Computational Methods for Bayesian Inference in Complex Systems

Thesis by

Thomas Anthony Catanach

In Partial Fulfillment of the Requirements for the
degree of
Doctor of Philosophy

CALIFORNIA INSTITUTE OF TECHNOLOGY
Pasadena, California

2017
Defended May 16, 2017

Thomas Anthony Catanach
ORCID: 0000-0002-4321-3159

ACKNOWLEDGEMENTS

I would first like to thank my advisor, Prof. Jim Beck. Jim has shaped how I think about the world by introducing me to the Bayesian perspective and has been an excellent role model. He has demonstrated how to effectively bridge pure and applied research and helped me develop the tools I need for my future career. Further, I have truly enjoyed talking with him about a whole range of topics from the geography of New Zealand to the Bayesian perspective on quantum physics.

I am also very thankful to the faculty of Caltech who have been instrumental in guiding me and my work. I would like to thank my thesis committee members, Prof. Andrew Stuart, Prof. Mark Simons, and Prof. Steven Low. Their input in my work and knowledge of their respective fields has been most helpful. The classes I took during graduate school really transformed how I think about my research, particularly those in CDS and probability. Therefore, I would like to thank Prof. Richard Murray, Prof. John Doyle, and Prof. Houman Owhadi.

The CMS department and the general Caltech community has been a great place to work. I would like to thank Maria, Sydney, Carmen, and Nikki for all your help. I would also like to thank Armeen, Noah, Utkan, Pinaky, Yong Sheng, Yoke Peng, Tony, Yorie, Niangjun, Ioannis, and Mickey. My friends here at Caltech have been great people to share this experience with. I have really benefitted from your company and your friendship.

I would also like to thank the Department of Energy Computational Science Graduate Fellowship program for providing me with both funding and guidance. I really enjoyed the two summers I spent working at Los Alamos National Laboratory and would like to thank Earl Lawrence, Russell Bent, and Scott Vander Wiel and all my other colleagues there.

I would also like to thank all of those who encouraged me to pursue science and mathematics; particularly, Sr. Cecilia, Mr. Morrow, Prof. John Poirier, and most importantly my family. My Grandpa George served as a role model for me on how to lead a long happy life full of intellectual curiosity and creativity. My parents have been incredibly supportive through my life in all things. I am so grateful for the values they instilled in me, the interests they cultivated, and all the love they showed me. They put in many long hours helping me with my science projects and teaching me math, science, and programming. Through this, they taught me

both how to love and how to learn. I have also been blessed to have two wonderful sisters, Therese and Maria. I have really enjoyed all the trips and times we have had together. Therese has been my partner in crime and Maria has been there to get us out of trouble when we needed it. I am happy that we share so much.

Finally, and most significantly, I want to thank my wonderful wife Ania for all of her love, strength, and support while working on this thesis. It is dedicated to you. I appreciate all the tea you have made for me, ice cream we have eaten together, and walks we have taken to Peet's and Trader Joe's that kept me going through these past few months. You have constantly encouraged me to be better, both in who I am and in my work. I look forward to many more years of love, companionship, and collaboration with you in all things.

ABSTRACT

Bayesian methods are critical for the complete understanding of complex systems. In this approach, we capture all of our uncertainty about a system's properties using a probability distribution and update this understanding as new information becomes available. By taking the Bayesian perspective, we are able to effectively incorporate our prior knowledge about a model and to rigorously assess the plausibility of candidate models based upon observed data from the system. We can then make probabilistic predictions that incorporate uncertainties, which allows for better decision making and design. However, while these Bayesian methods are critical, they are often computationally intensive, thus necessitating the development of new approaches and algorithms.

In this work, we discuss two approaches to Markov Chain Monte Carlo (MCMC). For many statistical inference and system identification problems, the development of MCMC made the Bayesian approach possible. However, as the size and complexity of inference problems has dramatically increased, improved MCMC methods are required. First, we present Second-Order Langevin MCMC (SOL-MC), a stochastic dynamical system-based MCMC algorithm that uses the damped second-order Langevin stochastic differential equation (SDE) to sample a desired posterior distribution. Since this method is based on an underlying dynamical system, we can utilize existing work in the theory for dynamical systems to develop, implement, and optimize the sampler's performance. Second, we present advances and theoretical results for Sequential Tempered MCMC (ST-MCMC) algorithms. Sequential Tempered MCMC is a family of parallelizable algorithms, based upon Transitional MCMC and Sequential Monte Carlo, that gradually transform a population of samples from the prior to the posterior through a series of intermediate distributions. Since the method is population-based, it can easily be parallelized. In this work, we derive theoretical results to help tune parameters within the algorithm. We also introduce a new sampling algorithm for ST-MCMC called the Rank-One Modified Metropolis Algorithm (ROMMA). This algorithm improves sampling efficiency for inference problems where the prior distribution constrains the posterior. In particular, this is shown to be relevant for problems in geophysics.

We also discuss the application of Bayesian methods to state estimation, disturbance detection, and system identification problems in complex systems. We introduce a Bayesian perspective on learning models and properties of physical systems based

upon a layered architecture that can learn quickly and flexibly. We then apply this architecture to detecting and characterizing changes in physical systems with applications to power systems and biology. In power systems, we develop a new formulation of the Extended Kalman Filter for estimating dynamic states described by differential algebraic equations. This filter is then used as the basis for sub-second fault detection and classification. In synthetic biology, we use a Bayesian approach to detect and identify unknown chemical inputs in a biosensor system implemented in a cell population. This approach uses the tools of Bayesian model selection.

PUBLISHED CONTENT AND CONTRIBUTIONS

[Bae+16] Ania-Ariadna Baetica, Thomas Anthony Catanach, Victoria Hsiao, Richard Murray, and James Beck. "A Bayesian approach to inferring chemical signal timing and amplitude in a temporal logic gate using the cell population distributional response". In: *bioRxiv* (2016), p. 087379.
TA Catanach developed this problem in the Bayesian framework and the computational methods used to solve it. The method presented in this paper is used in Chapter 5.

[CB17] Thomas A Catanach and James L Beck. "Bayesian System Identification using Auxiliary Stochastic Dynamical Systems". In: *International Journal of Non-Linear Mechanics* (2017). DOI: 10.1016/j.ijnonlinmec.2017.03.012. URL: https://doi.org/10.1016/j.ijnonlinmec.2017.03.012.
TA Catanach developed, implemented, and tested the SOL-MC method. These results are presented in Chapter 2.

[Gar+15] M. Garcia, T. Catanach, S.V. Wiel, R. Bent, and E. Lawrence. "Line Outage Localization Using Phasor Measurement Data in Transient State". In: *Power Systems, IEEE Transactions on* PP.99 (2015), pp. 1–9. ISSN: 0885-8950. DOI: 10.1109/TPWRS.2015.2461461.
TA Catanach worked on the feature selection used in this work. This machine learning approach to fault detection is discussed in Chapter 4.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

Number *Page*

Chapter 1

INTRODUCTION TO BAYESIAN METHODS

Bayesian methods for identification and estimation are critical to the robust understanding of a system because they allow us to quantify all of our uncertainty about the system using a probability distribution and to update this distribution with new information [BK98; BA02; Ves08; Bec10; Yue10b; Yue10a; WH12; APK15; GW15]. By taking the Bayesian approach, we are able to effectively capture our prior knowledge about a model and rigorously assess the plausibility of candidate model classes based on system data. Finally, we can then make robust probabilistic predictions that incorporate all uncertainties, allowing for better decision making and design. This robust approach is particularly relevant for complex system identification, where the inverse problems are often ill-posed, many candidate models exist to describe the behavior of a system, and stochastic models are common.

1.1 Bayesian Inference

The Bayesian framework is a rigorous probabilistic method for representing uncertainty using probability distributions. This philosophy is rooted in probability as a logic [Bec10; Cox46; Cox61; Jay03]. Within this framework, probability distributions are used to quantify uncertainty due to insufficient information, regardless of whether that information is believed to exist but is currently not available (epistemic uncertainty), or it is believed to not exist because of postulated inherent randomness (aleatory uncertainty). This notion of uncertainty makes the Bayesian framework the appropriate framework for posing system identification problems, where postulated system models have parameters whose values are uncertain rather than random. Therefore, we view system identification as updating a probability distribution that represents our beliefs about models of a system based on new information from system response data.

In general, the Bayesian inference problem uses Bayes' theorem to update the understanding of a system using data, where understanding means assigning a probability function $p()$ to different system descriptions. This updating process is visualized in Figure 1.1. The inference problem is formulated as follows: Given output measurements $z_i \in \mathcal{D}$, where $\mathcal{D}$ is the set of data, and a system description, $s \in \mathcal{S}$, where $\mathcal{S}$ is a set of descriptions, consisting of (a) a likelihood function

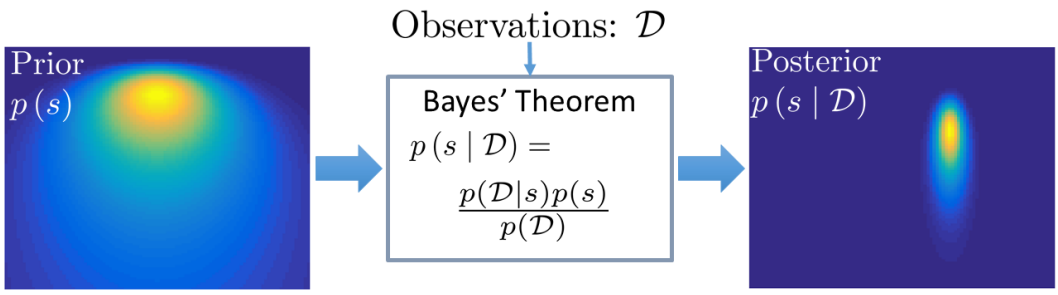

Figure 1.1: Bayesian updating for the system description s with the prior distribution $p(s)$ to the posterior distribution $p(s \mid \mathcal{D})$ given data $\mathcal{D}$

$p(\mathcal{D} \mid s)$, describing the plausibility of the data given the description s, and (b) a prior distribution, $p(s)$, representing the beliefs about the relative plausibility of s, find the posterior distribution $p(s \mid \mathcal{D})$ that represents the updated belief after integrating the observational data. For this, Bayes' Theorem is used:

$$p(s \mid \mathcal{D}) = \frac{p(\mathcal{D} \mid s)\, p(s)}{p(\mathcal{D})}. \tag{1.1}$$

The likelihood function, $p(\mathcal{D} \mid s)$, is the likelihood of observing the data $\mathcal{D}$ given the model of the system. This model in the Bayesian framework maps s to a probability distribution on the outputs z_i. The normalizing factor in equation (1.1), $p(\mathcal{D})$, is

$$p(\mathcal{D}) = \int_S p(\mathcal{D} \mid s)\, p(s)\, ds \tag{1.2}$$

We can broadly classify the posterior probability distributions that result from solving the inference problem into three types: globally identifiable, locally identifiable, and unidentifiable [BK98; KB98]. Globally identifiable probability distributions have a single pronounced peak around a unique maximum. Locally identifiable distributions have several separated peaks, each of approximately the same significance. Unidentifiable models do not have peaks, but instead have a manifold in the parameter space on which all values are approximately equally plausible based on the data and the selected prior information. When the problem results in a locally identifiable or unidentifiable distribution, Bayesian methods are essential since they can fully capture this complex distribution in a way optimization based system identification methods cannot. However, these types of problems still produce a

significant challenge to computational Bayesian methods since it is often difficult to find and explore all the peaks or the manifold of plausible solutions.

1.2 Bayesian Methods for Complex Systems

Inference Problems

Bayesian inference, as introduced in Section 1.1, for complex systems can broadly be divided into three fundamental problems: state estimation, parameter estimation, and model selection. These problems are interconnected and solving one cannot be done without assuming a solution to the others or solving them simultaneously. Together, these processes describe a hierarchy of Bayesian inference problems for dynamical systems, which we can use to design a Bayesian inference architecture.

This work considers both static and dynamic systems that can be described by a specific model $\mathcal{M}$ or a discrete set of possible models also called model classes, $\mathcal{M}_i$ for $i = 1...N$. These models may be informed by an understanding of the physics of the complex system or by some set of functions that are believed to capture the space of possible behaviors of the system.

In the model selection problem, the likelihoods of different models $\mathcal{M}_i$ for $i = 1...N$ can be computed based upon the data from a system, $\mathcal{D}$. The probability of any model $\mathcal{M}_i$, given observational data $\mathcal{D}$, is defined as $p(\mathcal{M}_i \mid \mathcal{D})$. Assuming the prior probability of a model is known and defined as $p(\mathcal{M}_i)$ such that $\sum_i^N p(\mathcal{M}_i) = 1$, then the posterior is

$$p(\mathcal{M}_i \mid \mathcal{D}) = \frac{p(\mathcal{D} \mid \mathcal{M}_i)\, p(\mathcal{M}_i)}{p(\mathcal{D})} \tag{1.3}$$

Using the law of total probability:

$$p(\mathcal{M}_i \mid \mathcal{D}) = \frac{p(\mathcal{D} \mid \mathcal{M}_i)\, p(\mathcal{M}_i)}{\sum_j^N p\left(\mathcal{M}_j\right) p\left(\mathcal{M}_j\right)} \tag{1.4}$$

For some basic Bayesian inference problems, such as those using non-parametric models or a dictionary of known behaviors, the inference problem stops here. However, for many inference problems, a model class $\mathcal{M}_i$ only describes some functional form whose range of possible behaviors is captured by setting parameters θ attached to that model class. This leads to the parameter estimation or system identification problem.

Applying the Bayesian inference formulation to parameter estimation and system identification yields the description of Bayesian system identification given in [Bec10]: Given observation data $\mathcal{D}$, and assuming a system model class $\mathcal{M}^*$ consisting of (a) a likelihood function $p(\mathcal{D} \mid \theta, \mathcal{M}^*)$, describing the plausibility of the data given a set of parameters, θ, and (b) a prior distribution, $p(\theta \mid \mathcal{M}^*)$, representing the initial beliefs about the relative plausibility of the possible values of the model parameter vector θ, find the posterior distribution $p(\theta \mid \mathcal{D}, \mathcal{M}^*)$ that represents the updated beliefs as

$$p(\theta \mid \mathcal{D}, \mathcal{M}^*) = \frac{p(\mathcal{D} \mid \theta, \mathcal{M}^*)\, p(\theta \mid \mathcal{M}^*)}{p(\mathcal{D} \mid \mathcal{M}^*)} \tag{1.5}$$

The normalizing factor in equation (1.5), $p(\mathcal{D} \mid \mathcal{M}^*)$, is the evidence for the model class $\mathcal{M}^*$. The evidence can be computed as

$$p(\mathcal{D} \mid \mathcal{M}^*) = \int p(\mathcal{D} \mid \theta, \mathcal{M}^*)\, p(\theta \mid \mathcal{M}^*)\, d\theta \tag{1.6}$$

This evidence can then be used in Equation 5.13 to solve the corresponding model class section problem if multiple model classes exist to describe the behavior.

The parameter estimation or system identification problem is the most typical formulation for Bayesian inference. It captures static systems and deterministic dynamical systems. However, for a stochastic dynamical system, where the observed behavior of the system is a function of the state of the system, $x(t)$, and an unknown input modeled as a stochastic process, estimating this internal state is itself a Bayesian inference problem called state estimation or Bayesian filtering.

State estimation is the process of using measurements from a stochastic dynamical system to infer the dynamical states. Assuming the measurements are available at discrete times, it is appropriate to treat the system as a discrete time system:

$$\begin{aligned} x_k &= \mathfrak{F}(x_{k-1}, w_k, \theta, \mathcal{M}^*) \\ z_k &= \mathfrak{H}(x_k, v_k, \theta, \mathcal{M}^*) \end{aligned} \tag{1.7}$$

Here, $\mathfrak{F}(x, w, \theta, \mathcal{M})$ is the discrete stochastic dynamic state evolution function from model class $\mathcal{M}$ and parameterized by θ, while $\mathfrak{H}(x, w, \theta, \mathcal{M})$ is the stochastic measurement function also from model class $\mathcal{M}$ and parameterized by θ. At time t_k for $k = 1, 2, \ldots$ x_k are the state variables and z_k are the observed outputs. Further, w_k is

an unknown stochastic input, while ν_k is an unknown measurement disturbance or error.

State estimation can be done using the Bayes Filter, where the posterior distribution $p(x_k \mid z_{1:k})$ is updated from the prior state estimate distribution $p(x_k \mid z_{1:k-1})$ using the measurements up to time t_k represented as $z_{1:k}$. Using Bayes' Theorem and the Markov property, the posterior is found recursively by applying two equations:

First prediction of x_k given $z_{1:k-1}$:

$$p(x_k \mid z_{1:k-1}) = \int p\left(x_k \mid x'_{k-1}\right) p\left(x'_{k-1} \mid z_{1:k-1}\right) dx'_{k-1} \tag{1.8}$$

Then the correction step integrates the new observations, so the prediction of x_k given $z_{1:k}$ is

$$p(x_k \mid z_{1:k}) \propto p(z_k \mid x_k)\, p(x_k \mid z_{1:k-1}) \tag{1.9}$$

This inference problem can be visualized through the graphical model in Figure 1.2. The Bayes Filter produces the minimum mean square error estimate of the state[Che03]. When the dynamical system is linear and the prior state estimate and noise distributions are white Gaussian, the Bayes Filter is the Kalman Filter[Sim06; Che03].

Solutions to the state estimation problem are then marginalized over to solve the parameter estimation problem:

$$\begin{aligned} p(\theta_{1:n} \mid z_{1:k}) &= \frac{p(z_{1:k} \mid \theta_{1:n})\, p(\theta_{1:n})}{p(z_{1:k})} \\ &\propto \left(\int p(z_{1:k} \mid x_{1:k}, \theta_{1:n})\, p(x_{1:k} \mid \theta_{1:n})\, dx_{1:k} \right) p(\theta_{1:n}) \end{aligned} \tag{1.10}$$

The statistical relationship between states and observations for the Bayesian state estimation problem is expressed using the graphical model in Figure 1.3.

Similarly, solutions of the parameter estimation problem for the stochastic dynamical system are then marginalized over to compute the model evidence for model selection:

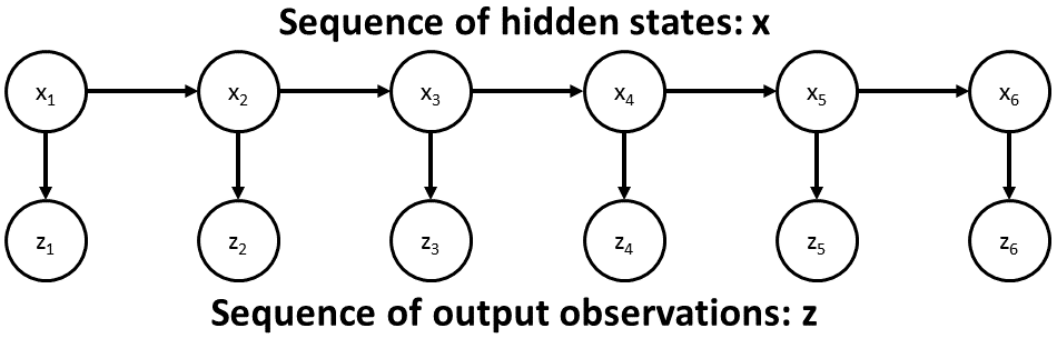

Figure 1.2: Graphical Model of Bayesian Filtering/State Estimation

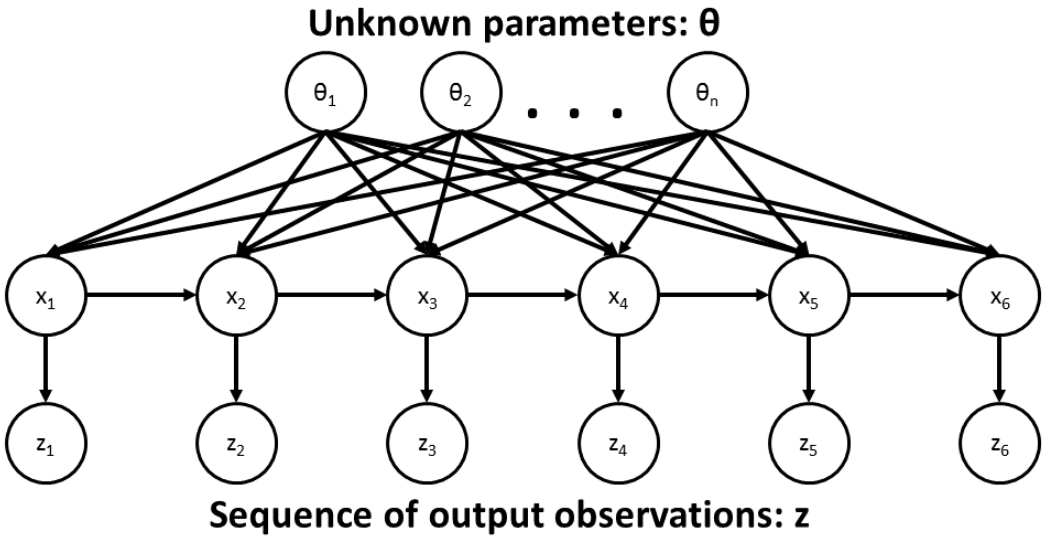

Figure 1.3: Graphical Model of Bayesian Parameter Estimation

$$
\begin{aligned}
p\left(\mathcal{M} \mid z_{1:k}\right) &= \frac{p\left(z_{1:k} \mid \mathcal{M}\right) p\left(\mathcal{M}\right)}{p\left(z_{1:k}\right)} \\
&\propto \left(\int p\left(z_{1:k} \mid \theta_{1:n}, \mathcal{M}\right) p\left(\theta_{1:n} \mid \mathcal{M}\right) d\theta_{1:n} \right) p\left(\mathcal{M}\right)
\end{aligned}
\tag{1.11}
$$

The statistical relationship between parameters, states, and observations is expressed as a graphical model in Figure 1.4.

Applications of Bayesian Methods

Taking a Bayesian perspective on studying complex dynamical systems enables scientists and engineers to quantitatively integrate all forms of uncertainty into their planning and decision making using probability. Solving inverse problems enables

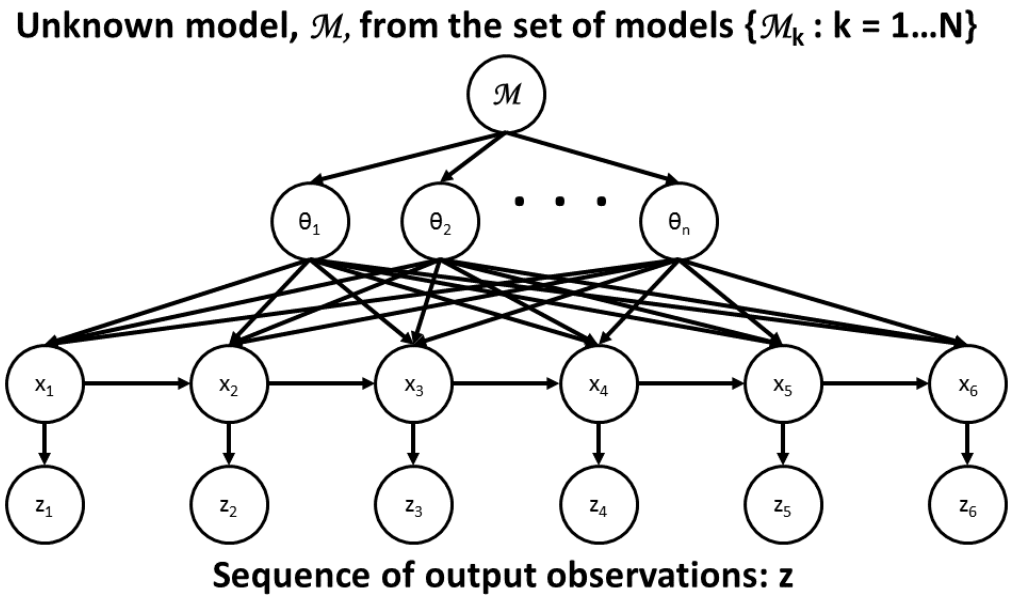

Figure 1.4: Graphical Model of Bayesian Model Selection

practitioners to develop a better understanding of system modeling uncertainties, which can be integrated into the Bayesian framework to solve problems in uncertainty quantification, Bayesian optimization, and optimal experimental design. These tools are critical to robust system development, operation, and investigation,

Uncertainty quantification makes robust predictions about the future taking into account all sources of uncertainty [KO01; Chk+13; BT13; Naj09]. After solving a Bayesian inference problem, this updated understanding of the system model can be incorporated into robust predictions by marginalizing over the collective modelling uncertainty and other sources of uncertainty.

Bayesian optimal experimental design is a method to close the loop between data collection and inference by addressing the question of what experiment should be performed or measurement taken to best improve the estimate of some quantity of interest [CV95; HM13; KSG08; Bus+13]. Typically, Bayesian optimal experimental design takes the form of maximizing the expected information gain, also known as the relative Shannon entropy or the Kullback-Leibler divergence, between the prior and posterior given this potential data. This approach relies on the ability to solve the inverse problem to estimate the posterior distribution. This type of experimental design is critical for complex systems where data is often very expensive and there are complex relationships that constrain the identifiability of the posterior.

Bayesian optimization allows practitioners to solve optimization problems with re-

spect to very complicated black-box objective functions [BCD10; Sri+09], including functions with noise and uncertainty. This is done using the Bayesian framework. This method can then be used to design and optimize systems under uncertainty and explore the space of potential solutions by balancing an exploration and exploitation tradeoff.

Bayesian Inference Architecture for Dynamical Systems

In order to efficiently integrate the Bayesian philosophy into the work of scientists and engineers, a general framework for inference problems for dynamical systems based upon theoretical models is needed. A conceptual approach to help formulate system level problems from the perspective of Bayesian inference is developed to do this in Figure 1.5. Typically, systems can be divided into processes that act on different temporal, spatial, and magnitude scales; therefore, learning and prediction algorithms should mirror these properties. By thinking about inference as a layered architecture, and utilizing this structure, inference can be performed quickly and flexibly.

On fast time scales, filtering methods are used for state estimation to determine the current operating state of the system. These methods are less flexible than full system identification estimation since they assume a system model but are very fast and can handel normal variations during operation. Then, by predicting the future distribution of responses for the system and comparing it to real observed data, large disturbances can quickly be detected so that appropriate actions can be taken to mitigate these events. This allows for greater flexibility in the detection of large, catastrophic disturbances, while still being fast. On slower time scales, as more data becomes available, the system model can be updated using system identification and model selection to identify disturbances or respond to gradual changes in the system. By allowing the model describing the system to be updated, the estimation architecture gains more flexibility. Using a fully Bayesian approach based on probability distributions, this architecture robustly combines inference, event detection, uncertainty quantification, and experimental design to have a closed loop approach to learning about a system's behavior.

1.3 Markov Chain Monte Carlo Methods for Bayesian Inference

Solving Bayesian Inference Problems

Prior to the development of scientific computing, Bayesian method were restricted to inference problems where the posterior distribution could be expressed as a simple

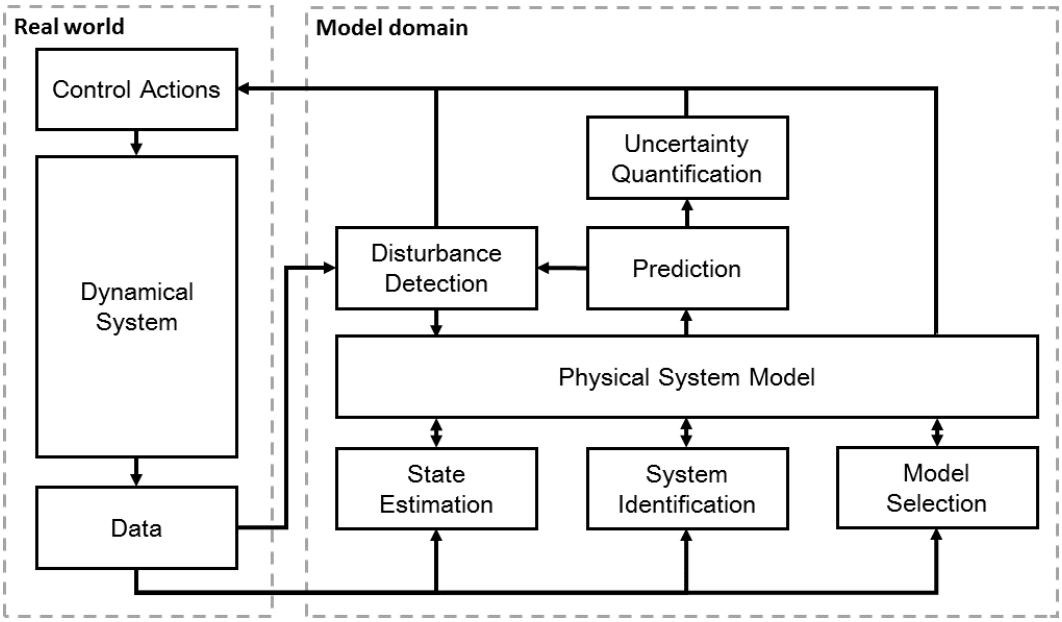

Figure 1.5: A layered Bayesian approach for fast and flexible estimation and control.

analytical distribution. While approximate methods exist, they often have difficultly handling locally identifiable or unidentifiable problems [Bec10; Gel+14], where Bayesian methods are most needed. As a result, sampling methods are commonly used. The most common family of sampling methods is Markov Chain Monte Carlo (MCMC) [Bro+11], which creates a Markov chain defined by a transition rule, or kernel, whose stationary distribution is the desired posterior. In order to make estimates accurately, the samples must discretely capture the posterior distribution in a probabilistically appropriate way. Generating these samples makes MCMC computational intensive, as often thousands to millions of model evaluations are needed to fully populate the high probability content of a complex posterior. While, by the central limit theorem, the estimate quality for the mean of a finite-variance stochastic variable scales independently of the dimension given independent samples, MCMC methods produce correlated samples, which can introduce poor high dimensional scaling. Many high dimensional problems where it is difficult to produce an efficient proposal distribution experience a “curse of dimensionality" because the sample correlation becomes very high. Thus, solving inference problems using Bayesian methods is often prohibitively expensive because sampling high dimensional distributions efficiently is challenging.

The basic idea of Monte Carlo estimation is to estimate quantities with respect to the posterior distribution using a population of samples from the posterior as follows:

$$\mathbb{E}\left[g\left(\theta\right) \mid \mathcal{D}, \mathcal{M}\right] = \int g\left(\theta\right) p\left(\theta \mid \mathcal{D}, \mathcal{M}\right) d\theta \approx \frac{1}{N} \sum_{i=1}^{N} g\left(\theta_i\right) \tag{1.12}$$

Assuming certain conditions hold, the quality of this estimate and its convergence can be assessed by the Markov chain central limit theorem [Gey11].

Markov Chain Monte Carlo

The basis for many MCMC methods is the Metropolis-Hastings algorithm, which produces a Markov chain with a desired stationary distribution, $\pi\left(\theta\right)$, by designing a transition kernel, $K\left(\theta' \mid \theta\right)$, such that the Markov chain is ergodic and reversible [Gey11; RC11]. Reversibility is a sufficient condition for the existence of a stationary distribution, $\pi\left(\theta\right)$, that satisfies the detailed-balance condition:

$$\pi\left(\theta\right) K\left(\theta' \mid \theta\right) = \pi\left(\theta'\right) K\left(\theta \mid \theta'\right) \tag{1.13}$$

This sufficient condition means that any transition kernel $K\left(\theta' \mid \theta\right)$ may be chosen to maintain the stationary distribution $\pi\left(\theta\right)$, as long as the reversibility condition (1.13) holds. Further, the composition of two kernels which have the same invariant distribution, $\pi\left(\theta\right)$, then also has $\pi\left(\theta\right)$ as its invariant distribution [Gey11]. This method can be used to create non reversible Markov chains with the correct stationary distribution out of a composition of reversible kernels.

Any proposal distribution $Q\left(\theta' \mid \theta\right)$ such that $Q\left(\theta' \mid \theta\right) \neq 0$ for $Q\left(\theta \mid \theta'\right) \neq 0$, can be used to construct such a $K\left(\theta' \mid \theta\right)$ by proposing a candidate sample θ' according to $Q\left(\theta' \mid \theta\right)$. Then the candidate, θ', is accepted with probability α given by:

$$\alpha\left(\theta' \mid \theta\right) = \min\left(1, \frac{\pi\left(\theta'\right) Q\left(\theta \mid \theta'\right)}{\pi\left(\theta\right) Q\left(\theta' \mid \theta\right)}\right) \tag{1.14}$$

If the candidate is rejected, the current sample θ is repeated. This leads to the Metropolis-Hastings algorithm:

1. Initialize the state θ_1 randomly, usually according to the prior, set $n = 1$
2. Pick a candidate state θ'_{n+1} according to the proposal $Q\left(\theta'_{n+1} \mid \theta_n\right)$

3. Accept or reject the candidate according to a sampled uniform variable ζ on $[0, 1]$:

$$\theta_{n+1} = \begin{cases} \theta'_{n+1} & \zeta \leq \alpha\left(\theta'_{n+1} \mid \theta_n\right) \\ \theta_n & \zeta > \alpha\left(\theta'_{n+1} \mid \theta_n\right) \end{cases} \tag{1.15}$$

4. Increment n and go to step 2

The evolution of the Markov chain according to the Metropolis-Hastings algorithm is illustrated in Figure 1.6. The resulting Markov chain has samples which are not independent, but correlated. Thus, the effective number of independent samples must be estimated to properly understand the convergence statistics. Based upon the Markov chain Central Limit Theorem, a standard method for judging the quality is the effective sample size (ESS) of the Markov chain $\theta_{1:N}$ defined by

$$ESS\left[\theta_{1:N}\right] = \frac{N}{1 + 2\sum_{k=1}^{N} \rho_k\left(\theta_{1:N}\right)} \tag{1.16}$$

where ρ_k is the k lag autocorrelation function [DP11]. This provides a guide for how to resample $\theta_{1:N}$ to generate a set of ESS effectively independent samples or how to incorporate the ESS into variance estimates. The effective sample size may be estimated for any function, $ESS\left[g\left(\theta_{1:N}\right)\right]$. This gives useful convergence information for the function, which is particularly relevant when evaluating variance estimates. The quality of the variance estimate is based upon the ESS of the second order moment of θ, not the ESS of θ itself.

The major challenge for the Metropolis-Hastings algorithm is designing an effective proposal distribution. The desired behavior of the Markov chain is for it to (1) converge quickly to the stationary distribution, that is, have a short burn-in time, and (2) have low correlation while sampling the stationary distribution. Optimal proposal distribution results only exist in simple cases [Ros+11; R+01].

Limitations of Metropolis-Hastings MCMC

Generally, finding a proposal distribution that escapes the “curse of dimensionality" can be difficult. Typically, for high dimensional inference problems, the high probability region of the posterior distribution lives on a low dimensional manifold in that space. This occurs when the data allows for a few parameters or the relationship between a set of parameters to be well characterized, but other parameters to remain very uncertain. The existence of this low dimensional manifold leads to the problem

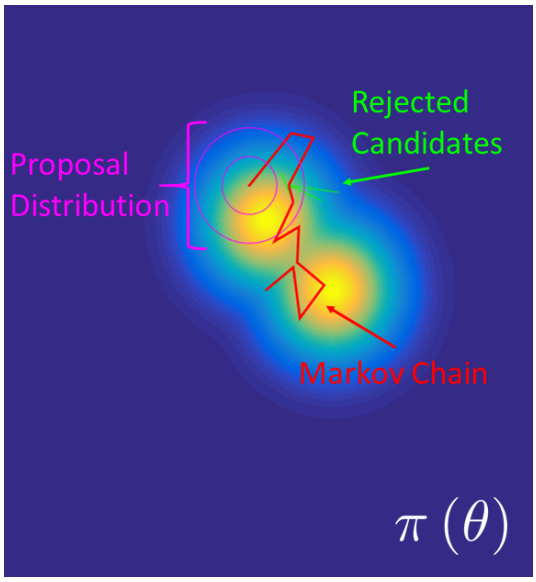

Figure 1.6: Illustration of the evolution of a Markov chain for Metropolis-Hastings MCMC.

of finding and sampling it efficiently. Even when starting on the manifold, randomly sampling the region around it without detailed knowledge of that manifold will lead to low probability samples. Thus, if the proposal distribution is ill informed, very short steps are needed to ensure high acceptance rates, which leads to highly correlated samples. These types of distributions are common in inverse problems for complex dynamical systems where the data is not sufficiently rich to detangle the complex relationships produced by the dynamics leading to unidentifiable or only locally identifiable posteriors.

Further, even for simple distributions without complicated geometry, Metropolis-Hastings MCMC requires many model evaluations to produce an effective number of samples. Practitioners often run chains for hundreds of thousands or millions of iterations to ensure they have sufficiently uncorrelated their samples. Avoiding slow mixing by developing more efficient samplers is therefore critical for solving inference problems involving PDEs or ODEs where evaluating the forward model is computationally intensive.

Finally, Metropolis-Hastings MCMC is constrained by the fact that it is based upon a Markov chain. This means it has sequential evaluation and very limited ability to adapt to its state without jeopardizing its reversibility and periodicity. The sequential updating of the chain makes MH MCMC unsuitable for high performance computing (HPC) because it cannot exploit parallelism and is difficult to adapt because it only

has local information about its current state. Solving computationally intensive inverse problems, like those for complex dynamical systems, requires being able to exploit parallelism and adaptation based upon global information.

Several methods have been proposed to address this problem; however, no method has emerged as a general solution. Population based methods like Transitional MCMC (TMCMC) [CC07] and the Asymptotically Independent Markov Sampler (AIMS) [BZ13] work well for moderately high dimensional problems where a population of samples is able to capture the global structure of the posterior. Methods based upon exploiting local structure are also used, such as Adaptive MCMC [Ros+11] and HMC [Nea11]. Adaptive MCMC methods allow the proposal distribution to change slowly over time to better exploit the local posterior distribution geometry, while still maintaining the ergodicity of the Markov chain under some conditions. This adaptation enables Adaptive MCMC to work well for high dimensional problems. HMC is able to use the local gradient structure of the posterior distribution to gain high acceptance rates. This algorithm samples along trajectories of almost constant energy, where the energy is defined as the joint probability of the posterior "potential energy" and a Gaussian momentum auxiliary variable or "kinetic energy" term. Riemannian Manifold HMC (RHMC) [GC11] further exploits local structure by making the kinetic energy position dependent [BS11]. The method introduced in [Ott+16] further extends HMC to infinite dimensional function spaces.

Contributions of this thesis

In Chapter 2, I discuss the Second-Order Langevin Monte Carlo sampler (SOL-MC) and its application to Bayesian inference. SOL-MC grew out of previous work which used the Second-Order Langevin stochastic differential equation (SDE) to approximately sample from probability distributions [TOM10; Tao11; MCF15]. Previous work also developed a Metropolized integrator, MAGLA, for this SDE [BO10; BV10] that maintains the correct invariant distribution, which we utilize. The results in this chapter are also found in [CB17]. My contribution in this work is to:

- Introduce SOL-MC for Bayesian inference
- Develop extensions of SOL-MC and MAGLA that incorporate more adaptivity to the geometry

In Chapter 3, I explore the broad class of Sequential Tempered MCMC (ST-MCMC) algorithms which incorporate annealing, importance sampling, and MCMC to gradually transform a sample population from the prior to the posterior. Previous work on the development of these algorithms and their basic theory can be found in [CC07; DDJ06; MSB13]. My contribution in this work is to:

- Develop a theory to describe under what conditions learning is possible using ST-MCMC
- Introduce a feedback controller for the proposal distribution scaling during the Metropolis step to allow better tuning
- Introduce the Rank One Modified Metropolis Algorithm (ROMMA) to better enable scaling to high dimensions

In Chapter 4, I apply Bayesian methods to develop a learning architecture for power systems, specifically studying dynamic state estimation and fault detection and classification. This project grew out of work on fault classification done in [Wie+14] and further developed in our paper [Gar+15]. My contribution in this work is to:

- Introduce an integrated Bayesian state estimation, fault detection, fault classification, and prediction architecture
- Develop a new implicit formulation of the Extended Kalman Filter for the state estimation of the differential algebraic equations describing the power system
- Formulate fault detection and classification as a Bayesian model selection problem
- Develop local and distributed methods for estimation on the power system to increase speed and robustness to disturbances

Finally, in Chapter 5, I apply Bayesian parameter estimation and model selection to the problem of the detection and identification of unknown inputs into a synthetic biological sensor. The model of the biosensor and a non-Bayesian approach to detection are presented in [Hsi+16]. For an early version of the results in this chapter see [Bae+16]. My contribution in this work is to:

- Transform the forward model of the biosensor into a suitable probabilistic model for Bayesian inference
- Formulate the detection and identification of unknown inputs as a Bayesian inference problem
- Employ ST-MCMC to provide fast solutions to this inference problem.

Chapter 2

SECOND ORDER LANGEVIN MARKOV CHAIN MONTE CARLO

In this chapter, we introduce the Second-Order Langevin Monte Carlo sampler, a MCMC algorithm which integrates many of the advantages of dynamical systems based sampling methods. The SOL-MC sampler is a based upon an auxiliary stochastic dynamical system, which enables us to study and optimize its performance using tools from the field of dynamical systems. We first provide motivation and discuss MCMC methods in general based upon auxiliary dynamical systems. Then we present a particular method based on the second-order Langevin stochastic differential equation to sample from a posterior distribution and show its relationship to the dynamical systems which underly other MCMC methods. Finally, we discuss the numerical implementation and metropolization of the SDE to ensure convergence to the correct posterior distribution, resulting in the final SOL-MC algorithm. Following the development of SOL-MC, we discuss tuning SOL-MC to optimize performance and present an example system identification problem to investigate SOL-MC under different conditions. Finally, we discuss some extensions of SOL-MC and compare these extensions to other dynamical systems based samplers using a simple test problem.

Much of the work presented here has appeared in:

[CB17] Thomas A Catanach and James L Beck. "Bayesian System Identification using Auxiliary Stochastic Dynamical Systems". In: *International Journal of Non-Linear Mechanics* (2017). DOI: 10.1016/j.ijnonlinmec.2017.03.012. URL: https://doi.org/10.1016/j.ijnonlinmec.2017.03.012.

2.1 Motivation

One of the most successful methods for sampling high dimensional distributions is Hamiltonian Monte Carlo (HMC) [Nea11; Dua+87]. HMC uses an auxiliary Hamiltonian dynamical system to propose samples far from the current sample in the parameter space but with similar probability. The position coordinates of this auxiliary system correspond to the inferred model parameters. The proposal trajectory conserves the Hamiltonian, which is related to the posterior probability. As a

result, even though the candidate samples are far from the current sample, they will have high acceptance probability, thus reducing sample correlation. This is achieved by constructing a Hamiltonian dynamical system whose potential energy function is the negative log posterior probability density function (PDF), while the kinetic energy function is quadratic in the velocity coordinates of the auxiliary system, giving the corresponding momentum vector a Gaussian distribution independent of the position coordinates in the simplest setting where the mass matrix is constant. The marginal position distribution of this system is the desired parameter posterior. An application of HMC to Bayesian updating of high-dimensional dynamic systems is given in [CB09].

This auxiliary dynamical systems approach can be extended to stochastic dynamical systems, described by a stochastic differential equation (SDE) whose stationary distribution corresponds to the posterior of the Bayesian inference problem. These SDE approaches can not only be used in a standard MCMC framework, but can also be used to approximate the distribution without Metropolis steps [WT11; CFG14; MCF15]. SDEs are an active area of research, so there is a great opportunity to leverage these results to study the properties of these algorithms, such as work in infinite-dimensional spaces [Ott+16; DS13; Bes+11]. MCMC sampling based upon the damped second-order Langevin equation was originally introduced as a generalization of HMC for molecular dynamics by [Hor91]. Recently it has also been used as a sampler for Bayesian inference [TOM10; CFG14; Ott+16]. This SDE is an effective choice because it combines Hamiltonian dynamics with an Ornstein-Uhlenbeck process, which enables the state to both follow likely trajectories and to diffuse.

We introduce a new sampling method, Second-Order Langevin Monte Carlo (SOL-MC), which extends previous work on SDE samplers, and increases its applicability to system identification problems. This sampler combines a non-metropolized SDE optimized for convergence to the posterior manifold with a metropolized SDE which can effectively sample the posterior while reducing sample correlation. We then apply results from dynamical system and control theory to tune the parameters of the SDE. We optimize the sampler's performance by balancing the influence of the "exploring" diffusion and "exploiting" Hamiltonian dynamics. Further, we are able to utilize new computational tools like automatic differentiation to make simulating this SDE tractable for system identification problems.

2.2 Dynamical Systems-based Samplers

Several dynamical systems-based methods for sampling arbitrary target probability distributions have been proposed, such as the Metropolis-adjusted Langevin Algorithm (MALA) [RS02], Hamiltonian/Hybrid Monte Carlo (HMC) [Dua+87; Nea11; GC11], and Stochastic Gradient MCMC (SG-MCMC) [WT11; CFG14; MCF15]. These methods combine a deterministic dynamic process, which encourages exploration of high probability regions, with a method to inject noise, which enables diffusion and random sampling. The goal is to maximize convergence rate and minimize correlation while sampling the stationary probability distribution.

HMC begins by creating a Hamiltonian system whose position coordinates θ correspond to the model parameters in the Bayesian inference problem and whose momentum coordinates p are auxiliary variables added to embed the dynamics. A separable Hamiltonian $H(\theta, p)$ is constructed where the potential energy $V(\theta)$ is the negative log of the target PDF, $\pi(\theta)$, and the kinetic energy $T(p)$ is quadratic in p with mass matrix M:

$$H(\theta, p) = V(\theta) + T(p) = -\log \pi(\theta) + \frac{1}{2} p^T M^{-1} p \tag{2.1}$$

The joint probability distribution $\Pi(\theta, p)$ is then given by

$$\Pi(\theta, p) \propto \exp(-H(\theta, p)) = \pi(\theta) \exp\left(-\frac{1}{2} p^T M^{-1} p\right) \tag{2.2}$$

Notice that p has a zero-mean Gaussian distribution with covariance M.

Because the joint position-momentum density (2.2) is a function of the Hamiltonian (2.1), it is conserved along trajectories for the corresponding Hamiltonian flow [Nea93; Bes+11; Sch98]. Therefore, the trajectories explore contours of constant probability even while moving far from their initial state and also have the correct invariant distribution. Posterior momentum samples can be easily generated because they are Gaussian distributed. Then a trajectory in position/parameter space is simulated for some time interval and the final position-momentum pair is accepted or rejected based upon a Metropolis step. Finally, the momentum is resampled and a new trajectory begins. This process is visualized in Figure 2.1. Ultimately, the marginal position distribution of these position-momentum (θ, p) samples is the posterior, $\pi(\theta)$.

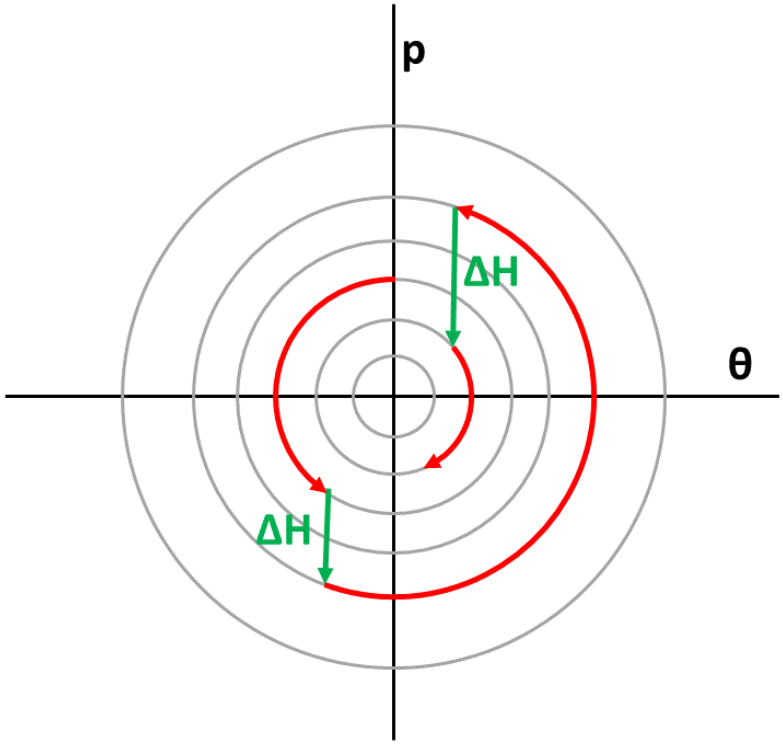

Figure 2.1: Illustration of the evolution of HMC for a 1D Gaussian posterior. The level sets of the Hamiltonian are seen in gray. The red trajectories illustrate the evolution of the system along a level set according to Hamiltonian dynamics. The green steps illustrate momentum resampling, which changes the level set.

