The Application of the Expectation-Maximization Algorithm to the Identification of Biological Models

Shuo Chen

Thesis submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Masters
in
Electrical Engineering

William T. Baumann, Chair
Yue Wang
Jason J. Xuan

December 11, 2006
Arlington, Virginia

Keywords: EM Algorithm, Gene Regulatory Networks

The Application of the Expectation-Maximization Algorithm to the Identification of Biological Models

Shuo Chen

(ABSTRACT)

With the onset of large-scale gene expression profiling, many researchers have turned their attention toward biological process modeling and system identification. The abundance of data available, while inspiring, is also daunting to interpret. Following the initial work of Rangel *et al.*, we propose a linear model for identifying the biological model behind the data and utilize a modification of the Expectation-Maximization algorithm for training it. With our model, we explore some commonly accepted assumptions concerning sampling, discretization, and state transformations. Also, we illuminate the model complexities and interpretation difficulties caused by unknown state transformations and propose some solutions for resolving these problems. Finally, we elucidate the advantages and limitations of our linear state-space model with simulated data from several nonlinear networks.

Acknowledgments

Special thanks to Dr. Bill Baumann for his invaluable direction during my graduate education. I sincerely appreciate all the times he has voiced his encouragements, assurances and research suggestions. Thank you for offering me the opportunity to work with you and in this field!

Also, thank you Mom, my staunchest supporter, for your unwavering support and enthusiasm throughout my undergraduate and graduate studies. You have always been and always will be my role model.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The advent of DNA microarray technology has introduced an abundance of data to the field of functional genomics in the form of large-scale gene expression profiles. A single profile can contain the expression level of thousands of genes by measuring their respective mRNA concentrations simultaneously. Each profile offers a snapshot of the state within a cell at an instant in time. Strung together, gene profiles at consecutive time instances can offer a glimpse of the signal pathways and gene activities that determine a cell's response to different physiological events. This tempting wealth of data has turned the scientific community toward this exact purpose, the identification of cellular processes or the mathematical mapping of gene regulatory networks (GRNs).

The ultimate goal of this research is to depict a fully reverse engineered gene regulatory network as a mathematical model that can effectively predict the behavior of a cell to specific physical circumstances. This knowledge would illuminate the chemistry behind the cell cycle, human growth and development, pathogenesis and many other intricate biological processes. In particular, it can be invaluable to devising medical responses to diseases like muscular dystrophy, cancer, diabetes etc.

Despite gaining steady momentum and making rapid progress, this scientific movement is still in its fledgling stages. While the available data contains a wealth of information about the chemical response pathways, much of it is buried under measurement noise, experiment specificity, and naturally cross-functional inter-

plays. Although many networks have been proposed, none have emerged particularly successful at fully mapping the complicated chemical pathways behind cellular processes. A few of the most popular proposed mathematical model forms include dynamic Bayesian networks (DBN), boolean networks, neural networks, gene clusters, and various differential equation forms.

1.1 Popular Mathematical Models

The purpose of this section is to describe a few of the popular variations on proposed mathematical forms as well as their pros and cons. We hope to offer the reader an idea of the current state of model identification and parameter estimation in genomics. We begin with one of the oldest proposed models, boolean networks.

1.1.1 Boolean Networks

Kauffman [16] first proposed the application of boolean networks to genetic modeling in 1969. The basic boolean network he described comprised N randomly connected binary nodes that represented genes in the genetic net. All connections stemmed and ended within the network; further, because all connections were directed, those ending in a node represented that node's input while those stemming from a node carried that node's current state as input to other nodes. The binary value of each node was governed by a boolean function acting on the current inputs of that node. A binary state of zero meant the gene was repressed while a binary state of one meant it was activated. The entire system was synchronous and deterministic, with at most 2^N possible combinations of node values or states. Because of the finite number of unique states, this system always eventually cycles back to a previous state. Finally, given a certain initial condition, the system always simulates the same dynamics.

In 1998, Liang *et al.*[18] used mutual information to infer the input connections of a boolean network from time series gene data. By systematically considering combinations of input lines and resulting gene output, he deduced the input set that maximized the mutual information between input and output. Akutsu *et al.*[1] formalized a proof of Liang's observation that a small number of input/output pairs are sufficient to surmise a boolean network. Further, he introduced a simpler algorithm along the same principles that employed an exhaustive search to identify the boolean function governing each node. He found that given a limiting up-

perbound on the inputs, K, a network identification problem could be solved in polynomial time and derived the minimum and maximum number of input/output pairs necessary to identify a network. In 2000, Ideker *et al.*[15] used steady-state gene profile data generated from a series of genetic and biological perturbations to infer boolean networks. He introduced two methods, the *predictor* and the *chooser* method, for the identification process. The first inferred all possible networks consistent with the profile and returned those with the fewest number of interactions; the latter proposed an additional perturbation that could differentiate the true network from the inferred set and used an entropy function to deduce the optimal network after the perturbation was added. Ideker used the two methods interactively and interatively to achieve his final solution. All models described thus far have been binary. Kyoda *et al.*[17] suggested a model that accounted for steady-state gene expression values via weights on the edges connecting gene nodes. Also, his algorithm returned the most parsimonious network found by eliminating all redundant routes from the network.

Boolean networks are intuitive and simple to work with but still capable of modeling nonlinear relations in regulatory networks. Nevertheless, there are still several fundamental drawbacks to its usage. First is its assumption of binary representation and variable criterion for "on" and "off" states. The criteria chosen can dramatically affect the final connectivity of the model and can result in loss of information as well as erroneous inference. Second is its idealization of a natural network as a synchronous state-machine when in actuality, a true network may be stochastic and asynchronous. Lastly, the designation of a finite number of states and discretization is a persistent problem in modeling a naturally continuous system with an unknown number of states [3].

1.1.2 Clustering Analysis

Often, gene expression patterns are hidden under a formidable amount of numerical data and measurement noise. Efficient analysis and identification of these patterns can prove virtually impossible without some organized method of data classification. Clustering analysis is one such popular classification technique [9]. Aggregative hierarchical clustering as originally proposed by Sneath and Sokal [25] in 1973 remains a favored choice in clustering applications. The original variant of this technique, called *average linkage*, presents data in the form of a binary tree. The algorithm iteratively links the closest pair of genes or gene groups until only one element remains. The definition of "closest" is based on a chosen metric characteristic to all the

gene nodes; the algorithm compares the average metric characteristic for gene groups [9].

Eisen *et al.*[11] were among the first to apply *average linkage* clustering to yeast and human gene data with promising results. Similarly, in 1999, Alon *et al.*[2] introduced a hierarchical cluster variant that separated unconnected nodes into two clusters iteratively based on their distribution along two fitted Gaussian distributions. The centroids of the Gaussian distributions were designated as new tiers of the binary tree at each iteration until only one centroid remained. His two-way algorithm was capable of both partitioning genes into functional groups and categorizing tissues based on gene expression. Specifically, his analysis showed that clustering grouped genes with similar functions across several tissue types and separated tumor and normal tissues. Another popular variant of clustering is K-means clustering, which Travazoie *et al.*[27] first applied to yeast gene expression in 1999. In K-means clustering, N genes are distributed into K predetermined clusters based on their proximity to the designated centroids of these clusters. At each iteration of the algorithm, the centroids are recalculated as the average over all the genes in the cluster and the genes redistributed based on the new centroids. The algorithm iterates until either no more changes occur in the distributions or until a predefined iteration limit has been reached [7]. Other variants on clustering techniques have been proposed since then; among them are the Self-Organized Map (SOM method) [26], fuzzy C-means [10], and clustering via mutual information [34].

Cluster analysis is useful when working with enormous amounts of data. It can classify thousands of genes into coherent groups with similar temporal expression patterns. But despite its efficiency in dealing with copious data, cluster analysis at best provides only a qualitative network that is hard to analyze mathematically. Specifically, there is no way of measuring the validity of the clusters [3]. Furthermore, clusters may not always correspond to actual biological relations and terms. Hence, cluster analysis may be most advantageous when used in conjunction with another modeling technique.

1.1.3 Linear Networks

A number of linear network models are popular in functional genomics. Some models simulate genetic networks in continuous time via sets of differential equations while others function in discrete time via sets of difference equations. Despite differing inference methodologies, all these networks follow the same principle

structure, where all gene regulations are represented as linear functions of other gene concentrations. For example, consider

$$g_i[k+1] = \sum_{j \in J} w_{ij} g_j[k] + b_i \text{ (difference equation)}$$
$$\frac{\partial g_i(t)}{\partial t} = \sum_{j \in J} w_{ij} g_j(t) + b_i \text{ (differential equation)}$$

where g_i is a certain gene i, J the set of gene inputs affecting gene i, w_{ij} the importance or weight of input gene j on gene i, and b_j is a constant representing the natural steady state [3].

One of the earlier models was proposed by Weaver *et al.*[31] in 1999. His discrete time model simulated regulation input r_i as

$$r_i(t) = \sum_{j \in J} w_{ij} u_j(t)$$

similar to the difference equation above, with g_j replaced by u_j. However, Weaver's model also mathematically defined the effect of this regulation input on the current gene concentration via a "squashing" function

$$u_i(t+1) = \frac{m_i}{1 + e^{-(\alpha_i r_i(t) + \beta_i)}}$$

where constants α_i and β_i control the shape of the "squashing" and m_i is the real expression maximum for gene i. By structuring the inputs and genes as vectors, he reshaped the weights on the input genes into a matrix so that he could use linear algebra principles to solve for his coefficients. From simulations and analysis, Weaver concluded that this model could predict very accurately despite measurement noise [31].

In the same year that Weaver *et al.*introduced their model, a similar continuous time model was produced by Chen *et al.*[5] in the form of

$$\frac{\partial x}{\partial t} = \begin{bmatrix} -V & C \\ L & -U \end{bmatrix} x$$

where $x = [r, p]^T$ and r represented measured mRNA concentrations, p measured protein concentrations, L translational constants, V mRNA degradation rates, U protein degradation rates, and C first order approximation of mRNA transcription rates. Chen used minimum weight solutions and Fourier transformations

along with knowledge of the cyclic behavior of cells to solve for the coefficient matrix.

Lastly, D'Haeseleer *et al.*[8] presented a discrete, additive, linear model and tested it on a compilation of mRNA data collected from rat hippocampus tissue and cervical spinal cord tissue. The first set of data was collected during natural development of the rat hippocampus; the latter was collected after kainate, "a glutamatergic agonist which causes seizures, localized cell death, and severely disrupts the normal gene expression patterns," was injected into the rat's central nervous system. The model he tested was

$$x_i[t + \delta t] = \sum_j w_{ij} x_j(t) + K_i \cdot kainate(t) + C_i + T_i$$

where $kainate(t)$ represents the current kainate receptor concentration, K_i its weight on gene i, C_i a constant bias factor, T_i a differentiating bias factor across tissue types. During testing, D'Haeseleer used a nonlinear interpolation scheme to ensure enough linearly independent constraints were available to solve the system in a least squares regression.

The linear models described thus far offer only a brief introduction to the linear network variants that have been proposed. Some others that have been proposed since incorporate singular value decomposition, principle component analysis and independent component analysis to improve fitting and prediction [3]. Though popular, linear models can only offer an approximation of the intrinsically nonlinear biological system. A linear model can never exactly reproduce the biological network it is trained on. For this reason, when coefficients are estimated, they remain difficult to interpret because they may be true coefficient values or they may be an aggregate of coefficients from unmodelled nonlinearities. Nevertheless, linear networks still achieve a good enough estimate of the nonlinear gene network as to afford very good predictions of gene regulatory patterns.

1.1.4 Nonlinear Networks

Although complicated, nonlinear networks remain a popular modeling method because genetic networks are known to be intrinsically nonlinear. Nonlinear networks have no principle underlying structure as linear networks do, hence models come in every shape and form in this category. We introduce two commonly seen

nonlinear models in this section, the S-system and the neural network, but a number of other models exist.

S-systems are a variant on the Biochemical Systems Theory (BST) framework. Given n dependent variables and m independent variables, an S-system models the dynamics of the system via change in the dependent variables as nonlinear differential equations. These equations consist of a growth term and a decay term summed together. Each growth or decay term is a product of power functions containing all variables that influence the dependent variable of interest. For example,

$$\dot{x}_i = \alpha_i \prod_{j=1}^{n+m} x_j^{g_{ij}} - \beta_i \prod_{k=1}^{n+m} x_k^{h_{ik}}$$

is a generic S-system differential equation, in which α_i and β_i are rate constants on the growth and decay terms, g_{ij} and h_{ik} are kinetic orders on the power functions, x_j are elements of the gene set that enlarge gene x_i, and x_k are elements of the gene set that diminish x_i [28]. Given enough data points and steady states, the system can be simplified to a set of linear algebraic equations via a logarithmic transformation and then solved for all coefficients via simple algebra.

A neural network has a completely different structure from an S-system. In the neural network model proposed by Wahde *et al.*[29] in 2001, N gene mRNAs were separated into four clusters which interact with each other as units and within which the temporal gene expression data is very similar. Each cluster has six parameters to be estimated:

1. a time constant determining the rate of the cluster's response to other gene concentration changes, τ_i
2. four constants corresponding to the effect of cluster expression levels on transcription rate; one for each cluster, including itself, w_{ij}
3. a constant representing the natural transcription rate of the cluster devoid of regulation, b_i

A generic differential equation for a cluster incorporating these constants is

$$\tau_i \frac{\partial S_i}{\partial t} = -S_i(t) + g\left(b_i + \sum_j w_{ij} S_j(t)\right)$$

where the cluster is represented by the average mRNA concentration in it, S_i. The function $g(x)$ is a nonlinear sigmoid function that is added to ensure the expression level of the cluster never exceeds a maximum limit or drops below zero. In this instance, this sigmoid function was designated as

$$g(x) = (1 + e^{-x})^{-1}$$

so that each cluster's expression level was normalized by its maximum value [29]. To estimate the unknown parameters, Wahde employed a genetic algorithm and the same data set used by D'Haeseleer *et al.*

Detailed nonlinear networks have the potential to fully map a gene regulatory network. Through a compilation of complex functions, nonlinear networks can expand their parameter space to encompass the set of parameters defining the gene network. However, due to the enlarged parameter space, nonlinear networks often require significant amounts of data to define a unique model. Current microarray methods can observe an enormous breadth of genes but we require many samples temporally to identify a true gene network. Due to data limitations, many researchers use nonlinear networks in combination with statistical or optimization techniques. For example, Ressom *et al.*[23] combined recurrent neural networks with ant colony optimization and particle swarm intelligence, two machine learning methods, to infer a gene regulatory network.

1.1.5 Statisical Networks

There are a multitude of reasons to model a gene expression network as a stochastic system with some uncertainty. The first and foremost is the common belief that gene expression models are inherently stochastic. Another is the fact that even if the physical network is deterministic, our methods of observation are still so primitive as to leave much of the physical network unobserved. And as such, the physical model may appear stochastic even if it isn't and it becomes necessary to incorporate some unknown hidden variables in our simulation model to account for hidden dynamics. Lastly, our measurements contain so much noise that hinders almost all model fitting techniques discussed thus far because all previously mentioned models have been deterministic [19].

Thus, in this section, we discuss the Dynamic Bayesian Network (DBN) as introduced by Murphy and

Mian [19] in 1999. Bayesian Networks are a special case of graphical models in which nodes designate random variables and directed arcs designate conditional dependencies between nodes. The entire system is a directed acyclic graph (DAG) that is parameterized as a joint distribution over all the random variables. A Bayesian network is classified as dynamic when it characterizes how a random variable or a set of random variables changes over time. Further, in Dynamic Bayesian Networks, the first-order Markov property applies, holding that a future state, given the present state, is independent of all past states. Thus, in the graphical representation of a dynamic Bayesian network, each node represents the state of a set of random variables at an instant in time and each node is dependent only on its parent node or nodes, the time state or states directly before it, and independent of all indirect ancestor nodes. A DBN can be represented in mathematical notation as a product of conditional probabilities

$$P(X_1, X_2, \ldots, X_T) = \prod_{i=1}^{T} P(X_i | X_{i-1})$$

where X_i represents the state of the random variable set at time i, X_{i-1} the previous time state X_i is conditionally dependent on, and T the total number of time states in the graph. A number of algorithms exist for learning a Dynamic Bayesian Network. The most common approach is to introduce a statistical objective function over the network in question and optimize this function with respect to the parameters in the system [20]. But some consideration must also be given to the structure, how the random states relate to each other within a time slice as well as how they relate across time slices, and observability, whether all random variables can be measured or not, of the Bayesian network before choosing a set learning method. The simplest parameter estimation case is one in which the model structure is known and all its random variables observable; in such a case, the objective function can incorporate the structure of the network, spanning all time instances, and be optimized using statistical sample analysis. However, in the case where the model structure is unknown but all its random variables are observable, a search algorithm over model space must be implemented. In the instances when a model structure is known but only some of the random variables are observable then iterative techniques like the Expectation-Maximization or the gradient ascent algorithm should be used. Finally, in the case where the structure is unknown and the random variables are only partially observed, the model space must be searched while the objective function is iteratively optimized as well so a technique like structural Expectation-Maximization should be used [19]. In general, the statistical objective function designates the temporal and spatial structure of the network while sample observations and search algorithms optimize the structure parameters. In instances when the structure is

unknown as well, a search algorithm must alternately optimize the objective function structure and the network parameters until the algorithm converges on an optimum.

Dynamic Bayesian Networks are probabilistic in nature so although a designated structure may not match the biological network, the structure can still reasonably approximate the network due to the presence of random variables. However because most identification techniques for this type of network optimize about a stochastic objective function of conditional probabilities, the accuracy of the network is very much dependent on the structure of the objective function. A poorly structured scoring function can lead to a poorly parameterized model. Further, if this objective function is nonlinear, the algorithm may converge erroneously on a local maximum that does not necessarily optimize the model closer to the real network. Lastly, the identified model may be hard to interpret because of the presence of stochastic variables.

1.2 A Special Case of Dynamic Bayesian Networks

Several of the models introduced in previous sections have been variants of Dynamic Bayesian Networks (DBNs). Among them are boolean network models, D'Haeseleer's linear, additive model, and Weaver's linear, weighted model [19]. In 2004, Rangel *et al.*[21] made use of another subclass of Dynamic Bayesian Networks: the Linear Dynamical System or the Linear-Gaussian State-Space Model with the form

$$x_{k+1} = Ax_k + Bu_k + w_k$$
$$y_k = Cx_k + Du_k + v_k$$

where w_k and v_k are Gaussian variables. This system is similar to the linear difference equations discussed in section 1.1.3. Its variation lies in the interpretation of the state vector x and the observation vector y as well as the addition of the random variables w and v. The structure parameter set to be estimated consists of the system coefficients, A, B, C, D, the covariance matrices of w_k and v_k, Q and R, respectively, and the initial state vector, π_1 and its covariance, V_1. An asset of this model structure is its variable number of states, which can be changed to track the hidden or unobserved variables of genetic networks. Furthermore, these additional states also help the linear model better track simple nonlinear models. Lastly, the estimated covariance matrix of the random variable w_k, which typically represents state noise, can be read as a measure of the degree of mismatch in the observed network.

Rangel *et al.*used the Expectation-Maximization (EM) algorithm to train the parameter set of this model. The EM algorithm, which was already briefly mentioned in the previous section as a favored optimization, search algorithm, is a well-known, maximum likelihood, optimization technique that has been used for a number of biological research ventures. In 2001, Husmeier and Wright used the EM to optimize a model that detected recombination events in DNA sequences [14]. In 2002, Friedman *et al.*used a variant of the EM algorithm to reconstruct a maximum likelihood phylogenetic tree from current day taxa [12]. In 2003, the EM algorithm was used to extract identification variables of translation initiation sites in eukaryotic genes [30]. These are just a few examples of EM implementation in research but they are sufficient to show its flexibility and robustness.

1.3 Our Thesis Objective

The objective of this thesis is to define a plausible Linear Dynamical System structure, train it using realistic, gene network simulations and the EM algorithm, and identify valid network pathways from the estimated structural parameters.

The contributions of this thesis are organized as follows. In Chapter 2, we detail a derivation of the EM algorithm as pertaining to a generalized state-space structure with inputs and Gaussian noise variables. In Chapter 3, we briefly elucidate the effect of basic state transformations on general state-space identification and then test variants of the state-space model and EM algorithm with some simple linear case studies. In this chapter, we also clarify the effects of discretization, sampling, and simple state transformations on system identification. In Chapter 4, we formally define our system model and address in detail the complications arising from state transformations. Specifically, we focus on identifiability and interpretation of the state-space model. In Chapter 5, we evaluate the usefulness of our model for identifying the structure of three nonlinear systems. We begin with a simple but abstract nonlinear case with five states; then, we move on to a more complex, feedthrough system; and finally, we end with a detailed, simulated system that models physiological cell processes. In the final instance, we test and evaluate our solution to state transformations. We will conclude with a discussion on the success of our venture and future work.

Chapter 2

The Expectation-Maximization Algorithm

2.1 Introduction to the Expectation-Maximization Algorithm

The Expectation-Maximization (EM) Algorithm, introduced by Dempster *et al.*[6] in 1977, is an iterative process that computes Maximum Likelihood (ML) estimates. Its purpose is to estimate system parameters that would maximize the occurrence probability of observed data. A key feature of the EM algorithm however is its optimized likelihood cost function that's defined in terms of both observed and unobserved (hidden) data. The hidden data may be incorporated because there actually exists missing data in the practical application or because they are necessary for computation of the likelihood function. In this thesis, because the EM algorithm is used to identify the dynamic Bayesian network underlying a gene transcription network, the hidden data models kinetics that are unobserved directly in gene expression. The EM algorithm has a framework that naturally makes use of the hidden data, making it particularly viable for our purpose. Its procedure is comprised of two alternating steps: the Expectation step and the Maximization step. The E-step computes the best estimate of the likelihood function using the current knowledge of the ML estimates and the observed data. The M-step re-estimates the parameters to maximize the new likelihood function. The following derivation of its general form follows closely from that presented in [4]. We present it here because we could find no derivation with all the details in the literature.

The log likelihood function is defined as

$$L(\theta) = \ln P(X|\theta) \tag{2.1}$$

where θ denotes the parameters to be estimated and X denotes the observed data set. The objective of ML estimation is to find the θ that maximizes the probability of X, or $P(X|\theta)$. Because the natural logarithm is a monotonically increasing function, maximizing $L(\theta)$ will also maximize $P(X|\theta)$. The EM algorithm iteratively computes θ such that each consecutive θ estimate either remains the same as or improves on the previous estimate.

$$L(\theta_{n+1}) \geq L(\theta_n)$$

In most cases the algorithm will converge to a local maxima. The process of maximizing $L(\theta)$ is the same as maximizing the difference in the log likelihood between iterations,

$$L(\theta) - L(\theta_n) = \ln P(X|\theta) - \ln P(X|\theta_n) \tag{2.2}$$

Suppose we are able to more easily calculate the likelihood of the observed data X and some hidden data z, $P(X, z|\theta)$. Then the marginal distribution $P(X|\theta)$ is given by

$$P(X|\theta) = \sum_z P(X, z|\theta) = \sum_z P(X|z, \theta)P(z|\theta) \tag{2.3}$$

Incorporating (2.3) into (2.2) yields

$$L(\theta) - L(\theta_n) = \ln \left[\sum_z P(X|z, \theta)P(z|\theta) \right] - \ln P(X|\theta_n) \tag{2.4}$$

Now consider Jensen's inequality, which states for any random variable, x, and any convex function $f(x)$,

$$E[f(x)] \geq f[E(x)] \tag{2.5}$$

For concave functions, this inequality is reversed to get

$$E[f(x)] \leq f[E(x)] \tag{2.6}$$

We can tailor it to our purposes by defining $f(x)$ as $\ln(x)$, which is a concave function, to get

$$\sum_{i=1}^{n} \lambda_i \ln(x_i) \leq \ln \sum_{i=1}^{n} \lambda_i x_i \tag{2.7}$$

where λ_i are constants greater than zero that sum to 1, ie. $\sum_{i=1}^{n} \lambda_i = 1$. In (2.7), we specify the expectations in (2.6) as a summation of weighted terms, where the weights, denoted by λ_i, are representative of the probabilities of their matched x_i value. We can rearrange (2.4) to derive a lowerbound on the new log likelihood calculated from the new ML estimate. In (2.4), we can introduce constants of the form $P(z|X, \theta_n)$, which are guaranteed to sum to 1 and be greater than 0 so that

$$\begin{aligned}
\ln \sum_z P(X|z,\theta)P(z|\theta) - \ln P(X|\theta_n) &= \ln \sum_z P(X|z,\theta)P(z|\theta) \cdot \frac{P(z|X,\theta_n)}{P(z|X,\theta_n)} - \ln P(X|\theta_n) \\
&= \ln \sum_z P(z|X,\theta_n) \left(\frac{P(X|z,\theta)P(z|\theta)}{P(z|X,\theta_n)} \right) - \ln P(X|\theta_n)
\end{aligned}$$

Then by Jensen's inequality the natural logarithm can be pulled within the summation to yield a lowerbound by equating $P(z|X, \theta_n)$ to λ_i and $\left(\frac{P(X|z,\theta)P(z|\theta)}{P(z|X,\theta_n)}\right)$ to x_i. This gives

$$\begin{aligned}
L(\theta) - L(\theta_n) &\geq \sum_z P(z|X,\theta_n) \ln \left(\frac{P(X|z,\theta)P(z|\theta)}{P(z|X,\theta_n)} \right) - \ln P(X|\theta_n) \\
&= \sum_z P(z|X,\theta_n) \ln \left(\frac{P(X|z,\theta)P(z|\theta)}{P(z|X,\theta_n)P(X|\theta_n)} \right) \\
&= \sum_z P(z|X,\theta_n) \ln \left(\frac{P(X,z|\theta)}{P(X,z|\theta_n)} \right) \\
&\equiv \Delta(\theta|\theta_n)
\end{aligned}$$

Moving $L(\theta_n)$ to the other side, we get

$$L(\theta) \geq L(\theta_n) + \Delta(\theta|\theta_n)$$

Furthermore, we observe that

$$\begin{aligned}\Delta(\theta_n|\theta_n) &= \sum_z P(z|X,\theta_n)\ln\left(\frac{P(X,z|\theta_n)}{P(X,z|\theta_n)}\right)\\ &= \sum_z P(z|X,\theta_n)\ln 1\\ &= 0 \end{aligned} \tag{2.8}$$

Thus, $L(\theta_n) + \Delta(\theta|\theta_n)$ is equivalent to $L(\theta)$ at θ_n and is also its lowerbound everywhere else. Increasing $\Delta(\theta|\theta_n)$ would hence be guaranteed to improve $L(\theta)$; consequently, the EM algorithm maximizes $\Delta(\theta|\theta_n)$ at each iteration in order to improve $L(\theta)$. Restated, the next θ-update can be computed via

$$\begin{aligned}\theta_{n+1} &= \arg\max_\theta\{\Delta(\theta|\theta_n)\}\\ &= \arg\max_\theta\left\{\sum_z P(z|X,\theta_n)\ln\left(\frac{P(X,z|\theta)}{P(X,z|\theta_n)}\right)\right\}\end{aligned}$$

The maximum operation ensures $\Delta(\theta|\theta_n) \geq 0$ so that $L(\theta)$ always increases or at least stays the same. The constant terms that do not affect the maximization can be dropped to yield

$$\begin{aligned}\theta_{n+1} &= \arg\max_\theta\left\{\sum_z P(z|X,\theta_n)\ln\big(P(X,z|\theta)\big)\right\}\\ &= \arg\max_\theta\left\{E_{z|X,\theta_n}\Big[\ln P(X,z|\theta)\Big]\right\}\end{aligned} \tag{2.9}$$

Equation (2.9) gives the objective function that replaces the actual log likelihood equation and is maximized in the EM algorithm. Thus, each iterate of EM maximizes $E_{z|X,\theta_n}\left[\ln P(X,z|\theta)\right]$, which acts as a lowerbound to the actual log likelihood function. Because each iteration starts with the previous iterate's estimates, maximization of the objective value will never decrease the log likelihood and can only improve it.

2.2 Maximum Likelihood Estimation of State-Space Model

There are several published articles that contain the derivation of the Expectation Maximization Algorithm for special cases; however, for our purposes, we desired a general derivation that could be simplified to all

cases. We consider state-space models of the form

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k + w_k \\ y_k &= Cx_k + Du_k + v_k \end{aligned} \tag{2.10}$$

where w_k and v_k are zero-mean Gaussian variables with covariance matrices Q and R respectively such that,

$$P(w_k) = \frac{1}{(2\pi)^{\frac{n}{2}}|Q|^{\frac{1}{2}}} e^{-\frac{1}{2}w_k^T Q^{-1} w_k} \tag{2.11}$$

$$P(v_k) = \frac{1}{(2\pi)^{\frac{p}{2}}|R|^{\frac{1}{2}}} e^{-\frac{1}{2}v_k^T R^{-1} v_k} \tag{2.12}$$

where p denotes the number of observed data states and n denotes the number of hidden data states. Our goal is to estimate the parameters A, B, C, D, Q, R, π_1, and V_1 in a maximum likelihood sense. For our purposes, the outputs, y_k, represent the observed data in our model and the state variables, x_k, represent the hidden, unobserved kinetics in our model. Ideally, our model would optimize the occurrence probability of both these quantities; thus, our state-space structure is aptly formulated for EM optimization, which incorporates both in its framework.

Recall from (2.1) that the true log likelihood function only takes into account the observed data set, $\{y_T, y_{T-1}, \ldots, y_1\}$. Let Y_k denote the set of observations up to k, then (2.1) is

$$L(\theta) = \ln P(Y_T|\theta) \tag{2.13}$$

Using Bayes' theorem, $P(x, y) = P(x|y)P(y)$, this joint probability can be rewritten as a product of conditional probabilities

$$L(\theta) = \ln \prod_{k=1}^{T} P(y_k|Y_{k-1}, \theta) \tag{2.14}$$

Because state-space models are Markov processes, the Markov property allows further simplification. The Markov property states that given the state of a process at all time steps up to the previous time step, the

conditional probability distribution of the current process state is dependent only on the previous state.

$$L(\theta) = \ln \prod_{k=1}^{T} P(y_k|y_{k-1}, \theta) \tag{2.15}$$

We use the notation

$$\hat{x}_k = E[x_k|Y_T, \theta] \tag{2.16}$$

$$\hat{x}_{k|k-1} = E[x_k|Y_{k-1}, \theta] \tag{2.17}$$

$$V_k = E[x_k x_k^T|Y_T, \theta] - \hat{x}_k \hat{x}_k^T \tag{2.18}$$

$$V_{k|k-1} = E[x_k x_k^T|Y_{k-1}, \theta] - \hat{x}_{k|k-1} \hat{x}_{k|k-1}^T \tag{2.19}$$

Then, from (2.10), the conditional probabilities of y_k are normally distributed with mean $C\hat{x}_{k|k-1} + Du_k$ and covariance $CV_{k|k-1}C^T + R$. Equation (2.15) becomes

$$L(\theta) = \ln \prod_{k=1}^{T} \frac{1}{(2\pi)^{\frac{p}{2}} |CV_{k|k-1}C^T + R|^{\frac{1}{2}}} e^{-\frac{1}{2}(y_k - C\hat{x}_{k|k-1} - Du_k)^T (CV_{k|k-1}C^T + R)^{-1} (y_k - C\hat{x}_{k|k-1} - Du_k)} \tag{2.20}$$

Note that for the initial case, $k = 1$, $x_{k|k-1}$ is π_1 and $V_{k|k-1}$ is V_1. Equation (2.20) is the true log likelihood. Observe that it does not explicitly contain state-space coefficients A or B. Thus, while it is capable of monitoring the run-time progress of the EM algorithm, it is not readily applicable to the derivation of all the EM maximization equations. So we progress to the previously defined EM objective function in (2.9); again, it can be considered a product of joint probability functions inclusive of the hidden and observed data, y_k and x_k. Let Y_k and X_k denote all observed and hidden data, respectively, up to time sample k, then

$$\begin{aligned} E_{X_T|Y_T,\theta}\Big[\ln P(Y_T, X_T|\theta)\Big] &= E_{X_T|Y_T,\theta_n}\Big[\ln \prod_{k=2}^{T} P(y_k|Y_{k-1}, X_T, \theta) P(y_1|X_T, \theta) P(X_T|\theta)\Big] \\ &= E_{X_T|Y_T,\theta}\Big[\ln \prod_{k=2}^{T} P(y_k|Y_{k-1}, X_T, \theta) P(y_1|X_T, \theta) \prod_{k=2}^{T} P(x_k|X_{k-1}, \theta) P(x_1|\theta)\Big] \end{aligned}$$

We apply the Markov property such that each successive x_k state is dependent only on x_{k-1}, the state at the previous time. Also, because y_k is independent of all other observations given x_T, we can eliminate the dependence on Y_{k-1} from all conditional y_k probabilities. Furthermore, applying the Markov property again,

each y_k is dependent only on the current state, x_k.

$$E_{X_T|Y_T,\theta}\Big[\ln P(Y_T, X_T|\theta)\Big] = E_{X_T|Y_T,\theta}\Big[\ln \prod_{k=1}^{T} P(y_k|x_k,\theta) \prod_{k=2}^{T} P(x_k|x_{k-1},\theta)P(x_1|\theta)\Big]$$

Or if we have multiple data sets from the same model, the probability of all observation sets must be maximized. Assuming that all experiments are independent of each other, the log likelihood would be the product of all the observation probabilities.

$$\begin{aligned} E_{X_T|Y_T,\theta}\Big[\ln P(Y_T, X_T|\theta)\Big] &= E_{X_T|Y_T,\theta}\left[\ln\left(\prod_{j=1}^{N}\left\{\prod_{k=1}^{T} P(y_{jk}|x_{jk},\theta) \times \prod_{k=2}^{T} P(x_{jk}|x_{j(k-1)},\theta) \times P(x_{j1}|\theta)\right\}\right)\right] \\ &= E_{X_T|Y_T,\theta}\left[\sum_{j=1}^{N}\left\{\sum_{k=1}^{T}\ln P(y_{jk}|x_{jk},\theta) + \sum_{i=2}^{T}\ln P(x_{jk}|x_{j(k-1)},\theta) + \ln P(x_{j1}|\theta)\right\}\right] \end{aligned} \tag{2.21}$$

The probabilities in (2.21) can be expanded using equations (2.10)-(2.12). The random variable, y_{jk} is distributed normally with mean $Cx_{jk} + Du_{jk}$ and covariance R so that

$$\begin{aligned} \ln P(y_{jk}|x_{jk},\theta) &= \ln\left(\frac{1}{(2\pi)^{\frac{p}{2}}|R|^{\frac{1}{2}}} e^{-\frac{1}{2}(y_{jk}-Cx_{jk}-Du_{jk})^T R^{-1}(y_{jk}-Cx_{jk}-Du_{jk})}\right) \\ &= -\frac{p}{2}\ln(2\pi) - \frac{1}{2}\ln|R| \\ &\quad - \frac{1}{2}(y_{jk}-Cx_{jk}-Du_{jk})^T R^{-1}(y_{jk}-Cx_{jk}-Du_{jk}) \end{aligned} \tag{2.22}$$

Similarly, the random state variable x_{jk} is distributed with mean $Ax_{j(k-1)} + Bu_{j(k-1)}$ and covariance Q so that

$$\begin{aligned} \ln P(x_{jk}|x_{j(k-1)},\theta) &= \ln\left(\frac{1}{(2\pi)^{\frac{n}{2}}|Q|^{\frac{1}{2}}} e^{-\frac{1}{2}(x_{jk}-Ax_{j(k-1)}-Bu_{j(k-1)})^T Q^{-1}(x_{jk}-Ax_{j(k-1)}-Bu_{j(k-1)})}\right) \\ &= -\frac{n}{2}\ln(2\pi) - \frac{1}{2}\ln|Q| \\ &\quad - \frac{1}{2}(x_{jk}-Ax_{j(k-1)}-Bu_{j(k-1)})^T Q^{-1}(x_{jk}-Ax_{j(k-1)}-Bu_{j(k-1)}) \end{aligned} \tag{2.23}$$

Furthermore, the initial state x_{j1} is assumed to be distributed normally with mean π_1 and variance V_1

leading to

$$\begin{aligned}\ln P(x_{j1}|\theta) &= \ln\left(\frac{1}{(2\pi)^{\frac{n}{2}}|V_1|^{\frac{1}{2}}}e^{-\frac{1}{2}(x_{j1}-\pi_1)^T V_1^{-1}(x_{j1}-\pi_1)}\right)\\ &= -\frac{n}{2}\ln(2\pi) - \frac{1}{2}\ln|V_1| - \frac{1}{2}(x_{j1}-\pi_1)^T V_1^{-1}(x_{j1}-\pi_1)\end{aligned} \tag{2.24}$$

Substituting (2.22)-(2.24) into the EM objective function, (2.21), and dropping the subscript notation on the expectation for convenience, gives:

$$\begin{aligned}
&E_{X_T|Y_T,\theta}\Big[\ln P(Y_T, X_T|\theta)\Big]\\
&= E\left[\sum_{j=1}^{N}\sum_{k=1}^{T}\left\{-\frac{p}{2}\ln(2\pi) - \frac{1}{2}\ln|R| - \frac{1}{2}(y_{jk}-Cx_{jk}-Du_{jk})^T R^{-1}(y_{jk}-Cx_{jk}-Du_{jk})\right\}\right]\\
&\quad + E\left[\sum_{j=1}^{N}\sum_{k=2}^{T}\left\{-\frac{n}{2}\ln(2\pi) - \frac{1}{2}\ln|Q| - \frac{1}{2}(x_{jk}-Ax_{j(k-1)}-Bu_{j(k-1)})^T Q^{-1}(x_{jk}-Ax_{j(k-1)}-Bu_{j(k-1)})\right\}\right]\\
&\quad + E\left[\sum_{j=1}^{N}\left\{-\frac{n}{2}\ln(2\pi) - \frac{1}{2}\ln|V_1| - \frac{1}{2}(x_{j1}-\pi_1)^T V_1^{-1}(x_{j1}-\pi_1)\right\}\right]\\
&= -\frac{p(N)(T)}{2}\ln(2\pi) - \frac{(N)(T)}{2}\ln|R| - \frac{n(N)}{2}\ln(2\pi) - \frac{N}{2}\ln|V_1| - \frac{n(N)(T-1)}{2}\ln(2\pi)\\
&\quad - \frac{(N)(T-1)}{2}\ln|Q| - \frac{1}{2}\sum_{j=1}^{N}E\left[(x_{j1}-\pi_1)^T V_1^{-1}(x_{j1}-\pi_1)\right]\\
&\quad - \frac{1}{2}\sum_{j=1}^{N}\sum_{k=1}^{T}E\left[(y_{jk}-Cx_{jk}-Du_{jk})^T R^{-1}(y_{jk}-Cx_{jk}-Du_{jk})\right]\\
&\quad - \frac{1}{2}\sum_{j=1}^{N}\sum_{k=2}^{T}E\left[(x_{jk}-Ax_{j(k-1)}-Bu_{j(k-1)})^T Q^{-1}(x_{jk}-Ax_{j(k-1)}-Bu_{j(k-1)})\right]
\end{aligned} \tag{2.25}$$

The expectations are with respect to x given y and θ_n. We can simplify the expectation of the quadratic forms by using a matrix trace. The result of a quadratic form is a scalar so taking the trace of the matrix products within the expectation would not change the result. Since, the trace of a matrix is a linear function, the expectation can be pulled within the trace and the terms can be rotated (because of the trace property $tr[AB] = tr[BA]$) so that constants with respect to the expectation such as u and y can be pulled out. Assume the previously defined notations (2.16)-(2.19); further, let

$$\tilde{x}_k = x_k - \hat{x}_k \tag{2.26}$$

such that $V_k = E[x_k x_k^T | Y_T, \theta] - \hat{x}_k \hat{x}_k^T = E[\tilde{x}_k \tilde{x}_k^T | Y_T, \theta]$. Then, the expectation of y given x simplies as follows

$$
\begin{aligned}
& E\left[(y_{jk} - Cx_{jk} - Du_{jk})^T R^{-1} (y_{jk} - Cx_{jk} - Du_{jk})\right] \\
&= E\left[tr\left(R^{-1}(y_{jk} - C\hat{x}_{jk} - Du_{jk} - C\tilde{x}_{jk})(y_{jk} - C\hat{x}_{jk} - Du_{jk} - C\tilde{x}_{jk})^T\right)\right] \\
&= tr\left[R^{-1}\left((y_{jk} - C\hat{x}_{jk} - Du_{jk})(y_{jk} - C\hat{x}_{jk} - Du_{jk})^T + CV_{jk}C^T\right)\right]
\end{aligned} \tag{2.27}
$$

where we utilized the knowledge that $x_k = \hat{x}_k + \tilde{x}_k$ in the second line and the knowledge that $E[\tilde{x}_k | Y_T, \theta] = 0$ in the third. Applying the same idea to the other expectations,

$$
\begin{aligned}
& E\left[(x_{jk} - Ax_{j(k-1)} - Bu_{j(k-1)})^T Q^{-1} (x_{jk} - Ax_{j(k-1)} - Bu_{j(k-1)})\right] \\
&= E\left[tr\left(Q^{-1}(x_{jk} - Ax_{j(k-1)} - Bu_{j(k-1)})(x_{jk} - Ax_{j(k-1)} - Bu_{j(k-1)})^T\right)\right] \\
&= E\big[tr\big(Q^{-1}(A\hat{x}_{jk} - A\hat{x}_{j(k-1)} - Bu_{j(k-1)} + A\tilde{x}_{jk} - A\tilde{x}_{j(k-1)}) \\
&\quad (A\hat{x}_{jk} - A\hat{x}_{j(k-1)} - Bu_{j(k-1)} + A\tilde{x}_{jk} - A\tilde{x}_{j(k-1)})^T\big)\big] \\
&= tr\big[Q^{-1}\big((\hat{x}_{jk} - A\hat{x}_{j(k-1)} - Bu_{j(k-1)})(\hat{x}_{jk} - A\hat{x}_{j(k-1)} - Bu_{j(k-1)})^T \\
&\quad + V_{jk} - AV_{j(k-1,k)} - V_{j(k,k-1)}A^T + AV_{j(k-1)}A^T\big)\big]
\end{aligned} \tag{2.28}
$$

$$
\begin{aligned}
& E\left[(x_{j1} - \pi_1)^T V_1^{-1} (x_{j1} - \pi_1)\right] \\
&= E\left[tr\left(V_1^{-1}(x_{j1} - \pi_1)(x_{j1} - \pi_1)^T\right)\right] \\
&= E\left[tr\left(V_1^{-1}(\hat{x}_{j1} - \pi_1 + \tilde{x}_{j1})(\hat{x}_{j1} - \pi_1 + \tilde{x}_{j1})^T\right)\right] \\
&= tr\left[V_1^{-1}\left((\hat{x}_{j1} - \pi_1)(\hat{x}_{j1} - \pi_1)^T + V_{j1}\right)\right]
\end{aligned} \tag{2.29}
$$

Substituting (2.27)-(2.29) into (2.25), our final objective function comes to:

$$
\begin{aligned}
& E_{X_T|Y_T,\theta}\Big[\ln P(Y_T, X_T|\theta)\Big] \\
= & -\frac{(N)(T)(p+n)}{2}\ln(2\pi) - \frac{(N)(T)}{2}\ln|R| - \frac{N}{2}\ln|V_1| - \frac{(N)(T-1)}{2}\ln|Q| \\
& - \frac{1}{2}\sum_{j=1}^{N}\sum_{k=1}^{T} tr\big[R^{-1}\big((y_{jk} - C\hat{x}_{jk} - Du_{jk})(y_{jk} - C\hat{x}_{jk} - Du_{jk})^T + CV_{jk}C^T\big)\big] \\
& - \frac{1}{2}\sum_{j=1}^{N} tr\left[V_1^{-1}\left((\hat{x}_{j1} - \pi_1)(\hat{x}_{j1} - \pi_1)^T + V_{j1}\right)\right] \\
& - \frac{1}{2}\sum_{j=1}^{N}\sum_{k=2}^{T} tr\big[Q^{-1}\big((x_{jk} - A\hat{x}_{j(k-1)} - Bu_{j(k-1)})(x_{jk} - A\hat{x}_{j(k-1)} - Bu_{j(k-1)})^T \\
& + V_{jk} - AV_{j(k-1,k)} - V_{j(k,k-1)}A^T + AV_{j(k-1)}A^T\big)\big]
\end{aligned}
\tag{2.30}
$$

2.2.1 The Maximization Step

To solve for the parameters that maximize the objective function on the nth iteration, we take the partial derivative of the objective function with respect to each of the parameters, (A, B, C, D, Q, R, π_1, and V_1), set them all to zero, and solve them simultaneously. Ideally, this locates a maximum in the objective function given the current state variable values and covariances. Let the following notation apply

$$P_k = V_k + \hat{x}_k\hat{x}_k^T = E[x_k x_k^T|Y_k, \theta] \tag{2.31}$$

To take the partial derivative, we make use of several matrix derivative properties.

$$\frac{\partial}{\partial X}AX = \frac{\partial}{\partial X}X^T A = A \tag{2.32}$$

$$\frac{\partial}{\partial X}\left\{tr\left[XA^T\right]\right\} = \frac{\partial}{\partial X}\left\{tr\left[A^T X\right]\right\} = \frac{\partial}{\partial X}\left\{tr\left[X^T A\right]\right\} = \frac{\partial}{\partial X}\left\{tr\left[AX^T\right]\right\} = A \tag{2.33}$$

$$\frac{\partial}{\partial X}\left\{tr\left[A^T XB^T\right]\right\} = \frac{\partial}{\partial X}\left\{tr\left[BX^T A\right]\right\} = AB \tag{2.34}$$

$$\frac{\partial}{\partial X}\left\{tr\left[XAX^T\right]\right\} = \frac{\partial}{\partial X}\left\{tr\left[AX^T X\right]\right\} = \frac{\partial}{\partial X}\left\{tr\left[X^T XA\right]\right\} = X(A + A^T) \tag{2.35}$$

$$\frac{\partial}{\partial X}X^{-1} = -X^{-1}\partial X X^{-1} \tag{2.36}$$

The trace derivatives make use of the general matrix derivative stated in (2.32) and use the trace commutative property to rotate all constants behind the differentiated variable before applying it.

