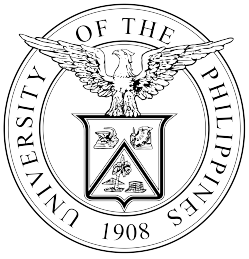
Bayesian Autoregressive Distributed Lag *via* Stochastic Gradient Hamiltonian Monte Carlo

by

Al-Ahmadgaid Bahauddin Asaad

Submitted to the School of Statistics
in partial fulfillment of the requirements for the degree of

Master of Science (M.Sc.) in Statistics



University of the Philippines Diliman
Diliman, Quezon City

May 2017

Abstract

The main objective of the study is the integration of the Stochastic Gradient Hamiltonian Monte Carlo (SGHMC) in estimating the parameters of the Bayesian Autoregressive Distributed Lag (BADL) model. In particular, the *a priori* set for the weights of the BADL is assumed to have fixed hyperparameters, and that is the standard Gaussian distribution. The main theoretical results derived for the proposed model (BADL-SGHMC) are given by: Proposition 5.1.1: the *a posteriori*; and Proposition 5.1.2: the stochastic gradient of the potential energy (the negative log *a posteriori*). The results of the proposed model were applied to forecasting the Philippine's economic growth. In this application, three cases were considered for preliminary evaluation on the behavior of the SGHMC in comparison to other Markov Chain Monte Carlo (MCMC) algorithms, namely: Metropolis-Hasting (MH) and Hamiltonian Monte Carlo (HMC). The markov chains in these three scenarios were assessed using Heidelberger-Welch's stationary test and Gelman-Rubin's convergence test. The results favored BADL-SGHMC over the contenders in the out-of-sample dataset. Lastly, the paper developed statistical packages for SGHMC algorithm both for R and Julia programming languages.

Keywords: *Big Data, Gelman-Rubin, Gross Domestic Product, Growth Rate, Gibbs Sampling, Heidelberger-Welch, Julia, Langevin Dynamics, Laplace's Approximation, Markov Chain Monte Carlo, Metropolis-Hasting, R*

Dedication

To Papa and Mama;
Neng Mirra, Jeh-Jeh and Ikka;
Toh Amil and NG;

and my grandparents:
Mbo' Hja. Tudjuhura and Mbo' Hja. Roswana;
Late Mbo' Hji. Musa and Mbo' Hji. Hussin;

to those who supported my Father's medication;
and to the people of Tawi-Tawi, *lahat ku kalasahan ni entom minsan ma kalawakan*.

Lastly, this work is also dedicated to the leaders
who chose peace over war.

"It is better for a leader to make a mistake in forgiving than to make a mistake in punishing."
Prophet Muhammad (s.a.w.)

"Peace cannot be kept by force. It can only be achieved through understanding."
Albert Einstein

Acknowledgement

(In the name of Allah, the Entirely Merciful, the Especially Merciful)

The author would like to acknowledge: the Department of Science and Technology (DOST) for the financial support; his thesis adviser, Dr. Joselito C. Magadia for the guidance; his thesis panels: Dr. Reynaldo R. Rey and Dr. Josefina V. Almeda, for their comments and suggestions; his uncle, Dr. Abubakar S. Asaad, his boss, Dr. Divina Gracia L. del Prado, and his officemates: Aaron, Ate Mae, Jay, Jhoanne, Ma'am Sarah and Ma'am Thelma for their moral support.

Lastly, the author would like to acknowledge, Ms. Minnette Lois R. Morales for her encouragement and trust.

Thank you po! - AL

Contents

List of Figures

List of Notations

Notation	Description
$\propto$	proportional to
$\mathbb{P}(\cdot)$	probability function
$\mathbb{E}_{\mathtt{p}}$	expected value under the probability distribution $\mathtt{p}$,
$\mathcal{L}(\cdot)$	likelihood function
$\ell(\cdot)$	log-likelihood function
$h(\cdot)$	hypothesis function
$\mathbb{Z}_{+}^{N}$	set of positive integers from 1 to N
$\mathcal{E}_{\text{in}}$	"in-sample error", error in the training dataset
$\mathscr{P}$	set of parameters
$\mathscr{A}\mathscr{B}\mathscr{C}\cdots$	uppercase math scripts for sets
$\triangleq$	equal to by definition
$\multimap$	end of example
$\square$	end of proof

Chapter 1

INTRODUCTION

The discussion in this chapter proceeds with the background and motivation of the study given in Section 1.1, followed by the objectives enumerated in Section 1.2, the scope of the study given in Section 1.3, and the contribution of the study presented in Section 1.4.