The choice of the auxiliary mass matrix M is critical for creating a high efficiency algorithm since it guides the momentum re-sampling. This makes HMC sensitive to the auxiliary system parameter selection [BS11]. Alternatively, HMC can be extended to better adapt the momentum distribution to the local geometry through Riemannian Manifold HMC (RM-HMC) [GC11]. In RM-HMC, the fixed mass matrix M is replaced by a position dependent metric $\mathbf{G}(\theta)$, which results in a new Hamiltonian:

$$H(\theta, p) = V(\theta) + \frac{1}{2}\log|\mathbf{G}(\theta)| + \frac{1}{2}p^T\mathbf{G}(\theta)^{-1}p \tag{2.3}$$

While this Hamiltonian is non-separable, making numerical integration more challenging and requiring high-order derivatives, it often performs better than HMC for high dimension and hierarchical inference problems. The metric enables the trajectory to better follow the low dimensional, high probability manifold. The potential energy of the Hamiltonian (2.3) is the sum of $V(\theta) = -\log\pi(\theta)$ and

a term proportional to the magnitude of local curvature, $\log |\mathbf{G}(\theta)|$. Thus, when a trajectory transitions from a region of high curvature to low curvature, it gains kinetic energy so it will move faster through that region, enabling it to sample farther from the initial point. Further, the metric orients the momentum distribution to align with the curvature of the manifolds, so trajectories favor moving along the dominant directions of the manifold. Several methods have been presented for choosing $\mathbf{G}(\theta)$, based upon the Hessian. Since the metric $\mathbf{G}(\theta)$ must be positive definite, the Hessian cannot be used directly. Instead, functions of the Hessian like the non-degenerate Fisher information matrix [GC11] and the SoftAbs metric [Bet13] are used.

While HMC uses momentum resampling then deterministic dynamics to propose candidate samples, SG-MCMC and MALA create a discrete-time approximation to an underlying stochastic differential equation. A Metropolis correction can then be applied to the discretization of this SDE to ensure it has the desired posterior as its stationary distribution. MALA is based upon the Metropolis-Hastings algorithm, but incorporates a Langevin diffusion into the proposal distribution for candidate θ' [RS02], which typically gives it a shorter burn-in period to the posterior distribution than Metropolis-Hastings because the gradient term directs exploration:

$$\theta' = \theta + hM\nabla \log \pi(\theta) + \sqrt{2hM}\xi \tag{2.4}$$

M is a preconditioning matrix, h the auxiliary time step, ξ has a standard multivariate Gaussian distribution, and $\sqrt{M}$ for a positive definite matrix M indicates a matrix square root. In general, the choice of diffusion used to sample the distribution $\pi(\theta)$ is not unique. A broad class of possible diffusions if presented in [RS02].

Similarly, SG-MCMC creates a SDE whose marginal stationary distribution can be any posterior $\pi(\theta)$. This leads to a general framework for constructing a Markov process [MCF15]:

$$dz = f(z)\,dt + \sqrt{2\mathbf{D}(z)}dW(t) \tag{2.5}$$

The states, z, of the SDE are a combination of the variables θ and additional auxiliary variables. Within this framework, the SG-MCMC method is defined by the choice of a diffusion $\mathbf{D}(z)$ and drift term $f(z)$. This may be done by choosing a Hamiltonian $H(z)$, positive-semidefinte diffusion matrix $\mathbf{D}(z)$, and skew-symmetric curl matrix $\mathbf{Q}(z)$, and then choosing $f(z)$ in (2.5) as follows:

$$
\begin{aligned}
f(z) &= -\left[\mathbf{D}(z) + \mathbf{Q}(z)\right] \nabla H(z) + \Gamma(z) \\
\Gamma(z)_i &= \sum_{j=1}^{d} \frac{\partial}{\partial z_j} \left[\mathbf{D}_{ij}(z) + \mathbf{Q}_{ij}(z)\right]
\end{aligned}
\tag{2.6}
$$

with the distribution on z then given by

$$
\pi(z) \propto \exp\left(-H(z)\right) \tag{2.7}
$$

2.3 SOL-MC Sampler

Second Order Langevin SDE

Similar to the methods in Section 2.2 we choose an underlying auxiliary dynamical system to derive the second-order Langevin (SOL) Stochastic Differential Equation and define our SOL-MC sampler. To do this, we use the described framework above from [MCF15]. We choose the Euclidean Hamiltonian (2.1), and the diffusion matrix and the curl matrix are taken as:

$$
\mathbf{D}(z) = \begin{bmatrix} 0 & 0 \\ 0 & C(z) \end{bmatrix} \tag{2.8}
$$

$$
\mathbf{Q}(z) = \begin{bmatrix} 0 & -I \\ I & 0 \end{bmatrix} \tag{2.9}
$$

These choices can be interpreted as creating an SDE that combines Hamiltonian dynamics with an Ornstein-Uhlenbeck (OU) process. Solving (2.6) using these choices for $H(\theta, p)$, $\mathbf{D}$, and $\mathbf{Q}$ yields the second-order Langevin or inertial Langevin SDE:

$$
\begin{aligned}
d\theta &= M^{-1} p dt \\
dp &= -\nabla V(\theta)\, dt - C M^{-1} p dt + \sqrt{2C} dW
\end{aligned}
\tag{2.10}
$$

It was shown much earlier than [TOM10] by [Cau63] that this SDE, which models a viscously-damped nonlinear elastic dynamic system excited by white noise, has a stationary distribution defined by the Boltzmann-Gibbs distribution:

$$
\Pi(\theta, p) \propto \exp\left[-H(\theta, p)\right] = \exp\left[-\frac{1}{2} p^T M^{-1} p - V(\theta)\right] \tag{2.11}
$$

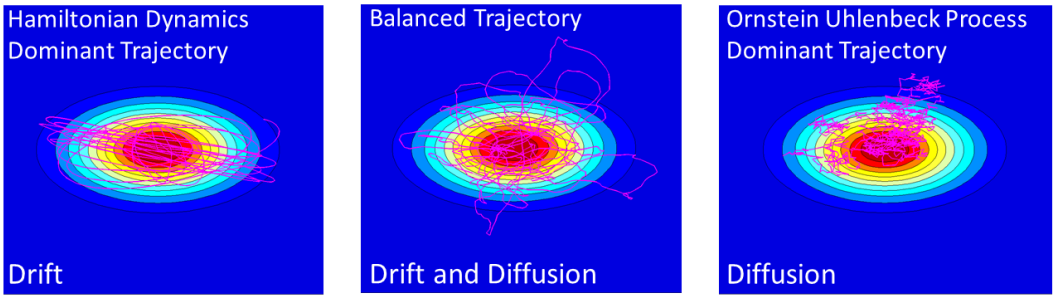

Figure 2.2: Illustration of the trade-off of choosing the damping. If there is very little damping, Hamiltonian dynamics dominate resulting in a highly correlated trajectory that can quickly explore the space. If there is high damping, the OU process dominates which results in less coherent evolution but also less exploration of the space. For optimal damping, the trajectories balances the drift and diffusion processes to explore the space without excessive correlation.

The Hamiltonian $H(\theta, p)$ is the sum of a quadratic kinetic energy term and a pseudo potential energy term $V(\theta)$:

$$V(\theta) = -\log \pi(\theta) \tag{2.12}$$

This SDE can be used to sample from an arbitrary PDF $\pi(\theta)$. To accelerate the convergence of the SDE to the appropriate stationary distribution $\pi(\theta)$, [TOM10] introduced a diffusion term using the damping matrix $C(\theta)$ in (2.10). The choice of $C(\theta)$ determines whether the Hamiltonian dynamics or the diffusive dynamics dominate the trajectory. This trade-off is illustrated by Figure 2.2. The optimal choice of damping is considered in a later sub-section. Alternatively, if we choose the Hamiltonian (2.3), instead of (2.1), as the basis for our sampler, then we get a Riemannian formulation of the SOL SDE sampler, SOL-RMC.

Numerical Integration

Now that we have found an auxiliary continuous-time dynamical system to sample from the desired posterior, we must choose an appropriate numerical implementation that combines numerical integration of the SDE and metropolized sampling of the candidates to ensure that SOL-MC samples from the correct posterior, despite numerical discretization errors. The discretization and integration of the SDE (2.10) was studied in [BO10], using a general method they call the Geometric Langevin Algorithm (GLA). The key observation is that the SDE can be split into the sum of Hamilton's equation and the Ornstein-Uhlenbeck equation. The solution to Hamil-

ton's equation is approximated using a symplectic integrator, while the solution to the Ornstein-Uhlenbeck equation is solved exactly since it is a linear equation.

The GLA solves this SDE using Strang-Type Splitting:

$$(\theta_{k+1}, p_{k+1}) = \psi_{t_k+h,t_k+h/2} \circ \Theta_h \circ \psi_{t_k+h/2,t_k} (\theta_k, p_k) \tag{2.13}$$

This is a composition of two integrators: $\psi_{t_k+h/2,t_k} (\theta, p)$, which integrates the Ornstein-Uhlenbeck process exactly from t_k to $t_k + h/2$, a half a time step, and $\Theta_h (\theta, p)$, which approximately integrates the Hamiltonian system forward by a full time step h:

$$\begin{aligned}
\hat{p}_n &= \exp\left(-\tfrac{h}{2} C_n M^{-1}\right) p_n + \sqrt{\left(I - \exp\left(-h C_n M^{-1}\right)\right) M}\hat{\xi}_n \\
p_n^{1/2} &= \hat{p}_n - \frac{h}{2}\nabla V(\theta_n) \\
\theta_{n+1} &= \theta_n + h M^{-1} p_n^{1/2} \\
\bar{p}_{n+1} &= p_n^{1/2} - \frac{h}{2}\nabla V(\theta_{n+1}) \\
p_{n+1} &= \exp\left(-\tfrac{h}{2} C_{n+1} M^{-1}\right) \bar{p}_{n+1} + \sqrt{\left(I - \exp\left(-h C_{n+1} M^{-1}\right)\right) M}\xi_n
\end{aligned} \tag{2.14}$$

We choose this 2nd-order GLA integrator because the 1st-order GLA integrator that is used in [TOM10] is not reversible and thus cannot be metropolized. It combines the exact integrator for the Ornstein-Uhlenbeck process with a symplectic Stőrmer-Verlet method for the Hamiltonian equation [BO10]. While this method is also not directly reversible, it is reversible when combined with a momentum flip. This momentum flip makes it similar to other metropolized dynamics based samplers such as HMC [BV10; Dua+87; Hor91].

Metropolization

The discretization in (2.14) for the SOL SDE in (2.10) introduces numerical errors so that (2.14) need not have the target stationary distribution. In order to ensure that the sampler exactly samples from the correct target $\pi(\theta)$, we add a Metropolis step after each integration time step. A metropolized formulation of a general GLA integrator for the inertial Langevin equation is introduced in [BV10] as the Metropolis Adjusted Geometric Langevin Algorithm (MAGLA). This metropolized formulations builds upon the original work of [Dua+87; Hor91]. The proposal $(\theta_n, p_n) \to \left(\theta^*_{n+1}, p^*_{n+1}\right)$ is defined by integrating these dynamic variables forward in time using one step of

the GLA integration algorithm (2.14). The Hamiltonian dynamics Θ_h are reversible when combined with a momentum flip. This reversibility means that

$$\begin{aligned} \left(\theta^*_{n+1}, p^*_{n+1}\right) &= \Theta_h\left(\theta_n, p_n\right) \\ \left(\theta_n, -p_n\right) &= \Theta_h\left(\theta^*_{n+1}, -p^*_{n+1}\right) \end{aligned} \tag{2.15}$$

The Metropolis acceptance step is defined by

$$\left(\theta_{n+1}, p_{n+1}\right) = \begin{cases} \left(\theta^*_{n+1}, p^*_{n+1}\right), \text{ if } \zeta_n < \alpha\left(\theta_n, p_n, \theta^*_{n+1}, p^*_{n+1}\right) \\ \left(\theta_n, -p_n\right), \text{ otherwise} \end{cases} \tag{2.16}$$

where ζ_n is a sampled uniform variable on $[0, 1]$ and the momentum is flipped if the candidate is rejected to maintain reversibility. The acceptance probability α in (2.16) is given by the Metropolis-Hastings algorithm (1.14):

$$\alpha\left(\theta_n, p_n, \theta^*_{n+1}, p^*_{n+1}\right) = \min\left\{1, \frac{Q\left(\theta_n, -p_n \mid \theta^*_{n+1}, p^*_{n+1}\right)\Pi\left(\theta^*_{n+1}, p^*_{n+1}\right)}{Q\left(\theta^*_{n+1}, -p^*_{n+1} \mid \theta_n, p_n\right)\Pi\left(\theta_n, p_n\right)}\right\} \tag{2.17}$$

where the posterior $\Pi(\theta, p)$ is defined by (2.11) while the ratio of the proposal PDFs Q is defined by

$$\begin{aligned} \frac{Q\left(\theta_n, -p_n \mid \theta^*_{n+1}, p^*_{n+1}\right)\pi\left(\theta^*_{n+1}, p^*_{n+1}\right)}{Q\left(\theta^*_{n+1}, -p^*_{n+1} \mid \theta_n, p_n\right)\pi\left(\theta_n, p_n\right)} &= \exp\left(-\Delta E\left(\theta_n, \theta^*_{n+1}\right)\right) \\ \Delta E\left(\theta_n, \theta^*_{n+1}\right) =& \frac{1}{2} D_2 L_d\left(\theta_n, \theta^*_{n+1}, h\right)^T M^{-1} D_2 L_d\left(\theta_n, \theta^*_{n+1}, h\right) + V\left(\theta^*_{n+1}\right) \\ & -\frac{1}{2} D_1 L_d\left(\theta_n, \theta^*_{n+1}, h\right)^T M^{-1} D_1 L_d\left(\theta_n, \theta^*_{n+1}, h\right) - V\left(\theta_n\right) \end{aligned} \tag{2.18}$$

While [BV10] show this result when the friction coefficient C is constant, it still holds when C is a function of θ. In (2.18), $L_d\left(\theta_n, \theta_{n+1}, h\right)$ is the discrete Lagrangian which defines the symplectic integration scheme. For the 2nd-order GLA, the derivatives of the discrete Lagrangian are

$$\begin{aligned} D_1 L_d\left(\theta_n, \theta^*_{n+1}, h\right) =& -M\frac{\theta^*_{n+1} - \theta_n}{h} - \frac{h}{2}\nabla V\left(\theta_n\right) \\ D_2 L_d\left(\theta_n, \theta^*_{n+1}, h\right) =& M\frac{\theta^*_{n+1} - \theta_n}{h} - \frac{h}{2}\nabla V\left(\theta^*_{n+1}\right) \end{aligned} \tag{2.19}$$

Optimization and Tuning of SOL-MC

When implementing SOL-MC prescribed by (2.10), we have the freedom to choose the auxiliary mass matrix M and damping matrix C of the SDE. While designing the sampler, our goal is to speed up convergence to the stationary distribution and minimize correlations once the stationary distribution is reached to keep the effective sample size, ESS in (1.16), as large as possible. The first step toward choosing these parameters is to study the behavior of SOL-MC when sampling a Gaussian distribution, because the SDE for sampling the Gaussian can be solved analytically.

Gaussian Linear System

When the target distribution is a zero-mean Gaussian, it is straightforward to prove several optimality results. The Hamiltonian for this system is quadratic:

$$H(\theta, p) = \tfrac{1}{2} p^T \mathbf{M}^{-1} p + \tfrac{1}{2} \theta^T \mathbf{G} \theta \tag{2.20}$$

Thus, the SDE reduces to a stochastic linear system, which yields the continuous time dynamical system of the form $\dot{z} = \mathbf{A}z + \mathbf{b}w$:

$$\begin{bmatrix} \dot{\theta} \\ \dot{p} \end{bmatrix} = \begin{bmatrix} 0 & \mathbf{M}^{-1} \\ -\mathbf{G} & -\mathbf{C}\mathbf{M}^{-1} \end{bmatrix} \begin{bmatrix} \theta \\ p \end{bmatrix} + \begin{bmatrix} 0 \\ \sqrt{2\mathbf{C}(\theta)} \end{bmatrix} w \tag{2.21}$$

Mass Matrix

The Gaussian potential in the Hamiltonian (2.20) has constant curvature, where $\mathbf{G}(\theta) = \mathbf{G} = \Sigma^{-1} \succ 0$. As discussed in Section 2.2 for HMC and RM-HMC, by choosing $\mathbf{M}$, the mass matrix for the auxiliary system, to be $\mathbf{G}$, the momentum distribution will align with the dominate directions of the Gaussian distribution. This reduces sample correlation. For the pseudo potential energy of a general posterior distribution, we locally approximate the system with a quadratic potential by taking the Hessian matrix of the potential, $\frac{\partial^2 V(\theta)}{\partial \theta^2}$. This a real symmetric matrix so it has a spectral decomposition in terms of the real eigenvalues Λ and orthonormal eigenvectors Q:

$$\begin{aligned} \mathbf{H}(\theta) =& \frac{\partial^2 V(\theta)}{\partial \theta^2} \\ =& Q \Lambda Q^T \end{aligned} \tag{2.22}$$

The Hessian is not guaranteed to be positive definite, and so it must be transformed to be a suitable choice for $\mathbf{M}$. We choose to use the SotfAbs metric [Bet13] $\tilde{\mathbf{H}}$. This choice smoothly transforms the small and negative eigenvalues of the matrix $\mathbf{H}$ using a smoothing parameter ϵ. Small eigenvalues are mapped close to ϵ and large negative eigenvalues λ are mapped to $-\lambda$. The transformation is

$$\begin{aligned} \mathbf{M} =& \tilde{\mathbf{H}}(\theta) \\ =& Q\Lambda \coth\left[\epsilon^{-1}\Lambda\right] Q^T \end{aligned} \tag{2.23}$$

When choosing ϵ, the smaller the ϵ, the flatter the quadratic estimate of the potential can be thus enabling the sampler to take larger steps in the flat directions, making generally small ϵ preferred. However, two factors keep ϵ from becoming too small. First, the long steps will cause numerical integration to be less accurate particularly when the problem is non-linear. Second, the prior distribution often introduces bounds for parameter values. Thus, if the posterior is flat, but bounded, and ϵ is small, the trajectories will often overshoot the boundary, causing a high rejection rate.

Since we are not concerned with maintaining detailed-balance while the SDE is converging to the stationary distribution during the burn-in period, we can adapt $\mathbf{M}$ to reflect the local geometry via (2.23). Once we judge that we have reached the stationary distribution based upon the statistics of multiple chains, $\mathbf{M}$ is fixed to reflect the local curvature of the stationary distribution.

Damping Matrix

For the linear stochastic dynamical system (2.21) with $\mathbf{M} = \mathbf{I}$, the damping $\mathbf{C}(\theta)$ is chosen in [TOM10] by minimizing the maximum real part of the eigenvalues of the dynamics matrix $\mathbf{A}$. This choice corresponds to critical damping, which maximizes the convergence to the stationary distribution and thus minimizes the burn-in period. They find that $\mathbf{C}(\theta) = 2\sqrt{\mathbf{G}}$. More generally, it can be shown that when $\mathbf{M}$, $\mathbf{C}$, $\mathbf{G}$ are simultaneously diagonalizable, $\mathbf{C}(\theta) = 2\sqrt{\mathbf{GM}}$. The eigenvalues of $\mathbf{A}$ in (2.21) are given by

$$\begin{aligned} \det(\mathbf{A} - \lambda\mathbf{I}) &= \det\left(\lambda\mathbf{C}\mathbf{M}^{-1} + \lambda^2 + \mathbf{M}^{-1}\mathbf{G}\right) \\ &= \det\left(\lambda\Lambda_C\Lambda_M^{-1} + \lambda^2 + \Lambda_M^{-1}\Lambda_G\right) \\ &= 0 \end{aligned} \tag{2.24}$$

where Λ_M, Λ_C, and Λ_G are the eigenvalues of $\mathbf{M}$, $\mathbf{C}$, and $\mathbf{G}$, respectively, expressed as diagonal matrices. Since the system is now diagonal, we can see that in general the eigenvalues λ_i take the form of

$$\lambda = -\frac{\lambda_{C,i}}{2\lambda_{M,i}} \pm \frac{1}{2}\sqrt{\frac{\lambda_{C,i}^2}{\lambda_{M,i}^2} - 4\frac{\lambda_{G,i}}{\lambda_{M,i}}} \tag{2.25}$$

The largest eigenvalue is then minimized when $\lambda_{C,i} = 2\sqrt{\lambda_{G,i}\lambda_{M,i}}$ so $\mathbf{C}(\theta) = 2\sqrt{\mathbf{GM}}$. For a more general case, when $\mathbf{M}$ and $\mathbf{G}$ do not necessarily commute, this choice of parameters will not necessarily be positive definite. Thus, we choose the positive definite matrix $2\sqrt{\mathbf{M}^{1/2}\mathbf{G}\mathbf{M}^{1/2}}$ for a general $\mathbf{M}$ and $\mathbf{G}$. When the dynamic system is non-linear, we linearize it by taking the Hessian matrix of the potential, $\frac{\partial^2 V(\theta)}{\partial\theta^2}$. We then transform this using (2.23) into a positive definite matrix, $\tilde{\mathbf{H}}$, to capture the curvature of the potential surface, giving it the ability to adapt to the local geometry. This is the same as locally fitting a Gaussian distribution to the posterior and using that linear system to design the damping. Ultimately this yields

$$\mathbf{C}(\theta) = 2\sqrt{\mathbf{M}^{1/2}\tilde{\mathbf{H}}(\theta)\,\mathbf{M}^{1/2}} \tag{2.26}$$

Convergence and Correlations

As shown in the previous section, minimizing the largest eigenvalue of the linear stochastic dynamical system (2.21), when $\mathbf{M}$ and $\mathbf{G}$ commute, maximizes the convergence rate since the system is critically damped. Furthermore, this also minimizes the autocorrelation function for this system. The general stochastic linear system $\dot{z} = \mathbf{A}z + \mathbf{b}w$ has the solution:

$$z(t) = \exp[\mathbf{A}t]\,z(0) + \int_0^t \exp[\mathbf{A}(t-s)]\,\mathbf{b}dw(s) \tag{2.27}$$

Let the dynamical system be converged to its stationary distribution and $z(0)$ be its state. Then assuming the stationary distribution is a zero-mean Gaussian, with covariance Σ, the covariances between $z(t)$ and $z(0)$ is:

$$cov[z(t), z(0)] = \exp[\mathbf{A}t]\,\Sigma \tag{2.28}$$

Using the assumption that the matrices $\mathbf{M}$, $\mathbf{C}$, $\mathbf{G}$ are simultaneously diagonalizable, we can look at each of the ith position-momentum pairs in the diagonalize

coordinate system individually. The dynamics of each of these 2D systems is described in [Per13]. The matrix exponential, $\exp[\mathbf{A}t]$, decays at a rate proportional to $\exp(Re[\lambda]t)$ and has a known form which depends on whether the eigenvalues are both real, both complex, or identical. Minimizing the maximum real part of the eigenvalues in (2.25) causes this decay rate to be fastest, which minimizes the autocorrelation over the trajectory. Thus the effective sample size of the MCMC sampler defined by this SDE will be highest.

Computational Considerations

Dynamical systems based sampling methods require taking derivatives of the likelihood function. Taking the gradient is unavoidable since it produces the underlying dynamics. Computing the Hessian is useful for capturing the correlation structure and the appropriate length scales. Traditional techniques like symbolic derivative solvers or finite differences do not scale, are computationally intensive, and often imprecise. Automatic Differentiation (AD) presents an alternative approach that typically outperforms these traditional methods. AD analyzes code as it compiles and computes the derivative based upon the underlying elementary operations being executed on the machine level using the chain rule. Since it takes derivatives of the computer code itself, it can take derivatives of much more complicated likelihood functions than symbolic methods. While AD libraries have been developed for a variety of languages, Julia, a language specifically designed for scientific computing is especially powerful [Bez+12; RLP16]. Julia looks like a scripting language but is "just-in-time" compiled, thus giving it the speed of a compiled language like C. We choose Julia for implementing SOL-MC.

As an example, for the shear building model in the next section, the likelihood function takes approximately 0.2 second to evaluate and has up to 12 parameters. Using AD, the gradient computation takes 1.8 seconds and the Hessian takes around 9.3 seconds. When using finite differences, the gradient takes 5.8 seconds and the Hessian 150 seconds. Thus we see significant computational cost savings.

2.4 System Identification Example

Hysteretic Structure Model

In order to evaluate the performance of SOL-MC, we consider the system identification problem of identifying the parameters of a three-story hysteretic building model from seismic response data. Depending upon the parameters, prior, and input, this

model can produce globally identifiable or unidentifiable cases. For the globally identifiable case, we show that SOL-MC is able to effectively find and sample the posterior efficiently. For the unidentifiable case, we show that SOL-MC can still sample the posterior, but samples with less efficiency than in the globally identifiable case.

System Identification Problem

We consider the problem of identifying unknown parameters for the three-story Masing shear building model where the building is excited by earthquake ground motion as in [MB08]. We used ten seconds from the Sylmar ground motion record at the Olive View Hospital parking lot recorded during the 1994 Northridge earthquake and available from the Strong-Motion Virtual Data Center [Eng]. This acceleration time history is sampled at a 50 Hz. The mass distribution for the structure can be estimated from structural drawings much more accurately than the other structural model parameters, and so we assume that the mass of each story is known and it is 1.25×10^5 kg. The pseudo-static influence vector b is known to be $[1\ 1\ 1]^T$. The small-amplitude inter-story stiffness K_i is unknown but has a true value of 2.5×10^8 N/m for each story when simulating the synthetic response data. The ultimate strength $r_{u,i}$ is unknown for each story, but has an actual value of 1.75×10^6 N. The smoothness parameter α_i is also unknown, but it is assumed to have the same value α for each story. Its true value is 4. We assume that the viscous damping matrix C takes the form of a Rayleigh damping matrix:

$$C = c_m M + c_k K \tag{2.29}$$

where the coefficients are unknown, but have true values of 0.293 for c_m and 2.64×10^{-4} for c_k.

The output $y(t) = \ddot{x}(t) + \nu(t)$ is the acceleration of each story sampled at 50 Hz, yielding 500 samples in the trajectory for each of the three stories. The output is subject to Gaussian additive noise, $\nu(t)$, with mean zero and standard deviation σ. The value of σ is unknown, but has a true value of 0.5 m/s^2. Ultimately, this yields a likelihood function for the output data $\mathcal{D}$ of the form:

$$p(\mathcal{D} \mid \theta, \sigma) = \left(2\pi\sigma^2\right)^{\frac{-N_d N_t}{2}} \exp\left[-\frac{1}{2\sigma^2} \sum_{i=1}^{N_d} \sum_{t=1}^{N_t} \left(y_t^{(i)}(\theta) - \hat{y}_t^{(i)}\right)^2\right] \tag{2.30}$$

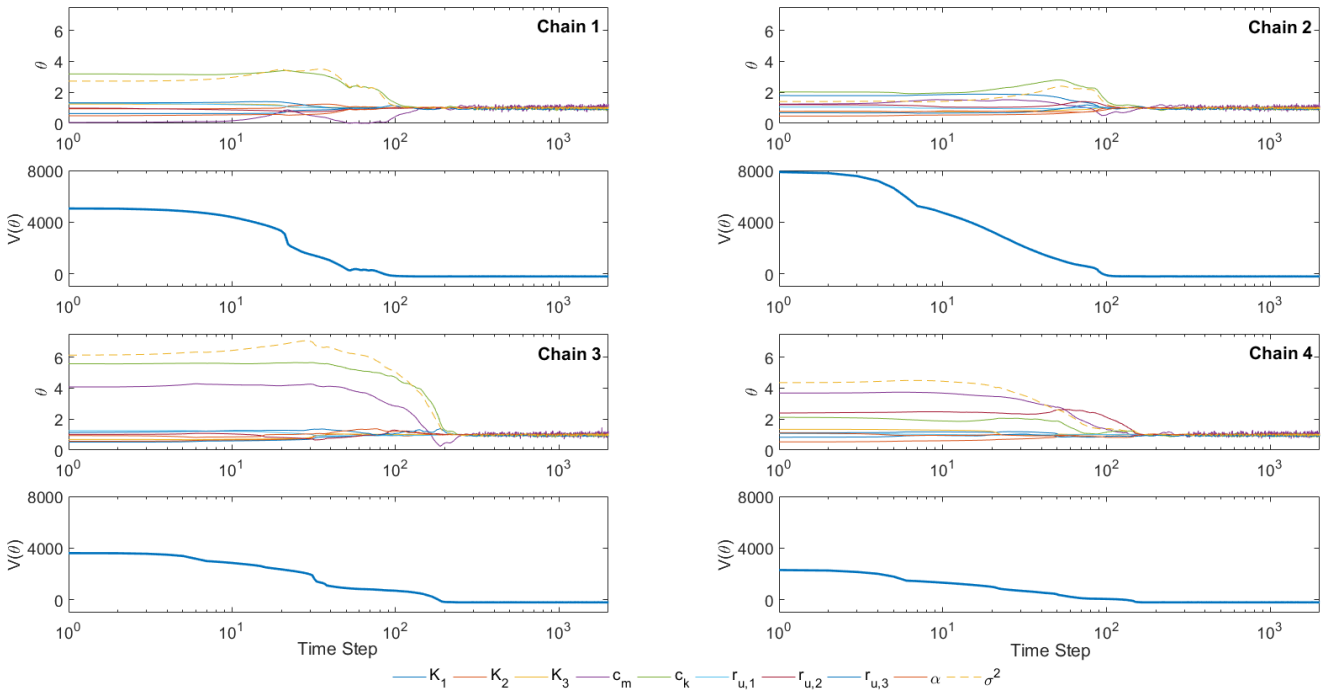

Figure 2.3: Position and potential energy trajectories of the four chains.

where the ten unknown parameters are $\theta = \left[k_1, k_2, k_3, c_m, c_k, r_{u,1}, r_{u,2}, r_{u,3}, \alpha, \sigma\right]^T$, $\hat{y}_t^{(i)}$ is the measured output at time step t of story i and $y_t^{(i)}(\theta)$ is the simulated expected output given parameters θ.

The prior distributions on the parameters are chosen as follows: K_i has a log normal prior with logarithmic mean $\log\left(2.5 \times 10^8\right)$ and logarithmic standard deviation 0.5; $r_{u,i}$ also has a log normal prior with logarithmic mean $\log\left(2.5 \times 10^8\right)$ and logarithmic standard deviation 0.5; α has a log normal prior with logarithmic mean $\log(4.0)$ and logarithmic standard deviation 0.5; σ^2 has a uniformly distributed prior between 0 and 3; c_m has a uniformly distributed prior between 0 and 1.5; and c_k has a uniformly distributed prior between 0 and 1.5×10^{-3}.

We implemented this model using MATLAB and Julia using the Newmark-beta method for numerical integration. Intermediate shorter time steps were added between measurements to increase accuracy and stability, particulary during velocity reversals and loop closures. The input was held constant during each intermediate integration period. The implementation of the extended Masing rules were reduced to a set of conditional statements, as described in [Thy89].

Results: Globally Identifiable Case

We began by simulating several Markov chains to estimate when the SDE reached stationary state. During this initial burn-in phase, 300 steps were simulated where the SDE was solved without Metropolis correction to speed up convergence. The steady state can be identified once the likelihood function stops changing significantly and the chains appear to be sampling from the same region. Four chains were selected based upon their posterior likelihood and they were used to simulate 2000 samples from the posterior using the metropolized sampler. The trajectory of the samples and the log posterior can be seen in Figure 2.3. We see that they converge to the stationary distribution in under 300 iterations and then begin sampling from it. The burn-in period was determined by inspecting the convergence of the log posterior function for the chains before the SDE is metropolized. During the burn-in period, the time step was set to 0.1 and the mass matrix M was adapted to the local geometry. Once the burn-in period ended, the time step was set to 1.0 and the mass matrix fixed to ensure that the chain had the correct stationary distribution for the sampling. The potential surface in this example is generally very non-smooth, and so a smaller time step is needed to converge faster to the high probability manifold. Once the trajectory finds the high probability manifold, a larger step can be taken since it is typically much smoother.

The trajectories of many sample chains during a burn-in period can be seen in Figures 2.4 and 2.5. For the projection onto the σ^2 and α plane given in Figure 2.4, we see that the trajectories generally converge to the same funnel like path as they move toward the stationary distribution at the bottom near (1,1). At early time steps, the trajectories increase the measurement variance σ^2 since the data fit is so poor. As σ^2 becomes very large, this effectively anneals the problem, allowing the trajectory to explore more space without getting stuck in a local minimum. As the data fit becomes better, σ^2 decreases constraining the trajectories, and forces them to converge to the high probability manifold. This works well since the model is globally identifiable.

In contrast, Figure 2.5 shows the projection onto the σ^2 and $r_{u,3}$ plane, where there is not a single path along which trajectories converge to the high probability manifold. We still observe a similar annealing behavior, where σ^2 increases when the data fit is poor then decreases as the trajectory hones in on the true values. However, some trajectories decrease in variance before they converge to the high probability region. This is because the posterior for $r_{u,3}$ has high variance, so it is flatter with longer

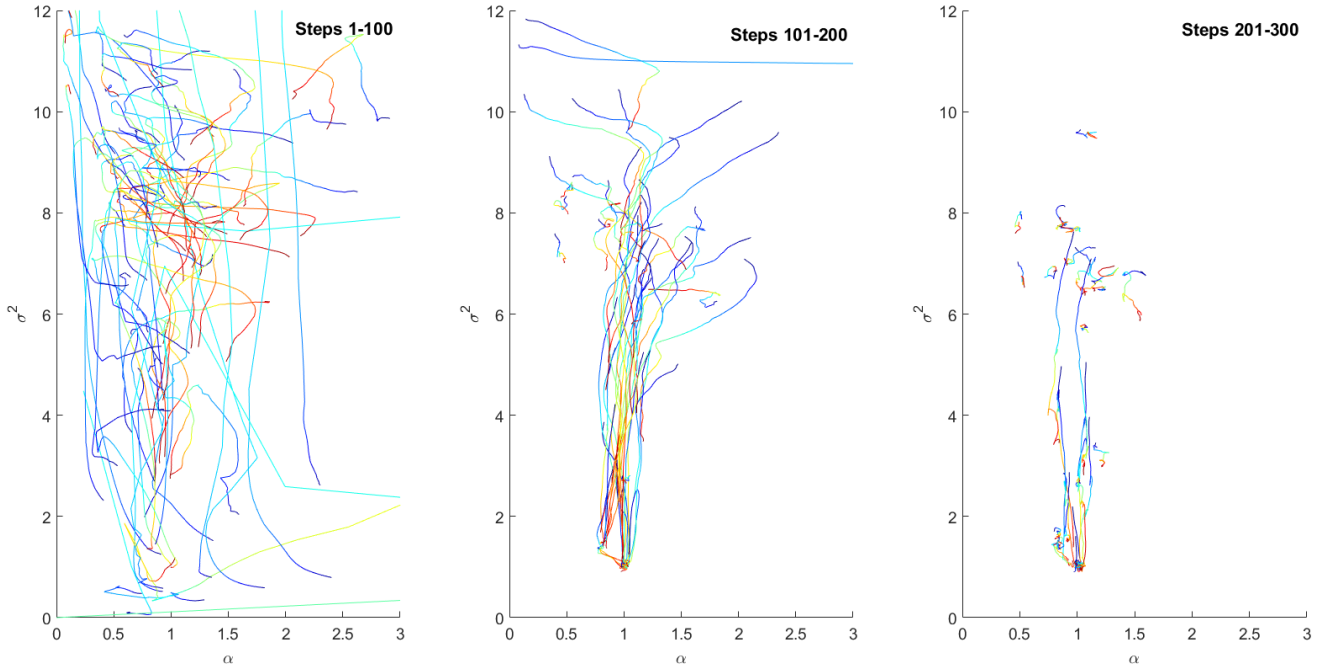

Figure 2.4: Sample trajectories projected onto the σ^2 and α plane where the color indicates the evolution in time moving from blue to red.

tails, as the model is close to unidentifiable. This can present additional challenges to the sampler.

Table 2.1 summarizes our results for four Markov chains sampling the posterior. We see that each chain produces similar effective sample sizes. For the 8000 total posterior samples generated by the four chains after their burn-in periods, we get at least 1745 effectively independent samples for each parameter. The sample mean and standard deviation are computed across the chains. This is done by weighting the contribution from each chain by its effective sample size. The results presented in Table 2.1 are scaled relative to the true value for easier comparison. These values are in good agreement with those presented in [MB08]. Further, we can also see in the last row that the effective number of samples for the energy, $H(\theta, p)$, is high. The energy ESS is a good approximation for the overall quality of variance estimates when the posterior distribution is nearly Gaussian because $H(\theta, p)$ then is almost quadratic. This means that the ESS of $H(\theta, p)$ approximates the ESS of the sum of the scaled and de-correlated second order moments of θ and p.

Figure 2.6 graphically shows the distribution of the posterior samples and their correlation diagrams. Each color corresponds to samples from a single chain. We see that the chains sample from the same region and do not seem to be biased, which

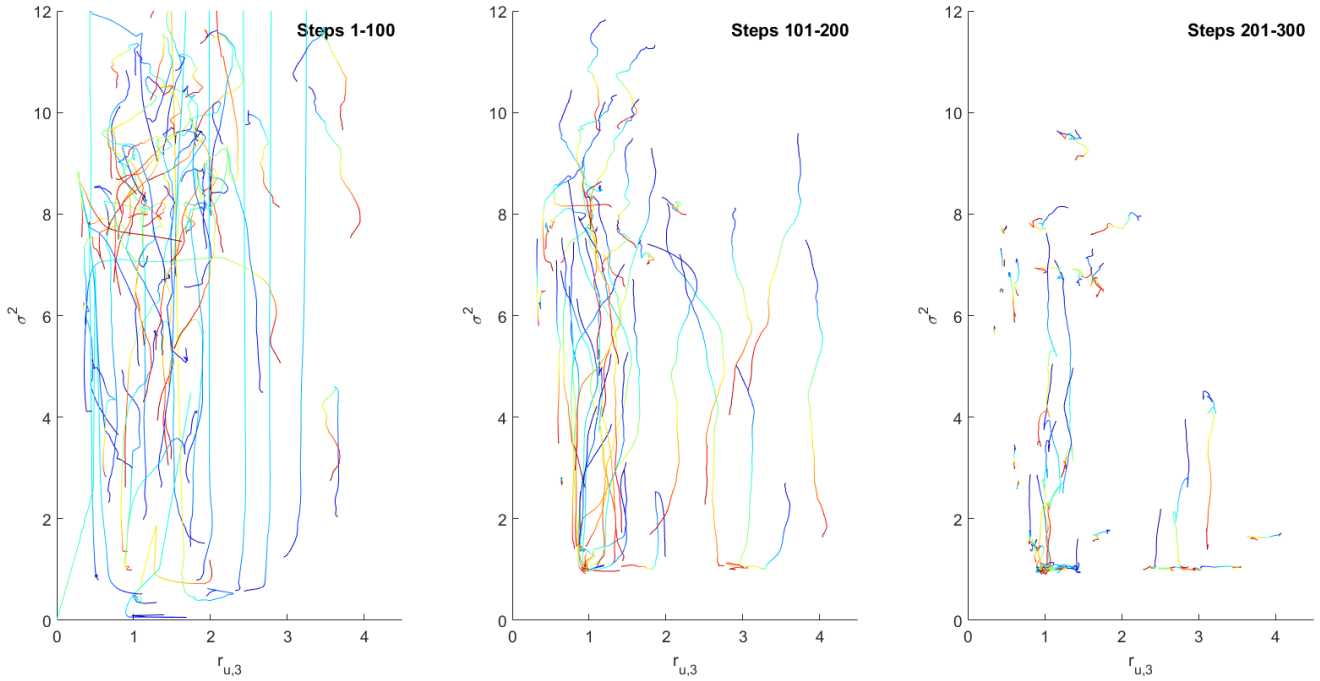

Figure 2.5: Sample trajectories projected onto the σ^2 and $r_{u,3}$ plane where the color indicates the evolution in time moving from blue to red.

is indicative of good convergence. Several of the variables exhibit high correlations, but are all globally identifiable. The correlation structure agrees with the results from [MB08] and what is expected from the structure of the model.

Next, we study the influence of the integration time step and the choice of damping matrix in sampling the posterior. We start chains at the true value of the parameters and sample the manifold using different integration time steps h and weights on the damping matrix. The damping matrix takes the form of $C = \frac{1}{\gamma}\bar{C}$, where $\bar{C}$ is the nominal damping matrix. We see the results for the effective sample size estimates when using different time steps and damping weights in Figure 2.7. As the integration time step increases, the ESS generally increases until it becomes too large and the proposed samples overshoot and leave the high probability region causing them to be rejected. The energy ESS peaks at a smaller time step than the state ESS because the larger the time step, the more it biases the kinetic energy to be smaller because higher kinetic energy proposals will similarly overshoot. Further, choosing small γ corresponds to a large degree of damping and noise. This choice reduces the energy correlation but increases the state correlation since the SDE has more of a random walk behavior. Large γ means that there is weak damping so the Hamiltonian dynamics dominate. Thus the energy is largely conserved along

Variable	ESS 1	ESS 2	ESS 3	ESS 4	ESS Total	Scaled Mean	Std
k_1	562.29	668.37	696.92	732.5	2660.1	1.006	0.0026
k_2	684.27	851.43	628.74	810.05	2974.5	0.994	0.0023
k_3	756.97	739.64	396.03	667.67	2560.3	1.003	0.002
c_m	520.49	505.23	562.53	555.55	2143.8	1.019	0.088
c_k	682.44	733.7	560.36	679.05	2655.6	0.955	0.044
$r_{u,1}$	501.49	562.45	545.26	527.38	2136.6	0.998	0.0038
$r_{u,2}$	492.37	433.89	749.63	757.86	2433.8	0.993	0.0091
$r_{u,3}$	595.76	415.89	204.19	529.31	1745.2	0.933	0.038
α	776.55	567.39	767.25	835.85	2947	1.018	0.015
σ^2	941.29	687.03	365.97	581.27	2575.6	0.981	0.033
$H(\theta, p)$	774.42	763.9	839.61	758	3135.9		

Table 2.1: Summary of the statistics for the four Markov Chains describing the effective sample size of each chain of length 2000 samples, total effective number of samples out of 8000 samples, mean estimate for each variable scaled relative to the true value, and the standard deviation.

the trajectory, which decreases the energy ESS. However, the trajectory can better explore the whole posterior and take larger steps with a high likelihood of acceptance, which increases the state ESS. From Figure 2.7, we see that the sampling efficiency appears to be more sensitive to the choice of time step than to the damping weight γ and that for this problem, these parameters influence the ESS fairly independently. Moreover, when choosing these parameters, it is important to balance the sample and energy ESS to ensure high quality parameter estimates of the means and variances.

Results: Unidentifiable Case

When the smoothing parameter α_i is not constrained to be the same for all stories, the model parameters α_3 and $r_{u,3}$ for the third story become unidentifiable because there is little yielding for that story under the excitation. This makes sampling the highly non-Gaussian manifold much more difficult since the SDE is optimized to sample Gaussian like distributions. Many Markov chains are simulated with an initial 300 burn-in samples to find the high likelihood region of the posterior distribution. Then ten chains were selected based upon their posterior likelihood and used to sample the posterior for 2000 steps. The chains were able to converge to the stationary distribution and sample the high probability content of the posterior. The chains sample the posterior less efficiently and with somewhat more bias and variability than in the globally identifiable case.

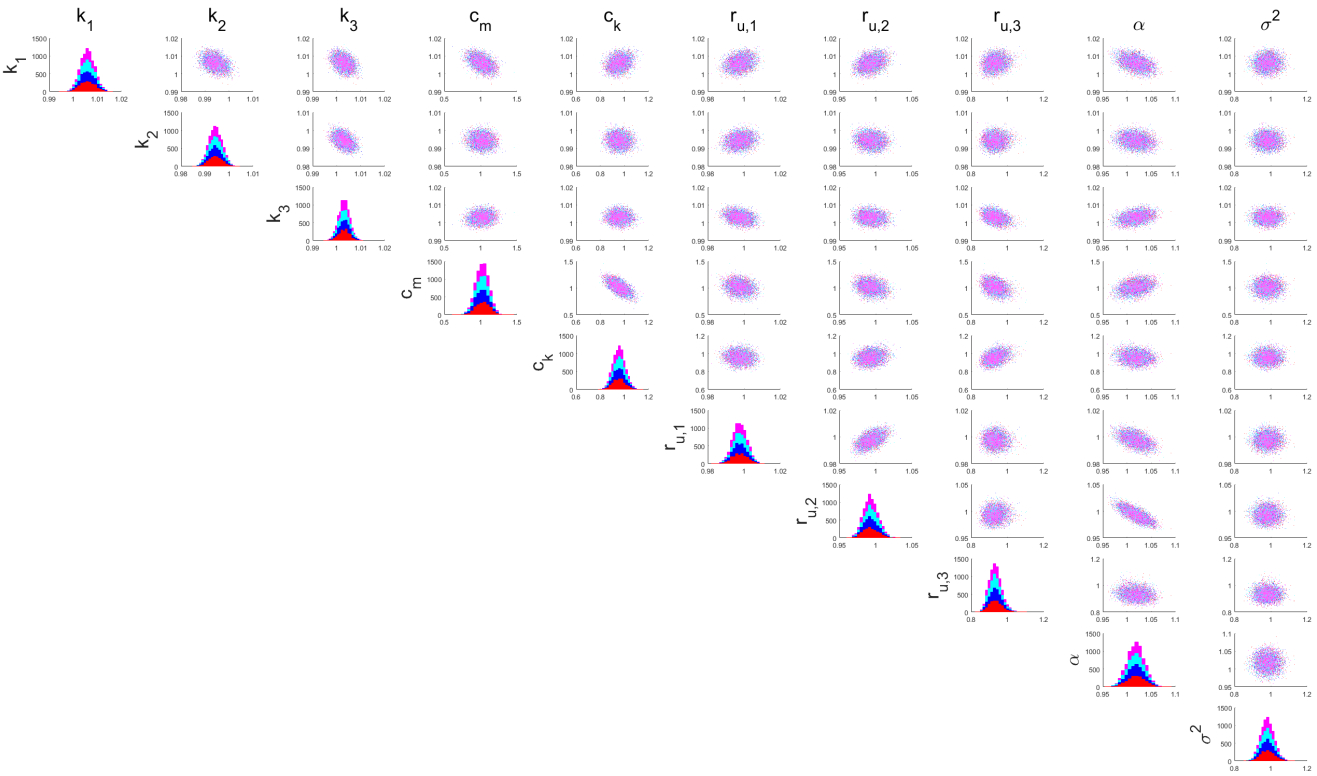

Figure 2.6: Samples showing parameter distributions and correlation for the globally identifiable case using SOL-MC.

Figure 2.8 shows the histograms for the estimated parameters for the ten different chains. We can see that the samples appear to generally capture the same region of the posterior; however, the red and green chains can be seen to over-sample some of the bins. This is indicative of the chains getting stuck temporarily and having a very high rejection rate for a few parameter values. This behavior will cause those chains to have a small ESS, so when posterior estimates are made, those chains carry less weight, leading to unbiased results. This variability is caused by the different chains having different initial mass matrices corresponding to their starting point from sampling the manifold.

Figure 2.9 graphically shows the correlation diagrams for the ten chains sampling this distribution. Since each chain converged to a different point on the manifold, each with different local geometry, their auxiliary mass matrices are different, causing them to sample the posterior differently. However, they still generally cover the same region and appear to be largely unbiased. We can see that the most challenging aspect of the posterior distribution is the relationship between α_3 and $r_{u,3}$, which has a very non-gaussian banana shape, and this leads to a lower ESS compared to the other

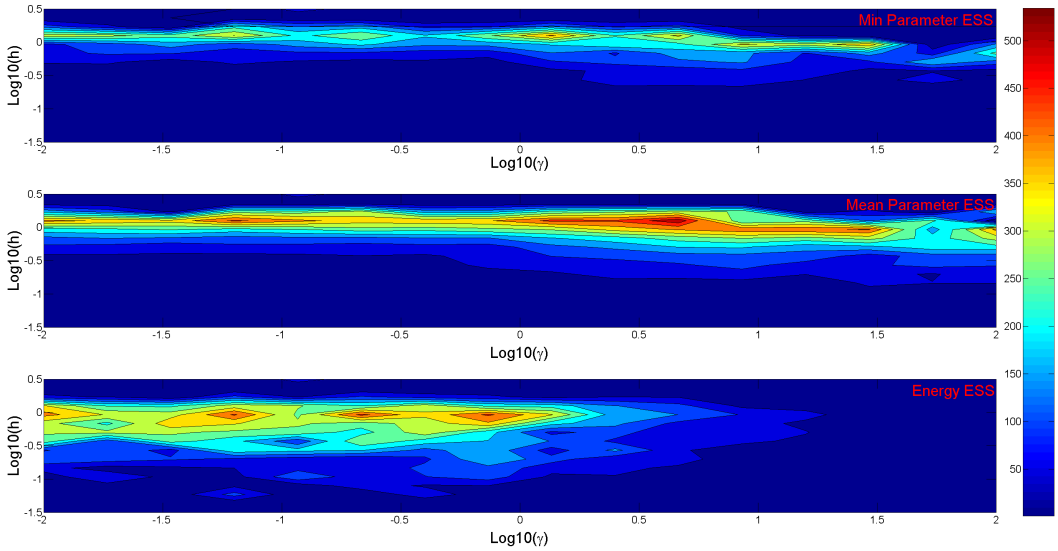

Figure 2.7: Plots of the minimum effective sample size for 1000 samples over all parameters, mean ESS over all parameters, and energy ESS for different sampler integration time steps and damping weight, $\frac{1}{\gamma}$.

parameters. Table 2.2 summarizes these results. We see that chains 6 and 10 have a very low ESS, which corresponds to the chains getting stuck, as seen in Figure 2.8.

