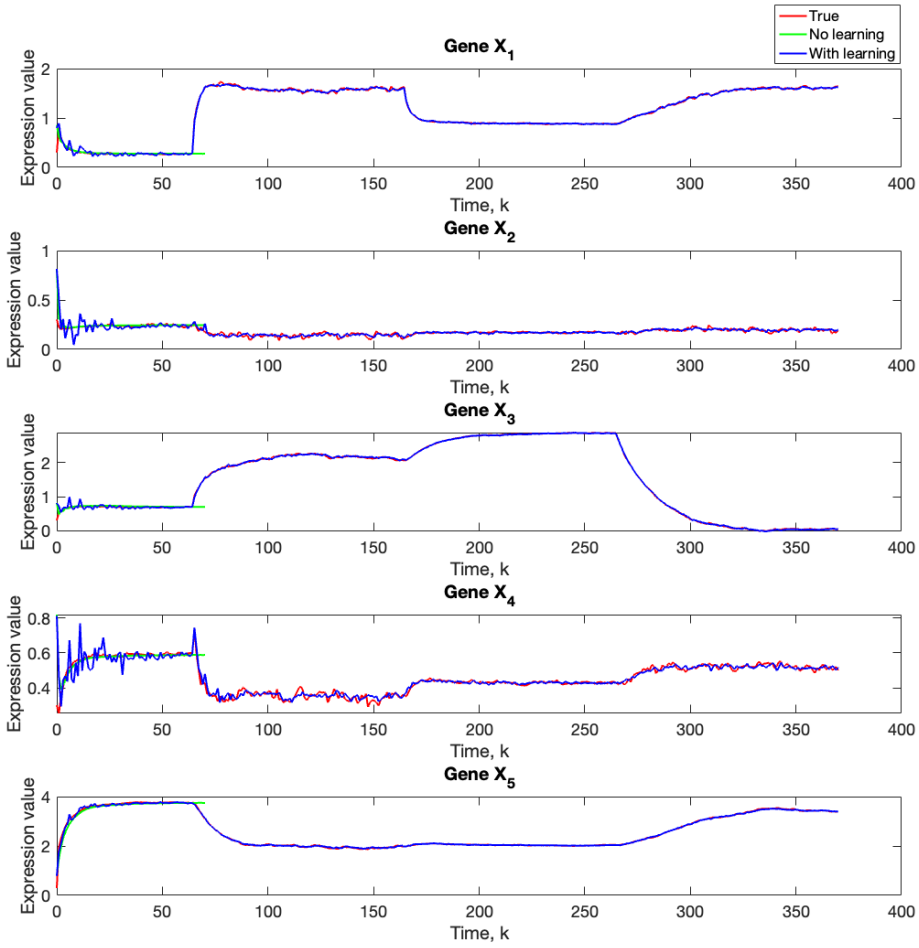


Figure 8.10: Comparison of true, PF estimated and BLT estimated expression level $x_{i,k}$ of Gene X_i, $i = 1, \ldots, 5$ in Scenario 2. The TV model configuration is provided in Table 8.4.

noise intensity is shown in Figure 8.22 and Figure 8.23. Note that as in the previous cases, the model estimation accuracy decreases as the measurement noise increases to 0.2 and above. As well, the estimation accuracy decreases as the process noise intensity increases.

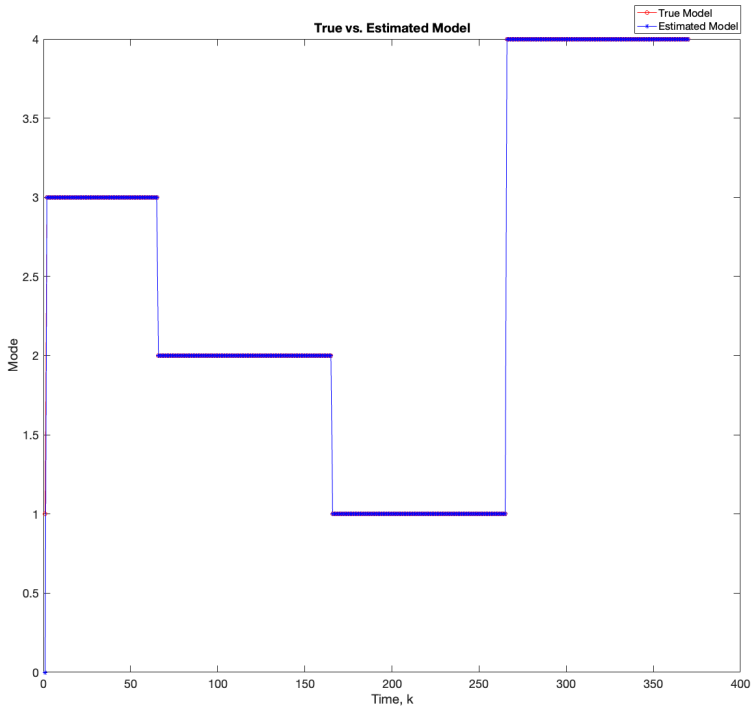


Figure 8.11: Comparison of true and BLT estimated labels of the configuration models in Scenario 2.

8.4 Conclusion

In this work, we introduced a fully Bayesian hierarchical model for learning nonlinear gene regulatory networks with dynamically switching subcircuit architectures. We showed that our algorithm, which employs sequential Monte Carlo (SMC) with a local Gibbs step, effectively estimates the correct model corresponding to the subcircuit architecture as well as the unknown state under varying measurement and process noise conditions with a high degree of accuracy, though with greater sensitivity to variations in the process noise. Through the use of conjugate priors, we have formulated an analytically tractable inference scheme which effectively addresses the challenges posed by the inherent complexities of the system. By learning the unknown

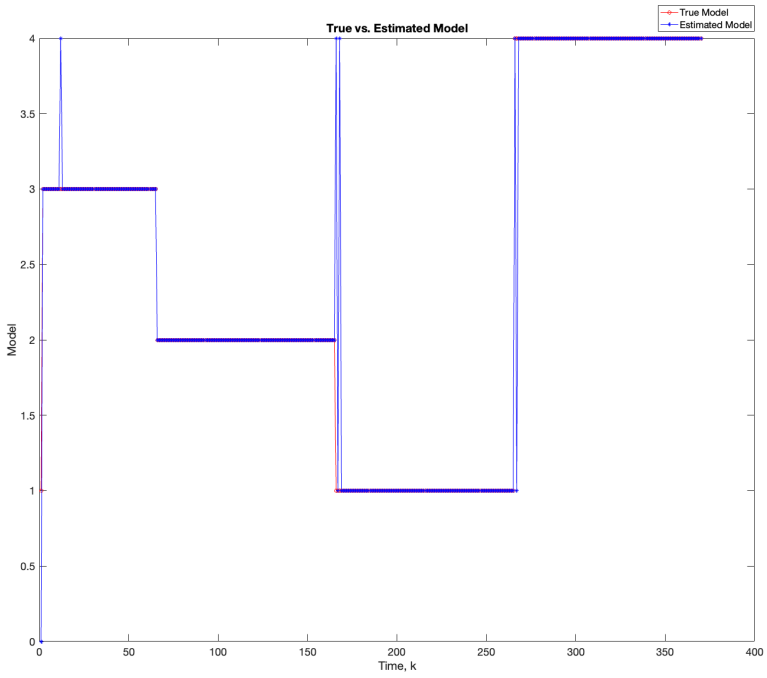


Figure 8.12: Comparison of true and BLT estimated labels of the configuration models in Scenario 2. The measurement noise intensity is 0.2.

transition probabilities, which describe changes between different network configurations and by learning the unknown measurement and process noise covariances using Bayesian updating of the Inverse-Wishart distribution, we were able to account for uncertainty at all levels of the hierarchy. Our approach has demonstrated robustness and versatility in identifying changes in subcircuit architectures of varying degrees of complexity, ranging from a simple single-input-module (SIM) to subcircuits consisting of a composition of different types. To showcase this robustness, we applied our algorithm to different scenarios where the architecture switches between three, four, and five types.

It is worth noting that the methodology developed in this work is not confined to the analysis of gene regulatory networks but can also be generalized to other domains

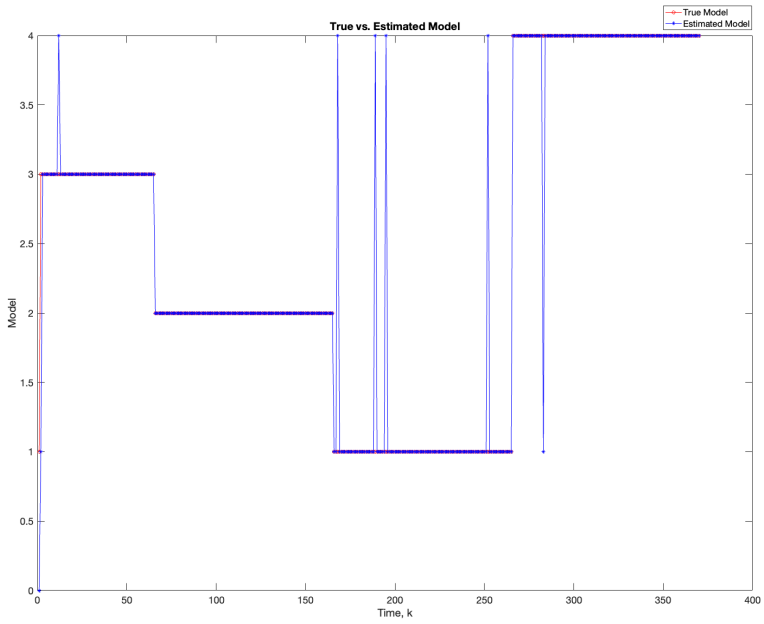


Figure 8.13: Comparison of true and BLT estimated labels of the configuration models in Scenario 2. The measurement noise intensity is 2.0.

which involve switching dynamics in nonlinear systems. For example, our methodology can be applied to change point detection problems in financial markets, where a sudden shift in the market dynamics could be caused by changes in nonlinear interactions between different economic factors. Future research will focus on extending our methodology to larger systems as well as applying it to a broader range of problems in nonlinear system analysis.

8.4.1 Tables

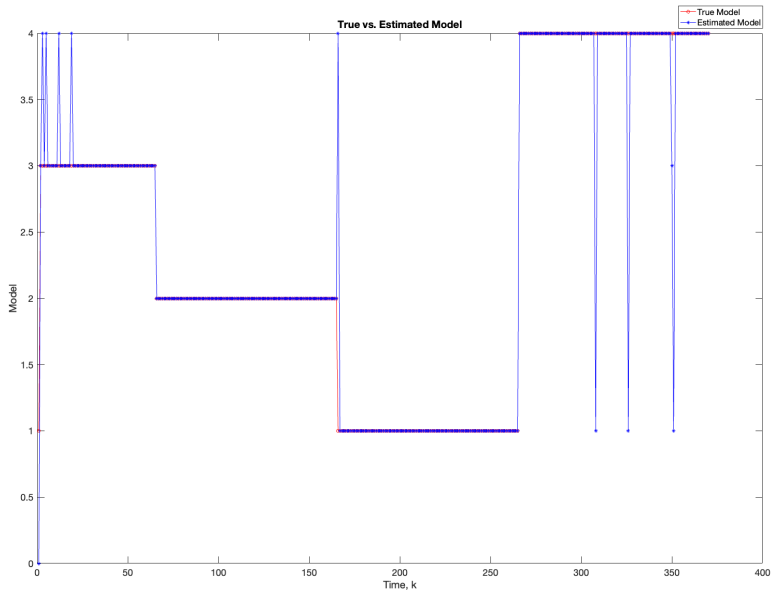


Figure 8.14: Comparison of true and BLT estimated labels of the configuration models in Scenario 1. The process noise intensity for segment 1 is 0.02, for segment 2 is 0.004, for segment 3 is 0.05, and for segment 4 is 0.002. The measurement noise intensity is 0.003.