<u>C Derivative</u>

$$\frac{\partial}{\partial C}\left\{-\frac{1}{2}\sum_{j=1}^{N}\sum_{k=1}^{T} tr\Big[R^{-1}\Big((y_{jk}-C\hat{x}_{jk}-Du_{jk})(y_{jk}-C\hat{x}_{jk}-Du_{jk})^T+CV_{jk}C^T\Big)\Big]\right\}$$

$$=-\frac{1}{2}\sum_{j=1}^{N}\sum_{k=1}^{T}\frac{\partial}{\partial C}\Big\{tr\Big[R^{-1}y_{jk}y_{jk}^T-R^{-1}C\hat{x}_{jk}y_{jk}^T-R^{-1}Du_{jk}y_{jk}^T-R^{-1}y_{jk}\hat{x}_{jk}^TC^T+R^{-1}C(V_{jk}+\hat{x}_{jk}\hat{x}_{jk}^T)C^T$$

$$+R^{-1}Du_{jk}\hat{x}_{jk}^TC^T-R^{-1}y_{jk}u_{jk}^TD^T+R^{-1}C\hat{x}_{jk}u_{jk}^TD^T+R^{-1}Du_{jk}u_{jk}^TD^T\Big]\Big\}$$

$$=-\frac{1}{2}\sum_{j=1}^{N}\sum_{k=1}^{T}\frac{\partial}{\partial C}\Big\{tr\Big[-R^{-1}C\hat{x}_{jk}y_{jk}^T-R^{-1}y_{jk}\hat{x}_{jk}^TC^T+R^{-1}CP_{jk}C^T+R^{-1}Du_{jk}\hat{x}_{jk}^TC^T+R^{-1}C\hat{x}_{jk}u_{jk}^TD^T\Big]\Big\}$$

$$=-\frac{1}{2}\sum_{j=1}^{N}\sum_{k=1}^{T}\Big[-R^{-1}y_{jk}\hat{x}_{jk}^T-R^{-1}y_{jk}\hat{x}_{jk}^T+2R^{-1}CP_{jk}+R^{-1}Du_{jk}\hat{x}_{jk}^T+R^{-1}Du_{jk}\hat{x}_{jk}^T\Big]$$

$$=R^{-1}\sum_{j=1}^{N}\sum_{k=1}^{T}\Big[y_{jk}\hat{x}_{jk}^T-CP_{jk}-Du_{jk}\hat{x}_{jk}^T\Big]$$

<u>D Derivative</u>

$$\frac{\partial}{\partial D}\left\{-\frac{1}{2}\sum_{j=1}^{N}\sum_{k=1}^{T} tr\Big[R^{-1}\Big((y_{jk}-C\hat{x}_{jk}-Du_{jk})(y_{jk}-C\hat{x}_{jk}-Du_{jk})^T+CV_{jk}C^T\Big)\Big]\right\}$$

$$=-\frac{1}{2}\sum_{j=1}^{N}\sum_{k=1}^{T}\frac{\partial}{\partial D}\Big\{tr\Big[-R^{-1}Du_{jk}y_{jk}^T+R^{-1}Du_{jk}\hat{x}_{jk}^TC^T-R^{-1}y_{jk}u_{jk}^TD^T+R^{-1}C\hat{x}_{jk}u_{jk}^TD^T+R^{-1}Du_{jk}u_{jk}^TD^T\Big]\Big\}$$

$$=-\frac{1}{2}\sum_{j=1}^{N}\sum_{k=1}^{T}\Big[-R^{-1}y_{jk}u_{jk}^T+R^{-1}C\hat{x}_{jk}u_{jk}^T-R^{-1}y_{jk}u_{jk}^T+R^{-1}C\hat{x}_{jk}u_{jk}^T+2R^{-1}Du_{jk}u_{jk}^T\Big]$$

$$=R^{-1}\sum_{j=1}^{N}\sum_{k=1}^{T}\Big[y_{jk}u_{jk}^T-C\hat{x}_{jk}u_{jk}^T-Du_{jk}u_{jk}^T\Big]$$

<u>R Derivative</u>

$$\frac{\partial}{\partial R^{-1}}\left\{-\frac{(N)(T)}{2}\ln|R|-\frac{1}{2}\sum_{j=1}^{N}\sum_{k=1}^{T} tr\Big[R^{-1}\Big((y_{jk}-C\hat{x}_{jk}-Du_{jk})(y_{jk}-C\hat{x}_{jk}-Du_{jk})^T+CVP_{jk}C^T\Big)\Big]\right\}$$

$$=\frac{(N)(T)}{2}R-\frac{1}{2}\sum_{j=1}^{N}\sum_{k=1}^{T}\Big[(y_{jk}-C\hat{x}_{jk}-Du_{jk})(y_{jk}-C\hat{x}_{jk}-Du_{jk})^T+CV_{jk}C^T\Big]$$

<u>A Derivative</u>

$$\frac{\partial}{\partial A}\Bigg\{-\frac{1}{2}\sum_{j=1}^{N}\sum_{k=2}^{T} tr\Bigg[Q^{-1}\Bigg((x_{jk}-A\hat{x}_{j(k-1)}-Bu_{j(k-1)})(x_{jk}-A\hat{x}_{j(k-1)}-Bu_{j(k-1)})^T$$

$$+V_{jk}-AV_{j(k-1,k)}-V_{j(k,k-1)}A^T+AV_{j(k-1)}A^T\Bigg)\Bigg]\Bigg\}$$

$$=-\frac{1}{2}\sum_{j=1}^{N}\sum_{k=2}^{T}\frac{\partial}{\partial A}\Bigg\{tr\Bigg[Q^{-1}(V_{jk}+\hat{x}_{jk}\hat{x}_{jk}^T)-Q^{-1}A(V_{j(k-1,k)}+\hat{x}_{j(k-1)}\hat{x}_{jk}^T)-Q^{-1}Bu_{j(k-1)}\hat{x}_k^T$$

$$-Q^{-1}(V_{j(k,k-1)}+\hat{x}_k\hat{x}_{j(k-1)}^T)A^T+Q^{-1}A(V_{j(k-1)}+\hat{x}_{j(k-1)}\hat{x}_{j(k-1)}^T)A^T+Q^{-1}Bu_{j(k-1)}\hat{x}_{j(k-1)}^TA^T$$

$$-Q^{-1}\hat{x}_ku_{j(k-1)}^TB^T+Q^{-1}A\hat{x}_{j(k-1)}u_{j(k-1)}^TB^T+Q^{-1}Bu_{j(k-1)}u_{j(k-1)}^TB^T)\Bigg]\Bigg\}$$

$$=-\frac{1}{2}\sum_{j=1}^{N}\sum_{k=2}^{T}\frac{\partial}{\partial A}\Bigg\{tr\Bigg[-Q^{-1}AP_{j(k-1,k)}-Q^{-1}P_{j(k,k-1)}A^T+Q^{-1}AP_{j(k-1)}A^T$$

$$+Q^{-1}Bu_{j(k-1)}\hat{x}_{j(k-1)}^TA^T+Q^{-1}A\hat{x}_{j(k-1)}u_{j(k-1)}^TB^T\Bigg]\Bigg\}$$

$$=-\frac{1}{2}\sum_{j=1}^{N}\sum_{k=2}^{T}\Bigg[-Q^{-1}P_{j(k,k-1)}-Q^{-1}P_{j(k,k-1)}+2Q^{-1}AP_{j(k-1)}+Q^{-1}Bu_{j(k-1)}\hat{x}_{j(k-1)}^T+Q^{-1}Bu_{j(k-1)}\hat{x}_{j(k-1)}^T\Bigg]$$

$$=Q^{-1}\sum_{j=1}^{N}\sum_{k=2}^{T}\Bigg[P_{j(k,k-1)}-AP_{j(k-1)}-Bu_{j(k-1)}\hat{x}_{j(k-1)}^T\Bigg]$$

<u>B Derivative</u>

$$\frac{\partial}{\partial B}\Bigg\{-\frac{1}{2}\sum_{j=1}^{N}\sum_{k=2}^{T} tr\Bigg[Q^{-1}\Bigg((x_{jk}-A\hat{x}_{j(k-1)}-Bu_{j(k-1)})(x_{jk}-A\hat{x}_{j(k-1)}-Bu_{j(k-1)})^T$$

$$+V_{jk}-AV_{j(k-1,k)}-V_{j(k,k-1)}A^T+AV_{j(k-1)}A^T\Bigg)\Bigg]\Bigg\}$$

$$=-\frac{1}{2}\sum_{j=1}^{N}\sum_{k=2}^{T}\frac{\partial}{\partial B}\Bigg\{tr\Bigg[-Q^{-1}Bu_{j(k-1)}\hat{x}_{jk}^T+Q^{-1}Bu_{j(k-1)}\hat{x}_{j(k-1)}^TA^T-Q^{-1}\hat{x}_{jk}u_{j(k-1)}^TB^T$$

$$+Q^{-1}A\hat{x}_{j(k-1)}u_{j(k-1)}^TB^T+Q^{-1}Bu_{j(k-1)}u_{j(k-1)}^TB^T\Bigg]\Bigg\}$$

$$=-\frac{1}{2}\sum_{j=1}^{N}\sum_{k=2}^{T}\Bigg[-Q^{-1}\hat{x}_{jk}u_{j(k-1)}^T+Q^{-1}A\hat{x}_{j(k-1)}u_{j(k-1)}^T-Q^{-1}\hat{x}_{jk}u_{j(k-1)}^T$$

$$+Q^{-1}A\hat{x}_{j(k-1)}u_{j(k-1)}^T+2Q^{-1}Bu_{j(k-1)}u_{j(k-1)}^T\Bigg]$$

$$=Q^{-1}\sum_{j=1}^{N}\sum_{k=2}^{T}\Bigg[\hat{x}_{jk}u_{j(k-1)}^T-A\hat{x}_{j(k-1)}u_{j(k-1)}^T-Bu_{j(k-1)}u_{j(k-1)}^T\Bigg]$$

<u>Q Derivative</u>

$$\frac{\partial}{\partial Q^{-1}}\Bigg\{-\frac{(N)(T-1)}{2}\ln|Q|-\frac{1}{2}\sum_{j=1}^{N}\sum_{k=2}^{T}tr\Bigg[Q^{-1}\Bigg((x_{jk}-A\hat{x}_{j(k-1)}-Bu_{j(k-1)})$$
$$\times(x_{jk}-A\hat{x}_{j(k-1)}-Bu_{j(k-1)})^T+V_{jk}-AV_{j(k-1,k)}-V_{j(k,k-1)}A^T+AV_{j(k-1)}A^T\Bigg)\Bigg]\Bigg\}$$
$$=\frac{(N)(T-1)}{2}Q-\frac{1}{2}\sum_{j=1}^{N}\sum_{k=2}^{T}tr\Bigg[\Bigg((x_{jk}-A\hat{x}_{j(k-1)}-Bu_{j(k-1)})(x_{jk}-A\hat{x}_{j(k-1)}-Bu_{j(k-1)})^T$$
$$+V_{jk}-AV_{j(k-1,k)}-V_{j(k,k-1)}A^T+AV_{j(k-1)}A^T\Bigg)\Bigg]$$

<u>π_1 Derivative</u>

$$\frac{\partial}{\partial\pi_1}\left\{-\frac{1}{2}\sum_{j=1}^{N}tr\left[V_1^{-1}\left(V_{j1}+(\hat{x}_{j1}-\pi_1)(\hat{x}_{j1}-\pi_1)^T\right)\right]\right\}$$
$$=-\frac{1}{2}\sum_{j=1}^{N}\frac{\partial}{\partial\pi_1}\left\{tr\left[V_1^{-1}V_{j1}+V_1^{-1}\hat{x}_{j1}\hat{x}_{j1}^T-V_1^{-1}\hat{x}_{j1}\pi_1^T-V_1^{-1}\pi_1\hat{x}_{j1}^T+V_1^{-1}\pi_1\pi_1^T\right]\right\}$$
$$=-\frac{1}{2}\sum_{j=1}^{N}\frac{\partial}{\partial\pi_1}\left\{tr\left[-V_1^{-1}\hat{x}_{j1}\pi_1^T-V_1^{-1}\pi_1\hat{x}_{j1}^T+V_1^{-1}\pi_1\pi_1^T\right]\right\}$$
$$=-\frac{1}{2}\sum_{j=1}^{N}\left[-V_1^{-1}\hat{x}_{j1}-V_1^{-1}\hat{x}_{j1}+2V_1^{-1}\pi_1\right]$$
$$=V_1^{-1}\sum_{j=1}^{N}\left[\hat{x}_{j1}-\pi_1\right]$$

<u>V_1 Derivative</u>

$$\frac{\partial}{\partial V_1}\left\{-\frac{N}{2}log|V_1|-\frac{1}{2}\sum_{j=1}^{N}tr\left[V_1^{-1}\left(V_{j1}+(\hat{x}_{j1}-\pi_1)(\hat{x}_{j1}-\pi_1)^T\right)\right]\right\}$$
$$=\frac{N}{2}V_1-\frac{1}{2}\sum_{j=1}^{N}\frac{\partial}{\partial V_1}\left\{tr\left[V_1^{-1}\left(V_{j1}+(\hat{x}_{j1}-\pi_1)(\hat{x}_{j1}-\pi_1)^T\right)\right]\right\}$$
$$=\frac{N}{2}V_1-\frac{1}{2}\sum_{j=1}^{N}\left[V_{j1}+(\hat{x}_{j1}-\pi_1)(\hat{x}_{j1}-\pi_1)^T\right]$$

Below are the entire set of derivatives to solve.

$$\sum_{j=1}^{N}\sum_{k=1}^{T}\left[y_{jk}\hat{x}_{jk}^{T} - CP_{jk} - Du_{jk}\hat{x}_{jk}^{T}\right] = 0 \tag{2.37a}$$

$$\sum_{j=1}^{N}\sum_{k=1}^{T}\left[y_{jk}u_{jk}^{T} - C\hat{x}_{jk}u_{jk}^{T} - Du_{jk}u_{jk}^{T}\right] = 0 \tag{2.37b}$$

$$\frac{(N)(T)}{2}R - \frac{1}{2}\sum_{j=1}^{N}\sum_{k=1}^{T}\left[(y_{jk} - C\hat{x}_{jk} - Du_{jk})(y_{jk} - C\hat{x}_{jk} - Du_{jk})^{T} + CV_{jk}C^{T}\right] = 0 \tag{2.37c}$$

$$\sum_{j=1}^{N}\sum_{k=2}^{T}\left[P_{j(k,k-1)} - AP_{j(k-1)} - Bu_{j(k-1)}\hat{x}_{j(k-1)}^{T}\right] = 0 \tag{2.37d}$$

$$\sum_{j=1}^{N}\sum_{k=2}^{T}\left[\hat{x}_{jk}u_{j(k-1)}^{T} - A\hat{x}_{j(k-1)}u_{j(k-1)}^{T} - Bu_{j(k-1)}u_{j(k-1)}^{T}\right] = 0 \tag{2.37e}$$

$$\begin{aligned}\frac{(N)(T-1)}{2}Q - \frac{1}{2}\sum_{j=1}^{N}\sum_{k=2}^{T}\Bigg[\Big((\hat{x}_{jk} - A\hat{x}_{j(k-1)} - Bu_{j(k-1)})(\hat{x}_{jk} - A\hat{x}_{j(k-1)} - Bu_{j(k-1)})^{T} \\ +V_{jk} - AV_{j(k-1,k)} - V_{j(k,k-1)}A^{T} + AV_{j(k-1)}A^{T}\Big)\Bigg] = 0\end{aligned} \tag{2.37f}$$

$$\sum_{j=1}^{N}\left[\hat{x}_{j1} - \pi_1\right] = 0 \tag{2.37g}$$

$$\frac{N}{2}V_1 - \frac{1}{2}\sum_{j=1}^{N}\left[V_{j1} + (\hat{x}_{j1} - \pi_1)(\hat{x}_{j1} - \pi_1)^{T}\right] = 0 \tag{2.37h}$$

The derivatives of C and D, shown in (2.37a) and (2.37b), are coupled so in order to solve them simultaneously we use matrix algebra. First, we move the non-variable-dependent constants to the other side of the equality and isolate the terms containing C and D for both equations. Then, after putting it in matrix form, we solve for C and D.

$$\begin{aligned}\sum_{j=1}^{N}\sum_{k=1}^{T}y_{jk}\hat{x}_{jk}^{T} &= C\sum_{j=1}^{N}\sum_{k=1}^{T}P_{jk} + D\sum_{j=1}^{N}\sum_{k=1}^{T}u_{jk}\hat{x}_{jk}^{T} \\ \sum_{j=1}^{N}\sum_{k=1}^{T}y_{jk}u_{jk}^{T} &= C\sum_{j=1}^{N}\sum_{k=1}^{T}\hat{x}_{jk}u_{jk}^{T} + D\sum_{j=1}^{N}\sum_{k=1}^{T}u_{jk}u_{jk}^{T}\end{aligned}$$

$$\begin{bmatrix}\sum_{j=1}^{N}\sum_{k=1}^{T}y_{jk}\hat{x}_{jk}^{T} & \sum_{j=1}^{N}\sum_{k=1}^{T}y_{jk}u_{jk}^{T}\end{bmatrix} = \begin{bmatrix}C & D\end{bmatrix}\begin{bmatrix}\sum_{j=1}^{N}\sum_{k=1}^{T}P_{jk} & \sum_{j=1}^{N}\sum_{k=1}^{T}\hat{x}_{jk}u_{jk}^{T} \\ \sum_{j=1}^{N}\sum_{k=1}^{T}u_{jk}\hat{x}_{jk}^{T} & \sum_{j=1}^{N}\sum_{k=1}^{T}u_{jk}u_{jk}^{T}\end{bmatrix}$$

$$\begin{bmatrix} \sum_{j=1}^{N}\sum_{k=1}^{T} y_{jk}\hat{x}_{jk}^T & \sum_{j=1}^{N}\sum_{k=1}^{T} y_{jk}u_{jk}^T \end{bmatrix} \begin{bmatrix} \sum_{j=1}^{N}\sum_{k=1}^{T} P_{jk} & \sum_{j=1}^{N}\sum_{k=1}^{T} \hat{x}_{jk}u_{jk}^T \\ \sum_{j=1}^{N}\sum_{k=1}^{T} u_{jk}\hat{x}_{jk}^T & \sum_{j=1}^{N}\sum_{k=1}^{T} u_{jk}u_{jk}^T \end{bmatrix}^{-1} = \begin{bmatrix} C_{new} & D_{new} \end{bmatrix} \tag{2.38}$$

Similarly, we solve for A and B. Start with (2.37d)-(2.37e)

$$\sum_{j=1}^{N}\sum_{k=2}^{T} P_{j(k,k-1)} = A\sum_{j=1}^{N}\sum_{k=2}^{T}(P_{j(k-1)} + \hat{x}_{j(k-1)}\hat{x}_{j(k-1)}^T) + B\sum_{j=1}^{N}\sum_{k=2}^{T} u_{j(k-1)}\hat{x}_{j(k-1)}^T$$
$$\sum_{j=1}^{N}\sum_{k=2}^{T} \hat{x}_{jk}u_{j(k-1)}^T = A\sum_{j=1}^{N}\sum_{k=2}^{T} \hat{x}_{j(k-1)}u_{j(k-1)}^T + B\sum_{j=1}^{N}\sum_{k=2}^{T} u_{j(k-1)}u_{j(k-1)}^T$$

$$\begin{bmatrix} \sum_{j=1}^{N}\sum_{k=2}^{T} P_{j(k,k-1)} & \sum_{j=1}^{N}\sum_{k=2}^{T} \hat{x}_{jk}u_{j(k-1)}^T \end{bmatrix} = \begin{bmatrix} A & B \end{bmatrix} \begin{bmatrix} \sum_{j=1}^{N}\sum_{k=2}^{T} P_{j(k-1)} & \sum_{j=1}^{N}\sum_{k=2}^{T} \hat{x}_{j(k-1)}u_{j(k-1)}^T \\ \sum_{j=1}^{N}\sum_{k=2}^{T} u_{j(k-1)}\hat{x}_{j(k-1)}^T & \sum_{j=1}^{N}\sum_{k=2}^{T} u_{j(k-1)}u_{j(k-1)}^T \end{bmatrix}$$

$$\begin{bmatrix} \sum_{j=1}^{N}\sum_{k=2}^{T} P_{j(k,k-1)} & \sum_{j=1}^{N}\sum_{k=2}^{T} \hat{x}_{jk}u_{j(k-1)}^T \end{bmatrix} \begin{bmatrix} \sum_{j=1}^{N}\sum_{k=2}^{T} P_{j(k-1)} & \sum_{j=1}^{N}\sum_{k=2}^{T} \hat{x}_{j(k-1)}u_{j(k-1)}^T \\ \sum_{j=1}^{N}\sum_{k=2}^{T} u_{j(k-1)}\hat{x}_{j(k-1)}^T & \sum_{j=1}^{N}\sum_{k=2}^{T} u_{j(k-1)}u_{j(k-1)}^T \end{bmatrix}^{-1}$$
$$= \begin{bmatrix} A_{new} & B_{new} \end{bmatrix} \tag{2.39}$$

Once, A, B, C, and D are defined, they are used to solve for Q and R, starting with equation (2.37c) and (2.37f), respectively. Substituting what we know of the new A - D values, we can greatly simplify their expressions.

$$R_{new} = \frac{1}{(N)(T)}\sum_{j=1}^{N}\sum_{k=1}^{T}\Big[(y_{jk} - C\hat{x}_{jk} - Du_{jk})(y_{jk} - C\hat{x}_{jk} - Du_{jk})^T + CV_{jk}C^T\Big]$$
$$\begin{aligned} R_{new} = &\frac{1}{(N)(T)}\sum_{j=1}^{N}\sum_{k=1}^{T}\Big[y_{jk}y_{jk}^T - C_{new}\hat{x}_{jk}y_{jk}^T - D_{new}u_{jk}y_{jk}^T \\ &- y_{jk}\hat{x}_{jk}^T C_{new}^T + C_{new}P_{jk}C_{new}^T + D_{new}u_{jk}\hat{x}_{jk}^T C_{new}^T \\ &- y_{jk}u_{jk}^T D_{new}^T + C_{new}\hat{x}_{jk}u_{jk}^T D_{new}^T + D_{new}u_{jk}u_{jk}^T D_{new}^T\Big] \end{aligned}$$

The second line of this expansion equals zero because (2.37a) holds true and similarly, the third line of this expansion also equals zero because (2.37b) holds true. As a result, we arrive at

$$R_{new} = \frac{1}{(N)(T)} \sum_{j=1}^{N} \sum_{k=1}^{T} \left[y_{jk} y_{jk}^T - C_{new} \hat{x}_{jk} y_{jk}^T - D_{new} u_{jk} y_{jk}^T \right] \tag{2.40}$$

The derivation of the Q_{new} equation simplifies similarly due to (2.37d) and (2.37e).

$$\begin{aligned}
Q_{new} =& \frac{1}{(N)(T-1)} \sum_{j=1}^{N} \sum_{k=2}^{T} \Bigg[\Bigg((\hat{x}_{jk} - A\hat{x}_{j(k-1)} - Bu_{j(k-1)})(\hat{x}_{jk} - A\hat{x}_{j(k-1)} - Bu_{j(k-1)})^T \\
&+ V_{jk} - AV_{j(k-1,k)} - V_{j(k,k-1)} A^T + AV_{j(k-1)} A^T \Bigg) \Bigg] \\
=& \frac{1}{(N)(T-1)} \sum_{j=1}^{N} \sum_{k=2}^{T} \Bigg[P_{jk} - A_{new} P_{j(k-1,k)} - B_{new} u_{j(k-1)} \hat{x}_{jk}^T \\
&- P_{j(k,k-1)} A_{new}^T + A_{new} P_{j(k-1)} A_{new}^T + B_{new} u_{j(k-1)} \hat{x}_{j(k-1)}^T A_{new}^T \\
&- \hat{x}_{jk} u_{j(k-1)}^T B_{new}^T + A_{new} \hat{x}_{j(k-1)} u_{j(k-1)}^T B_{new}^T + B_{new} u_{j(k-1)} u_{j(k-1)}^T B_{new}^T \Bigg] \\
Q_{new} =& \frac{1}{(N)(T-1)} \sum_{j=1}^{N} \sum_{k=2}^{T} \left[P_{jk} - A_{new} P_{j(k-1,k)} - B_{new} u_{j(k-1)} \hat{x}_{jk}^T \right]
\end{aligned} \tag{2.41}$$

Finally, π_1 must be solved first and then used to solve V_1, using (2.37g) and (2.37h) respectively.

$$\begin{aligned}
\sum_{j=1}^{N} \hat{x}_{j1} &= \sum_{j=1}^{N} \pi_1 \\
\sum_{j=1}^{N} \hat{x}_{j1} &= N\pi_1 \\
\pi_{1new} &= \frac{1}{N} \sum_{j=1}^{N} \hat{x}_{j1}
\end{aligned} \tag{2.42}$$

$$\begin{aligned}
\frac{N}{2} V_1 &= \frac{1}{2} \sum_{j=1}^{N} \left[V_{j1} + (\hat{x}_{j1} - \pi_1)(\hat{x}_{j1} - \pi_1)^T \right] \\
V_{1new} &= \frac{1}{N} \sum_{j=1}^{N} \left[V_{j1} + (\hat{x}_{j1} - \pi_1)(\hat{x}_{j1} - \pi_1)^T \right]
\end{aligned} \tag{2.43}$$

Equations (2.38)-(2.43) define the new update equations and complete our derivation of the Maximization step of the EM algorithm.

2.2.2 The Expectation Step Equations

The purpose of this section is to derive the Kalman filtering and smoothing update equations, which update the state and covariance estimates given the new parameter results from the Maximization step. It elaborates on the derivation outlined in [22] and certain portions follow closely the derivation presented in [32]. We start with the forward recursion formulas first, because they are used in the backward recursion formulas to yield the final optimal estimate $\hat{x}_k$. We know that the forward recursion maximizes the probability of the current states x_k given all observations up to y_k or $p(x_k|Y_k)$. This is equivalent to maximizing the log of this probability because the log is a linearly increasing function.

$$\begin{aligned}\log p(x_k|Y_k) &= \log p(x_k, Y_k) - \log p(Y_k) \\ &= \log[p(y_k|x_k)p(x_k|Y_{k-1})p(Y_{k-1})] - \log p(Y_k)\end{aligned} \tag{2.44}$$

where, the uppercase Y_k denotes the set of observations up to sample k, $\{y_1, y_2, \ldots, y_k\}$. To find this probability, we must first find the parameters of the normal distribution that define it. We begin by finding the mean and covariance of $p(x_k|Y_{k-1})$. First, we define the error and covariance term notations.

$$e_{k-1|k-1} = x_{k-1} - \hat{x}_{k-1|k-1} \tag{2.45a}$$

$$E[e_{k-1|k-1}e^T_{k-1|k-1}] = V_{k-1|k-1} \tag{2.45b}$$

$$e_{k|k-1} = x_k - \hat{x}_{k|k-1} \tag{2.45c}$$

$$E[e_{k|k-1}e^T_{k|k-1}] = V_{k|k-1} \tag{2.45d}$$

Next, we solve for the mean of $x_{k|k-1}$ in terms of its previous iterates in the Kalman filter:

$$\begin{aligned}\hat{x}_{k|k-1} &= E[x_k|Y_{k-1}] \\ &= E[Ax_{k-1} + Bu_{k-1} + w_{k-1}|Y_{k-1}] \\ &= A\hat{x}_{k-1|k-1} + Bu_{k-1}\end{aligned} \tag{2.46}$$

Substituting the result from (2.46) into (2.45d), we can solve for the covariance $V_{k|k-1}$ in terms of the previous iterates of the Kalman filter as well:

$$\begin{aligned}
V_{k|k-1} &= E[e_{k|k-1}e_{k|k-1}^T|Y_{k-1}] \\
&= E[(x_k - \hat{x}_{k|k-1})(x_k - \hat{x}_{k|k-1})^T|Y_{k-1}] \\
&= E[(Ax_{k-1} + Bu_{k-1} + w_{k-1} - A\hat{x}_{k-1|k-1} - Bu_{k-1}) \\
&\quad (Ax_{k-1} + Bu_{k-1} + w_{k-1} - A\hat{x}_{k-1|k-1} - Bu_{k-1})^T|Y_{k-1}] \\
&= E[(Ae_{k-1|k-1} + w_{k-1})(Ae_{k-1|k-1} + w_{k-1})^T|Y_{k-1}] \\
&= AE[e_{k-1|k-1}e_{k-1|k-1}^T]A^T + E[w_{k-1}w_{k-1}^T] \\
&= AV_{k-1|k-1}A^T + Q
\end{aligned} \tag{2.47}$$

where we have used the fact that $e_{k-1|k-1}$ is independent of w_{k-1}. Equations (2.46)-(2.47) give the distribution parameters of $p(x_k|Y_{k-1})$. Now, if we define the mean and covariance of $p(y_k|x_k)$, the log likelihood equation will be sufficiently defined to solve for an optimal estimate of $x_{k|k}$. Since

$$E[y_k|x_k] = Cx_k + Du_k \tag{2.48a}$$

$$cov(y_k|x_k) = R \tag{2.48b}$$

Substituting (2.46)-(2.48b) into (2.44), the combined log likelihood becomes

$$\begin{aligned}
\log p(x_k|Y_k) &= \log[p(y_k|x_k)p(x_k|Y_{k-1})p(Y_{k-1})] - \log p(Y_k) \\
&= \log\left[\frac{1}{(2\pi)^{\frac{r}{2}}|R|^{\frac{1}{2}}}e^{-\frac{1}{2}(y_k - Cx_k - Du_k)^T R^{-1}(y_k - Cx_k - Du_k)}\right. \\
&\quad \times \frac{1}{(2\pi)^{\frac{n}{2}}|V_{k|k-1}|^{\frac{1}{2}}}e^{-\frac{1}{2}(x_k - \hat{x}_{k|k-1})^T V_{k|k-1}^{-1}(x_k - \hat{x}_{k|k-1})} \\
&\quad \left. \times p(Y_{k-1})\right] - \log p(Y_k) \\
&= -\frac{r}{2}\log(2\pi) - \frac{1}{2}\log|R| - \frac{1}{2}(y_k - Cx_k - Du_k)^T R^{-1}(y_k - Cx_k - Du_k) \\
&\quad - \frac{n}{2}\log(2\pi) - \frac{1}{2}\log|V_{k|k-1}| - \frac{1}{2}(x_k - \hat{x}_{k|k-1})^T V_{k|k-1}^{-1}(x_k - \hat{x}_{k|k-1}) \\
&\quad + \log p(Y_{k-1}) - \log p(Y_k)
\end{aligned} \tag{2.49}$$

The optimal estimate $\hat{x}_{k|k}$ would be a maxima in this probability function. To find a maxima in this function, we take its derivative with respect to x_k, set that to zero, and solve for x_k. To simplify later steps in the derivation as well as notation, we can remove the terms that are not dependent on x_k from the log likelihood now because they will disappear when we take the derivative with respect to x_k anyhow. The simplified log likelihood is

$$\log p(x_k|Y_k) = \ldots - \frac{1}{2}(y_k - Cx_k - Du_k)^T R^{-1}(y_k - Cx_k - Du_k) - \frac{1}{2}(x_k - \hat{x}_{k|k-1})^T V_{k|k-1}^{-1}(x_k - \hat{x}_{k|k-1}) - \ldots \tag{2.50}$$

Now, taking the derivative of this simplified log likelihood yields,

$$\begin{aligned}
&\frac{\partial}{\partial x_k}\Big\{ -\frac{1}{2}(y_k - Cx_k - Du_k)^T R^{-1}(y_k - Cx_k - Du_k) - \frac{1}{2}(x_k - \hat{x}_{k|k-1})^T V_{k|k-1}^{-1}(x_k - \hat{x}_{k|k-1})\Big\} \\
&= -\frac{1}{2}(-2C^T R^{-1}(y_k - Cx_k - Du_k)) - \frac{1}{2}(2V_{k|k-1}^{-1}(x_k - \hat{x}_{k|k-1})) \\
&= C^T R^{-1} y_k - C^T R^{-1} C x_k - C^T R^{-1} D u_k - V_{k|k-1}^{-1} x_k + V_{k|k-1}^{-1}\hat{x}_{k|k-1} \\
&= -\left(C^T R^{-1} C + V_{k|k-1}^{-1}\right) x_k + C^T R^{-1} y_k - C^T R^{-1} D u_k + V_{k|k-1}^{-1}\hat{x}_{k|k-1}
\end{aligned} \tag{2.51}$$

Next, we can set this term equal to zero and solve to yield

$$\hat{x}_{k|k} = \left(C^T R^{-1} C + V_{k|k-1}^{-1}\right)^{-1}\left(C^T R^{-1} y_k - C^T R^{-1} D u_k + V_{k|k-1}^{-1}\hat{x}_{k|k-1}\right) \tag{2.52}$$

Using the Matrix Inversion Lemma, we can expand the inverse term in (2.52) to further simplify the result.

$$
\begin{aligned}
\left(C^T R^{-1} C + V_{k|k-1}^{-1}\right)^{-1} &= V_{k|k-1} - V_{k|k-1} C^T (C V_{k|k-1} C^T + R)^{-1} C V_{k|k-1} \\
\left(C^T R^{-1} C + V_{k|k-1}^{-1}\right)^{-1} C^T R^{-1} y_k &= \left(V_{k|k-1} - V_{k|k-1} C^T (C V_{k|k-1} C^T + R)^{-1} C V_{k|k-1}\right) C^T R^{-1} y_k \\
&= V_{k|k-1} C^T \left(R^{-1} - (C V_{k|k-1} C^T + R)^{-1} C V_{k|k-1} C^T R^{-1}\right) y_k \\
&= V_{k|k-1} C^T (C V_{k|k-1} C^T + R)^{-1} y_k \\
\left(C^T R^{-1} C + V_{k|k-1}^{-1}\right)^{-1} C^T R^{-1} D u_k &= \left(V_{k|k-1} - V_{k|k-1} C^T (C V_{k|k-1} C^T + R)^{-1} C V_{k|k-1}\right) C^T R^{-1} D u_k \\
&= V_{k|k-1} C^T (C V_{k|k-1} C^T + R)^{-1} D u_k \\
\left(C^T R^{-1} C + V_{k|k-1}^{-1}\right)^{-1} V_{k|k-1}^{-1} \hat{x}_{k|k-1} &= \left(V_{k|k-1} - V_{k|k-1} C^T (C V_{k|k-1} C^T + R)^{-1} C V_{k|k-1}\right) V_{k|k-1}^{-1} \hat{x}_{k|k-1} \\
&= \hat{x}_{k|k-1} - V_{k|k-1} C^T (C V_{k|k-1} C^T + R)^{-1} C \hat{x}_{k|k-1}
\end{aligned}
$$

Putting these expansions back into (2.52), we get

$$
\begin{aligned}
\hat{x}_{k|k} &= V_{k|k-1} C^T (C V_{k|k-1} C^T + R)^{-1} y_k \\
&\quad - V_{k|k-1} C^T (C V_{k|k-1} C^T + R)^{-1} D u_k \\
&\quad + \hat{x}_{k|k-1} - V_{k|k-1} C^T (C V_{k|k-1} C^T + R)^{-1} C \hat{x}_{k|k-1} \\
&= \hat{x}_{k|k-1} + V_{k|k-1} C^T (C V_{k|k-1} C^T + R)^{-1} \left(y_k - D u_k - C \hat{x}_{k|k-1}\right)
\end{aligned} \tag{2.53}
$$

Defining

$$
K_k = V_{k|k-1} C^T (C V_{k|k-1} C^T + R)^{-1} \tag{2.54}
$$

allows us to rewrite the update equation as

$$
\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k \left(y_k - D u_k - C \hat{x}_{k|k-1}\right) \tag{2.55}
$$

Equations (2.46),(2.47), (2.54) and (2.55) give the predictive state update equations. Using them, we can solve for the predictive covariance update equations as well.

$$
\begin{aligned}
e_{k|k} &= x_k - \hat{x}_{k|k} \\
&= Ax_{k-1} + Bu_{k-1} + w_{k-1} - \Big[A\hat{x}_{k-1|k-1} + Bu_{k-1} + K_k(Cx_k + Du_k + v_k - Du_k - C\hat{x}_{k|k-1})\Big] \\
&= A(x_{k-1} - \hat{x}_{k-1|k-1}) + w_{k-1} - K_k v_k - K_k\Big[C(Ax_{k-1} + Bu_{k-1} + w_{k-1}) - C(A\hat{x}_{k-1|k-1} + Bu_{k-1})\Big] \\
&= (I - K_kC)\Big[A(x_{k-1} - \hat{x}_{k-1|k-1}) + w_{k-1}\Big] - K_k v_k \\
&= (I - K_kC)\Big[Ae_{k-1|k-1} + w_{k-1}\Big] - K_k v_k \qquad (2.56) \\
V_{k|k} &= E[e_{k|k}e_{k|k}^T] \\
&= E\Big[\big((I - K_kC)\big[Ae_{k-1|k-1} + w_{k-1}\big] - K_kv_k\big)\big((I - K_kC)\big[Ae_{k-1|k-1} + w_{k-1}\big] - K_kv_k\big)^T\Big] \\
&= (I - K_kC)E\Big[\big(Ae_{k-1|k-1} + w_{k-1}\big)\big(Ae_{k-1|k-1} + w_{k-1}\big)^T\Big](I - K_kC)^T + E[K_kv_kv_k^TK_k^T] \\
&= (I - K_kC)V_{k|k-1}(I - K_kC)^T + K_kRK_k^T \\
&= (I - K_kC)V_{k|k-1} - (I - K_kC)V_{k|k-1}C^TK_k^T + K_kRK_k^T \\
&= (I - K_kC)V_{k|k-1} \qquad (2.57)
\end{aligned}
$$

where we used the fact that $(I - K_kC)V_{k|k-1}C^T = K_k^TR$ to get the final result.

Equation (2.56) completes the set of recursive predictive update equations. In summary, (2.54)-(2.56) are the optimal state estimates, given all observations up to the current sample, k; they are also called the measurement updates. However, in order to solve for (2.54)-(2.56) at each time sample k, we need optimal state estimates given all observations up to the previous sample, $k-1$, which are given by (2.46)-(2.47); these are called the time updates. These equations are recursive and their solution requires an initial condition. So in our case we assume $x_1^0 = \pi_1$ and $V_1^0 = V_1$, where π_1 and V_1 are part of the parameter set we are

estimating . The full set of forward equations are given by

$$\hat{x}_{k|k-1} = A\hat{x}_{k-1|k-1} + Bu_{k-1} \tag{2.58a}$$

$$V_{k|k-1} = AV_{k-1|k-1}A^T + Q \tag{2.58b}$$

$$K_k = V_{k|k-1}C^T(CV_{k|k-1}C^T + R)^{-1} \tag{2.58c}$$

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k\Big(y_k - Du_k - C\hat{x}_{k|k-1}\Big) \tag{2.58d}$$

$$V_{k|k} = (I - K_kC)V_{k|k-1} \tag{2.58e}$$

This does not complete our derivation of the Expectation step, however, because the estimates used in the objective function of the Maximization step are the smoothed estimates, which assume knowledge of all observations, Y_T. So, next we solve for the smoothing update equations, a process very similar to the predictive update derivation. The backward recursion formula maximizes the probability of the current states x_k given all observations Y_T or $p(x_k|Y_T)$. This is equivalent to maximizing the log of this probability because the log is a monotonically increasing function. But as its name suggests, the backward recursion works backward from the Tth time sample to the first so each state at time k is updated from the state a time sample after it, $k-1$.

$$\begin{aligned}
\log p(x_k, x_{k+1}|Y_T) &= \log p(x_k, x_{k+1}, Y_T) - \log p(Y_T) \\
&= \log\left[p(x_k, x_{k+1}, y_{k+1}, y_{k+2}, \ldots, y_T|Y_k)p(Y_k)\right] - \log p(Y_T) \\
&= \log\left[p(y_{k+1}, y_{k+2}, \ldots, y_T|x_{k+1})p(x_{k+1}|x_k)p(x_k|Y_k)p(Y_k)\right] - \log p(Y_T)
\end{aligned} \tag{2.59}$$

Because we are maximizing with respect to x_k, we are again only concerned with probability terms containing x_k: $logp(x_{k+1}|x_k)$ and $logp(x_k|Y_k)$. To define these probabilities, we must derive their distribution

parameters, the mean and the covariance.

$$E[x_k|Y_k] = \hat{x}_{k|k} \tag{2.60a}$$

$$E[e_{k|k}e_{k|k}^T] = V_{k|k} \tag{2.60b}$$

$$\begin{aligned} E[x_{k+1}|x_k] &= E[Ax_k + Bu_k + w_k|x_k] \\ &= Ax_k + Bu_k \end{aligned} \tag{2.60c}$$

$$\begin{aligned} E\big[(x_{k+1} - Ax_k - Bu_k)(x_{k+1} - Ax_k - Bu_k)^T\big] &= E\big[w_k w_k^T\big] \\ &= Q \end{aligned} \tag{2.60d}$$

Substituting (2.60a) through (2.60d) into (2.59), and removing all terms not containing x_k gives the simplified log likelihood equation,

$$\begin{aligned} \log p(x_k, x_{k+1}|Y_T) &= \ldots \log p(x_{k+1}|x_k) + \log p(x_k|Y_k) \ldots \\ &= \ldots - \frac{1}{2}(x_{k+1} - Ax_k - Bu_k)^T Q^{-1}(x_{k+1} - Ax_k - Bu_k) \\ &\quad - \frac{1}{2}(x_k - \hat{x}_{k|k})^T V_{k|k}^{-1}(x_k - \hat{x}_{k|k}) - \ldots \end{aligned} \tag{2.61}$$

Again, we solve for the maxima of this probability function to derive the optimal smoothed estimates. Thus, we must take the derivative, set it equal to zero, and solve for $x_{k|T}$.

$$\begin{aligned} &\frac{\partial}{\partial x_k}\Big\{ -\frac{1}{2}(x_{k+1} - Ax_k - Bu_k)^T Q^{-1}(x_{k+1} - Ax_k - Bu_k) - \frac{1}{2}(x_k - \hat{x}_{k|k})^T V_{k|k}^{-1}(x_k - \hat{x}_{k|k})\Big\} \\ &= -\frac{1}{2}(-2A^T Q^{-1}(x_{k+1} - Ax_k - Bu_k)) - \frac{1}{2}(2V_{k|k}^{-1}(x_k - \hat{x}_{k|k})) \\ &= -\left(A^T Q^{-1} A + V_{k|k}^{-1}\right)x_k + \left(A^T Q^{-1} x_{k+1} - A^T Q^{-1} B u_k + V_{k|k}^{-1}\hat{x}_{k|k}\right) \end{aligned}$$

$$\hat{x}_{k|T} = \left(A^T Q^{-1} A + V_{k|k}^{-1}\right)^{-1}\left(A^T Q^{-1}\hat{x}_{k+1|T} - A^T Q^{-1} B u_k + V_{k|k}^{-1}\hat{x}_{k|k}\right) \tag{2.62}$$

Using the Matrix Inversion Lemma, we have

$$\begin{aligned}
\left(A^T Q^{-1} A + V_{k|k}^{-1}\right)^{-1} &= V_{k|k} - V_{k|k} A^T (A V_{k|k} A^T + Q)^{-1} A V_{k|k} \\
\left(A^T Q^{-1} A + V_{k|k}^{-1}\right)^{-1} A^T Q^{-1} \hat{x}_{k+1|T} &= V_{k|k} A^T (A V_{k|k} A^T + Q)^{-1} \hat{x}_{k+1|T} \\
\left(A^T Q^{-1} A + V_{k|k}^{-1}\right)^{-1} A^T Q^{-1} B u_k &= V_{k|k} A^T (A V_{k|k} A^T + Q)^{-1} B u_k \\
\left(A^T Q^{-1} A + V_{k|k}^{-1}\right)^{-1} V_{k|k}^{-1} \hat{x}_{k|k} &= \hat{x}_{k|k} - V_{k|k} A^T (A V_{k|k} A^T + Q)^{-1} A \hat{x}_{k|k}
\end{aligned}$$

Thus we can write

$$\begin{aligned}
\hat{x}_{k|T} =& \left(A^T Q^{-1} A + V_{k|k}^{-1}\right)^{-1} \left(A^T Q^{-1} \hat{x}_{k+1|T} - A^T Q^{-1} B u_k + V_{k|k}^{-1} \hat{x}_{k|k}\right) \\
=& V_{k|k} A^T (A V_{k|k} A^T + Q)^{-1} \hat{x}_{k+1|T} - V_{k|k} A^T (A V_{k|k} A^T + Q)^{-1} B u_k \\
&+ \hat{x}_{k|k} - V_{k|k} A^T (A V_{k|k} A^T + Q)^{-1} A \hat{x}_{k|k} \\
=& \hat{x}_{k|k} + V_{k|k} A^T (A V_{k|k} A^T + Q)^{-1} \left(\hat{x}_{k+1|T} - B u_k - A \hat{x}_{k|k}\right) \\
=& \hat{x}_{k|k} + V_{k|k} A^T V_{k+1|k}^{-1} \left(\hat{x}_{k+1|T} - B u_k - A \hat{x}_{k|k}\right)
\end{aligned}$$

To simplify notation, define

$$J_k = V_{k|k} A^T V_{k+1|k}^{-1} \tag{2.63}$$

such that

$$\hat{x}_{k|T} = \hat{x}_{k|k} + J_k \left(\hat{x}_{k+1|T} - B u_k - A \hat{x}_{k|k}\right) \tag{2.64}$$

Equations (2.63) and (2.64) define the smoothed state estimates. Using them, we can derive the smoothed covariance estimates as well. From the definition

$$\begin{aligned}
e_{k|T} &= x_k - \hat{x}_{k|T} \\
&= x_k - \hat{x}_{k|k} - J_k \left(\hat{x}_{k+1|T} - B u_k - A \hat{x}_{k|k}\right)
\end{aligned} \tag{2.65}$$

we get that

$$\begin{aligned} e_{k|T} + J_k \hat{x}_{k+1|T} &= e_{k|k} + J_k B u_k + J_k A \hat{x}_{k|k} \\ &= e_{k|k} + J_k \hat{x}_{k+1|k} \end{aligned} \tag{2.66}$$

$$E\big[\big(e_{k|T} + J_k \hat{x}_{k+1|T}\big)\big(e_{k|T} + J_k \hat{x}_{k+1|T}\big)^T\big] = E\big[\big(e_{k|k} + J_k \hat{x}_{k+1|k}\big)\big(e_{k|k} + J_k \hat{x}_{k+1|k}\big)^T\big]$$

Because the state noise is independent across time samples, the expectation of the cross terms disappear.

$$\begin{aligned} V_{k|T} + J_k E\big[\hat{x}_{k+1|T}\hat{x}_{k+1|T}^T\big] J_k^T &= V_{k|k} + J_k E\big[\hat{x}_{k+1|k}\hat{x}_{k+1|k}^T\big] J_k^T \\ V_{k|T} + J_k \Big(E\big[x_{k+1}x_{k+1}^T\big] - V_{k+1|T} \Big) J_k^T &= V_{k|k} + J_k \Big(E\big[x_{k+1}x_{k+1}^T\big] - V_{k+1|k} \Big) J_k^T \\ V_{k|T} &= V_{k|k} + J_k \Big[E\big[x_{k+1}x_{k+1}^T\big] - V_{k+1|k} - E\big[x_{k+1}x_{k+1}^T\big] + V_{k+1|T} \Big] J_k^T \\ V_{k|T} &= V_{k|k} + J_k \Big[V_{k+1|T} - V_{k+1|k} \Big] J_k^T \end{aligned} \tag{2.67}$$

Equations (2.63)-(2.67) comprise the set of smoothed estimate update equations. Smoothed estimates at sample k are dependent on the Kalman filter estimates at sample k and $k+1$ as well as previous smoothed estimates at sample $k+1$; thus, the Kalman smoother iterates backward from the sample T to sample 1. The Kalman smoother equations are summarized below.

$$J_k = V_{k|k} A^T V_{k+1|k}^{-1} \tag{2.68a}$$

$$\hat{x}_{k|T} = \hat{x}_{k|k} + J_k \Big(\hat{x}_{k+1|T} - B u_k - A \hat{x}_{k|k} \Big) \tag{2.68b}$$

$$V_{k|T} = V_{k|k} + J_k \Big[V_{k+1|T} - V_{k+1|k} \Big] J_k^T \tag{2.68c}$$

Lastly, recall that the objective function of the Maximization step contains a cross-sample covariance term $V_{k,k-1|T}$. This term must also be updated and solved in the Expectation step. Since this term is also a backward recursion, it begins from the Tth sample, allowing us to use both equations (2.58d) and (2.68b).