1.1 Background and Motivation

In classical statistics, the parameters or weights of interest are assumed to be fixed but unknown. The modeling proceeds by partitioning the dataset into training and testing datasets. The weights are learned in the training dataset, and the evaluation of these weights is done in the testing dataset. The reason for partitioning the data follows from the fact that the weights and the complexity of the model must be carefully tuned in the training dataset, this is because the classical procedure can suffer from the problem of *overfitting* — a situation in which the model tends to memorize the sample and fails to do generalization on the testing dataset.

In Bayesian statistics, there is no problem of overfitting since the parameter of interest are treated as random variable. That is, not only the characteristics of the datasets are being modeled but also the complexity of the model. The complexity of the model often refer to its order, for example the order p for Autoregressive (AR) models, the number of hidden states in Hidden Markov Models (HMM), the number of hidden layers in Artificial Neural Networks

(ANN), and so on. Moreover, inference in Bayesian statistics is based on Bayes' theorem which computes the posterior distribution or the *a posteriori* of the parameters. However, the normalizing constant in Bayes' theorem is often difficult to obtain due to high dimensional integration involve in the computation. This is evident when the number of parameters or weights is huge. To address this limitation, approximation is used in inferring the posterior distribution. The approximation is done through the Markov Chain Monte Carlo (MCMC) algorithms, where the popular MCMC is called the Metropolis-Hasting or MH algorithm. MH depends on a proposal distribution which serves as the random walk in the domain of the posterior distribution. However, like any algorithm, MH has drawbacks. This particular MCMC becomes inefficient when the dimension of the posterior distribution is high. The second algorithm used in this study is called the Hamiltonian Monte Carlo (HMC). HMC is based on Hamiltonian dynamics and addresses the limitation of the MH MCMC. This is possible by using the gradient of the negative log *a posteriori* also known as the potential energy of the Hamiltonian dynamics. The HMC, like MH, uses proposal distribution through an auxiliary variable also referred to as the *Kinetic energy*, and for most studies (*see* Neal, 2011) the proposal distribution is assumed to be standard Gaussian. Theoretically, the proposed samples from this distribution is accepted with certainty. In practice, however, the fact that the Hamiltonian dynamics is simulated over continuous time variable on *phase space*, discretization is required to proceed with the numerical computations. This adjustment is the reason why HMC uses MH acceptance criterion for sampling.

The estimation procedure in frequentist's approach is done either using

Maximum Likelihood Estimation (MLE) or Mathematical Programming also known as Optimization. For optimization in particular, the popular method is the *Gradient Descent* (GD) algorithm which uses batch data (that is, all observations) in updating the parameters at each iteration. This updating can be computationally expensive especially for large datasets. In order to address the limitation, the algorithm must only use 1 or sample of observations (also known as *minibatch*) in updating the parameters. This approach is the motivation of the *Stochastic Gradient Descent* (SGD). In particular, one feature of this procedure is the *stochasticity*, which allows the gradient vector to jump randomly from possible local minima. In Machine and Deep Learning, this method is often (if not always) the choice for efficient estimation of models with nonlinear loss functions such as the popular Artificial Neural Networks (ANN).