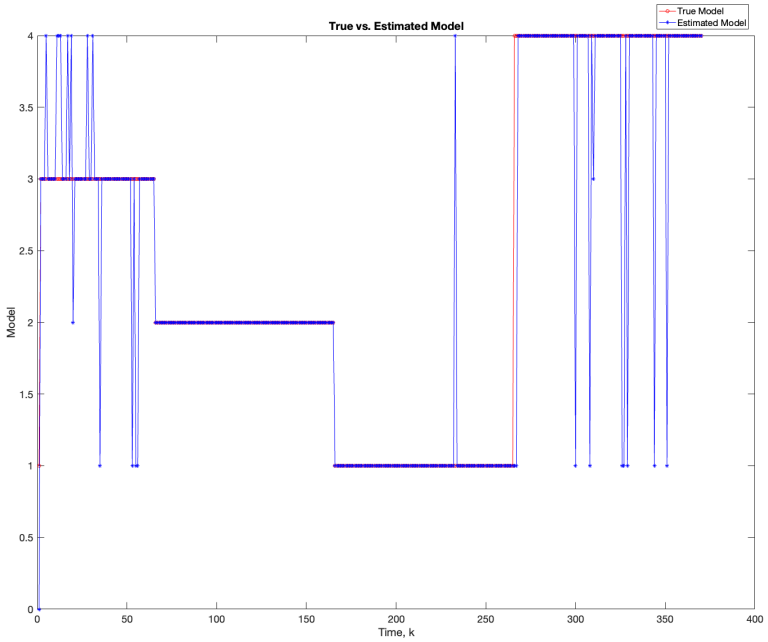


Figure 8.15: Comparison of true and BLT estimated labels of the configuration models in Scenario 1. The process noise intensity for segment 1 is 0.2, for segment 2 is 0.05, for segment 3 is 0.4, and for segment 4 is 0.02. The measurement noise intensity is 0.003.

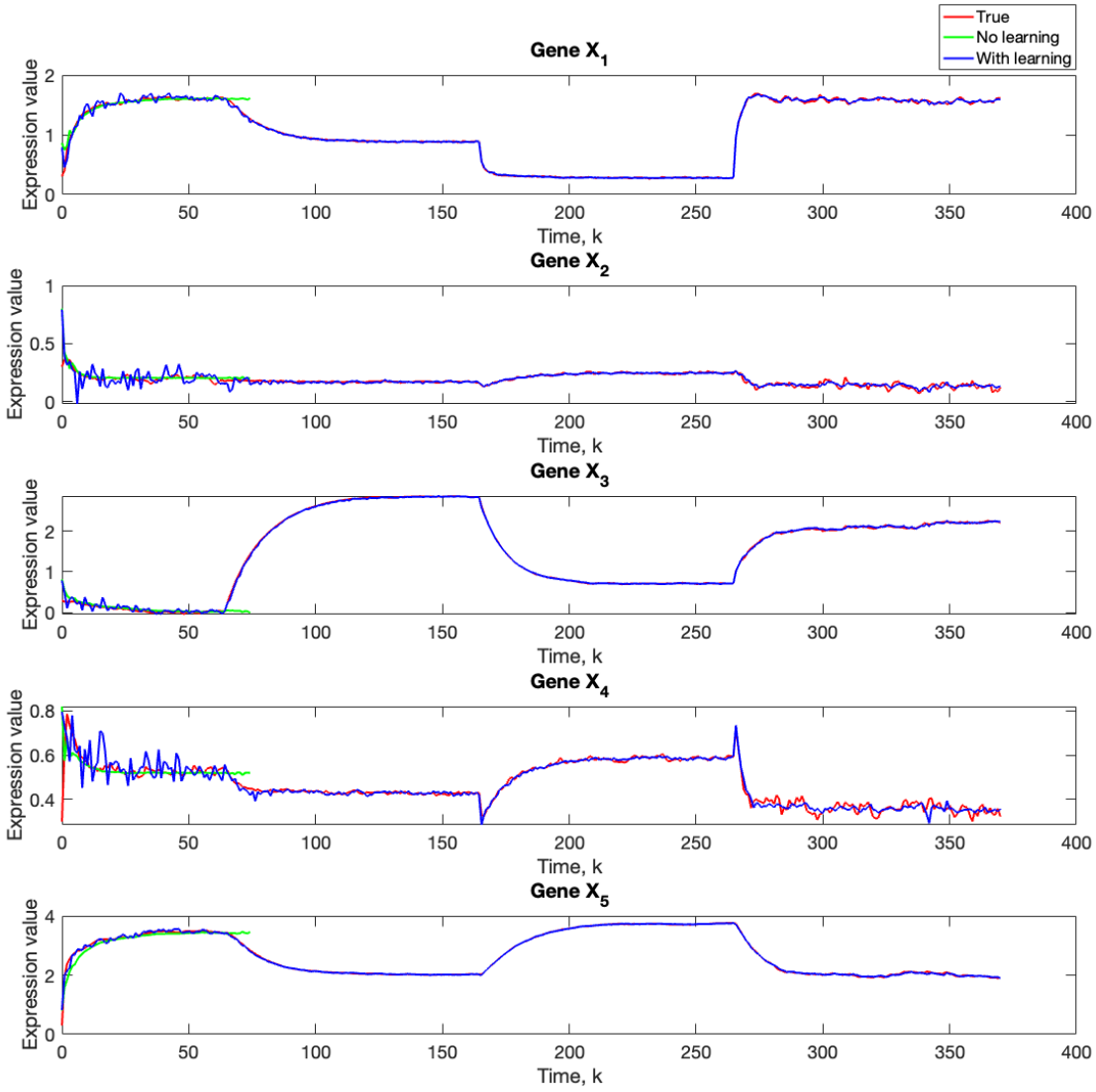


Figure 8.16: A comparison of our learning algorithm (blue) to the standard particle (green) for $\Sigma_v = 3e^{-5}\boldsymbol{I}_N$. The true trajectory is given by the red. We used 1,000 particles, 5,000 Monte Carlo runs, and 10,000 Gibbs iterations. For time steps $k = 1 : 65$, the dynamics are described by model 4; for time steps $k = 66 : 165$, the dynamics are described by model 1; for time steps $k = 166 : 265$, the dynamics are described by model 3; and for $k = 265 : 370$, the dynamics are described by model 2.

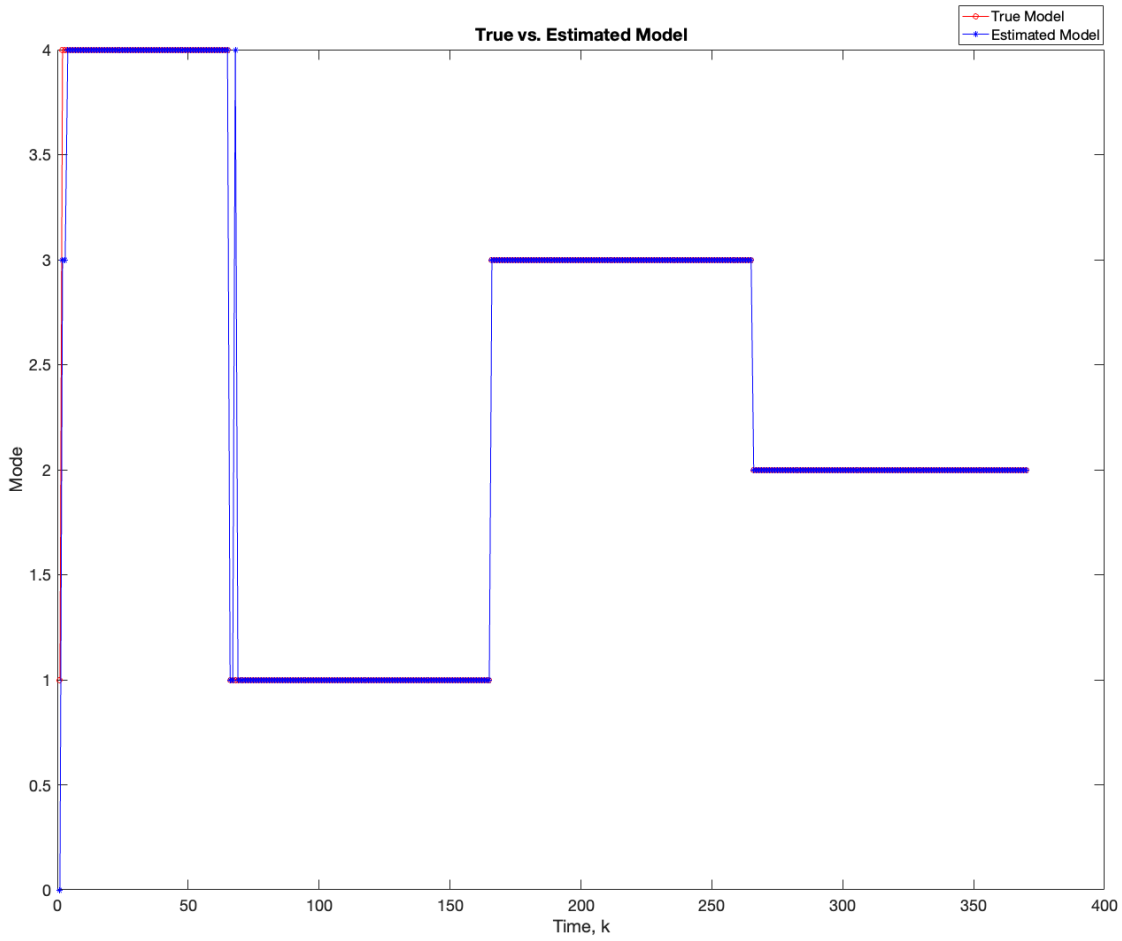


Figure 8.17: The true (red) versus estimated (blue) model for $\Sigma_v = 3e^{-5}\boldsymbol{I}_N$. We used 1,000 particles, 5,000 Monte Carlo runs, and 10,000 Gibbs iterations. For time steps $k = 1 : 65$, the dynamics are described by model 4; for time steps $k = 66 : 165$, the dynamics are described by model 1; for time steps $k = 166 : 265$, the dynamics are described by model 3; and for $k = 265 : 370$, the dynamics are described by model 2.

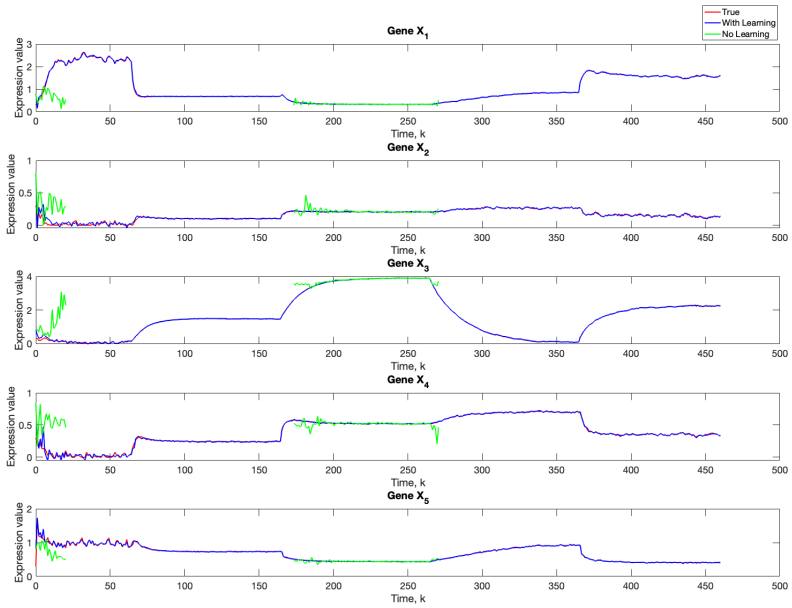


Figure 8.18: Comparison of true, PF estimated and BLT estimated expression level $x_{i,k}$ of Gene X_i, $i = 1, \ldots, 5$ in Scenario 3. The TV model configuration is provided in Table **??**.

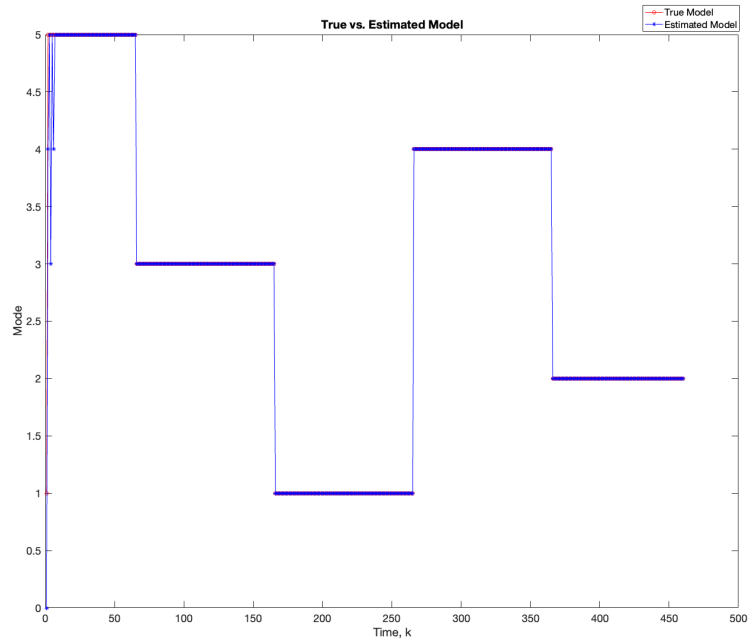


Figure 8.19: Comparison of true and BLT estimated labels of the configuration models in Scenario 3.