Consider

$$
\begin{aligned}
V_{k,k-1|k} &= E\left[e_{k|k}e_{k-1|k}^T\right] \\
&= E\left[\left(x_k - \hat{x}_{k|k-1} - K_k(y_k - C\hat{x}_{k|k-1} - Du_k)\right)\left(x_{k-1} - \hat{x}_{k-1|k-1} - J_{k-1}(\hat{x}_{k|T} - \hat{x}_{k|k-1})\right)^T\right] \\
&= E\big[\left(e_{k|k-1} - K_k(y_k - C\hat{x}_{k|k-1} - Du_k)\right) \\
&\quad \left(e_{k-1|k-1} - J_{k-1}(\hat{x}_{k|k-1} + K_k(y_k - C\hat{x}_{k|k-1} - Du_k) - \hat{x}_{k|k-1})\right)^T\big] \\
&= E\left[\left(e_{k|k-1} - K_k(y_k - C\hat{x}_{k|k-1} - Du_k)\right)\left(e_{k-1|k-1} - J_{k-1}K_k(y_k - C\hat{x}_{k|k-1} - Du_k)\right)^T\right] \\
&= E\left[\left(e_{k|k-1} - K_k(Ce_{k|k-1} + v_k)\right)\left(e_{k-1|k-1} - J_{k-1}K_k(Ce_{k|k-1} + v_k)\right)^T\right] \\
&= V_{k,k-1|k-1} - V_{k|k-1}C^TK_k^TJ_{k-1}^T - K_kCV_{k,k-1|k-1} + K_kCV_{k|k-1}C^TK_k^TJ_{k-1}^T + K_kRK_k^TJ_{k-1}^T \\
&= (I - K_kC)V_{k,k-1|k-1} - V_{k|k-1}C^TK_k^TJ_{k-1}^T + K_k(CV_{k|k-1}C^T + R)K_k^TJ_{k-1}^T \\
&= (I - K_kC)AV_{k-1|k-1} \\
V_{T,T-1|T} &= (I - K_TC)AV_{T-1|T-1} \qquad (2.69)
\end{aligned}
$$

Equations (2.58d) and (2.68b) could be used simultaneously in line two because we assume $k = T$. Cross-terms were removed because both $e_{k-1|k-1}$ and $e_{k|k-1}$ are independent of v_k. Since $K_k = V_{k|k-1}C^T(CV_{k|k-1}C^T + R)^{-1}$, the last two terms in the second to last line cancelled out. Finally, $V_{k,k-1|k-1} = AV_{k-1|k-1}$ allowed the final conclusion in (2.69). But equation (2.69) is only the initial condition, next we solve for the actual backward recursion beginning with (2.66)

$$e_{k|T} + J_k\hat{x}_{k+1|T} = e_{k|k} + J_k\hat{x}_{k+1|k} \qquad (2.70)$$

$$e_{k-1|T} + J_{k-1}\hat{x}_{k|T} = e_{k-1|k-1} + J_{k-1}\hat{x}_{k|k-1} \qquad (2.71)$$

and substitute in (2.58a)

$$e_{k|T} + J_k\hat{x}_{k+1|T} = e_{k|k} + J_k(A\hat{x}_{k|k} + Bu_k) \qquad (2.72)$$

$$e_{k-1|T} + J_{k-1}\hat{x}_{k|T} = e_{k-1|k-1} + J_{k-1}(A\hat{x}_{k-1|k-1} + Bu_{k-1}) \qquad (2.73)$$

$$E\left[\left(e_{k|T}+J_k\hat{x}_{k+1|T}\right)\left(e_{k-1|T}+J_{k-1}\hat{x}_{k|T}\right)^T\right]$$
$$=E\left[\left(e_{k|k}+J_k(A\hat{x}_{k|k}+Bu_k)\right)\left(e_{k-1|k-1}+J_{k-1}(A\hat{x}_{k-1|k-1}+Bu_{k-1})\right)^T\right]$$

$$\begin{aligned}
&V_{k,k-1|T}+J_kE\left[\hat{x}_{k+1|T}\hat{x}_{k|T}^T\right]J_{k-1}^T\\
&\quad=E\left[e_{k|k}e_{k-1|k-1}^T\right]+J_kAE\left[\hat{x}_{k|k}e_{k-1|k-1}^T\right]+J_kAE\left[\hat{x}_{k|k}\hat{x}_{k-1|k-1}^T\right]A^TJ_{k-1}^T\\
&\qquad+J_kA\hat{x}_{k|k}u_{k-1}^TB^TJ_{k-1}^T+J_kBu_k\hat{x}_{k-1|k-1}^TA^TJ_{k-1}^T+J_kBu_ku_{k-1}^TB^TJ_{k-1}^T
\end{aligned}\tag{2.74}$$

We simplify the expectation terms separately below.

$$\begin{aligned}
&E\left[\hat{x}_{k+1|T}\hat{x}_{k|T}^T\right]\\
&=E\left[x_{k+1}x_k^T\right]-V_{k+1,k|T}\\
&=E\left[(Ax_k+Bu_k+w_k)(Ax_{k-1}+Bu_{k-1}+w_{k-1})^T\right]-V_{k+1,k|T}\\
&=AE[x_kx_{k-1}^T]A^T+AE[x_k]u_{k-1}^TB^T+AE[x_kw_{k-1}^T]+Bu_kE[x_{k-1}^T]A^T+Bu_ku_{k-1}^TB^T-V_{k+1,k|T}\\
&=AE[x_kx_{k-1}^T]A^T+AE[x_k]u_{k-1}^TB^T+AE[(Ax_{k-1}+Bu_{k-1}+w_{k-1})w_{k-1}^T]+Bu_kE[x_{k-1}^T]A^T\\
&\quad+Bu_ku_{k-1}^TB^T-V_{k+1,k|T}\\
&=AE[x_kx_{k-1}^T]A^T+AE[x_k]u_{k-1}^TB^T+AQ+Bu_kE[x_{k-1}^T]A^T+Bu_ku_{k-1}^TB^T-V_{k+1,k|T}
\end{aligned}\tag{2.75}$$

$$\begin{aligned}
&E\left[e_{k|k}e_{k-1|k-1}^T\right]\\
&=E\left[(x_k-\hat{x}_{k|k})e_{k-1|k-1}^T\right]\\
&=E\left[\left(e_{k|k-1}-K_k(y_k-Du_k-C\hat{x}_{k|k-1})\right)e_{k-1|k-1}^T\right]\\
&=E\left[\left(e_{k|k-1}-K_k(Cx_k+Du_k+v_k-Du_k-C\hat{x}_{k|k-1})\right)e_{k-1|k-1}^T\right]\\
&=V_{k,k-1|k-1}-K_kE\left[(Ce_{k|k-1}+v_k)e_{k-1|k-1}^T\right]\\
&=V_{k,k-1|k-1}-K_kCV_{k,k-1|k-1}
\end{aligned}\tag{2.76}$$

where we apply (2.58d) and recombine the state and its estimate to form $e_{k|k-1}$ in line three, expand the y_k

term in line four, combine the state term and its estimate again in five and apply the assumption that the state noise w_k is independent across time samples and from the states.

$$
\begin{aligned}
& E\left[\hat{x}_{k|k} e_{k-1|k-1}^T\right] \\
&= E\left[\left(\hat{x}_{k|k-1} + K_k(y_k - Du_k - C\hat{x}_{k|k-1})\right) e_{k-1|k-1}^T\right] \\
&= E\left[\left(\hat{x}_{k|k-1} + K_k(Ce_{k|k} + v_k)\right) e_{k-1|k-1}^T\right] \\
&= K_k C V_{k,k-1|k-1}
\end{aligned} \tag{2.77}
$$

where we apply the same equation (2.58d) and simplifications as in the previous set of equations and use the same assumption of state noise independence.

$$
\begin{aligned}
& E\left[\hat{x}_{k|k} \hat{x}_{k-1|k-1}^T\right] \\
&= E\left[\left(\hat{x}_{k|k-1} + K_k(Ce_{k|k} + v_k)\right) \hat{x}_{k-1|k-1}^T\right] \\
&= E\left[\hat{x}_{k|k-1} \hat{x}_{k-1|k-1}^T\right] \\
&= E\left[x_k x_{k-1}^T\right] - V_{k,k-1|k-1}
\end{aligned} \tag{2.78}
$$

Now, we can substitute in equations (2.75)-(2.78) into (2.74).

$$
\begin{aligned}
& V_{k,k-1|T} \\
&= E\left[e_{k|k} e_{k-1|k-1}^T\right] + J_k A E\left[\hat{x}_{k|k} e_{k-1|k-1}^T\right] + J_k A E\left[\hat{x}_{k|k} \hat{x}_{k-1|k-1}^T\right] A^T J_{k-1}^T \\
&\quad + J_k A \hat{x}_{k|k} u_{k-1}^T B^T J_{k-1}^T + J_k B u_k \hat{x}_{k-1|k-1}^T A^T J_{k-1}^T + J_k B u_k u_{k-1}^T B^T J_{k-1}^T \\
&\quad - J_k E\left[\hat{x}_{k+1|T} \hat{x}_{k|T}^T\right] J_{k-1}^T \\
&= \left(V_{k,k-1|k-1} - K_k C V_{k,k-1|k-1}\right) + J_k A\left(K_k C V_{k,k-1|k-1}\right) + J_k A\left(E\left[x_k x_{k-1}^T\right] - V_{k,k-1|k-1}\right) A^T J_{k-1}^T \\
&\quad + J_k A \hat{x}_{k|k} u_{k-1}^T B^T J_{k-1}^T + J_k B u_k \hat{x}_{k-1|k-1}^T A^T J_{k-1}^T + J_k B u_k u_{k-1}^T B^T J_{k-1}^T \\
&\quad - J_k\left(AE[x_k x_{k-1}^T]A^T + AE[x_k] u_{k-1}^T B^T + AQ + B u_k E[x_{k-1}^T] A^T + B u_k u_{k-1}^T B^T - V_{k+1,k|T}\right) J_{k-1}^T \\
&= \left(V_{k,k-1|k-1} - K_k C V_{k,k-1|k-1}\right) + J_k A\left(K_k C V_{k,k-1|k-1}\right) - J_k A V_{k,k-1|k-1} A^T J_{k-1}^T \\
&\quad - J_k A Q J_{k-1}^T + J_k V_{k+1,k|T} J_{k-1}^T
\end{aligned} \tag{2.79}
$$

The first term in (2.79) can be further simplified. Consider

$$\begin{aligned} V_{k,k-1|k-1} - K_k C V_{k,k-1|k-1} &= (I - K_k C) V_{k,k-1|k-1} \\ &= (I - K_k C) A V_{k-1|k-1} \\ &= (I - K_k C) V_{k|k-1} V_{k|k-1}^{-1} A V_{k-1|k-1} \\ &= V_{k|k} J_{k-1}^T \end{aligned} \tag{2.80}$$

where we apply equations (2.58e) and (2.68a) in the last line. Also, we can gather the second, third and fourth terms to simplify to

$$\begin{aligned} & J_k A \left(K_k C V_{k,k-1|k-1} - V_{k,k-1|k-1} A^T J_{k-1}^T - Q J_{k-1}^T \right) \\ &= J_k A \left(K_k C V_{k|k-1} - V_{k,k-1|k-1} A^T - Q \right) J_{k-1}^T \\ &= J_k A \left(K_k C V_{k|k-1} - V_{k|k-1} \right) J_{k-1}^T \\ &= -J_k A V_{k|k} J_{k-1}^T \end{aligned} \tag{2.81}$$

where we apply (2.58e) and (2.68a) again, along with the fact that $V_{k|k-1} = V_{k,k-1|k-1} A^T + Q$ to get the final result. Substituting these new simplifications into (2.79)

$$\begin{aligned} V_{k,k-1|T} &= V_{k|k} J_{k-1}^T - J_k A V_{k|k} J_{k-1}^T + J_k V_{k+1,k|T} J_{k-1}^T \\ &= V_{k|k} J_{k-1}^T + J_k \left(V_{k+1,k|T} - A V_{k|k} \right) J_{k-1}^T \end{aligned} \tag{2.82}$$

Chapter 3

Linear Analysis

Ultimately our goal is to apply the EM algorithm to a highly nonlinear model, but first we evaluate its efficacy using a few straightforward linear test cases. The linear test cases are often subject to state transformations so we begin with a brief, preliminary note on the effects of transformations before proceeding into the identification results. One purpose of this section is to simulate the algorithm derived in the previous section as well as the original EM algorithms proposed by Shumway *et al.*[24] and Ghahramani *et al.*[13]. Another purpose is to explore the effect of some basic experimental assumptions on their performance. In particular, we look at the impact of time-discrete data on identification accuracy and interpretation to show the significant difference between estimated continuous-time and discrete-time systems. We also look at how to handle non-uniformly sampled data.

3.1 Basic State Transformations

Given only a set of inputs and observations, a system of the form described in (2.10) varies with state transformations. Specifically, given an invertible matrix F, a linear transform that satisfies

$$x_k = F z_k$$

the system in (2.10) can be rewritten as

$$\begin{aligned} z_{k+1} &= F^{-1}AFz_k + F^{-1}Bu_k + w_k \\ y_k &= CFz_k + Du_k + v_k \\ z_0 &= F^{-1}x_0 \end{aligned} \tag{3.1}$$

The input-output behavior of (3.1) is identical for any matrix F. Hence, there are an infinite number of state parameters, $A_{EM} = F^{-1}AF$, $B_{EM} = F^{-1}B$, and $C_{EM} = CF$, that the EM algorithm can identify. Thus, we must know the transformation matrix, F, in order to reconstruct the original state parameters. One simple method of identifying this transformation matrix is through the initial state vectors. Suppose, for our simulations, we chose initial state vectors that are all linearly independent standard basis vectors, such that they form an identity matrix when concatenated. Substituting this knowledge into (3.1) illuminates the fact that the smoothed EM estimates of the initial state vectors concatenated in the same order is equivalent to the inverse transform, F^{-1}. Thus, we can deduce the actual simulation parameters from the EM output parameters with the following post-processing:

$$\begin{aligned} F &= [\hat{x}_{10}, \hat{x}_{20}, \ldots, \hat{x}_{N0}]^{-1} \\ A &= FA_{EM}F^{-1} \\ B &= FB_{EM} \\ C &= C_{EM}F^{-1} \\ \pi_1 &= F\pi_{1EM} \end{aligned} \tag{3.2}$$

where N denotes the number of sets.

3.2 State-Space Models and Variations

The EM algorithm derived in the previous section was coded and simulated in MATLAB 7.0. An arbitrary test case was chosen for the structure in (2.10) using the system matrices shown in Table 3.1. We randomly designated an input vector, u. Five sets of fifty-point data were generated with the same input and differing initial conditions, x_0; each set excited a unique initial state to one and left all other initial states at zero. We chose this set of initial conditions because it spanned $\Re^5$, simplifying state-transformation reversal, and could

be used to generate any other randomly chosen initial condition. The processed EM estimated parameters are tabulated below in Table 3.2.

Table 3.1: Discrete system simulation matrices (A,B,C,D,Q,R).

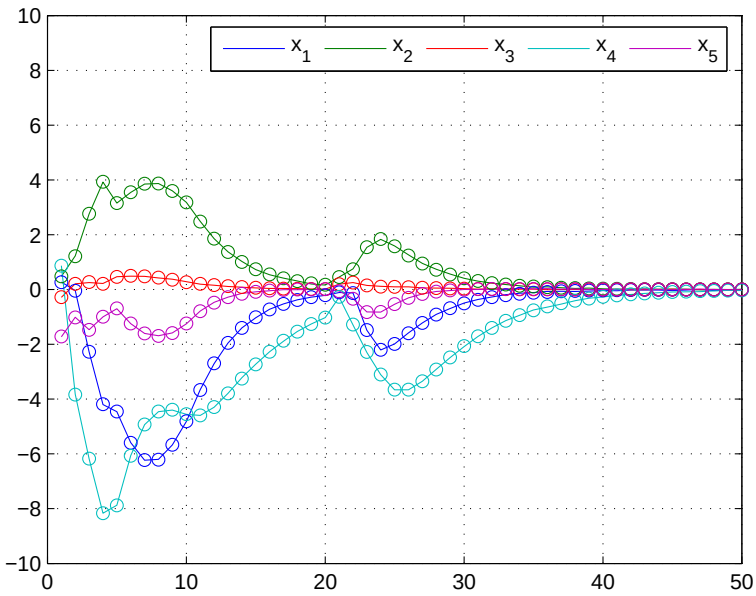
A					B		Q (10^{-5})				
0.23	-0.40	-0.07	0.22	0.82	0.90	-0.10	0.1	0	0	0	0
-0.33	0.08	-0.12	0.32	1.04	-0.41	0	0	0.1	0	0	0
-1.07	0.31	0.56	0.39	0.19	-1.63	0.12	0	0	0.1	0	0
-0.74	0.58	0.13	0.62	-0.41	-0.79	2.60	0	0	0	0.1	0
-0.14	-0.29	-0.08	0.18	1.17	2.10	0.86	0	0	0	0	0.1
C					D		R (10^{-3})				
0.20	-3.00	1.00	0	0	0.25	-0.75	0.1	0	0	0	0
0	0.90	0	-0.84	1.00	1.60	0.82	0	0.1	0	0	0
-0.56	0	0	0	0.61	0.94	-0.01	0	0	0.1	0	0
0	0	2.34	0.78	0	2.87	0	0	0	0	0.1	0
-1.58	0	0.09	0	0	-0.40	1.09	0	0	0	0	0.1

Table 3.2: EM estimated system matrices (A,B,C,D,Q,R).

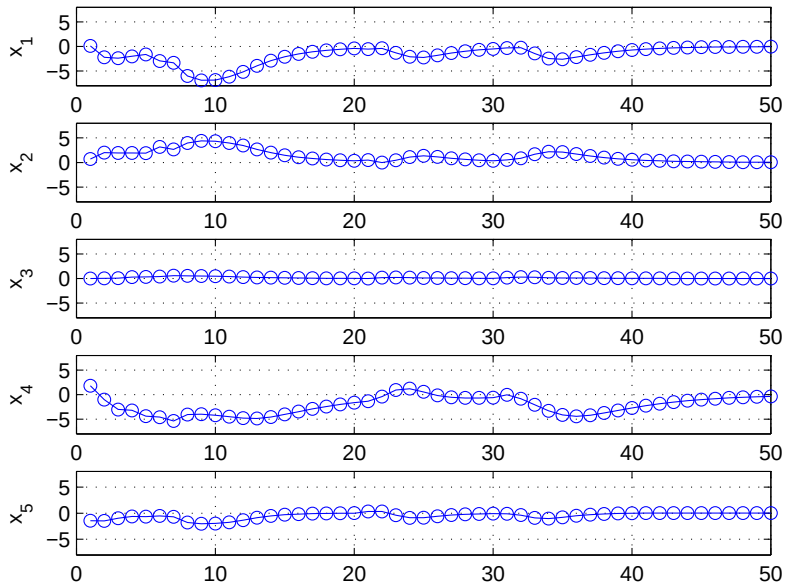
A					B		Q (10^{-4})				
0.21	-0.40	-0.07	0.22	0.84	0.90	-0.09	0.15	0	0	0	0
-0.35	0.08	-0.12	0.32	1.05	-0.42	0.02	0	0.46	0	0	0
-1.06	0.31	0.56	0.39	0.20	-1.63	0.09	0	0	0.12	0	0
-0.76	0.59	0.14	0.62	-0.41	-0.80	2.64	0	0	0	0.26	0
-0.15	-0.30	-0.08	0.19	1.19	2.11	0.87	0	0	0	0	0.06
C					D		R (10^{-3})				
0.20	-2.97	1.02	0.03	0.00	0.19	-0.75	0.12	0	0	0	0
0.00	0.89	-0.01	-0.84	0.99	1.62	0.82	0	0.12	0	0	0
-0.56	-0.00	0.00	0.00	0.61	0.94	-0.00	0	0	0.09	0	0
-0.02	-0.00	2.33	0.79	-0.00	2.88	0.02	0	0	0	0.09	0
-1.58	0.02	0.10	0.01	0.01	-0.43	1.09	0	0	0	0	0.10

A comparison of this parameter set with the actual parameter set shows that the estimated parameters consistently match the simulation parameters to within the second decimal place. The covariances observed in R are on approximately the same order of magnitude. There is more error in the covariance matrix Q but this may result from a combination of the estimation accuracy and small magnitude of the actual covariances. The precision of these results persists with other test cases. The accuracy of this result is unsurprising given that our test case is a linear system and can be matched exactly by our linear model. Graphical output shown in Fig. 3.1 further reaffirms our numerical results.

All training data sets were reconstructed using the post-processed EM output parameters. Each individual reconstruction was initialized with the smoothed Kalman estimate of its specific initial state. The final



(a) Re-estimation comparison



(b) Prediction comparison

Figure 3.1: Comparison of EM identified system (circles) with simulation system (solid line).

average over all the training data sets was re-constructed using the system parameter π_1 as the initial state estimate. The EM model can exactly reconstruct the training data; but more importantly, when given a randomly generated initial state vector and a new input sequence, u, the EM model can predict the output dynamics with the same precision.

The EM algorithm we have derived and simulated thus far can be varied in many ways. Our derivation is general because it has the largest unknown parameter set; all subsequent variations can be logically deduced from it without reworking the entire derivation.

One of the earliest variants was proposed by Shumway [24] in 1982. The proposed state-space model was

$$\begin{aligned} x_{k+1} &= Ax_k + w_k \\ y_k &= Cx_k + v_k \end{aligned} \tag{3.3}$$

where C was assumed to be some known matrix. Except for the lack of input matrix parameters, B and D, the Expectation step of the algorithm for this model is identical to that derived in the previous section. Thus, upon equating the parameters B and D in equations (2.58a)-(2.58e), (2.68a)-(2.68c), and (2.82) to zero, the Kalman filtering and smoothing equations still apply. The Maximization step, however, is changed because C is known and simplified by the lack of input matrix parameters, B and D. The partial derivatives for π_1 and V_1 are unaffected, but the absence of the B partial derivative, (2.37e), from the simultaneous equations simplifies the update equation of A to a single matrix division of covariances; this modification is accomplished by equating B to zero in (2.37d) and solving for A. The new Q update similarly simplifies by deleting all terms containing B from (2.37f). Because D does not exist and C is statically allocated, both the C and D partial derivatives, (2.37a) and (2.37b), disappear from the set of simultaneous equations. The remaining partial of R is reevaluated for this case by eliminating all terms containing D from (2.37c) and solving for R; no other simplifications apply because C isn't updated. The modified Maximization equations

are:

$$A_{new} = \left(\sum_{j=1}^{N} \sum_{k=2}^{T} P_{j(k,k-1)} \right) \left(\sum_{j=1}^{N} \sum_{k=2}^{T} P_{j(k-1)} \right)^{-1} \tag{3.4}$$

$$Q_{new} = \frac{1}{(N)(T-1)} \sum_{j=1}^{N} \sum_{k=2}^{T} \left[(P_{jk} + \hat{x}_{jk}\hat{x}_{jk}^T) - A_{new}(P_{j(k-1,k)} + \hat{x}_{j(k-1)}\hat{x}_{jk}^T) \right] \tag{3.5}$$

$$R_{new} = \frac{1}{(N)(T)} \sum_{j=1}^{N} \sum_{k=1}^{T} \left[y_{jk}y_{jk}^T - C_{new}\hat{x}_{jk}y_{jk}^T - y_{jk}\hat{x}_{jk}^T C_{new}^T + C_{new}P_{jk}C_{new}^T \right] \tag{3.6}$$

Our test case for this model was generated using the system matrices

Table 3.3: Discrete system simulation matrices (A,C,Q,R), C known.

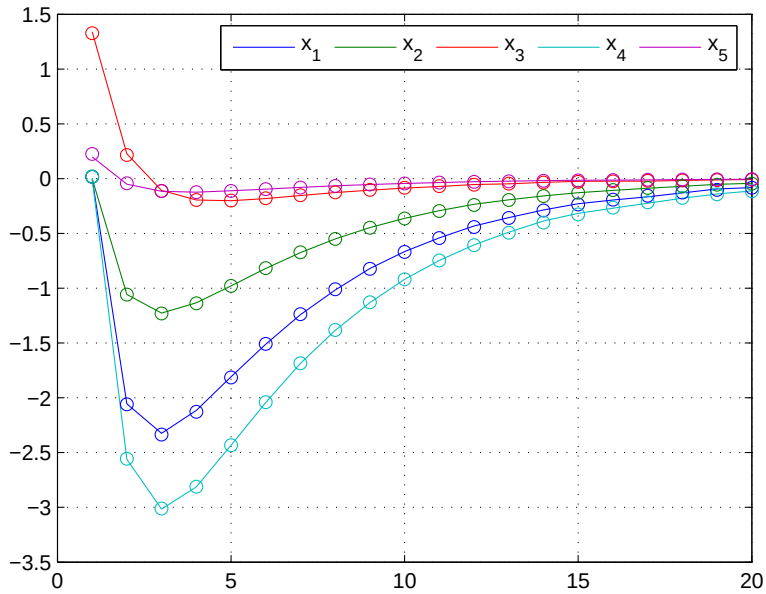
A					Q (10^{-5})				
0.23	-0.40	-0.07	0.22	0.82	0.1	0	0	0	0
-0.33	0.08	-0.12	0.32	1.04	0	0.1	0	0	0
-1.07	0.31	0.56	0.39	0.19	0	0	0.1	0	0
-0.74	0.58	0.13	0.62	-0.41	0	0	0	0.1	0
-0.14	-0.29	-0.08	0.18	1.17	0	0	0	0	0.1
C					R (10^{-3})				
0	3.00	1.00	0	0	0.1	0	0	0	0
0	0.90	0	0.84	1.00	0	0.1	0	0	0
1.34	0	0	0	0.61	0	0	0.1	0	0
0	0	2.34	0.08	0	0	0	0	0.1	0
0.20	0	0.09	0	0	0	0	0	0	0.1

Five sets of twenty-point data were generated with initial conditions consisting of the standard basis. Because C is known, full-rank, and statically allocated in the EM program for this system model, no linear transformation, except the identity, is possible. The EM estimated parameters are tabulated below.

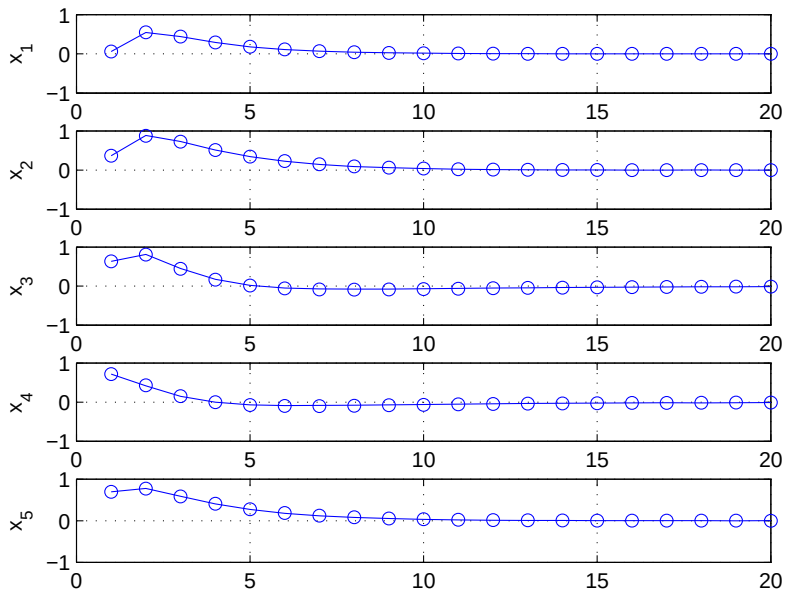
Table 3.4: EM estimated system matrices (A,C,Q,R), C known.

A					Q (10^{-4})				
0.22	-0.39	-0.08	0.22	0.82	0.09	-0.04	0.05	0.26	-0.10
-0.32	0.08	-0.12	0.32	1.03	-0.04	0.08	0.01	-0.22	0.12
-1.07	0.31	0.55	0.39	0.18	0.05	0.01	0.10	0.04	0.02
-0.79	0.60	0.13	0.64	-0.42	0.26	-0.22	0.04	0.99	-0.46
-0.13	-0.29	-0.09	0.18	1.17	-0.10	0.10	0.02	-0.46	0.23
C (statically assigned)					R (10^{-3})				
0	3.00	1.00	0	0	0.04	0.00	0.00	-0.03	0.01
0	0.90	0	0.84	1.00	0.00	0.09	0.01	-0.03	0.02
1.34	0	0	0	0.61	0.00	0.01	0.10	-0.02	-0.02
0	0	2.34	0.08	0	-0.03	-0.03	-0.02	0.04	0.00
0.20	0	0.09	0	0	0.01	0.02	-0.02	0.00	0.13

Graphical comparative plots follow.



(a) Re-estimation comparison



(b) Prediction comparison

Figure 3.2: Comparison of EM identified system (circles) with simulation system (solid line).

As in the previous system, the estimated state matrix, A, matches the actual matrix to within the second decimal place. The results are highly accurate and graphical output reaffirms our numerical results. The EM model accurately reconstructs the training data; furthermore, it can predict system outputs with the same precision when given a new initial state.

In 1996, Ghahramani *et al.*[13] expanded Shumway's model to include an unknown observation matrix, C. This addition of C to the unknown parameter set does not affect the Kalman filter and smoother that Shumway used but it does slightly change his parameter updates. Parameters V_1 and π_1 remain unaffected, but the partial derivative of C, (2.37a) with D set to zero, is now added to the set of simultaneous equations. Because D is trivial, it's partial derivative can be discarded and (2.37a) can be set to 0 in order to solve directly for C. Also, because C is recalculated for each iteration, it's definition can be used to further simplify R's update to (2.40) with D set to zero. The A and Q update equations remain the same as in Shumway's model. The modified Maximization equations are tabulated below.

$$C_{new} = \left(\sum_{j=1}^{N} \sum_{k=2}^{T} y_{jk} \hat{x}_{jk}^{T} \right) \left(\sum_{j=1}^{N} \sum_{k=2}^{T} P_{jk} \right)^{-1} \tag{3.7}$$

$$R_{new} = \frac{1}{(N)(T)} \sum_{j=1}^{N} \sum_{k=1}^{T} \left[y_{jk} y_{jk}^{T} - C_{new} \hat{x}_{jk} y_{jk}^{T} \right] \tag{3.8}$$

A test case was arbitrarily chosen using the system matrices

Table 3.5: Discrete system simulation matrices (A,C,Q,R), C unknown.

A					Q (10^{-5})				
0.91	0.36	0.08	-0.21	-0.97	0.1	0	0	0	0
0.35	1.24	0.20	-0.43	-1.80	0	0.1	0	0	0
1.33	-0.84	0.49	-0.35	0.27	0	0	0.1	0	0
0.98	-0.91	-0.19	0.52	0.69	0	0	0	0.1	0
0.14	0.49	0.16	-0.28	-0.68	0	0	0	0	0.1
C					R (10^{-3})				
0.17	0	1.80	-0.90	0	0.1	0	0	0	0
0	1.00	0	0.30	2.00	0	0.1	0	0	0
-0.23	0	0.77	0	-0.44	0	0	0.1	0	0
0	0.90	0	0.50	0	0	0	0	0.1	0
2.00	-0.90	0	0.60	0	0	0	0	0	0.1

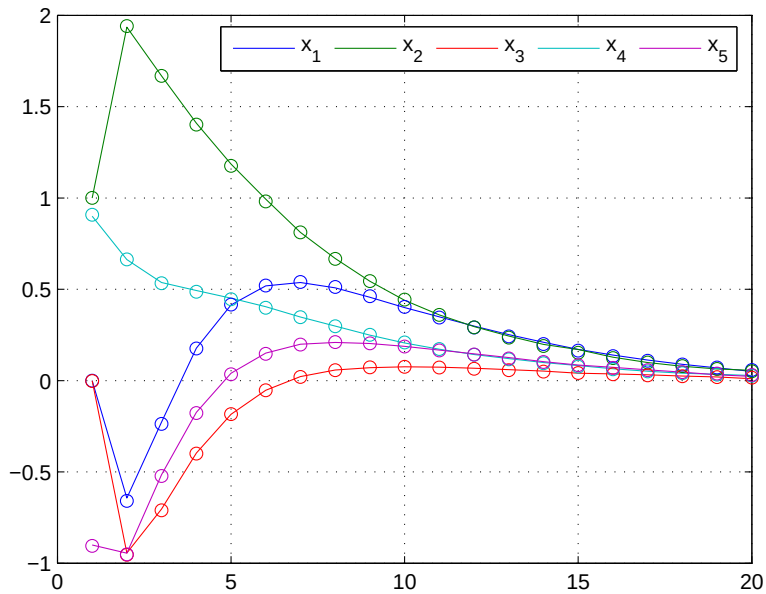
Five sets of twenty-point data were generated with initial conditions of standard basis vectors. Because C is estimated as well now, the identified system is related to the original system via a state transformation.

Post-processing equations (3.2) were applied to the data before it was tabulated below.

Table 3.6: EM estimated system matrices (A,C,Q,R), C unknown.

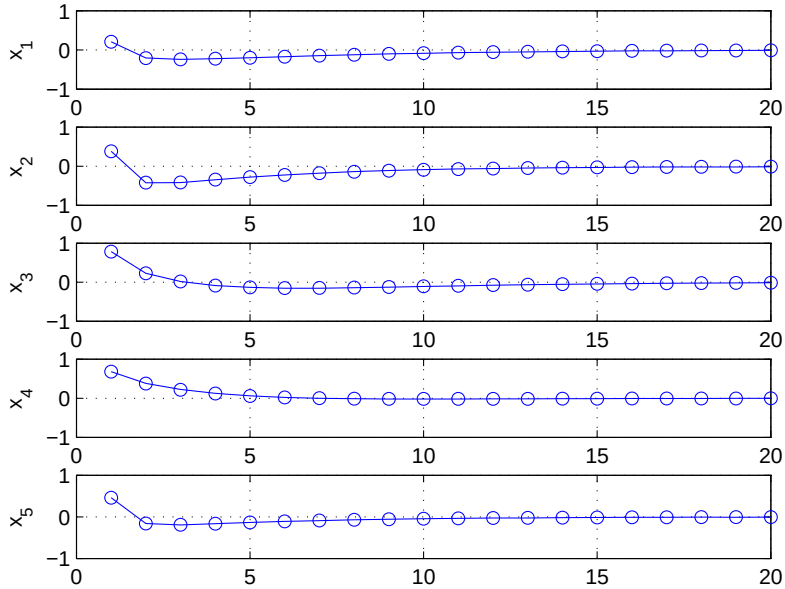
A					Q (10^{-3})				
0.90	0.37	0.08	-0.21	-0.98	0.01	0	0	0	0
0.33	1.25	0.20	-0.43	-1.80	0	0.08	0	0	0
1.33	-0.85	0.49	-0.35	0.27	0	0	0.12	0	0
0.99	-0.92	-0.19	0.51	0.69	0	0	0	0.04	0
0.13	0.50	0.15	-0.27	-0.68	0	0	0	0	0.08
C					R (10^{-3})				
0.16	-0.00	1.82	-0.90	0.01	0.12	0	0	0	0
0.00	1.00	0.02	0.31	2.00	0	0.11	0	0	0
-0.23	-0.00	0.76	0.00	-0.44	0	0	0.07	0	0
-0.01	0.91	0.01	0.50	-0.00	0	0	0	0.06	0
1.99	-0.91	-0.00	0.60	0.00	0	0	0	0	0.10

Visual comparison of the simulation data with the reconstructed data as well as new simulation data with predicted data confirm the accuracy of the EM estimates.



(a) Re-estimation comparison

Figure 3.3: Comparison of EM identified system (circles) with simulation system (solid line).



(b) Prediction comparison

Figure 3.3: Comparison of EM identified system (circles) with simulation system (solid line).

Our final variation of the EM algorithm is particularly relevant to our theoretical biological model. We describe a system that has a known observation matrix and an external input that affects the future state dynamics but not the immediate output.

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k + w_k \\ y_k &= Cx_k + v_k \end{aligned} \tag{3.9}$$

The Kalman filtering and smoothing equations for this system are (2.58a)-(2.58e), (2.68a)-(2.68c), and (2.82) with D set equal to zero. The addition of the B parameter to the system brings (2.37e), into the set of simultaneous equations. As a result, the partials of A and B become re-coupled as in our first system model and the derivation of their simultaneous solution is as in (2.39). Moreover, because both A and B are now being updated, the Q update equation can also further simplify to the solution derived in (2.41). Lastly, since C is assumed and D is trivially zero, the update for R reverts to Shumway's R update in equation (3.4). All other parameters, π_1 and V_1 remain unchanged. The modified updates for this system are below.

$$\begin{bmatrix} A_{new} & B_{new} \end{bmatrix} = \begin{bmatrix} \sum_{j=1}^{N}\sum_{k=2}^{T} P_{j(k,k-1)} & \sum_{j=1}^{N}\sum_{k=2}^{T} \hat{x}_{jk} u_{j(k-1)}^{T} \end{bmatrix} \begin{bmatrix} \sum_{j=1}^{N}\sum_{k=2}^{T} P_{j(k-1)} & \sum_{j=1}^{N}\sum_{k=2}^{T} \hat{x}_{j(k-1)} u_{j(k-1)}^{T} \\ \sum_{j=1}^{N}\sum_{k=2}^{T} u_{j(k-1)} \hat{x}_{j(k-1)}^{T} & \sum_{j=1}^{N}\sum_{k=2}^{T} u_{j(k-1)} u_{j(k-1)}^{T} \end{bmatrix}^{-1} \tag{3.10}$$

$$Q_{new} = \frac{1}{(N)(T-1)} \sum_{j=1}^{N}\sum_{k=2}^{T} \left[(P_{jk} + \hat{x}_{jk}\hat{x}_{jk}^{T}) - A_{new}(P_{j(k-1,k)} + \hat{x}_{j(k-1)}\hat{x}_{jk}^{T}) - B_{new} u_{j(k-1)} \hat{x}_{jk}^{T} \right] \tag{3.11}$$

$$R_{new} = \frac{1}{(N)(T)} \sum_{j=1}^{N}\sum_{k=1}^{T} \left[y_{jk} y_{jk}^{T} - C_{new}\hat{x}_{jk} y_{jk}^{T} - y_{jk}\hat{x}_{jk}^{T} C_{new}^{T} + C_{new} P_{jk} C_{new}^{T} \right] \tag{3.12}$$

The simulation parameters and identified parameters are shown below.

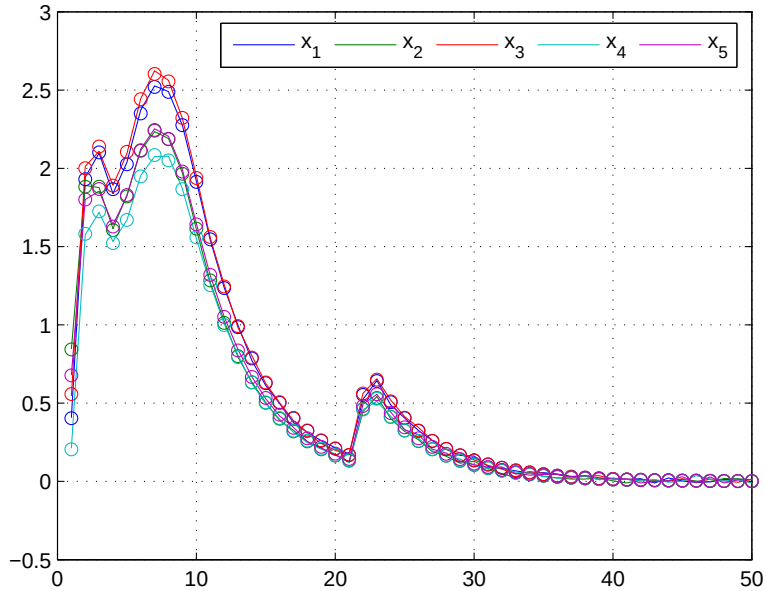
Table 3.7: Discrete system simulation (A,B,C,Q,R).

A					B		Q (10^{-4})				
0.61	-0.11	-0.00	-0.03	0.22	0.50	0.66	0.1	0	0	0	0
0.52	0.38	-0.08	-0.31	0.44	0.90	0.34	0	0.1	0	0	0
0.50	-0.45	0.50	-0.08	0.60	0.82	0.29	0	0	0.1	0	0
0.46	-0.22	0.02	0.26	0.25	0.64	0.34	0	0	0	0.1	0
0.39	-0.09	-0.07	-0.24	0.91	0.82	0.53	0	0	0	0	0.1
C					-		R (10^{-3})				
0.42	0.84	0.50	0.19	0.70			0.1	0	0	0	0
0.85	0.02	0.71	0.68	0.38			0	0.1	0	0	0
0.53	0.68	0.43	0.30	0.86			0	0	0.1	0	0
0.20	0.38	0.30	0.54	0.85			0	0	0	0.1	0
0.67	0.83	0.19	0.15	0.59			0	0	0	0	0.1

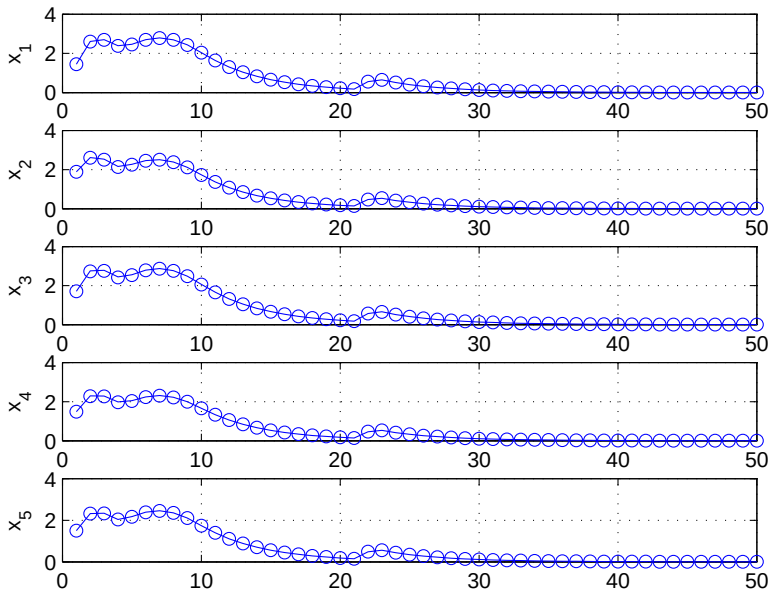
Table 3.8: EM estimated system matrices (A,B,C,Q,R).

A					B		Q (10^{-4})				
0.63	-0.12	-0.01	-0.03	0.23	0.51	0.67	0.03	0.01	0.04	-0.05	0.09
0.53	0.38	-0.09	-0.30	0.43	0.90	0.33	0.01	0.01	0.01	-0.03	0.05
0.50	-0.42	0.50	-0.0768	0.58	0.83	0.31	0.04	0.01	0.05	-0.07	0.11
0.46	-0.25	0.03	0.2598	0.25	0.63	0.32	-0.05	-0.03	-0.07	0.15	-0.20
0.38	-0.09	-0.06	-0.24	0.89	0.81	0.53	0.09	0.05	0.11	-0.20	0.30
C (statically assigned)					-		R (10^{-4})				
0.42	0.84	0.50	0.19	0.70			0.90	-0.19	-0.32	-0.15	-0.23
0.85	0.02	0.71	0.68	0.38			-0.19	0.95	-0.13	-0.16	-0.18
0.53	0.68	0.43	0.30	0.86			-0.32	-0.13	0.78	-0.38	-0.12
0.20	0.38	0.30	0.54	0.85			-0.15	-0.16	-0.38	0.69	-0.12
0.67	0.83	0.19	0.15	0.59			-0.23	-0.18	-0.12	-0.12	0.83

The EM-identified parameters of this simulation sample were a close match with precision to that previously observed in the other test cases. Plots verify these results.



(a) Re-estimation comparison



(b) Prediction comparison

Figure 3.4: Comparison of EM identified system (circles) with simulation system (solid line).

3.3 Continuous and Discrete System Differences

The biological systems we intend to study evolve in continuous time. Our model, however, is firmly entrenched in the discrete time domain because the data we get from experimentation will of necessity be sampled. This conversion from continuous-time to discrete-time is by no means trivial; hence, we consider, in this section, the effect of sampling on our identified dynamics matrices.

Consider a continuous system described by equation (3.3) with the state matrices shown below. Conversion of this system into discrete time using a zero-order hold, or a sample-and-hold, with a reasonable sample period of 0.25 s has a dramatic effect on the system parameters. Further, there are significant magnitude changes, as well as multiple sign reversals throughout the state dynamics matrix A. But despite these differences, the two models generate the same dynamics as shown in Figure 3.5.

Table 3.9: Comparison of continuous and discrete system matrices (A,C).

Continuous Time State-Space Model									
A					C				
-4.62	4.44	0.25	-0.58	-4.45	0.20	3.00	1.00	0	0
-1.29	-1.39	0.08	0.26	0.27	0	0.90	0	0.84	1.00
-7.94	19.80	-2.84	0.43	-22.48	0.56	0	0	0	0.61
-6.55	11.92	1.05	-3.37	-12.92	0	0	2.34	0.78	0
-0.42	0.56	-0.27	0.42	-2.22	1.58	0	0.09	0	0
Discrete Time State-Space Model									
A					C				
0.28	0.48	0.04	-0.06	-0.45	0.20	3.00	1.00	0	0
-0.20	0.70	0.01	0.05	0.04	0	0.90	0	0.84	1.00
-1.09	2.33	0.58	0.07	-2.57	0.56	0	0	0	0.61
-0.89	1.46	0.16	0.47	-1.55	0	0	2.34	0.78	0
-0.07	0.06	-0.03	0.06	0.61	1.58	0	0.09	0	0

The discrete data, shown in circles in the figure, was collected and input to the EM algorithm as observations. Consequently, the system that the algorithm identifies is the discrete state-space system. As the original simulation matrices show, the continuous-time and the discrete-time systems are not comparable. However, in experimentation, we know the sample period; hence, we can convert the EM-identified system from discrete time to continuous time and then compare the original simulation system with the estimated, continuous-time system. The EM system matrices are tabulated below in Table 3.10.