2.5 Extensions

Riemannian SOL-MC

If the Riemannian Hamiltonian (2.3) was used to construct the sampler based upon the same method described in Section 2.3, the sampler would be the SOL-MC equivalent of Riemannian Manifold HMC. Using the same numerical integration techniques employed by RM-HMC for the hamiltonian integration part of the Strang-Type splitting, Equation (2.13), a reversible proposal can be formed. Then a similar Metropolis step can be used to accept or reject the proposed sample in the SDE, as in the normal SOL-MC algorithm. Like RM-HMC, using Riemannian SOL-MC enables the sampler to better adapt to the local geometry of the posterior distribution, thus increasing effective sample size. This comes at the cost of having to take higher-order derivatives of the likelihood function.

Non-Reversible SOL-MC via Resampling

Constructing a non-reversible kernel enables HMC to sample unidentifiable posteriors well. By adapting these methods for SOL-MC, we can also increase its performance. By periodically resampling the momentum of the dynamical system

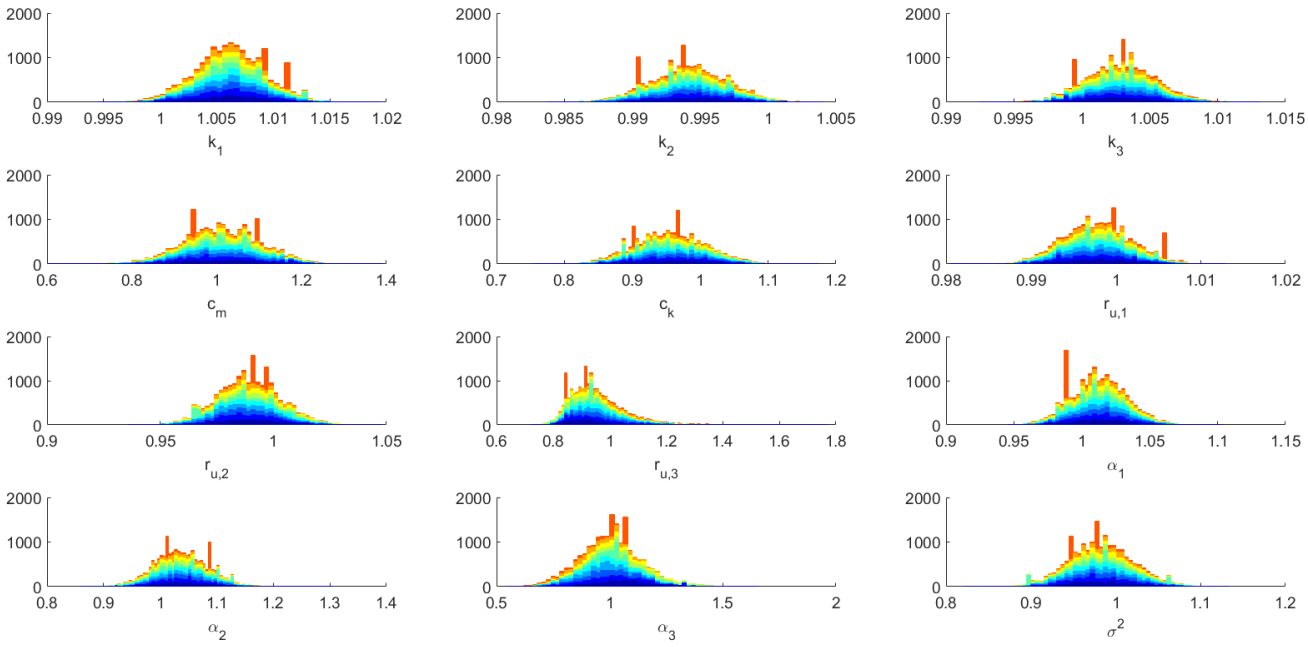

Figure 2.8: Samples showing parameter distributions for the unidentifiable case using SOL-MC. The colors represent the ten chains.

from its stationary distribution, the Markov chain will more quickly decorrelate. Since the composition of operations that maintain the invariant distribution of the Markov chain will still maintain the correct invariant distribution, this momentum refreshment will not effect the stationary distribution. In effect, this transforms the SOL-MC sampler from following the trajectory of a single dynamical system to a sampler that follows a trajectory for a number of steps, then after resampling the momentum, it follows a new trajectory. By studying the behavior of the linear, Gaussian problem, this type of non-reversible momentum resampling can increase the robustness of SOL-HMC over HMC. This can be seen in Figure 2.10. When the mass matrix of HMC and SOL-MC does not perfectly align with the covariance structure of the Gaussian they are sampling, HMC's performance can suffer and it can become sensitive to the choice of timestep. SOL-MC is less affected by this misalignment due to the robustness added by the stochasticity.

Example: Bayesian Logistic Regression

In order to evaluate the potential effectiveness of these extensions to SOL-MC, they are tested against HMC and RM-HMC while solving the Ripley Bayesian Logistic Regression test problem in seven dimensions [Rip07]. This test problem was used to evaluate RM-HMC in [GC11]. One million posterior samples were generated

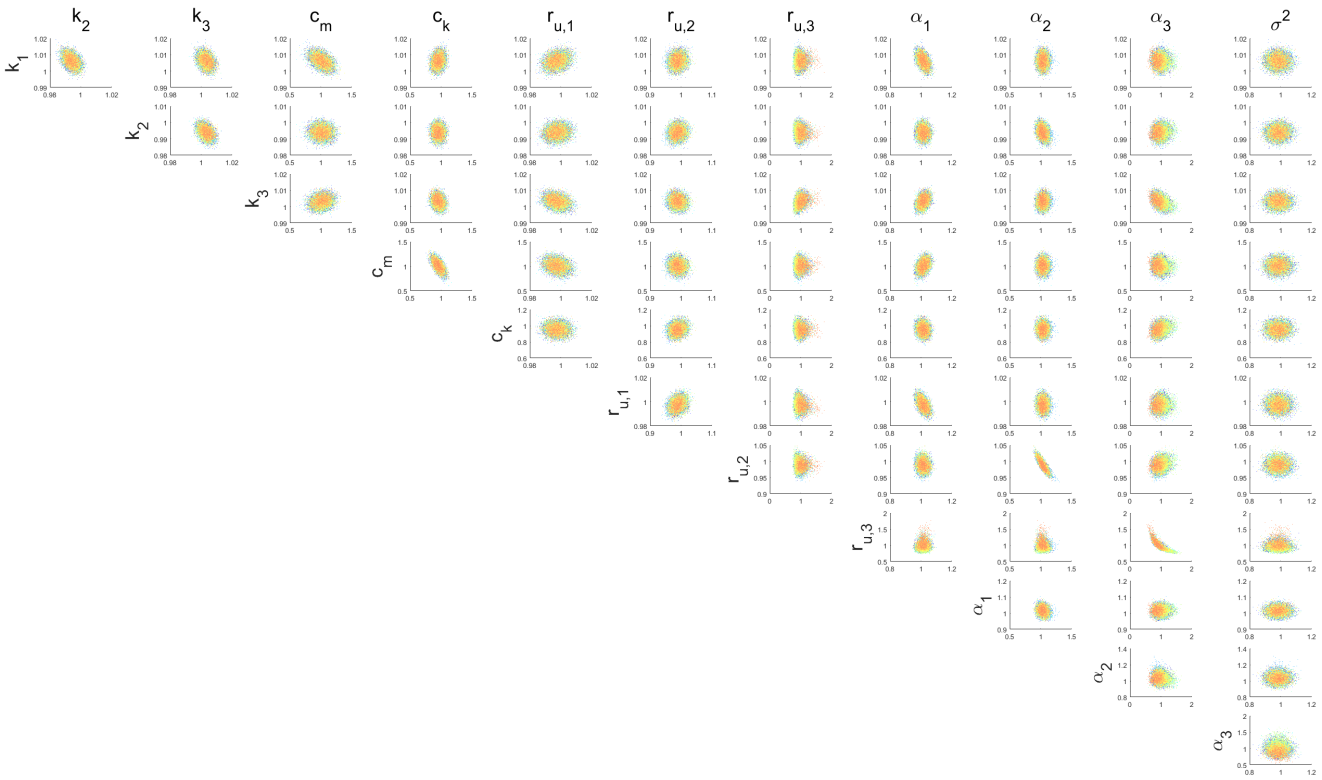

Figure 2.9: Samples showing parameter correlations for the unidentifiable case using SOL-MC. The colors represent the ten chains.

using a standard RM-HMC algorithm to serve as a baseline for estimating the error in the mean estimate and error in the covariance estimate of the posterior, quantified using the Frobenius norm. The performance space of each algorithm was explored by evaluating it at many different parameter values to see the range of possible performance. For HMC and RM-HMC, the parameters were the step size and number of intermediate steps. For Riemannian SOL-MC, standard SOL-MC, and fixed damping SOL-MC, the free parameters are the step size and scaling on the damping matrix. While for the non-reversible formulations, i.e. those with momentum resampling, of Riemannian SOL-MC, standard SOL-MC, and fixed damping SOL-MC, the free parameters are the step size, scaling on the damping matrix, and the frequency of momentum resampling. For each of these choices of parameters, 32 tests were run, each using a total of 17,500 model evaluations.

Figure 2.11 presents the results comparing HMC, fixed damping SOL-MC, and fixed damping SOL-MC with momentum resampling. For these algorithms, the mass and damping matrix were set at the start of the algorithm and fixed based upon the Fisher information matrix evaluated at the initial point. This means that

Variable	ESS 1	ESS 2	ESS 3	ESS 4	ESS 5	ESS 6	ESS 7	ESS 8	ESS 9	ESS 10	ESS Total	Scaled Mean	Std
k_1	566	235	603	350	724	31	729	504	250	13	4005	1.006	0.002
k_2	568	355	595	452	698	63	642	432	268	12	4087	0.994	0.002
k_3	466	103	507	297	340	71	466	305	159	8	2721	1.003	0.002
c_m	287	162	387	339	391	104	484	340	141	10	2649	1.013	0.08
c_k	332	273	488	290	419	47	472	243	177	12	2754	0.958	0.045
$r_{u,1}$	561	200	409	424	486	39	435	478	117	8	3156	0.998	0.003
$r_{u,2}$	240	246	291	245	213	15	638	251	251	127	2518	0.99	0.012
$r_{u,3}$	190	64	179	158	75	13	91	107	58	11	946	0.939	0.12
α_1	583	294	474	434	654	53	430	388	244	13	3567	1.013	0.015
α_2	392	360	597	627	367	12	456	295	426	6	3539	1.038	0.037
α_3	138	35	132	103	43	10	131	44	31	10	678	1.02	0.222
σ^2	782	439	738	492	635	19	601	743	204	15	4666	0.983	0.025
$H(\theta, p)$	456	269	481	315	484	36	609	289	139	61	3139		

Table 2.2: Summary of the statistics for the ten Markov Chains describing the effective sample size of each chain of length 2000 samples, total effective number of samples out of 20000 samples, mean estimate for each variable scaled relative to the true value and the scaled standard deviation.

after the initialization, only the gradient of the likelihood function is computed at each step. Each point on this plot represents the evaluation of a sampler using a specific set of parameters while the line indicates the Pareto optimal performance front based upon the different parameter choices. Looking at the Pareto fronts, both fixed damping SOL-MC and fixed damping SOL-MC with resampling are able to outperform HMC in terms of mean estimate error. However, fixed damping SOL-MC with resampling is also able to get lower covariance estimate error than HMC. This shows the resampling enables SOL-MC to best trade off covariance and mean error. The robustness of the algorithm also helps it in this case since the mass and damping are fixed so they do not always correspond to the correct geometry of the posterior in this example.

Figure 2.12 presents another set of results comparing HMC, standard SOL-MC, and standard SOL-MC with momentum resampling. For these algorithms, the mass was fixed at the start of the algorithm but the damping matrix was adapted at each step for the SOL-MC family algorithms based upon the Fisher information matrix. This means that the two SOL-MC algorithms compute the Fisher information at each step to adapt more to the geometry than standard HMC. In this example, the

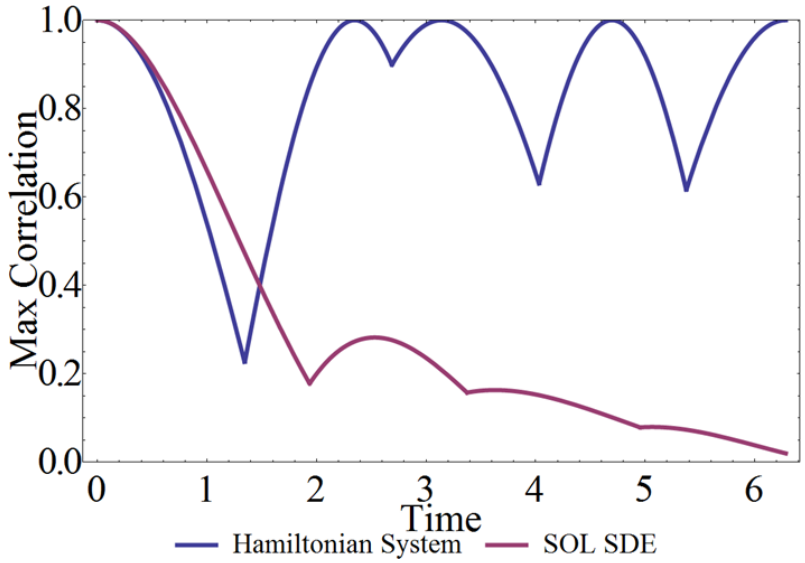

Figure 2.10: Autocorrelation for the Hamiltonian and SOL-SDE dynamics defined by a Gaussian posterior with imperfectly estimated covariance structure.

added information at each step enables both SOL-MC with resampling and standard SOL-MC to outperform HMC, and also their fixed counterparts, in both the mean and covariance estimate error. Since the damping matrix can adapt to the geometry, the robustness of SOL-MC becomes less of a concern unlike the fixed case.

Finally, Figure 2.13 presents the results comparing RM-HMC, Riemannian SOL-MC, and Riemannian SOL-MC with momentum resampling. For these algorithms, the mass and damping matrices are adapted every iteration based upon the Fisher information. In this case, there is less performance advantage of Riemannian SOL-MC with resampling over RM-HMC, although the added parameters allow Riemannian SOL-MC with resampling to better trade-off the objectives. This is probably because RM-HMC always has a good estimate of the local covariances structure of the posterior so the robustness that SOL-MC type algorithms provide is no longer superior. Riemannian SOL-MC is seen to generally under perform.

2.6 Discussion

MCMC methods based upon an auxiliary dynamical system have become popular for efficiently sampling the posterior of Bayesian inference problems. The underlying dynamical systems enable these algorithms to better exploit the geometry of the posterior distribution and to propose samples which converge quickly to the stationary distribution and have low sample correlation. We introduce a new sampler, Second-Order Langevin Monte Carlo, which explicitly treats the sampler as

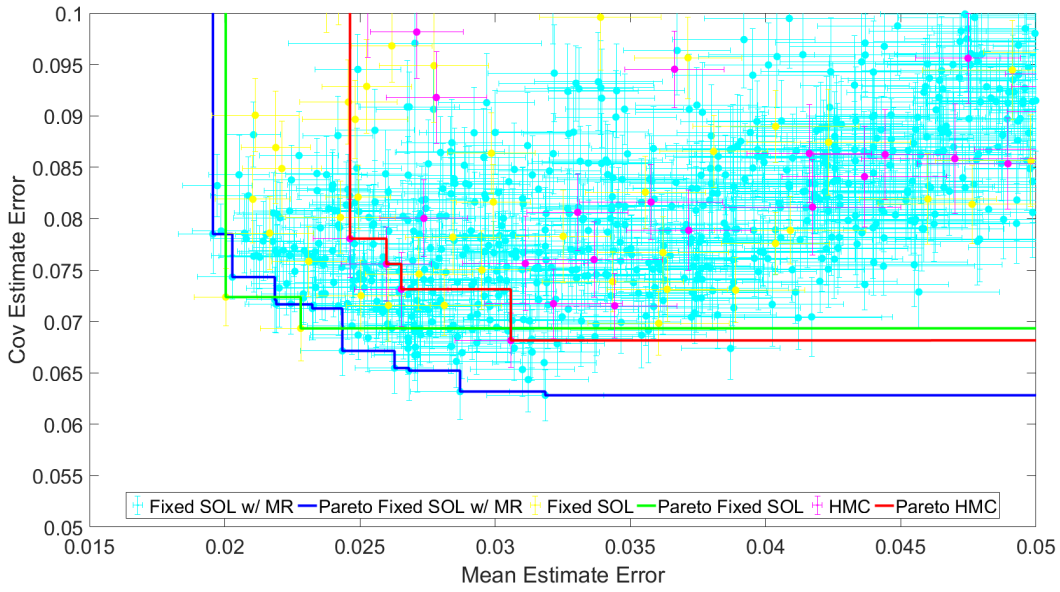

Figure 2.11: Mean and Covariance Estimate Error when using HMC, fixed damping SOL-MC, and fixed damping SOL-MC with momentum resampling (MR). The points indicate implementations with different parameters and their error bars. The lines show the Pareto optimal performance front for each algorithm.

a dynamical system described by a Stochastic Differential Equation. This SDE has the posterior as its stationary distribution and is solved using a Metropolized integrator to ensure that it samples exactly from the correct distribution, despite the approximate numerical solution of the dynamic equations of the sampler. Since the sampler is described by a stochastic dynamical system, we can choose its parameters by studying the behavior of the system. For the SOL SDE, we choose the auxiliary damping and mass matrices to optimize for Gaussian-like distributions by maximizing the convergence rate and minimizing correlation. Further, when implementing SOL-MC, we can take advantage of automatic differentiation which significantly speeds up derivative calculations over other methods.

We investigate the effectiveness of SOL-MC by solving two system identification problems based upon identifying the parameters of a three story building. The first problem yields a globally identifiable posterior. For this problem, SOL-MC is able to generate samples very efficiently and without bias. We also show how the choice of damping matrix and time step affect the efficiency of the sampler, which further helps inform our parameter choices. The second example has an unidentifiable posterior distribution, which makes the problem more challenging since it is highly non-Gaussian. We show that standard SOL-MC can find the high

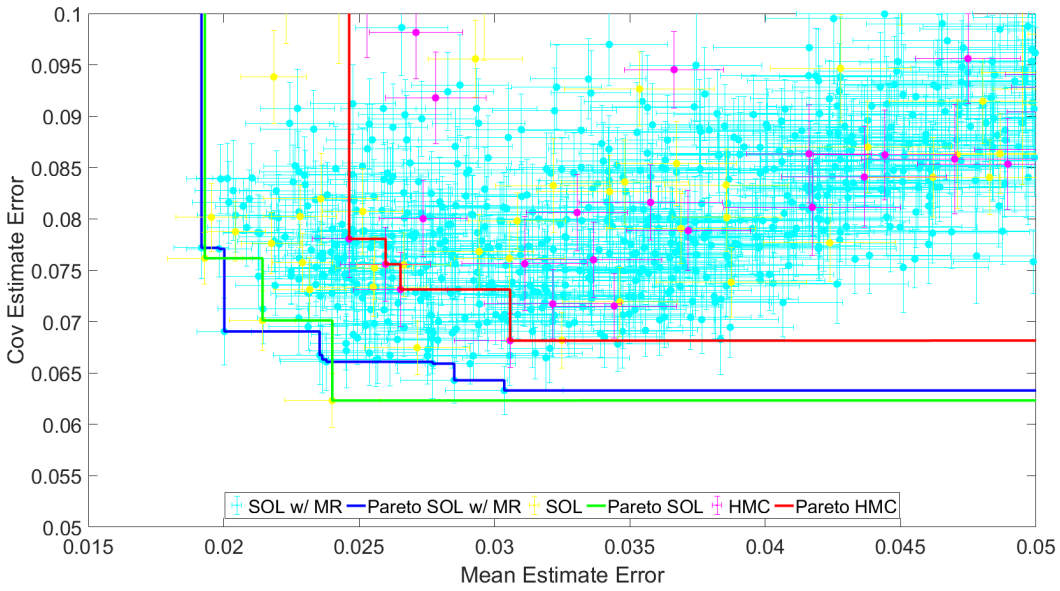

Figure 2.12: Mean and Covariance Estimate Error when using HMC, SOL-MC with adaptive damping, and SOL-MC with momentum resampling (MR) and adaptive damping. The points indicate implementations with different parameters and their error bars. The lines show the Pareto optimal performance front for each algorithm.

probability manifold but it is less efficient and exhibits some bias when sampling since the local geometry varies over the manifold. This illustrates the need to develop additional adaptive methods to optimize the sampler for non globally identifiable posteriors.

Other aspects of dynamical system can be exploited to further improve SOL-MC. Adaptive time step control can be used to improve convergence of the SDE to the stationary distribution. Time step control is common for integrating dynamical systems and can be employed here to continually find a good trajectory time step as it moves toward the high probability manifold. Further, when using a population of chains to sample a difficult posterior, using a population level estimate of the posterior covariance matrix weighted by the sample likelihood would help to better capture the global properties of the posterior. Finally, because scaling the damping matrix can affect the sampling by trading off the hamiltonian dynamics and diffusion, we could adjust this dynamically to better suit the local posterior geometry. In general, we see that continuing to exploit the properties of dynamical systems when designing SOL-MC will enhance its performance for challenging high dimensional inference problems.

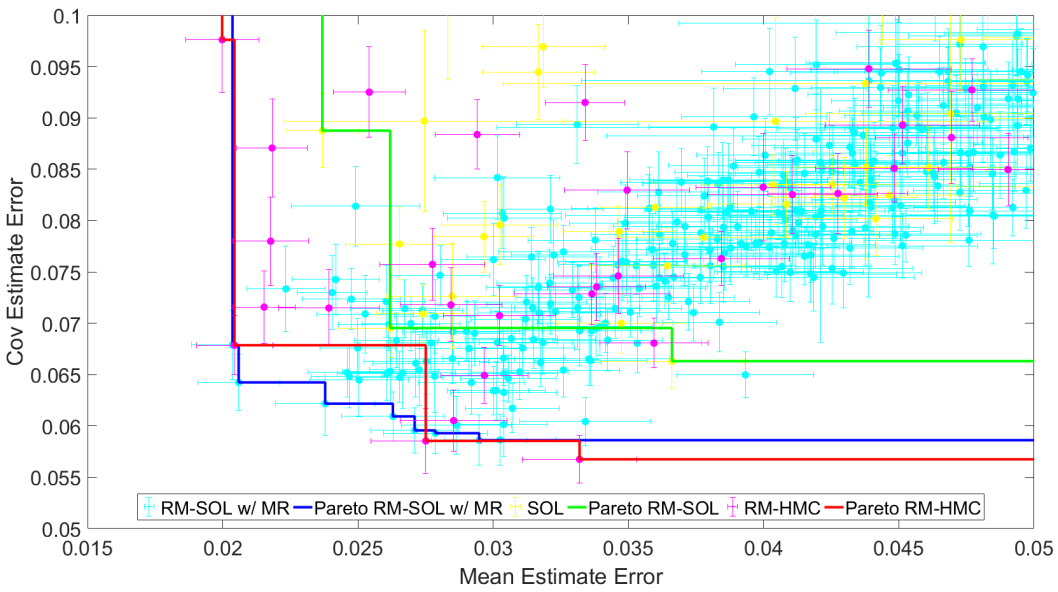

Figure 2.13: Mean and Covariance Estimate Error when using RM-HMC, Riemannian SOL-MC, and Riemannian SOL-MC with momentum resampling (MR). The points indicate implementations with different parameters and their error bars. The lines show the Pareto optimal performance front for each algorithm.

Chapter 3

SEQUENTIAL TEMPERED MCMC

It is important for many applications to solve Bayesian inference problems on distributed high performance computing (HPC) resources and to take advantage of new computing environments like GPUs. Typical MCMC methods are sequential and cannot directly take advantage of HPC. However, population-based MCMC methods, like transitional/multilevel MCMC [CC07; PC12], AIMS [BZ13], AlTar/CATMIP [MSB13], and most generally Sequential Monte Carlo (SMC) [KBJ14; DDJ06; Jas+11; Cho02] can exploit parallelism by evolving a population of Markov chains simultaneously. For example, AlTar/CATMIP has been developed for GPU cluster implementations to solve inference problems in geophysics. This class methods, mostly derived from TMCMC and SMC, combine importance sampling, Metropolis-Hastings, and annealing. In this thesis, we refer to this class of methods as Sequential Tempered MCMC (ST-MCMC). During ST-MCMC the population of samples is gradually transformed from being distributed according to the prior distribution to being distributed according to the posterior distribution over a series of levels. In this chapter, I present several improvements that are generally applicable to Sequential Tempered MCMC methods.

First, I have developed theory specifically for population-based MCMC to quantify the expected performance of the algorithm based upon Effective Sample Size (ESS). Previous work has discussed the convergence properties of these methods [Del04; DDJ06; CC07] and provided basic intuition for parameter choices [KBJ14], however theoretical results are still needed to relate parameters choices and performance. This theory defines how the population size, annealing rate, and the Markov chain autocorrelation influence the effective sample size of the population at each level, which controls the quality of the Monte Carlo estimates. I integrate this theory into new methods for adaptively tuning the parameters defining population MCMC algorithms to achieve better performance. Not only will this work make these methods faster, but they will make Sequential Tempered MCMC methods easier for practitioners to use and trust their results.

Second, I have developed better proposal methods for the MCMC step to explore and adapt to the high dimensional posterior distribution. I have introduced new tuning

methods for adapting the proposal distribution scaling (spread) using a feedback controller to achieve a desired acceptance rate. This controller enables the proposal distribution to learn from the population level estimate of the acceptance rate and tune the scaling accordingly based upon the underlying theory of Metropolis MCMC [Ros+11]. Further, moving beyond standard Metropolis, which suffers form the curse of dimensionality, I have adapted the Modified Metropolis Algorithm [AB01; ZK11], for use when the proposal distribution is a multivariate Gaussian with any covariance structure. This new algorithm, called the Rank One Modified Metropolis Algorithm (ROMMA), performs a series of rank one updates according the prior distribution to form a proposal candidate. ROMMA avoids many of the high-dimensional scaling issues seen with Metropolis MCMC particularly when the prior distribution has bounded support.

3.1 Motivation

Sequential Markov Chain Monte Carlo methods rely on generating samples by evolving a Markov Chain which explores the posterior distribution and estimates expectations with respect to the posterior based upon the ergodicity of the chain. These single chain sequential samplers are difficult to parallelize, tune, and adapt to complex posterior environments such as unidentifiable and locally identifiable models, such as multi modal distributions. Sequential Tempered MCMC methods are population-based methods that can more efficiently generate samples from complex posteriors and utilize high performance computing parallelism and thus overcome many of these challenges.

ST-MCMC methods evolve a population of Markov chains through a series of intermediate posterior distribution levels until, as a population, they reach the final posterior. First, the evolution of the Markov chains, typically the most computationally intensive aspect of the problem because it requires solving the forward model, can be done in parallel for these methods, thereby enabling HPC. Second, the population of samples that evolve through the levels provides a global perspective on the posterior distribution, enabling easier parameter tuning that does not rely on the ergodicity, like adaptive MCMC [Ros+11]. Finally, having a population of samples and using intermediate levels allows the different chains to simultaneously explore multiple modes of the posterior. Since each chain produces a single sample at the end of the final level, the mixing time of the Markov chain while sampling the ultimate posterior distribution is significantly less relevant and so it can be much more efficient to use these methods. The intermediate levels also enable the algorithm to

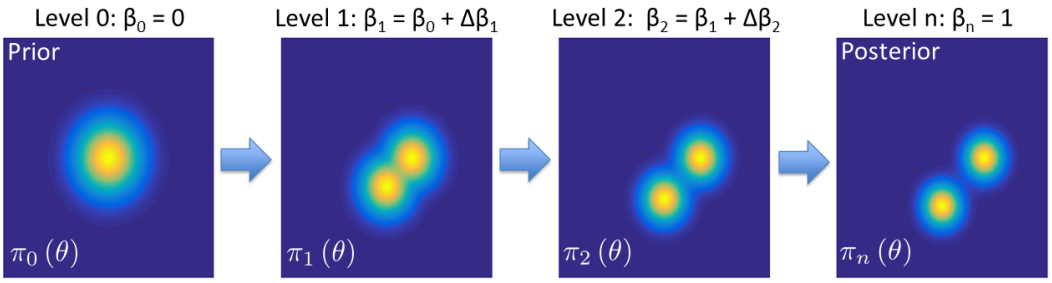

Figure 3.1: Illustration of a set of distributions which gradually transform a unimodal prior into a bimodal posterior. These intermediate distributions are defined by an annealing factor $\beta \in [0, 1]$

estimate the model evidence for solving model selection problems [CC07].

My work on ST-MCMC methods has been focused on optimizing the MCMC component of the algorithm and to better design the algorithm as a whole based upon theoretical results. While previous studies have introduced ways of tuning ST-MCMC methods, lots of work remains to better exploit the information that can be gained from the population of Markov chains. I have developed methods to better exploit the information that is learned from the samples to tune the proposal distribution and to introduce better sampling methods which do well in high dimensional spaces with difficult-to-sample geometries. Previously, little theoretical guidance has been provided for choosing the length of the Markov chains and the target variation between intermediate distributions. I introduce some theoretical results to help provide guidance on setting these parameters.

3.2 Sequential Tempered MCMC Methods

Sequential Tempered Algorithm

Sequential Tempered MCMC methods can be divided into three basic parts: annealing, MCMC, and importance resampling. The annealing step introduces intermediate distributions that gradually evolve the samples from the prior to the posterior instead of directly jumping to the posterior. An example of these intermediate distributions is presented in Figure 3.1. The MCMC step allows the samples to explore the intermediate distributions and adjust to changes as the distributions evolve. The importance resampling step discards unlikely samples and multiplies likely samples to maintain and rebalance the samples with respect to the changes in the intermediate distributions from one level to the next.

A general ST-MCMC algorithm is described in Algorithm 1. Depending on the

implementation, the choice for resampling, adaptation, and MCMC may vary. This algorithm could be presented even more generally as a form of Sequential Monte Carlo as found in [DDJ06]. For the rest of this section I will discuss the general algorithm and us AlTar/CATMIP in particular as introduced in [MSB13] to fix some choices. The primary difference between AlTar/CATMIP and other methods like TMCMC is that it has longer MCMC chains, each of equal length. In the subsequent sections, modifications to this algorithm will be discussed.

When initializing the algorithm, the initial sample population $\theta^{(0)}$ is drawn from the prior distribution. The number of samples in the population is typically fixed at all levels to be N. Parameters used for the algorithm are then initialized, such as the annealing factor β, level counter k, and parameters that define the proposal distribution $Q\left(\hat{\theta} \mid \theta\right)$. Usually, the proposal distribution is taken in the form of $\hat{\theta} = \theta + \eta$, $\eta \sim \mathcal{N}\left(0, \sigma^2\Sigma\right)$. Here σ^2 is a tuned scaling factor and Σ is the sample covariance.

At the beginning of each subsequent level k, the annealing factor β_k is computed. The annealing factor controls the influence of the data at every level by gradually transitioning the level k stationary distribution from the prior at $\beta_0 = 0$ to the posterior at $\beta_{final} = 1$. The increment $\Delta\beta$ at each level is chosen to ensure that the intermediate distributions are not too far apart, otherwise the sample population $\theta^{(k-1)}$ does a poor job representing the next level distribution $\pi_k(\theta)$. This increment is controlled by looking at the degeneracy of the sample importance weights, which are weights that allow us to transform samples $\theta^{(k-1)}$ from making estimates with respect to $\pi_{k-1}(\theta)$ to $\pi_k(\theta)$. This process is illustrated in Figure 3.2. This weight function takes the form of $w(\theta, \Delta\beta) = \frac{p(\mathcal{D}|\theta)^{\beta+\Delta\beta}\pi_0(\theta)}{p(\mathcal{D}|\theta)^{\beta}\pi_0(\theta)} = p(\mathcal{D} \mid \theta)^{\Delta\beta}$. The degeneracy in the weights is measured by computing their coefficient of variation $COV\left[w\left(\theta^{(k-1)}, \Delta\beta\right)\right]$ and trying to find a $\Delta\beta$ such that it is equal to some target threshold κ^*. Based upon the theory of importance sampling, the coefficient of variation is an estimate of the effective sample size, making it a good proxy for degeneracy. We use the sample coefficient of variation and so to solve for the right $\Delta\beta$, we must solve:

$$COV\left[w\left(\theta^{(k-1)}, \Delta\beta\right)\right] = \frac{\sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(w\left(\theta_i^{(k-1)}, \Delta\beta\right) - \frac{1}{N}\sum_{i=1}^{N} w\left(\theta_i^{(k-1)}, \Delta\beta\right)\right)^2}}{\frac{1}{N}\sum_{i=1}^{N} w\left(\theta_i^{(k-1)}, \Delta\beta\right)} = \kappa^* \tag{3.1}$$

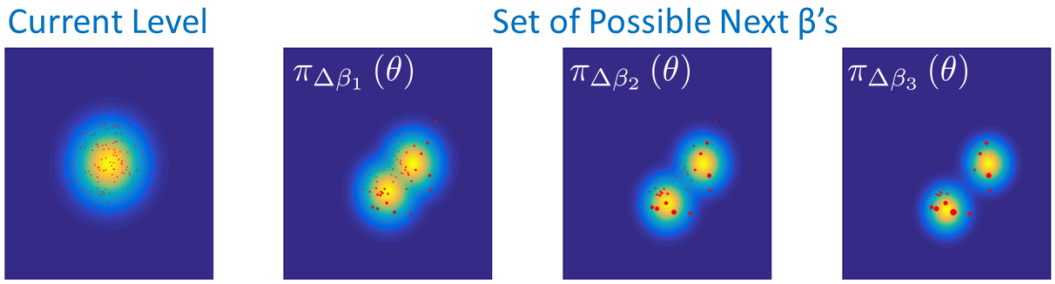

Figure 3.2: Illustration of finding $\Delta\beta$ that defines how much additional influence the data has in the next intermediate distribution level. Red dots indicate the sample and their size indicates their weight. If too large a $\Delta\beta$ step is made, only a few samples will have the majority of the weights, indicating that the samples poorly represent the distribution. If too small a $\Delta\beta$ step is made, the next distribution is too close to the current distribution making it an inefficient choice.

This equation is typically solved using a bisector method since we have an upper and lower bound for $\Delta\beta$. Once $\Delta\beta$ and $\beta_k = \beta_{k-1} + \Delta\beta$ are found, the sample population $\theta^{(k-1)}$ with normalized importance weights $\hat{w}^k$ can be used to make expectation estimates with respect to the k$^{\text{th}}$ level PDF $\pi_k(\theta)$:

$$\hat{w}_i^k = \frac{w\left(\theta_i^{(k-1)}, \Delta\beta\right)}{\sum_{j=1}^{N} w\left(\theta_j^{(k-1)}, \Delta\beta\right)} \tag{3.2}$$

For AlTar/CATMIP, we need the covariance of $\theta^{(k)}$ with respect to $\pi_k(\theta)$, which can be estimated as the covariance Σ of the weighted sample population.

The algorithm is now ready to produce the next level sample population. Importance resampling using the normalized importance weights produces an initial sample population ${}_0\theta^{(k)}$ that is asymptotically distributed for increasing samples size, according to $\pi_k(\theta)$. Several methods may be used for importance resampling but the most common for ST-MCMC is multinomial resampling where ${}_0\theta_i^{(k)}$ is picked at random from the weighted set of samples $\theta^{(k-1)}$ and the probability of choosing $\theta_j^{(k-1)}$ is $\hat{w}_j^k$. Because the new sample population is a subset of the previous level population, there is some added degeneracy. To now add diversity, the sample population ${}_0\theta^{(k)}$ is evolved according to $\pi_k(\theta)$ using MCMC.

The MCMC step for ST-MCMC is defined by the type of proposal distribution used and the length of the chain. Both of these factors can significantly influence the performance of the algorithm as the chains must be allowed to evolve sufficiently

Algorithm 1: General Sequential Tempered MCMC (ST-MCMC)

Step 0: Initialize Level k = 0

Draw $\theta_i^{(0)}$ for $i = 1...N$ from the prior $\pi_0(\theta)$

Set $\beta_0 = 0$ and $k = 1$

Define proposal $Q_1\left(\hat{\theta} \mid \theta\right)$

Step k.1: Find k[th] level PDF: $\pi_k(\theta) = p(\mathcal{D} \mid \theta)^{\beta_k} \pi_0(\theta)$

Define weights $w\left(\theta_i^{(k-1)}, \Delta\beta\right) = p\left(\mathcal{D} \mid \theta_i^{(k-1)}\right)^{\Delta\beta}$, for $i = 1...N$

Solve $COV\left[w\left(\theta^{(k-1)}, \Delta\beta\right)\right] = \kappa^*$ for $\Delta\beta$

Set $\beta_k = \beta_{k-1} + \Delta\beta$

if $\beta_k > 1$ **then**

$\beta_k = 1$

$\Delta\beta = 1 - \beta_{k-1}$

end

Step k.2: Compute some population level statistics for $\pi_k(\theta)$ from samples $\theta^{(k-1)}$ and weights $w\left(\theta^{(k-1)}, \Delta\beta\right)$

Step k.3: Resample the population $\theta^{(k-1)}$ according to the normalized weights $\hat{w}^k$ to get initial level k samples ${}_0\theta^{(k)}$

Step k.4: Adapt the MCMC proposal $Q_k\left(\hat{\theta} \mid \theta\right)$ using the population statistics

Step k.5: Evolve samples ${}_0\theta^{(k)}$ according to MCMC with respect to $\pi_k(\theta)$ and proposal $Q_k\left(\hat{\theta} \mid \theta\right)$ for a number of steps l

Return the ends of the chains as the final level k sample population $\theta^{(k)}$

Update Parameters of $Q_{k+1}\left(\hat{\theta} \mid \theta\right)$ using MCMC statistics

Step k.6: Stopping criterion

if $\beta_k == 1$ **then**

Stop

end

to ensure that they explore the distribution $\pi_k(\theta)$ and de-correlate from each other. Typically, within Metropolis-Hastings MCMC, a Gaussian proposal is used where the candidate $\hat{\theta} = \theta + \eta, \eta \sim \mathcal{N}\left(0, \sigma^2\Sigma\right)$. Σ is the sample covariance matrix computed using the weighted samples and σ^2 is the scaling factor which is adapted using the acceptance rate of the sampler at the previous MCMC level α_{k-1}. AlTar/CATMIP uses $\sigma = \frac{8}{9}\alpha_{k-1} + \frac{1}{9}$, where α_{k-1} is the average acceptance rate of the Metropolis-Hastings algorithm at the previous level, and the chains are evolved through a fixed number l of MCMC steps. The final sample population at level k is $\theta^{(k)}$ at the end of these chains. The algorithm then iterates unless $\beta_k = 1$ in which case the final samples are from the posterior distribution.

Application to Model Selection

One of the original motivations for the development of Sequential Tempered MCMC methods like TMCMC was to solve Bayesian model selection problems. Such problems can be very difficult to solve because they require computing the likelihood of the data given the model, $p(\mathcal{D} \mid \mathcal{M})$, called the model evidence. The model evidence is the high dimensional normalization factor in Bayes' Theorem for θ that MCMC was developed to avoid computing in the first place. It can be thought of as the expectation of the probability of the data with respect to the prior distribution generated by the model $\mathcal{M}$.

$$p(\mathcal{M} \mid \mathcal{D}) = \frac{p(\mathcal{D} \mid \mathcal{M})\, p(\mathcal{M})}{p(\mathcal{D})} \propto \left(\int p(\mathcal{D} \mid \theta, \mathcal{M})\, p(\theta \mid \mathcal{M})\, d\theta \right) p(\mathcal{M}) \quad (3.3)$$

This integral could be naively estimated using Monte Carlo sampling of the prior distribution $p(\theta \mid \mathcal{M})$. This estimate would be very computationally inefficient when the data is informative, since the high probability content of the prior may be very far from the high probability content of the posterior. However, the intermediate levels of ST-MCMC enable us to address this problem by decomposing the evidence computation over the intermediate levels. Let c_k denote the contribution of the evidence for the intermediate level k of the s levels, then:

$$\int p(\mathcal{D} \mid \theta, \mathcal{M})\, p(\theta \mid \mathcal{M})\, d\theta = \prod_{k=1}^{s} \frac{\int p(\mathcal{D} \mid \theta, \mathcal{M})^{\beta_k}\, p(\theta \mid \mathcal{M})\, d\theta}{\int p(\mathcal{D} \mid \theta, \mathcal{M})^{\beta_{k-1}}\, p(\theta \mid \mathcal{M})\, d\theta} = \prod_{k=1}^{s} c_k \quad (3.4)$$

For each intermediate level, we can perform a fairly accurate Monte Carlo estimate between the previous level and the current level since these distributions are designed to be relatively close to each other in terms of the relative effective sample size of samples coming from the previous level. Having a high ESS means Monte Carlo sampling will be effective. This leads to the Monte Carlo estimate:

$$c_k = \int p(\mathcal{D} \mid \theta, \mathcal{M})^{\Delta\beta_k} \frac{p(\mathcal{D} \mid \theta, \mathcal{M})^{\beta_{k-1}}\, p(\theta \mid \mathcal{M})}{\prod_{j=1}^{k-1} c_j} d\theta \approx \frac{1}{N} \sum_{i=1}^{N} p\left(\mathcal{D} \mid \theta_i^{(k-1)}, \mathcal{M}\right)^{\Delta\beta_k} \quad (3.5)$$

This integral can be thought of as the evidence provided by level k where $p(\mathcal{D} \mid \theta, \mathcal{M})^{\Delta\beta_k}$ is the data likelihood added by level k and $\frac{p(\mathcal{D}|\theta,\mathcal{M})^{\beta_{k-1}} p(\theta|\mathcal{M})}{\prod_{j=1}^{k-1} c_j}$ is the prior for level

k. The combined estimate of the model evidences c_k provides an asymptotically unbiased estimate of the total model evidence.

Relevant Theoretical Results

For the specific case of TMCMC [CC07], theoretical results show that it produces an asymptotically unbiased estimator of expectations taken with respect to the posterior distribution. The authors of [CC07] also present Central Limit Theorem results to show the asymptotic variance of these estimates. For Sequential Monte Carlo methods, [DDJ06; Del04] present a very general Central Limit Theorem result that leverages work from the Sequential Monte Carlo and filtering literature. These results prove the validity of ST-MCMC and can help estimate posterior confidence. However, these results address the question of how good an estimate is, rather than directly addressing the question of how parameter choices in the algorithm effect the estimator's quality. The theoretical results presented in this chapter seek to formally address this latter question. This question has been considered in [KBJ14], where they present some intuition for choosing parameters.

3.3 Effective Sample Size Theoretical Results

Effective Sample Size

The number of effective samples is a common measure used in sampling problems like MCMC and importance sampling. Because of sample weights or sample correlations, the estimate quality does not necessary behave the same as if the estimate was made using independent samples. Thus, the Effective Sample Size (ESS), given by:

$$N_{ess} = \frac{var\left(\theta\right)}{var\left(\hat{\mu}\left(\theta\right)\right)} \tag{3.6}$$

estimates the size of the population of independent samples that would give the same variance of the estimate $\hat{\mu}\left(\theta\right)$ as the weighted or correlated sample population. For example, in the case of a simple mean estimate, $\hat{\mu}\left(\theta\right) = \frac{1}{N}\sum_{i=1}^{N}\theta_i$. In order to estimate the ESS, for Sequential Tempered MCMC methods, we must consider the effects of weighting the samples in the Importance Sampling step, performing the resampling, and evolving the population during the MCMC step. In general, weighting and resampling reduce the ESS while the MCMC step increases the ESS.

Importance Weights

One basic estimate for the change in effective sample size due to sample weights for importance sampling is given by [Owe13]:

$$N_{ESS} \approx \frac{N}{1+\kappa^2} \tag{3.7}$$

Here κ is the sample coefficient of variation of the weights, which is defined as the ratio of the sample standard deviation of the weights σ_w divided by their sample mean μ_w:

$$\kappa = COV\,(w) = \frac{\sigma_w}{\mu_w} \tag{3.8}$$

$$\mu_w = \frac{1}{N}\sum_{i=1}^{N} w_i \tag{3.9}$$

$$\sigma_w = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(w_i - \mu_w)^2} \tag{3.10}$$

This estimate is very general since it is independent of the quantity of interest, giving us basic insight into the evolution of the effective sample size during importance sampling. However, in practice this estimate is imperfect and can be improved using more specific knowledge about the quantity of interest [Owe13].

Resampling

For multinomial resampling based on the sample weights, we can compute the reduction in the effective sample size by quantifying the added correlation in the population. Assuming that the weighted sample population has an effective sample size of N_{ESS_0}, we find the ESS of the resampled population, N_{ESS_1}, to be (see proof of Theorem 1 at the end of this section):

$$N_{ESS_1} \approx N\frac{N_{ESS_0}}{N_{ESS_0} + N - 1} \tag{3.11}$$

If the number of samples is large and $N_{ESS_0} = N$ we can see that $N_{ESS_1} = \frac{1}{2}N$. This means that using multinomial resampling can significantly reducing the effective sample size. Other sampling strategies, such as stratified resampling, have been

shown to have a smaller impact on ESS in other contexts like particle filtering [DC05].

MCMC

The MCMC step decorrelates samples in the population since they are evolved independently and this causes the effective sample size to increase. Under some assumptions, we can approximate the effective sample size N_{ESS_2} after the MCMC step in terms of N_{ESS_1}, the ESS of the initial population, by calculating the sample correlation ρ of the two populations (see proof of Theorem 2 at the end of this section):

$$N_{ESS_2} \approx N_{ESS_1} \frac{1}{\left(1 - \frac{N_{ESS_1}}{N}\right)\rho^2 + \frac{N_{ESS_1}}{N}} \tag{3.12}$$

$$\rho = \frac{\sum_{i=1}^{N} \left(\theta_i^{(0)} - \mu_{\theta^{(0)}}\right)\left(\theta_i^{(1)} - \mu_{\theta^{(1)}}\right)}{\sqrt{\sum_{i=1}^{N} \left(\theta_i^{(0)} - \mu_{\theta^{(0)}}\right)^2}\sqrt{\sum_{i=1}^{N} \left(\theta_i^{(1)} - \mu_{\theta^{(1)}}\right)^2}} \tag{3.13}$$

In defining the correlation, $\theta_i^{(0)}$ is the initial i$^{\text{th}}$ sample state and $\theta_i^{(1)}$ is the final i$^{\text{th}}$ sample state, while μ_θ corresponds to the sample mean for a given population.

Finding a good measure of the correlation of a multivariate sample population is important to insure that the sample population converges to the correct distribution. Even if the parameter-wise correlations are small, there might be large correlations in some transformed set of coordinates. One strategy to mitigate this issue is to use Canonical Correlation Analysis (CCA) [HS07]. CCA is a technique to efficiently find the direction and magnitude of maximum correlation between two populations, i.e., find vectors a and b to maximize corr $\left(a^T\theta_0, b^T\theta_1\right)$. By minimizing the canonical correlation, we can insure the correlation target is achieved.

Combining Results

By combining the results of (3.7), (3.11), and (3.12) for the importance weighting, resampling, and MCMC, we can derive the evolution of the effective sample size of the population as a function of the target coefficient of variation κ and correlation ρ between the start and end of the Markov chains at level $(k+1)$:

$$n_{k+1} \approx n_k \frac{N}{(N-1)\left(1+\kappa^2\right)\rho^2 + n_k} \tag{3.14}$$

If the population size is large and the COV κ and correlation ρ targets are constant for all steps, we can find a condition for the existence of a non-zero stationary number of effective samples by finding the fixed point of (3.14) as the number of levels increases:

$$\rho^2 < \frac{1}{1+\kappa^2} \tag{3.15}$$

If this condition holds, then an asymptotic expression for the effective number of samples is

$$N_{ESS} \approx N\left[1 - \left(1+\kappa^2\right)\rho^2\right] \tag{3.16}$$

In general, this analysis gives us a guide for setting the target COV and correlation to obtain a satisfactory number of samples from the posterior distribution. The region where learning is possible (i.e. the ESS will be non-zero at the end of all the levels) is found in Figure 3.3. For example, if $\kappa = 1$, then $N_{ESS} = N\left(1 - 2\rho^2\right)$ and it requires $\rho < \frac{1}{\sqrt{2}}$ for learning.