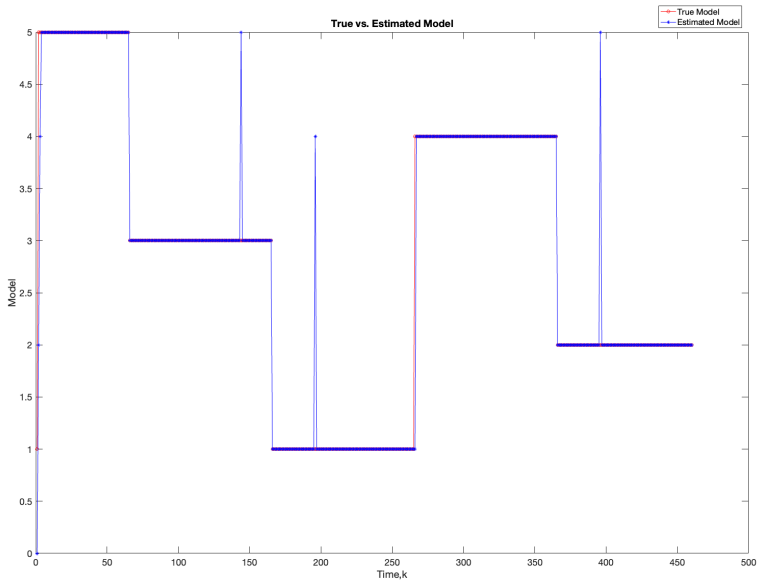


Figure 8.20: Comparison of true and BLT estimated labels of the configuration models in Scenario 3. The measurement noise intensity is 0.2

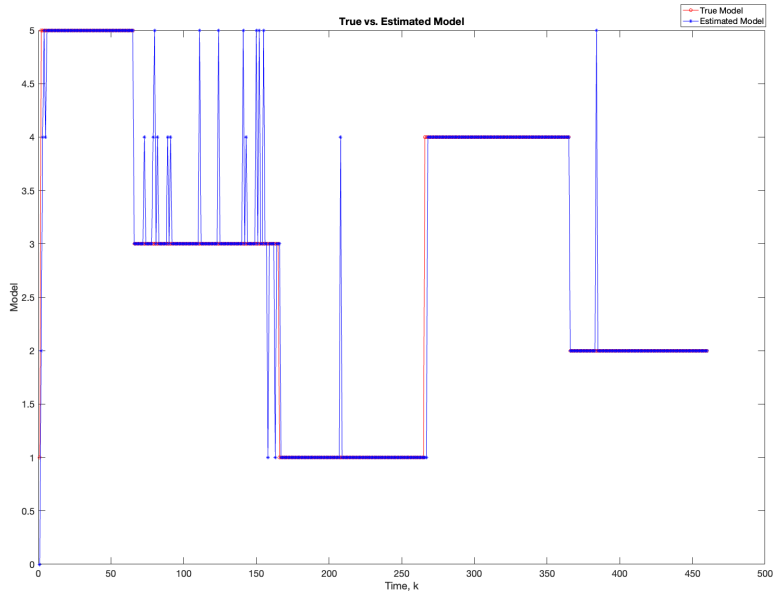


Figure 8.21: Comparison of true and BLT estimated labels of the configuration models in Scenario 3. The measurement noise intensity is 2.0

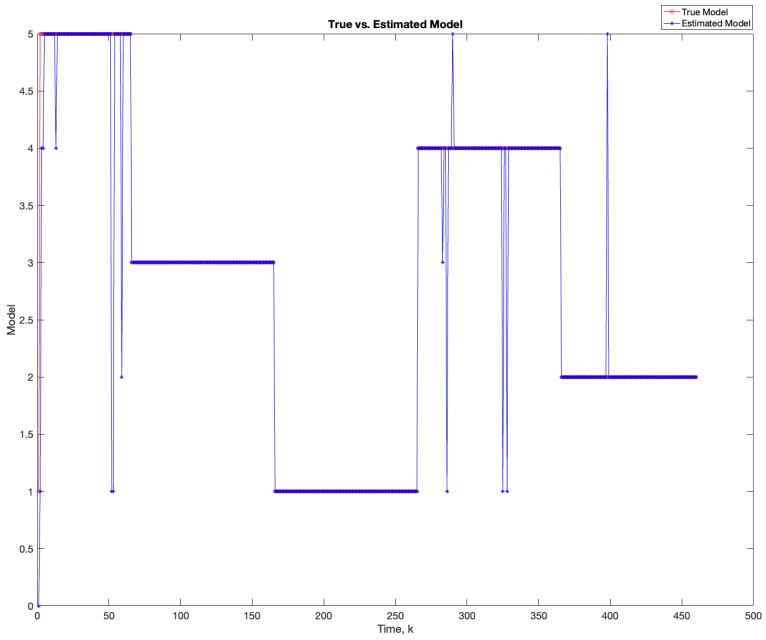


Figure 8.22: Comparison of true and BLT estimated labels of the configuration models in Scenario 1. The process noise intensity for segment 1 is 0.02, for segment 2 is 0.004, for segment 3 is 0.05, for segment 4 is 0.002, and for segment 5 is 0.03. The measurement noise intensity is 0.03.

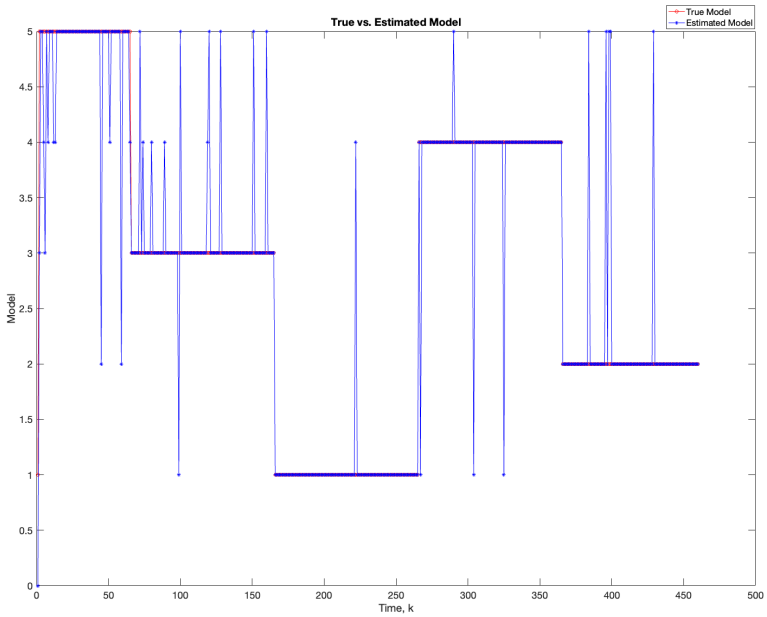


Figure 8.23: Comparison of true and BLT estimated labels of the configuration models in Scenario 1. The process noise intensity for segment 1 is 0.2, for segment 2 is 0.05, for segment 3 is 0.4, for segment 4 is 0.02, and for segment 5 is 0.3. The measurement noise intensity is 0.03.

Table 8.3: Parameter Values for the Five-Gene Network in Figure 1.

Gene ID	**MM Constant** K_{ij}	**Hill Coefficient** q_{ij}	**Max. Expression Rate** $V_{\max}$	**Deg. Rate** V_d
X_1	$K_{11} = 1.0$ $K_{12} = 2.0$ $K_{13} = 4.0$ $K_{14} = 1.0$ $K_{15} = 1.0$	1.0	14.0	0.50
X_2	$K_{21} = 1.0$ $K_{22} = 2.0$ $K_{23} = 4.0$ $K_{24} = 3.0$ $K_{25} = 1.0$	1.0	3.0	0.40
X_3	$K_{31} = 1.0$ $K_{32} = 2.0$ $K_{33} = 4.0$ $K_{34} = 3.0$ $K_{35} = 1.0$	1.0	4.0	0.10
X_4	$K_{41} = 1.0$ $K_{42} = 2.0$ $K_{43} = 4.0$ $K_{44} = 3.0$ $K_{45} = 1.0$	1.0	6.0	0.50
X_5	$K_{51} = 1.0$ $K_{52} = 2.0$ $K_{53} = 1.0$ $K_{54} = 1.0$ $K_{55} = 3.0$	1.0	11.0	0.80

Table 8.4: Kinetic order parameters for each of the five models in Figure 1.

Gene	**Kinetic Order Parameters**				
ID	$m = 1$	$m = 2$	$m = 3$	$m = 4$	$m = 5$
X_1	$g_{11} = 1, h_{11} = 1$ $g_{12} = 1, h_{12} = 1$ $g_{13} = 1, h_{13} = 1$ $g_{14} = 1, h_{14} = 1$ $g_{15} = 1, h_{15} = 1$	$g_{11} = 1, h_{11} = 1$ $g_{12} = 1, h_{12} = 0$ $g_{13} = 1, h_{13} = 1$ $g_{14} = 1, h_{14} = 1$ $g_{15} = 1, h_{15} = 1$	$g_{11} = 0, h_{11} = 1$ $g_{12} = 1, h_{12} = 0$ $g_{13} = 0, h_{13} = 1$ $g_{14} = 1, h_{14} = 0$ $g_{15} = 0, h_{15} = 1$	$g_{11} = 1, h_{11} = 1$ $g_{12} = 1, h_{12} = 0$ $g_{13} = 0, h_{13} = 1$ $g_{14} = 1, h_{14} = 1$ $g_{15} = 1, h_{15} = 0$	$g_{11} = 1, h_{11} = 0$ $g_{12} = 1, h_{12} = 1$ $g_{13} = 1, h_{13} = 1$ $g_{14} = 1, h_{14} = 1$ $g_{15} = 1, h_{15} = 1$
X_2	$g_{21} = 0, h_{21} = 1$ $g_{22} = 1, h_{22} = 1$ $g_{23} = 1, h_{23} = 1$ $g_{24} = 1, h_{24} = 1$ $g_{25} = 1, h_{25} = 1$	$g_{21} = 1, h_{21} = 1$ $g_{22} = 1, h_{22} = 1$ $g_{23} = 1, h_{23} = 0$ $g_{24} = 1, h_{24} = 1$ $g_{25} = 1, h_{25} = 1$	$g_{21} = 1, h_{21} = 0$ $g_{22} = 0, h_{22} = 1$ $g_{23} = 1, h_{23} = 0$ $g_{24} = 0, h_{24} = 1$ $g_{25} = 1, h_{25} = 0$	$g_{21} = 1, h_{21} = 1$ $g_{22} = 1, h_{22} = 1$ $g_{23} = 1, h_{23} = 0$ $g_{24} = 1, h_{24} = 1$ $g_{25} = 1, h_{25} = 1$	$g_{21} = 1, h_{21} = 1$ $g_{22} = 1, h_{22} = 0$ $g_{23} = 1, h_{23} = 1$ $g_{24} = 1, h_{24} = 1$ $g_{25} = 1, h_{25} = 1$
X_3	$g_{31} = 0, h_{31} = 1$ $g_{32} = 1, h_{32} = 1$ $g_{33} = 1, h_{33} = 1$ $g_{34} = 1, h_{34} = 1$ $g_{35} = 1, h_{35} = 1$	$g_{31} = 1, h_{31} = 1$ $g_{32} = 1, h_{32} = 1$ $g_{33} = 1, h_{33} = 1$ $g_{34} = 1, h_{34} = 0$ $g_{35} = 1, h_{35} = 1$	$g_{31} = 0, h_{31} = 1$ $g_{32} = 1, h_{32} = 0$ $g_{33} = 1, h_{33} = 0$ $g_{34} = 1, h_{34} = 0$ $g_{35} = 1, h_{35} = 1$	$g_{31} = 1, h_{31} = 1$ $g_{32} = 1, h_{32} = 1$ $g_{33} = 1, h_{33} = 0$ $g_{34} = 1, h_{34} = 1$ $g_{35} = 1, h_{35} = 1$	$g_{31} = 1, h_{31} = 1$ $g_{32} = 0, h_{32} = 1$ $g_{33} = 1, h_{33} = 0$ $g_{34} = 1, h_{34} = 0$ $g_{35} = 1, h_{35} = 1$
X_4	$g_{41} = 0, h_{41} = 1$ $g_{42} = 1, h_{42} = 1$ $g_{43} = 1, h_{43} = 1$ $g_{44} = 1, h_{44} = 1$ $g_{45} = 1, h_{45} = 1$	$g_{41} = 1, h_{41} = 1$ $g_{42} = 1, h_{42} = 1$ $g_{43} = 1, h_{43} = 1$ $g_{44} = 1, h_{44} = 1$ $g_{45} = 1, h_{45} = 0$	$g_{41} = 1, h_{41} = 1$ $g_{42} = 0, h_{42} = 1$ $g_{43} = 1, h_{43} = 1$ $g_{44} = 1, h_{44} = 1$ $g_{45} = 1, h_{45} = 1$	$g_{41} = 1, h_{41} = 1$ $g_{42} = 1, h_{42} = 1$ $g_{43} = 1, h_{43} = 0$ $g_{44} = 1, h_{44} = 1$ $g_{45} = 1, h_{45} = 1$	$g_{41} = 1, h_{41} = 1$ $g_{42} = 0, h_{42} = 1$ $g_{43} = 0, h_{43} = 1$ $g_{44} = 1, h_{44} = 0$ $g_{45} = 1, h_{45} = 1$
X_5	$g_{51} = 0, h_{51} = 1$ $g_{52} = 1, h_{52} = 1$ $g_{53} = 1, h_{53} = 1$ $g_{54} = 1, h_{54} = 1$ $g_{55} = 1, h_{55} = 1$	$g_{51} = 1, h_{51} = 0$ $g_{52} = 1, h_{52} = 1$ $g_{53} = 1, h_{53} = 1$ $g_{54} = 0, h_{54} = 1$ $g_{55} = 1, h_{55} = 1$	$g_{51} = 1, h_{51} = 1$ $g_{52} = 1, h_{52} = 1$ $g_{53} = 1, h_{53} = 1$ $g_{54} = 1, h_{54} = 1$ $g_{55} = 1, h_{55} = 1$	$g_{51} = 1, h_{51} = 1$ $g_{52} = 0, h_{52} = 1$ $g_{53} = 1, h_{53} = 0$ $g_{54} = 1, h_{54} = 1$ $g_{55} = 1, h_{55} = 1$	$g_{51} = 1, h_{51} = 0$ $g_{52} = 1, h_{52} = 0$ $g_{53} = 1, h_{53} = 0$ $g_{54} = 1, h_{54} = 0$ $g_{55} = 1, h_{55} = 0$