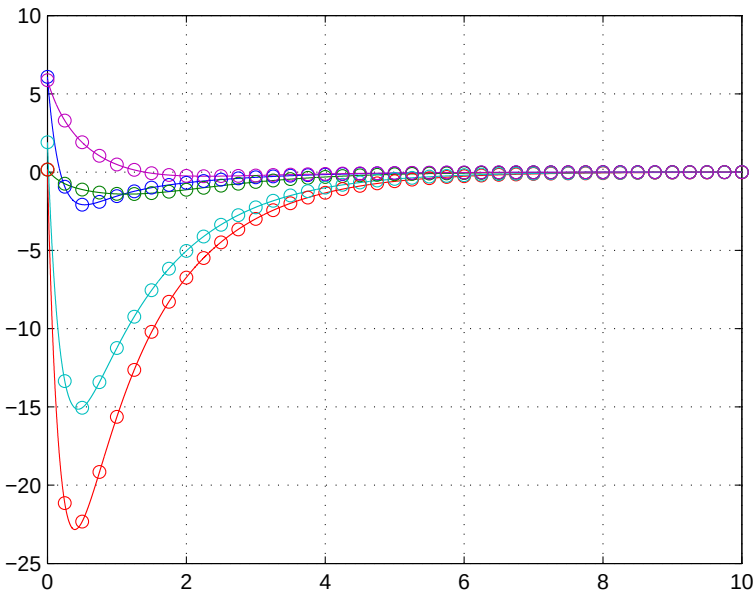


Figure 3.5: Comparison of dynamics from continuous system (solid line) and discrete (circles) system.

Table 3.10: EM estimated system matrices for discrete and continuous time.

Continuous Time State-Space Model									
A					C				
-4.79	4.55	0.33	-0.68	-4.64	0.20	2.99	0.99	0.01	-0.01
-1.29	-1.41	0.12	0.23	0.28	0.01	0.91	-0.01	0.85	0.99
-8.01	19.92	-2.87	0.43	-22.48	0.56	-0.00	-0.00	0.0026	0.62
-6.71	11.86	1.13	-3.47	-12.98	0.01	-0.00	2.34	0.78	-0.01
-0.37	0.56	-0.28	0.43	-2.13	1.58	0.00	0.08	0.00	0.01
Discrete Time State-Space Model									
A					C				
0.27	0.48	0.05	-0.07	-0.47	0.20	2.99	0.99	0.01	-0.01
-0.20	0.70	0.01	0.05	0.04	0.01	0.91	-0.01	0.85	0.99
-1.10	2.32	0.58	0.07	-2.57	0.56	-0.00	-0.00	0.00	0.62
-0.89	1.44	0.17	0.46	-1.55	0.01	-0.00	2.34	0.7766	-0.01
-0.06	0.06	-0.03	0.06	0.62	1.58	0.00	0.08	0.00	0.01

The EM-identified parameters are very similar in both the continuous and discrete case. Closer analysis shows that while the discrete-time estimate of A is accurate to within the second-decimal place at worst, the continuous-time estimate is precise to the first-decimal place at worst, losing a degree of magnitude or more in accuracy. The observation matrix C is unchanged during continuous-to-discrete and discrete-to-continuous time conversions so it is consistently accurate to within the second-decimal. Thus, the EM-identified system is still reasonably comparable to the actual, simulated system, but only if sampling is taken into account

and reversed.

The matrix changes inflicted by sampling and discretization significantly complicate, and even nullify, certain assumptions on the estimated parameters' structures. A specific example would be the assumption of sparsity on the state-to-state interaction matrix A. In continuous time, a matrix may be very sparse with few cogent connections; but there is no guarantee of sparsity once it is sampled into discrete-time. The continuous-to-discrete time transformation is by no means linear, hence any attempts to preempt zeros in the interaction matrix may in fact inhibit full identification of the system. Consider the sparse matrix shown in the table below. When converted into discrete time, the zeros become less and less obvious as the sampling period increases.

Consider the first column of the following continuous-time matrix, shown in Table 3.11. If this matrix represented gene-gene interaction, we would say that gene 1 affected genes 3 and 4 and had no effect on genes 2 and 5. In the first sampled matrix, however, we would conclude that the biggest affect of gene 1 was on gene 2. This erroneous conclusion is due to looking at the discrete-time propagation matrix of a continuous-time system. In this case, the results observed will depend mostly on the sampling time considered and not on the true interactions that occur. In addition, if zeros had been preempted in the A parameter matrix before the EM algorithm was run, the result would have been erroneous. This error could have propagated into the continuous domain and become worse.

Table 3.11: Comparison of matrix sparsity as sampling period increases.

Continuous Time Matrix				
-4.22	1.27	2.63	-1.63	-3.17
0.00	4.34	0.00	1.00	-14.90
2.13	8.50	-4.78	-1.06	-20.56
1.44	-1.92	0.60	-5.68	4.80
0.00	3.48	0.00	0.00	-8.49
Sampled Matrix (T = 0.1 s)				
0.0361	-0.2004	0.0412	-0.0534	0.0502
0.0353	-0.1042	0.0311	-0.0034	-0.1065
0.0254	-0.3378	0.0344	-0.0758	0.3469
0.0270	0.0058	0.0265	-0.0069	-0.1733
0.0180	0.0041	0.0147	0.0103	-0.1561

Sampled Matrix (T = 0.5 s)				
0.1721	0.2053	0.1488	-0.0676	-1.0291
0.0556	0.6755	0.0368	0.1471	-1.5918
0.1295	0.1176	0.1632	-0.0273	-1.2328
0.0828	0.1188	0.0678	0.0281	-0.2755
0.0181	0.3730	0.0113	0.0642	-0.6985
Sampled Matrix (T = 1 s)				
0.6659	0.1610	0.1655	-0.1019	-0.4314
0.0060	1.2749	0.0029	0.0893	-1.1714
0.1327	0.5493	0.6360	-0.0476	-1.4645
0.0919	-0.0872	0.0473	0.5524	0.2470
0.0006	0.2771	0.0003	0.0124	0.2593

3.4 Effects of Varying Sampling Periods

The previous section briefly broached the subject of sampling period as a means of traversing between the continuous time domain and the discrete time domain. This section will discuss it in more detail and show that it is a consequential part of the system identification process. In the previous section, the conversion from the continuous time domain to the discrete was actualized via a reasonable sampling period of 0.25 s. For our simulated system, this sampling period was sufficient to capture the necessary dynamics that identified the system to the EM algorithm. But suppose the sampling period was so large as to miss key dynamics in a physical system, then we would observe a noticeable deterioration in our identification accuracy.

Consider the system described in continuous time by

Table 3.12: Continuous system simulation matrices (A,C).

A					C				
-4.22	1.27	2.63	-1.63	-3.17	0.20	3.00	1.00	0.00	0.00
0.00	4.34	0.00	1.00	-14.90	0.00	0.90	0.00	0.84	1.00
2.13	8.50	-4.78	-1.06	-20.56	0.56	1.10	0.00	0.00	0.61
1.44	-1.92	0.60	-5.68	4.80	0.00	0.00	2.34	0.08	0.00
0.00	3.48	0.00	0.00	-8.49	1.58	3.20	0.09	0.00	0.96

During simulation, this system was sampled simultaneously at 0.1 s, 0.25 s and 0.5 s. There is a noticeable difference in the identification accuracy of the EM results when each set of data is input to the algorithm. The identified A matrices are tabulated below; the C matrices were omitted because they are unaffected by sampling.

Table 3.13: Comparison of EM estimation performance with varying sampling period.

Simulation Sampled at T = 0.1 s									
Simulation A (in continuous-time)					Simulation A (in discrete-time)				
-4.2229	1.2686	2.6286	-1.6286	-3.1714	0.6659	0.1610	0.1655	-0.1019	-0.4314
0.0	4.3400	0.0	1.0000	-14.9000	0.0060	1.2749	0.0029	0.0893	-1.1714
2.1257	8.5029	-4.7771	-1.0629	-20.5571	0.1327	0.5493	0.6360	-0.0476	-1.4645
1.4400	-1.9200	0.6000	-5.6800	4.8000	0.0919	-0.0872	0.0473	0.5524	0.2470
0.0	3.4800	0.0	0.0	-8.4900	0.0006	0.2771	0.0003	0.0124	0.2593
EM A (in continuous-time)					EM A (in discrete-time)				
-4.1183	1.1563	2.6358	-1.6514	-3.0070	0.6727	0.1542	0.1663	-0.1052	-0.4219
0.0325	4.3840	-0.0446	1.0177	-15.0063	0.0071	1.2773	0.0003	0.0908	-1.1773
2.1473	8.4625	-4.7992	-1.0825	-20.5354	0.1332	0.5467	0.6347	-0.0500	-1.4627
1.4081	-2.0906	0.6568	-5.5320	5.1127	0.0917	-0.0990	0.0507	0.5601	0.2698
0.0323	3.4777	-0.0203	0.0144	-8.5019	0.0025	0.2768	-0.0009	0.0132	0.2590
Simulation Sampled at T = 0.25 s									
Simulation A (in continuous-time)					Simulation A (in discrete-time)				
-4.2229	1.2686	2.6286	-1.6286	-3.1714	0.3822	0.3261	0.2116	-0.1183	-1.0041
0.0	4.3400	0.0	1.0000	-14.9000	0.0273	1.2397	0.0152	0.1643	-1.9188
2.1257	8.5029	-4.7771	-1.0629	-20.5571	0.1732	0.6328	0.3542	-0.0312	-2.0728
1.4400	-1.9200	0.6000	-5.6800	4.8000	0.1183	-0.0145	0.0764	0.2042	0.1155
0.0	3.4800	0.0	0.0	-8.4900	0.0056	0.4574	0.0030	0.0450	-0.4309
EM A (in continuous-time)					EM A (in discrete-time)				
-4.3401	1.2966	2.6408	-1.6137	-3.1699	0.3724	0.3308	0.2107	-0.1141	-1.0084
-0.0077	4.3656	0.0019	0.9699	-14.932	0.0294	1.2430	0.0162	0.1632	-1.9276
1.9679	8.3861	-4.7358	-0.9994	-20.3828	0.1657	0.6237	0.3556	-0.0254	-2.0571
1.3260	-2.1617	0.6128	-5.6036	5.2296	0.1071	-0.0393	0.0746	0.2085	0.1746
-0.0284	3.4758	-0.0010	-0.0139	-8.4671	0.0043	0.4580	0.0025	0.0443	-0.4308
Simulation Sampled at T = 0.5 s									
Simulation A (in continuous-time)					Simulation A (in discrete-time)				
-4.2229	1.2686	2.6286	-1.6286	-3.1714	0.1721	0.2053	0.1488	-0.0676	-1.0291
0.0	4.3400	0.0	1.0000	-14.9000	0.0556	0.6755	0.0368	0.1471	-1.5918
2.1257	8.5029	-4.7771	-1.0629	-20.5571	0.1295	0.1176	0.1632	-0.0273	-1.2328
1.4400	-1.9200	0.6000	-5.6800	4.8000	0.0828	0.1188	0.0678	0.0281	-0.2755
0.0	3.4800	0.0	0.0	-8.4900	0.0181	0.3730	0.0113	0.0642	-0.6985
EM A (in continuous-time)					EM A (in discrete-time)				
-3.7713	0.9138	2.4371	-2.1664	-2.3470	0.1897	0.2137	0.1401	-0.0783	-1.0364
0.1147	4.5024	-0.1413	1.0438	-15.3195	0.0500	0.6755	0.0367	0.1467	-1.5939
2.1341	8.2815	-4.6592	-1.0100	-20.3661	0.1323	0.1139	0.1675	-0.0243	-1.2411
1.2599	-2.9682	1.0911	-6.3727	7.0452	0.0817	0.1172	0.0709	0.0042	-0.2653
0.0736	3.4415	-0.0412	-0.0414	-8.4573	0.0182	0.3722	0.0098	0.0632	-0.6982

Our data confirms our analysis for as the time step gets bigger, the EM estimation of the continuous system deteriorates more and more. This is primarily due to two reasons: one, the sampling period is overstepping more information about the original system as it increases, and two, the slight differences between the EM estimated parameters and the discretized parameters in the discrete domain are magnified during the conversion from discrete to continuous time. Thus, the sampling period choice can significantly impact the

performance of the EM algorithm.

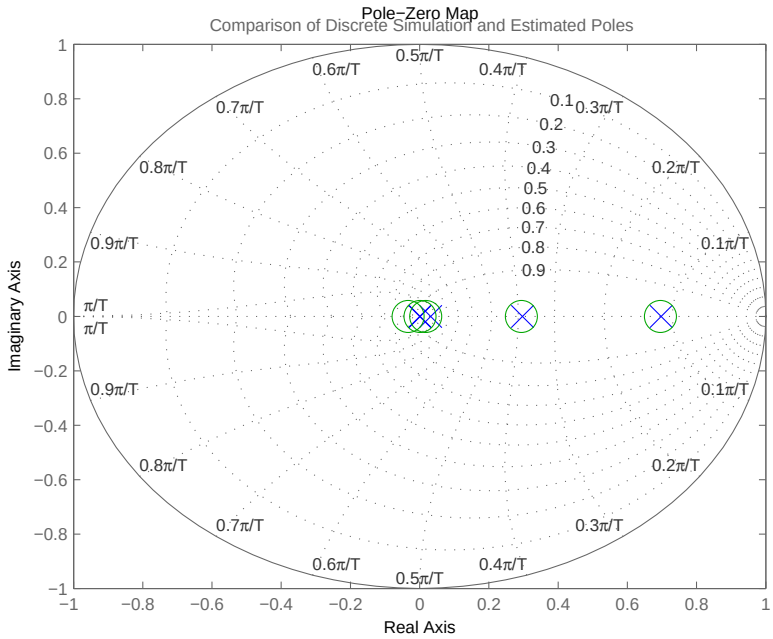
Another consideration concerning sampling period that does not directly impact the performance of the EM algorithm but does affect our overall identification process is the conversion from discrete to continuous time. In the previous section, we stated that the EM identified system must be converted back into the continuous time domain in order to make the original and identified system comparable. But consider a continuous system with a set of real poles widely dispersed in the left-hand s-plane such that a few poles are very negative while others are close to zero. When this system is sampled and mapped into the discrete domain, these poles are mapped into the discrete z-plane via the mathematical transform

$$z = e^{sT} \tag{3.13}$$

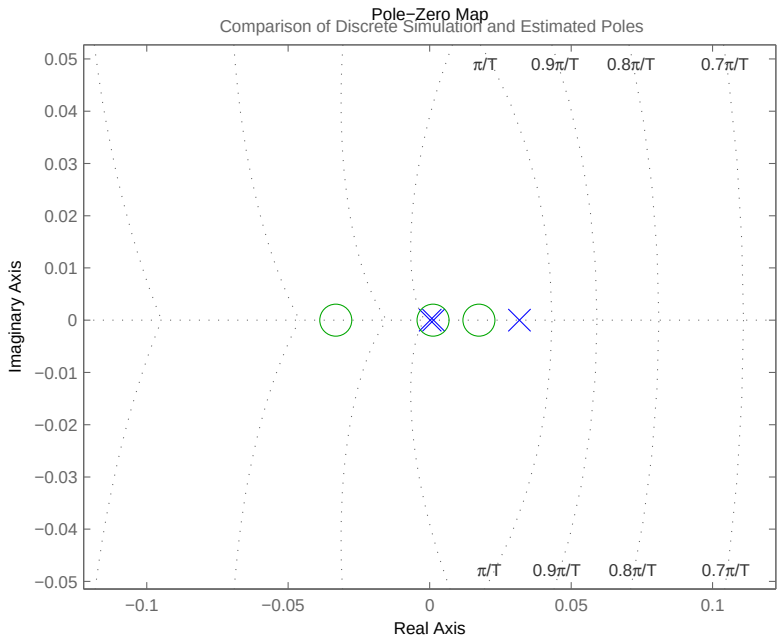
where z signifies the discrete domain pole, s the continuous domain pole, and T the sampling period. Because s is a negative real number, the mapping will place the z pole on the positive real axis in the z-plane. The more negative s is or the bigger the sampling period T is, the closer the discrete pole gets to the origin of the z-plane. As the placement of poles correspond closely to transient dynamics in the time domain, the EM algorithm's estimated state matrix should match the simulation poles in the z domain. However, given the degree of estimation accuracy observed in section 3.2, the EM algorithm could easily approximate a small, real, positive simulation pole with a small negative real pole. While this pole is very reasonable in the discrete domain, its presence makes the conversion of the estimated discrete system into a continuous system of the same order impossible because negative real poles in the z-domain correspond to complex poles in the s-domain. As all complex poles come in conjugate pairs, each estimated, negative, real pole corresponds to two complex poles in the continuous domain and increases the order of the estimated continuous system by one. Observe the movement of poles in the table and figures below.

Table 3.14: Comparison of simulation and estimated poles.

Discrete Poles		Continuous Poles	
Simulated System	Estimated System	Simulated System	Estimated System
0.6977	0.6966	-0.2400	-0.2410
0.2967	0.2982	-0.8100	-0.8067
0.0317	0.0284	-2.3000	-2.3754
0.0021	0.0022	-4.1000	-4.0944
0.0002	-0.0184	-5.6800	-2.6650 ± 2.0944i

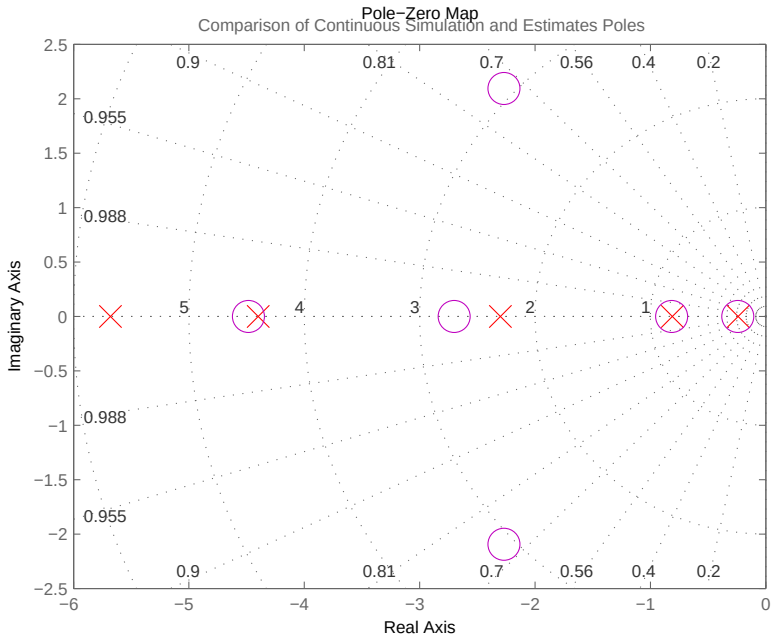


(a) Discrete simulation (blue crosses) and estimated (green circles) poles.



(b) Zoomed-in plot of (a)

Figure 3.6: Comparison of simulation and estimated poles in continuous and discrete time.



(c) Continuous simulation (red crosses) and estimated (magenta circles) poles.

Figure 3.6: Comparison of simulation and estimated poles in continuous and discrete time.

This data was collected using a sampling period of 1.5 s. The above table tabulates each individual discrete system pole with its corresponding discrete estimate and continuous counterparts in the same row. From the table, we can see that the less negative continuous poles are well matched in both domains but estimation accuracy wavers as the continuous poles become more negative and their discrete counterparts decrease exponentially toward zero. The most negative continuous simulation pole causes the negative discrete estimated pole and thus also the corresponding complex conjugate poles in the estimated system. The figures present this data graphically. From the first two figures, we observe that the two larger discrete poles are well matched while the remaining three about the origin encounter more fitting error. Nevertheless, when converted to the continuous domain, four of the poles still resemble their simulation counterparts but the complex pair appear random. Thus, sampling can also affect the reconstruction of the continuous time domain system.

In this linear case, the discrete to continuous pole conversion problem results from numerical, estimation error and we can remedy it readily with the knowledge that the estimated discrete system should never have negative real poles. A sensible fix for this error would be to shift the negative pole along the real axis

into the positive right-half plane with a slight positive perturbation above zero. This can be accomplished via an eigenvalue decomposition in which the state matrix, A, is decomposed into its eigenvalue (λ) and eigenvector (v) components. The negative eigenvalues can be reassigned to the positive real-axis and a new A reconstructed from the modified set of eigenvalues and the original eigenvectors. We apply this method to the EM estimated state matrix and get the set of continuous and discrete eigenvalues below. The change only affects the complex conjugate poles and the resulting continuous system is of the same order as the original discrete system.

Table 3.15: Modified estimated poles of discrete and continuous system.

Discrete System	Continuous System
0.6966	-0.2410
0.2982	-0.8067
0.0284	-2.3754
0.0022	-4.0944
0.0000	-9.2107

Another sampling concern is the effect of non-uniform sample times on model identification. Consider a linear system sampled in two ways, uniformly and non-uniformly.

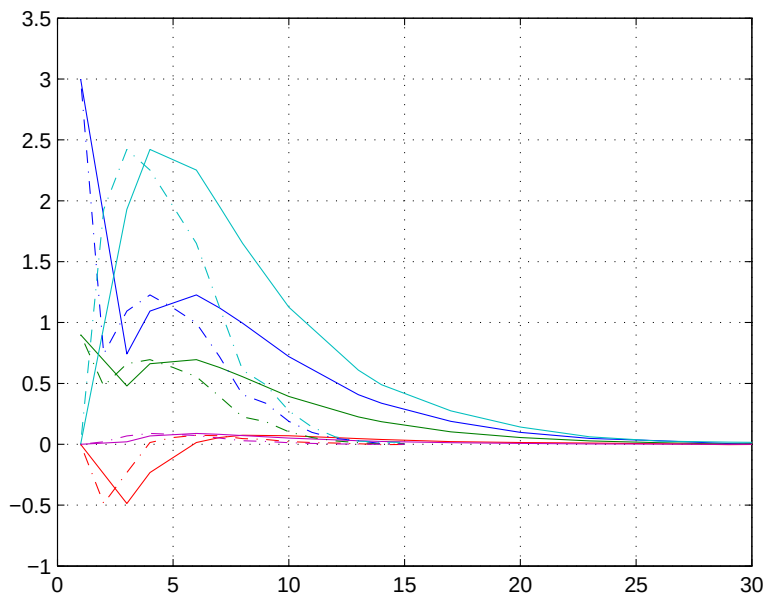


Figure 3.7: Comparison of uniformly (solid line) and non-uniformly (dash line) sampled data.

Plotted on the same interval, at uniform step sizes, the two data sets can appear completely different. De-

pending on how sporadic the non-uniform sampling is, a relatively linear system can appear vastly nonlinear due to the abnormal compression it undergoes during uniform processing. Observe in the above plot the changing sample steps translate into fluctuating degrees of time-delay in the non-uniformly sampled data set. Its system dynamics appear completely different from that of the uniformly sampled data set. Recall that discrete state-space is a set of fixed-step, difference equations; further, the EM algorithm assumes all data points are equally spaced. It follows then, that if non-uniformly sampled data was input to the algorithm, it would be treated as uniformly sampled data even though the two are not at all comparable in the same time-frame. Then, logically, the EM algorithm would not reconstruct the correct state space model that the data was taken from.

Given knowledge of the time step between every sample, the EM algorithm can account for sampling inconsistencies by assuming a standard step size and predicting all intermittent missing data points through the Kalman filter. Specifically, whenever a data point is unobserved at an anticipated sample time, the forward, predictive Kalman equations for state and covariance estimation, (2.58a) and (2.58b), are substituted in place of the forward, time update equations, (2.58d) and (2.58e) to derive a guess for the missing point. A sample test case to generate non-uniformly sampled data sets was created using the matrices in Table 3.16.

Table 3.16: Continuous system simulation matrices (A,C) for non-uniform sampling.

A					C				
0.23	-0.40	-0.07	0.22	0.82	0.0	3.0	1.0	0.0	0.0
-0.33	0.08	-0.12	0.32	1.04	0.0	0.9	0.0	0.84	1.0
-1.07	0.31	0.56	0.39	0.19	1.34	0.0	0.0	0.0	0.61
-0.74	0.58	0.13	0.62	-0.41	0.0	0.0	2.34	0.078	0.0
-0.14	-0.29	-0.08	0.18	1.17	0.2	0.0	0.09	0.0	0.0

Five sets of data were generated using initial conditions of the standard basis. The data sets were first generated for thirty consecutive time points; then, a subset of that data was taken at the time points $\{1, 3, 4, 6, 7, 8, 10, 13, 14, 17, 20, 23, 26, 30\}$ to get the effect of non-uniform sampling. The data was input to a regular EM algorithm that treated the data sets as uniformly sampled data and also a modified EM algorithm that estimated the missing data between known data points. The results are tabulated below in Table 3.17.

The difference in performance is conspicuously obvious. The modified EM algorithm tracks the dynamics

well despite the missing time points and produces a dynamics matrix that is accurate to at least the second decimal place. On the other hand, the unmodified EM algorithm identifies a completely different model that is not at all comparable to the simulation system. Thus, consistency of data sampling can make quite a difference on the performance of the EM algorithm. But if inconsistensies are unavoidable, then the EM can still be adjusted to accommodate the data non-uniformity.

Table 3.17: Comparison of EM performance given non-uniformly sampled data.

A (EM)					A (modified EM)				
0.0454	-0.2506	-0.0297	0.1367	0.5995	0.2441	-0.3861	-0.0597	0.1947	0.8032
-0.3265	-0.0089	-0.0852	0.2804	0.8971	-0.3191	0.1032	-0.1101	0.3050	1.0042
-1.2430	0.8315	0.4336	0.3398	-0.3161	-1.0585	0.3306	0.5678	0.3666	0.1607
-0.8758	1.0198	0.2385	0.3327	-0.8949	-0.7202	0.5604	0.1360	0.6078	-0.4002
-0.1546	-0.3084	-0.1058	0.2135	1.0720	-0.1394	-0.2793	-0.0801	0.1785	1.1591

Chapter 4

The Model and State Transformations

In our previous chapter, we briefly broached the subject of state transformations to enable coherent comparisons between the simulation systems and the identified systems. This section reopens the topic to fully elucidate the non-trivial problem transformations pose during actual experimentation and to propose some techniques for resolving this problem. To make this topic germane to our thesis, we will first detail our final model structure and then explain the effect of transformation upon it. We will close with a few proposed solutions to the transformation problem.

4.1 A Model with Biological Modifications

In 2004, Rangel et. al. [21] published an article describing a linear state-space system that modeled T-cell activation. The model extended simple linear regression approaches by incorporating a state space model so that the so called "hidden state" could potentially capture dynamics that were unobserved or unmeasurable in gene expression experiments. The structure she described was

$$\begin{aligned} x_{k+1} &= Ax_k + Bg_k + w_k \\ g_k &= Cx_k + Dg_{k-1} + v_k \end{aligned} \tag{4.1}$$

where x_k represented the set of unobserved dynamic states and g_k was a vector of gene expression data. Furthermore, w_k and v_k represented state and measurement noise respectively and were both assumed to

be orthogonal white noise processes. Having set this initial system structure, Rangel used the Expectation-Maximization algorithm to estimate the state coefficient matrices that actually described the system dynamics.

Rangel's model is a closed loop system in the sense that outputs are fed back to the input. However, during the identification process, the actual measurements, g_k, are used at each step so the model is effectively running open loop; as such, there is no guarantee that the identified system will always be stable. Furthermore, the reuse of observed states to calculate future hidden states incorporated observation noise into the hidden state dynamics, which seems physically implausible. Thus, we modify the model to close the loop by appending the gene expression states to the hidden states and instating a new output y_k. The augmentation of the hidden states changed our model structure to

$$\begin{aligned} \left[\begin{array}{c} x_{k+1} \\ \hline g_{k+1} \end{array}\right] &= \left[\begin{array}{c|c} A & B \\ \hline CA & CB+D \end{array}\right] \left[\begin{array}{c} x_k \\ \hline g_k \end{array}\right] + w_k \\ y_k &= \left[\begin{array}{c|c} 0 & I \end{array}\right] \left[\begin{array}{c} x_k \\ \hline g_k \end{array}\right] + v_k \end{aligned} \tag{4.2}$$

Because the augmented system incorporates g_k into the hidden states, the execution of the Kalman filter during the EM algorithm will operate on an augmented A matrix as well and should yield a stable identified system. Also, because g_k now operates as a hidden state, the measurement noise no longer affects the internal state dynamics. Lastly, because we know the observed states are simply the original gene expression states plus some measurement noise, we can completely designate the structure and content of the new observation matrix, C.

Although the system appears simpler now because of the absence of the inputs, the number of system coefficients has actually increased. This is because the augmented A matrix is the same size as the previous A, B, C, and D matrices combined; however, the new covariance matrix of w_k, Q, has increased in size to account for the covariances of the added hidden states. Likewise, the initial state estimate and covariance matrix, π_1 and V_1, must also account for the added hidden states. The measurement noise covariance R remains unchanged and the output matrix is completely identified. Despite the size increase, we still need

to augment this system further with one more state. Depending on the test case and type of data used, the observed data may not always have steady states about zero; in such cases, unaccounted for biases can throw off the Kalman filter estimation of the states. To counter this effect, we force a unity state into the hidden state vector. Thus, any biases in the data will be identified as part of the A matrix; if no such bias exists, these coefficients will be zero. The new system is shown below.

$$\begin{aligned} \left[\begin{array}{c} 1 \\ \hline x_{k+1} \\ \hline g_{t+1} \end{array}\right] &= \left[\begin{array}{c|c|c} 1 & 0 & 0 \\ \hline c_{bias} & A & B \\ \hline c_{bias} & CA & CB+D \end{array}\right] \left[\begin{array}{c} 1 \\ \hline x_k \\ \hline g_k \end{array}\right] + w_k \\ y_k &= \left[\begin{array}{c|c|c} 0 & 0 & I \end{array}\right] \left[\begin{array}{c} 1 \\ \hline x_k \\ \hline g_k \end{array}\right] + v_k \end{aligned} \tag{4.3}$$

It is convenient to additionally define a system here that is capable of accounting for an external input that affects only the state vector but not the output vector. In experimentation, the input could represent known, artificial stimulus injected by the experimentalist. Such a system would be similar to that described in Eq. (3.9) except, in this instance, the A matrix is augmented. It would have the implicit form described below. The portion of the augmented B matrix that corresponds to the artificial bias state is set to zero during each EM iteration.

$$\begin{aligned} \left[\begin{array}{c} 1 \\ \hline x_{k+1} \\ \hline g_{t+1} \end{array}\right] &= \left[\begin{array}{c|c|c} 1 & 0 & 0 \\ \hline c_{bias} & A & B \\ \hline c_{bias} & CA & CB+D \end{array}\right] \left[\begin{array}{c} 1 \\ \hline x_k \\ \hline g_k \end{array}\right] + \left[\begin{array}{c} 0 \\ B_x \\ B_g \end{array}\right] u_k + w_k \\ y_k &= \left[\begin{array}{c|c|c} 0 & 0 & I \end{array}\right] \left[\begin{array}{c} 1 \\ \hline x_k \\ \hline g_k \end{array}\right] + v_k \end{aligned} \tag{4.4}$$

The augmentation of (4.1) to (4.2) changes the EM algorithm variant used to identify the system. The state-space model described by Rangel mirrors that described in (2.10). The condensation of the g_k states into the hidden state vector,x_k, reverts the system to the original model proposed by Shumway et. al. [24], (3.3). Finally, the addition of the artificial input vector converts the system into a specialized case of (3.9). The EM algorithm used to identify each system model changes accordingly; both are described in the pre-

vious chapter. The addition of the bias state in (4.3) however introduces a new EM algorithm variant that has not been broached thus far. This system deviates from previous systems because it statically allocates specific sections of all the system parameters.

In addition to augmenting the system parameters A, B, and C with an identity row vector, an all-zero row vector, and an all-zero column vector, respectively, the bias state in the initial state vector, π_1, must be statically assigned to unity at each iteration of the EM algorithm to ensure that the bias state stays at one. Further, because the bias state is an artificial state, it has no real, physical effect on the other states nor does it vary with respect to itself, hence its cross-covariances in Q and V_1 are set to zero and its variances slightly perturbed from zero but still significantly lower than the other states' variances. These variances could not be directly set to zero as well because Q and V_1 must be nonsingular so that their inverses exist and can be calculated for the modified log likelihood of the EM algorithm.

Table 4.1: EM algorithm parameters with bias state added.

$$A_{aug} = \left[\begin{array}{c|c|c} 1 & 0 & 0 \\ \hline c_{bias} & A & B \\ \hline c_{bias} & CA & CB+D \end{array}\right] \quad B_{aug} = \left[\begin{array}{c} 0 \\ B_x \\ B_g \end{array}\right] \quad C_{aug} = \left[\begin{array}{c|c|c} 0 & 0 & I \end{array}\right]$$

$$Q_{aug} = \left[\begin{array}{c|c|c} 10^{-12} & 0 & 0 \\ \hline 0 & Q_{xx^T} & Q_{xg^T} \\ \hline 0 & Q_{gx^T} & Q_{gg^T} \end{array}\right] \quad R_{aug} = \left[\begin{array}{c|c} 10^{-12} & 0 \\ \hline 0 & R_{yy^T} \end{array}\right]$$

$$\pi_{1,aug} = \left[\begin{array}{c} 1 \\ \hline \hat{x}_1 \\ \hline \hat{g}_1 \end{array}\right] \quad V_{1,aug} = \left[\begin{array}{c|c|c} 10^{-12} & 0 & 0 \\ \hline 0 & V_{1,xx^T} & V_{1,xg^T} \\ \hline 0 & V_{1,gx^T} & V_{1,gg^T} \end{array}\right]$$

The static allocation of these parameters is done at the end of each M-step iteration after new parameters have been calculated using the previously derived parameter updates. Thus, the parameters are still maximized with respect to the current Kalman state and variance estimates via the update equations before parts of them are reassigned.

This static assignment is reasonable in the case of the bias state because we know the "true" bias-associated values in the parameter matrices, consequently setting them thus can only increase the log likelihood. However, because these values are set, they can not be optimized simultaneously with the other parameter components. Specifically, they have no derivative, hence their components in the simultaneous equations'

matrices are superfluous and should not be allowed to influence the outcome of the significant, unknown updates. Suppose we retrace the derivation of the M-step update equations, beginning with the A and B derivatives, and take the location of the known variables into account. Observe the following derivation step leading up to (2.39),

$$\begin{bmatrix} \sum_{j=1}^{N}\sum_{k=2}^{T} P_{j(k,k-1)} & \sum_{j=1}^{N}\sum_{k=2}^{T} \hat{x}_{jk} u_{j(k-1)}^{T} \end{bmatrix} = \left[\begin{array}{cc} A_{known} & B_{known} \\ \hline A_{unknown} & B_{unknown} \end{array}\right] \begin{bmatrix} \sum_{j=1}^{N}\sum_{k=2}^{T} P_{j(k-1)} & \sum_{j=1}^{N}\sum_{k=2}^{T} \hat{x}_{j(k-1)} u_{j(k-1)}^{T} \\ \sum_{j=1}^{N}\sum_{k=2}^{T} u_{j(k-1)} \hat{x}_{j(k-1)}^{T} & \sum_{j=1}^{N}\sum_{k=2}^{T} u_{j(k-1)} u_{j(k-1)}^{T} \end{bmatrix}$$

Both A_{known} and B_{known} are located in the top row of their respective matrices. As a result, they only affect the top row of the product and their own updates. The remaining unknown components' solutions do not contain any portion of A_{known} or B_{known}. In other words, the superfluous components of the A and B derivatives are partitioned in such a way that they affect only themselves. Hence, (2.39) can remain unchanged as long as the the static values are reset after each update. A similar effect can be observed in the Q derivative; observe the Q update containing the partitioned A and B matrices

$$\begin{aligned} Q_{new} = \frac{1}{(N)(T-1)} \sum_{j=1}^{N}\sum_{k=2}^{T} \Bigg[\Bigg(& (\hat{x}_{jk} - \begin{bmatrix} A_{known} \\ \hline A_{unknown} \end{bmatrix} \hat{x}_{j(k-1)} - \begin{bmatrix} B_{known} \\ \hline B_{unknown} \end{bmatrix} u_{j(k-1)}) \times \\ & (\hat{x}_{jk} - \begin{bmatrix} A_{known} \\ \hline A_{unknown} \end{bmatrix} \hat{x}_{j(k-1)} - \begin{bmatrix} B_{known} \\ \hline B_{unknown} \end{bmatrix} u_{j(k-1)})^{T} + V_{jk} - \begin{bmatrix} A_{known} \\ \hline A_{unknown} \end{bmatrix} V_{j(k-1,k)} \\ & - V_{j(k,k-1)} \begin{bmatrix} A_{known} \\ \hline A_{unknown} \end{bmatrix}^{T} + \begin{bmatrix} A_{known} \\ \hline A_{unknown} \end{bmatrix} V_{j(k-1)} \begin{bmatrix} A_{known} \\ \hline A_{unknown} \end{bmatrix}^{T} \Bigg) \Bigg] \end{aligned}$$

The known components of Q are in the first row and first column of the Q parameter. These components correspond to the only portions of the update affected by A_{known} and B_{known} so once again the unknown components are not influenced. The π_1 update is slightly different from the previous updates in that it depends on the Kalman estimate of the first state; each component is dependent on the mean of its own estimates so there is no fear of the bias state affecting the other initial states. Furthermore, due to the form

of A and B, the Kalman filter propagates the preset bias state at one through all its iterations once it has been set at the first iteration so it does not have to be reset after each update. Lastly, observe the update of V_1, with the partition on π_1

$$V_{1new} = \frac{1}{N}\sum_{j=1}^{N}\left[V_{j1} + (\hat{x}_{j1} - \left[\frac{\pi_{1,known}}{\pi_{1,unknown}}\right])(\hat{x}_{j1} - \left[\frac{\pi_{1,known}}{\pi_{1,unknown}}\right])^T\right]$$

Parameter V_1 is preset in the first row and column; again, these partitions correspond to the only parts of the parameter that are affected by the known state of π_1. Thus, the static allocation is validated and can be used without detracting the EM algorithm from its convergence properties.

In terms of system interpretation and gene pathway identification, Rangel et. al. [21] analyze the direct feedthrough matrix, $CB + D$, of their identified system to infer gene regulatory pathways. They chose this feedthrough matrix because it is not subject to the state transformation described in section 3.1, and it maps the influence of the gene states on themselves. These influences have a strong correspondence to actual, physiological connections in the simulated system and are readily interpretable. Further, this matrix will not vary with the number of hidden states because it maps the relation between the observed system states only. All other system matrices will vary with the number of hidden states used and may not always correspond to actual physiological connections in the simulated system. Thus, this matrix remains comparable across systems with varying numbers of hidden states. In our augmented system, the direct feedthrough matrix appears in the bottom right partition of our identified state matrix. It is easily accessible; for this reason and those described above, we will also use the feedthrough matrix as a method of comparison and a means of inferring gene pathways.

4.2 A Return to State Transformations

In Chapter 3, we observed the effects of state-transformations in our simulation results whenever all non-trivial, state-space parameters were unknown. These transformations occur because our only knowledge of the system derives from the input-output behavior we observe and there are an infinite number of state representations similar to the original simulated state-space that can produce the same dynamics. Thus, without specific knowledge of some system parameter, such as a full-rank column-wise C matrix or an initial

state π_1, there is no sure way to identify the exact state-space representation that generated the observations.

In our previous linear test cases, we remedied this identification problem utilizing the knowledge that the set of simulated initial states formed an identity matrix when concatenated in descending order and as a result, the inverse transform appeared exactly in the estimated initial state vectors when they were concatenated in the same order. Once the transform was identified, we restored the original state-space via an inverse transformation on the estimated parameters. While this is a handy corrective in simulation, this method is not plausible in experimentation. We assume in our remedy that we know the initial state when that is not true. Specifically, the unknown, or hidden, states in our true system can not be observed.

In our model formulation, we work with a fully known observation matrix, C, that is not full rank. Consider,

$$C^{m\times n} = \left[\begin{array}{c|c} 0^{m\times p} & I^{m\times(n-p)} \end{array}\right] \tag{4.5}$$

where I is the identity matrix and $p < n$. Then transform matrices of the form

$$F^{n\times n} = \left[\begin{array}{c|c} F_1^{p\times p} & F_2^{p\times(n-p)} \\ \hline 0^{(n-p)\times p} & I^{(n-p)\times(n-p)} \end{array}\right] \tag{4.6}$$

can still transform the state representation without affecting C.

$$\begin{aligned} C_F^{m\times n} &= C^{m\times n}F^{n\times n} \\ \left[\begin{array}{c|c} 0^{m\times p} & I^{m\times(n-p)} \end{array}\right] &= \left[\begin{array}{c|c} 0^{m\times p} & I^{m\times(n-p)} \end{array}\right] \left[\begin{array}{c|c} F_1^{p\times p} & F_2^{p\times(n-p)} \\ \hline 0^{(n-p)\times p} & I^{(n-p)\times(n-p)} \end{array}\right] \end{aligned} \tag{4.7}$$

Thus, while C remains unchanged, the other system parameters such as A,Q, π_1 etc. are still affected, making our model subject to state-transformations. Let x be the set of true hidden states, including our

bias state, and g the set of true gene states such that our system is generalized to

$$\begin{aligned} \begin{bmatrix} x \\ g \end{bmatrix}_{k+1} &= \begin{bmatrix} A & B \\ CA & CB+D \end{bmatrix} \begin{bmatrix} x \\ g \end{bmatrix}_k \\ y_{k+1} &= \begin{bmatrix} 0 & I \end{bmatrix} \begin{bmatrix} x \\ g \end{bmatrix}_{k+1} \end{aligned} \tag{4.8}$$

Then, given the constraints on C, our system is subject to any transformation of the form

$$\begin{bmatrix} z \\ g \end{bmatrix}_k = \begin{bmatrix} F_1 & F_2 \\ 0 & I \end{bmatrix} \begin{bmatrix} x \\ g \end{bmatrix}_k \tag{4.9}$$

where z is the set of transformed hidden state. The transformed state space system in terms of the transformed state vector is

$$\begin{aligned} \begin{bmatrix} z \\ g \end{bmatrix}_{k+1} &= \begin{bmatrix} F_1 & F_2 \\ 0 & I \end{bmatrix} \begin{bmatrix} A & B \\ CA & CB+D \end{bmatrix} \begin{bmatrix} F_1 & F_2 \\ 0 & I \end{bmatrix}^{-1} \begin{bmatrix} z \\ g \end{bmatrix}_k \\ y_{k+1} &= \begin{bmatrix} 0 & I \end{bmatrix} \begin{bmatrix} F_1 & F_2 \\ 0 & I \end{bmatrix}^{-1} \begin{bmatrix} z \\ g \end{bmatrix}_{k+1} \end{aligned} \tag{4.10}$$

which reduces to

$$\begin{aligned} \begin{bmatrix} z \\ g \end{bmatrix}_{k+1} &= \begin{bmatrix} F_1AF_1^{-1} + F_2CAF_1^{-1} & -F_1AF_1^{-1}F_2 - F_2CAF_1^{-1}F_2 + F_1B + F_2CB + F_2D \\ CAF_1^{-1} & -CAF_1^{-1}F_2 + CB + D \end{bmatrix} \begin{bmatrix} z \\ g \end{bmatrix}_k \\ y_{k+1} &= \begin{bmatrix} 0 & I \end{bmatrix} \begin{bmatrix} z \\ g \end{bmatrix}_{k+1} \end{aligned} \tag{4.11}$$

This transformed system has an identical structure to (4.8) except there are an infinite number of F_1's and F_2's that satisfy (4.9). Consequently, there are also an infinite number of transformed state space systems that have the same input-output behavior as (4.8). Unfortunately, the structure of this transformation influences the direct feedthrough portion of our state matrix. The transformed $CB + D$ matrix now contains

an additional term involving both F_1 and F_2. As F_1 and F_2 can be arbitrary, the transformed feedthrough matrix can bear little or no resemblance to the original feedthrough system. There appears to be no way to decouple the transform from the original system with just input and output observations hence this matrix is not identifiable.

This transformation is not unique to our model structure. Consider the unaugmented state space system described in [21]

$$\begin{aligned} x_{k+1} &= Ax_k + Bg_k \\ g_{k+1} &= Cx_{k+1} + Dg_k \end{aligned} \tag{4.12}$$

Given a state transformation that satisfies

$$z_k = F_1 x_k + F_2 g_k \tag{4.13}$$

Substitute (4.13) into (4.12) and consider the state space system in terms of z. The state equation in (4.12) becomes

$$\begin{aligned} F_1^{-1} z_{k+1} - F_1^{-1} F_2 g_{k+1} &= A\left(F_1^{-1} z_k - F_1^{-1} F_2 g_k\right) + Bg_k \\ z_{k+1} &= F_1 A\left(F_1^{-1} z_k - F_1^{-1} F_2 g_k\right) + F_1 B g_k + F_2\left(C\left(Ax_k + Bg_k\right) + Dg_k\right) \\ z_{k+1} &= F_1 A\left(F_1^{-1} z_k - F_1^{-1} F_2 g_k\right) + F_1 B g_k + F_2\left(C\left(A\left(F_1^{-1} z_k - F_1^{-1} F_2 g_k\right) + Bg_k\right) + Dg_k\right) \\ z_{k+1} &= \left(F_1 + F_2 C\right) A F_1^{-1} z_k - F_1 A F_1^{-1} F_2 g_k + F_1 B g_k - F_2 C A F_1^{-1} F_2 g_k + F_2 C B g_k + F_2 D g_k \\ z_{k+1} &= \left[\left(F_1 + F_2 C\right) A F_1^{-1}\right] z_k + \left[F_1 B - F_1 A F_1^{-1} F_2 - F_2 C A F_1^{-1} F_2 + F_2 C B + F_2 D\right] g_k \end{aligned}$$

The observation equation in (4.12) becomes

$$\begin{aligned} g_{k+1} &= C\left(F_1^{-1} z_{k+1} - F_1^{-1} F_2 g_{k+1}\right) + Dg_k \\ g_{k+1} + C F_1^{-1} F_2 g_{k+1} &= C F_1^{-1} z_{k+1} + Dg_k \\ g_{k+1} &= \left[\left(I + C F_1^{-1} F_2\right)^{-1} C F_1^{-1}\right] z_{k+1} + \left[\left(I + C F_1^{-1} F_2\right)^{-1} D\right] g_k \end{aligned}$$

The transformed state space in entirety is

$$
\begin{aligned}
z_{k+1} &= \left[\left(F_1 + F_2 C\right) A F_1^{-1}\right] z_k + \left[F_1 B - F_1 A F_1^{-1} F_2 - F_2 C A F_1^{-1} F_2 + F_2 C B + F_2 D\right] g_k \\
g_{k+1} &= \left[\left(I + C F_1^{-1} F_2\right)^{-1} C F_1^{-1}\right] z_{k+1} + \left[\left(I + C F_1^{-1} F_2\right)^{-1} D\right] g_k
\end{aligned}
\tag{4.14}
$$

This transformed system can exactly match the input-output behavior of (4.12) even though its parameters are vastly different. This transformation is possible because the observation states, g, are being fed back as input. The resultant transformed coefficients for this unaugmented system are identical to the transformed coefficients of our augmented system. Hence, these two systems have the same identifiability problem. To confirm our hypothesis, we propose the following test case,

$$
\begin{aligned}
x_{k+1} &= \begin{bmatrix} 0.36 & -0.16 & 0.16 \\ -0.25 & 0.28 & 0.27 \\ -0.57 & -0.44 & 1.11 \end{bmatrix} x_k + \begin{bmatrix} 0.31 & 0.02 & -0.03 \\ -0.02 & 0.15 & -0.03 \\ 0.05 & -0.03 & 0.29 \end{bmatrix} g_k + w_k \\
y_{k+1} &= \begin{bmatrix} 0.20 & -3.00 & 1.00 \\ 1.40 & 0.90 & -0.84 \\ -1.58 & 0.70 & 0.09 \end{bmatrix} x_{k+1} + \begin{bmatrix} 0.24 & 0.15 & -0.01 \\ -0.12 & 0.04 & 0.30 \\ -0.10 & 0.13 & 0.23 \end{bmatrix} g_k + v_k
\end{aligned}
\tag{4.15}
$$

Three sets of 50-point data were generated from this system. The state noise vector was generated from a Gaussian distribution with mean zero and variance 0.0001^2. The observation noise vector was generated from a Gaussian distribution with zero mean and variance 0.001^2. The noise variances were reduced to better observe the accuracy of the identified systems. The data was input to the EM algorithm with six random initial parameter sets. The system identified by the EM algorithm proved to be very dependent on the initial parameter values chosen. Consider the three systems identified below, each with a different starting parameter, π_1.