To take advantage of the features of SGD, Chen, Fox, and Guestrin (2014) combined SGD and HMC in their article entitled *Stochastic Gradient Hamiltonian Monte Carlo* (SGHMC). Their work has been inspired by Welling and Teh (2011); and Ahn, Korattikara, and Welling (2012). Although Chen et al. (2014) formulated the theory of the SGHMC, there is no discussion as to how well is the mixing of the samples, and empirically analyze the convergence of the algorithm to the posterior distribution. Further, the fact that the SGHMC was published only a couple of years ago, there is no theoretical results yet for simple models estimated using the said algorithm, for example for the case of linear regression models. In this thesis, the objective model is the Autoregressive Distributed Lag (ADL) which is a time series model and specialized type of dynamic linear models (Welty, Peng, Zeger, & Dominici, 2009). In particular, the response variable of this

model is dependent on predictors which includes the *autoregressive* term, which is the lag values of the response; and the *distributed lag* term, which composed of other explanatory variables known (or tested) to have effect on the response variable. This popular model have been used on different field of discipline, from econometrics, epidemiology to agriculture. Further, to the best knowledge of the author, none has yet integrated the SGHMC to Bayesian ADL (BADL). Thus in this paper, the proposed model is abbreviated as BADL-SGHMC.

The ADL model can be estimated using the "unrestricted" approach, where the coefficients are not constrained to some values, hence maximum likelihood estimation can be employed straightforward in this case. However, if the values of the predictors are highly correlated over time, then this will entail difficulty in estimation. According to Welty et al. (2009), a general solution is to constrain the coefficient as a function of lag, and common constraints include polynomial or spline. This practice of constraining the coefficients is already an application of prior knowledge to model specification (Welty et al., 2009). For this thesis, the goal is to demonstrate the integration of the SGHMC on BADL estimation. Specifically, the study considers constraining through specification of the *a priori*, which is uninformative since the parameters are assumed to be governed by a standard Gaussian distribution. Further, as emphasized by Kumar and Maity (2008), the basic assumption of time series models in classical approach is the temporal persistence of the statistical properties. Thus, once the parameters are determined, they are assumed to remain constant over time. This is not a valid assumption, since there are other factors influencing the dynamics which may cause changes in the statistical properties of the time series. In general, uncertainty associated

with observations is considered in traditional modeling. However, uncertainty in model parameter values and model structure, which is another important source of uncertainty is ignored in the frequentist's approaches.

Finally, the effort of scaling the algorithms to large datasets or even to Big data, computationally speaking, the tools for executing this problem rely heavily on the choice of a programming language. Although low-level programming language is always the best choice to achieve performance, scripting or the development stage can be very slow. An example of low-level programming language is C and C++. For most researchers and practitioners, however, the practical solution is to use high-level programming language, sacrificing performance but speeding up the development stage. An example of a popular high-level programming language is R and Python. To address the limitation of both worlds (low-level and high-level programming languages), Bezanson, Edelman, Karpinski, and Shah (2017) proposed Julia as the solution for numerical computation. The first announcement of the language contains the following reason as to why Julia (Bezanson et al., 2017) was created by the team:

> *"We want a language that's open source, with a liberal license. We want the speed of C with the dynamism of Ruby. We want a language that's homoiconic, with true macros like Lisp, but with obvious, familiar mathematical notation like Matlab. We want something as usable for general programming as Python, as easy for statistics as R, as natural for string processing as Perl, as powerful for linear algebra as Matlab, as good at gluing programs together as the shell. Something that is dirt simple to learn, yet keeps the most serious hackers happy. We want it interactive and we want it compiled.*
>
> *(Did we mention it should be as fast as C?)"*
>
> ~Why We Created Julia by Julia Creators (February 14, 2012).

Therefore, implied in the objective, this thesis is also expected to give a brief discussion on the use of Julia in comparison to R in scripting the necessary

algorithms presented in this paper.

1.2 Objectives of the Study

1.2.1 General Objectives

In view of the above discussions, this paper aims for the following general objectives:

1. derive the necessary theoretical results;
2. compare the performance of the proposed model, BADL-SGHMC, for three cases:
   (a) leapfrog step size $\gamma = .09$ for 1,000 iterations;
   (b) leapfrog step size $\gamma = .009$ for 10,000 iterations;
   (c) leapfrog step size $\gamma = .0009$ for 100,000 iterations.
3. apply the proposed model to forecasting Philippine's economic growth; and
4. create software packages for SGHMC for R and Julia programming languages.