Computational ESS Experiment

In order to evaluate how well these theoretical approximations for effective sample size work as a guideline for choosing parameters in ST-MCMC, we investigate estimating two quantities of interest (QOI) for a 10 dimensional Gaussian problem. Ten thousand samples were used to estimate the mean of the first parameter and the squared magnitude of the parameter vector and 64 runs were used to estimate the variance of the mean estimate so that the ESS could be computed by equation (3.6). The final ESS of the two QOI at the end of ST-MCMC using different target coefficient of variation and correlation values is showing in Figure 3.4. This experiment shows that in this case, the approximations made underestimate the effective sample size and that the cutoff line is overly conservative. In fact, many of the parameters yield an effective sample size ratio greater than one, indicating that the sample population outperforms independent samples. This type of behavior is not accounted for in our approximations.

To better understand these results, we can look at a set of parameters where the target COV is 1.0 and the target correlation is 0.6. As ST-MCMC runs, we can look at the effective sample size of the population with respect to the target distribution of that beta level. This ESS is evaluated at each of the three steps of the level i.e. the

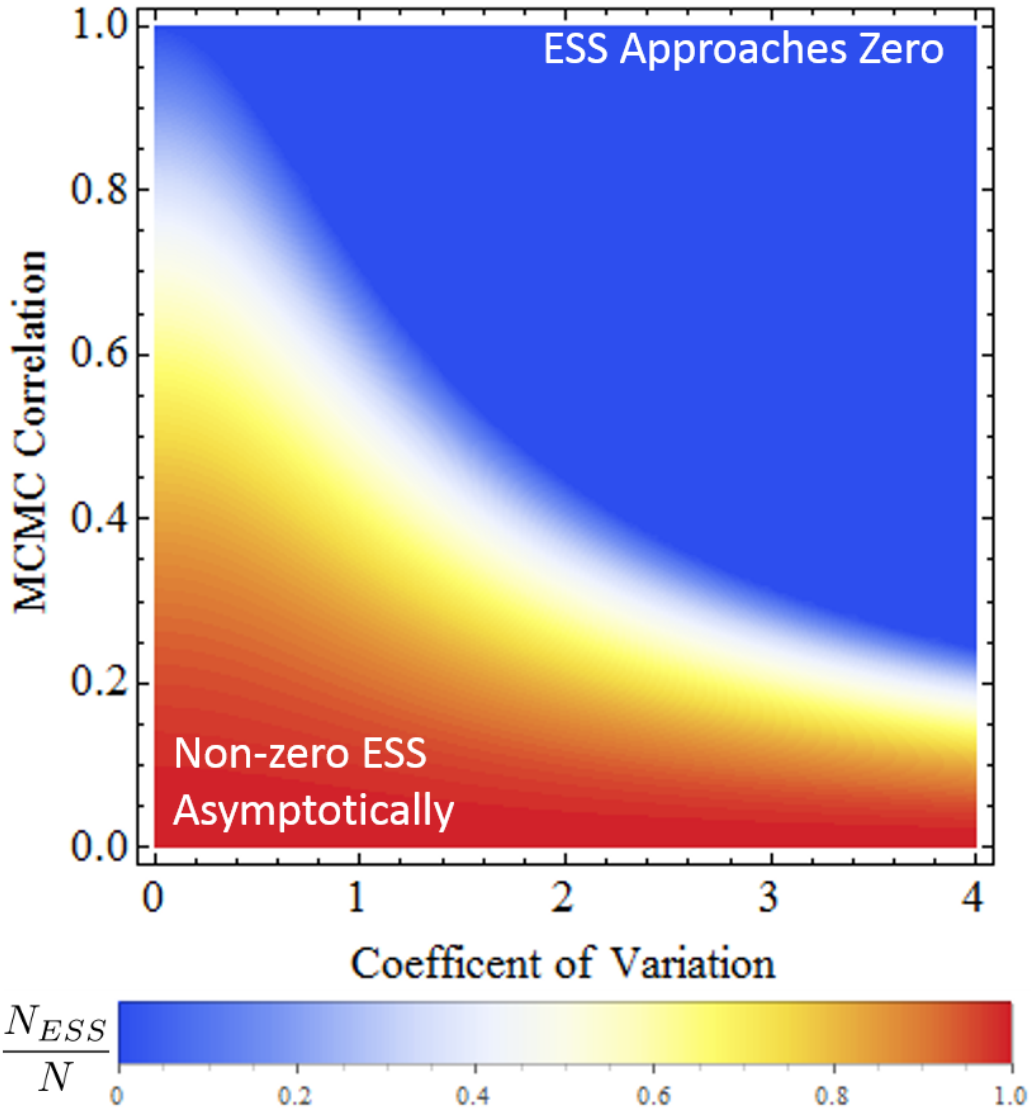

Figure 3.3: Asymptotically, with respect to the number of levels, the ratio of the effective numbers of samples to total samples is determined by the choices of the target coefficient of variation and MCMC correlation.

assignment of importance weights, the resampling, and the MCMC sampling. This trajectory is shown in Figure 3.5 for the two QOI and the theoretical estimates. The oscillations of the ESS during the trajectories correspond to the increase in the ESS during the MCMC steps and the decrease in ESS during the importance weighting and resampling steps. These trajectories can then be separated into these three steps as seen in Figure 3.6, where each line plots the increase or decrease factor of the ESS due to either the MCMC, resampling, or importance weighting steps. We can see that our theoretical results can approximately capture the relative importance of

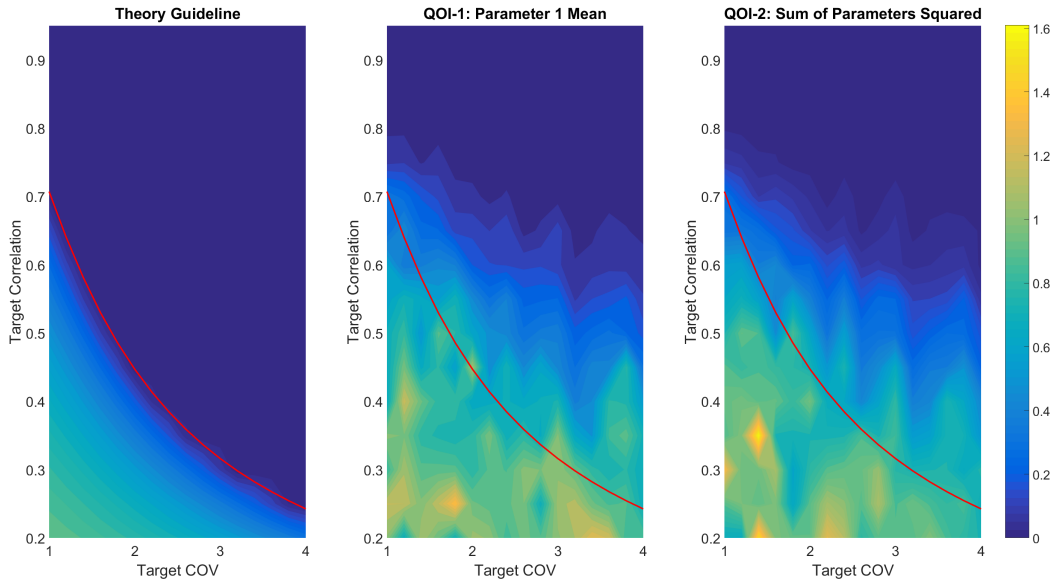

Figure 3.4: Comparison of the ratio of ESS to total number of samples for the theoretical results, equation (3.16), and the results for two actual quantities of interest computed by simulation. The red line is the theoretical learning cutoff.

these three steps, albeit with some bias. Also, it is evident that there is significant variability in the ESS that is not accounted for by the theory.

Derivation of ESS Results

ESS for Resampling

Multinomial resampling transforms a weighted sample population whose samples are $\theta^{(0)}$ and normalized weights $\hat{w}$ into an equally weighted sample population $\theta^{(1)}$. This is done by choosing a sample $\theta_j^{(1)}$ to be $\theta_i^{(0)}$ with probability $\hat{w}_i$. This leads to the following lemma and theorem:

Lemma 1. $\mathbb{E}\left[\theta_i^{(1)}\theta_j^{(1)}\right] = \mathbb{E}\left[\mu_{\theta^{(0)}}^2\right]$ *for* $i \neq j$

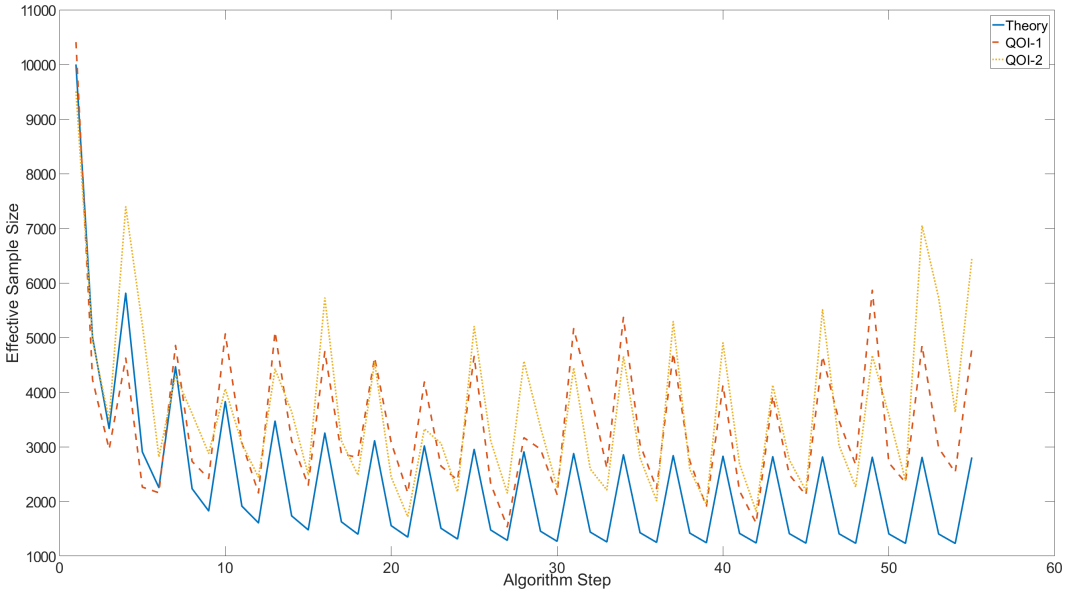

Figure 3.5: Trajectory of the ESS during a run of ST-MCMC for two quantities of interest, along with the theoretical trajectory.

Proof.

$$\begin{aligned}
\mathbb{E}\left[\theta_i^{(1)}\theta_j^{(1)}\right] &= \mathbb{E}\left[\mathbb{E}\left[\theta_i^{(1)}\theta_j^{(1)} \mid \theta^{(0)}\right]\right] \\
&= \mathbb{E}\left[\mathbb{E}\left[\theta_i^{(1)} \mid \theta^{(0)}\right]\mathbb{E}\left[\theta_j^{(1)} \mid \theta^{(0)}\right]\right] \\
&= \mathbb{E}\left[\left(\sum_{i=1}^{N} \hat{w}_i \theta_i^{(0)}\right)\left(\sum_{j=1}^{N} \hat{w}_j \theta_j^{(0)}\right)\right] \\
&= \mathbb{E}\left[\mu_{\theta^{(0)}}^2\right]
\end{aligned} \tag{3.17}$$

□

Theorem 1.

$$N_{ESS_1} = N\frac{N_{ESS_0}}{N_{ESS_0} + N - 1} \tag{3.18}$$

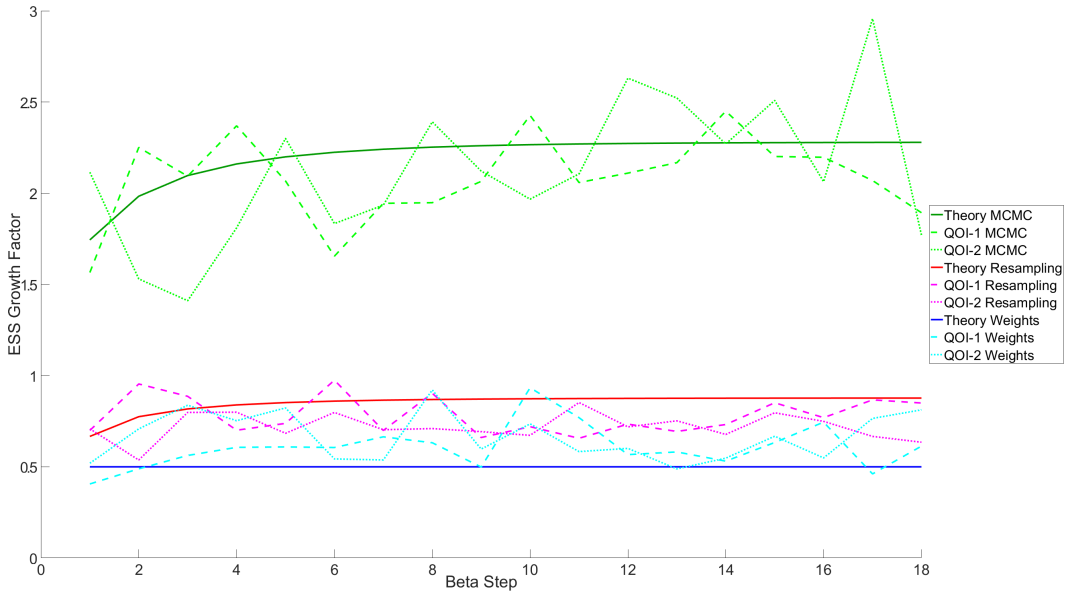

Figure 3.6: Change factor in the ESS for each step in the ST-MCMC algorithm for the two quantities of interest along with the theoretical change factor. Green corresponds to the MCMC step, red the resampling step, and blue the importance weighting step.

Proof.

$$
\begin{aligned}
var\left(\mu_{\theta^{(1)}}\right) &= \mathbb{E}\left[\left(\frac{1}{N}\sum_{i=1}^{N}\theta_i^{(1)}\right)\left(\frac{1}{N}\sum_{j=1}^{N}\theta_j^{(1)}\right)\right] - \bar{\theta}^2 \\
&= \mathbb{E}\left[\frac{1}{N^2}\sum_{i=1}^{N}\left(\theta_i^{(1)}\right)^2\right] + \mathbb{E}\left[\frac{1}{N^2}\sum_{i,j=1 i\neq j}^{N}\theta_i^{(1)}\theta_j^{(1)}\right] - \bar{\theta}^2 \\
&= \frac{1}{N}\mathbb{E}\left[\left(\theta_i^{(1)}\right)^2\right] + \frac{N-1}{N}\mathbb{E}\left[\theta_i^{(1)}\theta_j^{(1)}\right] - \bar{\theta}^2 \qquad \text{for some } i, j \text{ s.t. } i \neq j \\
&= \frac{1}{N}var\left(\theta_i^{(1)}\right) + \frac{N-1}{N}\mathbb{E}\left[\mu_{\theta^{(0)}}^2\right] - \frac{N-1}{N}\bar{\theta}^2 \\
&= \frac{1}{N}var\left(\theta_i^{(1)}\right) + \frac{N-1}{N}var\left(\mu_{\theta^{(0)}}\right) \\
&\implies \frac{1}{N_{ESS_1}} = \frac{1}{N} + \frac{N-1}{N}\frac{1}{N_{ESS_0}} \\
&\implies N_{ESS_1} = N\frac{N_{ESS_0}}{N_{ESS_0} + N - 1}
\end{aligned}
\tag{3.18}
$$

□

ESS for MCMC

Let $\theta_i^{(0)}$ be a sample from the initial population after the resampling step and $\theta_i^{(1)}$ be that same sample after the MCMC step. We can define $\theta_i^{(1)}$:

$$\theta_i^{(1)} = \theta_i^{(0)} + x_i \tag{3.19}$$

Assumption 1. *$\theta_i^{(0)}$ and $\theta_j^{(1)}$ are identically distributed.*

In the language of MCMC this means there is no burn-in required since $\theta^{(0)}$ and $\theta^{(1)}$ follow the same stationary distribution.

Lemma 2. $\mathbb{E}\left[\theta_i^{(0)} x_i\right] = (\rho - 1)\, var\left(\theta_i^{(0)}\right)$

Proof. Noting that $\theta_i^{(0)}$ and $\theta_i^{(1)}$ follow the same distribution and that therefore $\mathbb{E}[x_i]$ is zero.

$$\begin{aligned}
\rho &= corr\left(\theta_i^{(0)}, \theta_i^{(1)}\right) \\
&= \frac{cov\left(\theta_i^{(0)}, \theta_i^{(0)} + x_i\right)}{var\left(\theta_i^{(0)}\right)} \\
&= \frac{var\left(\theta_i^{(0)}\right) + cov\left(\theta_i^{(0)}, x_i\right)}{var\left(\theta_i^{(0)}\right)} \\
&= 1 + \frac{\mathbb{E}\left[\theta_i^{(0)} x_i\right] - \mathbb{E}\left[\theta_i^{(0)}\right]\mathbb{E}[x_i]}{var\left(\theta_i^{(0)}\right)} \\
&= 1 + \frac{\mathbb{E}\left[\theta_i^{(0)} x_i\right]}{var\left(\theta_i^{(0)}\right)} \\
&\implies \mathbb{E}\left[\theta_i^{(0)} x_i\right] = (\rho - 1)\, var\left(\theta_i^{(0)}\right)
\end{aligned} \tag{3.20}$$

□

The negative correlation between $\theta_i^{(0)}$ and x_i can be intuitively understood in the case of a Gaussian distribution. First, recall that x_i is the change in position for a Markov chain starting at $\theta_i^{(0)}$ and ending at $\theta_i^{(1)}$. In this case the stationary distribution is a Gaussian, so the expected direction of the movement of the chain would be away from $\theta_i^{(0)}$ toward the mean of the Gaussian. This yields the negative correlation.

Assumption 2. $\mathbb{E}\left[\theta_j^{(0)} \mid \theta_i^{(0)}\right] = (1-\alpha)\,\bar{\theta} + \alpha\theta_i^{(0)}$ *for* $i \neq j$ *and some* $\alpha \in [0,1]$

This assumption corresponds to multinomial resampling of an i.i.d. population where it is an exact result, since the resampling occurs with replacement, so some of the population $\theta^{(0)}$ are identical copies of other samples in the population. Therefore, with some probability α, $\theta_j^{(0)}$ and $\theta_i^{(0)}$ are the same. However, with probability $1-\alpha$, they are independent, so the expectation is just the mean of θ, $\mathbb{E}\left[\theta_j^{(0)}\right] = \bar{\theta}$.

Lemma 3. *If the population* $\theta^{(0)}$ *has an effective sample size of* N_{ESS_0} *then* $\alpha = \frac{N - N_{ESS_0}}{N_{ESS_0}(N-1)}$

Proof. When ${}_0\hat{\theta}$ is the mean estimate, the effective sample size can be defined by

$$\frac{1}{N_{ESS_0}} = \frac{var\left(\mu_{\theta^{(0)}}\right)}{var\left(\theta_i^{(0)}\right)} \tag{3.21}$$

Using the fact that all $\theta_i^{(0)}$ follow the same distribution, along with Equation 3.21 and Assumption 2:

$$\begin{aligned}
\frac{var\left(\theta_i^{(0)}\right)}{N_{ESS_0}} &= var\left(\mu_{\theta^{(0)}}\right) \\
&= \mathbb{E}\left[\left(\frac{1}{N}\sum_{i=1}^{N}\theta_i^{(0)}\right)\left(\frac{1}{N}\sum_{j=1}^{N}\theta_j^{(0)}\right)\right] - \bar{\theta}^2 \\
&= \frac{var\left(\theta_i^{(0)}\right)}{N} + \frac{N-1}{N}\mathbb{E}\left[\theta_i^{(0)}\theta_j^{(0)}\right] - \frac{N-1}{N}\bar{\theta}^2 \qquad \text{for some } i, j \text{ s.t. } i \neq j \\
&= \frac{var\left(\theta_i^{(0)}\right)}{N} + \frac{N-1}{N}\mathbb{E}\left[\theta_i^{(0)}\mathbb{E}\left[\theta_j^{(0)} \mid \theta_i^{(0)}\right]\right] - \frac{N-1}{N}\bar{\theta}^2 \\
&= \frac{var\left(\theta_i^{(0)}\right)}{N} + \frac{N-1}{N}\mathbb{E}\left[\theta_i^{(0)}\left[(1-\alpha)\,\bar{\theta} + \alpha\theta_i^{(0)}\right]\right] - \frac{N-1}{N}\bar{\theta}^2 \\
&= \left(\frac{1}{N} + \alpha\frac{N-1}{N}\right) var\left(\theta_i^{(0)}\right) \\
&\implies \alpha = \frac{N - N_{ESS_0}}{N_{ESS_0}\,(N-1)}
\end{aligned} \tag{3.22}$$

□

Lemma 4. $\mathbb{E}\left[\theta_i^{(0)} x_j\right] = \frac{N - N_{ESS_0}}{N_{ESS_0}(N-1)} (\rho - 1)\, var\left(\theta_i^{(0)}\right)$ *for* $i \neq j$

Proof. By applying the assumption and previous lemmas and noting $\mathbb{E}\left[x_j\right] = 0$:

$$\begin{aligned}
\mathbb{E}\left[\theta_i^{(0)} x_j\right] &= \mathbb{E}\left[\mathbb{E}\left[\theta_i^{(0)} x_j \mid \theta_j^{(0)}\right]\right] \\
&= \mathbb{E}\left[\mathbb{E}\left[x_j \mid \theta_j^{(0)}\right] \mathbb{E}\left[\theta_i^{(0)} \mid \theta_j^{(0)}\right]\right] \\
&= \mathbb{E}\left[\mathbb{E}\left[x_j \mid \theta_j^{(0)}\right] (1-\alpha)\,\bar{\theta} + \alpha \theta_j^{(0)}\right] \\
&= (1-\alpha)\,\bar{\theta}\mathbb{E}\left[x_j\right] + \alpha \mathbb{E}\left[\theta_j^{(0)} x_j\right] \\
&= \frac{N - N_{ESS_0}}{N_{ESS_0}(N-1)} (\rho - 1)\, var\left(\theta_i^{(0)}\right)
\end{aligned} \tag{3.23}$$

□

Assumption 3. $\mathbb{E}\left[x_i \mid \theta_i^{(0)}\right] = \frac{cov\left(x_i, \theta_i^{(0)}\right)}{var\left(\theta_i^{(0)}\right)} \left(\theta_i^{(0)} - \bar{\theta}\right) = \left(\theta_i^{(0)} - \bar{\theta}\right)(\rho - 1)$

Derived from a linear assumption on the conditional expectation i.e. that $\mathbb{E}[x \mid \theta] = a\theta + b$. *The values of a and b are found from the Law of Total Variance, Lemma 1, and the fact that* $\mathbb{E}\left[x_j\right] = 0$.

The linear assumption on the conditional expectation holds in the diffusion process limit of MCMC, described in [R+01], where the underlying posterior is a Gaussian. In this case, the diffusion limit is an Ornstein-Uhlenbeck process making x and θ jointly Gaussian. Jointly Gaussian random variables satisfy the linear assumption on the conditional expectation.

Lemma 5. $\mathbb{E}\left[x_i x_j\right] = \frac{N - N_{ESS_0}}{N_{ESS_0}(N-1)} (\rho - 1)^2\, var\left(\theta_i^{(0)}\right)$ *for* $i \neq j$

Proof. By applying the assumption and previous lemmas:

$$
\begin{aligned}
\mathbb{E}\left[x_i x_j\right] &= \mathbb{E}\left[\mathbb{E}\left[x_i x_j \mid \theta_i^{(0)}, \theta_j^{(0)}\right]\right] \qquad \text{for some } i, j \text{ s.t. } i \neq j \\
&= \mathbb{E}\left[\mathbb{E}\left[x_j \mid \theta_j^{(0)}\right] \mathbb{E}\left[x_i \mid \theta_i^{(0)}\right]\right] \\
&= \mathbb{E}\left[\left(\theta_i^{(0)} - \bar{\theta}\right)\left(\theta_j^{(0)} - \bar{\theta}\right)(\rho - 1)^2\right] \\
&= \left(\mathbb{E}\left[\theta_i^{(0)} \theta_j^{(0)}\right] - \bar{\theta}^2\right)(\rho - 1)^2 \\
&= \left(\mathbb{E}\left[\theta_i^{(0)} \mathbb{E}\left[\theta_j^{(0)} \mid \theta_i^{(0)}\right]\right] - \bar{\theta}^2\right)(\rho - 1)^2 \\
&= \left(\mathbb{E}\left[\theta_i^{(0)}\left((1 - \alpha)\,\bar{\theta} + \alpha \theta_i^{(0)}\right)\right] - \bar{\theta}^2\right)(\rho - 1)^2 \\
&= \alpha\,(\rho - 1)^2\, var\left(\theta_i^{(0)}\right)
\end{aligned}
\tag{3.24}
$$

□

Theorem 2.

$$
N_{ESS_1} = N_{ESS_0} \frac{1}{\left(1 - \frac{N_{ESS_0}}{N}\right)\rho^2 + \frac{N_{ESS_0}}{N}} \tag{3.25}
$$

Proof. Putting everything together:

$$
\begin{aligned}
var\left(\mu_{\theta^{(1)}}\right) &= \mathbb{E}\left[\left(\frac{1}{N}\sum_{i=1}^{N}\theta_i^{(1)}\right)\left(\frac{1}{N}\sum_{j=1}^{N}\theta_j^{(1)}\right)\right] - \bar{\theta}^2 \\
&= \mathbb{E}\left[\left(\frac{1}{N}\sum_{i=1}^{N}\left(\theta_i^{(0)}+x_i\right)\right)\left(\frac{1}{N}\sum_{j=1}^{N}\left(\theta_j^{(0)}+x_j\right)\right)\right] - \bar{\theta}^2 \\
&= \mathbb{E}\left[\left(\frac{1}{N}\sum_{i=1}^{N}\theta_i^{(0)}\right)\left(\frac{1}{N}\sum_{j=1}^{N}\theta_j^{(0)}\right)\right] - \bar{\theta}^2 + \mathbb{E}\left[\frac{2}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}\theta_i^{(0)}x_j\right] \\
&\qquad + \mathbb{E}\left[\frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}x_i x_j\right] \\
&= var\left(\mu_{\theta^{(0)}}\right) + \frac{2}{N^2}\mathbb{E}\left[\sum_{i=1}^{N}\theta_i^{(0)}x_i\right] + \frac{2}{N^2}\mathbb{E}\left[\sum_{i,j=1 i\neq j}^{N}\theta_i^{(0)}x_j\right] \\
&\qquad + \frac{1}{N^2}\mathbb{E}\left[\sum_{i=1}^{N}x_i^2\right] + \frac{1}{N^2}\mathbb{E}\left[\sum_{i,j=1 i\neq j}^{N}x_i x_j\right] \\
&= var\left(\mu_{\theta^{(0)}}\right) + \frac{1}{N}\mathbb{E}\left[2\theta_i^{(0)}x_i + x_i^2\right] + \frac{2\left(N-1\right)}{N}\mathbb{E}\left[\theta_i^{(0)}x_j\right] + \frac{\left(N-1\right)}{N}\mathbb{E}\left[x_i x_j\right] \\
&= var\left(\mu_{\theta^{(0)}}\right) + \frac{1}{N}\mathbb{E}\left[\left(\theta_i^{(1)}\right)^2 - \left(\theta_i^{(0)}\right)^2\right] \\
&\qquad + \frac{2\left(N-1\right)}{N}\frac{N-N_{ESS_0}}{N_{ESS_0}\left(N-1\right)}\left(\rho-1\right)var\left(\theta_i^{(0)}\right) \\
&\qquad + \frac{\left(N-1\right)}{N}\frac{N-N_{ESS_0}}{N_{ESS_0}\left(N-1\right)}\left(\rho-1\right)^2 var\left(\theta_i^{(0)}\right) \\
&= var\left(\mu_{\theta^{(0)}}\right) + \frac{2\left(N-N_{ESS_0}\right)}{NN_{ESS_0}}\left(\rho-1\right)var\left(\theta_i^{(0)}\right) \\
&\qquad + \frac{N-N_{ESS_0}}{NN_{ESS_0}}\left(\rho-1\right)^2 var\left(\theta_i^{(0)}\right) \\
\implies \frac{1}{N_{ESS_1}} &= \frac{1}{N_{ESS_0}} + \frac{2\left(N-N_{ESS_0}\right)}{NN_{ESS_0}}\left(\rho-1\right) + \frac{N-N_{ESS_0}}{NN_{ESS_0}}\left(\rho-1\right)^2 \\
\implies N_{ESS_1} &= N_{ESS_0}\frac{1}{\left(1-\frac{N_{ESS_0}}{N}\right)\rho^2 + \frac{N_{ESS_0}}{N}}
\end{aligned}
\tag{3.25}
$$

□

3.4 Optimal Acceptance Rate Adaptation for Sequential Tempered MCMC

Many MCMC methods used within ST-MCMC, like random walk Metropolis, rely on a Gaussian proposal distribution. Tuning this proposal distribution as ST-MCMC progresses through the intermediate distributions is an important challenge. The optimal acceptance rate for Metropolis Hastings MCMC can be found for certain classes of distributions. For a Gaussian distribution in high dimensions, the optimal acceptance rate is approximately 0.234 when the proposal distribution has a variance that is a scaled version of the variance of the desired Gaussian distribution [R+01; Ros+11]. This leads to a variance scaling factor of $\frac{2.38^2}{N_d}$, where N_d is the dimension. This optimal acceptance rate holds for a large class of high dimensional distributions with independent components and is often used as the target acceptance rate for Metropolis MCMC algorithms [R+01]. In order to tune the Metropolis sampler, we can develop a feedback control strategy to tune the covariance matrix scaling of the Gaussian proposal to produce the target acceptance rate of 0.234. This feedback controller is designed using the linearization of the dynamics relating the scaling factor of the proposal Gaussian and the theoretical acceptance rate for sampling a Gaussian distribution. When applying this controller to AlTar/CATMIP, we assume that the optimal value of the scaling factor does not change quickly as the probability distribution evolves through the intermediate levels of the algorithms, judged relative to the speed at which the acceptance rate converges to the target value using the controller.

The feedback controller is described in Algorithm 2. We assume a zero-mean Gaussian proposal distribution and that the covariance matrix used for the proposal is close to the covariance of the distribution we are sampling from. In AlTar/CATMIP, this is done by taking the proposal covariance matrix to be a scaled copy of the sample covariance. We then try to find a factor, σ^2, to scale the covariance matrix, Σ, i.e. the proposal distribution has the form $\mathcal{N}\left(0, \sigma^2\Sigma\right)$. The algorithm begins by initializing the scaling factor σ^2 to be the optimal scaling factor if the distribution where Gaussian and also by setting a target acceptance rate α^*. In the next step, the MCMC part is run for level k. This returns the population level estimate of the acceptance rate for that level. Then the feedback controller adjusts the scaling factor based upon the difference between the target and actual acceptance rate. The feedback controller is designed for $\log\sigma$ which allows us to look at this as a linear system. The feedback constant, $G = 2.1$, in Algorithm 2 was chosen based upon studying the dynamics of this system.

Under the assumption of a posterior target distribution with i.i.d. components and a smooth target density described in [R+01; Ros+11] in the diffusion limit, the asymptotic acceptance rate for random walk Metropolis with proposal distribution $\mathcal{N}\left(0, \sigma^2 I_{N_d}\right)$ is $\alpha(l) = 2\Phi\left(-\frac{\sqrt{I}}{2} l\right)$. Here, $\sigma = l/\sqrt{N_d}$ and Φ is the cumulative distribution function of a standard Gaussian, while I is a constant that depends on the posterior. They show that this function is optimized when $l_{opt} \approx 2.38/\sqrt{I}$, which leads to an optimal acceptance rate of $\alpha^* = 0.234$ [R+01; Ros+11]. Assuming the posterior in the inference problem has been appropriately transformed and decorrelated using the sample covariance, we can use these results to design a feedback controller to tune σ to achieve this target acceptance rate.

Since the scale factor σ is always positive, we can design a linear feedback controller for $\log \sigma$ in the form of

$$\log \sigma_{k+1} = \log \sigma_k + G\left[\alpha(\log \sigma_k) - \alpha^*\right] \tag{3.26}$$

In order to choose the feedback constant G, we first formulate the acceptance rate in terms of the current scaling factor using the complementary error function:

$$\alpha(\log \sigma_k) = \operatorname{erfc}\left(\frac{\sqrt{I N_d}\exp(\log \sigma_k)}{2\sqrt{2}}\right) \tag{3.27}$$

Then linearizing the dynamics (3.26) around the optimal choice of the scaling factor, $\log \sigma_{opt}$, which has an acceptance rate of α^*, yields

$$\log \sigma_{k+1} = \log \sigma_k + f(G, \alpha^*)\left(\log \sigma_k - \log \sigma_{opt}\right) \tag{3.28}$$

The linear factor $f(G, \alpha^*)$ describes how much a small perturbation away from the optimal scaling factor decays or grows over time. When $|f(G, \alpha^*)| < 1$ the linear dynamics are asymptotically stable so the feedback controller will tune the scaling factor to reach the target acceptance rate. The functional form of $f(G, \alpha^*)$ is

$$f(G, \alpha^*) = 1 - \frac{2G\exp\left[-\operatorname{erfc}^{-1}(\alpha^*)^2\right]\operatorname{erfc}^{-1}(\alpha^*)}{\sqrt{\pi}} \tag{3.29}$$

The stable region for this linear system is plotted as the shaded region in Figure 3.7 for different acceptance rate targets. The dynamical system is critically damped when $f(G, \alpha^*) = 0$. This occurs when $G = \frac{\sqrt{\pi}\exp\left[-\operatorname{erfc}^{-1}(\alpha^*)^2\right]}{2\operatorname{erfc}^{-1}(\alpha^*)}$. This optimal choice

of gain G corresponds to the red line in Figure 3.7. Critical damping allows the controller to reach the target the fastest without overshoot. For comparison, the trajectory of an underdamped, critically damped, and overdamped controller are shown in Figure 3.8. Finally, solving for $f(G, \alpha^*) = 0$ when $\alpha^* = 0.234$ yields $G \approx 2.1$.

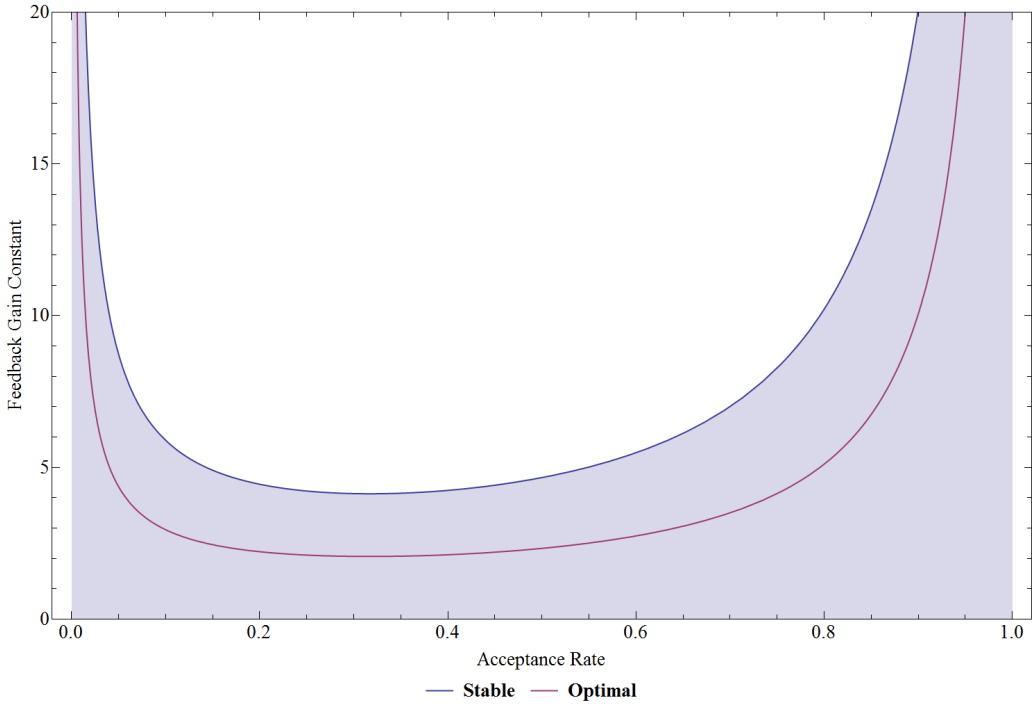

Figure 3.7: Stability region of the linearized system for different target acceptance rates. The optimal rate corresponds to critical damping.

Algorithm 2: MCMC Feedback Controller
Step 0: Initialize scaling factor σ^2, target acceptance rate α^*, and feedback gain G **Set** $\sigma_1 = \frac{2.38}{\sqrt{N_d}}$ **Set** $\alpha^* = 0.234$ **Set** $G = 2.1$ **Step k.1:** AlTar Level k MCMC **Get** σ_k **Return** Acceptance rate α_k **Step k.2:** Compute new scaling factor **Set** $\sigma_{k+1} = \sigma_k \exp[G(\alpha_k - \alpha^*)]$

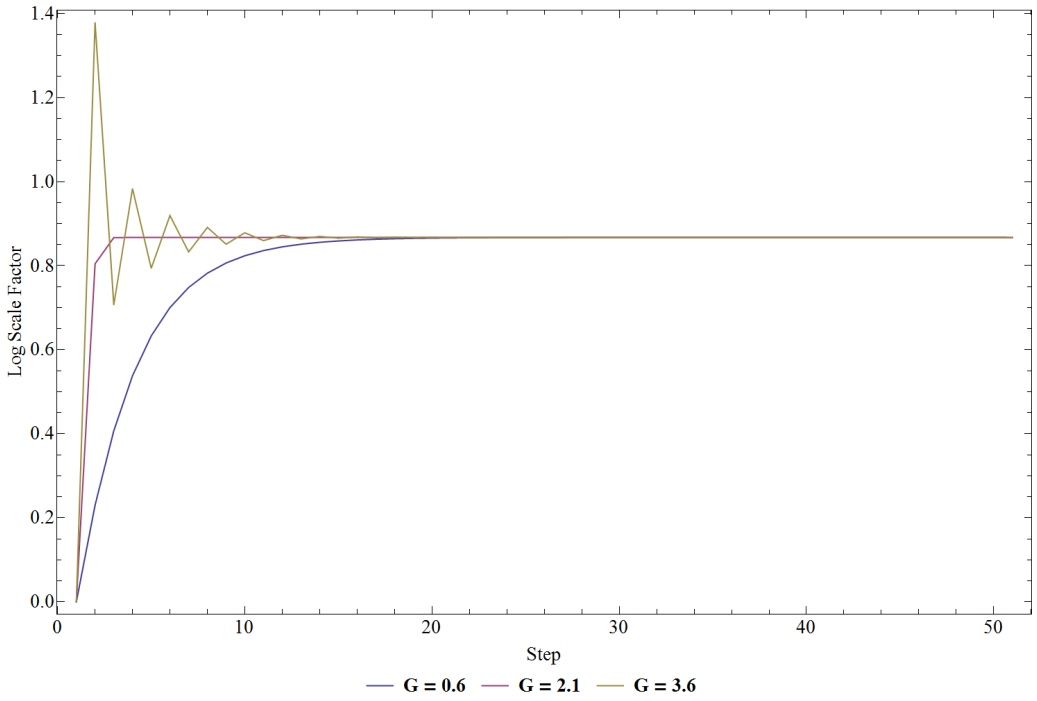

Figure 3.8: Comparison of log scaling factor trajectories for different feedback controller gains corresponding to underdamped (yellow), critically damped (red), and overdamped (blue).

3.5 Sequential Tempered MCMC based on the Modified Metropolis Algorithm

For most inference problems, the most computationally significant step of ST-MCMC is evolving the population of Markov chains during the MCMC step for each intermediate distribution. The computational cost is particularly high for high dimensional inference problems with priors which enforce constraints on the parameters, such as using uniform prior. This high cost is because if a candidate step in the Markov chain is proposed in high dimensions without knowledge of the constraints, it is highly likely that one of the constrains will be violated. Therefore, developing a proposal method that can integrate this prior information is important to developing an efficient ST-MCMC method for these types of problems. The Modified Metropolis Algorithm (MMA), developed in [AB01], addresses this problem under the assumption that the proposal distribution is a Gaussian with independent variables and that the prior has independent components. The Rank One Modified Metropolis Algorithm, presented in this thesis, generalizes the MMA algorithm to any prior distribution and any Gaussian proposal distribution.

Modified Metropolis Algorithm

The Modified Metropolis Algorithm was developed in [AB01] to overcome the curse of dimensionality that comes from sampling high dimensional posteriors when estimating small failure probabilities. This algorithm originally assumed that the posterior is the product of the prior and an indicator function but can be expanded to the more general Bayesian inference setting as in Algorithm 3. The algorithm assumes that the prior distribution $\pi(\theta)$ has independent components such that $\pi(\theta) = \prod_{j=1}^{N_d} \pi_j\left(\theta_j\right)$. In order to evolve the Markov chain that samples the posterior $p(\theta \mid \mathcal{D})$, the authors break it up into a two-step proposal. The first step of the proposal deals with the prior and can be done component-wise. The second step deals with the likelihood $p(\mathcal{D} \mid \theta)$, which is done by evaluating the forward model. By separating out the prior and evolving it component-wise, this algorithms avoids the poor dimensional scaling introduced by the prior. This is particularly important for priors with bounded support because they often have a significant impact on the posterior. This algorithm was developed for the case where the proposal distribution also has independent components.

Rank One Modified Metropolis Algorithm (ROMMA)

We develop a similar two-step proposal process for a more general setting where the proposal and prior may not correspond to independent variables. In particular, we study the case of a multivariate Gaussian proposal and general prior. The key idea is that instead of thinking of the algorithm as a set of component-wise proposals, think of it as a set of linearly independent rank one proposals. By employing this algorithm, we can significantly reduce the number of forward model evaluations. The tradeoff is that this algorithm requires an increasing number of prior evaluations, which scales linearly with dimension, and it is sensitive to the proposal covariance used to generate the rank one updates. However, by using this inside of a ST-MCMC algorithm, the covariance structure is well estimated by the samples and the scaling can be adaptively tuned.

The Rank One Modified Metropolis Algorithm (ROMMA) for general MCMC is described in Algorithm 4. In this algorithm, the correlation structure in the Gaussian proposal is handled using the matrix square root of the correlation matrix. The algorithm begins by computing the matrix square root, S, of the proposal covariance, Σ; however, in principle any matrix decomposition may be used. From this, we need two permutation matrices, P_+ and P_-. P_+ is the identity matrix and corresponds to performing the rank one updates in the forward direction while P_-

Algorithm 3: Modified Metropolis Algorithm

Step 0:

Set D ; // Diagonal positive definite matrix

Set N_d ; // Number of components of the vector θ

Set N_{steps} ; // Number of steps in the Markov chain

Step k:

for $i = 1$ **to** N_{steps} **do**

  Draw $\xi \sim \mathcal{N}\left(0, I_{N_d}\right)$

  for $j = 1$ **to** N_d **do**

    $\nu_j = D_j \xi_j$

    $\hat{\theta}_j = \theta^i_j + \nu_j$; // Perform component update

    Draw $\zeta \sim \mathcal{U}[0, 1]$

    /* Accept or Reject the component update according to the component prior likelihood */

    if $\zeta > \frac{\pi_j\left(\hat{\theta}_j\right)}{\pi_j\left(\theta^i_j\right)}$ **then**

      $\xi_j = 0$

    end

  end

  $\hat{\theta} = \theta^i + D\xi$

  Draw $\eta \sim \mathcal{U}[0, 1]$

  /* Accept or Reject the full update according to the data likelihood */

  if $\eta < \frac{p\left(\mathcal{D}|\hat{\theta}\right)}{p\left(\mathcal{D}|\theta^i\right)}$ **then**

    $\theta^{i+1} = \hat{\theta}$

  else

    $\theta^{i+1} = \theta^i$

  end

end

corresponds to reversing or flipping the indices of the variables and performing the updates in the reverse direction. Using these two permutation matrices is necessary to produce a reversible sampler.

Then for each step in the Markov chain, we initialize the candidate $\hat{\theta}$ to be the current sample θ^i and randomly choose the permutation P to be the forward or reverse ordering with equal probability. Based upon the choice of the permutation, the transformed matrix square root R is formed. The i$^{\text{th}}$ column of R, R_i, will be the i$^{\text{th}}$ rank one update. Finally, we draw a random standard Normal vector ξ, as when generating a zero-mean multivariate Gaussian with transformed covariance $P\Sigma P^T$ using $S\xi \sim \mathcal{N}\left(0, P\Sigma P^T\right)$.

Iterating through all of the N_d rank one updates, we construct a proposed candidate $\tilde{\theta}$ based upon the current rank one update vector R_i, as $\tilde{\theta} = \hat{\theta} + R_i\xi_i$. We then compute the ratio of the priors, $\frac{\pi(\tilde{\theta})}{\pi(\hat{\theta})}$ and choose whether to accept or reject the proposed rank one change according to a Metropolis step. If the component is rejected, $\hat{\theta}$ remains the same, else $\hat{\theta}$ is updated to $\tilde{\theta}$. These two steps are performed for all rank one updates until we reach the final $\hat{\theta}$. This set of rank one proposals can be thought of as evolving the Markov chain according to the prior since the prior distribution would be the invariant distribution of this Markov chain in the absence of the likelihood evaluation step that follows.

After choosing $\hat{\theta}$, we then perform a Metropolis step to accept or reject the entire vector $\hat{\theta}$ according to the likelihood $p\left(\mathcal{D} \mid \hat{\theta}\right)$. Thus, we compute the ratio of the likelihood for the candidate and current parameter vectors $\frac{p(\mathcal{D}|\hat{\theta})}{p(\mathcal{D}|\theta^i)}$. If the sample is accepted, then $\theta^{i+1} = \hat{\theta}$, else $\theta^{i+1} = \theta^i$.

Tuning ROMMA

The scaling for the spread of a MCMC proposal distribution is typically tuned by trying to find a scaling factor that achieves an acceptance rate target. In ROMMA, because there are multiple Metropolis steps, finding the appropriate definition of the acceptance rate is non-trivial. For example, having the function that relates the scaling factor to the acceptance rate be monotonic is important for many of the tuning algorithms to achieve a target acceptance rate that corresponds to a scaling factor that is neither too large or too small and thus induces low correlation. However, the acceptance rate in the second part of the ROMMA algorithm does not have this property. This is demonstrated in Figure 3.9 for the constrained Bayesian logistic regression problem in Section 3.6. When the scale factor is small, the second Metropolis step acceptance rate is high and generally decreases. Then once the scale factor gets sufficiently large, the acceptance rate starts to increase again since most of the rank one proposals in the first step of the algorithm are now getting rejected. This causes the bifurcation of the correlation with respect to the acceptance rate seen in the right panel.

An alternative definition of the acceptance rate is to look at the acceptance rate for a specific rank one component. This means the probability that a specific component proposal is accepted during step one of the ROMMA algorithm and also accepted as part of the combined candidate in step 2. Since there are multiple

Algorithm 4: Rank One Modified Metropolis Algorithm (ROMMA)

Step 0:

Set $S = \sqrt{\Sigma}$; // Square root of the proposal covariance Σ

Set $P_+ = I$; // Matrix for forward parameter ordering

Set $P_- = Flip(I)$; // Matrix for reverse parameter ordering

Set N_d ; // Number of components of the vector θ

Set N_{steps} ; // Number of steps in the Markov chain

Step k:

for $i = 1$ **to** N_{steps} **do**

  Draw $\xi \sim \mathcal{N}(0, I_{N_d})$

  Draw $\eta_1 \sim \mathcal{U}[0,1]$

  /* Randomly choose forward or reverse component ordering */

  if $\eta_1 < \frac{1}{2}$ **then**

    $P = P_+$

  else

    $P = P_-$

  end

  Set $R = PSP^T$; // Compute the transformed components

  Set $\hat{\theta} = \theta^i$

  for $j = 1$ **to** N_d **do**

    $\tilde{\theta} = \hat{\theta} + PR_j\xi_j$; // Perform rank one update

    Draw $\eta_2 \sim \mathcal{U}[0,1]$

    /* Accept or Reject the rank one update according to the prior likelihood */

    if $\eta_2 < \frac{\pi(\tilde{\theta})}{\pi(\hat{\theta})}$ **then**

      $\hat{\theta} = \tilde{\theta}$

    end

  end

  Draw $\eta_3 \sim \mathcal{U}[0,1]$

  /* Accept or Reject the full update according to the data likelihood */

  if $\eta_3 < \frac{p(\mathcal{D}|\hat{\theta})}{p(\mathcal{D}|\theta^i)}$ **then**

    $\theta^{i+1} = \hat{\theta}$

  else

    $\theta^{i+1} = \theta^i$

  end

end

rank one components, we take the minimum acceptance rate among all of them. As demonstrated by Figure 3.9, this quantity is generally monotonic since as the scale factor grows very large, the higher acceptance rate in step 2 is balanced by the higher

rejection rate in step 1. Since it is monotonic, it is a much better tuning mechanism to find a scaling factor that leads to low correlation. In the example of Figure 3.9, the correlation is minimized when the acceptance rate is close to 0.25.