Table 8.5: Averaged RMSE for varying measurement noise intensities. For time steps $k = 1 : 65$, the dynamics are described by model 1; for time steps $k = 66 : 165$, the dynamics are described by model 2; and for time steps $k = 166 : 270$, the dynamics are described by model 3. We use 1,000 particles, 5,000 Monte Carlo runs and 10,000 Gibbs iterations were used.

Gene	2.0	0.2	$3e^{-2}$	$3e^{-3}$	$3e^{-5}$
X_1	0.065	0.041	0.017	0.017	0.018
X_2	0.069	0.042	0.018	0.017	0.019
X_3	0.071	0.051	0.019	0.019	0.018
X_4	0.070	0.037	0.019	0.018	0.019
X_5	0.071	0.043	0.021	0.018	0.018

Table 8.6: Averaged RMSE for varying process noise intensities. For time steps $k = 1 : 65$, the dynamics are described by model 1; for time steps $k = 66 : 165$, the dynamics are described by model 2; and for time steps $k = 166 : 270$, the dynamics are described by model 3. We use 1,000 particles, 5,000 Monte Carlo runs and 10,000 Gibbs iterations were used.

Gene	Segment 1: $2e^{-1}$ Segment 2: $5e^{-2}$ Segment 3: $4e^{-1}$	Segment 1: $2e^{-2}$ Segment 2: $4e^{-3}$ Segment 3: $5e^{-2}$	Segment 1: $2e^{-5}$ Segment 2: $5e^{-4}$ Segment 3: $4e^{-5}$
X_1	0.073	0.054	0.017
X_2	0.074	0.055	0.017
X_3	0.078	0.057	0.019
X_4	0.070	0.053	0.018
X_5	0.075	0.059	0.018

Table 8.7: Averaged RMSE for varying measurement noise intensities. For time steps $k = 1 : 65$, the dynamics are described by model 2; for time steps $k = 66 : 165$, the dynamics are described by model 1; and for time steps $k = 166 : 270$, the dynamics are described by model 3. We use 1,000 particles, 5,000 Monte Carlo runs and 10,000 Gibbs iterations were used.

Gene	2.0	0.2	$3e^{-2}$	$3e^{-3}$	$3e^{-5}$
X_1	0.061	0.049	0.016	0.015	0.014
X_2	0.060	0.048	0.014	0.014	0.015
X_3	0.067	0.049	0.015	0.017	0.017
X_4	0.054	0.046	0.014	0.016	0.014
X_5	0.063	0.051	0.016	0.016	0.017

Table 8.8: Averaged RMSE for varying process noise intensities. For time steps $k = 1 : 65$, the dynamics are described by model 2; for time steps $k = 66 : 165$, the dynamics are described by model 1; and for time steps $k = 166 : 270$, the dynamics are described by model 3. We use 1,000 particles, 5,000 Monte Carlo runs and 10,000 Gibbs iterations were used.

Gene	Segment 1: $2e^{-1}$ Segment 2: $5e^{-2}$ Segment 3: $4e^{-1}$	Segment 1: $2e^{-2}$ Segment 2: $4e^{-3}$ Segment 3: $5e^{-2}$	Segment 1: $2e^{-5}$ Segment 2: $5e^{-4}$ Segment 3: $4e^{-5}$
X_1	0.078	0.067	0.015
X_2	0.080	0.065	0.014
X_3	0.082	0.067	0.017
X_4	0.080	0.066	0.016
X_5	0.092	0.077	0.016

Table 8.9: Root mean-square error (RMSE) averaged across all time steps for varying measurement noise intensities. For time steps $k = 1 : 65$, the dynamics are described by model 3; for time steps $k = 66 : 165$, the dynamics are described by model 2; for time steps $k = 166 : 265$, the dynamics are described by model 1; and for $k = 265 : 370$, the dynamics are described by model 4. We use 1,000 particles, 5,000 Monte Carlo runs and 10,000 Gibbs iterations were used.

Gene	2.0	0.2	$3e^{-2}$	$3e^{-3}$	$3e^{-5}$
X_1	0.070	0.054	0.018	0.018	0.016
X_2	0.066	0.053	0.019	0.019	0.018
X_3	0.078	0.055	0.021	0.019	0.018
X_4	0.071	0.055	0.018	0.019	0.018
X_5	0.070	0.058	0.020	0.021	0.020

Table 8.10: Root mean-square error (RMSE) averaged across all time steps for varying process noise intensities. For time steps $k = 1 : 65$, the dynamics are described by model 3; for time steps $k = 66 : 165$, the dynamics are described by model 2; for time steps $k = 166 : 265$, the dynamics are described by model 1; and for $k = 265 : 370$, the dynamics are described by model 4. We use 1,000 particles, 5,000 Monte Carlo runs and 10,000 Gibbs iterations were used.

Gene	Segment 1: $2e^{-1}$ Segment 2: $5e^{-2}$ Segment 3: $4e^{-1}$ Segment 4: $2e^{-2}$	Segment 1: $2e^{-2}$ Segment 2: $4e^{-3}$ Segment 3: $5e^{-2}$ Segment 4: $2e^{-3}$	Segment 1: $2e^{-5}$ Segment 2: $5e^{-4}$ Segment 3: $4e^{-5}$ Segment 4: $2e^{-4}$
X_1	0.076	0.070	0.019
X_2	0.078	0.062	0.020
X_3	0.083	0.080	0.022
X_4	0.084	0.070	0.019
X_5	0.081	0.072	0.022

Table 8.11: Root mean-square error (RMSE) averaged across all time steps for varying measurement noise intensities. For time steps $k = 1 : 65$, the dynamics are described by model 5; for time steps $k = 66 : 165$, the dynamics are described by model 3; for time steps $k = 166 : 265$, the dynamics are described by model 1; for $k = 265 : 360$, the dynamics are described by model 4; and for $k = 360 : 460$ the dynamics are described by model 5. We use 1,000 particles, 5,000 Monte Carlo runs and 10,000 Gibbs iterations were used.

Gene	2.0	0.2	$3e^{-2}$	$3e^{-3}$	$3e^{-5}$
X_1	0.078	0.053	0.018	0.019	0.017
X_2	0.074	0.046	0.019	0.018	0.018
X_3	0.075	0.050	0.020	0.018	0.018
X_4	0.078	0.046	0.018	0.017	0.016
X_5	0.080	0.049	0.018	0.018	0.018

Table 8.12: Root mean-square error (RMSE) averaged across all time steps for varying process noise intensities. For time steps $k = 1 : 65$, the dynamics are described by model 5; for time steps $k = 66 : 165$, the dynamics are described by model 3; for time steps $k = 166 : 265$, the dynamics are described by model 1; for $k = 265 : 370$, the dynamics are described by model 4; and for $k = 371 : 460$, the dynamics are described by model 2. We use 1,000 particles, 5,000 Monte Carlo runs and 10,000 Gibbs iterations were used.

Gene	Segment 1: $2e^{-1}$ Segment 2: $5e^{-2}$ Segment 3: $4e^{-1}$ Segment 4: $2e^{-2}$ Segment 5: $3e^{-1}$	Segment 1: $2e^{-2}$ Segment 2: $4e^{-3}$ Segment 3: $5e^{-2}$ Segment 4: $2e^{-3}$ Segment 5: $3e^{-2}$	Segment 1: $2e^{-5}$ Segment 2: $5e^{-4}$ Segment 3: $4e^{-5}$ Segment 4: $2e^{-4}$ Segment 5: $3e^{-3}$
X_1	0.071	0.063	0.019
X_2	0.074	0.063	0.018
X_3	0.081	0.073	0.018
X_4	0.085	0.061	0.017
X_5	0.080	0.064	0.018

8.5 Particle Filter Derivation

Here, we derive the steps for the particle filter. At each time step, the joint posterior PDF from which we aim to draw samples is

$$p(\mathbf{x}_{1:k}^{(i)}, \mathbf{s}_{1:k}^{(i)}, \Sigma_w^{m,(i)}, \Sigma_v^{(i)}, \boldsymbol{\pi}_{m,1:k}^{(i)} | \mathbf{y}_{1:k}, \boldsymbol{\Psi}_w^{m,(i)}, \boldsymbol{\Psi}_v^{(i)}, \nu_w^{m,(i)}, \nu_v^{(i)}, \alpha) \tag{8.20}$$

The importance weights are given by

$$w_k^{(i)} = \frac{p(\mathbf{x}_{1:k}^{(i)}, \mathbf{s}_{1:k}^{(i)}, \Sigma_w^{m,(i)}, \Sigma_v^{(i)}, \boldsymbol{\pi}_{m,1:k}^{(i)} | \mathbf{y}_{1:k}, \boldsymbol{\Psi}_w^{m,(i)}, \boldsymbol{\Psi}_v^{(i)}, \nu_w^{m,(i)}, \nu_v^{(i)}, \alpha)}{q(\mathbf{x}_{1:k}^{(i)}, s_{1:k}^{(i)}, \Sigma_w^{m,(i)}, \Sigma_v^{(i)}, \boldsymbol{\pi}_{m,1:k}^{(i)} | \mathbf{y}_{1:k}, \boldsymbol{\Psi}_w^{m,(i)}, \boldsymbol{\Psi}_v^{(i)}, \nu_w^{m,(i)}, \nu_v^{(i)}, \alpha)} \tag{8.21}$$