1.2.2 Specific Objectives

Specifically, the study is expected to deliver the following:

1. the derivation of the theoretical results:
   (a) given the *a priori* on the parameters of the BADL model, derive the posterior distribution; and
   (b) given the *a posteriori* from the preceding objective, derive the stochastic gradient of the potential energy;

2. the comparison on the performance of the proposed model, BADL-SGHMC, against the performance of the BADL-MH and BADL-HMC. This is done by considering four markov chains for each parameter of the BADL with dispersed random initial values from uniform distribution. The following are the statistical methodologies used for assessing the performance of the models:

   (a) Heidelberger-Welch, for stationarity test on the Markov chains;

   (b) Gelman-Rubin, for convergence test of averages of the Markov chains;

   (c) Autocorrelations, for assessing the assumption of the independent and identically distributed (IID) samples from the *a posteriori*; and

   (d) Root Mean Squared Error (RMSE), for assessing the in-sample and out-of-sample forecast performance of the three models.

   The above procedures are performed across three cases of the leapfrog step size mentioned in the General Objectives;

3. apply the derived theoretical results to forecasting Philippine's year-over-year economic growth rate. In particular, the comparison of the models detailed in the preceding objective is performed using this data. The following are the time series involved in modeling:

   (a) The Response Variable

      - Growth Rate of Gross Domestic Product

   (b) The Predictors

      - Growth Rate of Peso/US Dollar Exchange Rate;

- Growth Rate of Stock Price Index;
- Growth Rate of Gross International Reserves; and
- Growth Rate of Balance of Payments;

4. create software packages for SGHMC for R and Julia programming languages using Github.com as the repository. That is, the package can be installed from this website.

1.3 Scope of the Study

The study is limited to Bayesian Autoregressive Distributed Lag (BADL) model. In particular, the *a priori* set to the parameters of the model is assumed to have fixed hyperparameters, and that distribution is the standard Gaussian. As part of introducing Bayesian inference, Laplace's approximation and Gibbs sampling will be discussed in the basic definitions and basic results of this thesis, but will not be used for modeling BADL. Variational methods and other approximate Bayesian inference are excluded in this study. Further, for purpose of introducing BADL-SGHMC, the application is limited to BADL($p = 1, q = 1$)-SGHMC with SGHMC's parameter $\mathfrak{C}$ (part of the frictional force term) and $\mathfrak{A}$ (part of the variability of the random force term) set to identity matrix. Other orders of p and q, and other possible specification of the SGHMC's parameter, $\mathfrak{C}$ and $\mathfrak{A}$, are part of the recommendation. Finally, in line with other literatures (for example Neal, 2011) for Hamiltonian Monte Carlo, the kinetic energy is based on the assumed distribution of the momentum variable, which in this case is the standard Gaussian density.

1.4 Contribution of the Study

The main contribution of this study is the integration of the Stochastic Gradient Hamiltonian Monte Carlo (SGHMC) on Bayesian Autoregressive Distributed Lag (BADL) parameter estimation. The use of SGHMC is known to be efficient at handling huge datasets compared to Metropolis-Hasting (MH) and Hamiltonian Monte Carlo (HMC) algorithms. Further, the provision of the statistical packages as supplement for this thesis, for executing the theoretical results, is useful for reproducibility and for extended application of the BADL-SGHMC model. Finally, the package is not limited to BADL model but is generalized to account for other interesting models.

1.5 Thesis Organization

The paper is organized as follows: in Chapter 2, the review of literature is presented; Chapter 3, introduces optimization with the aim of tackling Stochastic Gradient Descent (SGD); in Chapter 4, the theory of SGHMC is discussed starting with HMC and Langevin dynamics, and then the theory of the ADL model. Chapter 5 presents the main results which contain the necessary propositions and the application of the proposed model, and Chapter 6 contains the conclusion and recommendations. Lastly, other supplemental results like tables and graphs are provided in the appendices.

Chapter 2

LITERATURE REVIEW

In this chapter, different Bayesian methods for estimating the parameters of the model are reviewed, starting with the Bayes' theorem, which is the foundation of Bayesian modeling, to the modern advances of the subject.