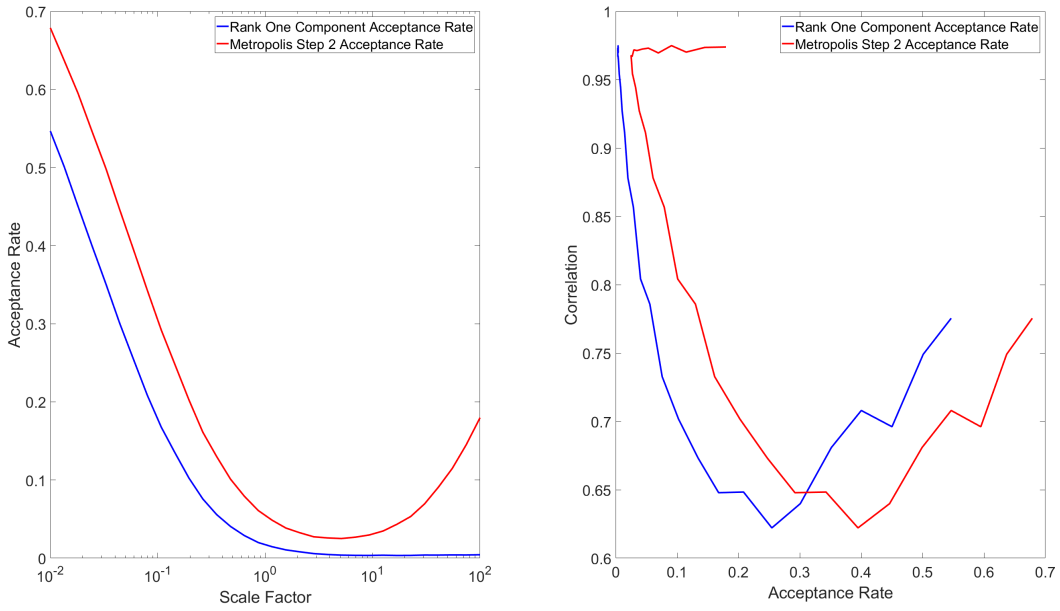

Figure 3.9: Empirical relationship between proposal scaling factor and acceptance rate (left panel) and acceptance rate and correlation (right panel) for different acceptance rate values.

Proof of Reversibility

The ROMMA MCMC Markov process step from θ to $\hat{\theta}$, with transition distribution denoted by $Q\left(\hat{\theta} \mid \theta\right)$, forms a reversible Markov chain whose invariant measure is the posterior distribution $p\left(\theta \mid \mathcal{D}\right) \propto p\left(\mathcal{D} \mid \theta\right) \pi\left(\theta\right)$.

Theorem 3. *Reversibility:* $p\left(\mathcal{D} \mid \hat{\theta}\right) \pi\left(\hat{\theta}\right) Q\left(\theta \mid \hat{\theta}\right) = p\left(\mathcal{D} \mid \theta\right) \pi\left(\theta\right) Q\left(\hat{\theta} \mid \theta\right)$

Proof. Let $P\left(\theta \rightarrow \hat{\theta}\right)$ denote the probability density that describes moving from θ to $\hat{\theta}$ under the Markov chain proposal, then the transition density from θ to $\hat{\theta}$, $Q\left(\hat{\theta} \mid \theta\right)$, is:

$$
\begin{aligned}
Q\left(\hat{\theta} \mid \theta\right) &= \min\left(\frac{p\left(\mathcal{D} \mid \hat{\theta}\right)}{p\left(\mathcal{D} \mid \theta\right)}, 1\right) P\left(\theta \to \hat{\theta}\right) \\
&= \min\left(\frac{p\left(\mathcal{D} \mid \hat{\theta}\right)}{p\left(\mathcal{D} \mid \theta\right)}, 1\right)\left[\frac{1}{2} P\left(\theta \to \hat{\theta} \mid P_{+}\right) + \frac{1}{2} P\left(\theta \to \hat{\theta} \mid P_{-}\right)\right]
\end{aligned}
\tag{3.30}
$$

We define $\hat{\theta}_i$ to be the i$^{\text{th}}$ intermediate evolution step of the candidate $\hat{\theta}$ and $\tilde{\theta}_i$ to be the i$^{\text{th}}$ proposal step under the i$^{\text{th}}$ rank one update and $P\left(\hat{\theta}_{i-1} \to \hat{\theta}_i \mid P\right)$ as the full transition probability according to both the i$^{\text{th}}$ rank one update and the Metropolis accept/reject step:

$$
P\left(\hat{\theta}_0 = \theta \to \hat{\theta}_N = \hat{\theta} \mid P\right) = \prod_{i=1}^{N} P\left(\hat{\theta}_{i-1} \to \hat{\theta}_i \mid P\right)
\tag{3.31}
$$

Each factor in this product can be express in two different ways that depend on whether the candidate was accepted or rejected at the i$^{\text{th}}$ step:

$$
\begin{aligned}
&P\left(\hat{\theta}_{i-1} \to \hat{\theta}_i \mid P\right) \\
&\qquad = \begin{cases}
\min\left(\frac{\pi(\tilde{\theta}_i)}{\pi(\hat{\theta}_{i-1})}, 1\right) P\left(\hat{\theta}_{i-1} \to \tilde{\theta}_i \mid P\right) & \hat{\theta}_i \neq \hat{\theta}_{i-1} \\
\int_{\tilde{\theta}} \left(1 - \min\left(\frac{\pi(\tilde{\theta})}{\pi(\hat{\theta}_{i-1})}, 1\right)\right) P\left(\hat{\theta}_{i-1} \to \tilde{\theta} \mid P\right) d\tilde{\theta} & \hat{\theta}_i = \hat{\theta}_{i-1}
\end{cases}
\end{aligned}
\tag{3.32}
$$

If we assume the structure of the rank one proposals introduced in Algorithm 4, i.e. $\tilde{\theta}_i = \hat{\theta}_{i-1} + PR_i\xi_i$, we find:

$$P\left(\hat{\theta}_{i-1} \rightarrow \hat{\theta}_i \mid P\right)$$

$$= \begin{cases} \min\left(\frac{\pi\left(\hat{\theta}_{i-1}+P\vec{R}_i\xi_i\right)}{\pi\left(\hat{\theta}_{i-1}\right)}, 1\right) P\left(\xi_i \mid P\right) & \hat{\theta}_i \neq \hat{\theta}_{i-1} \\ \int_{\xi}\left(1-\min\left(\frac{\pi\left(\hat{\theta}_{i-1}+P\vec{R}_i\xi\right)}{\pi\left(\hat{\theta}_{i-1}\right)}, 1\right)\right) P\left(\xi \mid P\right) d\tilde{\theta} & \hat{\theta}_i = \hat{\theta}_{i-1} \end{cases} \tag{3.33}$$

The key insight into proving reversibility is that the rank one update $P\vec{R}_i \mid P_+$ is the same as $P\vec{R}_{N-i+1} \mid P_-$ so we can undo all the updates from $\theta \rightarrow \hat{\theta}$ update using the ordering implied by P_+ by applying the reverse ordering P_- or vice versa. This leads to:

$$\frac{P\left(\hat{\theta}_i \rightarrow \hat{\theta}_{i-1} \mid P_-\right)}{P\left(\hat{\theta}_{i-1} \rightarrow \hat{\theta}_i \mid P_+\right)}$$

$$= \begin{cases} \frac{\min\left(\frac{\pi\left(\hat{\theta}_i - P_-\vec{R}_{N-i+1}\xi_i\right)}{\pi\left(\hat{\theta}_i\right)}, 1\right) P(-\xi_i|P_-)}{\min\left(\frac{\pi\left(\hat{\theta}_{i-1}+P_+\vec{R}_i\xi_i\right)}{\pi\left(\hat{\theta}_{i-1}\right)}, 1\right) P(\xi_i|P_+)} & \hat{\theta}_i \neq \hat{\theta}_{i-1} \\ \frac{\int_{\xi}\left(1-\min\left(\frac{\pi\left(\hat{\theta}_i - P_-\vec{R}_{N-i+1}\xi\right)}{\pi\left(\hat{\theta}_i\right)}, 1\right)\right) P(-\xi|P_-) d\tilde{\theta}}{\int_{\xi}\left(1-\min\left(\frac{\pi\left(\hat{\theta}_{i-1}+P_+\vec{R}_i\xi\right)}{\pi\left(\hat{\theta}_{i-1}\right)}, 1\right)\right) P(\xi|P_+) d\tilde{\theta}} & \hat{\theta}_i = \hat{\theta}_{i-1} \end{cases} \tag{3.34}$$

$$= \begin{cases} \frac{\pi\left(\hat{\theta}_{i-1}\right)}{\pi\left(\hat{\theta}_i\right)} & \hat{\theta}_i \neq \hat{\theta}_{i-1} \\ \frac{\pi\left(\hat{\theta}_{i-1}\right)}{\pi\left(\hat{\theta}_i\right)} = 1 & \hat{\theta}_i = \hat{\theta}_{i-1} \end{cases}$$

$$= \frac{\pi\left(\hat{\theta}_{i-1}\right)}{\pi\left(\hat{\theta}_i\right)}$$

Therefore, we can put these results together to find:

$$\begin{aligned}
\frac{P\left(\hat{\theta}_0 = \hat{\theta} \rightarrow \hat{\theta}_N = \theta \mid P_-\right)}{P\left(\hat{\theta}_0 = \theta \rightarrow \hat{\theta}_N = \hat{\theta} \mid P_+\right)} &= \prod_{i=1}^{N} \frac{P\left(\hat{\theta}_i \rightarrow \hat{\theta}_{i-1} \mid P_-\right)}{P\left(\hat{\theta}_{i-1} \rightarrow \hat{\theta}_i \mid P_+\right)} \\
&= \prod_{i=1}^{N} \frac{\pi\left(\hat{\theta}_{i-1}\right)}{\pi\left(\hat{\theta}_i\right)} \\
&= \frac{\pi(\theta)}{\pi\left(\hat{\theta}\right)}
\end{aligned} \tag{3.35}$$

Substituting this result into the Markov chain transition probability ratio $\frac{Q(\theta|\hat{\theta})}{Q(\hat{\theta}|\theta)}$, we can prove the reversibility of the Markov chain:

$$\frac{Q\left(\theta \mid \hat{\theta}\right)}{Q\left(\hat{\theta} \mid \theta\right)} = \frac{\min\left(\frac{p(\mathcal{D}|\theta)}{p(\mathcal{D}|\hat{\theta})}, 1\right)\left[\frac{1}{2}P\left(\hat{\theta} \to \theta \mid P_-\right) + \frac{1}{2}P\left(\hat{\theta} \to \theta \mid P_+\right)\right]}{\min\left(\frac{p(\mathcal{D}|\hat{\theta})}{p(\mathcal{D}|\theta)}, 1\right)\left[\frac{1}{2}P\left(\theta \to \hat{\theta} \mid P_+\right) + \frac{1}{2}P\left(\theta \to \hat{\theta} \mid P_-\right)\right]}$$

$$= \frac{p\left(\mathcal{D} \mid \theta\right)}{p\left(\mathcal{D} \mid \hat{\theta}\right)} \frac{P\left(\hat{\theta} \to \theta \mid P_-\right) + P\left(\hat{\theta} \to \theta \mid P_+\right)}{P\left(\theta \to \hat{\theta} \mid P_+\right) + P\left(\theta \to \hat{\theta} \mid P_-\right)}$$

$$= \frac{p\left(\mathcal{D} \mid \theta\right)}{p\left(\mathcal{D} \mid \hat{\theta}\right)} \left[\frac{P\left(\hat{\theta} \to \theta \mid P_-\right)}{P\left(\theta \to \hat{\theta} \mid P_+\right) + P\left(\theta \to \hat{\theta} \mid P_-\right)} + \frac{P\left(\hat{\theta} \to \theta \mid P_+\right)}{P\left(\theta \to \hat{\theta} \mid P_-\right) + P\left(\theta \to \hat{\theta} \mid P_+\right)}\right]$$

$$= \frac{p\left(\mathcal{D} \mid \theta\right)}{p\left(\mathcal{D} \mid \hat{\theta}\right)} \left[\frac{P\left(\hat{\theta} \to \theta \mid P_-\right)}{P\left(\theta \to \hat{\theta} \mid P_+\right)} \frac{1}{1 + \frac{P\left(\theta \to \hat{\theta} | P_-\right)}{P\left(\theta \to \hat{\theta} | P_+\right)}} + \frac{P\left(\hat{\theta} \to \theta \mid P_+\right)}{P\left(\theta \to \hat{\theta} \mid P_-\right)} \frac{1}{1 + \frac{P\left(\theta \to \hat{\theta} | P_+\right)}{P\left(\theta \to \hat{\theta} | P_-\right)}}\right]$$

$$= \frac{p\left(\mathcal{D} \mid \theta\right)}{p\left(\mathcal{D} \mid \hat{\theta}\right)} \frac{\pi\left(\theta\right)}{\pi\left(\hat{\theta}\right)} \left[\frac{1}{1 + \frac{P\left(\theta \to \hat{\theta} | P_-\right)}{P\left(\theta \to \hat{\theta} | P_+\right)}} + \frac{1}{1 + \frac{P\left(\theta \to \hat{\theta} | P_+\right)}{P\left(\theta \to \hat{\theta} | P_-\right)}}\right]$$

$$= \frac{p\left(\mathcal{D} \mid \theta\right)}{p\left(\mathcal{D} \mid \hat{\theta}\right)} \frac{\pi\left(\theta\right)}{\pi\left(\hat{\theta}\right)} \tag{3.36}$$

$$\frac{Q\left(\theta \mid \hat{\theta}\right)}{Q\left(\hat{\theta} \mid \theta\right)} = \frac{p\left(\mathcal{D} \mid \theta\right)}{p\left(\mathcal{D} \mid \hat{\theta}\right)} \frac{\pi\left(\theta\right)}{\pi\left(\hat{\theta}\right)} \tag{3.37}$$

$$\implies p\left(\mathcal{D} \mid \hat{\theta}\right) \pi\left(\hat{\theta}\right) Q\left(\theta \mid \hat{\theta}\right) = p\left(\mathcal{D} \mid \theta\right) \pi\left(\theta\right) Q\left(\hat{\theta} \mid \theta\right)$$

□

3.6 Examples

Constrained Bayesian Logistic Regression

In order to construct a good test problem to evaluate Sequential Tempered MCMC and ROMMA, we use the German credit Bayesian Logistic Regression test problem with 24 features [GC11; MST94]. We are interested in evaluating how well the algorithm works in high dimensions and with a uniform prior that restricts the solution to a restricted domain, which are characteristic features of the two illustrative examples in this section. Therefore, we construct the problem here to have 49 dimensions, corresponding to finding regression coefficients for the offset, the 24 features, and the squared magnitude of the 24 features, and then place a uniform prior between -1 and 0 on each of these coefficients. The posterior distribution of the constrained problem is then sampled using ST-MCMC with three differen MCMC samplers: Random Walk Metropolis, the Modified Metropolis Algorithm, and the Rank-One Modified Metropolis Algorithm. Figure 3.10 shows that there is very good agreement between the histograms of each coefficient for each of the samplers. Further, since many of the marginal distributions are significantly skewed towards the boundary, the prior is having a large effect on the final posterior and thus we expect ROMMA to perform much better. This performance advantage is seen in Figure 3.11, which shows that ROMMA requires 61% of the model evaluations of MMA and 4% of the model evaluations of RWM.

This toy problem illustrates where ROMMA outperforms RWM. The uniform prior decreases the acceptance rate of RWM significantly compared to ROMMA, forcing the proposal distribution for RWM to be much tighter. However, when the prior is not very relevant to the posterior, which would be the case when the prior is unconstrained, there would be no performance advantage to using ROMMA over RWM. One possible strategy to improve the performance of ROMMA so that it outperforms RWM on problems where the prior is not informative relative to the likelihood is to use a fast-to-compute estimate of the posterior as the distribution used in step one of ROMMA so that information from the posterior is being used to choose the rank one components.

Static Finite-fault Geophysical Inverse Problem

The AlTar/CATMIP Sequential Tempered MCMC method was developed to solve inverse problems in geophysics [MSB13]. One problem discussed in [MSB13] is that of inferring a static source model for a finite fault. In this problem, we wish to infer the slip of the fault on discretized patches of the fault domain in

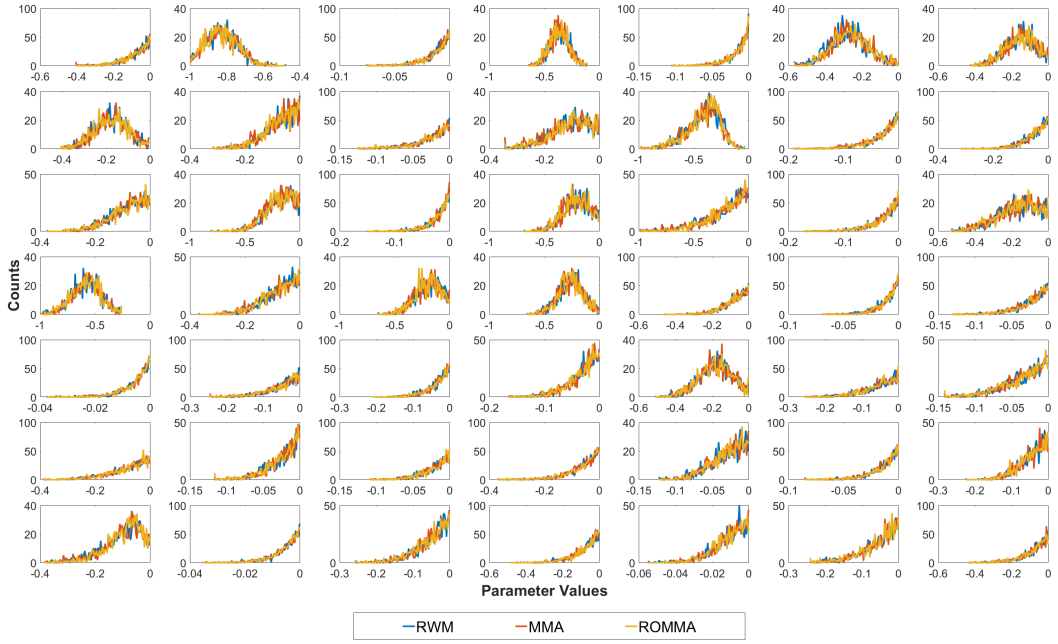

Figure 3.10: Histograms for the 49 parameters from the sample population of Random Walk Metropolis, the Modified Metropolis Algorithm, and the Rank One Modified Metropolis Algorithm for the constrained Bayesian logistic regression problem.

two orthogonal directions which lie in the fault plane. These two directions are chosen using seismological information already known about the event such that one direction is aligned along the dominant direction of the fault slip and the other is orthogonal to that direction. We call $\theta_{||}$ the slip parallel to the dominant direction and $\theta\perp$ the slip orthogonal to the dominant direction. The combined slip for all patches is contained in the parameter vector θ.

The prior distribution for $\theta\perp$ is taken to be $\mathcal{N}\left(0, \sigma^2\right)$, while the prior for $\theta_{||}$ is $\mathcal{U}\left(u_{min}, u_{max}\right)$. In this example, $\sigma = 0.5$, $u_{min} = -0.5$, and $u_{max} = 20$. The region is discretized into 144 patches yielding a 288 parameter vector.

The static source model assumes a linear forward model with Gaussian observation error. Thus, the likelihood function of the observation data, D, given the model parameters θ, is

$$p\left(D \mid \theta\right) = \frac{1}{\left(2\pi\right)^{\frac{N}{2}} \mid C \mid^{\frac{1}{2}}} \exp\left(-\frac{1}{2}\left(D - G\theta\right)^T C^{-1}\left(D - G\theta\right)\right) \tag{3.38}$$

Here N is the dimension of the parameter vector, C is the covariance of the observation error, and G is a Green's matrix function that maps the parameters to the static

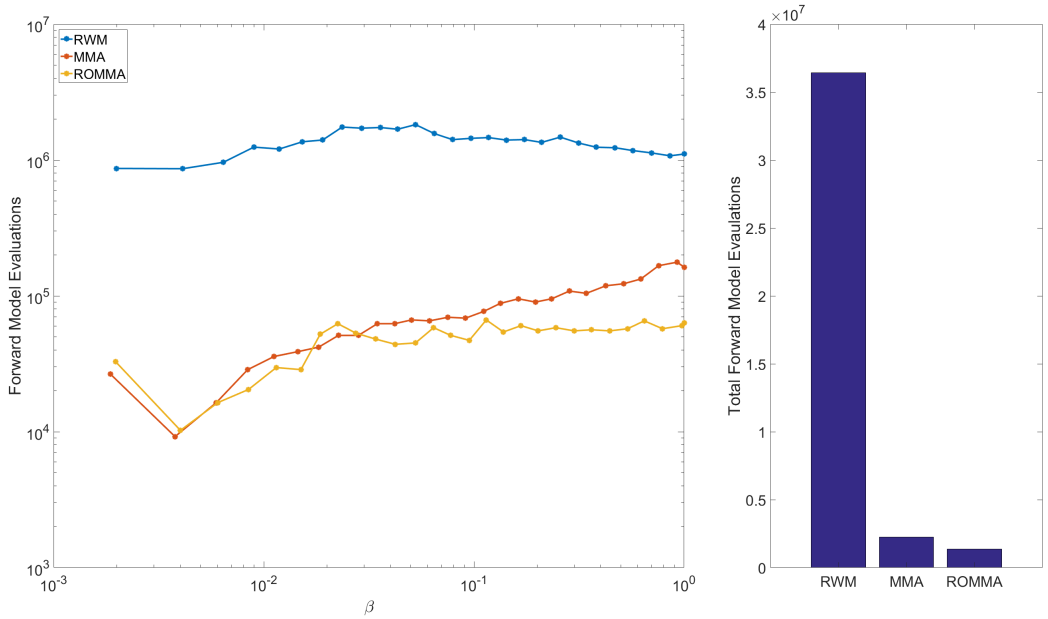

Figure 3.11: Required number of forward model evaluations using Random Walk Metropolis, the Modified Metropolis Algorithm, and the Rank One Modified Metropolis Algorithm to solve the constrained Bayesian logistic regression problem.

deformation observations.

As in the previous example, this example problem was then solved using ST-MCMC with three different MCMC samplers, Random Walk Metropolis, the Modified Metropolis Algorithm, and the Rank One Modified Metropolis Algorithm. Each algorithm uses a sample population of 1024 chains and has a set target COV of 1.0 and correlation target of 0.6. The mean and standard deviation of the slip velocity in the dominant direction is presented in Figure 3.12, while their histograms are plotted in Figure 3.13. There appears to be good agreement in the inference of these slip parameters for the three methods.

Figure 3.14 shows the performance comparison of the three methods. ROMMA requires 1.6% of the number of model evaluations of RWM and 7.8% of the model evaluations of MMA. MMA performs well when the beta step is small since the prior distribution well approximates the beta level distribution and there is little correlation between components, so the independent proposal assumption is not harmful. However, as the correlation becomes significant as beta increases, MMA starts to perform poorly, unlike ROMMA, which can adapt its proposal to this correlation structure. These computation gains come at the cost of many prior evaluations; however, for geophysics problems with more complex forward models

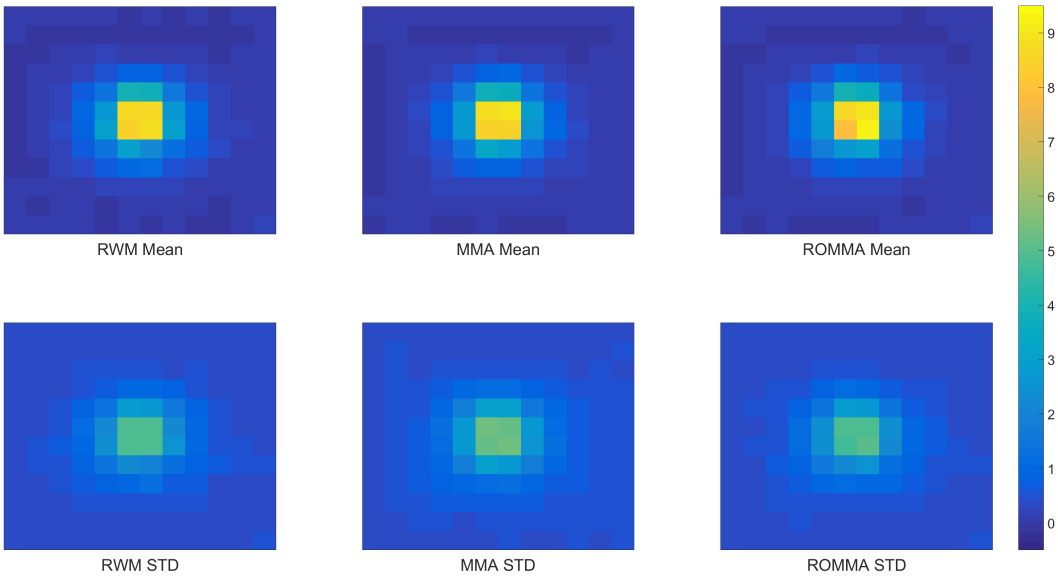

Figure 3.12: Sample Mean and Standard Deviation of the final sample population using Random Walk Metropolis, the Modified Metropolis Algorithm, and the Rank One Modified Metropolis Algorithm for the static finite fault model parameters $\theta_{\|}$.

like the kinematic finite fault model in [MSB13], this cost of prior evaluations becomes increasingly irrelevant compared with the evaluation of the likelihood function.

3.7 Extensions

Adaptation using Bayesian Optimization

While the feedback controller introduced in Section 3.4 is well designed for setting scaling factors in the Metropolis-Hastings algorithm, tuning parameters for other types of proposals remains challenging. One advantage of using ST-MCMC is that we can use the population of samples to explicitly measure the quality of our MCMC estimate with respect to the distribution. By looking at the beginning and end of each chain, and estimating the correlation coefficient ρ, we can explicitly try to minimize this estimate of ρ instead of a proxy for minimizing the correlation like tuning the acceptance rate to the "optimal" acceptance rate. We could do this by using Bayesian optimization to learn a functional relationship between our tuning parameters and the measure of correlation. One approach is to use a Gaussian process regression as a surrogate model to estimate this functional relationship and use a lower confidence bound on the estimated ρ to find the minimizer.

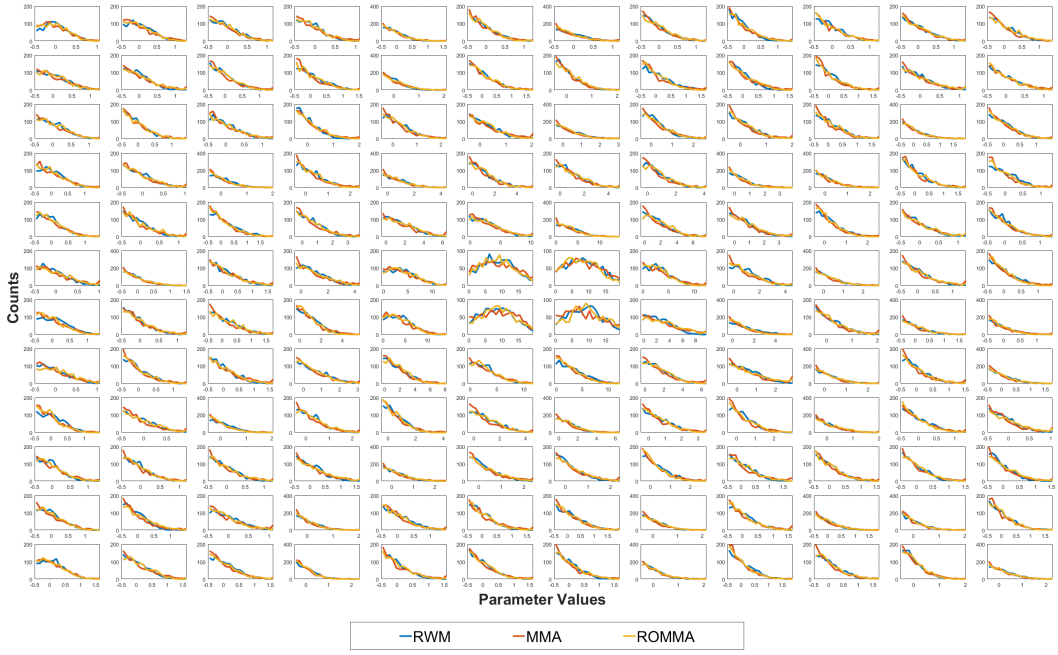

Figure 3.13: Histograms of the final sample population using Random Walk Metropolis, the Modified Metropolis Algorithm, and the Rank One Modified Metropolis Algorithm for the static finite fault model parameters $\theta_{\|}$.

Adaptive ROMMA

While ROMMA does not experience the same type of performance degradation that MMA suffers on correlated posteriors, it sees few performance gains when the posterior distribution is uninformed by the prior. The similar performance to RWM under these conditions is because the first step of ROMMA selects rank one proposal candidates based upon the prior and then accepts or rejects the final candidate based upon the data likelihood. Thus, if the prior has little information about the posterior, the first step is largely irrelevant to the performance of the algorithm. An extension of ROMMA would be to integrate more information from the data likelihood into the choice of the rank one components. In the current ROMMA, Algorithm 4, step 1 accepts or rejects with respect to the prior, $\pi(\theta)$, while step 2 accepts or rejects with respect to the data likelihood, $p(\mathcal{D} \mid \theta)$. However, we can separate the posterior into two normalizable distribution functions i.e. $p_1(\theta, \mathcal{D})$, for step 1, and $p_2(\theta, \mathcal{D})$, for step 2, as long as

$$p_1(\theta, \mathcal{D})\, p_2(\theta, \mathcal{D}) \propto p(\mathcal{D} \mid \theta)\, \pi(\theta) \tag{3.39}$$

By choosing $p_1(\theta, \mathcal{D})$ to be a fast-to-compute approximate distribution for the

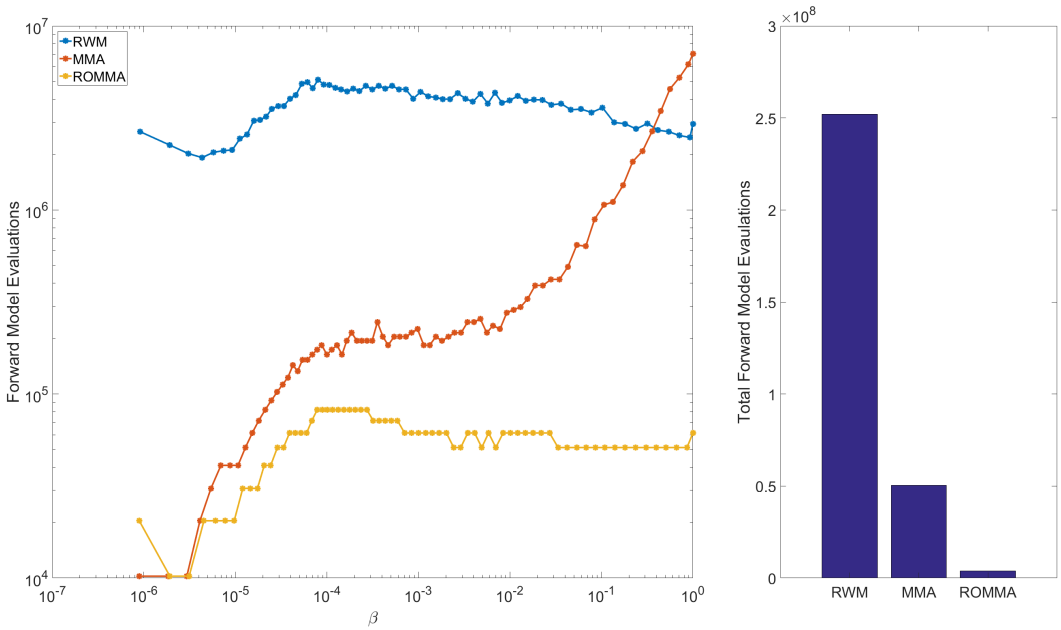

Figure 3.14: Required number of forward model evaluations using Random Walk Metropolis, the Modified Metropolis Algorithm, and the Rank One Modified Metropolis Algorithm for the static finite fault model.

posterior likelihood, we can integrate information from the posterior into the choice of the rank one components. Some examples of such approximate distributions would be a single multivariate Gaussian, a Gaussian mixture distribution, or the likelihood computed from only a small subset of the data.

To illustrate the importance of adaptation we return to the Bayesian logistic regression problem discussed in Section 3.6. This time we consider the unconstrained problem with a Gaussian prior distribution on the regression coefficients. The results of using RWM, ROMMA, and Adaptive ROMMA (A-ROMMA) are given in Figure 3.15. The approximate distribution used within A-ROMMA is a multivariate Gaussian distribution fitted according to the sample population. In this example, we see that ROMMA requires as many forward model evaluations as RWM as β increases toward 1. However, because A-ROMMA adapts to the intermediate distributions, it requires significantly fewer model evaluations than RWM even when β is close to 1 indicating that it can efficiently sample the posterior distribution.

3.8 Discussion

In this chapter, we have explored Sequential Tempered MCMC algorithms and proposed many adaptations which are beneficial for their performance. First, we

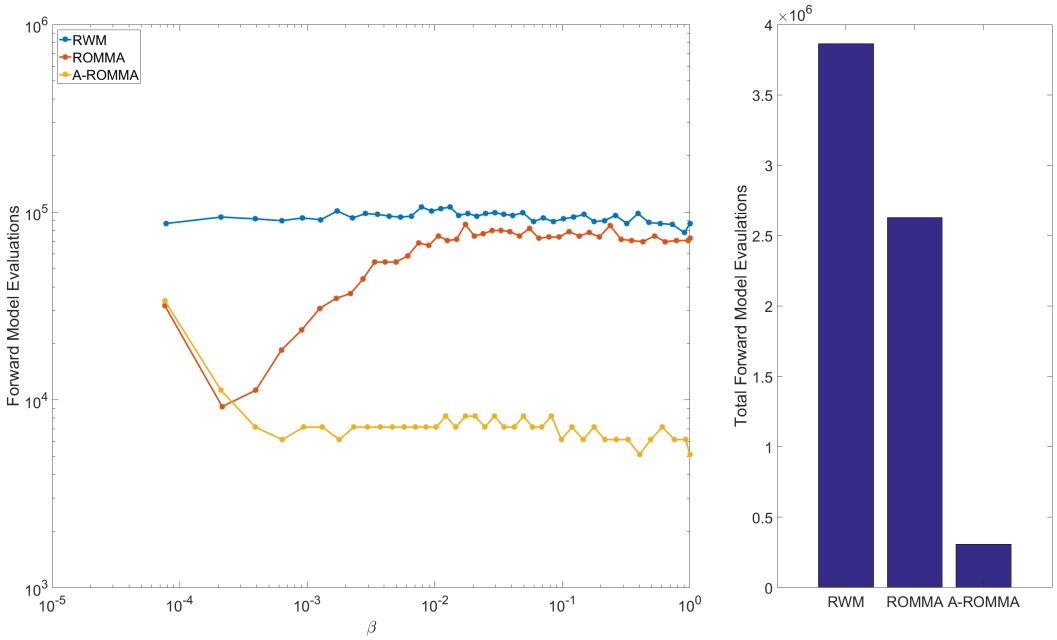

Figure 3.15: Required number of forward model evaluations using Random Walk Metropolis, the standard Rank One Modified Metropolis Algorithm, and the Adaptive Rank One Modified Metropolis Algorithm for an unconstrained Bayesian logistic regression problem.

discussed some basic theoretical results that provide some guidelines on how to choose tuning parameters in these algorithms so that the sample population can adequately capture the desired posterior. These approximate theoretical results are very general because they are independent of the distribution and quantity of interest. However, developing additional layers to this theory that can utilize this type of information, could improve the accuracy and robustness of these estimates. Further, we proposed two algorithm improvements to the MCMC step of the algorithm. We discussed a feedback controller, based upon the theory of Metropolis MCMC, which can effectively tune the sampler to a target acceptance rate. Then we presented the Rank-One Modified Metropolis Algorithm (ROMMA) which can help the sampler efficiently sample higher dimension posterior distributions with bounded support. ROMMA trades-off making many more prior likelihood evaluations to more effectively propose a candidate sample, reduces the number of forward model evaluations. This trade-off is generally favorable for inference problems involving complex systems where the forward model is computationally expensive to evaluate or when there is an abundance of data. The ROMMA algorithm was used to solve two higher-dimensional Bayesian inference problems, a constrained Bayesian logis-

tic regression problem and a static fault-model geophysics inverse problem. In both of these problems, we show the benefits of ROMMA.

Finally, further improvements to ST-MCMC and ROMMA are possible by extracting more information about the posterior distribution from the sample population. One example would be to use the history of each Markov chain in the population to adapt a multivariate Gaussian proposal distribution for each chain in the ST-MCMC algorithm. Fitting a local Gaussian for each chain instead of a single Gaussian proposal for all chains would allow the MCMC step to better adapt to the structure of the posterior distribution. This type of adaptation is beneficial for unidentifiable or locally identifiable distributions where the local structure is very important. However, it comes with higher computational and memory costs and with the added challenge of each Markov chain evolving according to a different Markov process, which means tuning and properly assessing the convergence and correlation of the chains will be more difficult. Another approach is to use a transport-map accelerated MCMC [PM14] to build better proposal distributions that speed up the MCMC sampling step. Using the samples from the population, which approximately capture the structure of the posterior, we can construct a map that transforms sampling the complex posterior distribution into sampling a distribution that is close to a simple distribution like a Gaussian, which can be efficiently sampled. After sampling on this simpler distribution, the samples are transformed back to the original posterior through the inverse map. The learned transport map can be evolved though the levels of ST-MCMC to gradually change and adapt to the posterior.

Chapter 4

INTEGRATED BAYESIAN APPROACH TO POWER SYSTEMS ESTIMATION

In this chapter, we present an integrated Bayesian approach to dynamic state estimation, fault detection, and fault classification within power systems. State estimation, based upon a new formulation of the Extended Kalman Filter, is used to determine the current operating state of the power system and make probability predictions about future behavior. Using these predictions we can then assess the likelihood of faults for detection and inference. Furthermore, by taking advantage of the network structure of the power grid, we can construct local and distributed estimators which are more robust and implementable in real power systems.

First, we motivate the necessity of dynamic state estimation and fault detection and classification in power systems and provide background on the system. Second, we present a new Extended Kalman Filter designed for Differential Algebraic Equations on network systems that combines both local and global estimation for robustness to faults. We then evaluate this filter on a test system and compare it to other techniques. Then we develop a method for fault detection and classification and test its performance. Finally, we show this work can also be formulated in a distributed context and conclude.

This work was done in collaboration with scientists at Los Alamos National Laboratory and grew out of work on fault classification presented in [Wie+14] and in our paper [Gar+15].

4.1 Motivation

Dynamic state estimation for power grids will enable new tools to quickly assess operational risk and respond to disturbances. Currently, state estimation techniques are used within the Distribution Management System (DMS) to estimate the steady-state power flow within the power grid. Existing methods use variations of least-squares algorithms applied to measured bus voltages to estimate unobserved flows between buses [AE04]. This works well when there are sufficient observations and the system is close to steady state. However, as fast timescale dynamics become increasingly important to the power grid due to the addition of intermitted renewable generation

and more dynamic loads, such an estimator is no longer sufficient [Hua+15]. Estimating a probability distribution over the dynamic states of the power system using Bayesian filtering, Section 1.2, provides robust estimates of the system's properties even with fast timescale and stochastic dynamics. These estimates can then be used to identify faults or changes in the power system in real time or to inform fast timescale control actions. This work introduces a hybrid filter that estimates local and global dynamic states and is robust to disturbances that can be used to address these challenges.

Dynamic state estimation can provide predictions for the states of loads, generators and controllers at resolutions of tenths or hundredths of a second, sufficient to capture transient behavior [Sau11]. These states completely determine the behavior of the power system according to a set of Differential Algebraic Equations (DAEs). Phasor measurement units (PMUs) make this estimation possible by providing data on frequency, voltage magnitude and phase at rates between 10 and 120 Hz[P+10]. The data can be aggregated within a central control facility or it can remain at the bus for local estimation and control. The Bayesian filtering inference problem is summarized in (1.9) as finding $p(x_k \mid z_{1:k})$, the probability distribution of the dynamic state describing the power system x_k at time t_k given $z_{1:k}$, the PMU measurements of the voltage phasor at times t_1 to t_k.

We present a state estimator that uses PMU data to estimate the dynamic state of the power system. This estimator makes the following key contributions. First, it introduces a modification to the Extended Kalman Filter that overcomes the performance and stability issues that arise when applying it to DAEs. Second, it combines global and local state estimation to form a hybrid estimator with the benefits of both methods and few of the drawbacks. Finally, since it uses the same approach as the local estimation, it is easy to formulate this estimator in a distributed context. These modifications make the estimator robust to disturbances and faults with unknown dynamics.

Similar to the dynamic state estimation problem, fast fault detection and classification are critical. The current topological estimation methods are based upon steady state behavior after a fault and do not use information from the dynamics [TO08]. This means fault classification and detection can be slow, making them less useful for prediction, uncertainty quantification, and control on fast timescales. Machine learning, clustering, and other data driven methods have become a popular approach to fast fault detection [XCK14; Gar+15]. In this work, instead of using machine

learning to develop some surrogate to classify different behaviors, we use different dynamic models of the power system and Bayesian model selection to quantify the likelihood of the different fault models. The fundamental tool for this is based on dynamic state estimation. This inference problem is summarized in (1.11) as finding $p(\mathcal{M} \mid z_{1:k})$, the probability of a specific fault, $\mathcal{M}$, having occurred given the observations $z_{1:k}$.

Finally, these estimation and inference methods are motivated as part of a future layered architecture, Fig. 4.1, that extends the DMS to faster time scales. This architecture is an implementation of the general Bayesian inference architecture presented in Section 1.2. It is able to provide state estimation, parameter estimation, disturbance detection, uncertainty quantification, and control, while being both fast and flexible [DC11; AD10]. Sensor data is used to update the system model through estimation and identify disturbances through change point detection. Using the current model, predictions are made to test model accuracy and to determine control actions.

The Dynamic State Estimation module from Fig. 4.1 is developed in Section , while the Topology Estimation and Change/Fault Detection modules are developed in Section . Because of we take a Bayesian approach, these modules give probabilistic estimates of the state and topology i.e. network model of the power system to the Power System DAE Model module, discussed in Section , based upon the available sensor data. This posterior distribution on the properties of the system can be used for probabilistic posterior predictions of future system behavior. Further Bayesian inference modules like the system identification problem discussed in 1.2 can be integrated into this architecture.

Literature Review

Many filtering methods have been considered for dynamic state estimation in power systems, such as the Extended Kalman Filter (EKF)[Hua+09; Zho+15; GK11; HSN07], Unscented Kalman Filter (UKF)[VT11; SP14; Zho+15; W+12], Ensemble Kalman filter (EnKF)[Li+12; Zho+15], and Particle Filter (PF)[ZML13; Men+12; Zho+15]. The UKF, EnKF, and PF prorogate a population of samples through the dynamics instead of computing derivatives for a linearization to prorogate a distribution function itself as in the EKF. The UKF is the typical choice because it is faster than the EnKF and PF, avoids the slow linearization of the EKF, and has higher accuracy [KFI08]. However, our approach uses an EKF based on the

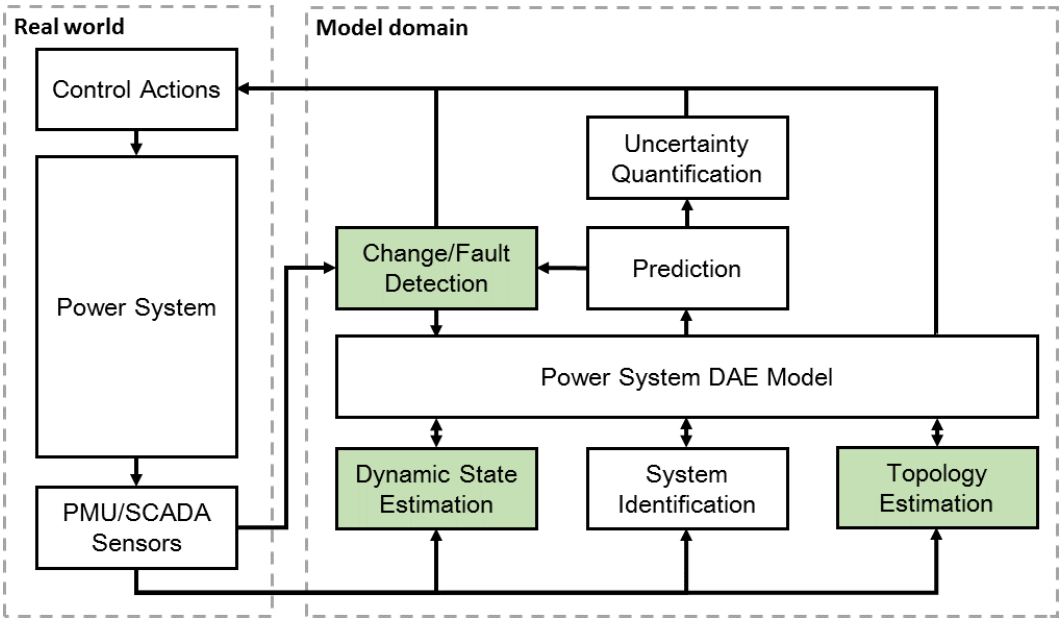

Figure 4.1: A layered approach for fast power system estimation and control based upon a Bayesian learning architecture. The shaded boxes describe inference methods developed in this work.

linearization of the DAE implicit integration rule. The resulting algorithm is faster than the UKF and has the same robustness and accuracy because many derivatives are already needed to solve the implicit integration regardless of which filtering method is selected.

In the area of dynamic state estimation, the literature has considered estimators that use either global information to estimate the entire state of the grid [W+12; HSN07] or local information to individually estimate the state of each machine on the grid [SP14; GK11; ZML13]. There are benefits and limitations to each approach. Recent studies have discussed the advantages of local methods over global methods for dynamic estimation [SP14]. In particular, local methods require only information from a PMU on the generator bus because it is independent of the rest of the network (conditioned on the state of the bus). Consequently, local methods are scalable and robust to disturbances. On the other hand, global methods model correlation between generators, estimate non-local parameters, and make predictions about future system behavior. Global methods also use data from any sensor on the network. Our contribution is a hybrid local-global estimator that combines many of these benefits without the limitations. Our estimator is a centralized method that, under normal operating conditions, uses a global estimator. The estimator then

switches to a local method when a disturbance is detected. Disturbance uncertainty is introduced and propagated between the estimators to demonstrate accuracy and robustness.

Additionally, studies have considered problems which contribute to the successes of a robust filter. Simultaneously estimating model parameters and state of the model is considered in [APS15; Hua+09]. Further, researchers have studied the robustness of filtering methods to disturbances, noise, and bad data [SP14; Zho+15; AKP15; VT11]. Finally, the problem of fast disturbance identification and classification has been considered in [XCK14; Gar+15].

4.2 System Description

The power system state variables can be separated into two sets, the dynamic state x and the algebraic variables y. The evolution of these state variables is defined by a set of differential algebraic equations (DAEs) under an external input u:

$$\begin{aligned} \dot{x} &= f(x, y, u) \\ 0 &= g(x, y, u) \end{aligned} \tag{4.1}$$

This DAE model is illustrated using specific models for a generator, control system, and network at the end of this section. The differential part of the DAE, $\dot{x} = f(x, y, u)$, is a nonlinear differential equation that describes the time evolution of the dynamic state x according to the generator dynamics, control dynamics, and properties of the electrical network connecting the generators and loads reflected by y. The vector x corresponds to the generator states such as rotor speed, rotor angle, transient axis voltages, and additional control system states. For the models used in this work this gives each generator around 5-10 dynamic states depending on the specifics of the generator and control systems. The algebraic part of the DAE, $0 = g(x, y, u)$, is a constraint that describes the relationship between the dynamic state x and the algebraic state variables y. This relationship primarily captures active and reactive power conservation on the electrical network. The state y contains the voltage (magnitude and phase) for each bus and additional algebraic variables for the generators and the controllers. For the models used here, this gives each bus 2 algebraic variables and each generator 3 variables. For The external inputs u are any unmodeled power injections. For the 37-Bus test system with 9 generators used in Section 4.4, $x \in \mathbb{R}^{48}$ and $y \in \mathbb{R}^{101}$. A description of the system is found in the remainder of the section, but for more details see [Mil10; Sau98].