The numerator can be factored as

$$p(\mathbf{x}_{1:k}^{(i)}, \mathbf{s}_{1:k}^{(i)}, \Sigma_w^{m,(i)}, \Sigma_v^{(i)}, \boldsymbol{\pi}_{m,1:k}^{(i)} | \mathbf{y}_{1:k}, \boldsymbol{\Psi}_w^{m,(i)}, \boldsymbol{\Psi}_v^{(i)}, \nu_w^{m,(i)}, \nu_v^{(i)}, \alpha) \tag{8.22}$$

$$= \frac{(\mathbf{x}_{1:k}^{(i)}, \mathbf{y}_{1:k}, \mathbf{s}_{1:k}^{(i)}, \Sigma_w^{m,(i)}, \Sigma_v^{(i)}, \boldsymbol{\pi}_{m,1:k}^{(i)}, \boldsymbol{\Psi}_w^{m,(i)}, \boldsymbol{\Psi}_v^{(i)}, \nu_w^{m,(i)}, \nu_v^{(i)}, \alpha)}{p(\mathbf{y}_{1:k}, \boldsymbol{\Psi}_w^{m,(i)}, \boldsymbol{\Psi}_v^{(i)}, \nu_w^{m,(i)}, \nu_v^{(i)}, \alpha)} \tag{8.23}$$

$$= p(\mathbf{y}_k | \mathbf{y}_{1:k-1}, \mathbf{x}_{1:k}^{(i)}, \Sigma_v^{(i)}, \boldsymbol{\Psi}_v, \nu_v^{(i)}) \frac{p(\mathbf{x}_{1:k}^{(i)}, \Sigma_w^{m,(i)}, \mathbf{s}_{1:k}^{(i)}, \boldsymbol{\pi}_{m,1:k}^{(i)} | \mathbf{y}_{1:k-1}, \boldsymbol{\Psi}_w^{m,(i)}, \nu_w^{m,(i)}, \alpha)}{p(\mathbf{y}_{1:k}, \boldsymbol{\Psi}_w^{m,(i)}, \boldsymbol{\Psi}_v^{(i)}, \nu_w^{m,(i)}, \nu_v^{(i)}, \alpha)} \tag{8.24}$$

$$\propto p(\mathbf{y}_k | \mathbf{y}_{1:k-1}, \mathbf{x}_{1:k}^{(i)}, \Sigma_v^{(i)}, \boldsymbol{\Psi}_v, \nu_v^{(i)}) p(\mathbf{x}_{1:k}^{(i)}, \Sigma_w^{m,(i)}, \mathbf{s}_{1:k}^{(i)}, \boldsymbol{\pi}_{m,1:k}^{(i)} | \mathbf{y}_{1:k-1}, , \boldsymbol{\Psi}_w^{m,(i)}, \nu_w^{m,(i)}, \alpha) \tag{8.25}$$

$$\propto p(\mathbf{y}_k | \mathbf{x}_k^{(i)}, \Sigma_v^{(i)}, \boldsymbol{\Psi}_v, \nu_v^{(i)}) \tag{8.26}$$

$$\cdot\, p(\mathbf{x}_k^{(i)}, \Sigma_w^{m,(i)}, s_k^{(i)}, \pi_{m,k}^{(i)} | \mathbf{y}_{1:k-1}, \mathbf{x}_{1:k-1}^{(i)}, \mathbf{s}_{1:k-1}^{(i)}, \boldsymbol{\pi}_{m,1:k-1}^{(i)}, \boldsymbol{\Psi}_w^{m,(i)}, \nu_w^{m,(i)}, \alpha) \tag{8.27}$$

$$\cdot\, p(\mathbf{x}_{1:k-1}^{(i)}, \Sigma_w^{m,(i)}, \mathbf{s}_{1:k-1}^{(i)}, \boldsymbol{\pi}_{m,1:k-1}^{(i)} | \mathbf{y}_{1:k-1}, \boldsymbol{\Psi}_w^{m,(i)}, \nu_w^{m,(i)}, \alpha) \tag{8.28}$$

$$\propto p(\mathbf{y}_k | \mathbf{x}_k^{(i)}, \Sigma_v^{(i)}, \boldsymbol{\Psi}_v, \nu_v^{(i)}) p(\mathbf{x}_k^{(i)}, \Sigma_w^{m,(i)} s_k^{(i)}, \pi_{m,k}^{(i)} | \mathbf{x}_{k-1}^{(i)}, \mathbf{s}_{k-1}^{(i)}, \boldsymbol{\Psi}_w^{m,(i)}, \nu_w^{m,(i)}, \alpha) \tag{8.29}$$

$$\cdot\, (\mathbf{x}_{1:k-1}^{(i)}, \mathbf{s}_{1:k-1}^{(i)}, \Sigma_w^{m,(i)}, \Sigma_v^{(i)}, \boldsymbol{\pi}_{m,1:k-1}^{(i)} | \mathbf{y}_{1:k-1}, \boldsymbol{\Psi}_w^{m,(i)}, \nu_w^{m,(i)}, \boldsymbol{\Psi}_v, \nu_v^{(i)}, \alpha) \tag{8.30}$$

$$\propto p(\mathbf{y}_k | \mathbf{x}_k^{(i)}, \Sigma_v^{(i)}, \boldsymbol{\Psi}_v, \nu_v^{(i)}) p(\mathbf{x}_k^{(i)} | \mathbf{x}_{k-1}^{(i)}, s_k^{(i)}, \Sigma_w^{m,(i)}) p(\Sigma_w^{m,(i)} | \boldsymbol{\Psi}_w^{m,(i)}, \nu_w^{m,(i)}) \tag{8.31}$$

$$\cdot\, p(\Sigma_v^{(i)} | \boldsymbol{\Psi}_v, \nu_v^{(i)}) p(s_k^{(i)} | s_{k-1}^{(i)}, \pi_{m,k}) p(\pi_{m,k} | \alpha) \tag{8.32}$$

$$\cdot\, p(\mathbf{x}_{1:k-1}^{(i)}, \mathbf{s}_{1:k-1}^{(i)}, \boldsymbol{\pi}_{m,1:k-1}^{(i)}, \Sigma_w^{m,(i)}, \Sigma_v^{(i)}, | \mathbf{y}_{1:k-1}, \boldsymbol{\Psi}_w^{m,(i)}, \nu_w^{m,(i)}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}, \alpha) \tag{8.33}$$

$$\tag{8.34}$$

where due to the Markov property and because $\pi_{m,k}^{(i)}$ does not depend on its previous values

$$p(\mathbf{x}_k^{(i)}, \Sigma_w^{m,(i)}, s_k^{(i)}, \pi_{m,k}^{(i)} | \mathbf{y}_{1:k-1}, \mathbf{x}_{1:k-1}^{(i)}, \mathbf{s}_{1:k-1}^{(i)}, \boldsymbol{\pi}_{m,1:k-1}^{(i)}, \boldsymbol{\Psi}_w^{m,(i)}, \nu_w^{m,(i)}, \alpha) \tag{8.35}$$

$$= p(\mathbf{x}_k^{(i)}, s_k^{(i)}, \Sigma_w^{m,(i)}, \pi_{m,k}^{(i)} | \mathbf{x}_{k-1}^{(i)}, s_{k-1}^{(i)}, \boldsymbol{\Psi}_w^{m,(i)}, \nu_w^{m,(i)}, \alpha) \tag{8.36}$$

$$= p(\mathbf{x}_k^{(i)} | \mathbf{x}_{k-1}^{(i)}, s_k^{(i)}, \Sigma_w^{m,(i)}) p(\Sigma_w^{m,(i)} | \boldsymbol{\Psi}_w^{m,(i)}, \nu_w^{m,(i)}) p(s_k^{(i)} | s_{k-1}^{(i)}, \pi_{m,k}^{(i)}) p(\pi_{m,k}^{(i)} | \alpha) \tag{8.37}$$

Next, we want to select the importance density so that if factorizes as

$$q(\mathbf{x}_{1:k}^{(i)}, s_{1:k}^{(i)}, \boldsymbol{\pi}_{m,1:k}^{(i)}, \Sigma_w^{m,(i)}, \Sigma_v^{(i)} | \mathbf{y}_{1:k}, \boldsymbol{\Psi}_w^{m,(i)}, \boldsymbol{\Psi}_v^{(i)}, \nu_w^{m,(i)}, \nu_v^{(i)}, \alpha) \tag{8.38}$$

$$= q(\mathbf{x}_k^{(i)}, s_k^{(i)}, \Sigma_w^{m,(i)}, \Sigma_v^{(i)}, \pi_{m,k}^{(i)} | \mathbf{x}_{1:k-1}^{(i)}, \mathbf{s}_{1:k-1}^{(i)}, \boldsymbol{\pi}_{m,1:k-1}^{(i)}, \mathbf{y}_{1:k}, \boldsymbol{\Psi}_w^{m,(i)}, \boldsymbol{\Psi}_v^{(i)}, \nu_w^{m,(i)}, \nu_v^{(i)}, \alpha) \tag{8.39}$$

$$\cdot\, q(\mathbf{x}_{1:k-1}^{(i)}, \mathbf{s}_{1:k-1}^{(i)}, \boldsymbol{\pi}_{m,1:k-1}^{(i)}, \Sigma_w^{m,(i)}, \Sigma_v^{(i)} | \mathbf{y}_{1:k-1}, \boldsymbol{\Psi}_w^{m,(i)}, \nu_w^{m,(i)}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}, \alpha) \tag{8.40}$$

For the weights, we have the following

$$w_k^{(i)} = \frac{\begin{bmatrix} p(\mathbf{y}_k | \mathbf{x}_k^{(i)}, \Sigma_v^{(i)}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}) p(\mathbf{x}_k^{(i)} | \mathbf{x}_{k-1}^{(i)}, \Sigma_w^{m,(i)}, s_k^{(i)}) p(\Sigma_w^{m,(i)} | \boldsymbol{\Psi}_w^{m,(i)}, \nu_w^{m,(i)}) p(\Sigma_v^{(i)} | \boldsymbol{\Psi}_v, \nu_v^{(i)}) \\ \cdot\, p(s_k^{(i)} | s_{k-1}^{(i)}, \pi_{m,k}^{(i)}) \\ p(\pi_{m,k}^{(i)} | \alpha) p(\mathbf{x}_{1:k-1}^{(i)}, \mathbf{s}_{1:k-1}^{(i)}, \boldsymbol{\pi}_{m,1:k-1}^{(i)}, \Sigma_w^{m,(i)}, \Sigma_v^{(i)} | \mathbf{y}_{1:k-1}, \boldsymbol{\Psi}_w^{m,(i)}, \nu_w^{m,(i)}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}, \alpha) \end{bmatrix}}{\begin{bmatrix} q(\mathbf{x}_k^{(i)}, s_k^{(i)}, \pi_{m,k}^{(i)}, \Sigma_w^{m,(i)}, \Sigma_v^{(i)} | \mathbf{x}_{1:k-1}^{(i)}, \mathbf{s}_{1:k-1}^{(i)}, \boldsymbol{\pi}_{m,1:k-1}^{(i)}, \mathbf{y}_{1:k}, \boldsymbol{\Psi}_w^{m,(i)}, \boldsymbol{\Psi}_v^{(i)}, \nu_w^{m,(i)}, \nu_v^{(i)}, \alpha) \\ \cdot\, q(\mathbf{x}_{1:k-1}^{(i)}, \mathbf{s}_{1:k-1}^{(i)}, \boldsymbol{\pi}_{m,1:k-1}^{(i)}, \Sigma_w^{m,(i)}, \Sigma_v^{(i)} | \mathbf{y}_{1:k-1}, \boldsymbol{\Psi}_w^{m,(i)}, \nu_w^{m,(i)}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)} \alpha) \end{bmatrix}} \tag{8.41}$$

$$= \frac{p(\mathbf{x}_{1:k-1}^{(i)}, \mathbf{s}_{1:k-1}^{(i)}, \boldsymbol{\pi}_{m,1:k-1}^{(i)}, \Sigma_w^{m,(i)}, \Sigma_v^{(i)} | \mathbf{y}_{1:k-1}, \boldsymbol{\Psi}_w^{m,(i)}, \nu_w^{m,(i)}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}, \alpha)}{q(\mathbf{x}_{1:k-1}^{(i)}, \mathbf{s}_{1:k-1}^{(i)}, \boldsymbol{\pi}_{m,1:k-1}^{(i)}, \Sigma_w^{m,(i)}, \Sigma_v^{(i)} | \mathbf{y}_{1:k-1}, \boldsymbol{\Psi}_w^{m,(i)}, \nu_w^{m,(i)}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}, \alpha)} \tag{8.42}$$

$$\cdot \begin{bmatrix} p(\mathbf{y}_k | \mathbf{x}_k^{(i)}, \Sigma_v^{(i)}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}) p(\mathbf{x}_k^{(i)} | \mathbf{x}_{k-1}^{(i)}, \Sigma_w^{m,(i)}, s_k^{(i)}) p(\Sigma_w^{m,(i)} | \boldsymbol{\Psi}_w^{m,(i)}, \nu_w^{m,(i)}) \\ \cdot\, p(\Sigma_v^{(i)} | \boldsymbol{\Psi}_v, \nu_v^{(i)}) p(s_k^{(i)} | s_{k-1}^{(i)}, \pi_{m,k}^{(i)}) p(\pi_{m,k}^{(i)} | \alpha) \end{bmatrix}$$

$$\cdot \frac{}{q(\mathbf{x}_k^{(i)}, s_k^{(i)}, \pi_{m,k}^{(i)}, \Sigma_w^{m,(i)}, \Sigma_v^{(i)} | \mathbf{x}_{1:k-1}^{(i)}, \mathbf{s}_{1:k-1}^{(i)}, \mathbf{y}_{1:k}, \boldsymbol{\Psi}_w^{m,(i)}, \boldsymbol{\Psi}_v^{(i)}, \nu_w^{m,(i)}, \nu_v^{(i)}, \alpha)} \tag{8.43}$$