In Bayesian statistics, Reverend Thomas Bayes (*see* Bayes, 1763) is known to be the first to formulate the Bayes' theorem, but the comprehensive mathematical formulation of this result is credited to the works of Laplace (1986). The Bayes' theorem has the following form:

$$\mathbb{P}(\mathbf{w}|\mathbf{y}) = \frac{\mathbb{P}(\mathbf{w})\mathbb{P}(\mathbf{y}|\mathbf{w})}{\mathbb{P}(\mathbf{y})}, \tag{2.1}$$

where $\mathbf{w}$ is the weight vector and $\mathbf{y}$ is the data. This simple formula is the main foundation of Bayesian modeling. Any model estimated using Maximum Likelihood can be estimated using the above conditional probability. As mentioned in the preceding chapter, the Bayes' theorem considers uncertainty not only on the observations but also uncertainty on the weights or the objective parameters. Although Equation (2.1) has simplicity in its form, this expression can be challenging to solve, especially when the dimension of the parameter $\mathbf{w}$ is high, or when the problem involves complex models. Specifically, the main concern lies in the computation of the $\mathbb{P}(\mathbf{y})$, which involve hierarchical summation for discrete variable or high-dimensional integration for continuous variable. This difficulty is far more challenging than the optimization approach of frequentists for problems with no closed-form solution. In order to address the limitation, Hastings (1970) worked on the generalization of the Metropolis, Rosenbluth, Rosenbluth, Teller,

and Teller (1953) algorithm. The idea of the proposed algorithm is to compute the *a posteriori* by approximation through the use of sampling, and this is called Markov Chain Monte Carlo (MCMC). The problem with MCMC, however, is the practical aspect of it since it requires fast processor computers which back then were not yet available. Hence, with the advances in machines, more MCMC algorithms were proposed for efficient sampling. As mentioned in Chapter 1, the Metropolis-Hasting (MH) algorithm has limitations. First, the specification of the proposal distribution is difficult for high-dimensional posterior distribution; and second, the assumption of independent samples is also difficult to achieve since the markov chains often have high autocorrelations.

The limitation of the MH MCMC was addressed by Duane, Kennedy, Pendleton, and Roweth (1987), on their paper entitled "Hybrid Monte Carlo". The idea of this algorithm is based on Hamiltonian dynamics using the concept of Gibbs sampling and MH, hence the name Hybrid (hybrid of Gibbs and MH) or Hamiltonian Monte Carlo (HMC). The algorithm uses auxilliary distribution, which contains the so called *kinetic energy*, along with the target distribution, also called the *potential energy*. However, HMC uses batch gradient descent in its algorithm which can be computationally expensive for large datasets. Thus, the ideal solution is to use samples or only one observation for computing the gradient, and this approach is called stochastic gradient descent or the minibatch-gradient descent. For example, Welling and Teh (2011) worked on combining the stochastic gradient with the Langevin dynamics. The paper is based on Langevin Monte Carlo, which uses gradient steps with injected Gaussian noise into the parameter updates. The gradient step sizes and the variances of the injected noise are balanced so that

the variance of the samples matches that of the posterior. Further, like Hamiltonian dynamics, the Langevin dynamics is motivated and originally derived as a discretization of a stochastic differential equation whose equilibrium distribution is the posterior distribution. However, the problem with the Stochastic Gradient Langevin Dynamics (SGLD) is that the mixing rate of the markov chain is slow. According to Ahn et al. (2012), SGLD takes large steps in directions of small variance and reversely, small steps in directions of large variance which hinders convergence of the Markov chain. Hence, the work of Ahn et al. (2012) is an extension of SGLD, this is done by leveraging the "Bayesian Central Limit Theorem" which states that when N is large (and under certain conditions) the posterior will be well approximated by a Gaussian distribution. The proposed algorithm which is based on Stochastic Gradient Fisher Scoring is designed so that for large step sizes (and thus at high mixing rates) it will sample from this approximate Gaussian distribution, while at smaller step sizes (and thus at slower mixing rates) it will generate samples from an increasingly accurate (non-Gaussian) approximation of the posterior. Similar to SGLD, another MCMC algorithm proposed is also based on stochastic gradient noise but using Hamiltonian dynamics instead of the Langevin. The algorithm builds on top of Hamiltonian Monte Carlo in the work of Chen et al. (2014) entitled "Stochastic Gradient Hamiltonian Monte Carlo" (SGHMC). The idea of this algorithm is to consider a frictional force other than the injected Gaussian noise. This force will align the algorithm to the stationary distribution, which is the *a posteriori*. In the article, Chen et al. (2014) demonstrated SGHMC on problems involving classification and probabilistic matrix factorization. The findings suggest that the SGHMC performed well compared to SGLD. However, the lack on