Table 4.2: EM-identified parameters from different initial parameter values.

Actual			ID System 1			ID System 2			ID System 3		
A											
0.36	-0.16	0.16	-0.02	0.05	0.03	0.16	0.14	0.09	0.24	0.13	0.15
-0.25	0.28	0.27	-0.40	0.47	0.08	0.26	0.21	0.17	0.18	0.12	0.11
-0.57	-0.44	1.11	0.09	0.04	0.12	0.20	-0.02	0.22	0.21	-0.08	0.22
B											
0.31	0.02	-0.03	-0.53	-0.94	-0.12	0.07	-0.03	0.09	0.22	0.23	0.14
-0.02	0.15	-0.03	-0.65	-1.70	0.25	0.71	1.32	0.34	0.70	1.29	0.36
0.05	-0.03	0.29	-0.38	-0.40	-0.54	-0.16	-0.02	-0.50	-0.05	0.17	-0.45
C											
0.20	-3.00	1.00	0.03	-0.02	-0.09	0.10	0.03	-0.03	0.10	0.01	-0.02
1.40	0.90	-0.84	-0.07	-0.02	0.02	-0.03	0.07	0.03	-0.01	0.08	0.04
-1.58	0.70	0.09	0.05	-0.01	0.04	0.01	-0.04	0.04	0.00	-0.05	0.03
D											
0.24	0.15	-0.01	0.72	-0.30	0.34	0.72	-0.30	0.36	0.73	-0.30	0.36
-0.12	0.04	0.30	0.18	0.75	-0.18	0.17	0.73	-0.19	0.16	0.72	-0.20
-0.10	0.13	0.23	-0.62	0.19	0.64	-0.62	0.19	0.64	-0.62	0.20	0.64

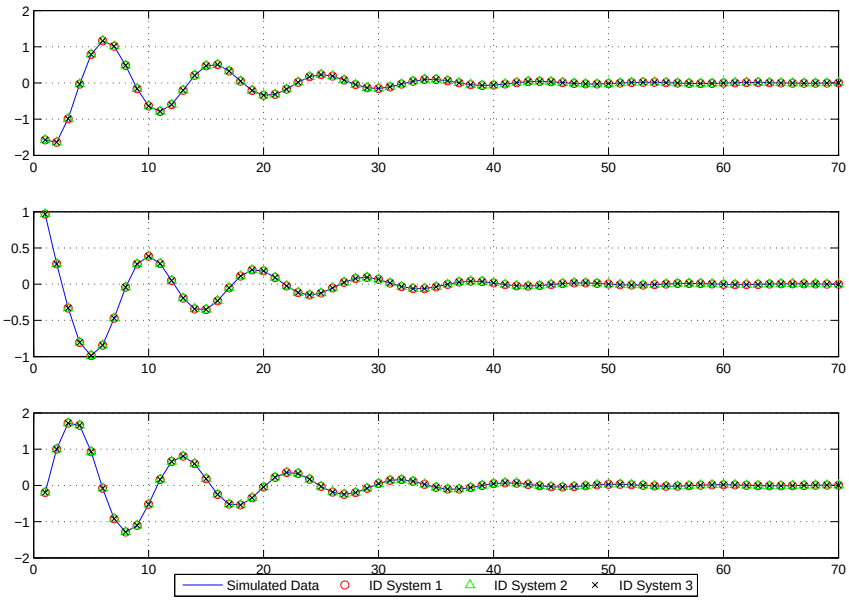


Figure 4.1: Comparison of EM identified systems identified with different initial parameter sets.

Except for the D matrix, the three identified systems are very dissimilar from each other as well as the original system. The D matrices are very similar among the three identified systems but are noticeably different from the original D matrix. Similarly, the corresponding feedthrough matrices are very dissimilar from the true $CB + D$ matrix because the transform acts on them as well.

Table 4.3: EM identified feedthrough matrix $CB + D$.

Actual			ID System 1			ID System 2			ID System 3		
CB+D											
0.41	-0.33	0.36	0.75	-0.26	0.39	0.76	-0.26	0.39	0.76	-0.26	0.39
0.25	0.23	-0.01	0.22	0.83	-0.19	0.21	0.82	-0.19	0.21	0.82	-0.19
-0.60	0.20	0.28	-0.65	0.14	0.61	-0.65	0.14	0.61	-0.65	0.14	0.61

The similarity in the D and $CB + D$ matrices result because only the initial π_1 parameter was varied during the EM identification process; all other parameters started from the same initial value. If other initial parameter variables were altered as well then the similarity in the D and $CB + D$ matrices disappear.

In this case, because we know the original system that generated our simulation data, we can back-solve for the transformation matrices, F_1 and F_2, of each identified system and undo the transform on each. There are several ways for back-solving for the transformations, we describe the method we used below.

$$\begin{aligned} C_{identified}A_{identified} &= C_{original}A_{original}F_1^{-1} \\ F_1 &= \left[C_{identified}A_{identified}\right]^{-1} C_{original}A_{original} \end{aligned} \tag{4.16}$$

$$\begin{aligned} A_{identified} &= F_1A_{original}F_1^{-1} + F_2C_{original}A_{original}F_1^{-1} \\ F_2 &= \left[A_{identified} - F_1A_{original}F_1^{-1}\right]\left[C_{original}A_{original}F_1^{-1}\right]^{-1} \end{aligned} \tag{4.17}$$

Using (4.16) and (4.17) we solved for the transformation matrices of each identified system and deduced the identified system in its original form. Our method of solving for F_1 and F_2 matches untransformed A and CA matrices to the simulated system's A and CA matrices exactly. As a result, any estimation error in the A or C parameter is transferred to the other estimated parameters by our transformation. The untransformed systems are tabulated below. As transformations don't affect input-output behavior, the simulation accuracy of the three systems remains unchanged from that shown in Figure 4.1.

Table 4.4: Re-transformed EM parameters.

Actual			ID System 1			ID System 2			ID System 3		
A											
0.36	-0.16	0.16	0.36	-0.16	0.16	0.36	-0.16	0.16	0.36	-0.16	0.16
-0.25	0.28	0.27	-0.25	0.28	0.27	-0.25	0.28	0.27	-0.25	0.28	0.27
-0.57	-0.44	1.11	-0.57	-0.44	1.11	-0.57	-0.44	1.11	-0.57	-0.44	1.11
B											
0.31	0.02	-0.03	0.30	-0.00	-0.04	0.30	-0.01	-0.04	0.30	-0.02	-0.05
-0.02	0.15	-0.03	-0.01	0.15	-0.01	-0.00	0.15	-0.02	-0.01	0.14	-0.02
0.05	-0.03	0.29	0.08	-0.01	0.29	0.09	-0.02	0.29	0.09	-0.03	0.28
C											
0.20	-3.00	1.00	0.20	-3.00	1.00	0.20	-3.00	1.00	0.20	-3.00	1.00
1.40	0.90	-0.84	1.40	0.90	-0.84	1.40	0.90	-0.84	1.40	0.90	-0.84
-1.58	0.70	0.09	-1.58	0.70	0.09	-1.58	0.70	0.09	-1.58	0.70	0.09
D											
0.24	0.15	-0.01	0.29	0.19	0.09	0.28	0.18	0.06	0.27	0.16	0.05
-0.12	0.04	0.30	-0.05	0.18	0.29	-0.05	0.19	0.31	-0.03	0.21	0.31
-0.10	0.13	0.23	-0.17	0.04	0.17	-0.16	0.03	0.17	-0.16	0.02	0.17

While not perfect, these matrices are a lot closer to the original simulation ones. Understandably, the corresponding feedthrough matrix is likewise transformed.

Table 4.5: The re-transformed feedthrough matrix $CB + D$.

Actual			ID System 1			ID System 2			ID System 3		
CB+D											
0.41	-0.33	0.36	0.45	-0.28	0.41	0.45	-0.28	0.40	0.45	-0.29	0.39
0.25	0.23	-0.01	0.30	0.32	-0.02	0.30	0.32	-0.01	0.31	0.34	-0.01
-0.60	0.20	0.28	-0.64	0.15	0.25	-0.63	0.15	0.25	-0.63	0.14	0.25

A suitable test that would noticeably differentiate between the transformed and untransformed systems would be a prediction case. Both systems match the training sets perfectly despite differing initial states and system parameters but a new random initial condition would logically cause different dynamics in the system that does not match the original one. A new set was generated from a random initial state. The same state was input to an identified system and an untransformed one. The results are plotted below in Figure 4.2 and confirm our hypothesis.

The EM identified system doesn't match the simulation at all while the untransformed system remains accurate. Only one of the identified systems were used to illustrate this point but all three systems produce the same behavior. The same results can be produced using the augmented system. Given the same simulation system described in (4.15), we identified the following systems from six randomly generated data

sets. In the augmented system, all the simulation matrices, $\{A, B, C, D\}$, are lumped into the state matrix $A_{augmented}$.

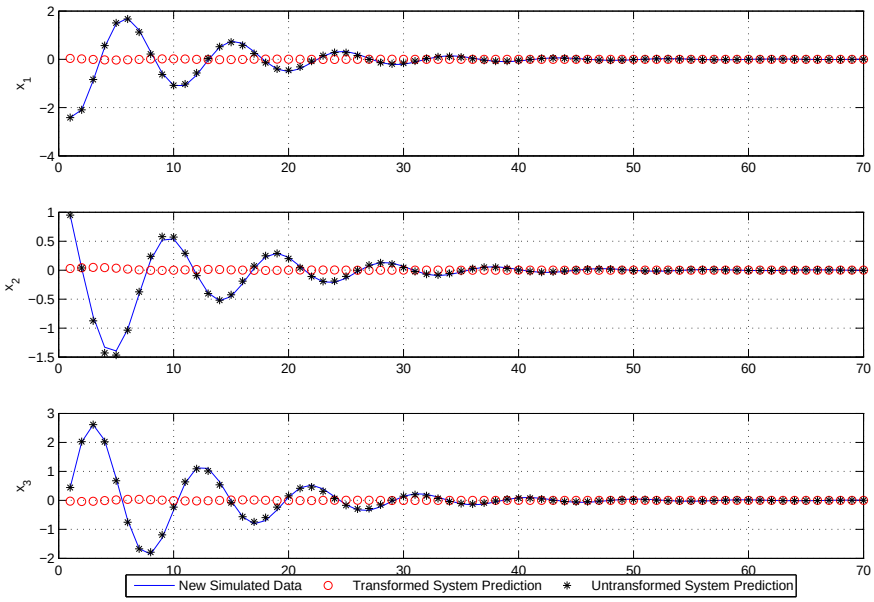


Figure 4.2: Comparison of the prediction ability of the original EM identified system and the untransformed system.

Table 4.6: EM estimated A parameter of augmented state space.

Actual					
0.360	-0.160	0.160	0.310	0.020	-0.030
-0.250	0.280	0.270	-0.020	0.150	-0.030
-0.570	-0.440	1.110	0.050	-0.030	0.290
0.252	-1.312	0.332	0.412	-0.326	0.364
0.758	0.398	-0.465	0.254	0.228	-0.013
-0.795	0.409	0.036	-0.599	0.201	0.283
ID System 1					
0.608	-0.051	0.493	-4.976	-8.474	-5.047
0.424	0.000	0.371	9.005	12.820	6.663
0.034	-0.015	0.018	11.108	20.173	6.978
0.005	-0.004	0.000	0.893	-0.083	0.459
0.007	-0.000	0.009	0.083	0.594	-0.251
0.002	0.002	0.001	-0.742	-0.002	0.565

ID System 2					
0.124	-0.171	0.292	-14.554	-23.751	-10.099
-0.274	0.420	-0.728	-4.246	-8.575	-3.973
-0.007	-0.042	0.083	-0.450	-3.029	1.170
0.004	0.002	-0.005	0.893	-0.083	0.459
-0.005	0.004	-0.009	0.082	0.593	-0.251
-0.002	0.002	-0.000	-0.742	-0.002	0.565
ID System 3					
-0.069	0.398	0.045	-8.138	-11.201	-6.368
-0.170	0.608	0.101	-6.729	-11.949	-3.161
-0.142	0.554	0.085	10.942	19.750	8.325
0.005	-0.002	-0.003	0.892	-0.084	0.459
0.002	-0.011	0.001	0.084	0.596	-0.250
-0.002	-0.002	-0.001	-0.741	-0.001	0.565

The top portion of the identified augmented A matrices are vastly different across the identified systems. The bottom portion which corresponds to the transformed $CB + D$ and CA matrices are similar across the identified systems but clearly different from the true form. Again, this similarity results because only the initial parameter value of π_1 was varied from identification to identification. If other initial parameters values were also varied, this similarity disappears.

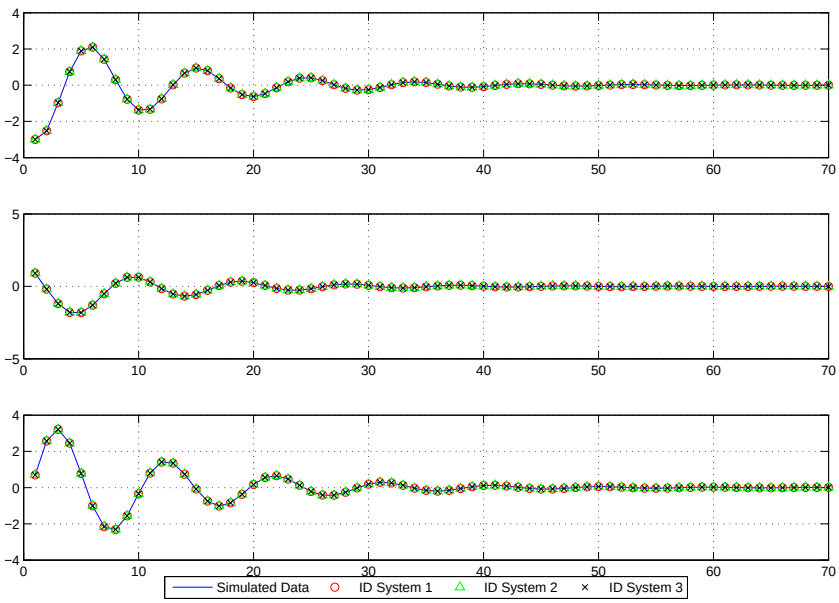


Figure 4.3: Comparison of EM identified augmented state space systems identified with different initial parameter sets.

From the plot, we see that the same degree of accuracy is achieved by all systems when matching the training

data sets. Because they all have the same input-output behavior, these systems must all be related by some arbitrary linear transforms. From the system in (4.11), we observe that equations (4.16) and (4.17) can still be used to solve for the reverse transforms. The untransformed A matrix follow.

Table 4.7: The re-transformed augmented A parameter.

Actual					
0.360	-0.160	0.160	0.310	0.020	-0.030
-0.250	0.280	0.270	-0.020	0.150	-0.030
-0.570	-0.440	1.110	0.050	-0.030	0.290
0.252	-1.312	0.332	0.412	-0.326	0.364
0.758	0.398	-0.465	0.254	0.228	-0.013
-0.795	0.409	0.036	-0.599	0.201	0.283
ID System 1					
0.360	-0.160	0.160	0.293	0.017	-0.027
-0.250	0.280	0.270	-0.028	0.151	-0.036
-0.570	-0.440	1.110	-0.001	-0.033	0.289
0.252	-1.312	0.332	0.407	-0.321	0.346
0.758	0.398	-0.465	0.256	0.225	-0.003
-0.795	0.409	0.036	-0.593	0.194	0.295
ID System 2					
0.360	-0.160	0.160	0.293	0.017	-0.027
-0.250	0.280	0.270	-0.028	0.151	-0.036
-0.570	-0.440	1.110	-0.001	-0.033	0.289
0.252	-1.312	0.332	0.407	-0.321	0.346
0.758	0.398	-0.465	0.256	0.225	-0.003
-0.795	0.409	0.036	-0.593	0.194	0.295
ID System 3					
0.360	-0.160	0.160	0.294	0.017	-0.027
-0.250	0.280	0.270	-0.028	0.151	-0.036
-0.570	-0.440	1.110	-0.000	-0.033	0.289
0.252	-1.312	0.332	0.407	-0.321	0.346
0.758	0.398	-0.465	0.255	0.225	-0.003
-0.795	0.409	0.036	-0.593	0.194	0.295

As in the original state space case, the system parameters are a lot closer to the simulation parameters after a reverse transformation. We test a new random data set to observe the difference in prediction ability. The results are very similar to those observed in the unaugmented system's case.

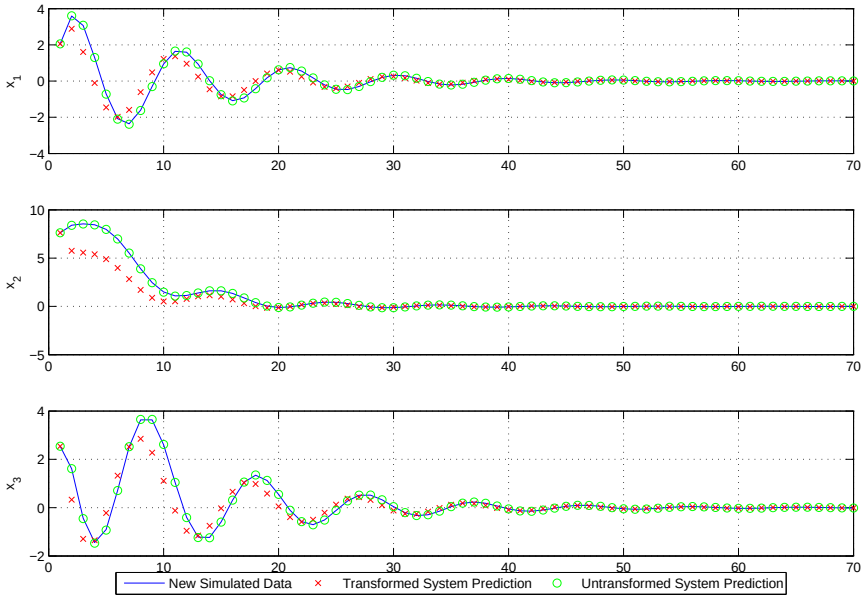


Figure 4.4: Comparison of prediction ability of identified augmented system and untransformed augmented system.

This simple example confirms our hypothesis of the state transformation form and proves that both the unaugmented and augmented systems are unidentifiable.

4.3 A Few Proposed Solutions

The fact that the state transformation appears irreversible does not render our identified system useless. Rather, it illuminates the fact that there is no true system when the hidden states have no physical correspondence to the actual, physical model. For, given only the observed gene transcript concentration data, we have no way of knowing exactly how many chemical pathways are inducing these concentrations. Instead, every transformed system contains the same information but certain transformations are more decipherable than others. For example, transformations that maximize the gene-to-gene dynamics matrix will facilitate gene-to-gene pathway inference while transformations that spread the dynamics between the hidden states and the gene states will obscure gene interactions.

Due to the limitations of our linear state space model, our model can at best only approximate a linearization

of the true gene-to-gene interaction matrix. Consider a state transform that can elucidate this linearization by maximizing the direct feedthrough portion of our augmented state matrix. Despite linear limitations, this approximation can potentially detail a general sketch of the activator/inhibitor activity flow in a complicated gene regulatory network. In this transformation, the influence of the hidden states should be minimized so that at most they only model interactions that can not be modeled by one-step gene-to-gene interactions. Thus ideally, with a choice transform, we hope to shift most of the state dynamics into the $CB+D$ partition of our state matrix, where we can decipher a linearization of the overall network, and keep the overall effect of the hidden states to a minimum.

For convenience, generalize our current model to a system of the form

$$\begin{aligned} \begin{bmatrix} x \\ g \end{bmatrix}_{k+1} &= \begin{bmatrix} A_1 & A_2 \\ A_3 & A_4 \end{bmatrix} \begin{bmatrix} x \\ g \end{bmatrix}_k \\ y_k &= \begin{bmatrix} 0 & I \end{bmatrix} \begin{bmatrix} x \\ g \end{bmatrix}_k \end{aligned} \tag{4.18}$$

which is subject to arbitrary transformations of the form

$$\begin{bmatrix} z \\ g \end{bmatrix}_k = \begin{bmatrix} F_1 & F_2 \\ 0 & I \end{bmatrix} \begin{bmatrix} x \\ g \end{bmatrix}_k$$

The transformed systems is

$$\begin{aligned} \begin{bmatrix} z \\ g \end{bmatrix}_{k+1} &= \begin{bmatrix} F_1A_1F_1^{-1} + F_2A_3F_1^{-1} & -F_1A_1F_1^{-1}F_2 - F_2A_3F_1^{-1}F_2 + F_1A_2 + F_2A_4 \\ A_3F_1^{-1} & -A_3F_1^{-1}F_2 + A_4 \end{bmatrix} \begin{bmatrix} z \\ g \end{bmatrix}_k \\ y_{k+1} &= \begin{bmatrix} 0 & I \end{bmatrix} \begin{bmatrix} z \\ g \end{bmatrix}_{k+1} \end{aligned} \tag{4.19}$$

4.3.1 Solution 1: Gram-Schmidt Orthogonalization

In this solution, we attempt to remove all influences of the hidden states via orthogonalization. As mentioned before, we are specifically interested in the interdynamics between the subset of states, g_k, which is roughly encoded in $A_{F4} = -A_3 F_1^{-1} F_2 + A_4$ in the transformed state-space in (4.19). Thus, even though we observe the states g_k, the interaction matrix we identify may bear no resemblance to A_4.

Theoretically, the original partition A_4 can be further subdivided into a linear combination of direct $g_{\{k,k-1,\ldots,1\}}$ to g_{k+1} interaction, $\tilde{A}_4$, and indirect, cross-sample $x_{\{k-1,k,\ldots,1\}}$ to g_{k+1} interaction, $A_3\alpha$. Substituting this subdivision of A_4 into A_{F4}, we get $A_{F4} = -A_3 F_1^{-1} F_2 + A_3\alpha + \tilde{A}_4$. In this form, we can isolate $\tilde{A}_4$ using Gram-Schmidt Orthogonalization to remove all components of A_3 from A_{F4}. Because we don't actually know A_3, we remove all components of $A_{F3} = A_3 F_1^{-1}$ from A_{F4}. Consider the orthogonal statement

$$\begin{aligned} A_{F3}^T \tilde{A}_4 &= 0 \\ A_{F3}^T \left(A_{F4} - A_{F3} G\right) &= 0 \\ A_{F3}^T A_{F4} &= A_{F3}^T A_{F3} G \\ \left(A_{F3}^T A_{F3}\right)^{-1} A_{F3}^T A_{F4} &= G \end{aligned} \tag{4.20}$$

where we solve for G as the composite proportion of A_{F3} in A_{F4}. With G known, we can now isolate $\tilde{A}_4$ via

$$\tilde{A}_4 = A_{F4} - A_{F3} G \tag{4.21}$$

and analyze it for the direct state to state dynamics among the g_k states.

In theory, the Gram-Schmidt Orthogonalization is removing all hidden state influences from the direct gene-to-gene matrix. But mathematically, this process is attempting to isolate two orthogonal matrices, which is plausible if we assume the effects of the hidden states is already minimal in A_{F4} such that every hidden state does not influence every gene state. In practice, this assumption may only hold true if the number of hidden states was less than the number of gene states so that the span of z was smaller than the span g. Then,

in this instance, there will exist portions of A_{F4} that map dynamics independent of those in A_3. However, if the number of hidden states was greater than or equal to the number of gene states, then between the arbitrary transformation, which could distribute weights randomly about the state matrix, and system noise, the A_3 matrix would most likely be full rank and span the same vector space as A_{F4}. In such an instance, the orthogonalization would yield a negligible matrix for $\tilde{A}_4$ because A_3 will appear to influence all gene state dynamics modeled by A_{F4}.

4.3.2 Solution 2: Optimization

In this section, we pose a least-squares optimization process for minimizing the hidden states, x. In terms of the state vector, the transformation in (4.9) affects only the hidden states, x, and leaves the genes states unchanged. Specifically, the transformed hidden state vector is a linear combination of the original hidden state vector and the gene state vector.

$$z_k = F_1 x_k + F_2 g_k \tag{4.22}$$

In our posed optimization, we want to minimize the effect of the transformed hidden states on the gene states. By (4.19), the gene state update equation is

$$g_{k+1} = A_3 F_1^{-1} z_k + [A_4 - A_3 F_1^{-1} F_2] g_k \tag{4.23}$$

Thus, our goal is to optimize the objective statement

$$\min\ A_3 F_1^{-1} z_k = \min\ A_3 x_k + A_3 F_1^{-1} F_2 g_k \tag{4.24}$$

The form of the right hand side statement implies that only the combined term $F_1^{-1} F_2$ affects the optimization so we can choose $F_1 = I$ without loss of generality. Again, we state our objective function below, in a least squares sense with our assumption on F_1.

$$\begin{aligned} \arg\min_{F_2}\ ||A_3 F_1^{-1} z_k|| &= \arg\min_{F_2}\ ||A_3(x_k + F_2 g_k)|| \\ &= \arg\min_{F_2}\ (x_k + F_2 g_k)^T A_3^T A_3 (x_k + F_2 g_k) \end{aligned} \tag{4.25}$$

So far, we have generically referred to the states at time instance k. In reality, we will employ all the time estimates of the states when we solve for F_2 to ensure that the overall, transformed hidden state is minimized. So, let $X = x_T, x_{T-1}, \ldots, x_1$ and $G = g_T, g_{T-1} \ldots, g_1$ and substitute these terms for x_k and g_k in (4.25). As we are only interested in the quadratic terms, we will employ a trace to isolate and sum over them.

$$\arg\min_{F_2} \; ||A_3 F_1^{-1} z_k|| = \arg\min_{F_2} \; tr\{(X + F_2 G)^T A_3^T A_3 (X + F_2 G)\} \tag{4.26}$$

We minimize the resultant quadratic, objective function by employing the methods described in chapter 2. First we derive its derivative, then we solve for F_2. The derivative of the objective function is

$$\frac{\partial}{\partial F_2} tr\{(X + F_2 G)^T A_3^T A_3 (X + F_2 G)\} = 2A_3^T A_3 X G^T + 2A_3^T A_3 F_2 G G^T \tag{4.27}$$

Setting this derivative to zero and solving for F_2, we get

$$F_2 = X G^{-1} \tag{4.28}$$

Combining F_1 and F_2 into a transform of the form in (4.9), we can apply this transform directly on the identified, transformed matrices.

4.3.3 Solution 3: Redefining the State Vector

In this section, we propose a redefinition of the hidden states in terms of only the gene states and solve the new state space parameters. The transformation in (4.9) adds arbitrary amounts of g to the hidden state z and obscures the direct interaction matrix as a measure of how g_k affects g_{k+1}. To counter this effect, we require a transformation of (4.18) into a unique, interpretable form. The solution is to generate an input-output description where the states are $[g_1^T, g_2^T, \ldots, g_T^T]^T$. For clarity, we remove the bias state from the A matrix and consider it as the constant that it is. Let b_x denote the biases identified on the x states and b_g the biases identified on the g states.

Consider the simplest case, where the number of hidden states is equal to the number of gene states. We

begin by rewriting (4.8) so that the state is $[g_{k-1}, g_k]^T$. Since

$$g_{k+1} = A_3 x_k + A_4 g_k + b_g$$

we must redefine x_k in terms of the gene states, g_k and g_{k-1}. Begin with the bottom equation in (4.18).

$$\begin{aligned} g_k &= A_3 x_{k-1} + A_4 g_{k-1} + b_g \\ x_{k-1} &= A_3^{-1}(g_k - A_4 g_{k-1} - b_g) \end{aligned} \tag{4.29}$$

where A_3 is assumed to be nonsingular; we will deal with the singular case later. Substitute (4.29) into the top line of (4.18) to get

$$\begin{aligned} x_k &= A_1 A_3^{-1}(g_k - A_4 g_{k-1} - b_g) + A_2 g_{k-1} + b_x \\ x_k &= A_1 A_3^{-1} g_k + (A_2 - A_1 A_3^{-1} A_4) g_{k-1} + (b_x - A_1 A_3^{-1} b_g) \end{aligned} \tag{4.30}$$

Substituting (4.30) back into (4.18), we get

$$\begin{aligned} g_{k+1} &= A_3 A_1 A_3^{-1} g_k + A_3(A_2 - A_1 A_3^{-1} A_4) g_{k-1} + A_3(b_x - A_1 A_3^{-1} b_g) + A_4 g_k + b_g \\ g_{k+1} &= A_3(A_2 - A_1 A_3^{-1} A_4) g_{k-1} + (A_3 A_1 A_3^{-1} + A_4) g_k + (A_3 b_x - A_3 A_1 A_3^{-1} b_g + b_g) \end{aligned} \tag{4.31}$$

The modified state-space representation becomes

$$\begin{aligned} \begin{bmatrix} g_k \\ g_{k+1} \end{bmatrix} &= \begin{bmatrix} 0 & I \\ A_3(A_2 - A_1 A_3^{-1} A_4) & A_3 A_1 A_3^{-1} + A_4 \end{bmatrix} \begin{bmatrix} g_{k-1} \\ g_k \end{bmatrix} + \begin{bmatrix} 0 \\ A_3 b_x - A_3 A_1 A_3^{-1} b_g + b_g \end{bmatrix} \\ y_k &= \begin{bmatrix} 0 & I \end{bmatrix} \begin{bmatrix} g_{k-1} \\ g_k \end{bmatrix} \end{aligned} \tag{4.32}$$

Denote $\tilde{A}_3 = A_3(A_2 - A_1 A_3^{-1} A_4)$ and $\tilde{A}_4 = A_3 A_1 A_3^{-1} + A_4$. In this transformation, $\tilde{A}_4$ represents the direct interaction of genes from one time step to the next and $\tilde{A}_3$ represents delayed interactions that occurr indirectly through unknown intermediaries. This formulation is much easier to interpret and more transparent than the unknown states formulation of (4.18). Further, it can be easily adapted to incorporate user-defined input vector.

Now consider the general case in which the number of hidden states does not equal the number of gene states. Again, we wish to substitute a function of $g_k, g_{k+1}, \ldots, g_{k+m}$ for x_k. Writing

$$
\begin{aligned}
g_{k+1} &= A_3 x_k + A_4 g_k && x_{k+1} = A_1 x_k + A_2 g_k \\
g_{k+2} &= A_3 x_{k+1} + A_4 g_{k+1} \\
&= A_3 A_1 x_k + A_3 A_2 g_k + A_4 g_{k+1} && x_{k+2} = A_1 x_{k+1} + A_2 g_{k+1} \\
g_{k+3} &= A_3 x_{k+2} + A_4 g_{k+2} \\
&= A_3 A_1 x_{k+1} + A_3 A_2 g_{k+1} + A_4 g_{k+2} \\
&= A_3 A_1^2 x_k + A_3 A_1 A_2 g_k + A_3 A_2 g_{k+1} + A_4 g_{k+2} && x_{k+3} = A_1 x_{k+2} + A_2 g_{k+2} \\
g_{k+4} &= A_3 x_{k+3} + A_4 g_{k+3} \\
&= A_3 A_1 x_{k+2} + A_3 A_2 g_{k+2} + A_4 g_{k+3} \\
&= A_3 A_1^2 x_{k+1} + A_3 A_1 A_2 g_{k+1} + A_3 A_2 g_{k+2} + A_4 g_{k+3} \\
&= A_3 A_1^3 x_k + A_3 A_1^2 A_2 g_k + A_3 A_1 A_2 g_{k+1} + A_3 A_2 g_{k+2} + A_4 g_{k+3} \\
&\vdots
\end{aligned}
$$

can obtain m "measurements" of x_k, concatenate all m equations and rearrange the state to get

$$
\begin{bmatrix} A_3 \\ A_3 A_1 \\ A_3 A_1^2 \\ A_3 A_1^3 \\ \vdots \\ A_3 A_1^{m-1} \end{bmatrix} x_k = \begin{bmatrix} g_{k+1} - A_4 g_k \\ g_{k+2} - A_4 g_{k+1} - A_3 A_2 g_k \\ g_{k+3} - A_4 g_{k+2} - A_3 A_2 g_{k+1} - A_3 A_1 A_2 g_k \\ g_{k+4} - A_4 g_{k+3} - A_3 A_2 g_{k+2} - A_3 A_1 A_2 g_{k+1} - A_3 A_1^2 A_2 g_k \\ \vdots \\ g_{k+m} - A_4 g_{k+m-1} - \sum_{j=1}^{m-1} A_3 A_1^{j-1} A_2 g_{k+m-1-j} \end{bmatrix} \tag{4.33}
$$

$$x_k = \begin{bmatrix} A_3 \\ A_3A_1 \\ A_3A_1^2 \\ A_3A_1^3 \\ \vdots \\ A_3A_1^{m-1} \end{bmatrix}^{-1} \begin{bmatrix} g_{k+1} - A_4g_k \\ g_{k+2} - A_4g_{k+1} - A_3A_2g_k \\ g_{k+3} - A_4g_{k+2} - A_3A_2g_{k+1} - A_3A_1A_2g_k \\ g_{k+4} - A_4g_{k+3} - A_3A_2g_{k+2} - A_3A_1A_2g_{k+1} - A_3A_1^2A_2g_k \\ \vdots \\ g_{k+m} - A_4g_{k+m-1} - \sum_{j=1}^{m-1} A_3A_1^{j-1}A_2g_{k+m-1-j} \end{bmatrix} \tag{4.34}$$

The coefficient matrix before x_k is a part of the observability matrix of the x states from measurements of the g states. If this matrix is not full rank for some m, then there exists some x states that are unobservable from the g states and the number of hidden states should be reduced. In general, especially due to noise, the coefficient matrix will be full rank for some m. When this statement holds true, we can choose n linearly independent rows of (4.33), where n is the number of hidden states, and solve for x_k in terms of $g_k, g_{k+1}, \ldots, g_{k+m}$ such that

$$x_k = S_0g_{k+m} + S_1g_{k+m-1} + \ldots + S_mg_k$$

Noting the pattern in (4.33), we can propagate the gene state forward once more to get the next gene state.

$$g_{k+m+1} = A_3A_1^m x_k + A_4g_{k+m} + \sum_{l=1}^{m-1} A_3A_1^lA_2g_{k+m-l}$$

Substituting for x_k yields the state equation

$$\begin{bmatrix} g_{k+1} \\ g_{k+2} \\ g_{k+3} \\ \vdots \\ g_{k+m+1} \end{bmatrix} = \begin{bmatrix} 0 & I & 0 & \ldots & 0 \\ 0 & 0 & I & \ldots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \ldots & I \\ \tilde{A}_{m+1} & \tilde{A}_m & \tilde{A}_{m-1} & \ldots & \tilde{A}_1 \end{bmatrix} \begin{bmatrix} g_k \\ g_{k+1} \\ g_{k+2} \\ \vdots \\ g_{k+m} \end{bmatrix}$$
$$y_{k+m} = \begin{bmatrix} 0 & 0 & 0 & \ldots & I \end{bmatrix} \begin{bmatrix} g_k \\ g_{k+1} \\ g_{k+2} \\ \vdots \\ g_{k+m} \end{bmatrix} \tag{4.35}$$

For convenience, it makes sense to choose the number of hidden states as a multiple of the number of gene states so that the left hand side matrix in (4.33) is most likely invertible. This doesn't always hold true and in general more sophisticated techniques should be used.

Chapter 5

Nonlinear Analysis

Having studied the linear case and state transformations in depth, we are ready to move on to the nonlinear test cases. This section starts by applying the model we outlined in the previous section to a simple nonlinear example outlined in Ressom *et al.*[23] and progresses to increasingly more complicated nonlinear systems. This section concludes with a complex, simulated gene regulatory network and applies the third state transformation solution described in the previous section.

5.1 A Few Simple Detection Cases

To test the merit of our revised model, we consider a simple nonlinear test system from Ressom et. al [23]. We first evaluate the functionality of our state-space augmentation with g_k and then evaluate the advantage of the bias state.

5.1.1 A Nonlinear Simulation with Zero Steady-State

We begin with a sparse, primarily linear state space system in the discrete domain, described by the following set of difference equations

$$
\begin{aligned}
x_1[k+1] &= 0.5x_5[k] \\
x_2[k+1] &= 0.3x_1[k] \\
x_3[k+1] &= -0.6x_2[k] + x_4[k] \\
x_4[k+1] &= 0.7x_4^2[k] \\
x_5[k+1] &= -0.8\sqrt{(|x_3[k]|)}
\end{aligned}
\tag{5.1}
$$

Two nonlinearities are embedded in the fourth and fifth states. The fourth nonlinear state affects only itself but the fifth nonlinear state propagates to state one and through state one to state two and three. Despite its nonlinearity, this system has unperturbed steady-states of zero. Ten data sets of twenty five time point data were generated from this system using five initial conditions of the standard basis and five arbitrarily generated vectors. Noise was added both during the data simulation and after to simulate state and observation noise; the state and observation noises were generated from Gaussian distributions with mean zero and variances 1×10^{-5} and 1×10^{-3}, respectively.

The data was first input to the EM algorithm described by Shumway [24] in (3.3). The C matrix was assumed to be an identity matrix because all states were observed in the simulation. The re-estimated parameters are tabulated below in Table 5.1

Table 5.1: EM estimated parameters (A,Q,R) for nonlinear simulation 1 using Shumway's system.

A					Q (10^{-5})				
0.01	-0.00	0.01	0.00	0.51	1.09	0.20	-0.48	-1.41	15.79
0.31	0.00	-0.01	-0.01	-0.01	0.20	0.80	0.02	1.07	-0.79
-0.00	-0.61	0.02	1.02	-0.01	-0.48	0.02	5.23	4.13	2.66
-0.03	-0.04	-0.06	0.60	0.04	-1.41	1.07	4.13	26.60	24.38
0.10	-0.23	-0.63	-0.26	0.26	15.79	-0.79	2.66	24.38	1072.62
C (statically assigned)					R (10^{-4})				
1	0	0	0	0	1.03	-0.04	0.16	0.18	0.28
0	1	0	0	0	-0.04	0.91	-0.04	-0.13	0.05
0	0	1	0	0	0.16	-0.04	0.69	-0.02	-0.04
0	0	0	1	0	0.18	-0.13	-0.02	0.61	-0.00
0	0	0	0	1	0.28	0.05	-0.04	-0.00	0.49

Similarly, the data was processed as though it were generated from the augmented system, (4.2). In this form, the C matrix was assumed to be

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

The resultant estimated parameters were

Table 5.2: EM estimated parameters (A,Q,R) for nonlinear simulation 1 using augmented system.

A									
-0.15	-0.09	0.05	-0.09	-0.01	0.08	-0.12	-0.01	-0.01	0.14
1.60	-0.11	-0.33	0.08	-0.12	-0.32	1.85	-0.24	0.41	-0.62
0.89	0.64	0.69	-0.00	-0.13	0.64	-1.61	0.54	0.24	0.84
0.65	0.66	-0.12	-0.40	-0.36	-0.61	-1.42	0.41	0.06	0.53
0.36	0.82	0.34	0.06	-0.06	-0.18	0.57	0.80	1.22	-0.34
0.01	0.00	0.00	-0.00	0.00	-0.00	0.01	-0.01	0.00	0.50
0.01	-0.00	0.00	0.00	-0.00	0.30	-0.00	0.00	-0.00	-0.00
-0.01	-0.00	-0.00	-0.00	0.00	-0.00	-0.61	0.00	1.01	0.01
-0.08	-0.01	0.00	0.01	-0.01	-0.04	-0.02	-0.08	0.57	0.03
0.05	-0.20	0.11	0.01	-0.00	0.01	-0.16	-0.46	-0.39	0.28

Q (10^{-3})									
0.38	-0.07	0.77	-2.72	-0.20	0.03	0.01	-0.03	0.03	0.10
-0.07	0.22	-0.64	1.54	-0.78	0.02	-0.06	-0.05	-0.04	-0.02
0.77	-0.64	4.56	-7.43	-2.12	0.01	0.24	0.25	0.15	0.31
-2.72	1.54	-7.43	29.25	-8.62	0.12	-0.28	0.02	-0.47	-0.68
-0.20	-0.78	-2.12	-8.62	24.60	-0.33	0.20	-0.51	0.27	-0.38
0.03	0.02	0.01	0.12	-0.33	0.02	0.00	-0.02	-0.01	0.01
0.01	-0.06	0.24	-0.28	0.20	0.00	0.04	0.00	0.01	0.01
-0.03	-0.05	0.25	0.02	-0.51	-0.02	0.00	0.08	0.01	-0.00
0.03	-0.04	0.15	-0.47	0.27	-0.01	0.01	0.01	0.01	0.01
0.10	-0.02	0.31	-0.68	-0.38	0.01	0.01	-0.00	0.01	0.04

R (10^{-4})				
0.84	0.03	0.21	0.02	-0.13
0.03	0.51	-0.05	-0.18	-0.03
0.21	-0.05	0.21	-0.22	-0.05
0.02	-0.18	-0.22	0.94	-0.20
-0.13	-0.03	-0.05	-0.20	0.84

Lastly, the data was evaluated for the augmented, biased system of (4.3). In this form, the C matrix is

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

The estimated parameters for this instance were

Table 5.3: EM estimated parameters (A,Q,R) for nonlinear simulation 1 using augmented bias system.

A										
1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
0.03	-0.45	-0.02	0.06	-0.11	0.00	0.29	0.01	0.15	-0.56	0.16
0.01	0.41	-0.03	0.31	0.19	-0.03	-0.83	1.73	-0.23	0.43	-0.31
-0.03	-1.64	-0.30	0.19	0.15	0.42	0.30	-0.07	-0.15	0.04	-0.38
-0.19	-0.95	0.79	0.20	0.11	0.05	-0.30	-0.07	1.53	-0.69	-0.19
-0.16	1.30	0.09	-0.77	0.30	0.19	0.06	-0.26	-0.18	2.89	-1.06
-0.00	0.00	0.00	-0.00	0.00	-0.00	-0.00	0.01	-0.01	0.01	0.49
0.00	0.00	0.00	-0.00	-0.00	0.00	0.30	-0.00	-0.01	0.02	0.00
-0.00	-0.01	-0.00	0.00	-0.00	0.00	-0.01	-0.60	0.00	0.99	-0.01
0.00	-0.08	-0.02	0.00	0.01	-0.01	-0.03	-0.03	-0.07	0.53	0.0409
-0.05	-0.03	-0.21	0.04	-0.00	0.02	-0.01	-0.07	-0.51	-0.32	0.14

Q (10^{-3})										
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
0.00	0.41	0.29	1.48	-1.01	0.96	0.04	0.04	-0.04	0.07	0.02
0.00	0.29	1.10	2.35	-0.28	3.18	-0.02	0.04	-0.13	0.08	-0.01
0.00	1.48	2.35	10.84	-11.49	9.27	0.10	0.10	-0.16	0.35	0.11
0.00	-1.01	-0.28	-11.49	42.09	-3.01	-0.22	-0.39	0.29	-0.55	-0.85
0.0000	0.96	3.18	9.27	-3.01	24.79	-0.75	0.59	-0.15	0.43	0.07
0.00	0.04	-0.02	0.10	-0.22	-0.75	0.05	-0.03	0.01	-0.01	-0.01
0.00	0.04	0.044	0.1	-0.39	0.59	-0.03	0.06	-0.02	0.03	0.02
0.00	-0.04	-0.13	-0.16	0.29	-0.15	0.01	-0.02	0.06	-0.02	-0.02
0.00	0.07	0.08	0.35	-0.55	0.43	-0.01	0.03	-0.02	0.02	0.01
0.00	0.02	-0.01	0.11	-0.85	0.07	-0.01	0.02	-0.02	0.01	0.03

R (10^{-4})				
0.63	0.19	-0.10	0.16	0.12
0.19	0.53	-0.07	-0.20	-0.23
-0.10	-0.07	0.11	0.04	-0.00
0.16	-0.20	0.04	0.98	0.10
0.12	-0.23	-0.00	0.10	0.28

As the purpose of this section is to compare the performance of the augmented systems to that of the unaugmented system, a common basis for comparison is needed. In the linear analysis chapter, the state matrix

of the simulation was compared to the estimated state matrix to evaluate the accuracy of the EM approximation; a similar comparison can ensue if we only consider a section of the augmented system parameters.

In the simulated system, the state vector is observed exactly and can also be considered the output vector. The corresponding state components in the augmented system are the g_k states which are also observed exactly in the output vector y_k. Stated thus, the augmented parameters that most closely match the state matrix of the simulated system is the portion of A that models the direct dynamics between the previous g_{k-1} states and the current g_k states. This portion of A occurs at the bottom right corner of the state matrix and is referred to as the direct feed-through matrix, or $CB + D$ in (4.2) and (4.3). The direct feed-through partitions of the augmented A and the augmented-with-bias A are tabulated below along with the entire A matrix from the unaugmented state-space for easier comparison.

Table 5.4: Comparison of estimated A and feed-through matrices for nonlinear simulation 1.