Choosing the importance density to be the prior PDF of the unknown parameters

$$q(\mathbf{x}_k^{(i)}, s_k^{(i)}, \pi_{m,k}^{(i)}, \Sigma_w^{m,(i)}, \Sigma_v^{(i)} | \mathbf{x}_{1:k-1}^{(i)}, \mathbf{s}_{1:k-1}^{(i)}, , \boldsymbol{\pi}_{m,1:k-1}^{(i)}, \mathbf{y}_{1:k}, \boldsymbol{\Psi}_w^{m,(i)}, \boldsymbol{\Psi}_v^{(i)}, \nu_w^{m,(i)}, \nu_v^{(i)}, \alpha) \tag{8.44}$$

$$= p(\mathbf{x}_k^{(i)}, s_k^{(i)}, \pi_{m,k}^{(i)}, \Sigma_w^{m,(i)}, \Sigma_v^{(i)} | \mathbf{x}_{k-1}^{(i)}, s_{k-1}^{(i}, \Sigma_w^{m,(i)}, \Sigma_v^{(i)}, \boldsymbol{\Psi}_w^{m,(i)}, \boldsymbol{\Psi}_v^{(i)}, \nu_w^{m,(i)}, \nu_v^{(i)}, \alpha) \tag{8.45}$$

$$= p(\mathbf{x}_k^{(i)} | \mathbf{x}_{k-1}^{(i)}, s_k^{(i)}, \Sigma_w^{m,(i)}) p(s_k^{(i)} | s_{k-1}^{(i)}, \pi_{m,k}^{(i)}) p(\Sigma_w^{m,(i)} | \boldsymbol{\Psi}_w^{m,(i)}, \nu_w^{m,(i)}) \tag{8.46}$$

$$\cdot\, p(\pi_{m,k}^{(i)} | \alpha) p(\Sigma_v^{(i)} | \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}) \tag{8.47}$$

we obtain

$$w_k^{(i)} = w_{k-1}^{(i)} \frac{\begin{bmatrix} p(\mathbf{y}_k | \mathbf{x}_k^{(i)}, \Sigma_v^{(i)}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}) p(\mathbf{x}_k^{(i)} | \mathbf{x}_{k-1}^{(i)}, s_k^{(i)}, \Sigma_w^{m,(i)}) p(s_k^{(i)} | s_{k-1}^{(i)}, \pi_{m,k}^{(i)}) \\ \cdot\, p(\Sigma_w^{m,(i)} | \boldsymbol{\Psi}_w^{m,(i)}, \nu_w^{m,(i)}) p(\Sigma_v^{(i)} | \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}) p(\pi_{m,k}^{(i)} | \alpha) \end{bmatrix}}{\begin{bmatrix} p(\mathbf{x}_k^{(i)} | \mathbf{x}_{k-1}^{(i)}, s_k^{(i)}, \Sigma_w^{m,(i)}) p(s_k^{(i)} | s_{k-1}^{(i)}, \pi_{m,k}^{(i)}) p(\Sigma_w^{m,(i)} | \boldsymbol{\Psi}_w^{m,(i)}, \nu_w^{m,(i)}) \\ \cdot\, p(\Sigma_v^{(i)} | \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}) p(\pi_{m,k}^{(i)} | \alpha) \end{bmatrix}} \tag{8.48}$$

$$= w_{k-1}^{(i)} p(\mathbf{y}_k | \mathbf{x}_k^{(i)}, \Sigma_v^{(i)}, \boldsymbol{\Psi}_v^{(i)}, \nu_v^{(i)}) \tag{8.49}$$

Chapter 9

CONCLUSION AND FUTURE DIRECTIONS

This dissertation has centered on the development of Bayesian learning methods for identifying different types of variation in gene regulatory networks (GRNs). We introduced a new nonlinear multivariate state-space model for modeling GRNs. We also develop several stochastic models of gene regulation aimed at more precisely capturing the complexity of GRNs. We used these models in Chapter 6 to estimate unknown gene expression trajectories as well as the unknown noise model in GRNs under switching noise dynamics. Next, we considered the problem of learning variation in time-varying GRN architectures and developed a sequential Monte Carlo (SMC) algorithm for estimating the unknown gene expression trajectories as well as indicator corresponding to each GRN architecture. Finally, we extended this Bayesian model to a fully Bayesian hierarchical model, which incorporates uncertainty in both the process and measurement noise covariance matrices. Since the unknown state depends on the unknown process noise covariance, we implemented a local Gibbs step in our SMC algorithm which showed improved detection accuracy.

It is worth noting that the methodology developed in this work is not confined to the analysis of gene regulatory networks but can also be generalized to other domains which involve switching dynamics in nonlinear systems. For example, our methodology can be applied to change point detection problems in financial markets, where a sudden shift in the market dynamics could be caused by changes in nonlinear interactions between different economic factors. Future research will focus on extending our methodology to larger systems as well as applying it to a broader range of problems in nonlinear systems analysis.

Another area for future research centers on the exploration of Bayesian non-parametric methods. Variation in biological and technical factors can give rise to multimodal measurement noise in microarray data. For example, differences in the quality of samples, data processing effects, and different experimental protocols can all contribute to the presence of multimodality in microarray measurement noise. This presents two main challenges in estimating gene regulatory networks from microarray data. The first is that the number of modes in the measurement noise distribution is unknown a priori. Second, the measurement noise can vary with time, particularly if the data are collected under different experimental conditions or for a system exhibiting biological stochasticity, such as a change in the developmental process. This facilitates the development of flexible statistical methods aimed at estimating gene regulatory networks in the presence of multimodal and time-varying measurement noise. Thus, an avenue for further development is to explore the use of time-dependent Dirichlet processes to estimate the gene expression state and the time-varying measurement noise density in gene regulatory networks.

References

Aalto, A., L. Viitasaari, P. Ilmonen, L. Mombaerts and J. Goncalves, "Continuous time gaussian process dynamical models in gene regulatory network inference", arXiv preprint arXiv:1808.08161 (2018).

Adabor, E. S. and G. K. Acquaah-Mensah, "Restricted-derestricted dynamic bayesian network inference of transcriptional regulatory relationships among genes in cancer", Computational Biology and Chemistry **79**, 155–164 (2019).

Ahmed, A. and E. P. Xing, "Recovering time-varying networks of dependencies in social and biological studies", Proceedings of the National Academy of Sciences **106**, 29, 11878–11883 (2009).

Ahmed, S. S., S. Roy and J. Kalita, "Assessing the effectiveness of causality inference methods for gene regulatory networks", IEEE/ACM transactions on computational biology and bioinformatics **17**, 1, 56–70 (2018).

Äijö, T. and H. Lähdesmäki, "Learning gene regulatory networks from gene expression measurements using non-parametric molecular kinetics", Bioinformatics **25**, 22, 2937–2944 (2009).

Akers, K. and T. Murali, "Gene regulatory network inference in single-cell biology", Current Opinion in Systems Biology **26**, 87–97 (2021).

Alon, U., "Network motifs: Theory and experimental approaches", Nature Reviews Genetics **8**, 6, 450–461 (2007).

Alon, U., *An introduction to systems biology: design principles of biological circuits* (CRC press, 2019).

Amor, N., A. Meddeb, S. Marrouchi and S. Chebbi, "A comparative study of nonlinear bayesian filtering algorithms for estimation of gene expression time series data", Turkish Journal of Electrical Engineering and Computer Sciences **27**, 2648–2665 (2019).

Ancherbak, S., E. E. Kuruoglu and M. Vingron, "Time-dependent gene network modelling by sequential monte carlo", IEEE/ACM Transactions on Computational Biology and Bioinformatics **13**, 6, 1183–1193 (2016a).

Ancherbak, S., E. E. Kuruoglu and M. Vingron, "Time-dependent gene network modelling by sequential monte carlo", IEEE/ACM Transactions on Computational Biology and Bioinformatics **13**, 6, 1183–1193 (2016b).

Anteneodo, C. and C. Tsallis, "Multiplicative noise: A mechanism leading to nonextensive statistical mechanics", Journal of Mathematical Physics **44**, 11, 5194–5203 (2003).

Arasaratnam, I. and S. Haykin, "Cubature kalman filters", IEEE Transactions on Automatic Control **54**, 6, 1254–1269 (2009).

Arulampalam, M. S., S. Maskell, N. Gordon and T. Clapp, "A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking", IEEE Transactions on signal processing **50**, 2, 174–188 (2002a).

Arulampalam, M. S., S. Maskell, N. Gordon and T. Clapp, "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking", IEEE Transactions on Signal Processing **50**, 174–188 (2002b).

Asif, H. S. and G. Sanguinetti, "Simultaneous inference and clustering of transcriptional dynamics in gene regulatory networks", Statistical applications in genetics and molecular biology **12**, 5, 545–557 (2013).

Assi, S. A., M. R. Imperato, D. J. Coleman, A. Pickin, S. Potluri, A. Ptasinska, P. S. Chin, H. Blair, P. Cauchy, S. R. James *et al.*, "Subtype-specific regulatory network rewiring in acute myeloid leukemia", Nature genetics **51**, 1, 151–162 (2019).

Banf, M. and S. Y. Rhee, "Computational inference of gene regulatory networks: approaches, limitations and opportunities", Biochimica et Biophysica Acta (BBA)-Gene Regulatory Mechanisms **1860**, 1, 41–52 (2017).

Barman, S. and Y.-K. Kwon, "A novel mutual information-based boolean network inference method from time-series gene expression data", PloS one **12**, 2, e0171097 (2017).

Bellot Pujalte, P., "Study of gene regulatory networks inference methods from gene expression data", (2017).

Blackwell, D. and J. B. MacQueen, "Ferguson distributions via pólya urn schemes", The annals of statistics **1**, 2, 353–355 (1973).

Bonev, B. and G. Cavalli, "Organization and function of the 3d genome", Nature Reviews Genetics **17**, 11, 661 (2016).

Bugallo, M. F., Ç. Taşdemir and P. M. Djurić, "Estimation of gene expression by a bank of particle filters", in "23rd European Signal Processing Conference (EUSIPCO)", pp. 494–498 (2015).

Burdziak, C., E. Azizi, S. Prabhakaran and D. Pe'er, "A nonparametric multi-view model for estimating cell type-specific gene regulatory networks", arXiv preprint arXiv:1902.08138 (2019).

Callaham, J., J.-C. Loiseau, G. Rigas and S. Brunton, "Nonlinear stochastic modelling with langevin regression", Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences **477**, 20210092 (2021).

Chai, L. E., S. K. Loh, S. T. Low, M. S. Mohamad, S. Deris and Z. Zakaria, "A review on the computational approaches for gene regulatory network construction", Computers in biology and medicine **48**, 55–65 (2014).

Chang, L., B. Hu, A. Li and F. Qin, "Transformed unscented kalman filter", IEEE Transactions on Automatic Control **58**, 1, 252–257 (2012).

Chopin, N., "A sequential particle filter method for static models", Biometrika **89**, 539–551 (2002).

Chowdhury, A., M. Chetty and R. Evans, "Stochastic s-system modeling of gene regulatory network", Cognitive Neurodynamics **9**, 5, 535–547 (2015).

Clune, J., J.-B. Mouret and H. Lipson, "The evolutionary origins of modularity", Proceedings of the Royal Society b: Biological sciences **280**, 1755, 20122863 (2013).

Cortez, M. J., H. Hong, B. Choi, J. K. Kim and K. Josić, "Hierarchical bayesian models of transcriptional and translational regulation processes with delays", Bioinformatics **38**, 1, 187–195 (2022).

Dari, A., B. Kia, X. Wang, A. R. Bulsara and W. Ditto, "Noise-aided computation within a synthetic gene network through morphable and robust logic gates", Physical Review E **83**, 4, 041909 (2011).

Darwin, C., *The origin of species* (PF Collier & son New York, 1909).

de Campos, L. M., A. Cano, J. G. Castellano and S. Moral, "Combining gene expression data and prior knowledge for inferring gene regulatory networks via bayesian networks using structural restrictions", Statistical Applications in Genetics and Molecular Biology **18**, 3 (2019).

de Leon, S. B.-T. and E. H. Davidson, "Gene regulation: gene control network in development", Annual Review of Biophysics and Biomolecular Structure **36**, 191–212 (2007).

de Luis Balaguer, M. A. and R. Sozzani, "Inferring gene regulatory networks in the arabidopsis root using a dynamic bayesian network approach", in "Plant Gene Regulatory Networks", pp. 331–348 (Springer, 2017).