In this work, generators are modeled using the one or two axis model [Mil10; Sau98]. Further, we consider generators with turbine governors and automatic voltage regulators (AVR) that can return the system to steady state after a disturbance. The turbine governor adjusts the mechanical power using a simple steam turbine governor model to maintain a reference frequency. The AVR system adjusts the exciter voltage of the generator using the standard IEEE type 1 model to maintain a reference output voltage. The dynamics of these components are described using the DAEs found in [Mil10; Sau98].

For illustration, a two axis generator model with a type II turbine governor from [Mil10] is described using equations (4.2), (4.3), and (4.4). The continuous time dynamics of the generator subject to a mechanical power input p_m is given by:

$$\begin{aligned}
\dot{\delta} &= \Omega_b \left(\omega - 1\right) \\
\dot{\omega} &= \frac{\left(p_m - p_e - D\left(\omega - 1\right)\right.}{M} \\
\dot{e}'_q &= \frac{-e'_q - \left(x_d - x'_d\right) i_d + v_f}{T'_{d_0}} \\
\dot{e}'_d &= \frac{-e'_d + \left(x_q - x'_q\right) i_q}{T'_{q_0}}
\end{aligned} \tag{4.2}$$

The constants Ω_b, D, M, x_d, x'_d, x_q, x'_q, v_f, and T'_{q_0} describe the generator. The variables δ, ω, e'_q, and e'_d are the dynamic states. While, i_d, and i_q are algebraic variables. Further, the turbine governor controls the mechanical power input to the generator in order to reach a desired frequency ω_{ref} according to the following dynamics:

$$\begin{aligned}
\dot{x}_g &= \frac{1}{T_2}\left[\frac{1}{R}\left(1 - \frac{T_1}{T_2}\right)\left(\omega_{ref} - \omega\right) - x_g\right] \\
p^*_m &= x_g + \frac{T_1}{RT_2}\left(\omega_{ref} - \omega\right) + p^0_m \\
p_m &= \begin{cases} p^*_m, & for p^{min} \leq p^*_m \leq p^{max} \\ p^{min}, & for p^*_m < p^{min} \\ p^{max}, & for p^*_m > p^{max} \end{cases}
\end{aligned} \tag{4.3}$$

For this control system, T_1, T_2, R, ω_{ref}, p^0_m, p^{min}, and p^{max} are constants. While, x_g is a dynamic state and p_m is an algebraic variable. Finally, the interaction between

the internal generator states and the state of bus which connects it to the network is captured through a set of algebraic constraints:

$$\begin{aligned} v_d &= v \sin(\delta - \theta) \\ v_q &= v \cos(\delta - \theta) \\ 0 &= v_q + r_a i_q - e'_q + x'_d i_d \\ 0 &= v_d + r_a i_q d - e'_d + x'_q i_d \\ p_e &= \left(v_q + r_a i_q\right) i_q + \left(v_d + r_a i_d\right) i_d \end{aligned} \tag{4.4}$$

Here, r_a is a constant and v_d and v_q are additional algebraic states. These equations describing this generator system are part of the DAE model (4.1) where (4.2) and (4.3) are part of the differential equation $\dot{x} = f(x, y, u)$ and (4.4) is part of the algebraic equation constraints, $0 = g(x, y, u)$.

The network is comprised of n buses, lines, PQ loads and PV generators. A PQ load, under normal operating conditions, draws a constant amount of active and reactive power at a bus. A PV generator produces a constant amount of active power at a fixed voltage magnitude. When the voltage magnitude is outside the nominal range, the PQ loads are transformed into constant impendence loads. The total active and reactive power loads at bus i are written as P_{li} and Q_{li} respectively , while the total active and reactive power generation at bus i are written as P_{gi} and Q_{gi}. The active and reactive power conservation at bus i is given by:

$$\begin{aligned} 0 &= \sum_{k=1}^{n} |V_i||V_k| \left(G_{ik} \cos\left[\theta_i - \theta_k\right] + B_{ik} \sin\left[\theta_i - \theta_k\right]\right) - P_{gi} + P_{li} \\ 0 &= \sum_{k=1}^{n} |V_i||V_k| \left(G_{ik} \sin\left[\theta_i - \theta_k\right] - B_{ik} \cos\left[\theta_i - \theta_k\right]\right) - Q_{gi} + Q_{li} \end{aligned} \tag{4.5}$$

These constraints form part of the algebraic equations of the DAE, $0 = g(x, y, u)$, which must be satisfied at each time step. Here, the admittance matrix $\mathbf{Y} \in \mathbb{C}^{n \times n}$ has been decomposed into the conductance matrix $\mathbf{G} \in \mathbb{R}^{n \times n}$ and susceptance matrix $\mathbf{B} \in \mathbb{R}^{n \times n}$ such that $\mathbf{Y} = \mathbf{G} + i\mathbf{B}$. These matrices capture the properties of the electrical network. The pair $|V_i|$ and θ_i are the voltage magnitude and phase at bus i respectively. These are variables contained within the algebraic state y.

Small disturbances are modeled as additive noise injected into the state variable evolution at each time step, while large disturbances are modeled as cleared three

phase faults on the network. These faults can be line or bus faults, including the loss of a generator or load.

On the network, we model PMUs that measure the bus voltage phasor and current phasors over the lines. We introduce measurement error to simulate real sensors. Since voltage magnitude and phase are algebraic variables in the DAE in equation (4.1), the voltage measurement equation is the implicit solution to the algebraic constraint. The current measurement equation can then be expressed in terms of the voltage and the admittance matrix $\mathbf{Y}$, which is assumed to be known. Because of computational limitations, our implementation of the estimator only uses the voltage phasor measurements for estimating the state since these are algebraic variables. The current measurements on lines connecting the partitions are used to compute the uncertain inputs needed to decentralize the estimator. In the future, experimental design techniques could be used to better select the set of measurements that best estimate the state.

4.3 Power System Filtering

We introduce a novel approach to dynamic state estimation that overcomes many of the limitations of the EKF and combines the benefits of both global and local estimation methods. In this section, we first discuss the state estimation problem in general. Then we present global estimation and the modified EKF. Next, we discuss local estimation for handling disturbances. Finally, we show how to integrate the global and local estimation methods into a hybrid estimator.

State Estimation Problem

The state estimation problem is introduced in Section 1.2 for the discrete dynamical system:

$$\begin{aligned} x_k &= \mathfrak{F}_k\left(x_{k-1}\right) + w_k \\ z_k &= \mathfrak{H}_k\left(x_k\right) + v_k \end{aligned} \tag{4.6}$$

For the power system: x_k are the state variables, i.e. dynamic generator states, and z_k are the observed outputs, i.e. PMU measurements, at time t_k. Further, w_k is the additive process noise that corresponds to disturbances, while v_k is the measurement noise. $\mathfrak{F}_k\left(x_k\right)$ is the dynamic state evolution function and $\mathfrak{H}_k\left(x_k\right)$ is the measurement function.

Within the context of global estimation of power system dynamic states, the DAE

model (4.1) is solved implicitly and discretized into the form of (4.6). The algebraic states y_k and known inputs u_k can be embedded into the functions $\mathfrak{F}_k(x_k)$ and $\mathfrak{H}_k(x_k)$, making these functions time dependent and only a function of x_k. The algebraic states y_k can be implicitly solved for given the dynamic states x_k using the constraint $0 = g(x, y, u)$. Thus, using the PMU measurements, z_k, and knowledge of prior probabilities on the state, we are able to estimate the posterior probability distribution of the global state.

Global State Estimation

We develop a state estimator for the global power system DAE presented in Section 4.2. The state estimator we choose is a modified Extended Kalman Filter (EKF), which approximates the Bayes Filter (1.8) - (1.9). This EKF uses a linearization of the implicit trapezoid rule integrator instead of the standard Euler integrator. This modification produces a more accurate and robust filter.

Extended Kalman Filter The Extended Kalman Filter (EKF) is an extension of the Kalman Filter to nonlinear systems[Sim06; Che03], such as the discrete system (4.6). While it does not possess the optimality guarantees of the Kalman Filter, the EKF provides a good approximation when the nonlinearities are small with respect to the time step.

The algorithm begins with a prediction step:

$$\begin{aligned} \hat{x}_{k|k-1} &= \mathfrak{F}_k\left(\hat{x}_{k-1|k-1}\right) \\ P_{k|k-1} &= F_{k-1} P_{k-1|k-1} F_{k-1}^T + Q_{k-1} \\ \hat{z}_{k|k-1} &= \mathfrak{H}_k\left(\hat{x}_{k|k-1}\right) \\ S_{k|k-1} &= H_k P_{k|k-1} H_k^T + R_k \end{aligned} \tag{4.7}$$

The first equation computes the future state $\hat{x}_{k|k-1}$ by evolving the current estimate $\hat{x}_{k-1|k-1}$ according to the dynamics $\mathfrak{F}_k$. The covariance of the state prediction $P_{k|k-1}$ is then estimated by the second equation using a linearization F_{k-1} of the dynamics $\mathfrak{F}_k$ around $\hat{x}_{k-1|k-1}$, to evolve the past covariance $P_{k-1|k-1}$ forward in time with additive process noise covariance Q_{k-1}. Generally, the EKF uses an Euler approximation to discretize and linearize a continuous system, but the EKF works with any discretization and linearization scheme. In the third equation of (4.7), the predicted output $\hat{z}_{k|k-1}$ is calculated from $\hat{x}_{k|k-1}$ based on the output function $\mathfrak{H}_k$. Finally, the covariance of the output $S_{k|k-1}$ is estimated using the linearization H_k of

the output function $\mathfrak{H}_k$ around $\hat{x}_{k|k-1}$ with additive measurement noise covariance R_k.

After making the prediction, the state estimate is corrected using the observed data z_k at the current time step:

$$\begin{aligned} K_{k|k-1} &= P_{k|k-1} H_k^T S_{k|k-1}^{-1} \\ \hat{x}_{k|k} &= \hat{x}_{k|k-1} + K_{k|k-1} \left(z_k - \hat{z}_{k|k-1} \right) \\ P_{k|k} &= \left(I - K_{k|k-1} H_k \right) P_{k|k-1} \end{aligned} \tag{4.8}$$

In the second equation, the posterior state estimate $\hat{x}_{k|k}$ conditioned on z_k is determined using feedback on the observation error $z_k - \hat{z}_{k|k-1}$, with the Kalman feedback gain $K_{k|k-1}$, defined in the first equation. The posterior state covariance, $P_{k|k}$, is then found by propagating the state prediction covariance, $P_{k|k-1}$, through the linearized feedback function.

For many applications, sampling methods like the UKF, EnKF, and PF are favored over the EKF because they do not require computing the linearization F_k. For power systems of moderate size, like the ones considered in this work, the dimension of the dynamic state is on the order of tens or hundreds of variables so the computation of the linearization is still tractable in realtime. Furthermore, since all these methods must solve the original DAE, the implicit solver generally is the most computationally expensive step not the linearization making the EKF still attractive. For large systems where the dimension prohibits computing the linearization, a better strategy is to break the system up into a set of subsystems and solve the filtering problem in a distributed fashion as discussed in Section 4.8.

Numerical Integration Scheme Irrespective of the filtering method we use, a numerical integration scheme must be used to transform the continuous time DAE (4.1) into a discrete time system. The algebraic constraints of the DAE typically create a set of equations that must be solved implicitly using solvers like the Newton-Raphson method. The choice of the integration scheme can have a significant effect on the computational complexity, numerical accuracy, and stability of the integration. Each integration scheme implies a different discretization and local linearization of the dynamics. Typically, the discretization and linearization are done simultaneous using an Euler approximation. This approach for power system dynamic state estimation is illustrated in [Hua+09; Zho+15; GK11; HSN07]; however, it has issues with stability and accuracy requiring small time steps [Zho+15].

To increase integration accuracy and robustness, we perform the discretization using an implicit trapezoid rule. The output function $\mathfrak{H}_k$ uses a matrix C to select algebraic variables y to measure.

$$\begin{aligned} x_k &= x_{k-1} + \frac{\Delta t}{2}\left(f\left(x_{k-1}, y_{k-1}, u_{k-1}\right) + f\left(x_k, y_k, u_k\right)\right) \\ 0 &= g\left(x_{k-1}, y_{k-1}, u_{k-1}\right) \\ 0 &= g\left(x_k, y_k, u_k\right) \end{aligned} \tag{4.9}$$

$$\begin{aligned} z_k &= \mathfrak{H}_k\left(x_k\right) \\ &= C y_k \end{aligned} \tag{4.10}$$

Solving equation (4.9) implicitly implies that $x_k = \mathfrak{F}_k\left(x_{k-1}\right)$. When solving DAEs, families of implicit integrators such as the Implicit Runge-Kutta or the Backward Differentiation Formula methods are preferred because of their stability over long time steps [KM06]. Functions $\mathfrak{F}_k$ and $\mathfrak{H}_k$ are evaluated with an implicit solver and never formed explicitly, but their Jacobians can be formed explicitly for the linearization:

$$\begin{aligned} F_{k-1} &= \frac{\partial \mathfrak{F}_k}{\partial x} \Big|_{x_{k-1}} \\ &= \left(I - \frac{\Delta t}{2}\frac{\partial f}{\partial x} - \frac{\Delta t}{2}\frac{\partial f}{\partial y}\frac{\partial y}{\partial x}\right)^{-1} \Big|_{x_k, y_k} \times \left(I + \frac{\Delta t}{2}\frac{\partial f}{\partial x} + \frac{\Delta t}{2}\frac{\partial f}{\partial y}\frac{\partial y}{\partial x}\right) \Big|_{x_{k-1}, y_{k-1}} \end{aligned} \tag{4.11}$$

$$\begin{aligned} H_k &= \frac{\partial \mathfrak{H}_k}{\partial x} \Big|_{x_k} \\ &= C\frac{\partial y}{\partial x} \Big|_{x_k, y_k} \end{aligned} \tag{4.12}$$

$$\frac{\partial y}{\partial x} = -\frac{\partial g}{\partial y}^{-1}\frac{\partial g}{\partial x} \tag{4.13}$$

This method is an A-stable method [Ise09], so it can take long time steps without becoming unstable. While this method is typically more intensive than Euler's method, when solving DAEs the methods have similar computational complexity because of the algebraic constraints. Further, the linearization is much more accurate. This alleviates many of the concerns with EKF observed in [VT11; Men+12; ZML13; Hua+09; SP14; Zho+15].

Local Estimation for Fault Handling

When the power system dynamics change due to a fault disturbance, the models we use to estimate the global state are no longer valid. If state estimation continues with the wrong models, the state estimate can be driven far away from the true state, where the estimator might fail because the algebraic constraints are no longer satisfied. This suggests the need for a more robust estimator that handles faults. A local estimator provides robustness since it only uses the local machine dynamics and the state of the local bus connecting the machine to the network to compute a local estimate. Thus, we combine a global state estimator and a local state estimator. When a fault is detected by a change point detection module, the estimator switches from the global method to the local method. When the fault clears or is classified, the system switches back to the global estimator using the local dynamic states and an updated model to restart the estimator. The local estimation method requires that all generator buses have a PMU and that faults be detectable. In Section 4.6 we discuss the fast fault detection and classification problem.

Local State Tracking and Estimation The state of the bus connecting a generator to the network fully captures the interaction of the generator with the power grid. Thus, if a PMU is placed on the generator bus, local state estimation is possible. A subset of the PMU measurements are viewed as uncertain inputs to the generator, while any other measurements are viewed as outputs and used for estimation[SP14]. In this work, we will consider the case where only voltage is measured, thus making forward propagation of the dynamic generator state possible, but not full estimation.

The local generator model can be defined as a DAE[Sau98; Mil10], as in (4.14). The dynamic states x are the generator and control states, the input u is the local bus voltage, and the algebraic states y are the bus current and electric and mechanical power. The differential equation (4.14) below describes the evolution of the generator and control dynamic states, while the algebraic equations describe the relationship between generator and the bus. For the two axis generator model with a turbine governor, this DAE would be the differential equations (4.2) and (4.2), while the algebraic equations would be (4.4). These algebraic equations can be solved explicitly in terms of the dynamic states and the observed bus voltage phasor v and θ. Therefore, the algebraic variables are expressed as a function of the input and state.

$$\begin{aligned} \dot{x} &= f\left(x, y, u\right) \\ y &= g\left(x, u\right) \end{aligned} \tag{4.14}$$

Much like the full system, these continuous equations can be solved numerically in discrete time using the integration scheme described earlier in the section. The discretized system has the form of (4.16) for each generator i at bus j at time t_k. The input u is discretely measured using the PMUs and intermediate values are found using linear interpolation. We can make the explicit substitution of $y = g\left(x, u\right)$ into $f\left(x, y, u\right)$ to remove the algebraic variable dependence. Therefore, we define and ODE for the dynamics of generator i connected to bus j as:

$$\begin{aligned} \dot{x}^i &= f_i\left(x^i, g\left(x^i, u^j\right), u^j\right) \\ &= \phi_i\left(x^i, u^j\right) \end{aligned} \tag{4.15}$$

This leads to the discretized dynamics using the implicit trapezoidal method:

$$\begin{aligned} x_k^i &= x_{k-1}^i + \frac{\Delta t}{2}\left(\phi_i\left(x_{k-1}^i, u_{k-1}^j\right) + \phi_i\left(x_k^i, u_k^j\right)\right) \\ u_k^j &= \left[v_k^j, \theta_k^j\right]^T \end{aligned} \tag{4.16}$$

The local generator states are independent from the rest of the system given the true state of their connection bus. However, if there is measurement noise, the local states estimates will still have correlations because the algebraic network constraints are not perfectly satisfied. We do not assume knowledge of the global model during local estimation, so these correlations are not captured when estimating the state. However, uncertainty due to measurement and process noise is propagated for the local state estimate. As in the global estimator, we use a Gaussian noise model with covariance Σ and linearized dynamic model F to capture the relationship between the set of local states x_{k-1} and set of inputs u_{k-1}, u_k:

$$\Sigma_{k|k} = \begin{bmatrix} P_{k|k} & & 0 \\ & & 0 \\ 0 & 0 & R \end{bmatrix} \tag{4.17}$$

$$F_k = \begin{bmatrix} \frac{\partial x_k}{\partial x_{k-1}} & \frac{\partial x_k}{\partial u_{k-1}} & \frac{\partial x_k}{\partial u_k} \\ 0 & 0 & I \end{bmatrix}$$

This uncertainty is then propagated:

$$P_{k|k} = F_k \Sigma_{k-1|k-1} F_k^T + \begin{bmatrix} Q & 0 \\ 0 & 0 \end{bmatrix} \tag{4.18}$$

Hybrid Estimator: Handling the Fault Event

It is important to include the fault time uncertainty in both the local and global estimators. A good model for the disturbance time random variable, Δt_f, is a uniform distribution over the time step, since a disturbance could have occurred anytime between the previous and current sample. However, since we are using Gaussian models, we map the uniform distribution onto a Normal distribution by minimizing the Kullback-Leibler divergence between the two distributions. We find that the optimal Normal distribution for Δt_f is $Q^* = \mathcal{N}\left(\frac{\Delta t}{2}, \frac{\Delta t^2}{12}\right)$ where Δt is the sampling time step. This distribution is propagated through the linearized model to estimate the uncertainty due to the fault time.

We use a two-step approach when dealing with the fault event and the clearing of the event. When a fault is detected, we first propagate the global state forward by half a time step (4.19):

$$\begin{aligned} \hat{x}_+ &= \mathfrak{F}_G\left(\hat{x}_{k-1}, \frac{\Delta t}{2}\right) \\ \hat{z}_+ &= \mathfrak{H}\left(\hat{x}_+\right) \end{aligned} \tag{4.19}$$

Then the estimator switches to the local method and we use the local PMU measurements to continue propagating the local generator states for the remaining half a time step (4.20):

$$\hat{x}_k = \mathfrak{F}_L\left(\hat{x}_+, U\left(\hat{z}_+\right), U\left(z_k\right), \frac{\Delta t}{2}\right) \tag{4.20}$$

Similarly, when we detect that the fault has cleared, we continue using the local method to propagate the states half a time step, then the estimator switches to the global method for the remainder of the time step. We restart the global estimator using the new measurements and local state estimates.

We propagate the uncertainty in the fault time through the dynamics of the local and global models using:

$$P_{k|k} = F_k \Sigma_{+|k-1} F_k^T + \begin{bmatrix} \frac{1}{2}Q & 0 \\ 0 & 0 \end{bmatrix} \tag{4.21}$$

The linearization F_k is given by:

$$F_k = \begin{bmatrix} \frac{\partial \mathfrak{F}_L}{\partial x_+} & \frac{\partial \mathfrak{F}_L}{\partial U(z_+)} & \frac{\partial \mathfrak{F}_L}{\partial U(z_k)} & -\frac{\partial \mathfrak{F}_L}{\partial \Delta t} \\ 0 & 0 & I & 0 \end{bmatrix}$$

$$\begin{aligned} F_+ &= \frac{\partial \mathfrak{F}_G}{\partial x} \mid_{\hat{x}_+} \\ H_+ &= \frac{\partial \mathfrak{H}}{\partial x} \mid_{\hat{x}_+} \\ T_+ &= \frac{\partial \mathfrak{F}_G}{\partial \Delta t} \mid_{\frac{\Delta t}{2}} \end{aligned} \tag{4.22}$$

The local state and input covariance $\Sigma_{+|k-1}$ (4.23) is:

$$\Sigma_{+|k-1} = \begin{bmatrix} P_{+|k-1} & P_{+|k-1}H_+^T & 0 & \frac{\Delta t^2}{12}T_+ \\ H_+P_{+|k-1} & S_{+|k-1} & 0 & 0 \\ 0 & 0 & R & 0 \\ \frac{\Delta t^2}{12}T_+^T & 0 & 0 & \frac{\Delta t^2}{12} \end{bmatrix} \tag{4.23}$$

4.4 Case Study: Filtering for a 37-Bus Test System

The local-global dynamics state estimation method was evaluated on a 37 Bus Test system, Fig. 4.2, with nine generators based on the test system found in[GSO11]. This system is implemented in the Power System Analysis Toolbox[Mil05] modeling environment in MATLAB. We solved the power system DAE using an implicit trapezoidal method with a time step of 0.01 seconds. We also simulated additive independent Gaussian process and measurement noise. In the simulations, a bus fault disturbs the system while it is in steady state and then clears after a specified fault period. Unless otherwise specified, the fault corresponds to the disconnection of the generators on Bus 54 from the system for a short period of time. We sample the PMUs and perform the state update at 25 Hz, a time step of 0.04 seconds, unless specified. For more details on the simulation see [Cat15].

The noise distributions were adapted from[W+12]. The variance for the generator states is: 10^{-5} for the rotor angle δ, 10^{-8} for the frequency ω, 10^{-7} for the q-axis transient voltage e_q, and 10^{-20} for the d-axis transient voltage e_d. Each turbine

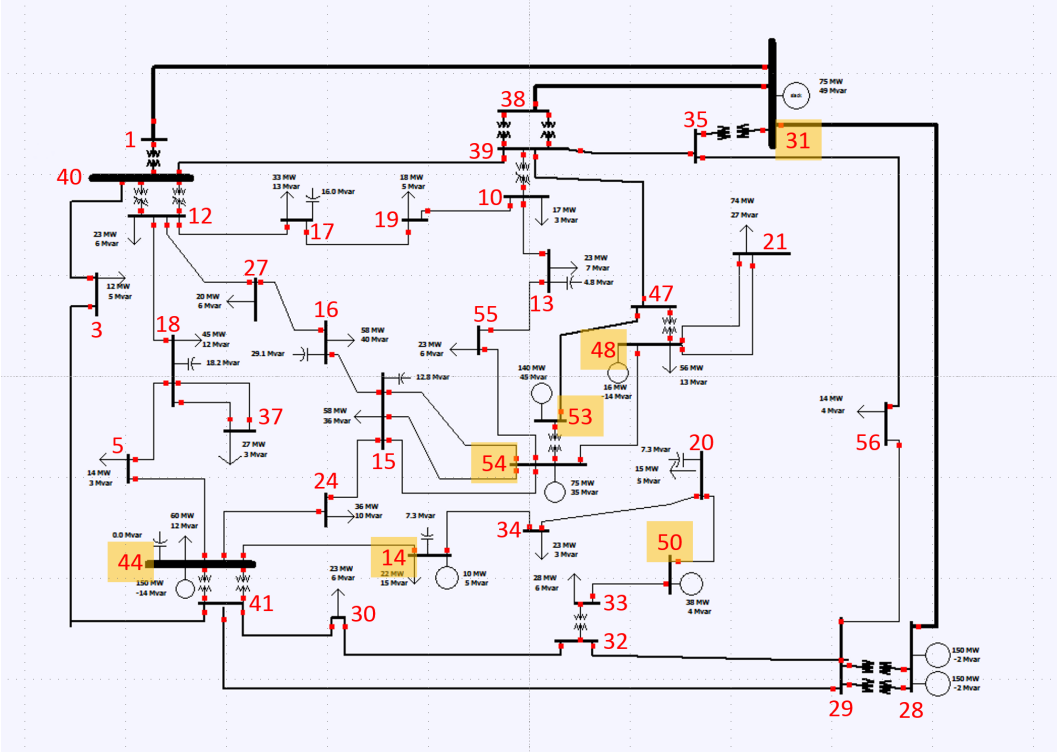

Figure 4.2: 37-bus test system with nine generators on seven buses (Yellow).

governor has up to four control states, where the variance on x_g, x_{g_1}, and x_{g_2} is 10^{-4} and x_{g_3} has a variance of 10^{-20}. The small variances of size 10^{-20} correspond to values that effectively do not change during the simulations. The PMU measurements have a variance of 10^{-3} for the magnitude v and 10^{-4} for the phase θ. Here, all magnitudes are expressed using the per-unit system and phase is in radians.

Fault Handling

We performed a series of experiments to investigate different methods for handling cleared faults. Four different EKFs were considered, each dealing with the fault period in a different manner, as illustrated in Fig. 4.3. The first filter (blue) did not use fault detection, so it continued using the pre-fault physical model to perform state estimation. This method rejects any measurements that cause the algebraic constraint to no longer have a solution. The second filter (red) can only detect if the system has a fault or not. When it detects a fault, it rejects all measurements and propagates the past state forward in time using the pre-fault model without update. The third filter (orange) is an oracle that assumes perfect information about the fault, so it develops a new physical model and continues global estimation. Finally, the fourth filter (purple) is the local-global filter that switches to a local estimation

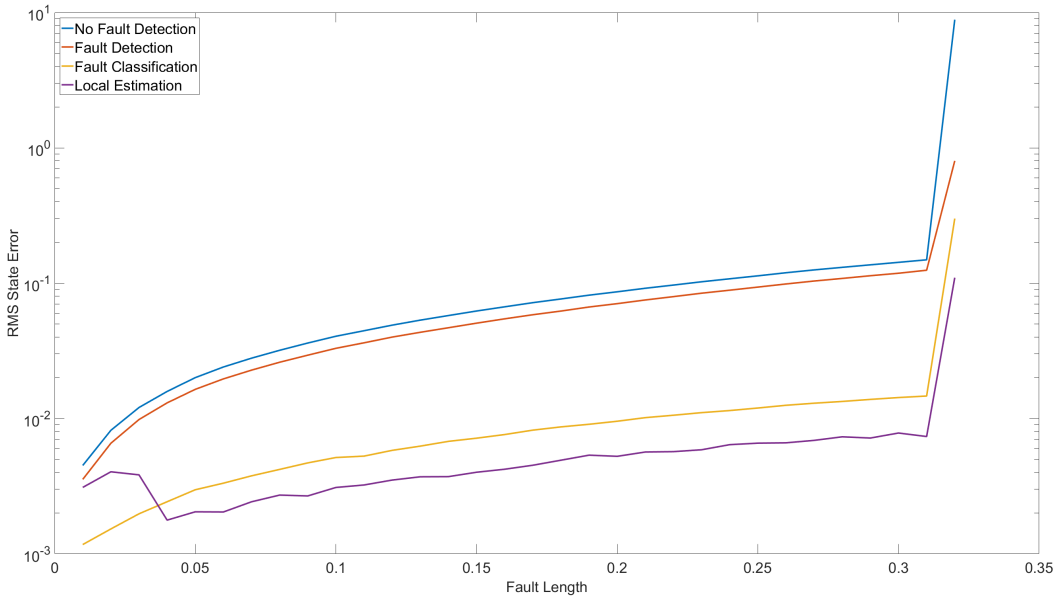

Figure 4.3: Comparison of the performance of fault handling methods for different fault durations. RMSE is averaged over all states for a five second trajectory.

method during the fault. In our experiment, the fault length varies between 0.01 and 0.32 seconds.

The results in Fig. 4.3 show that properly handling the fault period is crucial, reducing the error by an order of magnitude. As long as the fault lasts long enough to be detected, we see that the local-global estimation method performs as well as full fault classification. We see that ignoring the data during the fault is a poor strategy. This is because the post-fault state estimate is far from the truth, so it takes many measurements for it to recover. In contrast, we find that local estimation does as well as actually knowing the true model during the fault period. We note that the local estimation method will typically overestimate the disturbance for short faults because it only has time resolution as good as the sampling time step. It starts out as poorly as the less informed methods, but then quickly converges to the same performance as the true model. All filters perform poorly for the longest fault that causes the system to go unstable.

Numerical Integration

Solving a continuous differential system with a discrete integration method introduces numerical errors. These numerical integration errors depend on the system, the integration method used, and the time step used to evaluate the system. It is common to model these integration errors as additional process noise injected into

the system. This integration error is estimated by simulating the power system without injected noise and then finding the intensity of process noise that minimizes the estimation error. The intensity is defined relative to the nominal process noise values described earlier in this Section. We find that the optimal process noise estimate for the implicit trapezoidal integrator is around 10^{-4}, while the optimal intensity for the Euler integrator corresponds with the nominal values found in [W+12]. Since these nominal values overestimate the integration error, the estimator does not add any additional integration error noise for the experiments in the rest of this section.

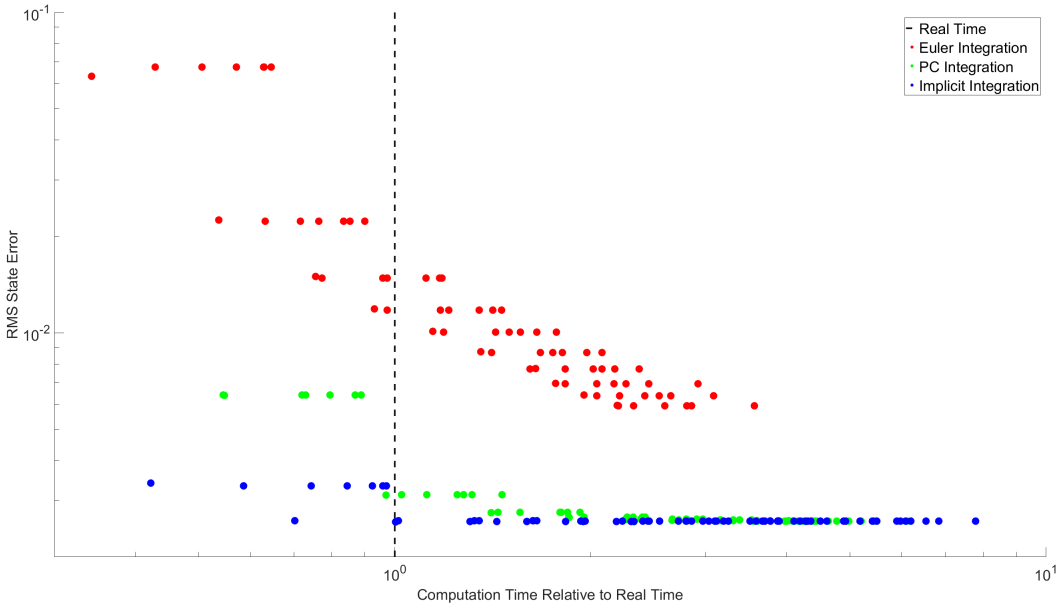

Figure 4.4: Performance comparison of integration methods. Each circle corresponds to an estimator that uses different parameters for the numerical integrator. The computation time is measured relative to the length of the time period simulated.

We consider the question of the trade-off between computation time and error for different integration methods in Fig. 4.4. Each point corresponds to different integration parameters such as the number of intermediate integration steps and to the tolerance of the implicit solver for the algebraic constraints. We can see that the Euler integration (red) shows a very distinct trade-off between accuracy and computation time. We observe this trade-off for an intermediate Predictor-Corrector (PC) method (green), but it is almost indiscernible for the implicit trapezoidal method (blue). As intermediate integration steps are added or as the error tolerance is decreased, the implicit method realizes very small gains in accuracy, but experiences increased computation time. We can also see that changing error tolerance has very little effect on all methods, as the horizontal bands in the figure correspond to a changes

in error tolerance. A lower tolerance means fewer iterations per Newton solve, but only small gains in accuracy. We observe that the implicit trapezoidal integration method has by far the best properties; for any computation time, it has the lowest error. Finally, we see that the implicit trapezoidal method is able to reduce the error significantly, while still performing in real time for this test system.

Filter Selection

We tested three filtering schemes, the EKF, the unscented Kalman filter (UKF), and the particle filter (PF) on the 37 Bus system with a bus fault in order to characterize differences in performance and robustness. One important difference between these methods is the computational time needed to solve each of these problems. We find that that EKF is the fastest estimation method. Both the UKF and PF are sampling based methods, where each sample has about the same computational complexity as the full EKF because of the implicit integration. The speed and accuracy of the PF depends on the number of particles used to estimate the probability distribution. For this test, 1000 particles were used. We use the EKF as the proposal distribution for the particle filter because it helps increase sampling efficiency.

One important consideration when designing the filter is its robustness to incorrectly modeled process noise because process noise is notoriously difficult to determine. We can see from Fig. 4.5 that the EKF and UKF are more robust than the particle filter to an incorrect characterization of the process noise. When the process noise model is incorrect, the proposal distribution for the PF is not able to adequately sample the posterior distribution, causing the estimate to be inaccurate.

We also observe that the nonlinearity does not affect the EKF any more than the UKF or the PF by investigating the performance of the filters with an increasing time step. If the linearization were very poor, we would expect the performance of the EKF to degrade much more than that of the UKF or the PF as the time step increases. Fig. 4.6 shows that this is not the case, as the UKF and EKF perform almost identically. The increasing error of all filters corresponds to the decreased sampling rate and to the increased integration error.

Design Considerations

We explore some of the design considerations for the hybrid estimator, namely the sampling rate, the fidelity of the sensors, and the number of sensors. First, we consider the relationship between the measurement noise and the sampling time step in Fig. 4.7. Generally, if the measurement noise decreases, a larger time step

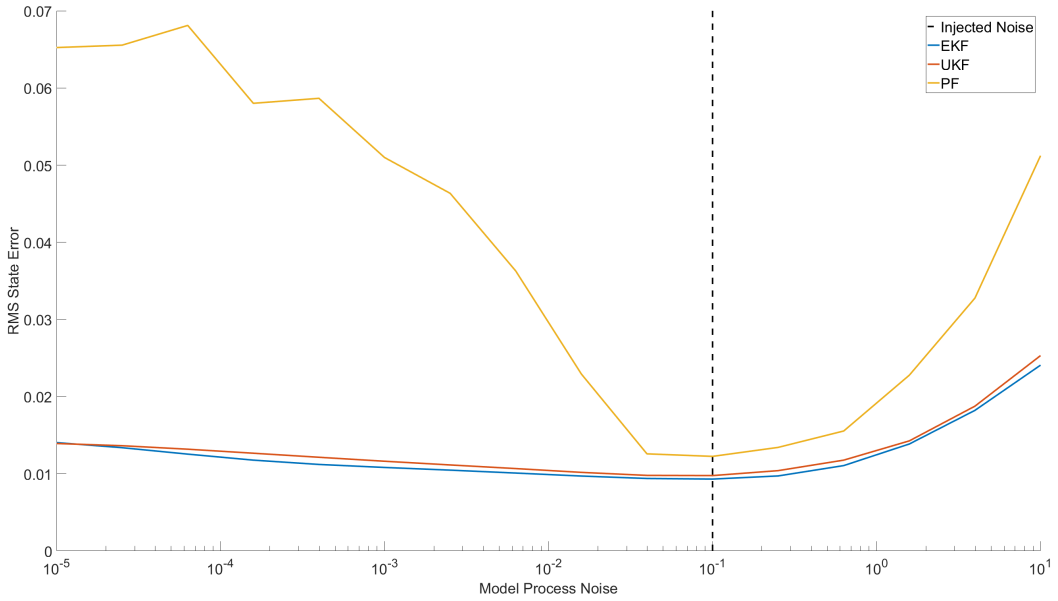

Figure 4.5: Performance of the Extended Kalman Filter (EKF), Unscented Kalman Filter (UKF), and Particle Filter (PF) for different specifications of modeled process noise and given an actual process noise level of 10^{-1}.

between samples is acceptable without increasing the error. This trade-off seems to be logarithmic in measurement noise, but linear in time step. However, this no longer holds once the sampling time step increases substantially above 0.1 sec in this experiment. When this occurs, the integration error of the longer time step yields a poor linear approximation for the nonlinear system. Thus, even with high quality measurements, the EKF cannot learn well from them.

Another important design consideration is the number of PMUs used for the filter (Fig. 4.8). Seven PMUs are necessary to make the local filter work since there are seven buses with generators. We then choose additional PMUs based upon the topology through inspection. We plot the performance of each bus fault (grey) and the mean performance (red). If we place PMUs on all the buses, the average performance only increases 9% over the case where PMUs are only on the generator buses. Thus, we conclude that few PMUs are needed for dynamic state estimation, which is very different than steady state estimation. Moreover, experimental design strategies based upon selecting sensors which maximize the information gain were also considered but saw only marginal improvement.

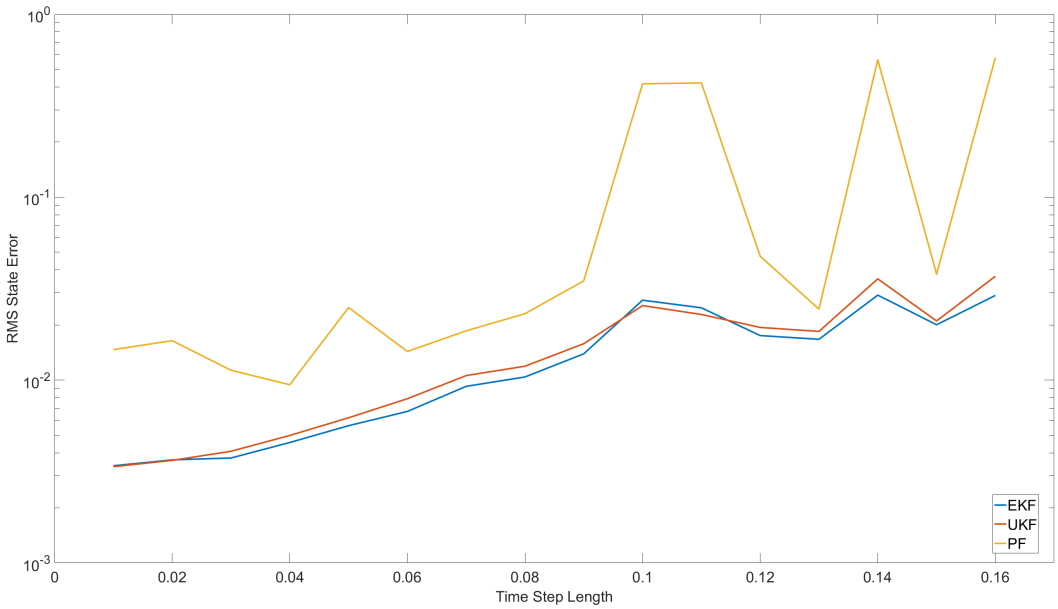

Figure 4.6: Performance of the EKF (blue), UKF (red), and PF (orange) with respect to the sampling and integration time step.

4.5 Case Study: Filtering for a 148-Bus Test System

To investigate the scaling of the estimator, we evaluated its performance on a larger test system comprised of four interconnected versions of the 37 bus systems described above, with the addition of exciters and generator damping to add stability. We apply bus faults to buses 15, 31, and 54 of the original system and 87 and 110, which correspond to 31 and 54 on a connecting system. These faults last between 0.01 and 0.32 seconds. We see in Fig. 4.9 that the error generally increases with the fault length but there is a periodic component which holds for each of the five faults. The error observed in Fig. 4.9 is higher than that seen in Fig. 4.3 due to the increased complexity of the system. We find that the states with the highest error tend to be the generator q-axis transient voltage and the exciter states which control the voltage reference. In general, we find that modeling the exciter system increases the sensitivity to numerical integration errors, requiring our estimator to use intermediate integration steps, which is a topic for future study.

Further, we apply our estimator to this large test system when it is close to instability to evaluate if it can track the system through a catastrophic event and if it can be used for small signal analysis. We see two tests in Fig. 4.10, where a fault on bus 31 lasts for 1.5 seconds (blue - Stable) and for 1.65 seconds (red - Unstable). We see that the estimator can very effectively track the frequency of the generator. Further,

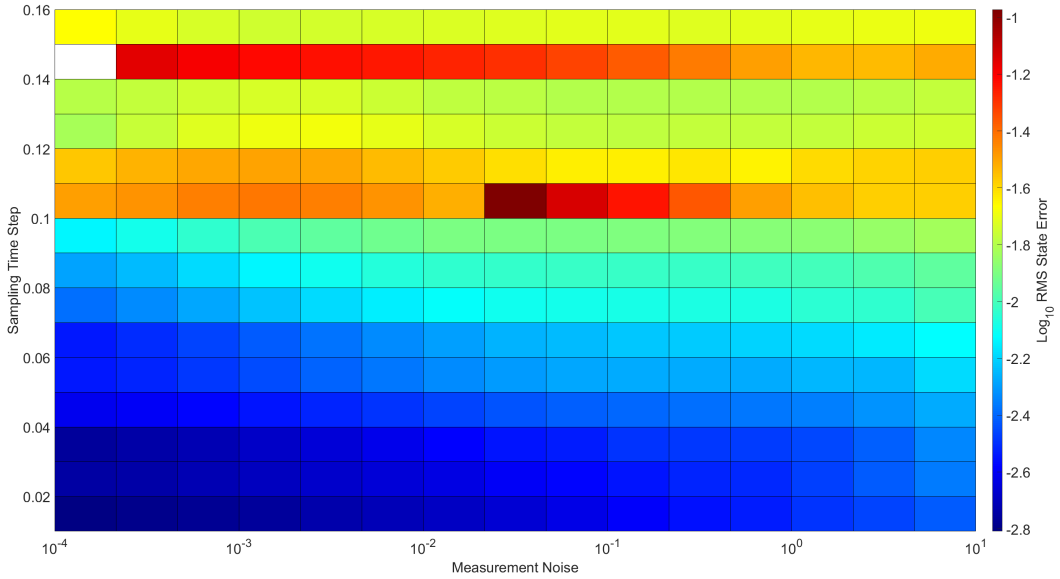

Figure 4.7: Estimator performance under different sampling rates and measurement error intensities

we can see that using dynamic state estimation, we can make real time assessments of the small signal stability of the current operating point, which can be used to predict future stability concerns.

4.6 Fault Detection and Classification

Machine Learning Based Detection and Classification

Fault detection and classification are critical to an effective state estimation system. Many methods have been developed to use PMU data to detect and classify faults quickly[XCK14; Gar+15]. These methods often rely on machine learning techniques to learn subspaces for the PMU data features corresponding to normal operation or different faults. The data features can be extracted using PCA-based dimension reduction as in [XCK14] or Fourier Transforms as in [Gar+15]. For change point detection, the algorithm detects a disturbance when the current feature state moves far away from the nominal set in feature space. Similarly, for classification, the classifier determines when the current feature state moves in feature space from the set corresponding to nominal operation to the set corresponding to a specific fault event. In [Gar+15], this is done using a multinomial regression classifier with less than one second of data form the power system after a fault.

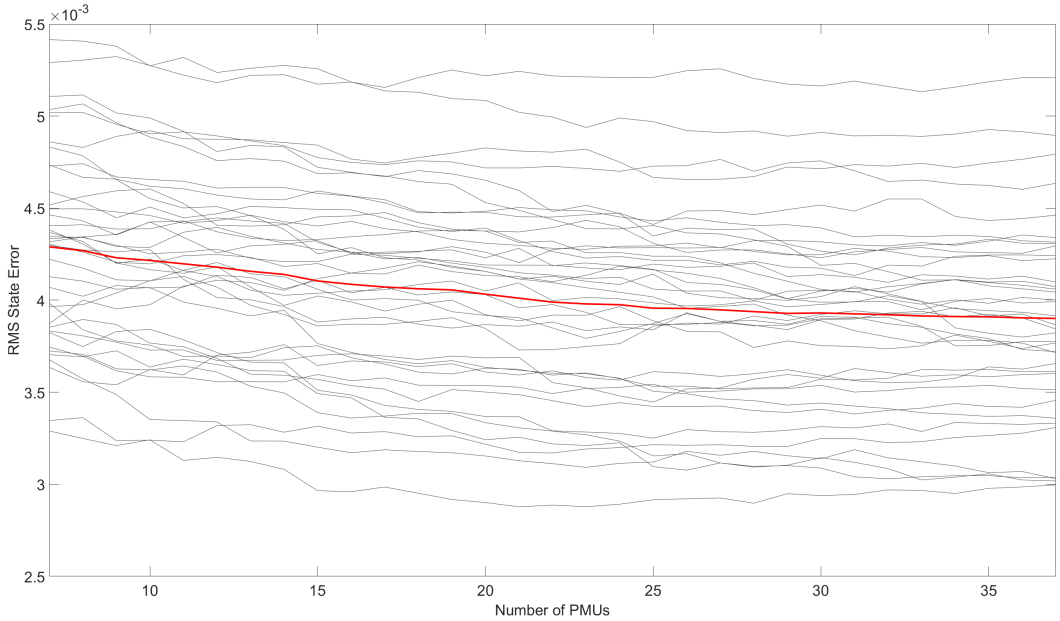

Figure 4.8: Performance impact of using additional PMUs for the local-global state estimator, the figure shows the average error (red) over all possible bus faults (grey).

Robust Detection and Classification using State Estimation

Another approach is to use the current state estimate and physical model to make predictions about the future. We can evolve the current state forward in time using the model and then make predictions about future measurements. When the predicted measurements differ significantly from the observed data, a possible change point can be flagged and different models used to identify the cause of the discrepancy through model selection as discussed in Section 1.2.

The development of fast methods for dynamic state estimation makes robust, model based, fault detection and classification possible. Using these tools, we describe the probability distribution of future inputs with respect to the different models. The observations are then compared to these distributions to assess the likelihood of a disturbance. This allows us to formulate fault detection and classification in the Bayesian context.