A (unaugmented)				
0.01	-0.00	0.01	0.00	<u>0.51</u>
<u>0.31</u>	0.00	-0.01	-0.01	-0.01
-0.00	<u>-0.61</u>	0.02	<u>1.02</u>	-0.01
-0.03	-0.04	-0.06	<u>0.60</u>	0.04
0.10	-0.23	<u>-0.63</u>	-0.26	0.26
A_{CB+D} (augmented)				
-0.00	0.01	-0.01	0.00	<u>0.50</u>
<u>0.30</u>	-0.00	0.00	-0.00	-0.00
-0.00	<u>-0.61</u>	0.00	<u>1.01</u>	0.01
-0.04	-0.02	-0.08	<u>0.57</u>	0.03
0.01	-0.16	<u>-0.46</u>	-0.39	0.28
A_{CB+D} (augmented with bias)				
-0.00	0.01	-0.01	0.01	<u>0.49</u>
<u>0.30</u>	-0.00	-0.01	0.02	0.00
-0.01	<u>-0.60</u>	0.00	<u>0.99</u>	-0.01
-0.03	-0.03	-0.07	<u>0.53</u>	0.04
-0.01	-0.07	<u>-0.51</u>	-0.32	0.14

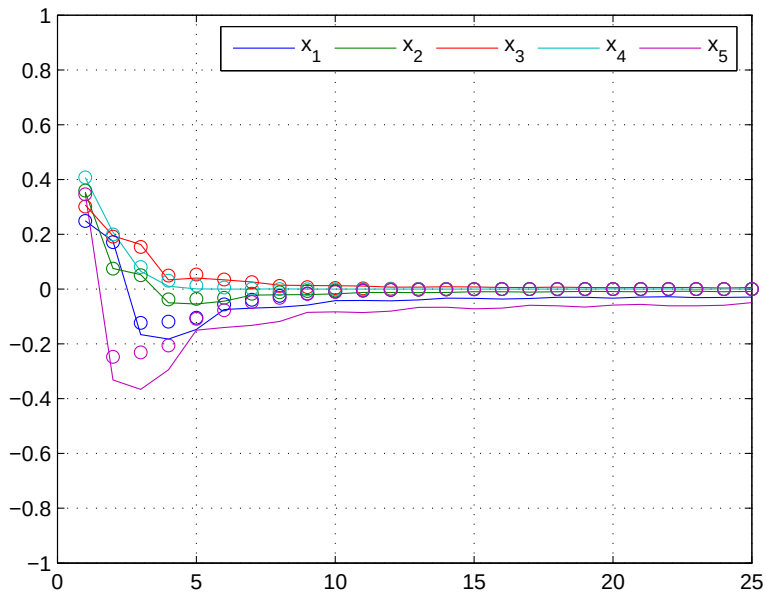
The underlined coefficients denote the non-zero terms in the actual simulation system. In terms of estimation accuracy, all three matrices are particularly accurate when estimating the coefficients for the linear states x_1 through x_3. For these linear states, the trivial coefficients corresponding to non-existent state-to-state interactions are at least an order of magnitude lower than accurately detected coefficients. Unsurprisingly, the matrices fare comparatively worse when approximating the coefficients of the nonlinear states. There is greater estimation error for the existing simulation coefficients and more false detection of non-existent

system interactions. Because our state-space model is a linear system, the occurrence of these inaccuracies when our model is applied to a nonlinear system is not unexpected. Our system can only approximate a linearization of these nonlinearities so the greater the deviation of the nonlinearity from a linear function, the greater the prediction error of our model. This proportionality can be observed in states x_4 and x_5. State x_4 contains only one nonlinearity, a power of two, which is easily approximated at small numbers about zero with a line. In contrast, state x_5 has two nonlinearities, a square root and an absolute value function, the latter of which is particularly hard to approximate about zero. The difference in our model's performance for these two states is clearly observable; there is significantly more false detection error for x_5 than for x_4 and about the same amount of true detection error for both.

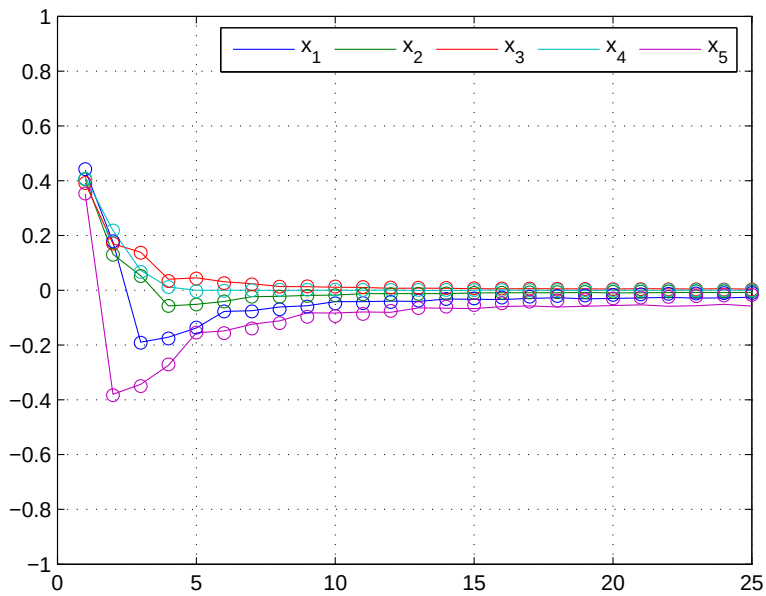
Compared to each other, all three matrices match very closely on the significant, non-zero terms and vary more on the trivial terms. In particular, the three matrices differ most in the coefficients for the nonlinear states on rows four and five. While the unaugmented system is the closest to matching the actual coefficients of the simulation system, it also has the most error or deviation from zero in its trivial terms. The other two systems fare slightly better in the estimation of the trivial terms and of the two, the augmented system with bias seems to be the clearer.

Thus far, the extra states of the augmented system have been disregarded in favor of a comparable set of feed-through matrices. Their effect on the system performance can not be observed in these matrices and this oversight can lead to a deceptive perception of system performance. A visual comparison of the data fitting and prediction ability of the systems offers a more complete perspective of the effect of these extra states. The average fit over all the initial conditions are graphed below in Figure 5.1.

The plots are examples of the fitting capability of each system. Other training data sets offer similar results. But just these three graphical representations are sufficient proof of the superior performance of the augmented system with a bias state. Its data fit matches the original simulation data exactly. While the augmented system appears to perform to a similar caliber at first, it deteriorates more and more toward the steady-states. The unaugmented system degenerates so rapidly for the nonlinear states that even the linear states are affected. Thus, the augmentation offers a significant advantage over the original, unaugmented system.

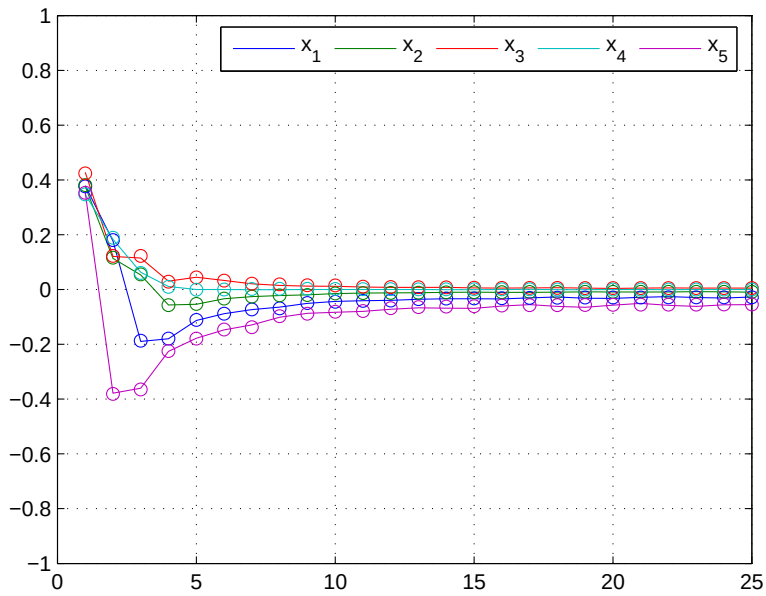


(a) Identified unaugmented system (circles). Simulation system (solid line)



(b) Identified augmented system (circles). Simulation system (solid line)

Figure 5.1: Comparison of EM identified systems with simulation systems.



(c) Identified augmented bias system (circles). Simulation system (solid line)

Figure 5.1: Comparison of EM identified systems with simulation systems.

Consider a comparison of the prediction ability of each system shown below.

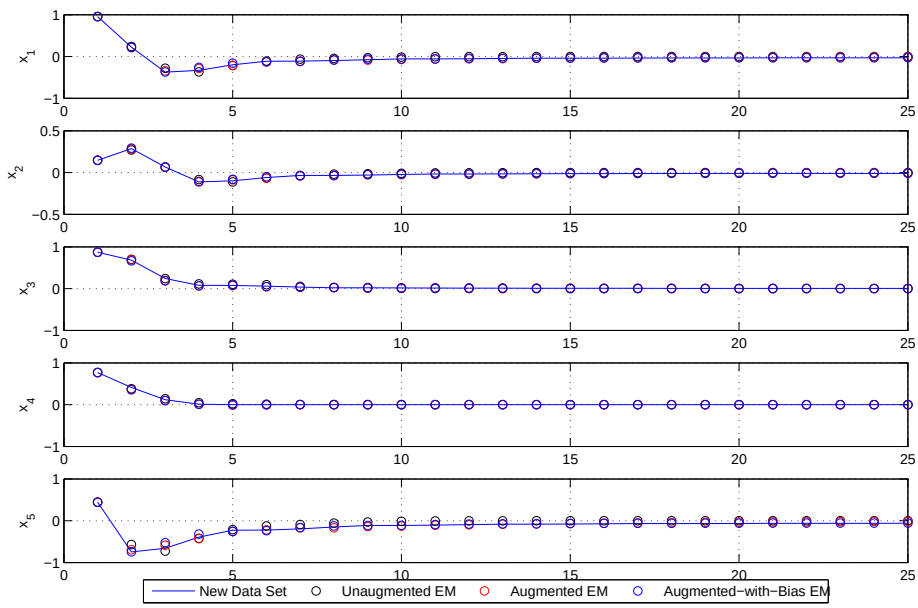


Figure 5.2: Comparison of unmodified and modified EM systems' prediction performance.

In this example, the systems appear more evenly matched although the unaugmented system still has noticeably more error than the augmented systems.

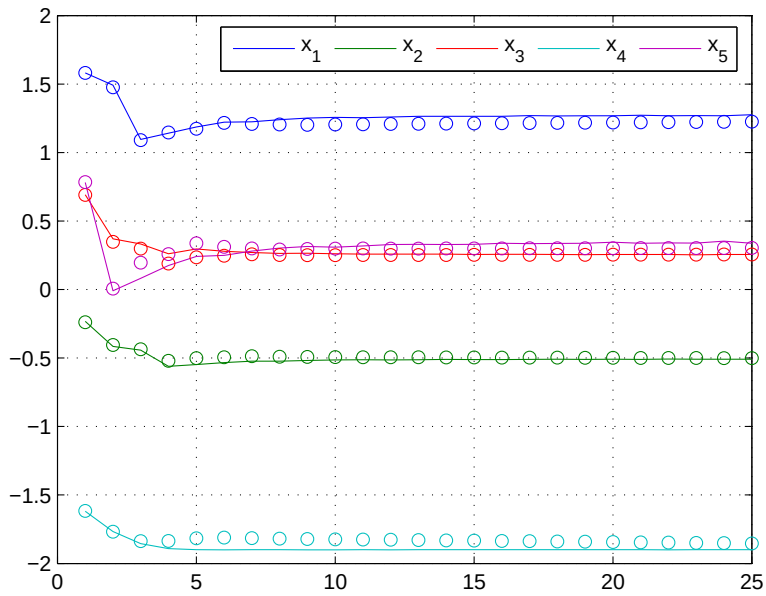
5.1.2 A Nonlinear Simulation with Steady-State

Suppose we consider the same simulation system with a few artificially added steady-states. Specifically, during data simulation, steady-state values of $\{1.3, -0.5, 0.25, -1.9, 0.4\}$ were added on top of the state dynamics. Ten sets of data were generated with the same types of initial conditions and noise vectors and processed through the three systems. The comparable feed-through matrices are tabulated below.

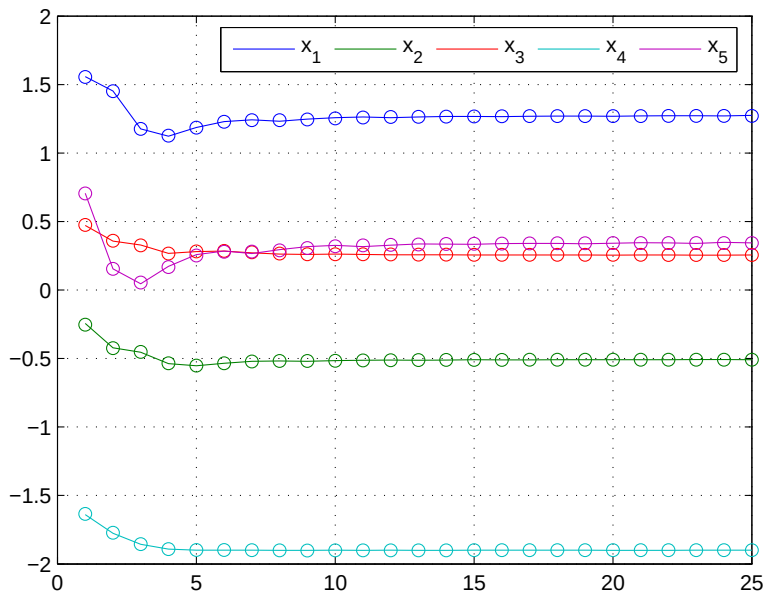
Table 5.5: Comparison of estimated A and feed-through matrices for nonlinear simulation 2.

A (unaugmented)				
1.55	4.49	3.41	-0.13	1.48
-0.35	-1.74	-1.40	0.25	-0.38
0.57	0.04	0.89	0.40	0.20
-2.15	-6.76	-4.94	0.50	-1.41
0.34	1.36	0.52	-0.20	0.38
A_{CB+D} (augmented)				
0.21	0.02	0.23	-0.41	0.52
0.13	-0.01	-0.20	0.33	-0.03
0.35	-0.57	0.40	0.31	0.04
-0.17	0.02	-0.24	0.85	0.03
0.03	-0.05	-0.57	-0.19	0.13
A_{CB+D} (augmented with bias)				
-0.01	0.02	0.01	-0.02	0.51
0.30	0.00	0.01	-0.01	0.00
0.01	-0.58	0.03	0.95	0.01
0.00	-0.01	-0.00	0.40	0.05
-0.05	-0.04	-0.60	-0.12	0.06

In this instance, the advantage of the bias state becomes readily apparent even in the state matrices. The original simulation coefficients are barely discernible in the unaugmented system matrix. The augmented system matrix improves upon it significantly such that some coefficients resemble the simulation coefficients but almost all trivial coefficients are on the same order of magnitude as non-trivial ones regardless of whether the state is linear or not. In contrast, the augmented bias system matrix remains virtually unaffected. Coefficient estimation for the linear states are very accurate for both trivial and non-trivial simulation coefficients. Estimation performance on the nonlinear states resembles simulation coefficients. As noted before, there is greater estimation and detection error in the fifth state than the fourth. Visual examples affirm our analysis.

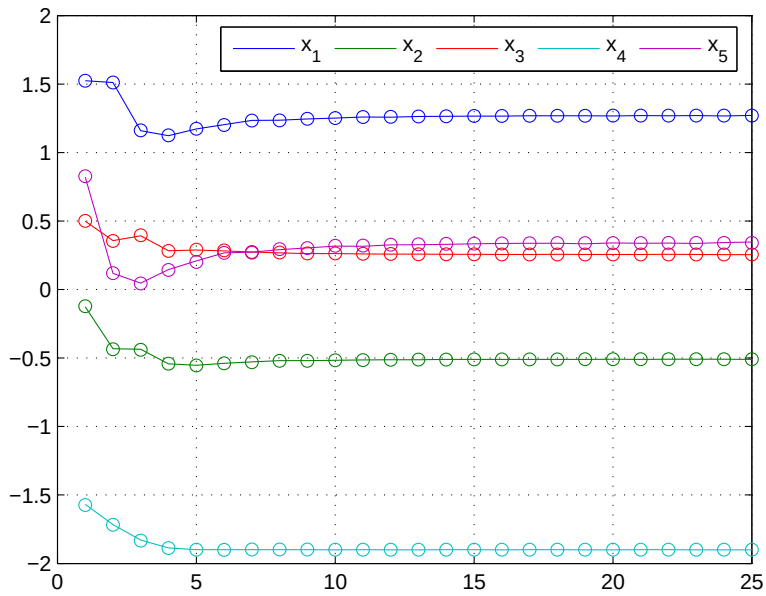


(a) Identified unaugmented system (circles). Simulation system (solid line)



(b) Identified augmented system (circles). Simulation system (solid line)

Figure 5.3: Comparison of modified and unmodified EM systems' prediction performance. New data set (solid line), EM prediction (circles).



(c) Identified augmented bias system (circles). Simulation system (solid line)

Figure 5.3: Comparison of modified and unmodified EM systems' prediction performance. New data set (solid line), EM prediction (circles).

The fitting and prediction of the augmented system with a bias state is consistently superior to the other two systems. The augmented system without the bias state improves upon the unaugmented state but not to the degree of the bias system. Both augmented systems are facilitated by the presence of extra, hidden states. The augmented model with the bias state has a distinct advantage over all previous models described thus far and holds enough merit to be tested with another, more complicated nonlinear system.

5.2 A More Complicated System

In his 2000 article, *Biochemical systems analysis of genome-wide expression data*, Voit *et al.*[28] describe an S-system model of yeast glycolysis. The model simulates the fermentation pathway mapped in the figure below.

The X variables labeled from one through five in the figure denote the observable chemical concentrations of the system. They are dependent on the internal chemical flux rates labeled as V variables in the figure. These variables are independent from external control and unobservable hence they are equivalent to the

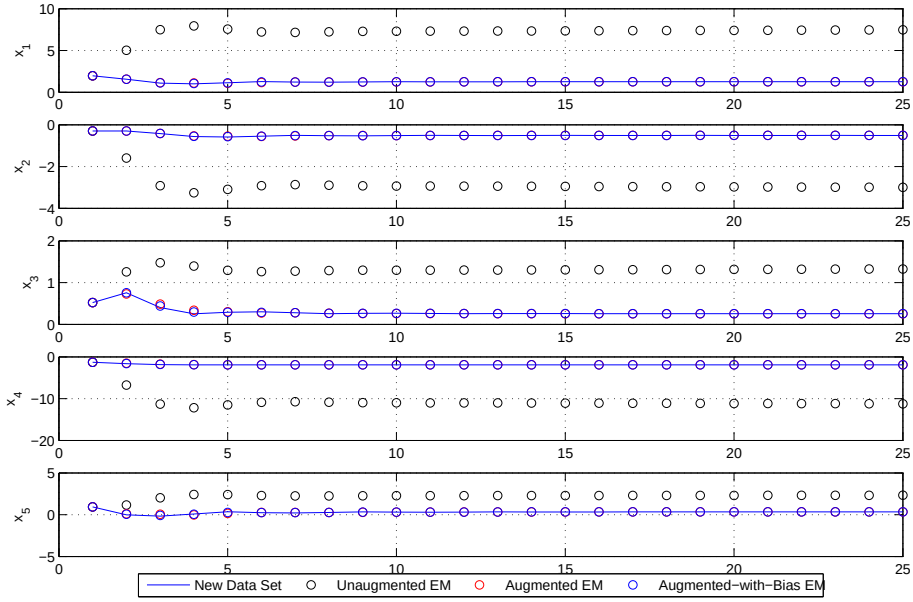


Figure 5.4: Comparison of modified and unmodified EM systems' prediction performance. New data set (solid line), EM prediction (circles).

unknown dynamics of the system, which are identified as hidden states in our augmented-with-bias EM model. The observed variables are logically modeled as the observation states. Voit's S-system is shown below.

$$
\begin{aligned}
\dot{x}_1 &= 0.9018x_2^{-0.2344}x_6 - 3.1833x_1^{0.7464}x_5^{0.0243}x_7 \\
\dot{x}_2 &= 3.1833x_1^{0.7464}x_5^{0.0243}x_7 - 0.8187x_2^{0.664}x_5^{-0.354}x_8^{0.899}x_{11}^{0.0008}x_{15}^{0.0963} \\
\dot{x}_3 &= 0.5234x_2^{0.7318}x_5^{-0.3941}x_8 - 0.0148x_3^{0.584}x_4^{0.03}x_5^{0.119}x_9^{0.944}x_{12}^{0.056}x_{14}^{-0.575} \\
\dot{x}_4 &= 0.022x_3^{0.6159}x_5^{0.1308}x_9x_{14}^{-0.6088} - 0.0945x_3^{0.05}x_4^{0.533}x_5^{-0.0822}x_{10} \\
\dot{x}_5 &= 0.0880x_3^{0.3505}x_4^{0.2665}x_5^{0.0243}x_9^{0.5}x_{10}^{0.5}x_{14}^{-0.3044} \\
&\quad - 3.3270x_1^{0.2134}x_2^{0.190}x_5^{0.362}x_7^{0.286}x_8^{0.257}x_{11}^{0.0002}x_{13}^{0.457}
\end{aligned}
\tag{5.2}
$$

Voit's model is outlined in the continuous time domain as a set of differential equations. States one through five correspond to the observed variables, X, while states six through fifteen correspond to the hidden variables, V. Their specific correspondence to chemical structures and their known steady-states are tabulated below.

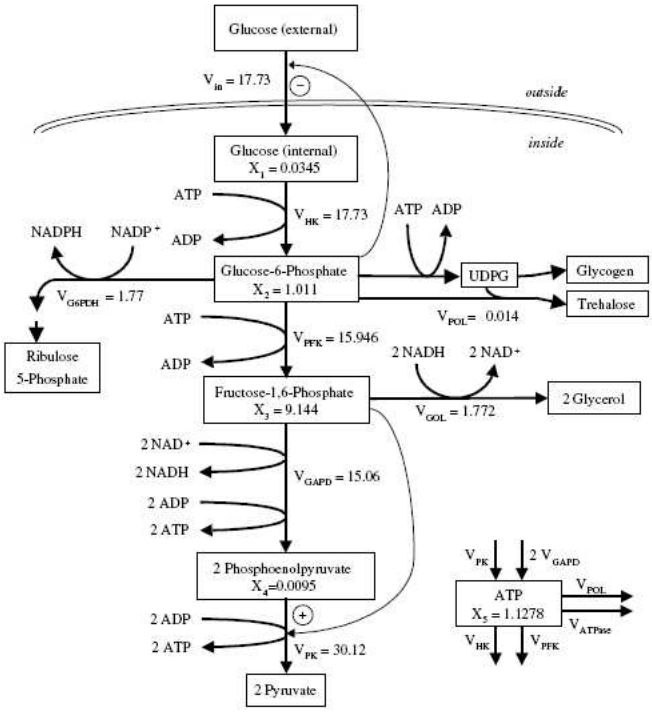


Figure 5.5: Visual model simulated by Voit's S-system [28].

Table 5.6: Chemical name of states and their steady states [28].

State Variable	Chemical Name	Steady State (nM/min)
x_1	internal glucose	0.0345
x_2	G6P	1.011
x_3	FDP	9.144
x_4	PEP	0.0095
x_5	ATP	1.1278
x_6	glucose transport	19.7
x_7	HXK/GLK	68.5
x_8	PFK	31.7
x_9	TDH	49.9
x_{10}	PK	3440
x_{11}	trehalose/glycogen production	14.31
x_{12}	glycerol production	203
x_{13}	ATPase	25.1
x_{14}	NADH/NAD+	0.042
x_{15}	G6PDH	1.77

This system was chosen as our second nonlinear simulation, despite its high degree of nonlinearity, because it is a straightforward feed-forward system with no feedback loops. Furthermore, in simulation, the model

simplifies as all hidden states are held constant at their steady-state, reducing the model complexity to

$$
\begin{aligned}
\dot{x}_1 &= 17.7655x_2^{-0.2344} - 218.0561x_1^{0.7464}x_5^{0.0243} \\
\dot{x}_2 &= 218.0561x_1^{0.7464}x_5^{0.0243} - 19.3182x_2^{0.664}x_5^{-0.354} \\
\dot{x}_3 &= 16.5918x_2^{0.7318}x_5^{-0.3941} - 4.9445x_3^{0.584}x_4^{0.03}x_5^{0.119} \\
\dot{x}_4 &= 7.5629x_3^{0.6159}x_5^{0.1308} - 325.08x_3^{0.05}x_4^{0.533}x_5^{-0.0822} \\
\dot{x}_5 &= 95.6963x_3^{0.3505}x_4^{0.2665}x_5^{0.0243} - 118.226x_1^{0.2134}x_2^{0.190}x_5^{0.362}
\end{aligned}
\tag{5.3}
$$

Thirty sets of data were generated using five initial conditions that perturbed the observed states individually from their known steady-state and twenty-five random initial conditions close to the known steady-state. The data was processed four times through the EM algorithm with a different number of assumed hidden states each time: 0, 5, 10, 15, and 20. The comparable feed-through portions of the state matrices are shown below, preceded by a graphical examples of each system's fitting ability. The data was post-processed to remove negative, discrete real poles before being converted into the continuous systems compared below.

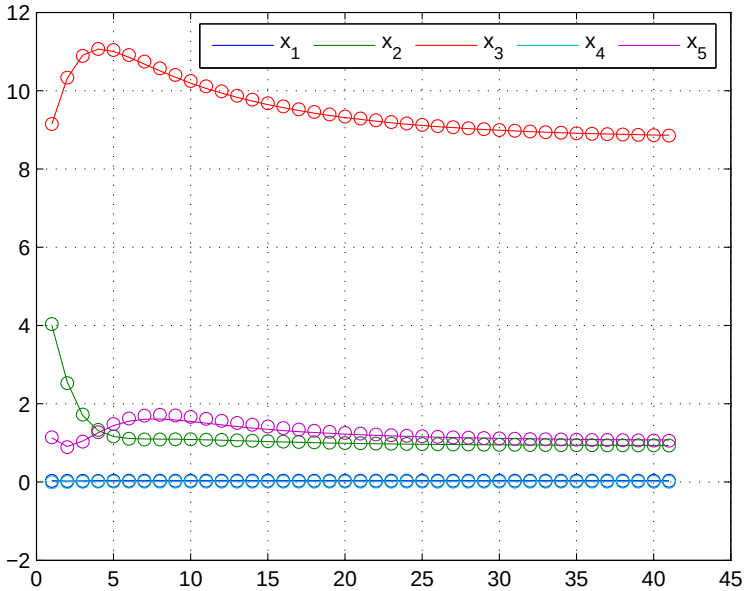


Figure 5.6: Comparison of system performance with no hidden states. Training data set (solid line), EM re-construction (circles).

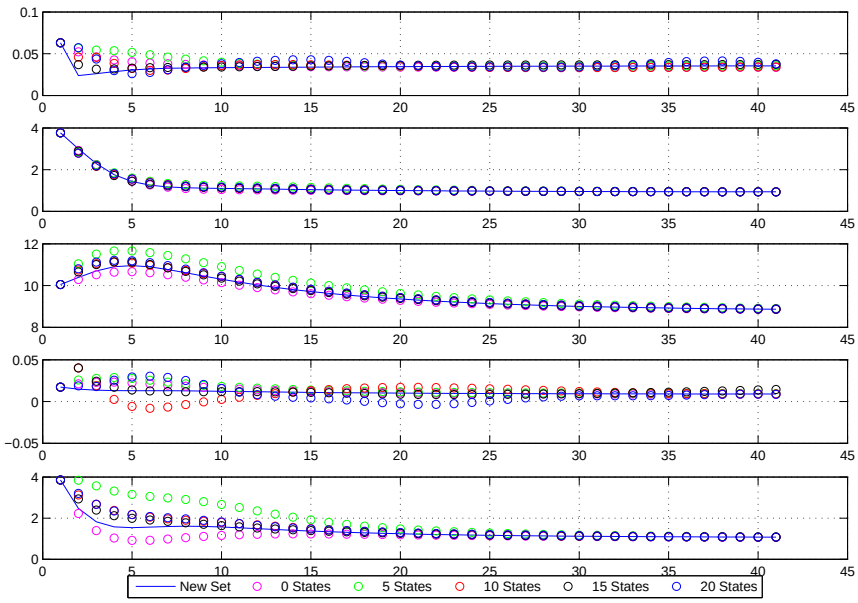
Table 5.7: Comparison of feedthrough matrices for identified systems with 0, 5, 10, 15, and 20 states.

Discrete CB+D Matrix					Continuous CB+D Matrix				
System identified assuming no hidden states									
0.60	-0.00	-0.00	0.21	0.00	-10.10	-0.01	-0.02	5.94	0.01
7.65	0.53	0.02	-0.56	0.09	290.04	-12.37	0.25	-19.87	3.10
-8.14	0.36	0.94	-6.52	-0.15	-304.17	9.78	-1.05	-149.79	-4.41
-0.03	0.00	-0.00	0.85	-0.00	-1.43	0.07	-0.00	-3.20	-0.01
-8.15	-0.14	0.18	-18.42	0.62	-208.05	-5.23	4.60	-449.52	-8.59
System identified assuming 5 hidden states									
-0.03	0.00	-0.00	0.04	0.01	-294.11	1.49	-1.46	20.47	2.11
0.05	0.58	0.00	-0.15	0.09	111.85	-9.73	0.85	-234.25	2.30
0.19	0.44	0.86	0.34	-0.03	148.73	10.05	-2.57	94.09	-1.91
-0.02	0.00	0.00	0.02	0.00	-10.99	1.15	0.38	-265.99	0.51
-1.56	0.12	-0.02	1.28	0.89	-297.03	4.49	-2.74	45.79	1.79
System identified assuming 10 hidden states									
-0.01	-0.00	-0.01	0.02	0.00	24.90	0.29	0.47	116.84	-1.31
0.41	0.58	0.02	-0.04	0.07	-130.42	-11.50	-0.81	-240.81	5.30
0.55	0.36	0.89	-0.15	-0.09	-109.18	8.23	-3.58	-222.42	0.31
-0.08	0.00	-0.00	-0.01	0.01	-324.26	-2.85	-2.77	-480.37	6.27
-1.01	-0.06	0.06	0.03	0.76	-52.56	-3.11	1.18	5.49	-4.69
System identified assuming 15 hidden states									
0.00	-0.00	-0.00	-0.01	0.00	-274.84	-0.84	-1.71	-6.49	3.38
0.33	0.59	0.00	-0.13	0.09	25.49	-6.98	-0.87	-211.70	4.52
0.04	0.32	0.89	0.04	-0.06	-67.80	5.72	-2.30	153.75	-1.87
-0.04	0.00	-0.00	-0.02	0.01	-9.56	2.20	-1.27	-268.00	3.41
-1.69	-0.16	0.06	0.87	0.81	-922.66	-10.09	-4.20	242.29	6.31
System identified assuming 20 hidden states									
-0.06	-0.00	-0.01	0.01	0.01	24.24	-1.77	1.27	196.34	-2.37
0.85	0.57	0.03	-0.23	0.09	-63.96	-7.48	-1.24	-277.32	6.31
0.34	0.33	0.86	0.08	-0.06	-3.08	8.41	-4.27	-67.33	-0.21
-0.04	0.00	0.00	-0.25	0.00	-40.56	0.36	-0.61	-118.94	1.19
-1.14	-0.12	0.00	0.21	0.82	-19.51	-5.51	-0.48	138.16	-2.92

From the table, it is very hard to observe any strong association between the estimated feed-through matrices and the actual system coefficients, even in sign. The estimates fluctuate over several magnitudes from system to system and the signs seem completely arbitrary. This is probably a direct result of the arbitrary matrix transform on the state matrices but may also be exacerbated by the high degree of nonlinearity in the system. This nonlinearity is particularly obvious in the state noise covariance matrix, Q. Magnitudes in the Q matrix for this test case are several degrees of magnitude higher than those observed in the linear test cases. This is understandable as all nonlinearities that can not be fit in the training cases are accounted for in the noise covariance matrices.

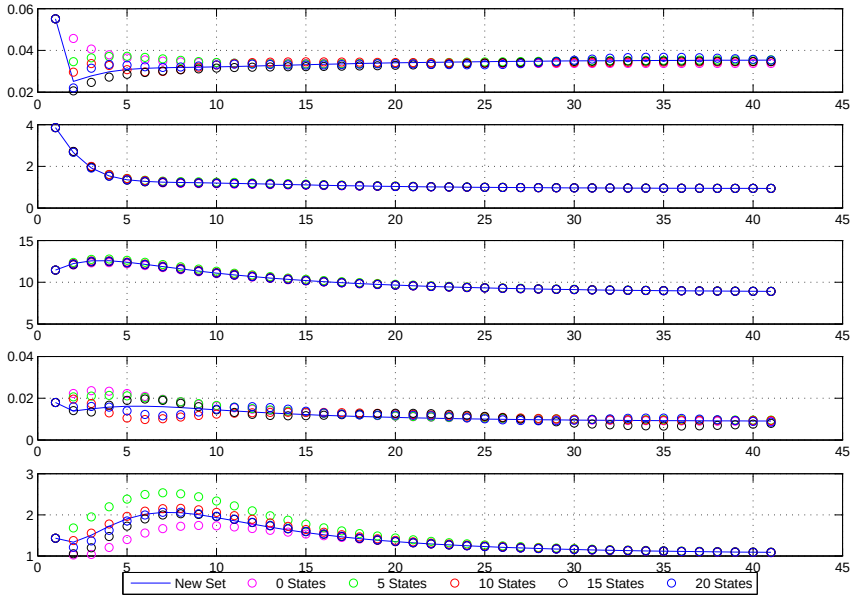
On a more positive note, all five systems consistently match their training data to approximately the same degree of precision. Only one plot is shown for reference but it is sufficient to show the performance of the systems. The fit is similarly to that observed in the chapter 3 in accuracy. Hence, we observe a few prediction cases, shown below in Figure 5.7.

In general, the prediction ability of the estimated systems vary depending on the random initial condition of the new data set. Yet overall, the systems with 10, 15, and 20 hidden states appear to predict the best despite large differences in their parameter coefficients. In general, they manage to track the system dynamics fairly well with either some occasional delay or magnitude error.

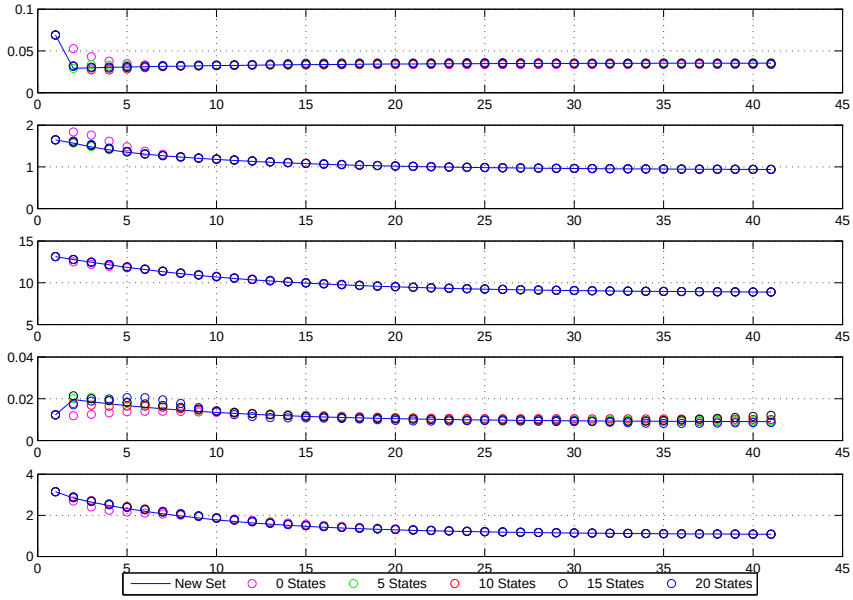


(a) A poor prediction.

Figure 5.7: Sample of prediction ability of EM identified systems.



(b) A fair prediction.

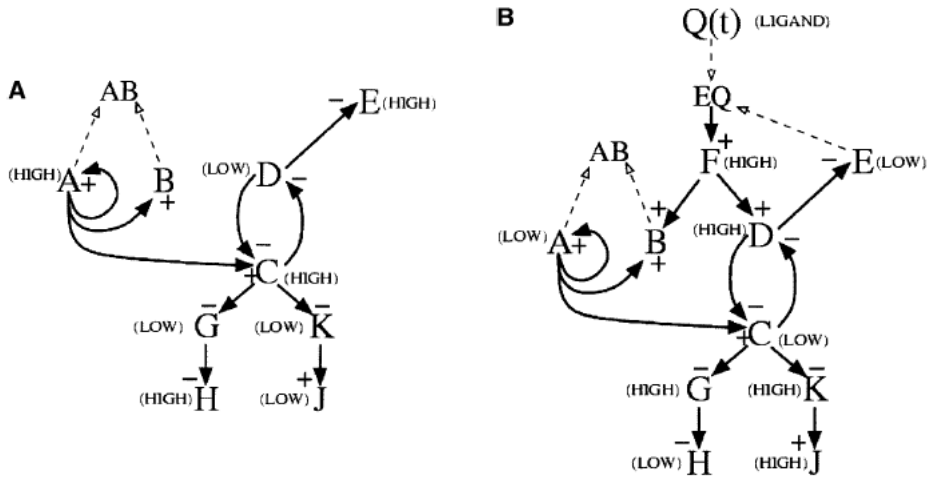


(c) A good prediction.

Figure 5.7: Sample of prediction ability of EM identified systems.

5.3 An Advanced Gene Regulatory Network

In their 2003 article, Zak *et al.*[33] mathematically describe an in silico genetic regulatory network. Although the described system models no particular physical network, it was assembled from several common themes observed in the literature. The themes, known in the article as regulatory motifs, are unique interaction mechanisms characteristic to actual gene networks. Specifically, the motifs are cascade, mutual repression, auto-activation and sequestration, and agonist-induced receptor down-regulation. The cascade motif is a one-way chain reaction of activation or repression through multiple genes. The mutual repression motif consists of a pair of genes that repress each other. The auto-activation and sequestration motif is the self-regulation of certain genes. And lastly, the agonist-induced receptor down-regulation motif models receptor binding that subsequently leads to receptor down-regulation. The assembled network is mapped below. In the ensuing network, genes C, G, H, J, and K comprise the cascade motif; genes C and D the mutual repressive motif; genes A and B the auto-activation; and genes E, F, and D the agonist-induced receptor down-regulation motif. Although the model sounds simple with only ten observed variables, 118 reactions with 44 species modeled by differential equations and 97 parameters were used to simulate the proteins, transcripts, dimers, and promoters that generated its dynamics.



(a) System without Input (b) System with Input

Figure 5.8: Visual model simulated by Zak's network [33].

We tested the identification of each motif separately, starting with the cascade motif. Each motif was tested with 5, 10, 15, and 20 hidden states. The individual testing procedure and results for each case are outlined below. The final section combines all four motifs and outlines the results.

5.3.1 Motif 1: Cascade

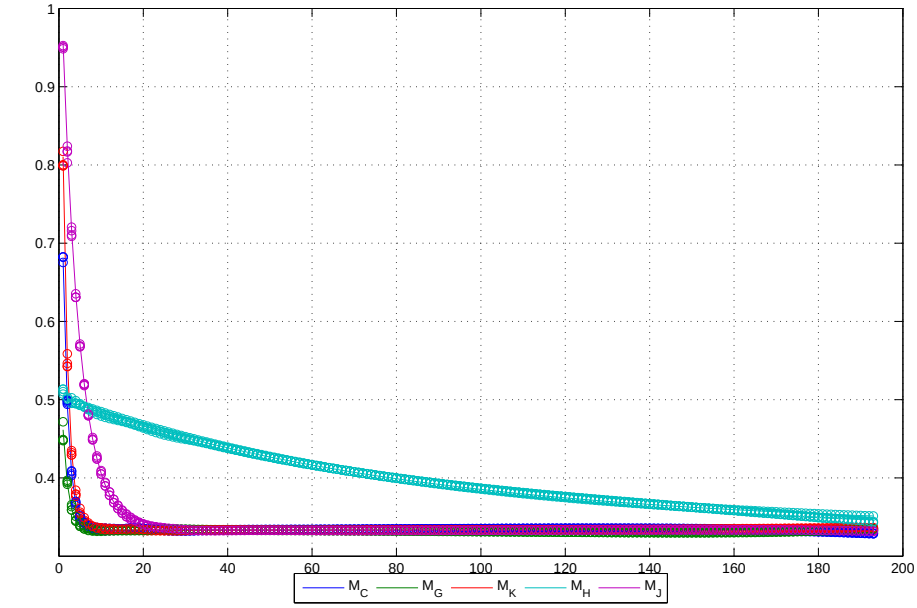
To isolate the cascade motif in Zak's model, we zeroed out the transcriptional parameters that did not contribute to the creation of the relevant C, G, H, J, and K genes. As a result, all other motifs were suppressed because their gene transcript concentrations could only decay to or remain at steady state. In Zak's model, the transcriptional parameters were

Table 5.8: Relevant transcriptional parameters to the cascade motif [33].

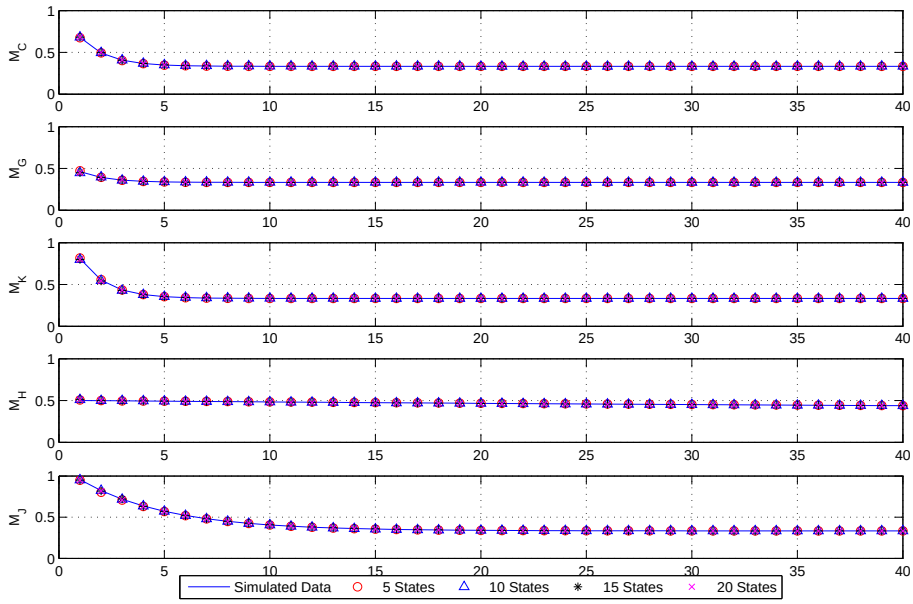
Gene C	
$k_{RP_{CAD}}$	7.3×10^{-4}
$k_{RAP_{CAD}}$	7.3×10^{-4}
$k_{RD_2P_{CAD}}$	0
$k_{RAD_2P_{CAD}}$	7.3×10^{-2}
Gene G	
$k_{RP_{GC}}$	7.3×10^{-1}
$k_{RC_2P_{GC}}$	7.3×10^{-4}
Gene H	
$k_{RP_{HG}}$	4.2×10^{-3}
$k_{RG_2P_{HG}}$	4.2×10^{-6}
Gene J	
$k_{RP_{JK}}$	7.3×10^{-4}
$k_{RK_2P_{JK}}$	7.3×10^{-1}
Gene K	
$k_{RP_{KC}}$	7.3×10^{-1}
$k_{RC_2P_{KC}}$	7.3×10^{-4}

In simulation, 50 sets of data were generated using random initial conditions and normalized about the highest measurement of each state across all 50 data sets. Because the cascade motif involves no input, all observable dynamics were stimulated using random, non-steady-state initial conditions. Despite the reduced number of states in this motif, the model is still very complex and the observed outputs have very different dynamic coefficients; a few states decay significantly slower than others. To reduce the number of measurement points and thus also the execution time of the EM algorithm, we sampled rapidly at the beginning of each simulation to capture fast exponential dynamics but then lengthened our sampling steps as the simulation wore on and increasing numbers of states reached steady-state. Although samplings steps varied during simulations, the pattern of sampling remained the same across all simulations. Overall, less data was input to the EM algorithm and the modified algorithm described in 3.4 was used to process the data. Thirty data sets were used to train the EM algorithm with 5, 10, 15, and 20 hidden states; the remaining twenty sets were saved to observe prediction during post-simulation analysis. Some plots of training and prediction

results and the comparable matrices follow.

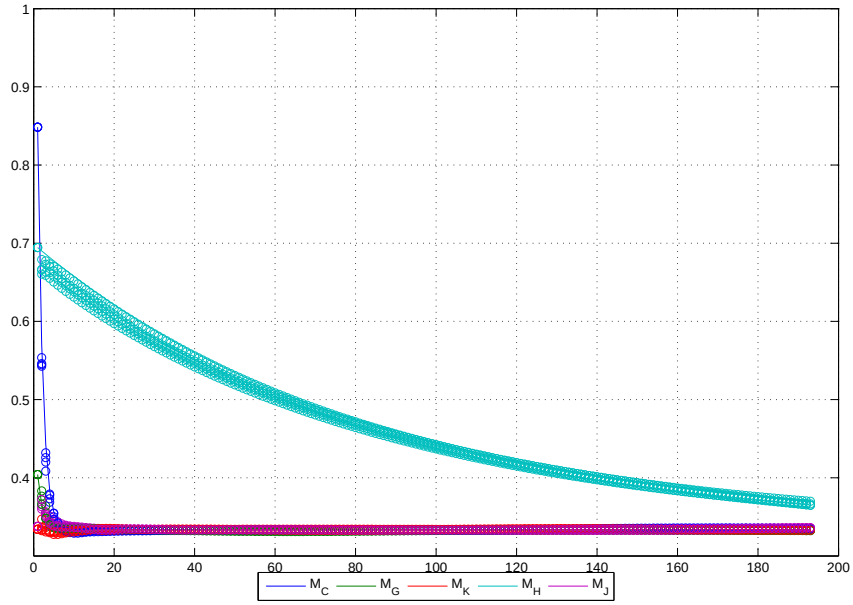


(a) Training data set fit.

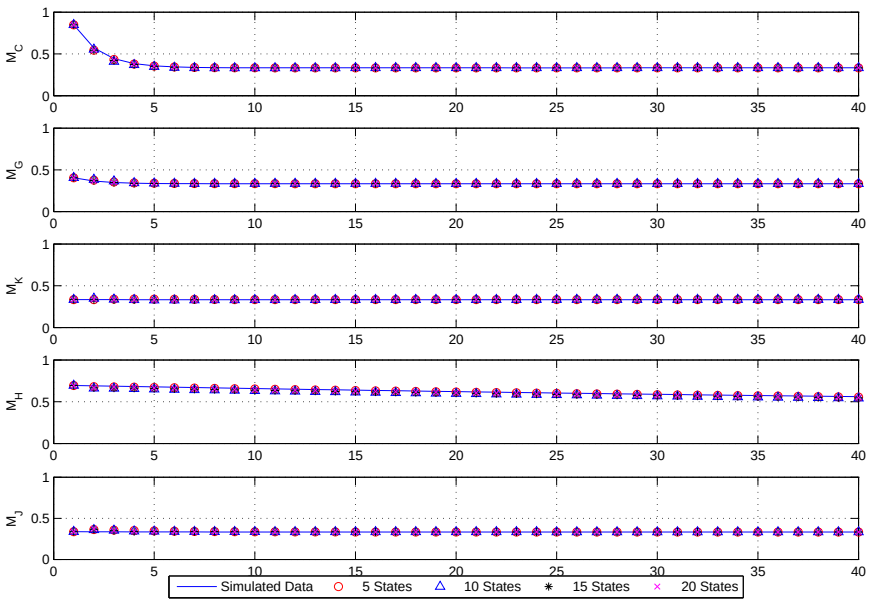


(b) Zoomed in training data set fit.

Figure 5.9: Comparison of identified model estimation and prediction for cascade motif.



(c) Prediction of new data set.



(d) Zoomed in prediction fit.

Figure 5.9: Comparison of identified model estimation and prediction for cascade motif.