Delgado, F. M. and F. Gómez-Vela, "Computational methods for gene regulatory networks reconstruction and analysis: a review", Artificial intelligence in medicine **95**, 133–145 (2019).

Dondelinger, F., S. Lebre and D. Husmeier, "Non-homogeneous dynamic bayesian networks with bayesian regularization for inferring gene regulatory networks with gradually time-varying structure", Machine Learning **90** (2013).

Dony, L., F. He and M. P. Stumpf, "Parametric and non-parametric gradient matching for network inference", bioRxiv p. 254003 (2018).

Elahi, F. and A. Hasan, "A method for estimating hill function-based dynamic models of gene regulatory networks", (2018).

Erwin, D., "Evolution of the bilaterian body plan", (2019).

Erwin, D. H. and E. H. Davidson, "The evolution of hierarchical gene regulatory networks", Nature Reviews Genetics **10**, 2, 141–148 (2009).

Ferguson, T. S., "A bayesian analysis of some nonparametric problems", The annals of statistics pp. 209–230 (1973).

Fraser, G. J., C. D. Hulsey, R. F. Bloomquist, K. Uyesugi, N. R. Manley and J. T. Streelman, "An ancient gene network is co-opted for teeth on old and new jaws", PLoS Biol **7**, 2, e1000031 (2009).

Gelfand, A. E., S. E. Hills, A. Racine-Poon and A. F. M. Smith, "Illustration of Bayesian inference in normal data models using Gibbs sampling", Journal of the American Statistical Association **85**, 972–985 (1990).

Gelman, A., J. B. Carlin, H. S. Stern and D. B. Rubin, *Bayesian data analysis* (Chapman and Hall/CRC, 1995a).

Gelman, A., J. B. Carlin, H. S. Stern and D. B. Rubin, *Bayesian Data Analysis* (Chapman and Hall/CRC, 1995b).

Grzegorczyk, M. and D. Husmeier, "Non-stationary continuous dynamic bayesian networks", Advances in neural information processing systems **22** (2009).

Ha, D., D. Kim, I. Kim, Y. Oh, J. Kong, S. K. Han and S. Kim, "Evolutionary rewiring of regulatory networks contributes to phenotypic differences between human and mouse orthologous genes", Nucleic Acids Research **50**, 4, 1849–1863, URL `https://doi.org/10.1093/nar/gkac050` (2022).

Hallac, D., Y. Park, S. Boyd and J. Leskovec, "Network inference via the time-varying graphical lasso", in "Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining", pp. 205–213 (2017).

Hardison, R. C. and J. Taylor, "Genomic approaches towards finding cis-regulatory modules in animals", Nature Reviews Genetics **13**, 7, 469–483 (2012).

Hare, E. E., B. K. Peterson, V. N. Iyer, R. Meier and M. B. Eisen, "Sepsid even-skipped enhancers are functionally conserved in drosophila despite lack of sequence conservation", PLoS Genet **4**, 6, e1000106 (2008).

Hartwell, L. H., J. J. Hopfield, S. Leibler and A. W. Murray, "From molecular to modular cell biology", Nature **402**, 6761, C47–C52 (1999).

Hasty, J., J. Pradines, M. Dolnik and J. Collins, "Noise-based switches and amplifiers for gene expression", Proceedings of the National Academy of Sciences **97**, 5, 2075–2080 (2000).

Hinman, V. F. and A. M. Cheatle Jarvela, "Developmental gene regulatory network evolution: Insights from comparative studies in echinoderms", genesis **52**, 3, 193–207, URL `https://onlinelibrary.wiley.com/doi/abs/10.1002/dvg.22757` (2014).

Ho, J. W. and M. A. Charleston, "Network modelling of gene regulation", Biophysical Reviews **3**, 1, 1–13 (2011).

Hossain, S. M. M., A. A. Halsana, L. Khatun, S. Ray and A. Mukhopadhyay, "Discovering key transcriptomic regulators in pancreatic ductal adenocarcinoma using dirichlet process gaussian mixture model", Scientific reports **11**, 1, 1–15 (2021).

Huynh-Thu, V. A. and P. Geurts, "dyngenie3: dynamical genie3 for the inference of gene networks from time series expression data", Scientific reports **8**, 1, 1–12 (2018).

Imani, M. and U. M. Braga-Neto, "Particle filters for partially-observed boolean dynamical systems", Automatica **87**, 238–250 (2018).

Jensen, S. T., G. Chen and C. J. Stoeckert Jr, "Bayesian variable selection and data integration for biological regulatory networks", The Annals of Applied Statistics **1**, 2, 612–633 (2007).

Jordan, M. I. and Y. W. Teh, "A gentle introduction to the dirichlet process, the beta process and bayesian nonparametrics [draft: Please do not distribute]", (2015).

Kauffman, S., "Homeostasis and differentiation in random genetic control networks", Nature **224**, 5215, 177–178 (1969).

Kaur, P., A. Singh and I. Chana, "Computational techniques and tools for omics data analysis: state-of-the-art, challenges, and future directions", Archives of Computational Methods in Engineering **28**, 7, 4595–4631 (2021).

Kherdjemil, Y., R. L. Lalonde, R. Sheth, A. Dumouchel, G. de Martino, K. M. Pineault, D. M. Wellik, H. S. Stadler, M.-A. Akimenko and M. Kmita, "Evolution of hoxa11 regulation in vertebrates is linked to the pentadactyl state", Nature **539**, 7627, 89–92 (2016).

Krishnan, M., M. Small, A. Bosco and T. Stemler, "Network using michaelis–menten kinetics: Constructing an algorithm to find target genes from expression data", Journal of Complex Networks **8**, 1, cnz016 (2020a).

Krishnan, M., M. Small, A. Bosco and T. Stemler, "Network using Michaelis-Menten kinetics: Constructing an algorithm to find target genes from expression data", Journal of Complex Networks **8** (2020b).

Lacquaniti, F., Y. P. Ivanenko, A. d'Avella, K. Zelik and M. Zago, "Evolutionary and developmental modules", Frontiers in Computational Neuroscience **7**, 61 (2013).

Lähdesmäki, H., S. Hautaniemi, I. Shmulevich and O. Yli-Harja, "Relationships between probabilistic boolean networks and dynamic bayesian networks as models of gene regulatory networks", Signal processing **86**, 4, 814–834 (2006).

Lebre, S., J. Becq, F. Devaux, M. P. Stumpf and G. Lelandais, "Statistical inference of the time-varying structure of gene-regulation networks", BMC systems biology **4**, 1, 1–16 (2010).

Lee, D. S. and N. K. Chia, "A particle algorithm for sequential Bayesian parameter estimation and model selection", IEEE Transaction on Signal Processing **50**, 326–336 (2002).

Lee, T. H., S. Lakshmanan, J. H. Park and P. Balasubramaniam, "State estimation for genetic regulatory networks with mode-dependent leakage delays, time-varying delays, and markovian jumping parameters", IEEE Transactions on NanoBioscience **12**, 4, 363–375 (2013).

Li, P., P. Gong, H. Li, E. J. Perkins, N. Wang and C. Zhang, "Gene regulatory network inference and validation using relative change ratio analysis and time-delayed dynamic bayesian network", EURASIP Journal on Bioinformatics and Systems Biology **2014**, 1, 1–10 (2014).

Li, Y., H. Chen, J. Zheng and A. Ngom, "The max-min high-order dynamic bayesian network for learning gene regulatory networks with time-delayed regulations", IEEE/ACM transactions on computational biology and bioinformatics **13**, 4, 792–803 (2015).

Li, Y. and A. Papandreou-Suppappola, "Instantaneous frequency estimation using sequential Bayesian techniques", in "Asilomar Conference on Signals, Systems, and Computers", pp. 569–573 (2006).

Liang, Y. and A. Kelemen, "Bayesian state space models for dynamic genetic network construction across multiple tissues", Statistical Applications in Genetics and Molecular Biology **15**, 4, 273–290 (2016).

Lim, C. Y., H. Wang, S. Woodhouse, N. Piterman, L. Wernisch, J. Fisher and B. Göttgens, "Btr: training asynchronous boolean models using single-cell expression data", BMC bioinformatics **17**, 1, 1–18 (2016).

Liu, F., S.-W. Zhang, W.-F. Guo, Z.-G. Wei and L. Chen, "Inference of gene regulatory network based on local bayesian networks", PLoS computational biology **12**, 8, e1005024 (2016).

Lobo, V., A. Patil, A. Phatak and N. Chandra, "Free radicals, antioxidants and functional foods: impact on human health. pharmacogn rev 4:118-126", Pharmacognosy reviews **4**, 118–26 (2010).

Lopes, M. and G. Bontempi, "Experimental assessment of static and dynamic algorithms for gene regulation inference from time series expression data", Frontiers in genetics **4**, 303 (2013).

Madhamshettiwar, P. B., S. R. Maetschke, M. J. Davis, A. Reverter and M. A. Ragan, "Gene regulatory network inference: evaluation and application to ovarian cancer allows the prioritization of drug targets", Genome medicine **4**, 5, 1–16 (2012).

Maki, Y., T. Ueda, M. Okamoto, N. Uematsu, K. Inamura, K. Uchida, Y. Takahashi and Y. Eguchi, "Inference of genetic network using the expression profile time course data of mouse p19 cells", Genome Informatics **13**, 382–383 (2002).

Margolin, A. A., I. Nemenman, K. Basso, C. Wiggins, G. Stolovitzky, R. Dalla Favera and A. Califano, "Aracne: an algorithm for the reconstruction of gene regulatory networks in a mammalian cellular context", in "BMC bioinformatics", vol. 7, p. S7 (Springer, 2006).

Martik, M. L. and M. E. Bronner, "Regulatory logic underlying diversification of the neural crest", Trends in Genetics **33**, 10, 715–727 (2017).

Martik, M. L. and D. R. McClay, "Deployment of a retinal determination gene network drives directed cell migration in the sea urchin embryo", Elife **4**, e08827 (2015).

McClenny, L. D., M. Imani and U. M. Braga-Neto, "Boolean kalman filter with correlated observation noise", in "2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)", pp. 866–870 (IEEE, 2017).

McKenzie, G. J., R. S. Harris, P. L. Lee and S. M. Rosenberg, "The sos response regulates adaptive mutation", Proceedings of the National Academy of Sciences of the United States of America **97**, 12, 6646–6651, URL `http://www.jstor.org/stable/122691` (2000).

McNames, J., "Optimal tracking", (2016).

Meister, A., C. Du, Y. Li and W. Wong, "Modeling stochastic noise in gene regulatory systems", Quantitative biology **2**, 1, 1–29 (2014).

Mercatelli, D., L. Scalambra, L. Triboli, F. Ray and F. M. Giorgi, "Gene regulatory network inference resources: A practical overview", Biochimica et Biophysica Acta (BBA)-Gene Regulatory Mechanisms **1863**, 6, 194430 (2020).

Michailidis, G. and F. d'Alché Buc, "Autoregressive models for gene regulatory network inference: Sparsity, stability and causality issues", Mathematical biosciences **246**, 2, 326–334 (2013).

Mittal, N., J. Guimaraes, T. Gross, A. Schmidt, A. Vina, D. Nedialkova, F. Aeschimann, S. Leidel, A. Spang and M. Zavolan, "The gcn4 transcription factor reduces protein synthesis capacity and extends yeast lifespan", Nature Communications **8** (2017).

Moignard, V., S. Woodhouse, L. Haghverdi, A. J. Lilly, Y. Tanaka, A. C. Wilkinson, F. Buettner, I. C. Macaulay, W. Jawaid, E. Diamanti *et al.*, "Decoding the regulatory network of early blood development from single-cell gene expression measurements", Nature biotechnology **33**, 3, 269–276 (2015).

Monteiro, A. and O. Podlaha, "Wings, horns, and butterfly eyespots: How do complex traits evolve?", PLoS Biol **7**, 2, e1000037 (2009).

Moraffah, B., *Bayesian nonparametric modeling and inference for multiple object tracking* (Arizona State University, 2019).

Mundt, M., A. Anders, S. Murray and V. Sourjik, "A system for gene expression noise control in yeast", ACS Synthetic Biology **7**, 11, 2618–2626 (2018).

Nadal, E. and F. Posas, "The hog pathway and the regulation of osmoadaptive responses in yeast", FEMS Yeast Research **22** (2022).

Nguyen, H., D. Tran, B. Tran, B. Pehlivan and T. Nguyen, "A comprehensive survey of regulatory network inference methods using single cell rna sequencing data", Briefings in bioinformatics **22**, 3, bbaa190 (2021).