While this method is general and can employ any dynamic state estimator, we consider the EKF developed in the previous section. Using this estimator we find that the probability distribution for future measurements conditioned on the previous observations up to time t_k, $z_{1:k}$, is approximated by a multivariate Gaussian:

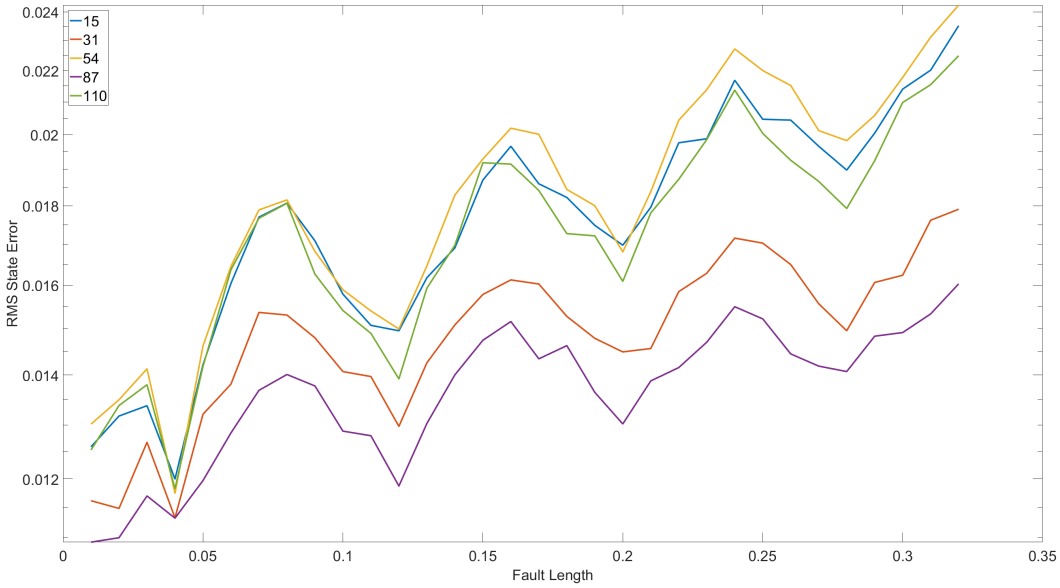

Figure 4.9: Performance of the hybrid estimator for a large 148 bus system for faults on buses 15, 31, 45, 87, and 110, lasting between 0.01 and 0.32 seconds.

$$z_{k+1:k+m} \sim \mathcal{N}\left(\mu_{k+1:k+m}, \Gamma_{k+1:k+m}\right)$$

$$\begin{aligned} \mu_{k+i} = \hat{z}_{k+i} &= \mathfrak{H}\left(\mathfrak{F}^i\left(\hat{x}_{k|k}\right)\right) \\ \Gamma\left(z_{k+i}, z_{k+j}\right) &= H_{k+i}\Big[\textstyle\sum_{l=1}^{i} F_{k+i-1:k+l} Q F_{k+l:k+j-1}^T \\ &\quad + F_{k+i-1:k} P_{k|k} F_{k:k+j-1}^T\Big] H_{k+j}^T + R\delta(i,j) \end{aligned} \tag{4.24}$$

The index m captures how many future observation steps are predicted. In order to perform fault detection, we must assess the likelihood of the observations given normal operating conditions and fault conditions using (4.24). In principle this can be done using a filter bank for the dictionary of faults, but this could become quite computationally intensive. Instead we design a threshold method to discriminate normal operating condition from a likely fault condition and then perform the classification using the full model bank only when a fault is likely. The fault is declared when the error function $\mathcal{E}\left(z_{k+1:k+m}\right)$ exceeds the threshold, λ.

$$\mathcal{E}\left(z_{k+1:k+m}\right) = \frac{\left(z_{k+1:k+m} - \mu_{k+1:k+m}\right)^T \Gamma_{k+1:k+m}^{-1} \left(z_{k+1:k+m} - \mu_{k+1:k+m}\right)}{m|x|} \tag{4.25}$$

The threshold is based upon calculating an error function derived from the chi square statistic. This threshold is designed using Monte Carlo simulations from the

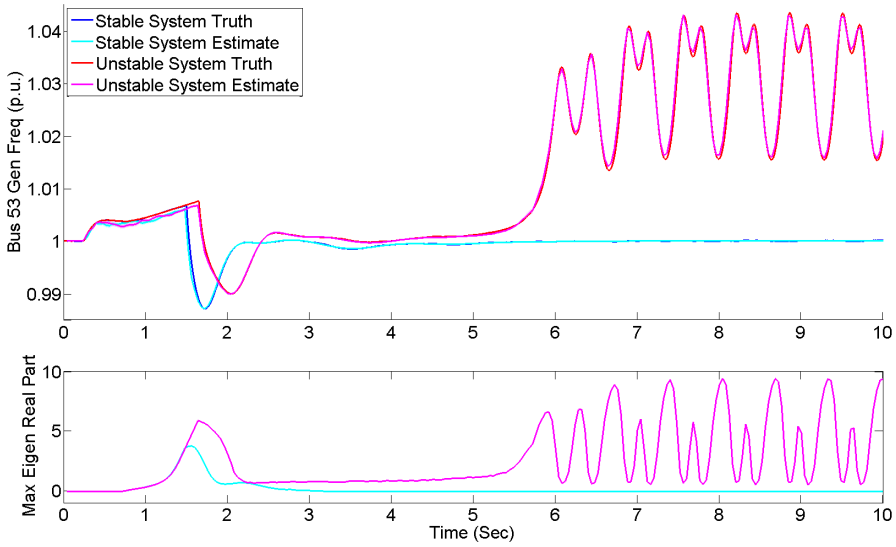

Figure 4.10: a) Comparison of the estimator for tracking a stable (blue) and unstable (red) trajectory. b) Small-signal stability analysis for the system

different operating and fault conditions to minimize the false detection rate with respect to prior likelihood of a fault. The high error indicates that the current model inaccurately tracks the system, meaning that the dynamics have changed so a new model is needed from the model bank.

$$\begin{aligned}\lambda = \operatorname{argmin} & P\left(\mathcal{E}\left(z_m\right) > \lambda \mid z_m \sim \mathcal{M}_0\right) P\left(\mathcal{M}_0\right) \\ & + \textstyle\sum_{i=1}^{N} P\left(\mathcal{E}\left(z_m\right) < \lambda \mid z_m \sim \mathcal{M}_i\right) P\left(\mathcal{M}_i\right)\end{aligned} \tag{4.26}$$

If a fault condition is likely, Bayesian model class selection is done based upon the likelihood of the observations after the fault for a given model from the dictionary of possible disturbances. The unrealized likelihood of a fault model $\mathcal{M}_i$ is given by:

$$\begin{aligned}P\left(\mathcal{M}_i \mid z_{k+1:k+m}\right) &\propto P\left(z_{k+1:k+m} \mid \mathcal{M}_i\right) P\left(\mathcal{M}_i\right) \\ z_{k+1:k+m} \mid \mathcal{M}_i &\sim \mathcal{N}\left(\mu_{\mathcal{M}_i}, \Gamma_{\mathcal{M}_i}\right)\end{aligned} \tag{4.27}$$

4.7 Case Study: Fault Detection and Classification

This integrated dynamic state estimation, fault detection, and fault classification is used to design a robust estimation architecture for the thirty seven bus system shown in Figure 4.2 with PMUs only the generator buses. First, the threshold is selected to optimize the detection of faults by minimizing the faults detection rate as in equation

(4.26). The results are seen in Figure 4.11, where the log false classification rate is plotted for different threshold values and prior fault probabilities. The threshold selected was 60.

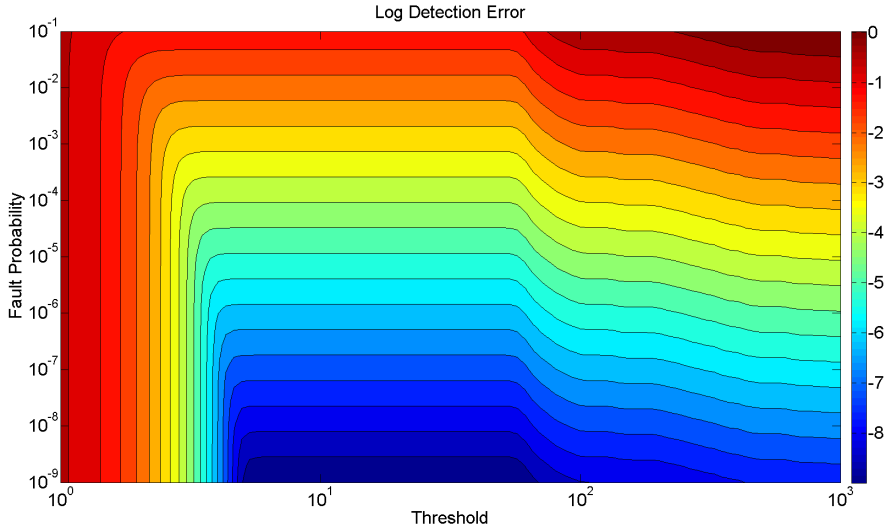

Figure 4.11: False detection rate for different levels of threshold and prior fault probability.

The architecture is then tested using a fault on Bus 37, which is the hardest fault to identify in the network. In Figure 4.12, we see that quickly after the fault occurs the error function spikes, thereby indicating a fault has occurred. It is only after the fault clears that the error measure declines to a small, nominal value.

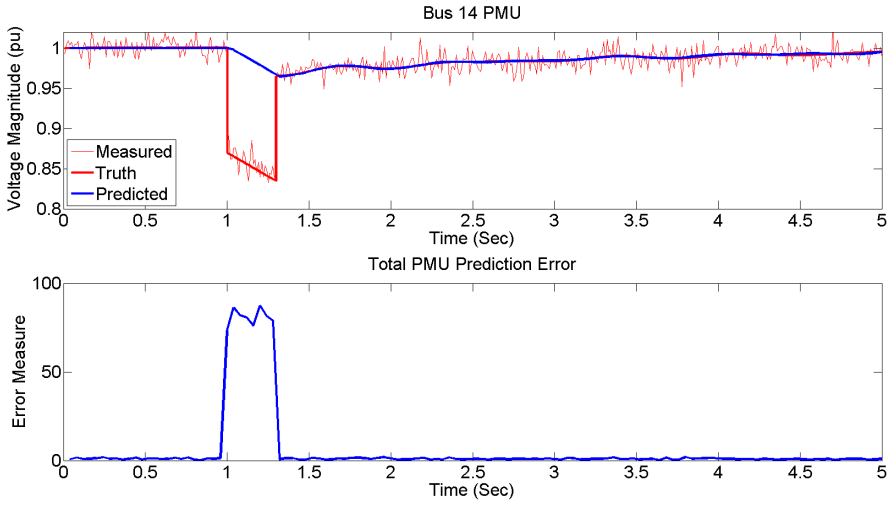

Figure 4.12: Example fault detection for a cleared fault.

Based upon the model bank, using data from the first sample measurement after the fault, the fault can be classified as either on Bus 18 or 37, Figure 4.13, where the colors are for a log base 10 scale. Then as more measurements are used the fault can be quickly and correctly classified as being on Bus 37 with very high probability.

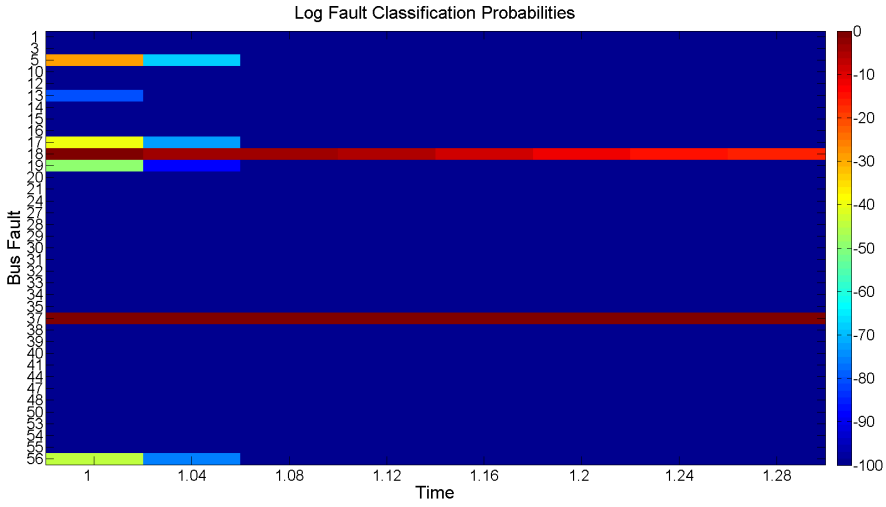

Figure 4.13: Example fault classification as more data is integrated. Colors indicate the log probability of the fault being on a certain bus.

4.8 Decomposition for Power Systems for Estimation

The EKF is known to scale poorly to large state dimensions because it requires computing the Jacobian matrix for the nonlinear dynamics at each time step. To avoid these concerns, we introduce a decentralized implementation of EKF for the power system. This implementation not only enables the estimator to scale to much larger power systems, but it also makes it more robust and easier to apply to real systems where different power system operators control different parts of the power grid. Only local information about each partition is needed for the estimation, so it is more fault tolerant, while still being able to learn the full algebraic and dynamic state.

Under the decentralized implementation, the power grid is partitioned into subsystems whose estimation problem can be solved in real time with the modified EKF. This decomposition should generally minimize the number of links between partitions to reduce the number of uncertain inputs to each partition. While power system decomposition is not the subject of this work, methods for partitioning have been studied in the context of power systems and graph theory in [Hog+13; Cot+13;

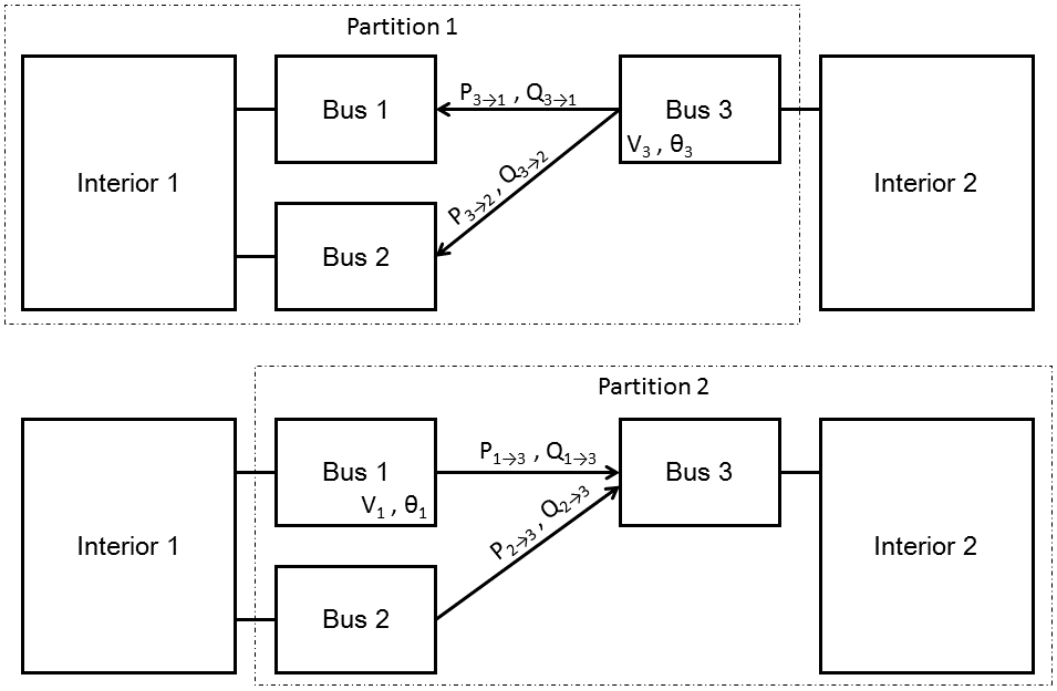

Figure 4.14: Illustration of dividing a power system into two partitions. The power flows on lines connecting the two regions are modeled as uncertain inputs. The voltage phasor on the input bus with the swing bus is also fixed.

LLS10].

The idea is to consider the active and reactive power flow at time t_k on the lines between partitions as an uncertain input u_k into the subsystem. From (4.5), we see that the only way the state in one part of the network effects the state in another part of the network is through the satisfaction of power flow constraints. Therefore by measuring the power flow from one region to another we can capture this dependence. This idea extends the local formulation from Section 4.3 to the more general setting of a partitioned domain. u_k is a function of the measured current and voltage magnitudes (I_{m_k}, V_{m_k}) and angles (I_{θ_k}, V_{θ_k}) as follows:

$$\begin{aligned} u_k &= \left[V^j_{m_k}, V^j_{\theta_k}, I^{jl}_{m_k}, I^{jl}_{\theta_k}\right] \\ P^{jl} &= V^j_{m_k} I^{jl}_{m_k} cos\left(V^j_{\theta_k} - I^{jl}_{\theta_k}\right) \\ Q^{jl} &= V^j_{m_k} I^{jl}_{m_k} sin\left(V^j_{\theta_k} - I^{jl}_{\theta_k}\right) \end{aligned} \tag{4.28}$$

Here, $\{j\}$ are the buses that serve as the input to the subsystem and $\{jl\}$ are lines that connect the input bus j to the rest of the subsystem through bus l. The voltage

on one of the buses from $\{j\}$ is fixed and that bus is used as a swing bus for the partition. The discretization of the DAE (4.1) on each partition is then given as:

$$\begin{aligned} x_k &= x_{k-1} + \tfrac{\Delta t}{2}\left(f\left(x_{k-1}, y_{k-1}, u_{k-1}\right) + f\left(x_k, y_k, u_k\right)\right) \\ 0 &= g\left(x_{k-1}, y_{k-1}, u_{k-1}\right) \\ 0 &= g\left(x_k, y_k, u_k\right) \end{aligned} \tag{4.29}$$

$$\begin{aligned} z_k &= \mathfrak{H}\left(x_k, u_k\right) \\ &= C y_k \end{aligned} \tag{4.30}$$

Estimation can then proceed using the filter introduced in Section 4.3, except that additional uncertainty must be propagated to capture the effect of the uncertain input. We define the input covariance as Q_u. Using equations (4.28) and (4.29), we can find the linearized input dynamics $\frac{dx}{du}$ and $\frac{dy}{du}$ that are needed to propagate the input uncertainty.

4.9 Case Study: Decomposition for 259-Bus Test System

To investigate the scaling of the estimator, we evaluated its performance on a larger test system as seen in Figure 4.15 that is comprised of seven connected versions of the 37 bus systems arranged in a ring. A fault occurs in subsystem 4 near Bus 54, causing a loss of generation, and is cleared half a second later. Further, breakers on the lines connecting subsystems 3 and 4 and subsystems 5 and 6 are open, separating the power system into two islands for 1.25 seconds after the initial fault. In this example, we investigate the response of subsystem 1 to the fault in the remote subsystem 4.

During this simulation, process noise is also added using the same distributions as above. We added exciters to the system to improve stability and make the problem more realistic. The different control system models have exciter states v_m, v_{r1}, v_{r2}, and v_f with variance of 10^{-6}, 10^{-6}, 10^{-8}, and 10^{-8}, respectively, and the turbine governor has state t_g with variance 10^{-6}. PMUs are placed on the input buses and on the generater buses, giving each partition seven PMUs for output measurements and two PMUs for input measurements. The measurement and input variances on the voltage and current are 10^{-4} for both magnitude and phase. Further detail for the model can be seen in [Cat15].

Figures 4.16 and 4.17 show the performance of the decentralized estimator on two generators from the first partition. We can see for both generators that the estimator

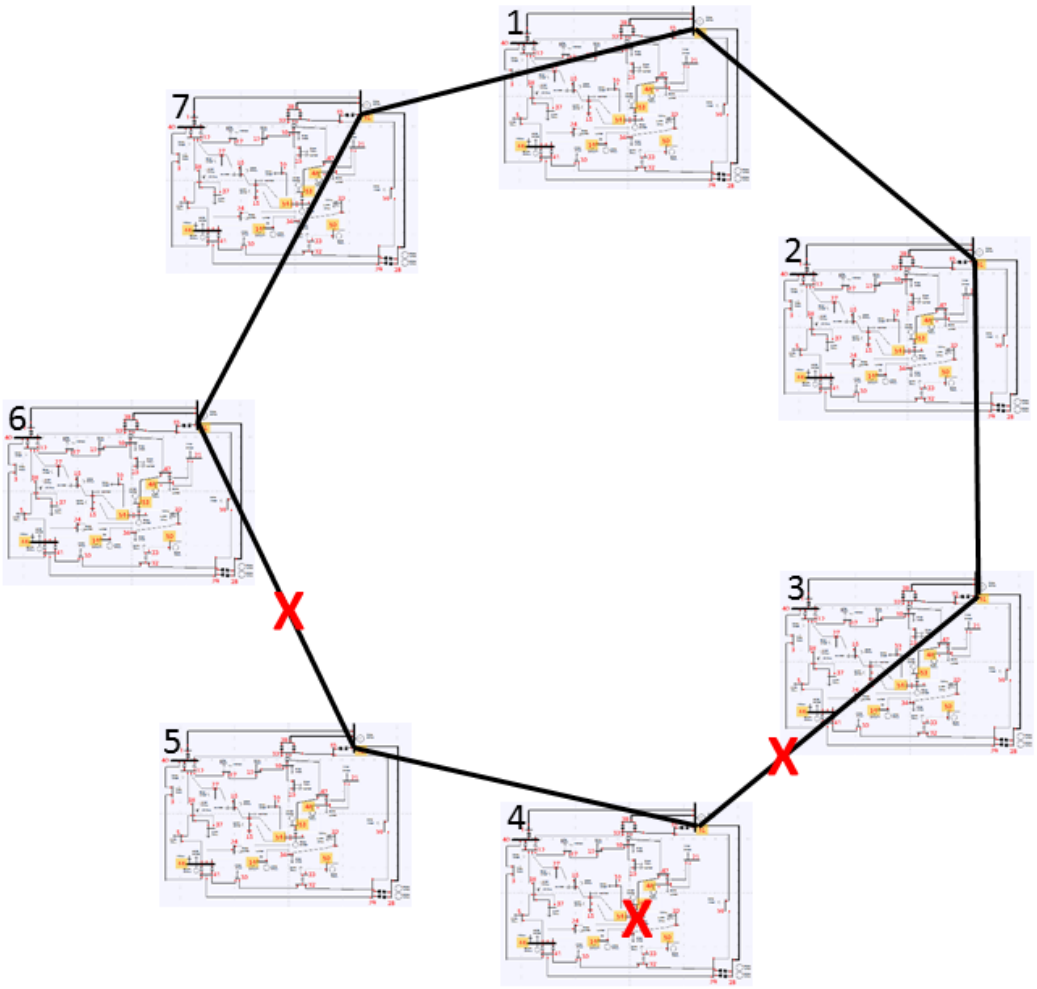

Figure 4.15: Collection of Seven 37-bus subsystems arranged in a cycle. A fault occurs in subsystem 4 and breakers on the lines 3-4 and 5-6 open to temporarily split the cycle into two connected components.

is able to effectively track the dynamic state through the fault, despite the injected process, measurement noise, and use of only local information. Furthermore, the estimated uncertainty generally captures the true uncertainty, despite the linearization of the nonlinear dynamics.

4.10 Discussion

In this work, we evaluated the efficacy of a new framework for integrated dynamic state estimation and fault detection and classification. The state estimator that we developed, is a modified extended Kalman filter using data from PMUs. The EKF uses a linearization of the implicit trapezoidal method to infer the DAE state, instead of the standard Euler linearization. We demonstrate that our method performs as

well as a UKF on a 37-bus test system and that it can solve the DAE faster. We further observe that the implicit method for the EKF performs best among the integration methods we tested. Further, this estimator can be formulated using only local information to enable the estimator to be robust to faults.

Further, we developed a decentralized implementation of the implicit EKF method that can effectively scale to large systems. This method partitions the power system into subsystems and views the power flows between those subsystems, which are measured using PMUs, as uncertain inputs into the local partition. The implicit EKF can then estimate the state of the local generators using only local information and these uncertain inputs. We then demonstrate that this decentralized implementation can still effectively track the dynamic state through a fault disturbance on a 259-bus system.

Then, using the state estimator, we introduced a Bayesian approach to fault detection and classification. This method detects faults when the observed measurements are far from the predicted distribution of expected behaviors of the system based. Then, if a fault is detected, it computes the likelihood of different fault condition models to classify the fault based upon the observed data. The effectiveness of this integrated approach was then demonstrated on the 37-bus test system.

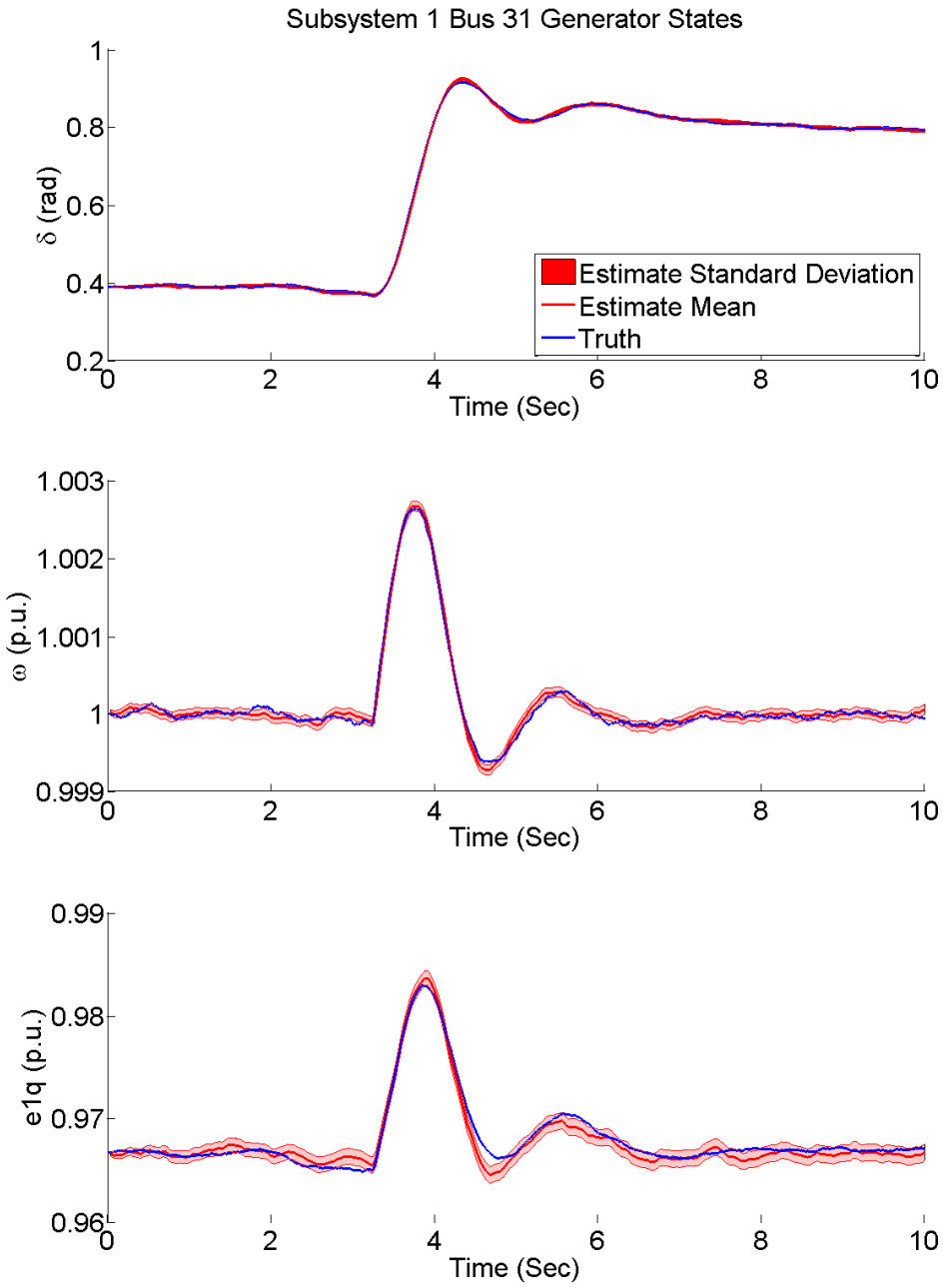

Figure 4.16: Comparison of the estimated (Red) and true dynamic states (Blue) of the generator on Bus 31 of subsystem 1. The estimated one standard deviation uncertainty is shaded red.

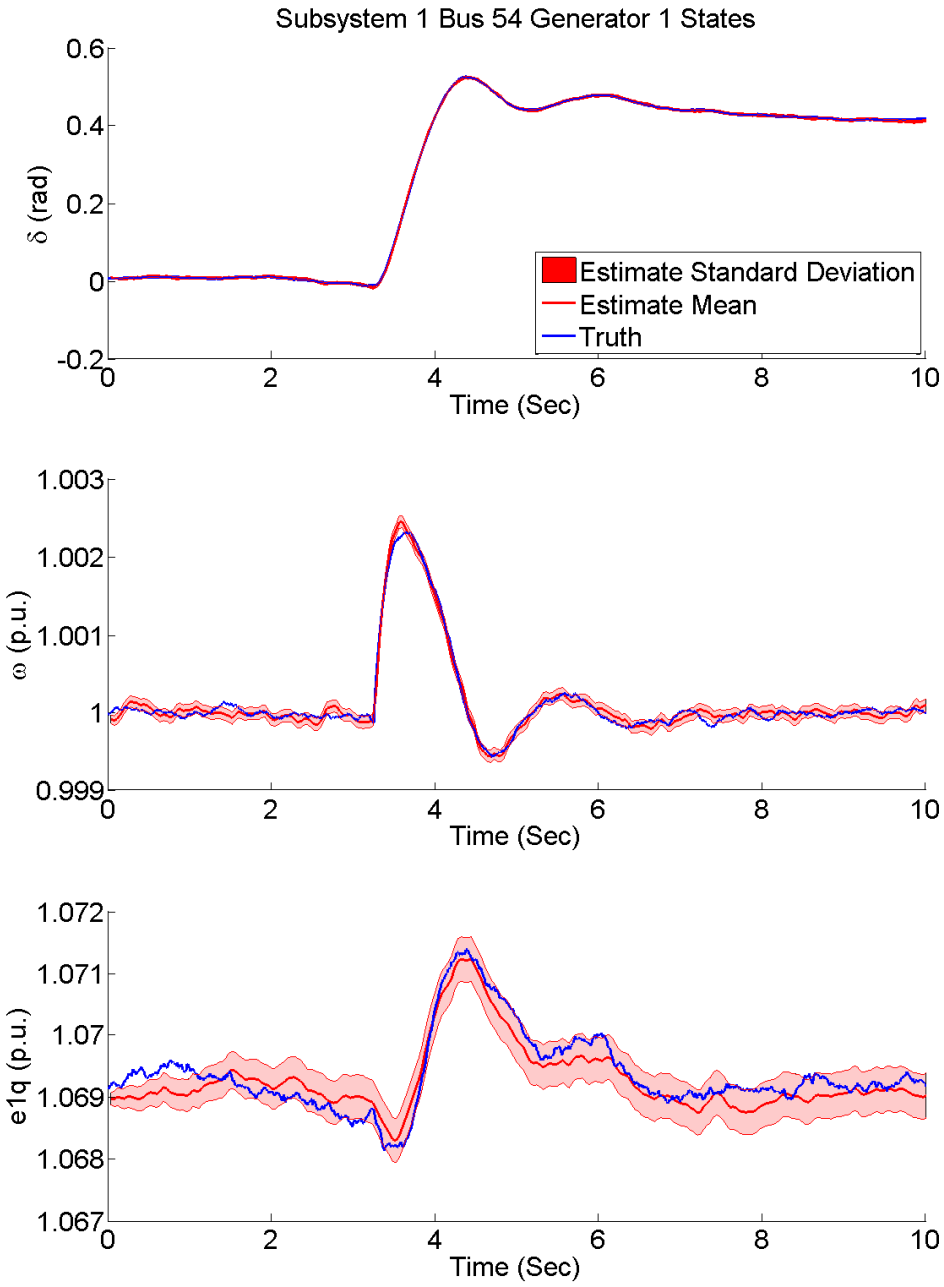

Figure 4.17: Comparison of the estimated (Red) and true dynamic states (Blue) for one of three generators on Bus 54 of subsystem 1. The estimated one standard deviation uncertainty is shaded red.

Chapter 5

INTEGRATED BAYESIAN APPROACH TO A SYNTHETIC BIOLOGICAL EVENT DETECTOR

In this chapter, we present an integrated approach for both unknown input detection and input parameter estimation in a biological context. We use the synthetic biological event detector circuit, introduced in [Hsi+16], to extract information about the chemical recorded in the distributional response of a cell population of a biological sensor. We use the heterogeneous cell population response to infer whether any event has occurred, and if it has, to infer its properties such as timing and amplitude. Bayesian inference provides a natural framework to answer questions about chemical signal occurrence, timing, and amplitude by formulating the problem as model class selection and parameter estimation.

First, we provide motivation and introduce the event detector circuit stochastic model for a heterogeneous cell population. Then we formulate the event detection and inference problem in the Bayesian framework and show how it can be solved using Sequential Tempered MCMC. Finally, we illustrate the results of applying the Bayesian framework to the problem of inferring chemical inducer properties, discuss the results, and make conclusions.

My contribution to this work was to formulate the problem in the Bayesian context and then to utilize the computational methods developed in Chapter 3.2 to solve it. Our Bayesian approach to inferring chemical events is presented in [Bae+16], which forms the basis for this chapter.

[Bae+16] Ania-Ariadna Baetica, Thomas Anthony Catanach, Victoria Hsiao, Richard Murray, and James Beck. “A Bayesian approach to inferring chemical signal timing and amplitude in a temporal logic gate using the cell population distributional response”. In: *bioRxiv* (2016), p. 087379.

5.1 Motivation

Stochastic gene expression poses an important challenge for engineering robust behaviors in a heterogeneous cell population [EE10]. Cells address this challenge by making decisions that consider the distribution of cellular responses during some regulation and differentiation processes [Oga93]. Similarly, the event detector circuit

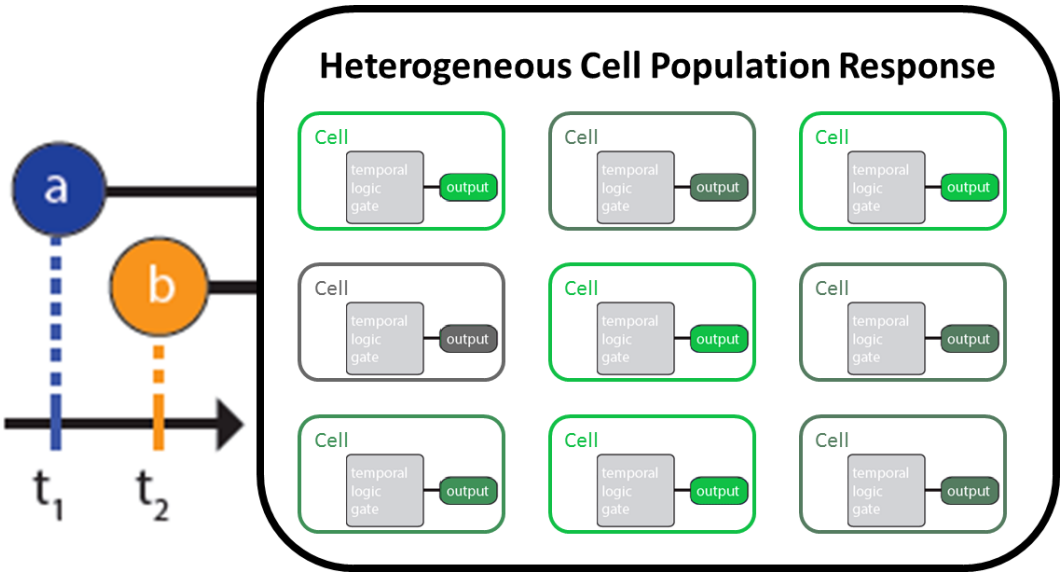

Figure 5.1: Graphic demonstrating the heterogeneous cell population response to two chemical inputs. Colors indicate different cell florescent outputs.

design in Hsiao et al. [Hsi+16] considers the distribution of responses across a cell population. This circuit records the temporal order of two chemical signals. Hsiao et al. [Hsi+16] also used the heterogeneous cell population response to infer the timing and duration of the two chemical signals for a small set of events. A graphical representation of the heterogeneous cell population response to two chemical inputs is given in Figure 5.1.

We develop a probabilistic model based on the event detector circuit model in [Hsi+16] that integrates both predictive modeling uncertainty and sampling error statistics. In this way, we incorporate uncertainty in how well our event detector model captures the cell population and in how well a sample of measured cells represents the entire population. Using our probabilistic model and cell population measurements taken every five minutes on simulated data, we ask how likely it was to observe the data for parameter values that describe square-shaped inducer pulses. The likelihood function associated with the probabilistic model answers the question of how likely the data is by comparing the likelihood values for the model where chemical signal pulses are turned off against the model where the pulses are on. Therefore, we determine whether an event of chemical induction of integrase expression has occurred or not.

Using Sequential Tempered Markov Chain Monte Carlo, we sample the posterior distribution of pulse parameters and then estimate the posterior probability of the

two chemical signal events. We implement this method and obtain accurate results for detecting chemical inducer pulse timing, length, and amplitude. We can detect and identify chemical inducer pulses that fall under biologically relevant conditions.

Using the Bayesian framework to solve our problem enables us to obtain probability distributions over chemical signal occurrence, timing, and amplitude, as well as to test the limits of chemical input identifiability. There are other alternative methods for the problems posed in this work. The first problem of detecting an event falls under the broad field of anomaly and change point detection. Many approaches within this field deal with the problem of using models or approximations to quantify the typical behavior of a system and then setting a threshold to determine whether the signal is within this typical set of behaviors. There are data driven methods like clustering methods or spectral methods that do not need a physical model of the system, but use data to determine a typical set of behaviors [CBK09]. Alternatively, when a physical model is known and computationally tractable, a hypothesis testing framework can be employed. If a set of possible models of the event is also known, a likelihood ratio test is often used to determine whether to accept or reject the hypothesis [B+93]. The methodology of Bayesian model selection enables us to better handle uncertainty within the models, to avoid setting detection thresholds that are not based on probabilities, and to avoid overfitting [Gel+14; Bec10; KR95].

The second problem of identifying the parameters of the chemical inducer pulse is often approximated by solving a maximum likelihood or maximum posterior estimation problem to find the best set of parameters that describe the data [Has+10; Anz12]. While the optimization problem can be nonconvex, several methods have been developed to find good solutions for these problems such as Expectation-Maximization [Anz12]. Maximum likelihood methods can also be integrated into the change point detection framework [B+93]. However, using MCMC to sample the posterior distribution of likely parameters enables us to make a more robust estimate about the set of possible pulse parameters. This also enables us to detect when the pulse is weak since this corresponds to the posterior being very broad and close to unidentifiable.

5.2 System Description

Event detector circuit function

The two-input event detector circuit, also called a temporal logic gate, uses two integrases to engineer an *E. coli* strain with four possible DNA states that record

the temporal order of chemical inputs in the cell population response. Using the heterogeneous response of the *E. coli* population, Hsiao et al. [Hsi+16] infer and record the order of chemical inputs.

The event detector circuit uses serine integrases TP901-1 (int A) and Bxb1 (int B) to flip the DNA between different states. Each cell in the population can be in one of 4 identifiable DNA states: no input (state S_0), only input **a** detected (state S_a), input **b** detected before input **a** (state S_b), or input **a** then **b** detected (state S_{ab}) as illustrated in Table 5.1. Fluorescent proteins are used to read the DNA state of each cell. The change in DNA state caused by detection of either input **a** or **b** is irreversible, and thus the detection is recorded in DNA memory. For more details on the event detector circuit, see reference [Hsi+16].

Stochastic modeling of the event detector circuit using the chemical master equation

To capture the stochastic behavior of the event detector circuit, we model DNA and integrase interactions in each cell using the chemical master equation [Gil92]. We assume that each *E. coli* cell in the population can be uniquely characterized by the triplet of DNA state and molecule numbers of integrases A and B. This state is denoted as (S_i, n_A, n_B), where DNA state $S_i \in \mathbf{S} = \{S_0, S_a, S_b, S_{ab}\}$ and integrase copy numbers $n_A, n_B \in \mathbb{N}_{\geq 0}$. The dynamics of the probability vector, $p(t)$, of a cell being in each state is then:

$$\frac{\partial p(t)}{\partial t} = Ap(t) \tag{5.1}$$

The matrix A is the transition matrix of the continuous-time Markov process that describes the evolution of the cell population. The details of the Markov transitions between states are found in [Bae+16; Hsi+16].

Event	DNA state	Fluorescent Output
None	S_0	None
a only	S_a	RFP
b only	S_b	None
a and **b** both	S_{ab}	GFP

Table 5.1: The table describes the inputs, DNA states, and outputs to the event detector. Table adapted from [Hsi+16].

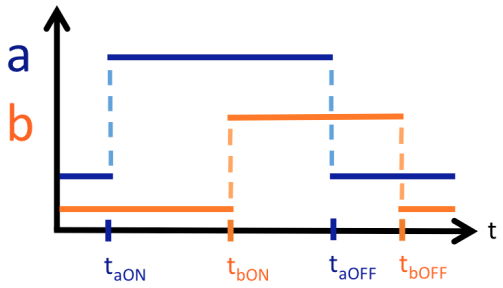

Figure 5.2: We restrict the chemical inducer inputs **a** and **b** to be square waves. They turn on at times t_{aON} and t_{bON} and turn off at times t_{aOFF} and t_{bOFF}.

The chemical inputs alter the Markov transition rates through terms $\gamma_A(t)$ and $\gamma_B(t)$ that represent the production of the two integrases. The chemical inputs that we consider are square waves as illustrated in Figure 5.2. We assume that chemical inputs **a** and **b** are turned off, then turned turn on, and then turned back off. This defines our production rates of integrase molecules A and B, $\gamma_A(t)$ and $\gamma_B(t)$, respectively, as

$$\gamma_A(t) = \begin{cases} k_{\text{prodA}} + k_{\text{leakA}}, \text{ if chemical inducer } \mathbf{a} \text{ is on} \\ k_{\text{leakA}}, \text{ if } \mathbf{a} \text{ is off} \end{cases} \tag{5.2}$$

$$\gamma_B(t) = \begin{cases} k_{\text{prodB}} + k_{\text{leakB}}, \text{ if chemical inducer } \mathbf{b} \text{ is on} \\ k_{\text{leakB}}, \text{ if } \mathbf{b} \text{ is off} \end{cases} \tag{5.3}$$

Here k_{prodA} and k_{prodB} are the production rates of the two integrases, while k_{leakA} and k_{leakB} represent leakiness, which are the background production rates of integrase molecules A and B.

Solving the chemical master equation model

We simulate the CME model of the event detector circuit using the finite state projection algorithm (FSP) in [MK06] as follows. We first transform our three dimensional state space into a one dimensional state vector by iterating over the four DNA states and the pairs of integrase copy numbers. The resulting infinite state space vector is given by

$$p(t) = (P_t(S_0, 0, 0), P_t(S_a, 0, 0), P_t(S_b, 0, 0), P_t(S_{ab}, 0, 0), P_t(S_0, 0, 1), \dots) \tag{5.4}$$

The transition matrix, A, is square and extends to infinity since the state vector has an infinite number of entries. We separate our transition matrix into a sum of a constant matrix and two matrices multiplied by production rates:

$$A = A_{\text{const}} + \gamma_A(t)A_{\text{a}} + \gamma_B(t)A_{\text{b}} \tag{5.5}$$

Matrices A_{const}, A_{a}, and A_{b} are still infinite in both of their dimensions. We truncate them according to the FSP algorithm to a maximum of 20 copies of integrases A and B in each dimension. Following the truncation of the transition matrix using FSP, the CME formulation of the event detector model is

$$\frac{\partial p(t)}{\partial t} = (A_{\text{const}} + \gamma_A(t)A_{\text{a}} + \gamma_B(t)A_{\text{b}})p(t) \tag{5.6}$$

where $p(t)$ is now a truncated version of that in (5.4) of dimension 1600. We solve for the probability distribution over the heterogeneous cell population by computing the standard matrix exponential solution

$$p(t) = E(t, a, b)p(0) \tag{5.7}$$

The matrix $E(t, a, b)$ is the product of exponential matrices according to the ordering and time at which inducers **a** and **b** turn on and off. For example, if inducer **a** turns on instantaneously and inducer **b** turns on at time t_{bON} and they both subsequently remain on, then the expression for $E(t, a, b)$ at time t follows from

$$E(t,a,b) = \begin{cases} \exp[(A_{\text{const}} + (k_{\text{prodA}} + k_{\text{leakA}})A_a)t], \text{ if } t < t_{\text{bON}} \\ \exp[(A_{\text{const}} + (k_{\text{prodA}} + k_{\text{leakA}})A_a)t_{\text{bON}}] \cdot \exp[(A_{\text{const}} + \\ + (k_{\text{prodA}} + k_{\text{leakA}})A_a + k_{\text{leakB}}A_b)(t - t_{\text{bON}})], \text{ if } t \geq t_{\text{bON}} \end{cases} \tag{5.8}$$

Similar expressions for the matrix $E(t, a, b)$ can be derived for any combination of chemical inducers **a** and **b**. The result can be then used in equation (5.7) to derive how the heterogeneous cell population evolves as a function of time by patching together the different solutions for each time interval.

5.3 Bayesian Problem Formulation

In order to formulate the Bayesian inference problem for detecting events and determining the properties of these events, we create the following model classes.

The first model class $\mathcal{M}_0$ describes the dynamics of the cell population when there is no event, while $\mathcal{M}_1$ describes the dynamics of the population when there is an event described as the addition of chemical inducers **a** and **b**. The events in $\mathcal{M}_1$ are parameterized by a vector θ defined in Table 5.2. The parametrization chosen is done to reduce the correlation between states and avoid having prior distributions with bounded support, which can accelerate our sampling methods.

Var	Model Parameter	Prior	Description
θ_1	t_{aON}	Unif $[0, t_{max}]$	Start time of input **a**
θ_2	t_{bON}	Unif $[0, t_{max}]$	Start time of input **b**
θ_3	$\log(t_{aOFF} - t_{aON})$	$\mathcal{N}(\mu_l, \sigma_l)$ s.t. $t_{aOFF} \leq t_{max}$	Log length of input **a**
θ_4	$\log(t_{bOFF} - t_{bON})$	$\mathcal{N}(\mu_l, \sigma_l)$ s.t. $t_{aOFF} \leq t_{max}$	Log length of input **b**
θ_5	$\log(k_{prodA} \cdot (t_{aOFF} - t_{aON}))$	$\mathcal{N}(\mu_{kl}, \sigma_{kl})$	Log of the pulse area i.e. the product of the length and the magnitude of input **a**
θ_6	$\log(k_{prodB} \cdot (t_{bOFF} - t_{bON}))$	$\mathcal{N}(\mu_{kl}, \sigma_{kl})$	Log of the pulse area i.e. the product of the length and the magnitude of input **b**

Table 5.2: The variables, θ, parameterize the event that chemical inducers **a** and **b** are added based upon the start time, end time, and magnitude.

The forward models for $\mathcal{M}_0$ and $\mathcal{M}_1$ have the same dynamics structure, and therefore we can construct the likelihood function for the model class in the same way for observations of the cell population. The forward model describes the evolution of a probability vector over the states, according to the CME. The data we are considering is the observed fraction of cells in a given DNA and molecule number state at an instance in time. In order to find the likelihood of the data given our model, we consider two sources of uncertainty. First, in order to account for predictive modeling errors, we assume that the probability vector $p(t)$, which we model as generating our observation $x(t)$, at time t, is not the same as the one generated by the forward model $\bar{p}(t)$, but comes from a distribution centered around the vector predicted by the forward model. Thus, in our model class, the forward model describes not the evolution of a probability vector, but the evolution of a distribution of probability vectors.

This distribution of probability vectors is modeled by a Dirichlet distribution with parametrization $\alpha(t)$, which we take to be $\alpha(t) = \alpha\bar{p}(t)$. Here α is a constant that controls the variance of the "distribution of distributions", i.e. the believed accuracy of our prediction. Secondly, we model the sampled number of cells in a given state, $x(t)$ at time t using the multinomial distribution defined by the probability vector from the Dirichlet distribution, $p(t) \sim Dir(\alpha(t))$. The Dirichlet distribution is a common choice for quantifying uncertainty about a multinomial distribution since it is the conjugate prior of this distribution. By simultaneously considering these two sources of uncertainty, we are better able to replicate the uncertainty found in real systems. When the number of measured cells is small, the sampling uncertainty will dominate, while when the number of measured cells is large, the prediction uncertainty will dominate. This formulation of the predictive model is summarized in Figure 5.3.

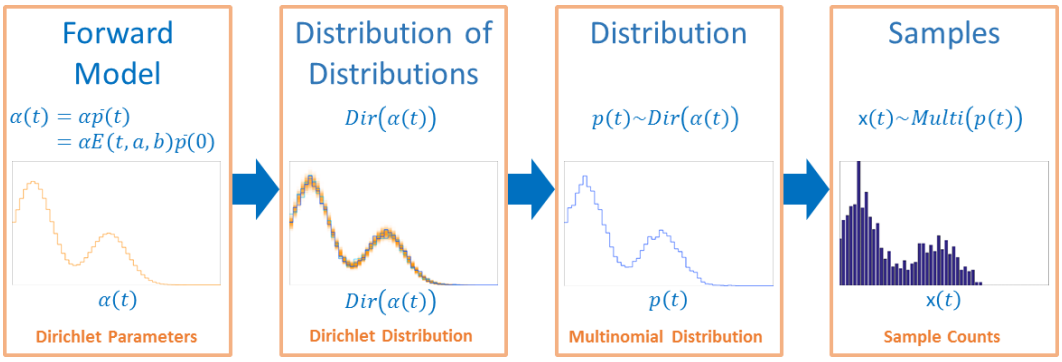

Figure 5.3: The probabilistic model used to construct the likelihood of the observed data is based on two sources of uncertainty: model prediction uncertainty and random sampling errors.