Table 5.9: Comparison of feed-through matrices for Zak's cascade motif. Rows and columns denote, in order, $\{M_C, M_G, M_K, M_H, M_J\}$.

Discrete CB+D Matrix					Continuous CB+D Matrix (10^{-2})				
System identified assuming 5 hidden states									
0.41	0.03	0.02	0.01	0.01	-2.56	-0.31	1.34	0.43	0.99
0.03	0.41	0.01	0.00	0.02	0.23	-2.76	0.61	0.28	0.88
-0.00	0.02	0.45	0.01	0.00	-0.11	-0.19	-1.18	0.29	0.26
-0.01	-0.00	0.02	0.95	0.00	-0.60	-0.54	0.97	0.11	0.70
0.05	0.02	-0.01	-0.01	0.70	0.51	0.99	-0.84	-0.29	-1.64
System identified assuming 10 hidden states									
0.41	0.03	0.03	0.15	0.01	-2.58	-0.24	1.71	9.29	1.28
0.01	0.44	-0.01	0.11	0.01	-0.50	-1.85	0.03	-0.48	1.43
0.02	0.02	0.44	0.05	-0.01	-0.14	0.30	-0.18	5.53	-0.41
0.01	0.01	0.01	0.63	-0.03	0.79	0.49	-0.47	-4.38	-1.64
0.03	0.04	-0.00	0.05	0.70	0.65	0.40	-0.17	-2.97	-1.84
System identified assuming 15 hidden states									
0.36	0.06	0.01	0.13	0.02	-1.59	0.67	0.09	3.21	1.56
-0.04	0.44	-0.00	0.09	-0.00	-0.31	-2.15	0.15	-1.10	0.37
0.03	0.02	0.46	0.00	0.01	0.75	0.69	-1.33	1.05	0.69
-0.00	0.01	0.02	0.69	-0.05	0.21	0.36	-0.13	-3.04	-1.84
0.04	0.04	-0.02	0.04	0.69	1.61	0.04	-0.11	-4.83	-1.43
System identified assuming 20 hidden states									
0.38	0.04	-0.01	0.04	0.02	-1.58	0.02	0.07	1.03	0.35
0.01	0.42	0.03	-0.01	0.01	-0.00	-1.22	0.11	0.60	0.24
0.03	0.03	0.46	0.05	-0.00	0.14	-0.25	-0.93	-0.09	0.07
-0.01	0.01	0.01	0.70	-0.04	-0.35	-0.04	0.03	-1.97	-0.42
0.06	0.02	-0.03	-0.00	0.70	0.51	-0.55	0.16	-2.20	-1.14

Although the number of hidden states does not match the number of unmeasured variables in the actual model at all, the EM algorithm still trains our state-space model very well. The training data estimates appear very accurate in the plots above and the predictions only fair slightly worse. Only one prediction instance was shown here so there is not much room for comparison but the accuracy of prediction does tend to vary with initial state values. The closer random initial state values are to the trained data sets' initial state values, the better the prediction but as initial state values deviate more and more, predictions tend to deteriorate in terms of tracking. But even despite these losses in tracking accuracy, the general dynamic shape and speed of decay is still reasonably approximated in all predictions.

Unfortunately, we do not observe much information in the direct feedthrough matrix. But this problem arises for a multitude of reasons, the foremost of which is an unknown state transformation, as we have seen before in 5.2. A second reason lies in the structure of our model. As the number of hidden states do not come close

to the number of unseen variables in the simulation model, there is no physical interpretation of what these hidden states mean or do. Each hidden state could be modeling a combination of unseen variable dynamics or they could be approximating nonlinearities in all the variables. As we do not know the influence of the hidden states, we can not measure how much information is distributed to the feedthrough matrix and how much to the hidden states. This distribution of information itself can be a kind of state transformation as well.

Nevertheless, ideally, we hope to observe some semblance of a cascade effect in the coefficients such that the largest influence in each state's coefficient row corresponds to the state preceding it in the cascade. For example, since the order of the states is $\{M_C, M_G, M_K, M_H, M_J\}$ and M_C affects M_G and M_K directly, we expect to see the largest magnitudes in rows 2 and 3, the coefficient rows of M_G and M_K respectively, to be in column 1, the influence constant preceding M_C. In general, we do not see this effect because the highest magnitude coefficients appear in the diagonal of the feedthrough matrix, which corresponds to the genes' influence on themselves. This effect probably reflects the natural decay of a state. This decay appears in all the identified systems across all numbers of hidden states while other effects are transient and vary with the number of hidden states.

5.3.2 Motif 2: Mutual Repression

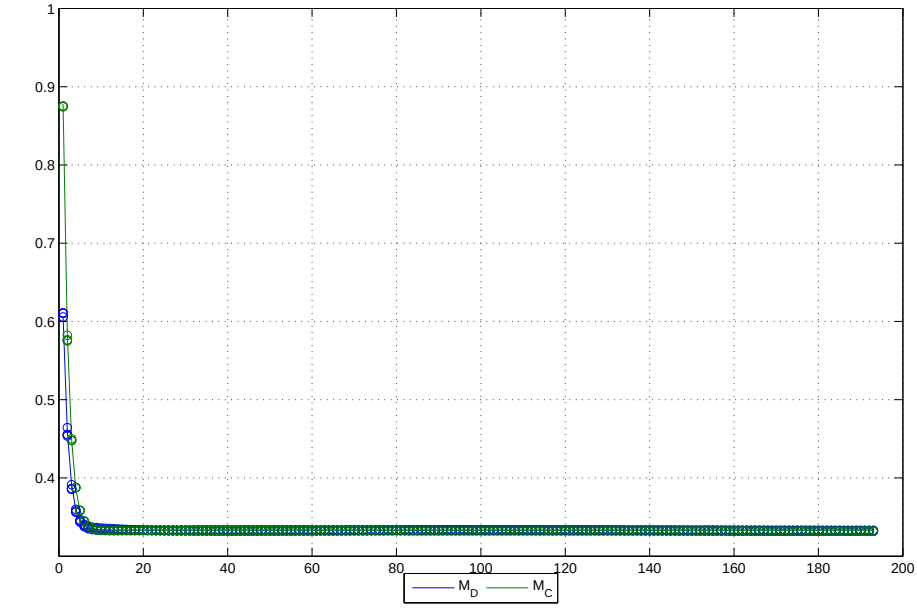
Similar to the procedure in the previous section, non-relevant transcripts were suppressed during simulation and data collection, leaving only genes C and D unaffected.

Table 5.10: Relevant transcriptional parameters to the mutual repression motif [33].

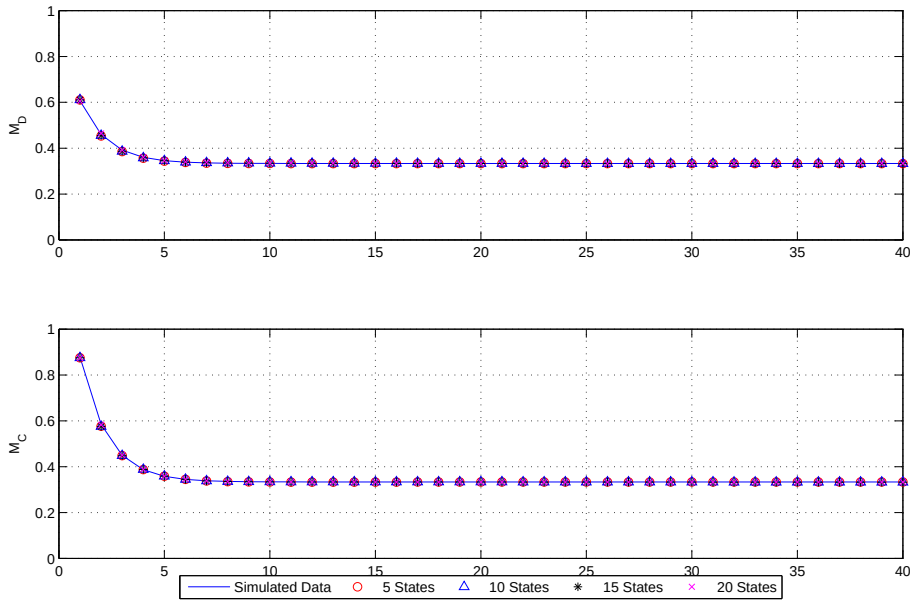
Gene C	
$k_{RP_{CAD}}$	7.3×10^{-4}
$k_{RAP_{CAD}}$	7.3×10^{-4}
$k_{RD_2P_{CAD}}$	0
$k_{RAD_2P_{CAD}}$	7.3×10^{-2}
Gene D	
$k_{RP_{DCF}}$	7.3×10^{-4}
$k_{RC_2P_{DCF}}$	0
$k_{RF_2P_{DCF}}$	7.3×10^{-1}
$k_{RC_2F_2P_{DCF}}$	7.3×10^{-1}

Fifty sets of data were generated using random initial conditions and input to the EM algorithm after being resampled at uneven time steps and normalized. Some sample figures of training and prediction are shown

below.

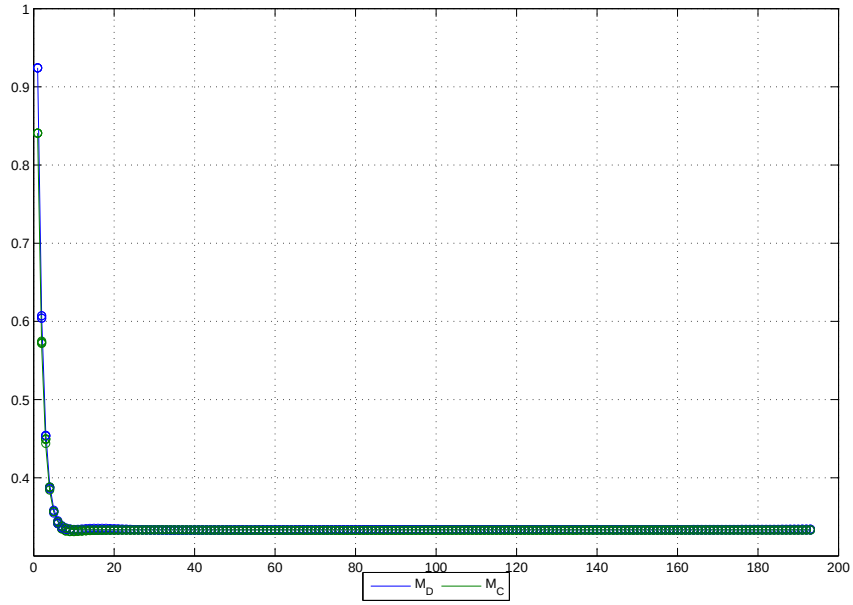


(a) Training data set fit.

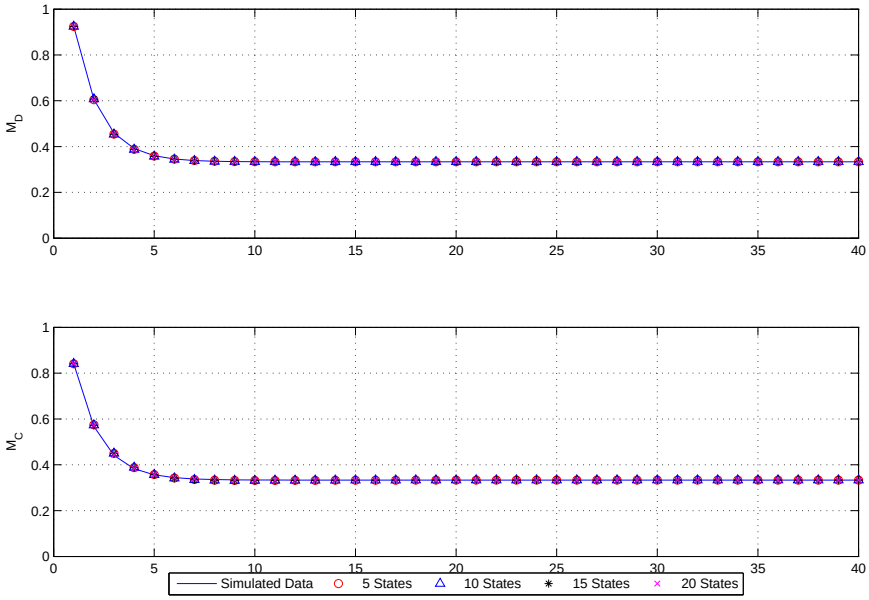


(b) Zoomed in training data set fit.

Figure 5.10: Comparison of identified model estimation and prediction for mutual repression motif.



(c) Prediction of new data set.



(d) Zoomed in prediction of new data set.

Figure 5.10: Comparison of identified model estimation and prediction for mutual repression motif.

There are significantly fewer parameters to estimate in this motif than in the previous motif due to the

decreased number of observed variables. Consequently, the dynamics of this motif are much easier to track and train, as is obvious in our recorded plots. The predictions are as accurate as the training estimates. In this motif, the states are, in order, $\{M_D, M_C\}$. As they mutually repress each other, we expect to observe negative magnitudes in the cross-state coefficients of the feedthrough matrix to mirror this effect. Unfortunately, this effect is not readily observed. Although, the matrices across number of hidden states do appear to vary less. The identified feedthrough matrix follows.

Table 5.11: Comparison of estimated A and feed-through matrices for Zak's mutual repression motif. Rows and columns denote, in order, $\{M_D, M_C\}$.

Discrete		Continuous (10^{-2})	
5 hidden states			
0.43	0.04	-1.72	0.81
0.06	0.40	0.06	-1.22
10 hidden states			
0.44	0.04	-1.64	1.25
0.06	0.39	0.06	-0.93
15 hidden states			
0.44	0.04	-1.70	1.54
0.05	0.40	-0.01	-0.72
20 hidden states			
0.39	0.08	-2.12	1.33
0.02	0.45	0.31	-1.68

5.3.3 Motif 3: Auto-Activation and Sequestration

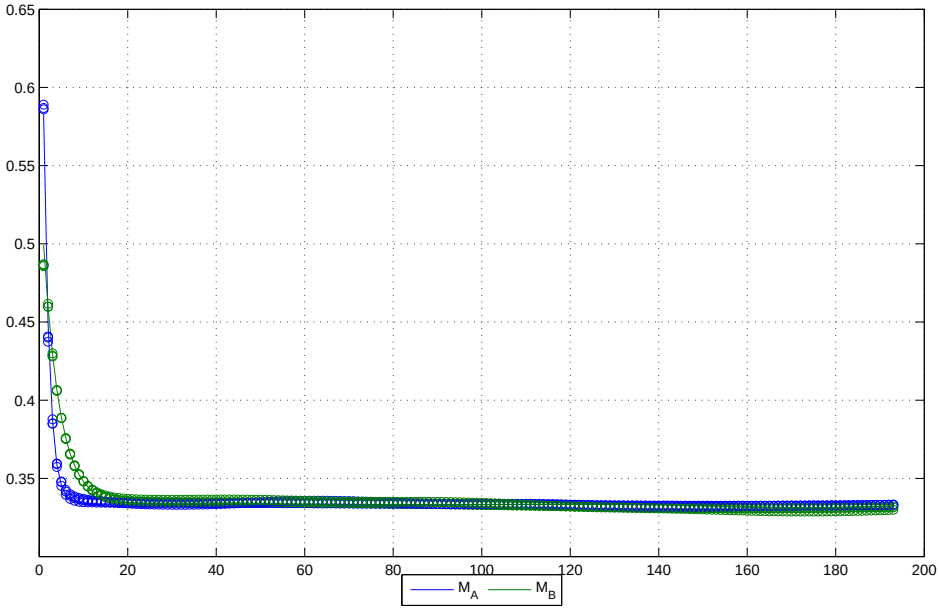
The simulation and training process follows that outlined in the previous sections. The relevant transcripts were

Table 5.12: Relevant transcriptional parameters to the auto-activation and sequestration motif [33].

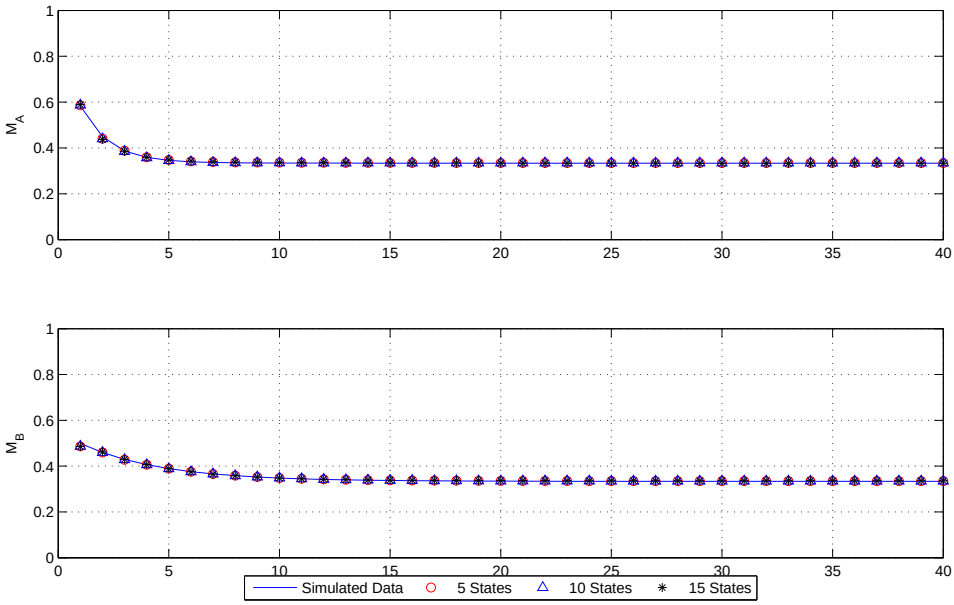
Gene A	
$k_{RP_{AA}}$	7.3×10^{-4}
$k_{RAP_{AA}}$	7.3×10^{-4}
Gene B	
$k_{RP_{BAF}}$	3.6×10^{-4}
$k_{RAP_{BAF}}$	3.6×10^{-4}
$k_{RF_2P_{BAF}}$	3.6×10^{-1}
$k_{RAF_2P_{BAF}}$	7.2×10^{-1}

All other transcripts were suppressed. Thirty data sets with randomly chosen initial conditions were used to train the estimated model. Twenty sets were saved for prediction analysis. The data was sampled at uneven

time steps in pre-processing and normalized. Estimation and prediction plots follow

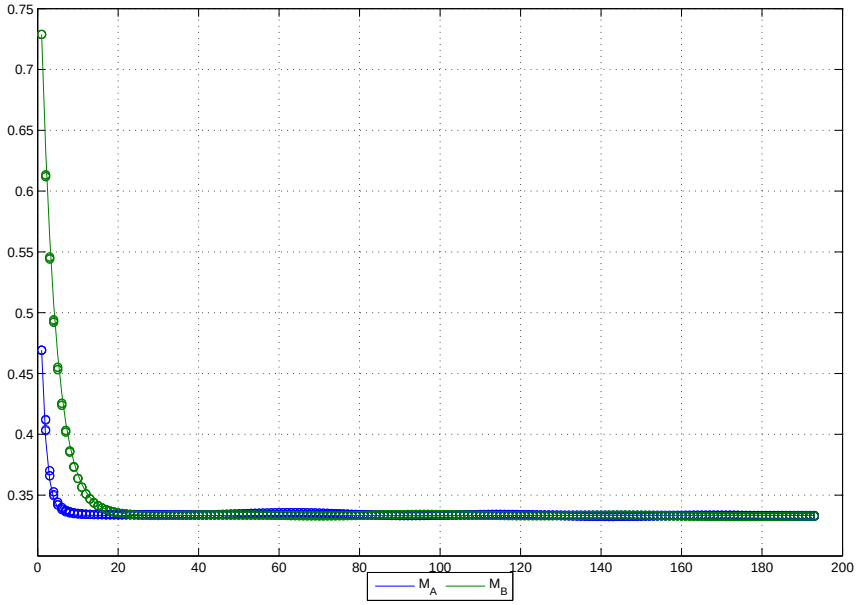


(a) Training data set fit.

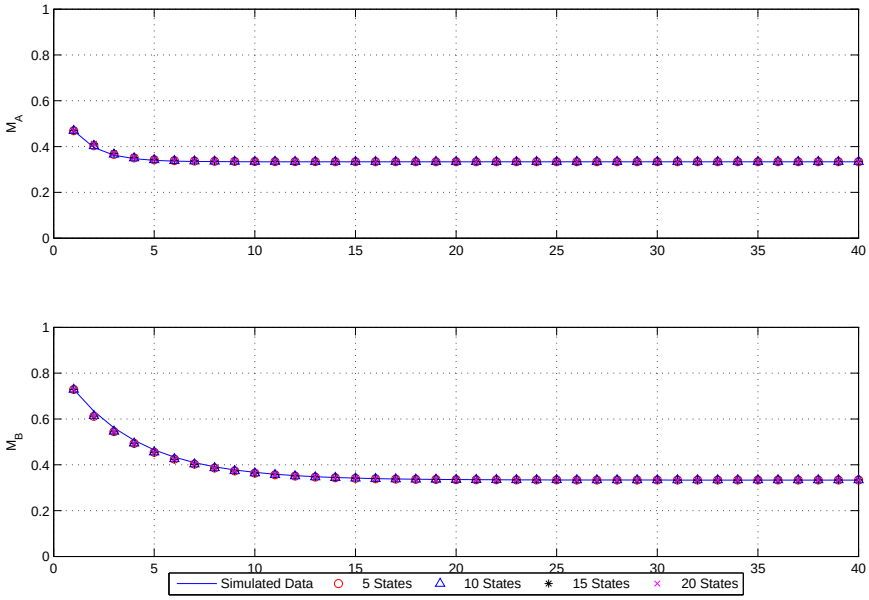


(b) Zoomed in training data set fit.

Figure 5.11: Comparison of identified model estimation and prediction for auto-activation motif.



(c) Prediction of new data set.



(d) Zoomed in prediction of new data set.

Figure 5.11: Comparison of identified model estimation and prediction for auto-activation motif.

Table 5.13: Comparison of estimated A and feed-through matrices for Zak's auto-activation and sequestration motif. Rows and columns denote, in order, $\{M_A, M_B\}$.

Discrete		Continuous (10^{-2})	
5 hidden states			
0.44	0.02	-1.61	0.37
0.05	0.64	0.54	-1.73
10 hidden states			
0.44	0.02	-1.71	0.21
0.05	0.66	0.51	-1.49
15 hidden states			
0.42	0.05	-2.22	1.21
0.04	0.64	0.51	-2.03
20 hidden states			
0.41	0.05	-2.31	1.29
0.04	0.64	0.46	-1.97

Like the previous motif, this motif has only two observed states, $\{M_A, M_B\}$, in order. As there are fewer dynamics to fit simultaneously, the training estimates and predictions are almost identically accurate in our plots. In this motif, we should observe auto-activation and sequestration from M_A on itself and M_B, respectively, so M_A should have a positive influence on itself and M_B should have a noticeable positive coefficient with respect to M_A in its coefficient row. We observe the first case in all our estimated feedthrough matrices; however, as this has been a trend in all our estimated feedthrough matrices when it was not expected, it is not a significant gain. The second case is not particularly obvious in our estimates because the coefficients on the diagonal dominate. Knowing the state transform or using a state transform to ease interpretation of the feedthrough matrix may make the state-to-state connections more obvious but in this instance, perhaps more coefficients are necessary to make connections more distinctive.

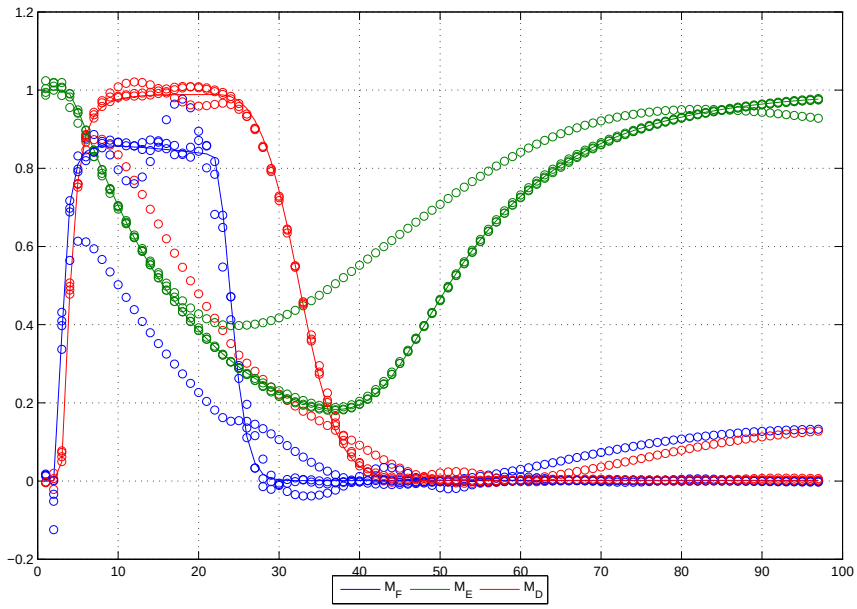
5.3.4 Motif 4: Agonist-Induced Receptor Down-Regulation

Like in the previous sections, we suppress the non-valid transcriptionals factors and keep the valid ones, which are

Table 5.14: Relevant transcriptional parameters to the agonist-induced receptor down-regulation motif [33].

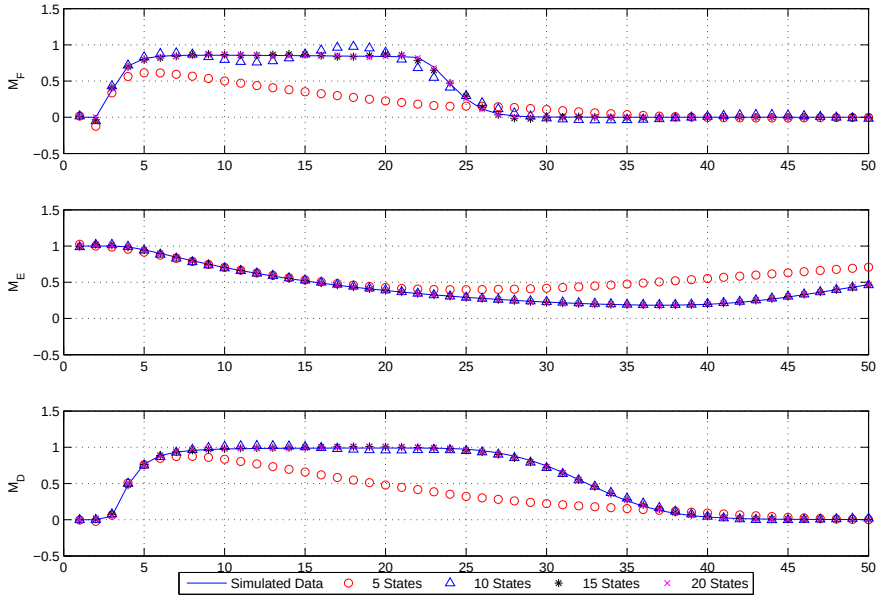
Gene D	
$k_{RP_{DCF}}$	7.3×10^{-4}
$k_{RC_2P_{DCF}}$	0
$k_{RF_2P_{DCF}}$	7.3×10^{-1}
$k_{RC_2F_2P_{DCF}}$	7.3×10^{-1}
Gene E	
$k_{RP_{ED}}$	4.2×10^{-3}
$k_{RD_2P_{ED}}$	4.2×10^{-7}
Gene F	
$k_{RP_{FEQ}}$	7.3×10^{-4}
$k_{REQP_{FEQ}}$	7.3×10^{-1}

Unlike the previous motifs, the dynamics in this motif were triggered by the stimulation of a single input state rather than random initial conditions. This input state varied from data set to data set in terms of a randomly chosen pulse width and magnitude. The length of the pulse lengthened the non-steady-state dynamics of the system and allowed us to use longer, even time steps during data simulation. Again, fifty sets of data were generated and normalized; thirty were used to train our state-space model and twenty were reserved to observe prediction ability. Plots of estimation and prediction follow.

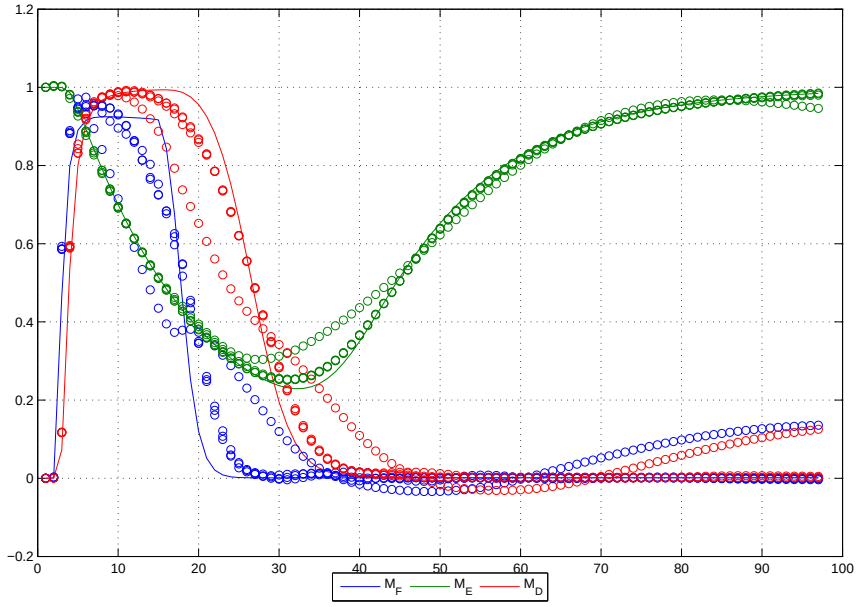


(a) Training data set fit.

Figure 5.12: Comparison of identified model estimation and prediction for receptor motif.

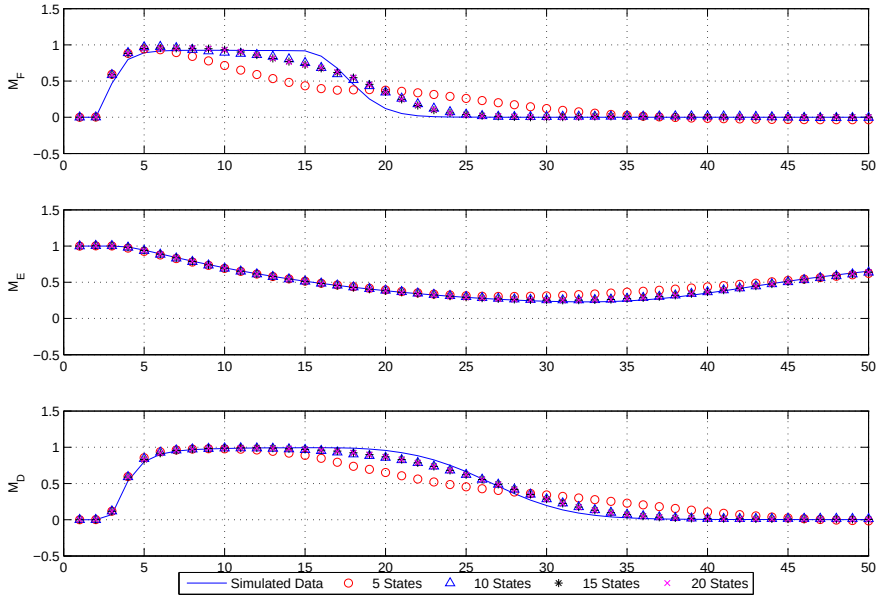


(b) Zoomed in training data set fit.



(c) Prediction of new data set.

Figure 5.12: Comparison of identified model estimation and prediction for receptor motif.



(d) Zoomed in prediction of new data set.

Figure 5.12: Comparison of identified model estimation and prediction for receptor motif.

Table 5.15: Comparison of estimated A and feed-through matrices for Zak's receptor motif. Rows and columns denote, in order, $\{M_F, M_E, M_D\}$.

Discrete			Continuous (10^{-2})		
5 hidden states					
0.86	0.17	-0.15	-0.00	0.23	-0.26
-0.02	0.95	-0.03	-0.03	-0.08	-0.05
0.07	-0.03	0.84	0.08	0.35	-0.39
10 hidden states					
0.74	0.18	-0.11	-0.28	0.46	-0.22
-0.02	0.93	-0.03	-0.03	-0.11	-0.08
0.05	0.05	0.77	0.23	0.37	-0.76
15 hidden states					
0.63	0.28	-0.05	-0.59	0.73	-0.12
-0.02	0.92	-0.03	-0.04	-0.13	-0.07
0.04	0.10	0.77	0.84	0.31	-0.99
20 hidden states					
0.60	0.24	-0.05	-0.76	0.53	-0.06
-0.02	0.92	-0.04	-0.05	-0.15	-0.08
0.03	0.07	0.77	0.44	-0.06	-0.70

In this motif, we observe significantly better training estimates than predictions. Further, the training estimation quality varies with the number of hidden states used. The estimates from the model with five

hidden states fare the worst and cannot track the dynamics of the motif at all. The model with ten hidden states fares significantly better but its estimates still deviate more from the actual observations than we have ever observed before in our previous motifs. The models with fifteen and twenty hidden states achieve the estimation quality that has become the norm in our previous simulation motifs. In general, the dynamics of M_F appear to be the hardest to track and is probably the most nonlinear; M_D appears the second hardest and M_E the easiest.

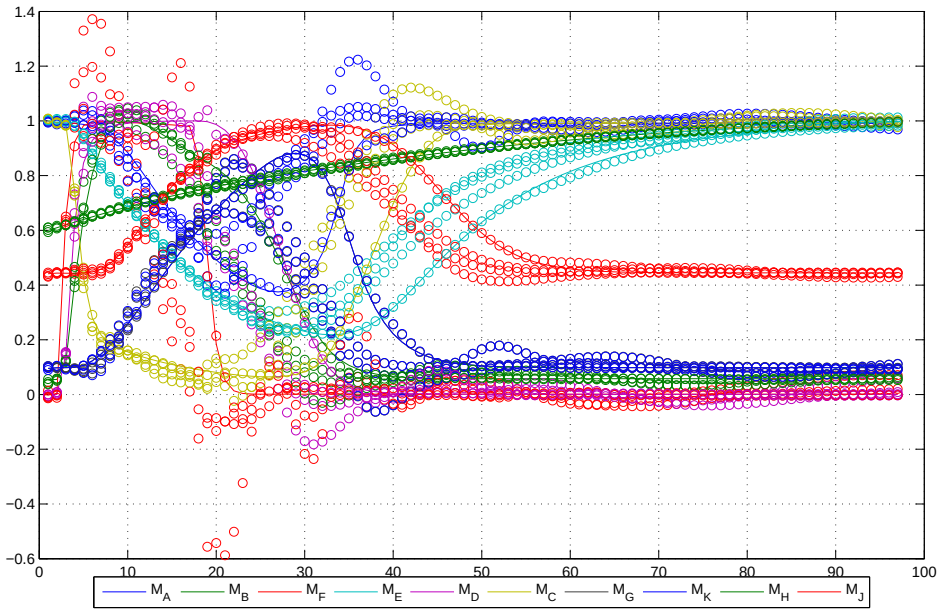
The model predictions deteriorate a great deal from the training estimations. Only one prediction is shown above as an average of the prediction performance we have observed. There are data sets that produce better predictions; these data sets are probably close approximations of a training data set. But there are also data sets that produce worse predictions; these probably result from data sets that vary more from those that the models were trained on. Yet, the average prediction is sufficient to show that none of the models are able to reproduce the actual dynamics exactly. The predictions of state M_F produces the most tracking error, confirming that M_F is probably the most nonlinear and hence the most likely to vary from its observed dynamics. State M_D produces similar prediction errors but is still easier to model than M_D. Finally, state M_E appears to be accurately modeled by the ten to twenty hidden state models. In general, in the instances of poor prediction, the dynamics appeared too sharp and fast for the model could approximate. But as training and prediction appeared to only improve with increasing numbers of hidden states, more hidden states would probably eventually produce an estimation and prediction quality akin to what we have observed in previous motifs.

In this motif, we also observe cyclic down-regulation. The input state triggers M_F by combining with M_E. M_F induces more M_D production, which in turn suppresses M_E production. As M_E was necessary in the first place to produce M_F, the M_F concentration ultimately decreases. In terms of the feedthrough matrix, we hope to observe significant positive coefficients corresponding to M_F's influence on M_D and M_E's influence on M_F as well as a distinctive negative coefficient corresponding to M_D's influence on M_E. There may also be some cross-state interference in which M_D appears to suppress M_F because it is suppressing M_E, the factor that induces it, and M_F appears to suppress M_E as well because it produces more M_D. Surprisingly, we observe most of these sign changes in the feedthrough matrix. In M_F's coefficient row, we see a negative coefficient corresponding to M_D's influence across all the hidden state models, both continuous and discrete.

Likewise, in M_E's coefficient row, we see negative coefficients corresponding to the suppression we predicted. Finally, in M_D's coefficient row, we occasionally observe a negative influence from M_E but, as it varies from model to model, this connection does not seem very dependable. In general, it was surprising that these connections were so distinctive in the feedthrough matrix despite the poor estimation and prediction we observed in our plots.

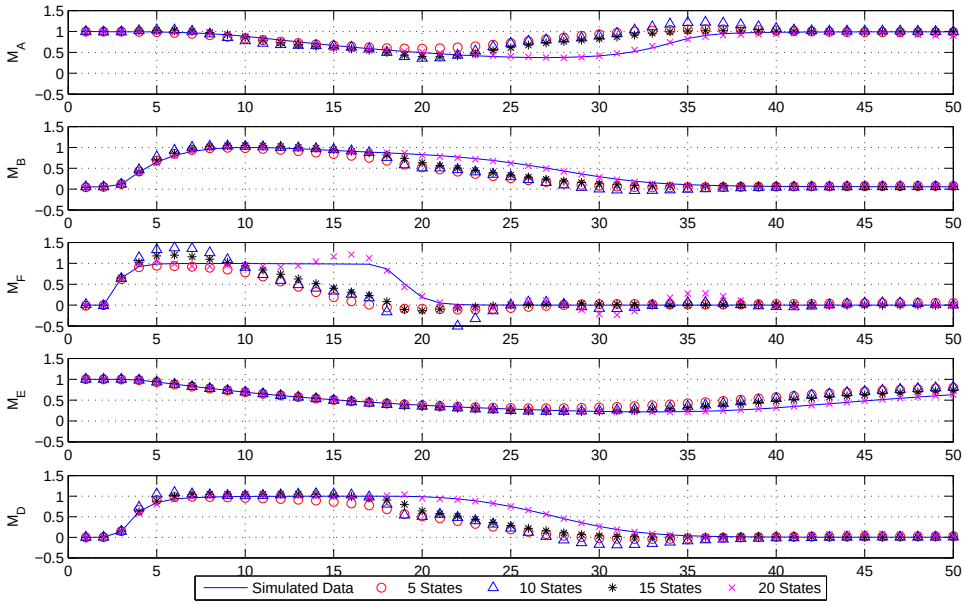
5.3.5 The Entire Model

Finally, all four motifs were combined and simulated to generate data as input to the EM algorithm. The combined simulation model is dependent on the stimulation of a single input, like the fourth motif. As this type of input triggered lengthier state dynamics, the same data steps were used throughout sampling. The data was normalized before being input to the algorithm. Thirty data sets were used to train the model; twenty were used to test it. First, we observe the estimation and prediction results.

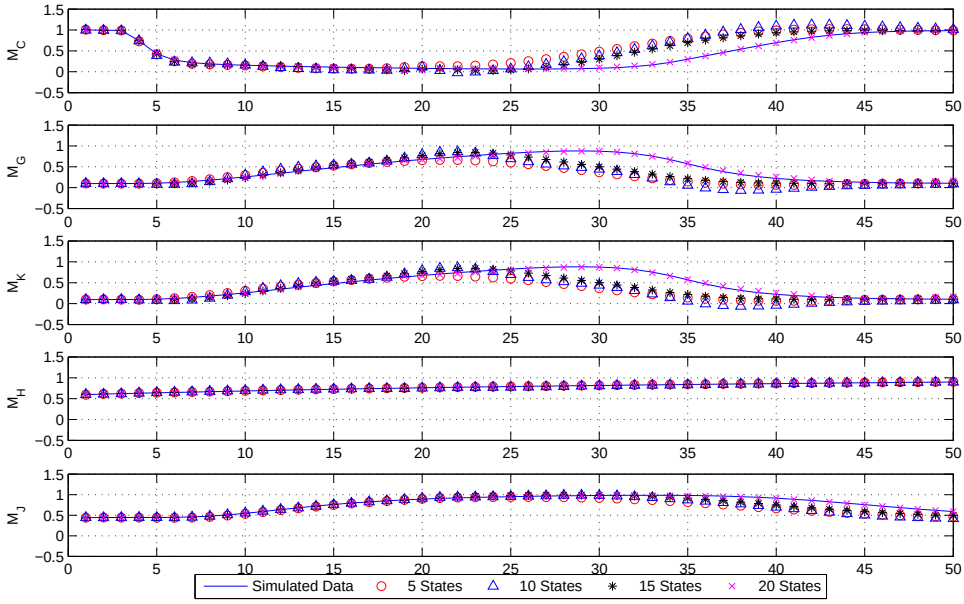


(a) Training data set fit.

Figure 5.13: Comparison of identified model estimation and prediction for entire model.

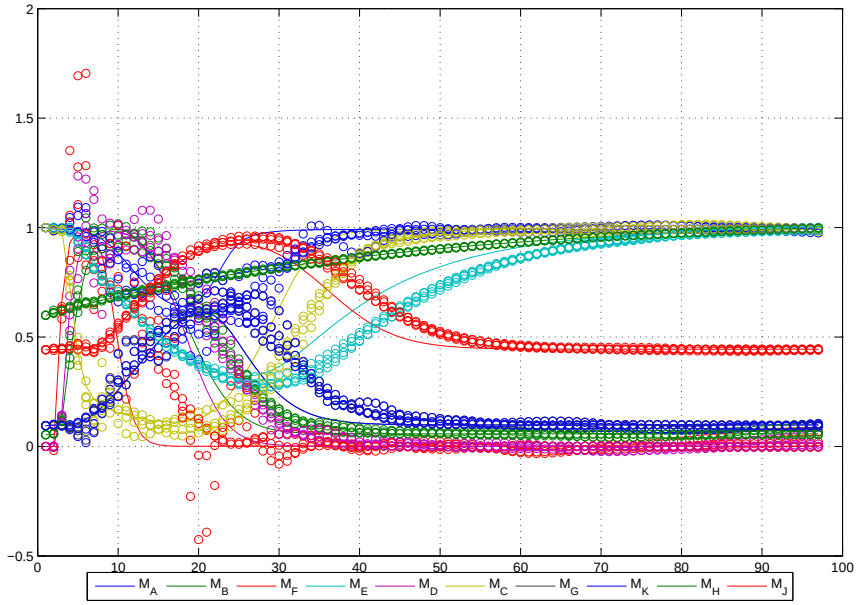


(b) Zoomed in training data set fit.

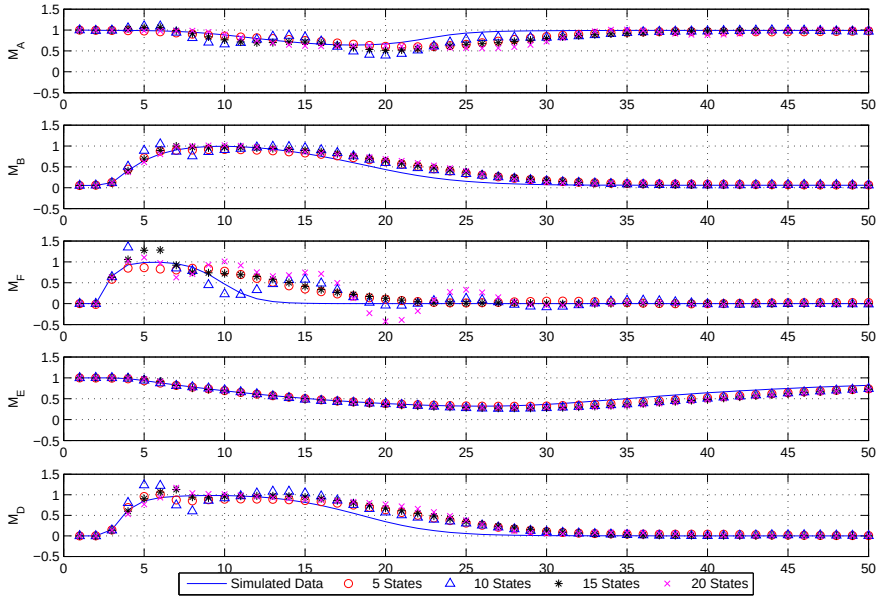


(c) Zoomed in training data set fit.

Figure 5.13: Comparison of identified model estimation and prediction for entire model.

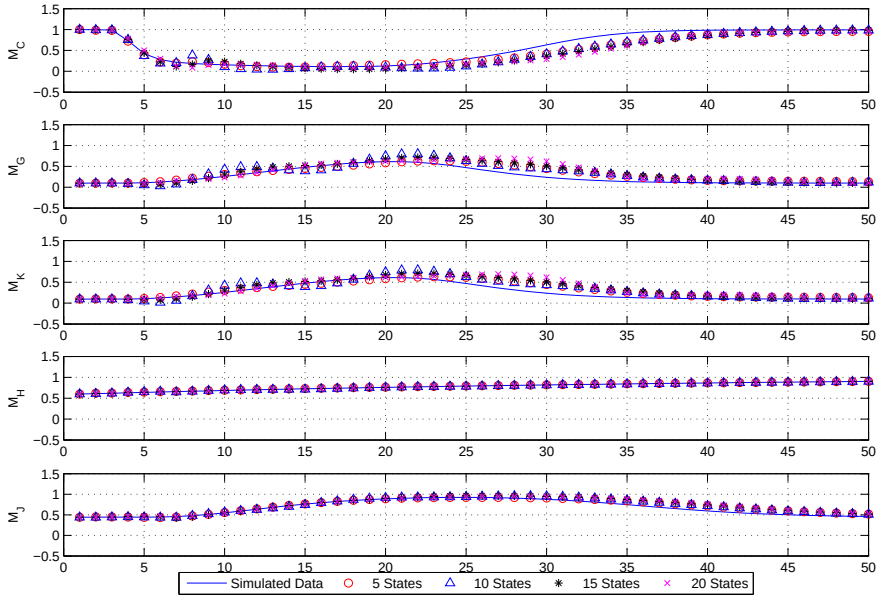


(d) Prediction of new data set.



(e) Zoomed in prediction of new data set.

Figure 5.13: Comparison of identified model estimation and prediction for entire model.



(f) Zoomed in prediction of new data set.

Figure 5.13: Comparison of identified model estimation and prediction for entire model.

The combined system contains the dynamics of 44 different equations, which all interconnect somehow through promotion or suppression. Despite the enormous complexity the simulation model encompasses and the limited amount of dynamics we observe, our model still tracks its observed dynamics fairly well. In the training estimations, the model with twenty hidden states reproduces the observations almost exactly. All other models perform worse; specifically, they track appropriate changes at erroneous times. We observe the most estimation error in the M_F case, which is not unexpected as it was one of the hardest states to track even when the system was broken up into smaller motifs. Unfortunately, state M_F is the first state affected by our input state, so errors in its estimation most likely propagate to all other states' estimations. In our training estimation plots, the poor estimates of state M_F drop off a lot earlier than the actual observations. In terms of the system dynamics, this means the input stimulating the system is dying out early because state M_F is initial state that triggers the all other motifs. The early decrease in M_F could very likely causes the early dynamics we observe in all the other states. The fact that the best estimation case, with 20 hidden states, nails M_F's dynamics as well as all the other states' dynamics supports our argument. So although, the other states are tracking and modeling accurately, the error in M_F estimation is throwing off all the other estimates.