Ni, Y., F. C. Stingo and V. Baladandayuthapani, "Bayesian nonlinear model selection for gene regulatory networks", Biometrics **71**, 3, 585–595 (2015).

Niswander, L., "Principles of development", (2019).

Noor, A., E. Serpedin, M. Nounou and H. Nounou, "Inferring gene regulatory networks via nonlinear state-space models and exploiting sparsity", IEEE/ACM Transactions on Computational Biology and Bioinformatics **9**, 4, 1203–1211 (2012).

Novikov, E. and E. Barillot, "Model selection in the reconstruction of regulatory networks from time-series data", BMC research notes **2**, 1, 1–7 (2009).

Pan, W., F. Menolascina and G.-B. Stan, "Online model selection for synthetic gene networks", in "2016 IEEE 55th Conference on Decision and Control (CDC)", pp. 776–782 (IEEE, 2016).

Panchal, V. and D. F. Linder, "Reverse engineering gene networks using global–local shrinkage rules", Interface focus **10**, 1, 20190049 (2020).

Pearl, J., "Introduction to probabilities, graphs, and causal models", Causality: Models, Reasoning and Inference pp. 1–40 (2000).

Peter, I., "Introduction to gene regulatory networks and the genomic control of development", Gene Regulatory Networks in Development (2019).

Peter, I. S. and E. H. Davidson, "Evolution of gene regulatory networks controlling body plan development", Cell **144**, 6, 970–985, URL `https://www.cell.com/fulltext/S0092-8674(11)00131-0` (2011).

Peter, I. S. and E. H. Davidson, *Genomic Control Process: Development and Evolution* (Academic Press, 2015).

Pirgazi, J. and A. R. Khanteymoori, "A robust gene regulatory network inference method base on kalman filter and linear regression", PLOS ONE **13**, 7, 1–17, URL `https://doi.org/10.1371/journal.pone.0200094` (2018).

Pitt, M. K. and N. Shephard, "Filtering via simulation: Auxiliary particle filters", Journal of the American Statistical Association **94**, 446, 590–599 (1999).

Potkin, S. G., F. Macciardi, G. Guffanti, J. H. Fallon, Q. Wang, J. A. Turner, A. Lakatos, M. F. Miles, A. Lander, M. P. Vawter and X. Xie, "Identifying gene regulatory networks in schizophrenia", NeuroImage **53**, 3, 839–847, URL `https://www.sciencedirect.com/science/article/pii/S1053811910008803`, imaging Genetics (2010).

Pyne, S., "Improving reconstruction efficiency of time-varying gene regulatory networks", (2020).

Quintana, F. A., P. Müller, A. Jara and S. N. MacEachern, "The dependent dirichlet process and related models", Statistical Science **37**, 1, 24–41 (2022).

Rand, D. A. and L.-S. Young, *Dynamical Systems and Turbulence, Warwick 1980: Proceedings of a Symposium Held at the University of Warwick 1979/80*, vol. 898 (Springer, 2006).

Raser, J., M.Jonathan and E. O'shea, "Noise in gene expression: origins, consequences, and control", Science **309**, 5743, 2010–2013 (2005).

Robinson, J. and A. Hartemink, "Non-stationary dynamic bayesian networks", Advances in neural information processing systems **21** (2008).

Sadeghi, M., B. Ranjbar, M. R. Ganjalikhany, F. M. Khan, U. Schmitz, O. Wolkenhauer and S. K. Gupta, "Microrna and transcription factor gene regulatory network analysis reveals key regulatory elements associated with prostate cancer progression", PloS one **11**, 12, e0168760 (2016).

Saint-Antoine, M. M. and A. Singh, "Network inference in systems biology: recent developments, challenges, and applications", Current opinion in biotechnology **63**, 89–98 (2020).

Sanguinetti, G. and V. A. Huynh-Thu, "Gene regulatory network inference: an introductory survey", in "Gene Regulatory Networks", pp. 1–23 (Springer, 2019).

Santillán, M., "On the use of the hill functions in mathematical models of gene regulatory networks", Mathematical Modelling of Natural Phenomena **3**, 2, 85–97 (2008).

Saunders, L. and D. McClay, "Sub-circuits of a gene regulatory network control a developmental epithelial-mesenchymal transition", Development (Cambridge, England) **141** (2014).

Schulz, E., M. Speekenbrink and A. Krause, "A tutorial on gaussian process regression: Modelling, exploring, and exploiting functions", Journal of Mathematical Psychology **85**, 1–16 (2018).

Sebe-Pedros, A., B. M. Degnan and I. Ruiz-Trillo, "The origin of metazoa: A unicellular perspective", Nature Reviews Genetics **18**, 8, 498 (2017).

Serin, E. A., H. Nijveen, H. W. Hilhorst and W. Ligterink, "Learning from co-expression networks: possibilities and challenges", Frontiers in plant science **7**, 444 (2016).

Sethuraman, J., "A constructive definition of dirichlet priors", Statistica sinica pp. 639–650 (1994).

Shen, X. and H. Vikalo, "Inferring parameters of gene regulatory networks via particle filtering", EURASIP Journal on Advances in Signal Processing **2010**, 1–9 (2010).

Shmulevich, I. and J. Aitchison, "Deterministic and stochastic models of genetic regulatory networks", Methods in Enzymology **467**, 335–356 (2009).

Shmulevich, I., E. R. Dougherty, S. Kim and W. Zhang, "Probabilistic boolean networks: a rule-based uncertainty model for gene regulatory networks", Bioinformatics **18**, 2, 261–274 (2002).

Simon, D., *Optimal state estimation: Kalman, H infinity, and Nonlinear approaches* (John Wiley & Sons, 2006).

Song, L., M. Kolar and E. Xing, "Time-varying dynamic bayesian networks", Advances in neural information processing systems **22** (2009).

Stumpf, M. P., "Inferring better gene regulation networks from single-cell data", Current Opinion in Systems Biology **27**, 100342 (2021).

Sudderth, E., "Conjugate priors & Bayesian learning", Lecture Notes, `http://www.gatsby.ucl.ac.uk/~porbanz/papers/porbanz_BNP_draft.pdf`, [Online; accessed June 11, 2023] (2016).

Sun, J., C. Chen, N. Miyamoto, R. Li, J. D. Sigwart, T. Xu, Y. Sun, W. C. Wong, J. C. Ip, W. Zhang *et al.*, "The scaly-foot snail genome and implications for the origins of biomineralised armour", Nature communications **11**, 1, 1–12 (2020).

Suter, P., J. Kuipers and N. Beerenwinkel, "Discovering gene regulatory networks of multiple phenotypic groups using dynamic bayesian networks", Briefings in Bioinformatics **23**, 4, bbac219 (2022).

Teh, Y., M. Jordan, M. Beal and D. Blei, "Sharing clusters among related groups: Hierarchical dirichlet processes", Advances in neural information processing systems **17** (2004).

Thompson, A., M. R. May, B. R. Moore and A. Kopp, "A hierarchical bayesian mixture model for inferring the expression state of genes in transcriptomes", Proceedings of the National Academy of Sciences **117**, 32, 19339–19346 (2020).

Thorne, T., "Netdiff–bayesian model selection for differential gene regulatory network inference", Scientific Reports **6**, 1, 1–9 (2016).

Thorne, T. and M. P. Stumpf, "Inference of temporally varying bayesian networks", Bioinformatics **28**, 24, 3298–3305 (2012).

Tian, D., Q. Gu and J. Ma, "Identifying gene regulatory network rewiring using latent differential graphical models", Nucleic acids research **44**, 17, e140–e140 (2016).

Tsigos, C. and G. P. Chrousos, "Hypothalamic–pituitary–adrenal axis, neuroendocrine factors and stress", Journal of Psychosomatic Research **53**, 4, 865–871, URL `https://www.sciencedirect.com/science/article/pii/S0022399902004294` (2002).

Tulsyan, A., R. B. Gopaluni and S. R. Khare, "Particle filtering without tears: A primer for beginners", Computers & Chemical Engineering **95**, 130–145 (2016).

Van den Bulcke, T., K. Van Leemput, B. Naudts, P. van Remortel, H. Ma, A. Verschoren, B. De Moor and K. Marchal, "Syntren: A generator of synthetic gene expression data for design and analysis of structure learning algorithms", BMC Bioinformatics **7**, 1, 43 (2006).

Van Der Wijst, M. G., D. H. de Vries, H. Brugge, H.-J. Westra and L. Franke, "An integrative approach for building personalized gene regulatory networks for precision medicine", Genome medicine **10**, 1, 1–15 (2018).

Vélez-Cruz, N., B. Moraffah and A. Papandreou-Suppappola, "Sequential bayesian inference using stochastic models of gene regulatory networks", in "2021 55th Asilomar Conference on Signals, Systems, and Computers", pp. 568–572 (IEEE, 2021).

Wai, H.-T., A. Scaglione, B. Barzel and A. Leshem, "Joint network topology and dynamics recovery from perturbed stationary points", IEEE Transactions on Signal Processing **67**, 17, 4582–4596 (2019).

Wan, E. A., R. Van Der Merwe and S. Haykin, "The unscented kalman filter", Kalman filtering and neural networks **5**, 2007, 221–280 (2001).

Wang, H. and D. Aberra, "Noise analysis of gene regulatory networks using particle filter", International Journal of Clinical Biostatistics and Biometrics pp. 1–10 (2015).

Wang, H. and D. Aberra, "Noisy nonlinear gene regulatory networks analysis using ensemble kalman filter based particle filter without a model", in "Proceedings of the 7th International Conference on Computational Systems-Biology and Bioinformatics", pp. 8–13 (2016).

Wang, H., L. Qian and E. Dougherty, "Inference of gene regulatory networks using s-system: A unified approach", in "2007 IEEE Symposium on Computational Intelligence and Bioinformatics and Computational Biology", pp. 82–89 (2007).

Wang, L. and X. Wang, "Hierarchical dirichlet process model for gene expression clustering", EURASIP Journal on Bioinformatics and Systems Biology **2013**, 1, 1–14 (2013).

Wang, Z., X. Liu, Y. Liu, J. Liang and V. Vinciotti, "An extended kalman filtering approach to modeling nonlinear dynamic gene regulatory networks via short gene expression time series", IEEE/ACM Transactions on Computational Biology and Bioinformatics **6**, 3, 410–419 (2009).

Werhli, A. V. and D. Husmeier, "Reconstructing gene regulatory networks with bayesian networks by combining expression data with multiple sources of prior knowledge", Statistical applications in genetics and molecular biology **6**, 1 (2007).

Xiao, Y., "A tutorial on analysis and simulation of boolean gene regulatory network models", Current genomics **10**, 7, 511–525 (2009).

Xiong, J. and T. Zhou, "A kalman-filter based approach to identification of time-varying gene regulatory networks", PLOS ONE **8**, 10, 1–8, URL `https://doi.org/10.1371/journal.pone.0074571` (2013).

Xu, T., L. Ou-Yang, X. Hu and X.-F. Zhang, "Identifying gene network rewiring by integrating gene expression and gene network data", IEEE/ACM Transactions on Computational Biology and Bioinformatics **15**, 6, 2079–2085 (2018).

Yang, J. and J. Peng, "Estimating time-varying graphical models", Journal of Computational and Graphical Statistics **29**, 1, 191–202 (2020).

Youseph, A., M. Chetty and G. Karmakar, "Gene regulatory network inference using michaelis-menten kinetics", in "2015 IEEE Congress on Evolutionary Computation (CEC)", pp. 2392–2397 (2015).

Youseph, A., M. Chetty and G. Karmakar, "Reverse engineering genetic networks using nonlinear saturation kinetics", Biosystems **182**, 30–41, URL `https://www.sciencedirect.com/science/article/pii/S0303264718303976` (2019).

Zhang, Y., Y. Pu, H. Zhang, Y. Cong and J. Zhou, "An extended fractional kalman filter for inferring gene regulatory networks using time-series data", Chemometrics and Intelligent Laboratory Systems **138**, 57–63, URL `https://www.sciencedirect.com/science/article/pii/S0169743914001543` (2014).

Zhao, H. and Z.-H. Duan, "Cancer genetic network inference using gaussian graphical models", Bioinformatics and biology insights **13**, 1177932219839402 (2019).

Zhong, C., Z. Ma, J. Shen and C. Liu, "Dependent dirichlet processes for analysis of a generalized shared frailty model", in "Computational Statistics and Applications", (IntechOpen, 2021).

Zhou, Z. and R. Ji, "Modeling gene regulatory networks based on nonlinear state-space model", in "36th Chinese Control Conference", pp. 10056–10061 (2017).