Using this formulation of the model class, we can now define a likelihood function based on the forward model, Dirichlet distribution, and multinomial distribution as follows:

$$\begin{aligned} p(\mathcal{D} \mid \mathcal{M}_0) &\propto \prod_{i=1}^{N} P(x(t_i) \mid \alpha(t_i) = E(t_i, \emptyset, \emptyset)\, p(0)) \\ p(\mathcal{D} \mid \theta, \mathcal{M}_1) &\propto \prod_{i=1}^{N} P(x(t_i) \mid \alpha(t_i) = \alpha E(t_i, a(\theta), b(\theta))\, p(0)) \end{aligned} \tag{5.9}$$

Here $x(t_i)$ are the cell counts at time t_i, $\alpha(t_i)$ is the parametrization of the Dirichlet distribution, and N the number of observations in time. This also models our predic-

tion of cell population measurements as independent in time. We are only interested in likelihood functions up to a constant of proportionality for the computational methods used to solve the inference problem. Thus, equation (5.9) reduces to the following log likelihood function:

$$\begin{aligned} \log p\left(\mathcal{D} \mid \mathcal{M}_0\right) &\propto \sum_{i=1}^{N} \sum_{j=1}^{S} \log \Gamma(x_j\left(t_i\right) + \alpha_j\left(t_i\right)) - \log \Gamma(\alpha_j\left(t_i\right)) \\ \log p\left(\mathcal{D} \mid \theta, \mathcal{M}_1\right) &\propto \sum_{i=1}^{N} \sum_{j=1}^{S} \log \Gamma(x_j\left(t_i\right) + \alpha_j\left(t_i\right)) - \log \Gamma(\alpha_j\left(t_i\right)) \end{aligned} \tag{5.10}$$

Here, $x_j\left(t_i\right)$ is the number of cells in state j at time t_i, $\alpha_j\left(t_i\right)$ is the parametrization of state j of the Dirichlet distribution at time t_i, Γ is the Gamma function, and S is the total number of states in the state vector of the CME.

5.4 Detection and Inference

Assuming that an event occurs, we can use Bayesian inference to infer the posterior distribution of the event parameters conditional on the measured data as discussed in Section 1.2. We use the priors and likelihood functions in Section 5.3 to define the posterior distribution:

$$p\left(\theta \mid \mathcal{D}, \mathcal{M}_1\right) \propto \frac{p\left(\mathcal{D} \mid \theta, \mathcal{M}_1\right) p\left(\theta \mid \mathcal{M}_1\right)}{p\left(\mathcal{D} \mid \mathcal{M}_1\right)} \tag{5.11}$$

Using this function and ST-MCMC, we can then sample the posterior distribution as discussed in Chapter 3.2.

Then using Bayesian model class selection introduced in Section 1.2, we can also consider the probability of any event from $\mathcal{M}_1$ occurring given the cell population measurements, $p\left(\mathcal{M}_1 \mid \mathcal{D}\right)$. We assume that the prior probability of any event happening is known and defined as $p\left(\mathcal{M}_1\right) = 1 - p\left(\mathcal{M}_0\right)$. Therefore, we can perform event detection by computing

$$p\left(\mathcal{M}_1 \mid \mathcal{D}\right) = \frac{p\left(\mathcal{D} \mid \mathcal{M}_1\right) p\left(\mathcal{M}_1\right)}{p\left(\mathcal{D}\right)} \tag{5.12}$$

Using the law of total probability, $p\left(\mathcal{D}\right) = p\left(\mathcal{D} \mid \mathcal{M}_0\right) p\left(\mathcal{M}_0\right) + p\left(\mathcal{D} \mid \mathcal{M}_1\right) p\left(\mathcal{M}_1\right)$, we find that

$$p(\mathcal{M}_1 \mid \mathcal{D}) = \frac{p(\mathcal{D} \mid \mathcal{M}_1)\, p(\mathcal{M}_1)}{p(\mathcal{D} \mid \mathcal{M}_0)\, p(\mathcal{M}_0) + p(\mathcal{D} \mid \mathcal{M}_1)\, p(\mathcal{M}_1)} \tag{5.13}$$

where $p(\mathcal{D} \mid \mathcal{M}_0)$ is defined in equation (5.9) and so it remains to estimate the evidence, $p(\mathcal{D} \mid \mathcal{M}_1)$, using $p(\mathcal{D} \mid \theta, \mathcal{M}_1)$ in equation (5.9), as

$$p(\mathcal{D} \mid \mathcal{M}_1) = \int p(\mathcal{D} \mid \theta, \mathcal{M}_1)\, p(\theta \mid \mathcal{M}_1)\, d\theta \tag{5.14}$$

This integral is estimated using ST-MCMC as discussed in Section 3.2.

5.5 Computational Experiments

In order to judge the efficacy of our Bayesian method, we test it under different conditions. We consider different chemical inducer pulse properties by simulating a biologically relevant long pulse and an unrealistic very short pulse to see the limits of our method. We also consider the case where only the state of the cell population at the end of the simulation is observed and one in which the time series evolution of the cell population is observed. The end point observation case was considered in [Hsi+16] where they try to find the separation time between pulses and the pulse width of pulse **b**. By taking a fully Bayesian approach, we seek to identify a posterior distribution over all pulse parameters. For the Bayesian inference problem, the log pulse duration prior is $\mathcal{N}(2.0, 0.7)$ and the log pulse area prior is $\mathcal{N}(2.5, 1.0259)$. The forward computations of the matrix exponential vector products uses Expokit [Sid98].

Case 1: Nominal pulses using only endpoint data

First, we consider the parameters for a typical event that has inducer **a** on for 7.0 hours and inducer **b** on for 5.0 hours. We use Sequential Tempered MCMC method to generate 1024 posterior samples. In Figure 5.4, we plot the histogram for the posterior samples and the correlation diagrams for each of the parameters. The mean and standard deviation estimates of the posterior distribution are found in Table 5.3 along with the true simulations parameters used to generate the endpoint observation.

Since only the endpoint data was used, the posterior distribution still has significant uncertainty about the pulse characteristics. We see that only the areas for pulses **a**and **b** are well identified since the endpoint data captures the integrated effect of the pulses and not their duration or intensity alone. Furthermore, the correlation

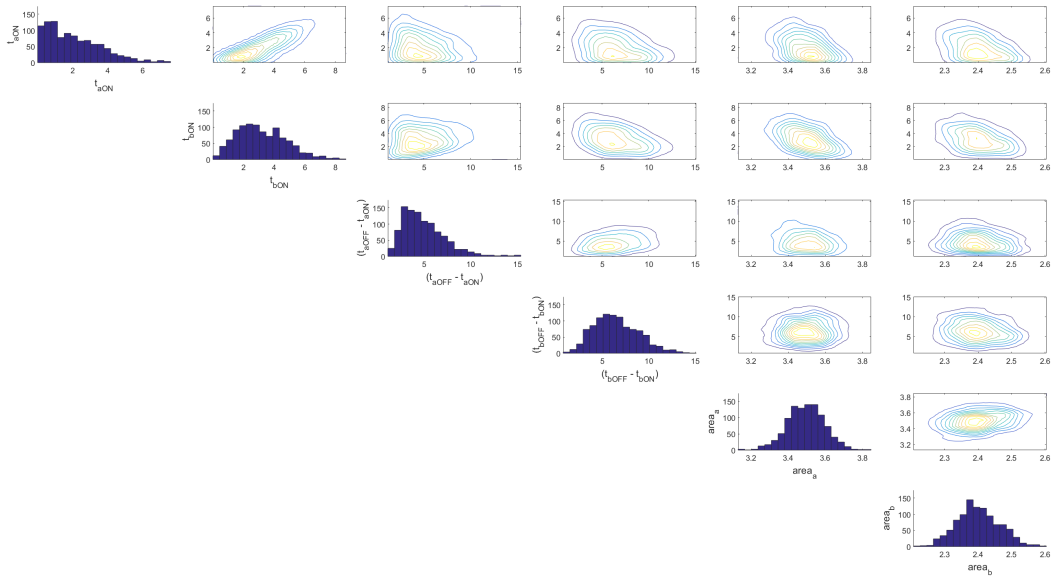

Figure 5.4: Histograms of the posterior sample and kernel density plots showing the correlation in the posterior for Case 1.

Model Variable	True Value	Posterior Mean	Posterior std
t_{aON}	3.0	2.098	1.584
t_{bON}	5.0	3.189	1.609
$(t_{aOFF} - t_{aON})$	7.0	4.997	2.339
$(t_{bOFF} - t_{bON})$	5.0	6.515	2.404
$area_a$	3.5	3.494	0.101
$area_b$	2.5	2.400	0.063

Table 5.3: Posterior estimates for Case 1

between the start times of pulse **a** and pulse **b** indicates that it is highly likely that pulse **b** starts after pulse **a**. Finally, when model class selection is performed to distinguish between the model classes, we find that the likelihood of the pulse events is effectively 1, despite the parameter uncertainty.

Case 2: Nominal pulses using time series data

Next we consider the same pulses but with time series observations every 5 minutes. In Figure 5.5, we plot the histogram for the posterior samples and the correlation diagrams for each of the parameters. The mean and standard deviation estimates of the posterior distribution are found in Table 5.4 along with the true simulations parameters.

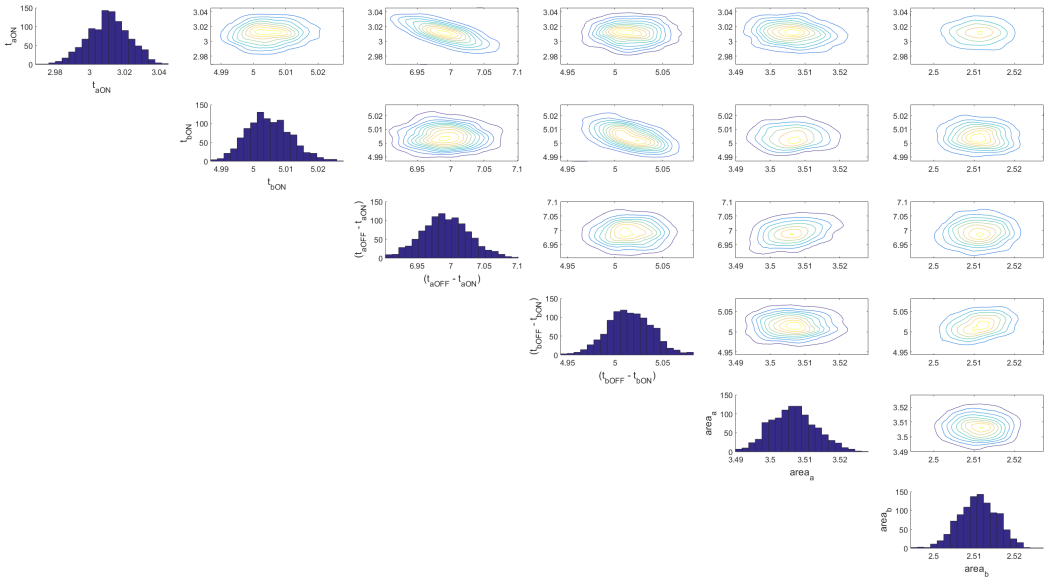

Figure 5.5: Histograms of the posterior sample and kernel density plots showing the correlation in the posterior for Case 2. Note that the scale on the axes differ from Figure 5.4.

Model Variable	True Value	Posterior Mean	Posterior std
t_{aON}	3.0	3.011	0.012
t_{bON}	5.0	5.005	0.007
$(t_{aOFF} - t_{aON})$	7.0	6.993	0.035
$(t_{bOFF} - t_{bON})$	5.0	5.015	0.023
$area_a$	3.5	3.507	0.007
$area_b$	2.5	2.511	0.005

Table 5.4: Posterior estimates for Case 2

Unlike Case 1, using the full time series provides much more information about the properties of the pulses, thus enabling us to identify all of its parameters quite precisely. We see that the posterior is globally identifiable and well concentrated around its mode. This illustrates the value of observing the biosensor over time. Moreover, as for Case 1, we find that the likelihood of the pulse events is effectively 1.

From the computational perspective, we can see that our choice of parameterization was good to produce a globally identifiable posterior distribution. The correlation coefficient for the area and length of **b** is 0.37, while the correlation coefficient between the length and amplitude is -0.91. Thus choosing to use pulse area was a

good choice.

Case 3: Short inducer pulses using only endpoint data

For Case 3, we consider very short inducer pulses that are not biologically motivated and thus very unlikely according to the prior. These short pulses illustrate the limits of the inference and detection framework. The inducer **a** pulse is on for 6.0 minutes and inducer **b** is on for 3.0 minutes, which makes the pulses approximately two orders of magnitude weaker than Case 1. First, we consider the case where only endpoint data is observed. In Figure 5.6, we plot the histogram for the posterior samples and the correlation diagrams, while Table 5.5 presents the mean and standard deviation estimates of the posterior distribution along with the true simulations parameters.

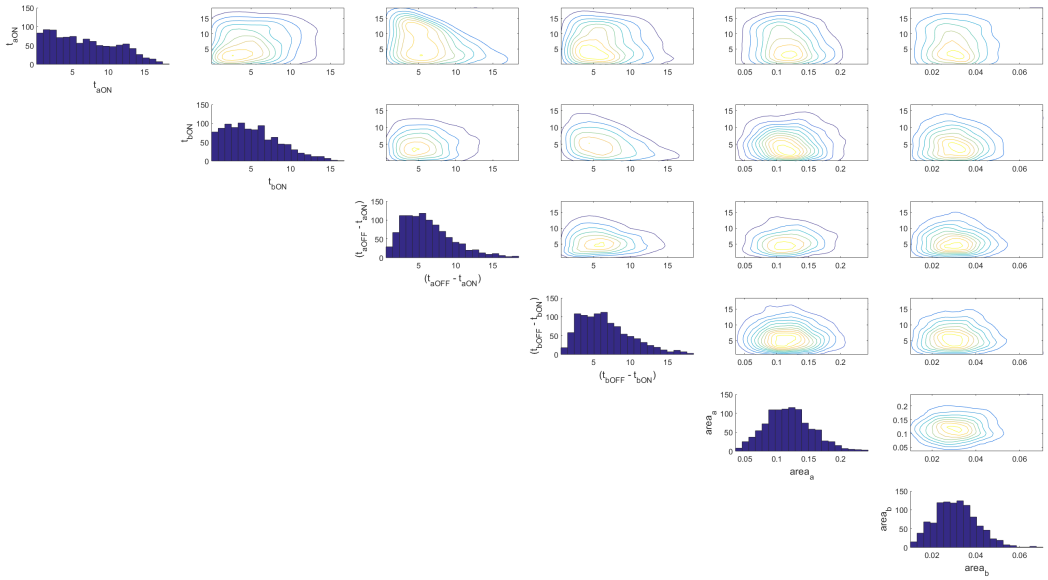

Figure 5.6: Histograms of the posterior sample and kernel density plots showing the correlation in the posterior for Case 3.

Model Variable	True Value	Posterior Mean	Posterior std
t_{aON}	3.0	6.605	4.464
t_{bON}	5.0	5.271	3.419
$(t_{aOFF} - t_{aON})$	0.1	6.294	3.373
$(t_{bOFF} - t_{bON})$	0.05	6.682	3.496
$area_a$	0.05	0.119	0.036
$area_b$	0.005	0.031	0.010

Table 5.5: Posterior estimates for Case 3

We see that the posterior is very uncertain and that the timing parameters are still close to their prior distributions since they are relatively uninformed by the endpoint data. The area parameters are constrained to be very small, although there is uncertainty in how small they should be since their standard deviations are relatively large with respect to their means. Since the prior assumes pulses on the order of hours and with larger areas, we see that the posterior over-estimates these values. When model selection is performed using the endpoint data the likelihood of a pulse is found to be very close to 0. This is because our choice of prior indicates that the posterior pulse parameters are very unlikely and the available data is relatively uninformative, since we are only using end point observations.

Case 4: Short inducer pulses using time series data

Next for Case 4, we consider the case where the time series is observed. In Figure 5.7, we plot the histogram for the posterior samples and the correlation diagrams, while Table 5.6 presents the mean and standard deviation estimates of the posterior distribution along with the true simulations parameters.

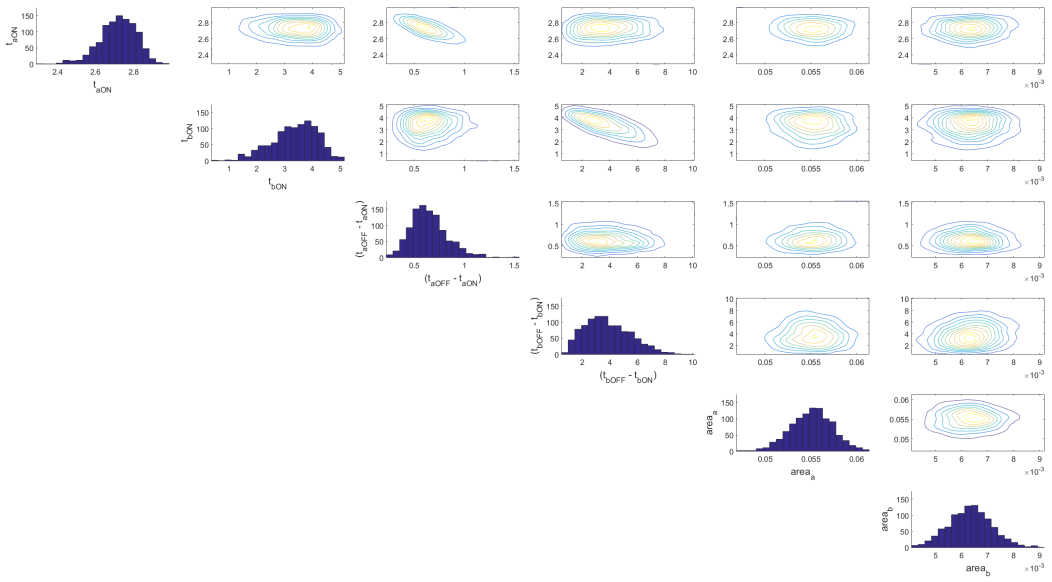

Figure 5.7: Histograms of the posterior sample and kernel density plots showing the correlation in the posterior for Case 4.

We see that the posterior is less uncertain than when only endpoint data is observed. The timing of pulse **a** is relatively well identified, although the length is still over-estimated since the prior prefers longer, lower intensity pulses over shorter, higher intensity pulses. These pulses are more biologically relevant. The timing of pulse

Model Variable	True Value	Posterior Mean	Posterior std
t_{aON}	3.0	2.717	0.096
t_{bON}	5.0	3.357	0.813
$(t_{aOFF} - t_{aON})$	0.1	0.651	0.187
$(t_{bOFF} - t_{bON})$	0.05	3.861	1.662
$area_a$	0.05	0.055	0.002
$area_b$	0.005	0.006	0.001

Table 5.6: Posterior estimates for Case 4

b is less constrained since its intensity is an order of magnitude weaker than pulse **a**. We do find that the areas of the pulses are highly informed and well determined. When model selection is performed using the much richer time series data, the likelihood of pulses being present is now close to 1, even though the inferred pulse parameters are very unlikely with respect to the prior. This shows the value in the time series data for detecting weak pulses.

5.6 Discussion

Using the framework of Bayesian inference, we have answered questions about events recorded in the heterogeneous distributional response of a cell population. We were able to identify the occurrence, timing, and amplitude of chemical inducer pulses in a event detector circuit. We used the cell population response to determine whether an event of chemical induction of integrase expression had occurred. We also obtained accurate results for chemical inducer pulse timing, length, and amplitude using Markov Chain Monte Carlo methods for Bayesian inference. Under biologically relevant conditions, we were able to identify the area and ordering of pulses using endpoint observations and fully identify the pulse using time series information.

In the future, it would be interesting to increase the complexity and the number of parameters that describe the chemical inputs and to solve the inference problem for a chemical inducer class of functions that correspond to square wave trains, which represent inducers that are repeatedly turning on and off. Furthermore, we could seek to use the Bayesian framework to answer questions of experimental design to help improve the design of biosensors and to increase their ability to do pulse detection and inference.

Chapter 6

CONCLUSION

6.1 Summary

In this work, we discussed computational methods for solving Bayesian inference problems and presented several applications of Bayesian methods in complex systems. Bayesian inference can broadly be separated into three types of problems: state estimation, parameter estimation, and model selection. Typically, these inference problems update our understanding about a system on different temporal, spatial, and magnitude scales. Therefore, by thinking about inference as a layered architecture, and utilizing this structure, we presented how inference can be performed quickly and flexibly. However, solving these problems is still often computationally challenging, thus necessitating the development of advanced algorithms that are computationally tractable.

In Chapter 2, we presented the Second-Order Langevin Monte Carlo (SOL-MC) sampler. This sampler is a Markov Chain Monte Carlo (MCMC) algorithm based on an underlying stochastic dynamical system. We showed how this system can be designed to optimize performance and to more efficiently explore and sample the posterior distribution in Bayesian inference. Further, we presented two extensions of this sampler: one for the case where more information about the geometry of the posterior distribution is available for the sampler to use, and another for the case when we allow the sampler to be non-reversible. The effectiveness of this sampler was explored using parameter estimation in a hysteretic structure model and a Bayesian logistic regression problem.

In Chapter 3, we presented advances for Sequential Tempered MCMC (ST-MCMC) algorithms that combine annealing, importance sampling, and MCMC to gradually transform a population of samples from the prior to the posterior through a series of intermediate distribution levels. We derive theoretical results that help tune parameters such as the target correlation and acceptance rate during MCMC sampling, as well as, the amount of variation between each level. Further, we introduced the Rank-One Modified Metropolis Algorithm (ROMMA) to speed up sampling efficiency in high dimensional problems, particularly where the prior has bounded support. These improvements are demonstrated using a constrained Bayesian logis-

tic regression problem and a static fault model inference problem in geophysics.

In Chapter 4, we discussed the application of Bayesian methods to sub-second estimation problems in power network systems. We developed a new Extended Kalman Filter state estimator for the differential algebraic equations that describe the power grid. Using the network structure of the power grid, this estimator can be implemented in a local, distributed, and global context. Further, we formulated the fault detection and classification problem as a Bayesian model class selection problem. Using the dynamic state estimator we developed, we can perform fault detection and classification in under a second of a fault's occurrence. These algorithms provide an integrated approach for power system estimation and fault detection, which was demonstrated on a test system.

Finally, in Chapter 5, we used Bayesian model selection and parameter estimation to detect and identify unknown inputs in a biological system. The goal of this work was to identify the timing and area of two chemical pulses detected by a synthetic biosensor and to see what measurements were necessary for detection and identification. We used ST-MCMC to provide a fast solution to this problem and to demonstrate its effectiveness for a set of realistic chemical pulses.

6.2 Future Work

There are many avenues of future work for the methods developed in this thesis. First, better adaptive methods could be used within SOL-MC and ROMMA to improve their efficiency and scaling to high-dimensions by finding an approximate distribution to the posterior. This approximate distribution could be used within SOL-MC to improve the numerical integration, which would enable longer steps that still have a high probability of acceptance. For ROMMA, the approximate distribution could be used within the rank-one proposal step instead of the prior distribution. Therefore, ROMMA would integrate information from the approximate distribution into the proposal before the final accept or reject step, thereby increasing sampling efficiency.

Another avenue for future work is merging Sequential Tempered MCMC and Transport map accelerated MCMC. The sample population in ST-MCMC could be used to construct a transport map that transforms the complex posterior distribution into a simpler distribution. Sampling would then be done with respect to this simpler distribution. The population would then be transformed back to the original posterior through the inverse transform. Furthermore, since ST-MCMC uses a series of

intermediate distributions to move from the prior to the posterior, the transport map could be iteratively constructed over the course of the levels.

Finally, there are many more opportunities to integrate Bayesian methods into the inference problems described in power systems and biosensors. Specifically in power systems, developing more flexible disturbance identification methods that do not rely on a set of possible disturbances would increase these systems' ability to adapt to variations. As for the synthetic biosensor application, we are interested in identifying more complex pulse profiles and using Bayesian experimental design to optimize the sensor's performance.

BIBLIOGRAPHY

[AB01] Siu-Kui Au and James L Beck. "Estimation of small failure probabilities in high dimensions by subset simulation". In: *Probabilistic Engineering Mechanics* 16.4 (2001), pp. 263–277.

[AD10] David L Alderson and John C Doyle. "Contrasting views of complexity and their implications for network-centric infrastructures". In: *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on* 40.4 (2010), pp. 839–852.

[AE04] Ali Abur and Antonio Gomez Exposito. *Power system state estimation: theory and implementation*. CRC Press, 2004.

[AKP15] Abubeker Alamin, Haris M Khalid, and Jimmy C-H Peng. "Power system state estimation based on Iterative Extended Kalman Filtering and bad data detection using normalized residual test". In: *Power and Energy Conference at Illinois (PECI), 2015 IEEE*. IEEE. 2015, pp. 1–5.

[Anz12] Yuichiro Anzai. *Pattern Recognition & Machine Learning*. Elsevier, 2012.

[APK15] Panagiotis Angelikopoulos, Costas Papadimitriou, and Petros Koumoutsakos. "X-TMCMC: Adaptive kriging for Bayesian inverse modeling". In: *Computer Methods in Applied Mechanics and Engineering* 289 (2015), pp. 409–428.

[APS15] MAM Ariff, BC Pal, and AK Singh. "Estimating dynamic model parameters for adaptive protection and control in power system". In: *Power Systems, IEEE Transactions on* 30.2 (2015), pp. 829–839.

[B+93] Michèle Basseville, Igor V Nikiforov, et al. *Detection of abrupt changes: theory and application*. Vol. 104. Prentice Hall Englewood Cliffs, 1993.

[BA02] James L Beck and Siu-Kui Au. "Bayesian updating of structural models and reliability using Markov Chain Monte Carlo simulation". In: *Journal of Engineering Mechanics* 128.4 (2002), pp. 380–391.

[Bae+16] Ania-Ariadna Baetica, Thomas Anthony Catanach, Victoria Hsiao, Richard Murray, and James Beck. "A Bayesian approach to inferring chemical signal timing and amplitude in a temporal logic gate using the cell population distributional response". In: *bioRxiv* (2016), p. 087379.

[BCD10] Eric Brochu, Vlad M Cora, and Nando De Freitas. "A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning". In: *arXiv preprint arXiv:1012.2599* (2010).

[Bec10] James L Beck. "Bayesian system identification based on probability logic". In: *Structural Control and Health Monitoring* 17.7 (2010), pp. 825–847.

[Bes+11] Alexandros Beskos, Frank J Pinski, Jesús Marıa Sanz-Serna, and Andrew M Stuart. "Hybrid monte carlo on hilbert spaces". In: *Stochastic Processes and their Applications* 121.10 (2011), pp. 2201–2230.

[Bet13] Michael Betancourt. "A general metric for Riemannian manifold Hamiltonian Monte Carlo". In: *Geometric science of information*. Springer, 2013, pp. 327–334.

[Bez+12] Jeff Bezanson, Stefan Karpinski, Viral B Shah, and Alan Edelman. "Julia: A fast dynamic language for technical computing". In: *arXiv preprint arXiv:1209.5145* (2012).

[BK98] James L Beck and Lambros S Katafygiotis. "Updating models and their uncertainties. I: Bayesian statistical framework". In: *Journal of Engineering Mechanics* 124.4 (1998), pp. 455–461.

[BO10] Nawaf Bou-Rabee and Houman Owhadi. "Long-run accuracy of variational integrators in the stochastic context". In: *SIAM Journal on Numerical Analysis* 48.1 (2010), pp. 278–297.

[Bro+11] Steve Brooks, Andrew Gelman, Galin Jones, and Xiao-Li Meng. *Handbook of Markov Chain Monte Carlo*. CRC press, 2011.

[BS11] Michael Betancourt and Leo C Stein. "The Geometry of Hamiltonian Monte Carlo". In: *arXiv preprint arXiv:1112.4118* (2011).

[BT13] James L Beck and Alexandros A Taflanidis. "Prior and posterior robust stochastic predictions for dynamical systems using probability logic". In: *International Journal for Uncertainty Quantification* 3.4 (2013).

[Bus+13] Alberto Giovanni Busetto, Alain Hauser, Gabriel Krummenacher, Mikael Sunnåker, Sotiris Dimopoulos, Cheng Soon Ong, Jörg Stelling, and Joachim M Buhmann. "Near-optimal experimental design for model selection in systems biology". In: *Bioinformatics* 29.20 (2013), pp. 2625–2632.

[BV10] Nawaf Bou-Rabee and Eric Vanden-Eijnden. "Pathwise accuracy and ergodicity of metropolized integrators for SDEs". In: *Communications on Pure and Applied Mathematics* 63.5 (2010), pp. 655–696.

[BZ13] James L Beck and Konstantin M Zuev. "Asymptotically independent Markov sampling: a new Markov Chain Monte Carlo scheme for Bayesian inference". In: *International Journal for Uncertainty Quantification, (2013)* (2013).

[Cat15] Thomas Catanach. *PSAT Power System Models*. 2015. URL: `https://drive.google.com/open?id=0B_g56w3d4uygeTU2R212YktFSGc`.

[Cau63] Thomas K Caughey. “Derivation and Application of the Fokker-Planck Equation to Discrete Nonlinear Dynamic Systems Subjected to White Random Excitation”. In: *The Journal of the Acoustical Society of America* 35.11 (1963), pp. 1683–1692.

[CB09] Sai Hung Cheung and James L Beck. “Bayesian model updating using Hybrid Monte Carlo simulation with application to structural dynamic models with many uncertain parameters”. In: *Journal of Engineering Mechanics* 135.4 (2009), pp. 243–255.

[CB17] Thomas A Catanach and James L Beck. “Bayesian System Identification using Auxiliary Stochastic Dynamical Systems”. In: *International Journal of Non-Linear Mechanics* (2017). DOI: 10.1016/j.ijnonlinmec.2017.03.012. URL: https://doi.org/10.1016/j.ijnonlinmec.2017.03.012.

[CBK09] Varun Chandola, Arindam Banerjee, and Vipin Kumar. “Anomaly Detection: A Survey”. In: *ACM Comput. Surv.* 41.3 (July 2009), 15:1–15:58. ISSN: 0360-0300. DOI: 10.1145/1541880.1541882. URL: http://doi.acm.org/10.1145/1541880.1541882.

[CC07] Jianye Ching and Yi-Chu Chen. “Transitional Markov Chain Monte Carlo method for Bayesian model updating, model class selection, and model averaging”. In: *Journal of Engineering Mechanics* 133.7 (2007), pp. 816–832.

[CFG14] Tianqi Chen, Emily Fox, and Carlos Guestrin. “Stochastic Gradient Hamiltonian Monte Carlo”. In: *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*. 2014, pp. 1683–1691.

[Che03] Zhe Chen. “Bayesian filtering: From Kalman filters to particle filters, and beyond”. In: *Statistics* 182.1 (2003), pp. 1–69.

[Chk+13] Oksana Chkrebtii, David A Campbell, Mark A Girolami, and Ben Calderhead. “Bayesian uncertainty quantification for differential equations”. In: *arXiv preprint arXiv:1306.2365* (2013).

[Cho02] Nicolas Chopin. “A sequential particle filter method for static models”. In: *Biometrika* 89.3 (2002), pp. 539–552.

[Cot+13] Eduardo Cotilla-Sanchez, Paul DH Hines, Clayton Barrows, Seth Blumsack, and Mitesh Patel. “Multi-attribute partitioning of power networks based on electrical distance”. In: *Power Systems, IEEE Transactions on* 28.4 (2013), pp. 4979–4987.

[Cox46] Richard T Cox. “Probability, frequency and reasonable expectation”. In: *American Journal of Physics* 14.1 (1946), pp. 1–13.

[Cox61] Richard T Cox. *The Algebra of Probable Inference*. Johns Hopkins University Press, 1961.

[CV95] Kathryn Chaloner and Isabella Verdinelli. “Bayesian Experimental Design: A Review”. In: *Statistical Science* 10.3 (1995), pp. 273–304. ISSN: 08834237. URL: http://www.jstor.org/stable/2246015.

[DC05] Randal Douc and Olivier Cappé. “Comparison of resampling schemes for particle filtering”. In: *Image and Signal Processing and Analysis, 2005. ISPA 2005. Proceedings of the 4th International Symposium on*. IEEE. 2005, pp. 64–69.

[DC11] John C Doyle and Marie Csete. “Architecture, constraints, and behavior”. In: *Proceedings of the National Academy of Sciences* 108.Supplement 3 (2011), pp. 15624–15630.

[DDJ06] Pierre Del Moral, Arnaud Doucet, and Ajay Jasra. “Sequential monte carlo samplers”. In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 68.3 (2006), pp. 411–436.

[Del04] Pierre Del Moral. “Feynman-Kac Formulae”. In: *Feynman-Kac Formulae*. Springer, 2004, pp. 47–93.

[DP11] David A van Dyk and Taeyoung Park. “Partially collapsed Gibbs sampling and path-adaptive Metropolis-Hastings in high-energy astrophysics”. In: *Handbook of Markov Chain Monte Carlo* (2011), pp. 383–397.

[DS13] Masoumeh Dashti and Andrew M Stuart. “The Bayesian Approach to Inverse Problems”. In: *arXiv preprint arXiv:1302.6989* (2013).

[Dua+87] Simon Duane, Anthony D Kennedy, Brian J Pendleton, and Duncan Roweth. “Hybrid Monte Carlo”. In: *Physics letters B* 195.2 (1987), pp. 216–222.

[EE10] Avigdor Eldar and Michael B Elowitz. “Functional roles for noise in genetic circuits”. In: *Nature* 467.7312 (2010), pp. 167–173.

[Eng] Center for Engineering Strong Motion Data. *Strong-Motion Virtual Data Center*. http://strongmotioncenter.org/vdc/scripts/default.plx.

[Gar+15] M. Garcia, T. Catanach, S.V. Wiel, R. Bent, and E. Lawrence. “Line Outage Localization Using Phasor Measurement Data in Transient State”. In: *Power Systems, IEEE Transactions on* PP.99 (2015), pp. 1–9. ISSN: 0885-8950. DOI: 10.1109/TPWRS.2015.2461461.

[GC11] Mark Girolami and Ben Calderhead. “Riemann manifold Langevin and Hamiltonian Monte Carlo methods”. In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 73.2 (2011), pp. 123–214.

[Gel+14] Andrew Gelman, John B Carlin, Hal S Stern, and Donald B Rubin. *Bayesian data analysis*. Vol. 2. Chapman & Hall/CRC Boca Raton, FL, USA, 2014.

[Gey11] Charles Geyer. “Introduction to Markov Chain Monte Carlo”. In: *Handbook of Markov Chain Monte Carlo* (2011), pp. 3–48.

[Gil92] Daniel T Gillespie. “A rigorous derivation of the chemical master equation”. In: *Physica A: Statistical Mechanics and its Applications* 188.1 (1992), pp. 404–425.

[GK11] Esmaeil Ghahremani and Innocent Kamwa. “Dynamic state estimation in power system by applying the extended Kalman filter with unknown inputs to phasor measurements”. In: *Power Systems, IEEE Transactions on* 26.4 (2011), pp. 2556–2566.

[GSO11] J Duncan Glover, Mulukutla Sarma, and Thomas Overbye. *Power System Analysis & Design, SI Version*. Cengage Learning, 2011.

[GW15] PL Green and K Worden. “Bayesian and Markov Chain Monte Carlo methods for identifying nonlinear systems in the presence of uncertainty”. In: *Phil. Trans. R. Soc. A* 373.2051 (2015), p. 20140405.

[Has+10] Jan Hasenauer, Steffen Waldherr, Nicole Radde, Malgorzata Doszczak, Peter Scheurich, and Frank Allgöwer. “A maximum likelihood estimator for parameter distributions in heterogeneous cell populations”. In: *Procedia Computer Science* 1.1 (2010), pp. 1655–1663.

[HM13] Xun Huan and Youssef M Marzouk. “Simulation-based optimal Bayesian experimental design for nonlinear systems”. In: *Journal of Computational Physics* 232.1 (2013), pp. 288–317.

[Hog+13] Emilie Hogan, Eduardo Cotilla-Sanchez, Mahantesh Halappanavar, Shaobu Wang, Patrick Mackey, Paul Hines, and Zhenyu Huang. “Towards effective clustering techniques for the analysis of electric power grids”. In: *Proceedings of the 3rd International Workshop on High Performance Computing, Networking and Analytics for the Power Grid*. ACM. 2013, p. 1.

[Hor91] Alan M Horowitz. “A generalized guided Monte Carlo algorithm”. In: *Physics Letters B* 268.2 (1991), pp. 247–252.

[HS07] Wolfgang Härdle and Léopold Simar. “Canonical Correlation Analysis”. In: *Applied Multivariate Statistical Analysis*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 321–330. ISBN: 978-3-540-72244-1. DOI: 10.1007/978-3-540-72244-1_14. URL: http://dx.doi.org/10.1007/978-3-540-72244-1_14.

[Hsi+16] Victoria Hsiao, Yutaka Hori, Paul WK Rothemund, and Richard M Murray. “A population-based temporal logic gate for timing and recording chemical events”. In: *Molecular systems biology* 12.5 (2016), p. 869.

[HSN07] Zhenyu Huang, Kevin Schneider, and Jarek Nieplocha. “Feasibility studies of applying Kalman filter techniques to power system dynamic state estimation”. In: *Power Engineering Conference, 2007. IPEC 2007. International*. IEEE. 2007, pp. 376–382.

[Hua+09] Zhenyu Huang, Pengwei Du, Dmitry Kosterev, and Bo Yang. “Application of extended Kalman filter techniques for dynamic model parameter calibration”. In: *Power & Energy Society General Meeting, 2009. PES’09. IEEE*. IEEE. 2009, pp. 1–8.

[Hua+15] Zhenyu Huang, Ning Zhou, Ruisheng Diao, Shaobu Wang, Steve Elbert, Da Meng, and Shuai Lu. “Capturing real-time power system dynamics: Opportunities and challenges”. In: *Power & Energy Society General Meeting, 2015 IEEE*. IEEE. 2015, pp. 1–5.

[Ise09] Arieh Iserles. *A first course in the numerical analysis of differential equations*. 44. Cambridge University Press, 2009.

[Jas+11] Ajay Jasra, David A Stephens, Arnaud Doucet, and Theodoros Tsagaris. “Inference for Lévy-Driven Stochastic Volatility Models via Adaptive Sequential Monte Carlo”. In: *Scandinavian Journal of Statistics* 38.1 (2011), pp. 1–22.

[Jay03] Edwin T Jaynes. *Probability theory: the logic of science*. Cambridge University Press, 2003.

[KB98] Lambros S Katafygiotis and Jim L Beck. “Updating models and their uncertainties. II: Model identifiability”. In: *Journal of Engineering Mechanics* 124.4 (1998), pp. 463–467.

[KBJ14] Nikolas Kantas, Alexandros Beskos, and Ajay Jasra. “Sequential Monte Carlo Methods for High-Dimensional Inverse Problems: A Case Study for the Navier-Stokes Equations”. In: *SIAM/ASA Journal on Uncertainty Quantification* 2.1 (2014), pp. 464–489.

[KFI08] Rambabu Kandepu, Bjarne Foss, and Lars Imsland. “Applying the unscented Kalman filter for nonlinear state estimation”. In: *Journal of Process Control* 18.7 (2008), pp. 753–768.

[KM06] Peter Kunkel and Volker Ludwig Mehrmann. *Differential-algebraic equations: analysis and numerical solution*. European Mathematical Society, 2006.

[KO01] Marc C. Kennedy and Anthony O’Hagan. “Bayesian Calibration of Computer Models”. In: *Journal of the Royal Statistical Society. Series B (Statistical Methodology)* 63.3 (2001), pp. 425–464. ISSN: 13697412, 14679868. URL: http://www.jstor.org/stable/2680584.

[KR95] Robert E Kass and Adrian E Raftery. “Bayes factors”. In: *Journal of the american statistical association* 90.430 (1995), pp. 773–795.

[KSG08] Andreas Krause, Ajit Singh, and Carlos Guestrin. "Near-optimal sensor placements in Gaussian processes: Theory, efficient algorithms and empirical studies". In: *Journal of Machine Learning Research* 9.Feb (2008), pp. 235–284.

[Li+12] Yulan Li, Zhenyu Huang, Ning Zhou, Barry Lee, Ruisheng Diao, and Pengwei Du. "Application of ensemble Kalman filter in power system state tracking and sensitivity analysis". In: *Transmission and Distribution Conference and Exposition (T&D), 2012 IEEE PES*. IEEE. 2012, pp. 1–8.

[LLS10] Juan Li, Chen-Ching Liu, and Kevin P Schneider. "Controlled partitioning of a power network considering real and reactive power balance". In: *Smart Grid, IEEE Transactions on* 1.3 (2010), pp. 261–269.

[MB08] Matthew Muto and James L Beck. "Bayesian updating and model class selection for hysteretic structural models using stochastic simulation". In: *Journal of Vibration and Control* 14.1-2 (2008), pp. 7–34.

[MCF15] Yi-An Ma, Tianqi Chen, and Emily Fox. "A complete recipe for stochastic gradient MCMC". In: *Advances in Neural Information Processing Systems*. 2015, pp. 2899–2907.

[Men+12] Da Meng, Ning Zhou, Shuai Lu, and Guang Lin. "Estimate the electromechanical states using particle filtering and smoothing". In: *Power and Energy Society General Meeting, 2012 IEEE*. IEEE. 2012, pp. 1–7.

[Mil05] Federico Milano. "An open source power system analysis toolbox". In: *Power Systems, IEEE Transactions on* 20.3 (2005), pp. 1199–1206.

[Mil10] F Milano. *PSAT version 2.1.6 User Manual*. 2010.

[MK06] Brian Munsky and Mustafa Khammash. "The finite state projection algorithm for the solution of the chemical master equation". In: *The Journal of chemical physics* 124.4 (2006), p. 044104.

[MSB13] SE Minson, M Simons, and JL Beck. "Bayesian inversion for finite fault earthquake source models I-theory and algorithm". In: *Geophysical Journal International* (2013), ggt180.

[MST94] Donald Michie, David J Spiegelhalter, and Charles C Taylor. "Machine learning, neural and statistical classification". In: (1994).

[Naj09] Habib N Najm. "Uncertainty quantification and polynomial chaos techniques in computational fluid dynamics". In: *Annual review of fluid mechanics* 41 (2009), pp. 35–52.

[Nea11] Radford M Neal. "MCMC using Hamiltonian dynamics". In: *Handbook of Markov Chain Monte Carlo, (2011)* (2011).

[Nea93] Radford M Neal. "Probabilistic inference using Markov chain Monte Carlo methods". In: (1993).

[Oga93] Makio Ogawa. “Differentiation and proliferation of hematopoietic stem cells”. In: *Blood* 81.11 (1993), pp. 2844–2853.

[Ott+16] Michela Ottobre, Natesh S Pillai, Frank J Pinski, and Andrew M Stuart. “A function space HMC algorithm with second order Langevin diffusion limit”. In: *Bernoulli* 22.1 (2016), pp. 60–106.

[Owe13] Art B. Owen. *Monte Carlo theory, methods and examples*. 2013.

[P+10] M Patel, S Aivaliotis, E Ellen, et al. “Real-time application of synchrophasors for improving reliability”. In: *NERC Report, Oct* (2010).

[PC12] Ernesto Prudencio and Sai Hung Cheung. “Parallel adaptive multilevel sampling algorithms for the Bayesian analysis of mathematical models”. In: *International Journal for Uncertainty Quantification* 2.3 (2012).

[Per13] Lawrence Perko. *Differential equations and dynamical systems*. Vol. 7. Springer Science & Business Media, 2013.

[PM14] Matthew Parno and Youssef Marzouk. “Transport map accelerated Markov chain Monte Carlo”. In: *arXiv preprint arXiv:1412.5492* (2014).

[R+01] Gareth O Roberts, Jeffrey S Rosenthal, et al. “Optimal scaling for various Metropolis-Hastings algorithms”. In: *Statistical science* 16.4 (2001), pp. 351–367.

[RC11] Christian Robert and George Casella. “A Short History of MCMC: Subjective Recollections from Incomplete Data”. In: *Handbook of Markov Chain Monte Carlo* (2011), p. 49.

[Rip07] Brian D Ripley. *Pattern recognition and neural networks*. Cambridge university press, 2007.

[RLP16] J. Revels, M. Lubin, and T. Papamarkou. “Forward-Mode Automatic Differentiation in Julia”. In: *arXiv:1607.07892 [cs.MS]* (2016). URL: https://arxiv.org/abs/1607.07892.

[Ros+11] Jeffrey S Rosenthal et al. “Optimal proposal distributions and adaptive MCMC”. In: *Handbook of Markov Chain Monte Carlo* (2011), pp. 93–112.

[RS02] Gareth O Roberts and Osnat Stramer. “Langevin diffusions and Metropolis-Hastings algorithms”. In: *Methodology and computing in applied probability* 4.4 (2002), pp. 337–357.

[Sau11] Peter W Sauer. “Time-scale features and their applications in electric power system dynamic modeling and analysis”. In: *American Control Conference (ACC), 2011*. IEEE. 2011, pp. 4155–4159.

[Sau98] Peter W Sauer. *Power system dynamics and stability*. Prentice Hall, 1998.

[Sch98] Christof Schütte. "Conformational dynamics: modelling, theory, algorithm, and application to biomolecules". In: *Habilitation dissertation, Free University Berlin* (1998).

[Sid98] Roger B Sidje. "Expokit: a software package for computing matrix exponentials". In: *ACM Transactions on Mathematical Software (TOMS)* 24.1 (1998), pp. 130–156.

[Sim06] Dan Simon. *Optimal state estimation: Kalman, H infinity, and nonlinear approaches*. John Wiley & Sons, 2006.

[SP14] Abhinav Kumar Singh and Bikash C Pal. "Decentralized dynamic state estimation in power systems using unscented transformation". In: *Power Systems, IEEE Transactions on* 29.2 (2014), pp. 794–804.

[Sri+09] Niranjan Srinivas, Andreas Krause, Sham M Kakade, and Matthias Seeger. "Gaussian process optimization in the bandit setting: No regret and experimental design". In: *arXiv preprint arXiv:0912.3995* (2009).

[Tao11] Molei Tao. "Multiscale geometric integration of deterministic and stochastic systems". PhD thesis. Citeseer, 2011.

[Thy89] Ravi Shanker Thyagarajan. "Modeling and analysis of hysteretic structural behavior". In: *Doctoral dissertation in Civil Engineering, California Institute of Technology, 1989* (1989).

[TO08] J. E. Tate and T. J. Overbye. "Line Outage Detection Using Phasor Angle Measurements". In: *IEEE Transactions on Power Systems* 23.4 (Nov. 2008), pp. 1644–1652. ISSN: 0885-8950. DOI: 10.1109/TPWRS.2008.2004826.

[TOM10] Molei Tao, Houman Owhadi, and Jerrold E Marsden. "Temperature and friction accelerated sampling of boltzmann-gibbs distribution". In: *arXiv preprint arXiv:1007.0995* (2010).

[Ves08] Fabrizio Vestroni. "Structural Identification Parametric Models and Idefem Code". In: *Dynamic Methods for Damage Detection in Structures*. Springer, 2008, pp. 95–109.

[VT11] Gustavo Valverde and Vladimir Terzija. "Unscented Kalman filter for power system dynamic state estimation". In: *IET generation, transmission & distribution* 5.1 (2011), pp. 29–37.

[W+12] Shaobu Wang, Wenzhong Gao, et al. "An alternative method for power system dynamic state estimation based on unscented transform". In: *Power Systems, IEEE Transactions on* 27.2 (2012), pp. 942–950.

[WH12] K Worden and JJ Hensman. "Parameter estimation and model selection for a class of hysteretic systems using Bayesian inference". In: *Mechanical Systems and Signal Processing* 32 (2012), pp. 153–169.

[Wie+14] Scott Vander Wiel, Russell Bent, Emily Casleton, and Earl Lawrence. "Identification of topology changes in power grids using phasor measurements". In: *Applied Stochastic Models in Business and Industry* 30.6 (2014), pp. 740–752.

[WT11] Max Welling and Yee W Teh. "Bayesian learning via stochastic gradient Langevin dynamics". In: *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*. 2011, pp. 681–688.

[XCK14] Le Xie, Yang Chen, and P Roshan Kumar. "Dimensionality reduction of synchrophasor data for early event detection: Linearized analysis". In: *Power Systems, IEEE Transactions on* 29.6 (2014), pp. 2784–2794.

[Yue10a] Ka-Veng Yuen. *Bayesian methods for structural dynamics and civil engineering*. John Wiley & Sons, 2010.

[Yue10b] Ka-Veng Yuen. "Recent developments of Bayesian model class selection and applications in civil engineering". In: *Structural Safety* 32.5 (2010), pp. 338–346.

[Zho+15] Ning Zhou, Da Meng, Zhenyu Huang, and Greg Welch. "Dynamic state estimation of a synchronous machine using PMU data: A comparative study". In: *Smart Grid, IEEE Transactions on* 6.1 (2015), pp. 450–460.

[ZK11] Konstantin M Zuev and Lambros S Katafygiotis. "Modified Metropolis–Hastings algorithm with delayed rejection". In: *Probabilistic engineering mechanics* 26.3 (2011), pp. 405–412.

[ZML13] Ning Zhou, Da Meng, and Shuai Lu. "Estimation of the dynamic states of synchronous machines using an extended particle filter". In: *Power Systems, IEEE Transactions on* 28.4 (2013), pp. 4152–4161.