Our model predicts a lot worse than it re-estimates. Further, in this prediction instance, the models with higher numbers of hidden states actually predict worse than models with fewer hidden states. This suggests that the more complicated models were over-trained to our training data sets such that they began modeling the exact noise dynamics on top of the model dynamics as well as the actual model dynamics. These noise dynamics are never replicated because they are generated from random white noise distributions hence the portion of the model trained onto these dynamics are virtually useless and possibly detrimental to the overall prediction performance. On another note, we observe a delay in reproducing the dynamics of state M_F in this prediction data set. Consequently, we also observe propagating delays in almost all other states, which are triggered by it. The closer a transcript state is to M_F in the connection network, the greater its prediction delay. This behavior further confirms our hypothesis that the other states may be modeled accurately but perform poorly because of the error in modeling M_F. Ironically, the twenty hidden state model, which performed the best in the training simulation, performs the worst in the prediction case with the worst time delay. As a result, its other state predictions lag more than all the other models' predictions as well.

In terms of feedthrough matrix coefficients, we would hope to observe a combination of all the influences described in the previous sections. However, comparing the parameter estimates across all the models, there is too much fluctuation in the parameters to report positively whether or not a solid connection exists. A state transformation could potentially bring out these connections. We know they exist because almost all the dynamics were modeled accurately by our identified model,except for some time delay, and all of our plots of estimation and prediction prove this fact. However, we are experiencing tremendous difficulty identifying the relevant network connections in the feedthrough matrix. We could use a state transformation to increase the feedthrough dynamics as best as we can but even then, we would only have an approximate linearization of the model's state-to-state dynamics at best. Furthermore, if the natural system is extremely nonlinear, as it is in this Zak's model, linearization may not provide the distinction we hope to see. Hence we should consider other methods of identifying connections from our identified model. Instead of trying to discern influences by scrutinizing the feedthrough matrix, we could try simulating selective initial conditions on our model and observing the resulting dynamics analytically or knocking out entire observation states and observing the missing the dynamics.

Table 5.16: Comparison of discrete feed-through matrices for Zak's entire model. Rows and columns denote, in order, $\{M_A, M_B, M_F, M_E, M_D, M_C, M_G, M_K, M_H, M_J\}$.

Discrete									
CB+D (5 hidden states)									
1.09	-0.14	-0.00	-0.07	0.01	-0.10	1.66	-1.55	-0.02	-0.17
0.20	0.46	0.11	0.08	0.40	0.04	-2.49	2.67	-0.20	0.02
0.31	0.30	0.96	0.36	-0.17	0.36	8.03	-7.51	-0.40	0.41
-0.08	0.04	-0.00	0.93	-0.06	0.04	0.26	-0.36	0.02	0.01
0.50	0.04	0.19	0.25	1.03	0.43	-3.13	3.87	-0.31	0.30
-0.45	0.39	-0.08	-0.04	-0.45	0.73	3.65	-4.36	0.22	0.15
-0.34	0.13	-0.00	0.02	-0.07	-0.00	-0.14	0.84	-0.01	0.03
-0.30	0.14	-0.00	0.02	-0.07	0.01	-0.91	1.65	-0.01	0.03
-0.03	0.00	0.00	0.00	-0.02	-0.01	-0.07	0.03	0.96	0.00
0.28	0.19	0.00	0.07	-0.06	0.10	-1.38	1.74	0.01	0.98
CB+D (10 hidden states)									
0.97	-0.19	0.00	-0.12	0.01	-0.12	0.66	-0.65	-0.10	-0.21
-0.05	0.44	0.10	0.05	0.36	-0.06	-0.16	0.03	-0.22	0.01
-0.19	-0.08	1.00	0.14	-0.14	0.02	-0.15	0.11	-0.66	0.03
-0.02	0.03	-0.01	0.90	-0.03	0.06	-0.03	0.01	0.00	-0.02
0.04	-0.09	0.16	0.20	1.01	0.20	-0.11	0.27	-0.35	0.24
-0.26	0.34	-0.06	-0.26	-0.43	0.76	-0.17	-0.26	0.08	-0.18
-0.25	0.16	-0.00	0.10	-0.06	0.03	0.31	0.47	0.06	0.13
-0.22	0.15	-0.01	0.10	-0.04	0.03	0.39	0.41	0.06	0.12
0.01	-0.01	-0.00	-0.01	0.01	-0.00	-0.01	0.02	0.97	-0.02
0.06	0.12	-0.00	0.06	-0.07	-0.00	0.03	0.08	0.02	0.97
CB+D (15 hidden states)									
1.02	-0.19	0.02	-0.11	0.02	-0.10	0.01	0.06	-0.21	-0.12
0.11	0.48	0.11	-0.05	0.40	0.11	0.13	0.00	-0.15	0.02
-0.65	0.31	0.99	0.54	-0.66	-0.17	-0.11	-0.34	-0.20	0.19
-0.15	-0.09	-0.01	0.59	0.01	0.03	-0.06	-0.11	0.02	-0.30
0.15	0.05	0.18	0.35	0.93	0.32	0.26	0.08	-0.14	0.48
-0.33	0.33	-0.13	-0.10	-0.53	0.53	-0.51	-0.22	0.25	-0.17
-0.27	0.13	-0.01	0.02	-0.04	0.03	0.32	0.45	0.06	-0.01
-0.29	0.12	-0.01	-0.01	-0.03	0.03	0.34	0.41	0.08	-0.02
-0.01	-0.04	0.00	-0.10	0.02	-0.00	-0.03	0.01	0.11	-0.06
0.02	-0.02	-0.01	-0.36	0.03	0.02	0.02	0.04	0.04	0.63
CB+D (20 hidden states)									
0.85	-0.14	0.03	-0.15	-0.04	-0.13	0.46	-0.54	-0.24	-0.06
0.18	0.60	0.11	-0.19	0.36	0.21	0.12	0.12	-0.16	0.06
-0.65	0.11	0.84	0.27	-0.43	-0.20	0.02	-0.64	-0.17	0.11
-0.04	-0.10	-0.01	0.50	0.02	0.06	-0.08	0.01	-0.02	-0.35
0.23	0.13	0.18	-0.00	0.92	0.45	-0.02	0.45	0.05	0.43
-0.39	0.18	-0.11	-0.12	-0.53	0.41	-0.85	-0.00	0.33	-0.26
-0.29	0.11	-0.01	0.07	-0.03	0.02	-0.02	0.76	0.18	0.03
-0.30	0.11	-0.01	0.07	-0.03	0.02	-0.02	0.75	0.20	0.04
-0.05	-0.05	0.00	-0.10	0.03	-0.01	-0.04	-0.01	0.10	-0.07
0.08	0.02	-0.01	-0.31	0.00	0.05	0.14	-0.03	0.05	0.70

Table 5.17: Comparison of continuous feed-through matrices for Zak's entire model. Rows and columns denote, in order, $\{M_A, M_B, M_F, M_E, M_D, M_C, M_G, M_K, M_H, M_J\}$.

Continuous									
CB+D (5 hidden states)									
0.30	-0.28	0.00	-0.14	0.10	-0.14	3.47	-3.18	0.10	-0.30
0.14	-0.67	0.03	0.39	0.36	0.18	-1.70	1.85	-1.85	0.23
0.17	0.03	0.30	0.36	-0.10	0.34	8.59	-8.01	0.51	0.52
-0.13	0.18	-0.02	-0.05	-0.19	0.16	0.74	-0.82	-0.27	0.09
0.92	1.38	-0.06	1.41	-1.13	1.67	0.74	1.13	-5.11	1.42
-1.07	2.03	-0.22	0.58	-1.94	0.16	10.06	-11.06	-1.53	1.05
-0.58	0.17	0.01	-0.01	-0.09	-0.05	-2.11	1.62	0.16	-0.02
-0.51	0.16	0.02	-0.03	-0.06	-0.03	-1.95	1.52	0.21	-0.04
-0.05	0.14	-0.03	0.08	-0.13	0.06	0.47	-0.48	-0.38	0.08
0.57	0.26	0.01	0.09	-0.02	0.17	-3.40	4.10	0.16	-0.05
CB+D (10 hidden states)									
-0.23	0.32	0.06	0.68	-0.58	-0.15	12.30	-12.40	-0.19	0.75
-1.01	-0.09	0.27	1.16	-0.41	-0.26	-2.36	1.31	-0.67	1.32
2.32	-4.09	-0.27	-3.95	3.85	0.08	0.24	1.71	-0.02	-5.10
0.03	-0.05	-0.02	-0.25	0.05	0.12	-0.45	0.49	-0.01	-0.16
-3.05	3.30	0.74	4.14	-3.92	0.10	-9.01	6.86	-1.82	5.05
-2.74	3.00	0.14	1.06	-3.58	-0.55	-10.31	7.92	-1.41	1.42
-0.38	0.01	-0.02	-0.33	0.11	0.02	-8.83	8.52	0.03	-0.42
-0.24	0.34	-0.01	0.66	-0.07	0.11	13.14	-13.34	0.46	0.86
0.16	-0.22	-0.04	-0.25	0.24	0.00	-0.28	0.40	-0.03	-0.32
0.27	0.02	-0.03	-0.03	0.11	0.01	1.46	-1.16	0.12	-0.19
CB+D (15 hidden states)									
0.36	-0.37	-0.09	-0.79	0.11	-0.03	1.31	-0.99	1.12	-0.91
-0.41	-1.23	0.02	-1.38	0.81	0.11	2.83	-3.47	5.67	-1.25
3.33	0.29	0.42	3.81	0.56	0.99	5.56	-1.20	-7.30	2.79
-0.20	-0.11	-0.18	-1.13	-0.02	0.18	3.05	-3.39	5.55	-1.17
-1.55	0.57	0.14	-0.85	-1.03	0.48	0.63	-2.18	8.50	-0.32
0.42	2.51	-0.36	2.05	-2.33	-0.37	-3.76	3.49	5.17	1.15
0.15	0.93	0.09	1.40	-0.53	0.11	-12.45	12.76	-4.39	1.16
-0.64	-0.14	-0.10	-0.53	0.25	0.07	8.86	-9.44	4.42	-0.60
0.26	-0.42	-0.03	-1.04	0.33	0.09	-0.63	0.84	-6.05	-0.82
0.40	-0.20	-0.16	-0.91	0.31	0.20	2.69	-2.28	4.89	-1.16
CB+D (20 hidden states)									
-0.90	-0.78	0.16	-1.47	0.24	-0.27	14.66	-15.39	-0.26	-0.64
1.06	-1.12	0.02	-0.79	1.14	0.49	2.72	-1.88	0.15	-0.76
-0.97	0.63	0.18	0.23	-0.88	0.34	-3.95	3.77	-1.36	0.70
0.56	-0.21	-0.12	-1.54	0.18	0.41	-0.79	1.27	1.58	-1.33
2.48	0.04	-0.19	2.08	0.34	0.91	2.95	-1.01	0.32	0.54
1.66	1.75	-0.84	2.82	-2.21	-0.93	-22.02	22.04	-0.84	-0.44
-0.32	0.22	0.14	-0.59	0.04	0.29	-19.70	19.78	-3.78	-0.25
-0.66	0.30	-0.05	0.81	-0.22	-0.15	0.85	-1.54	2.16	0.46
-1.18	-0.99	0.50	-3.99	0.86	0.44	-2.70	2.35	-9.55	-1.62
0.38	0.03	-0.12	-0.56	0.03	0.13	4.02	-3.71	3.82	-0.67

5.4 Unraveling the State Transformation Problem

In this thesis, we will only consider the simple case described in section 4.3.3 in which the number of hidden states is equal to the number of gene states. But we will apply this solution to two identified systems, Zak's motif 1 and Zak's entire model. We begin with the simpler of the two, the cascade motif.

5.4.1 Interpreting Motif 1: Cascade

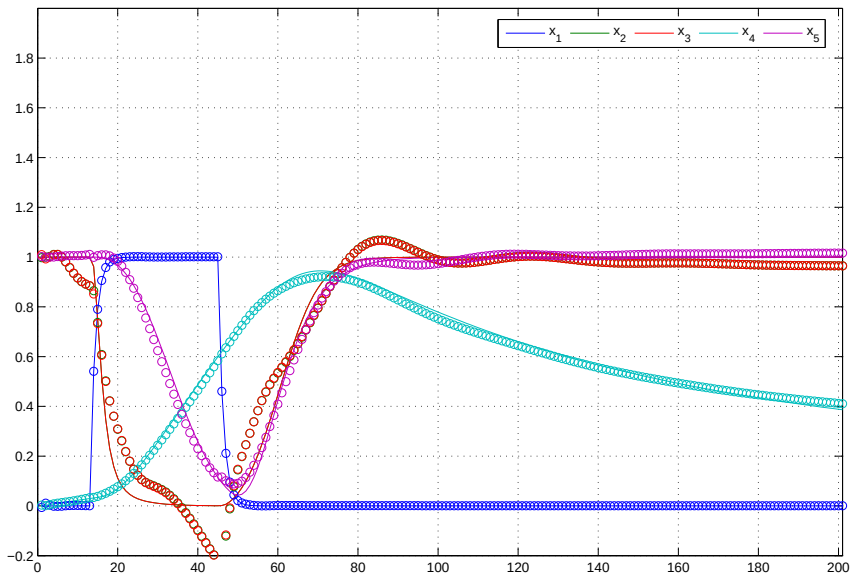
Before applying the solution described in section 4.3.3, we re-simulated the cascade system with an artificial input stimulus to gene transcript C and trained an eleven state model to the generated data - six states were hidden, with one being the bias state, and five were observed. We added the artificial input so that more dynamics and causal relationships were observable; in our previous simulations, the dynamics of genes C, G, and K due to initial conditions had been to fast to distinguish causal influences. The addition of the artificial input vector and input parameter matrix expanded our solution model in Eq. (4.32) into

$$\begin{aligned}
\begin{bmatrix} g_k \\ g_{k+1} \end{bmatrix} &= \begin{bmatrix} 0 & I \\ A_3(A_2 - A_1A_3^{-1}A_4) & A_3A_1A_3^{-1} + A_4 \end{bmatrix} \begin{bmatrix} g_{k-1} \\ g_k \end{bmatrix} + \\
&\begin{bmatrix} 0 & 0 \\ A_3(B_1 - A_1A_3^{-1}B_2) & B_2 \end{bmatrix} \begin{bmatrix} u_{k-1} \\ u_k \end{bmatrix} + \begin{bmatrix} 0 \\ A_3b_x - A_3A_1A_3^{-1}b_g + b_g \end{bmatrix} \\
y_k &= \begin{bmatrix} 0 & I \end{bmatrix} \begin{bmatrix} g_{k-1} \\ g_k \end{bmatrix}
\end{aligned} \tag{5.4}$$

Denote the bottom two partitions of A as $\tilde{A}_3$ and $\tilde{A}_4$ from left to right. Denote the bottom two partition of B as $\tilde{B}_3$ and $\tilde{B}_4$ as well. The identified state parameters, A and B, and sample plots of its estimation ability follow.

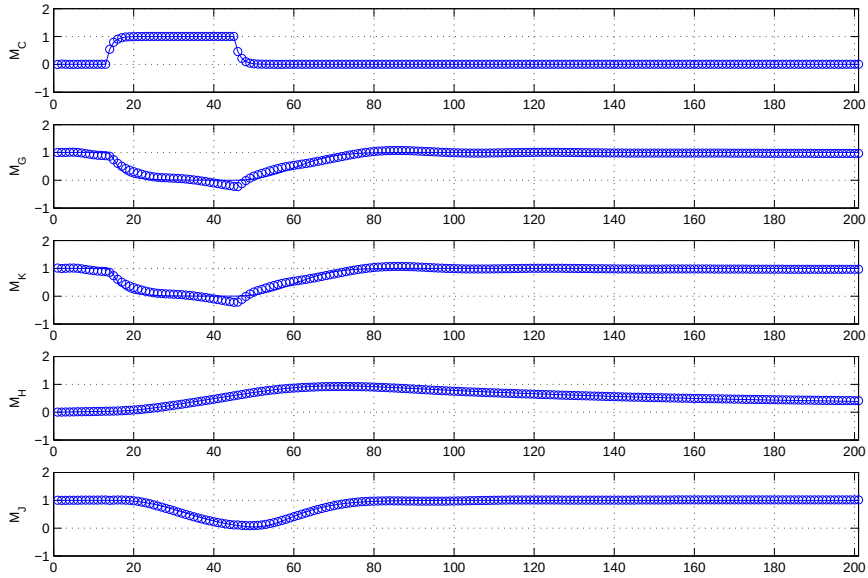
Table 5.18: Identified A matrix for Zak's cascade motif.

A											B
1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
0.06	0.17	0.36	-0.52	-0.51	0.04	0.38	-1.27	1.17	-0.03	0.06	-0.89
0.03	0.71	0.50	-0.22	0.23	0.00	0.10	-3.78	3.28	0.09	0.40	-0.52
-0.09	0.40	-0.10	0.60	0.15	-0.04	0.20	0.50	-0.62	0.14	0.12	-0.37
0.09	0.02	-0.03	-0.01	0.88	-0.02	-1.00	-0.75	1.03	-0.03	-0.35	1.16
1.18	-3.43	2.20	1.80	0.08	0.83	-2.58	25.05	-26.19	-0.26	0.07	1.29
0.00	-0.08	0.05	-0.04	-0.04	0.00	0.44	-0.21	0.23	-0.00	-0.03	0.54
0.04	-0.02	0.00	-0.11	-0.12	0.01	-0.04	0.11	0.84	0.03	-0.01	-0.02
0.04	-0.00	-0.00	-0.10	-0.10	0.00	-0.03	0.18	0.76	0.03	0.00	-0.03
0.03	0.06	-0.03	0.05	0.03	0.00	0.03	-0.01	-0.02	0.99	0.00	-0.00
0.02	-0.04	0.06	-0.07	0.01	0.00	-0.05	0.18	-0.16	0.02	0.96	-0.01

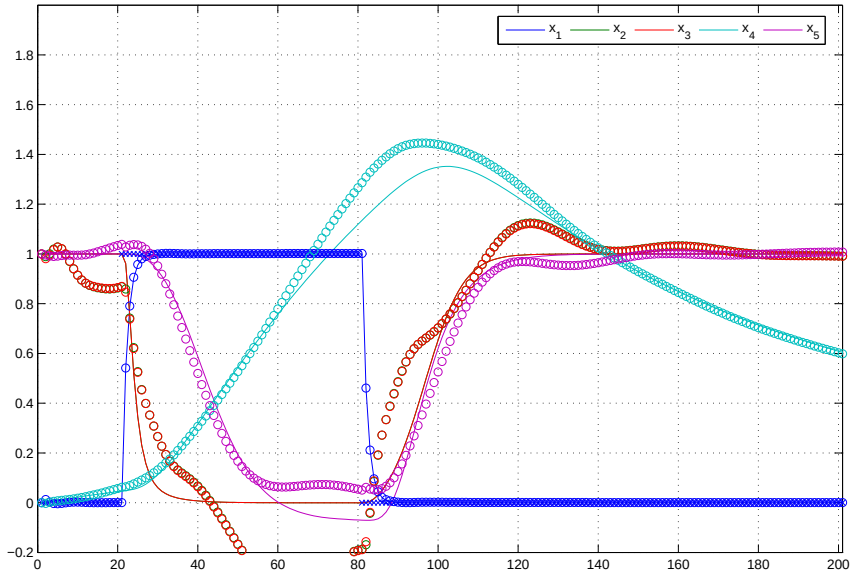


(a) Training data set fit.

Figure 5.14: Identified model estimation and prediction performance for artificially stimulated cascade model.

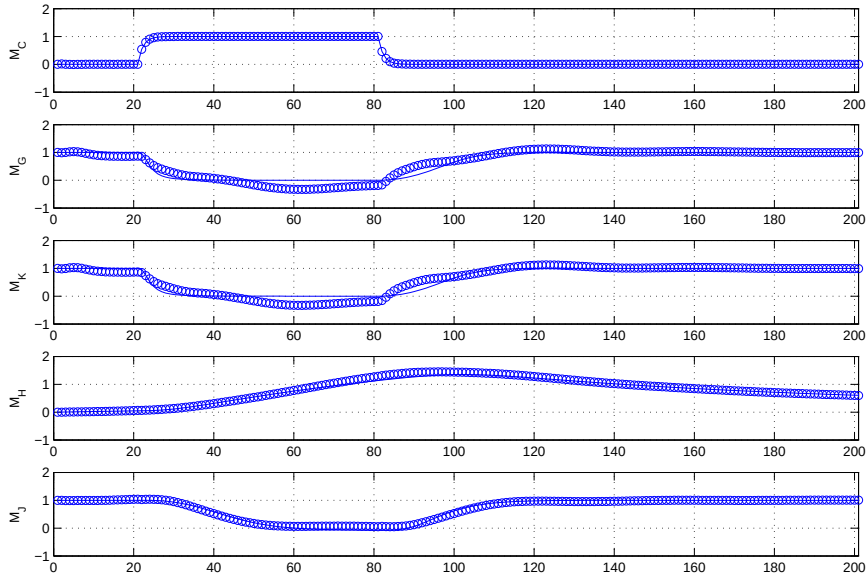


(b) Zoomed in training data set fit.



(c) Prediction of new data set.

Figure 5.14: Identified model estimation and prediction performance for artificially stimulated cascade model.



(d) Zoomed in prediction of new data set.

Figure 5.14: Identified model estimation and prediction performance for artificially stimulated cascade model.

For the transformed state space described in (4.32), we calculate $\tilde{A}_3$ and $\tilde{A}_4$. We tabulate these two matrices below for both the continuous and discrete case.

Table 5.19: Transformed matrices $\tilde{A}_3$ and $\tilde{A}_4$.

$\tilde{A}_3$					$\tilde{A}_4$					$\tilde{B}_3$	$\tilde{B}_4$
-0.10	-0.04	0.06	-0.14	0.07	0.67	-0.74	0.73	0.14	-0.07	-0.10	0.54
-0.24	0.59	-1.08	-1.84	-0.26	0.71	5.00	-3.59	1.87	0.36	-0.53	-0.02
-0.18	0.45	-0.97	-1.60	-0.31	0.58	4.45	-3.02	1.62	0.41	-0.44	-0.03
-0.23	0.15	-0.18	-0.66	-0.01	0.50	-0.09	0.12	1.66	0.01	-0.26	-0.00
0.41	-0.46	0.52	0.10	-0.93	-0.94	1.11	-1.18	-0.10	1.93	0.52	-0.01

The state vector change is still a state transformation, hence we expect the input-output behavior of this redefined state space system to remain identical to the identified system's behavior.

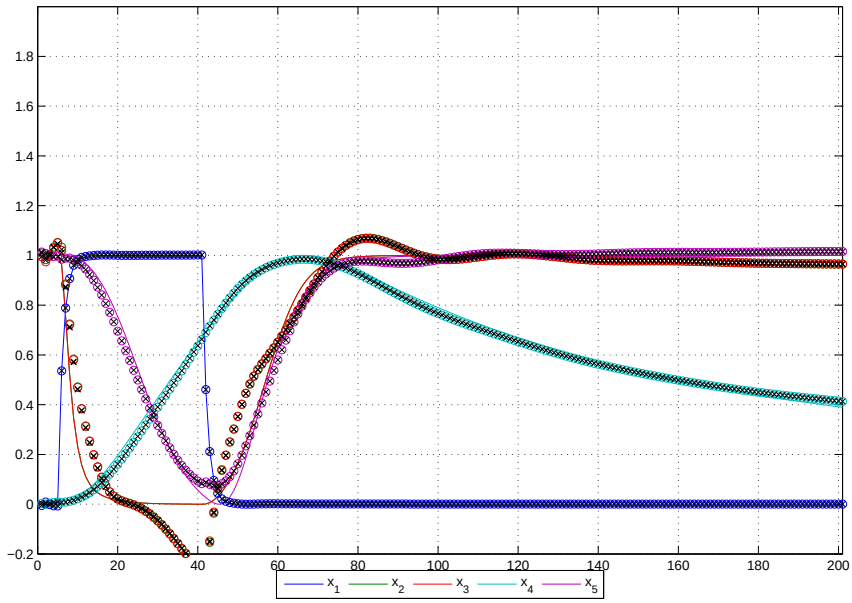


Figure 5.15: Comparison of estimation between the identified state space and the transformed state space.

From the $\tilde{A}_4$ matrix, we observe the direct interaction between genes $\{M_C, M_G, M_K, M_H, M_J\}$ across one time step. Ideally, we hope to find signed magnitudes that match the activation/repression map in Figure 5.8, similar to the table below.

Table 5.20: A tabular representation of the cascade motif described in Figure 5.8.

	C_k	G_k	K_k	H_k	J_k
C_{k+1}	x	x	x	x	x
G_{k+1}	$\ominus$	x	x	x	x
K_{k+1}	$\ominus$	x	x	x	x
H_{k+1}	$\oplus$	$\ominus$	x	x	x
J_{k+1}	$\ominus$	x	$\oplus$	x	x

Furthermore, from the $\tilde{A}_3$ matrix, we observe the indirect interaction across multiple time steps. Combined these two matrices should delineate a structure similar to that outlined in the table above. Unfortunately, these connections are not readily obvious in either of the discrete transformed matrices; neither completely delineate the true activation/repression scheme of the simulated system. But there is a plausible explanation for this. From the sample figures above, note that states M_G, and M_K are very similar in speed and dynamics. In fact, in Zak's model, the two states are identical. With only knowledge of the input-output behavior, the two appear indifferentiable. Their influences can be arbitrarily attributed to either of them without

changing the input-output behavior. Thus, a simulated influence from one of them could be mistakenly identified as an influence from the other.

Hypothesize that the influences of M_G and M_K are cumulative because the two states have the same dynamics. Then some of the connections become more distinct. The indirect activation of M_H by M_C and the indirect repression of M_J by M_C are distinguishable in the $\tilde{A}_4$ matrix. Direct influences are not very obvious in the direct gene to gene matrices but some of this inaccuracy can be attributed to the input to gene matrices, $\tilde{B}_3$ and $\tilde{B}_4$. The most immediate influence of the input, observed in $\tilde{B}_4$, is on M_C, which is reasonable because they directly stimulate M_C in Zak's modified model simulation. In the indirect input to gene matrix, $\tilde{B}_3$, we observe a large repressive effect from the input on states M_G and M_K, suggesting that the influences of M_C have been attributed to the artificial input. Further, we also observe that the signs of the coefficients corresponding to the input's influence on M_H and M_J match those predicted for M_G's effect on M_H and M_K's effect on M_J. So, we can also deduce that the influences of M_G and M_K have been attributed to the input as well.

5.4.2 Interpreting the Entire Model

In this instance, we can simply use the identified model in section 5.3.5. We are specifically interested in the case with five hidden states, one hidden bias state, and five observed states. As this model was already previously identified, its matrices and relevant plots are not reshown here. Instead, we calculate the correspondent $\tilde{A}_3$ and $\tilde{A}_4$ matrices directly. They are tabulated below. From the previous example, we know that the input-output behavior remains unchanged.

Table 5.21: Transformed matrices $\tilde{A}_3$ and $\tilde{A}_4$ for the entire model.

Discrete									
$\tilde{A}_3$									
-1.30	1.66	-0.06	-14.62	-0.55	0.35	0.46	0.31	4.33	1.02
-0.84	12.22	-0.56	-68.05	-4.83	1.52	9.02	-0.76	7.31	10.51
-3.86	5.73	-0.54	-114.71	-2.05	-0.96	-5.99	0.80	91.30	9.07
0.06	-0.43	0.02	1.10	0.20	-0.06	-0.17	-0.02	0.23	-0.39
-1.98	23.69	-1.01	-135.69	-9.06	2.73	16.06	-1.40	18.83	19.23
0.05	-6.66	0.31	16.87	3.21	-0.89	-4.02	0.17	5.79	-6.48
0.38	0.08	0.02	4.19	-0.10	-0.10	-0.36	-0.28	-2.28	0.48
0.40	-0.04	0.03	5.40	-0.08	-0.12	-0.46	-0.21	-2.35	0.39
0.01	0.17	0.00	-0.34	-0.08	0.01	0.11	-0.00	-0.13	0.19
0.05	0.17	0.01	-0.69	-0.09	-0.01	0.06	0.06	0.93	-0.44
$\tilde{A}_4$									
1.54	-2.30	0.05	15.74	0.91	-2.11	5.63	-7.71	-4.35	-1.07
-3.77	-15.47	0.51	73.40	7.54	-10.73	14.51	-30.51	-7.09	-11.45
4.16	-9.51	1.48	123.15	4.77	-10.32	-37.11	39.43	-94.32	-10.19
0.08	0.55	-0.02	-0.26	-0.28	0.34	0.06	0.35	-0.24	0.43
-6.26	-31.74	0.91	146.30	15.28	-20.51	26.43	-55.34	-18.59	-20.96
2.45	8.29	-0.29	-18.30	-4.14	5.05	0.53	6.83	-6.05	7.13
-0.42	-0.03	-0.01	-4.49	0.07	0.42	-2.23	3.91	2.30	-0.55
-0.42	0.14	-0.02	-5.79	0.00	0.58	-3.06	4.85	2.38	-0.45
-0.08	-0.21	-0.00	0.37	0.10	-0.09	-0.26	0.06	1.11	-0.21
-0.08	-0.18	-0.00	0.74	0.09	-0.10	-0.60	0.44	-0.96	1.37

In this larger system, an optimistic connection matrix would contain the following signs.

Table 5.22: A tabular representation of the entire model described in Figure 5.8.

	A_k	B_k	F_k	E_k	D_k	C_k	G_k	K_k	H_k	J_k
A_{k+1}	$\oplus$	x	x	x	x	x	x	x	x	x
B_{k+1}	$\oplus$	x	$\oplus$	x	x	x	x	x	x	x
F_{k+1}	x	x	x	$\oplus$	x	x	x	x	x	x
E_{k+1}	x	x	x	x	$\ominus$	x	x	x	x	x
D_{k+1}	x	x	$\oplus$	x	x	$\ominus$	x	x	x	x
C_{k+1}	$\oplus$	x	x	x	$\ominus$	x	x	x	x	x
G_{k+1}	x	x	x	x	x	$\ominus$	x	x	x	x
K_{k+1}	x	x	x	x	x	$\ominus$	x	x	x	x
H_{k+1}	x	x	x	x	x	x	$\ominus$	x	x	x
J_{k+1}	x	x	x	x	x	x	x	$\oplus$	x	x

The entire system described by Zak is much more complex than the single motif model so the model is significantly harder to interpret. For the discrete case, we added the two transformed matrices to combine direct and indirect gene-to-gene interaction and found that it improved the accuracy of the connection map.

Some connections are blatantly obvious. The auto-activation of state M_A is readily apparent as the largest positive coefficient in the state's coefficient row. The repression of M_C by M_D is significant in magnitude and accurate in sign. Similarly, the influence of M_A on M_C is correctly realized as activating and fairly large compared to other influencing factors on M_C. The activation of M_F by M_E is glaringly obvious in magnitude and correct in sign. Finally, the repression of M_H by M_G and the activation of M_J by M_K are both accurately mapped as significant influences.

Other connections, however, are more obscure though not necessarily wrong. The coefficient modeling activation of M_B by M_F is accurate in sign but small in magnitude when compared with other coefficients in their row. Similarly, repression of M_E by M_D is accurate in sign but inaccurately small in magnitude. This circumstance also holds true for the influences of M_F on M_D. We observed these coefficients because we knew they represented valid connections. However, in a true identification instance, when the true connections are unknown, these connections could easily have been overlooked or dismissed because they appear so negligible. Still, other connections are completely wrong in both magnitude and sign. For example, the activation of M_B by M_A is wrongly mapped as an repressing relation. The influences of M_C on M_G and M_K are wrong in sign.

Our transform solution has illuminated the linearized gene-to-gene interactions but full model identification still eludes us. Our interpretation results may improve with the application of the general transformation case, where the number of hidden states does not equal the number of gene states. In such an instance, the identified system may pick up more accurate attributes from the simulation system such that the direct feedthrough matrices offer a better approximation of the average gene-to-gene interaction matrix. Added hidden states will improve tracking. Further, the general solution incorporates more time delayed versions of the gene to gene interaction matrices, which may facilitate the interpretation direct and indirect influences.

Chapter 6

Conclusions and Future Work

In this thesis, we introduced a linear dynamical model for simulating gene regulatory networks. It differed from Rangel's proposed state-space model [21], from which it was derived, in that it treated gene states as hidden states so that observation noises did not affect the dynamics of the states via feedback. Further, it incorporated an additional bias state that accounted for non-zero steady-states in the measurement data. We trained our model using the Expectation-Maximization algorithm, of which we included a detailed derivation and proof.

Using simple linear test cases, we explored the parameter differences between discrete and continuous systems, the effects of discretization and uneven sampling, the effects of discrete-to-continuous conversion, and the identifiability of state transformed systems. For the simple linear instances we used, the state transformation did not impact the identifiability of the direct feedthrough matrix, which we analyzed for gene-to-gene relations. However, when training our proposed system on nonlinear test cases, we discovered that the feedback loop in our system allowed a transform which did not cancel out in the direct feedthrough matrix. Further, we realized that this transform plagued Rangel's model as well. The presence of this transformation greatly obscured the gene-to-gene relations and proved difficult to eradicate.

Nevertheless, we propose several solutions to the problem of identifiability caused by state transformations that appear to improve the interpretation of the feedthrough matrix. In particular, we propose another state

transformation that redefines the original state vector in terms of only the gene states. Due to this transform, the entire state matrix of our model becomes a linearized measure of the genes' effects on themselves.

Our thesis offers a comprehensive idea of the limitations and assets of linear dynamic models. In particular, we found that the presence of the hidden states greatly facilitate the modeling of highly nonlinear gene pathways. We also discovered that the state-space model was capable of capturing many of the dynamics with the help of hidden states and could even predict system dynamics with some time delay error. On the other hand, we realized that it was possible to overtrain the model with too many hidden states such that, although it matched the training data perfectly, the model performed drastically worse when predicting new data set dynamics. Also, we realized the complexity of the interpretation problem given the trained state matrix parameters. For, although the identified systems could generate very good estimates and predictions, we could only analyze a portion of it, the direct feedthrough matrix, consistently across multiple-state systems. Thus, even though the system appeared well identified in performance, our method of analysis still overlooked several relevant gene-to-gene relations.

In future work, we plan to apply the general case of the state transformation solution to our identified systems. The general case allows us to work with larger systems and more hidden states. As increased numbers of hidden states improve the truth of the identified model, the general transformation method may prove more informative and accurate than our simple solution case. However, it is possible that the true connection network still eludes us because of our method of analysis and identification. The feedthrough matrix we analyze offers a linear approximation of the gene pathways; depending on how nonlinear the true network connections are, this approximation may deviate a great deal from the actual pathways that generated them. Other methods such as selective initial state stimulation, knockout state simulations, and deductive analysis of combinatorial simulation data may offer an alternative to regulatory pathway identification. Finally, we consider an elaboration of our linear state space system into the nonlinear state space domain. Such a model change would radically change our method of training but may also improve the accuracy and precision of our identified models significantly.

Appendix

Normalization

On a side note, normalization of the data affects the system that the EM algorithm identifies because it changes the output behavior of the system. Hence its effects should either be accounted for during the identification process or removed from the estimated system after identification but before analysis. In the first method, it can be accounted for by treating the normalization as weights on the diagonal elements of the observation matrix, C. So whereas before the C matrix consisted of a horizontal concatenation of a zero matrix and an identity matrix, now the identity partition is replaced with a diagonal matrix consisting of the normalization weights used on each state.

Table A-1: Revised observation matrix C accounting for normalization.

C						
0	...	0	$M_{y_1\ weight}$	0	...	0
⋮	⋱	⋮	0	$M_{y_2\ weight}$	⋱	0
0	...	0	0	0	...	$M_{y_{n-p}\ weight}$

The second method treats normalization as a linear transform on the augmented system states. This transform is a diagonal matrix that consists of ones on rows corresponding to the hidden states and normalization factors on rows corresponding to the observed states. Thus, once the EM output the identified A matrix, this transform would be reversed via a state transformation reversal before it was analyzed, i.e. $A_{unnormalized} = NA_{normalized}N^{-1}$ etc. where N is the normalization transform.

A few identified feedthrough matrices were unnormalized and re-tabulated below. But unnormalization is probably not a viable solution in our system because the uneven scaling tends to emphasize some high-

Table A-2: Normalization transform matrix, N.

N					
1	...	0	0	...	0
⋮	⋱	⋮	⋮	...	⋮
0	...	1	0	...	0
0	...	0	$M_{y_1\ weight}$	...	0
⋮	...	⋮	⋮	⋱	⋮
0	...	0	0	...	$M_{y_{n-p}\ weight}$

concentration transcripts and understate others, making it even harder to see inducing and suppressing influences. Normalization puts the coefficients in the right numerical range of actual "concentration" levels but when we are merely trying to discern influencing connections, these fluctuating values are not beneficial to our objective. If our model was nonlinear and capable of precisely modeling the natural system, then, prior to prediction, unnormalization would be necessary to discern potential relevant concentration fluctuations during experiments. For our purposes, the actual concentration magnitudes are irrelevant and would probably deter us from accurately identifying influences as the table below illustrates.

Table A-3: Comparison of normalized estimated A and feed-through matrices for Zak's cascade motif.

Discrete CB+D Matrix					Continuous CB+D Matrix (10^-2)				
System identified assuming 5 hidden states									
0.41	0.00	0.00	0.00	0.00	-2.56	-0.00	0.00	0.00	0.00
29.48	0.41	0.01	0.00	0.01	230.09	-2.76	0.61	0.79	0.29
-0.88	0.02	0.45	0.03	0.00	-105.70	-0.19	-1.18	0.83	0.09
-3.10	-0.00	0.01	0.95	0.00	-210.95	-0.19	0.34	0.11	0.08
139.37	0.05	-0.02	-0.05	0.70	1528.23	2.97	-2.51	-2.45	-1.64
System identified assuming 10 hidden states									
0.41	0.00	0.00	0.00	0.00	-2.58	-0.00	0.00	0.03	0.00
6.82	0.44	-0.01	0.32	0.00	-496.14	-1.85	0.03	-1.36	0.48
22.89	0.02	0.44	0.14	-0.00	-135.67	0.30	-0.18	15.73	-0.14
2.49	0.00	0.00	0.63	-0.00	276.12	0.17	-0.17	-4.38	-0.19
80.76	0.11	-0.01	0.40	0.70	1949.26	1.20	-0.51	-25.30	-1.84
System identified assuming 15 hidden states									
0.36	0.00	0.00	0.00	0.00	-1.59	0.00	0.00	0.01	0.00
-39.56	0.44	-0.00	0.25	-0.00	-306.02	-2.15	0.15	-3.12	0.12
32.74	0.02	0.46	0.00	0.00	754.02	0.69	-1.33	2.99	0.23
-1.47	0.00	0.01	0.69	-0.01	75.12	0.13	-0.04	-3.04	-0.22
112.39	0.12	-0.06	0.32	0.69	4837.71	0.13	-0.32	-41.22	-1.43
System identified assuming 20 hidden states									
0.38	0.00	-0.00	0.00	0.00	-1.58	0.00	0.00	0.00	0.00
6.88	0.42	0.03	-0.04	0.00	-3.10	-1.22	0.11	1.70	0.08
27.14	0.03	0.46	0.13	-0.00	136.50	-0.25	-0.93	-0.26	0.02
-2.99	0.00	0.00	0.70	-0.00	-122.55	-0.01	0.01	-1.97	-0.05
175.41	0.05	-0.09	-0.01	0.70	1516.39	-1.65	0.48	-18.76	-1.14

Bibliography

[1] Tatsuya Akutsu, Satoru Miyano, and Satoru Kuhara. Identification of genetic networks from a small number of gene expression patterns under the boolean network model. In *Pacific Symposium on Biocomputing 4*, pages 17–28, 1999.

[2] U. Alon, N. Barkai, D. A. Notterman, K. Gish, S. Ybarra, D. Mack, and A. J. Levine. Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. In A. J. Levine, editor, *Proc. Natl. Acad. Sci. USA*, page 67456750, June 1999.

[3] Nan Bing. *Statistical Analysis of Gene Expression Profile: Transcription Network Inference and Sample Classification*. PhD thesis, Virginia Polytechnic Institute and State University, April 2004.

[4] Sean Borman. The expectation maximization algorithm – a short tutorial. Introduces the Expectation Maximization (EM) algorithm and fleshes out the basic mathematical results, including a proof of convergence. The Generalized EM algorithm is also introduced., July 2004.

[5] T. Chen, H. L. He, and G. M. Church. Modeling gene expression with differential equations. In *Pacific Symposium on Biocomputing 4*, pages 29–40, 1999.

[6] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 39(1):1–38, 1977.

[7] P. D'Haeseleer, S. Liang, and R. Somogyi. Genetic network inference: from co-expression clustering to reverse engineering. *Bioinformatics*, 16(8):707–726, June 2000.

[8] P. D'Haeseleer, X. Wen, S. Fuhrman, and R. Somogyi. Linear modeling of mrna expression levels during cns development and injury. In *Pacific Symposium on Biocomputing 4*, pages 41–52, 1999.

[9] Joaquin Dopazo, Edward Zanders, Ilaria Dragoni, Gillian Amphlett, and Francesco Falcianic. Methods and approaches in the analysis of gene expression data. *Journal of Immunological Methods*, 250:93112, 2001.

[10] E.R. Dougherty, J. Barrera, M. Brun, S. Kim, R.M. Cesar, Y. Chen, M. Bittner, and J.M. Trent. Inference from clustering: Application to gene-expression time series. *Journal of Computational Biology*, 9(1), 2002.

[11] Michael B. Eisen, Paul T. Spellman, Patrick O. Brown, and David Botstein. Cluster analysis and display of genome-wide expression patterns. In David Botstein, editor, *Proc. Natl. Acad. Sci. USA*, page 1486314868, December 1998.

[12] Nir Friedman, Matan Ninio, Itsik Pe'er, and Tal Pupko. A structural em algorithm for phylogenetic inference. *Journal of Computational Biology*, 9(2):331–353, 2002.

[13] Zoubin Ghahramani and Geoffrey E. Hinton. Parameter estimation for linear dynamical systems. Technical Report CRG-TR-96-2, Department of Computer Science, University of Toronto, February 1996.

[14] Dirk Husmeier and Frank Wright. Detection of recombination in dna multiple alignments with hidden markov models. *Journal of Computational Biology*, 8(4):401–427, 2001.

[15] Trey E. Ideker, Vesteinn Thorsson, and Richard M. Karp. Discovery of regulatory interactions through perturbation: Inference and expreimental design. In *Pacific Symposium on Biocomputing 5*, pages 302–313, 2000.

[16] S. A. Kauffman. Metabolic stability and epigenesis in randomly constructed genetic nets. *Journal of Theoretical Biology*, 22(3):437–467, July 1969.

[17] Koji M. Kyoda, Mineo Morohashi, Shuichi Onami, and Hiroaki Kitano. A gene network inference method from continuous-value gene expression data of wild-type and mutants. *Genome Informatics*, 11:196204, 2000.

[18] Shoudan Liang, Stephanie Fuhrman, and Roland Somogyi. Reveal, a general reverse engineering algorithm for inference of genetic network architectures. In *Pacific Symposium on Biocomputing 3*, pages 18–29, 1998.

[19] Kevin Murphy and Saira Mian. Modelling gene expression data using dynamic bayesian networks. Technical report, Computer Science Division, University of California, 2001.

[20] Dana Pe'er, Aviv Regev, Gal Elidan, and Nir Friedman. Inferring subnetworks from perturbed expression profiles. *Bioinformatics*, 1(1):1–9, 2001.

[21] Claudia Rangel, John Angus, Zoubin Ghahramani, Maria Lioumi, Elizabeth Sotheran, Alessia Gaiba, David L. Wild, and Francesco Falciani. Modeling t-cell activation using gene expression profiling and state space models. *Bioinformatics*, 20(9):1361–1372, June 2004.

[22] H. E. Rauch, F. Tung, and C. T. Striebel. Maximum likelihood estimates of linear dynamic systems. *AIAA Journal*, 3(3):1445–1450, August 1965.

[23] Habtom W. Ressom, Yuji Zhang, Jianhua Xuan, Yue Wang, and Robert Clarke. Inference of gene regulatory networks from time course gene expression data using neural networks and swarm intelligence. To appear in the Proceedings of the 28th IEEE Engineering in Medicine and Biology Society Annual International Conference, the First International Conference on Computational Systems Biology, and the 2006 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology., 2006.

[24] R. H. Shumway and D. S. Stoffer. An approach to time series smoothing and forecasting using the EM algorithm. *Journal of Time Series Analysis*, 3(4):253–264, 1982.

[25] P.H.A. Sneath and R.R. Sokal. *Numerical Taxonomy.* W.H. Freeman, San Francisco, 1973.

[26] P. Tamayo, D. Slonim, J. Mesirov, Q. Zhu, S. Kitareewan, E. Dmitrovsky, E.S. Lander, and T.R. Golub. Interpreting patterns of gene expression with self-organizing maps: Methods and application to hematopoietic differentiation. In *Proc. Natl. Acad. Sci. USA 96*, page 29072912, 1999.

[27] S. Tavazoie, J. D. Hughes, M. J. Campbell, R. J. Cho, and G. M. Church. Systematic determination of genetic network architecture. *Nature Genetics*, 22(3):281–285, July 1999.

[28] Eberhard O. Voit and Tomas Radivoyevitch. Biochemical systems analysis of genome-wide expression data. *Bioinformatics*, 16(11):1023–1037, 2000.

[29] Mattias Wahde and John Hertz. Modeling genetic regulatory dynamics in neural development. *Journal of Computational Biology*, 8(4):429–442, 2001.

[30] Yonghong Wang, Hongyu Ou, and Fengbiao Guo. Recognition of translation initiation sites eukaryotic genes based on an em algorithm. *Journal of Computational Biology*, 10(5):699–708, 2003.

[31] D. C. Weaver, C. T. Workman, and G. D. Stormo. Modeling regulatory networks with weight matrices. In *Pacific Symposium on Biocomputing 4*, pages 112–123, 1999.

[32] Max Welling. The kalman filter. Classnotes made available on the web offering an introduction to and derivation of the Kalman filter and smoother algorithms.

[33] Daniel E. Zak, Gregory E. Gonye, James S. Schwaber, and Francis J. Doyle III. Importance of input perturbations and stochastic gene expression in the reverse engineering of genetic regulatory networks: Insights from an identifiability analysis of an in silico network. *Genome Research*, 13:2396–2405, September 2003.

[34] Xiaobo Zhou, Xiaodong Wang, Edward R. Dougherty, Daniel Russ, and Edward Suh. Gene clustering based on clusterwide mutual information. *Journal of Computational Biology*, 11(1):147–161, 2004.