the empirical convergence diagnostics on the experiment done by Chen et al. (2014), and the comparison of the SGHMC to the non-stochastic gradient MCMC are the main objective of this thesis, and this is done using Bayesian Autoregressive Distributed Lag (BADL) model. In literature, there are already studies on Bayesian ADL (BADL), *see* for example Ravines, Schmidt, and Migon (2006); their worked on ADL is based on Gibbs sampling; Welty et al. (2009) also worked on Distributed Lag (DL) models by comparing Bayesian DL (BDL) through Gibbs sampling with the frequentist's approach to estimation; and Buss (2010) used BADL on optimal prior for economic forecast. Finally, to the best knowledge of the author, there is no research yet on the use of SGHMC for BADL estimation.

Chapter 3

BASIC DEFINITIONS AND RESULTS

The main subjects of this chapter are the basic concepts and results for Bayesian inference and mathematical optimization. These include an overview of Markov Chain Monte Carlo (MCMC) algorithms such as the Metropolis-Hasting (MH) and the Gibbs sampling. Inference on posterior summaries through convergence analysis of the markov chains are also provided, and some insights to the general applicability of these tools are illustrated using Bayesian linear regression model.

3.1 Gradient Descent

There are several ways to numerically estimate the parameters of the model using mathematical programming. The popular algorithm that is very common in Machine Learning is the *gradient descent* (GD) given in Algorithm 1. Suppose $\nabla\mathcal{E}_{\text{in}}(\hat{\mathbf{w}}^{(r)})$ is the gradient of the cost function at the rth iteration. $\mathcal{E}_{\text{in}}$ is defined as the *in-sample error* or the error in the training dataset, γ is the *learning-rate* parameter of the algorithm and ν is the *precision* parameter. As an illustration, consider Example 3.1.1.

Example 3.1.1. Suppose the loss function is given by

$$\mathcal{E}_{\text{in}}(w) \triangleq w^4 - 3w^3 + 2. \tag{3.1}$$

The first derivative of the above equation with respect to w is given by $\mathcal{E}'_{\text{in}}(w) = 4w^3 - 9w^2$. Let the initial guess be $\hat{w}^{(0)} = .1$ and let $\gamma = .01$ with $\nu = .00001$. Then $\nabla\mathcal{E}_{\text{in}}(\hat{w}^{(0)}) = \mathcal{E}'_{\text{in}}(\hat{w}^{(0)}) = -0.086$, so that $\hat{w}^{(1)} \triangleq \hat{w}^{(0)} - .01(-0.086) = 0.10086$,

Algorithm 1 *Gradient Descent*

1: Initialize $\hat{\mathbf{w}}^{(r)}, r = 0$
2: **while** $||\hat{\mathbf{w}}^{(r+1)} - \hat{\mathbf{w}}^{(r)}|| > \nu$ **do**
3: $\quad \hat{\mathbf{w}}^{(r+1)} \triangleq \hat{\mathbf{w}}^{(r)} - \gamma \nabla \mathcal{E}_{\text{in}}(\hat{\mathbf{w}}^{(r)})$
4: $\quad r \triangleq r + 1$
5: **end while**
6: **return** $\hat{\mathbf{w}}^{(r)}$ and r.

and $|\hat{w}^{(1)} - \hat{w}^{(0)}| = 0.00086 > \nu$. It turns out that 173 iterations are needed to satisfy the inequality, $|\hat{w}^{(r+1)} - \hat{w}^{(r)}| \not> \nu$. The plot is given in Figure 3.1. ⊸•

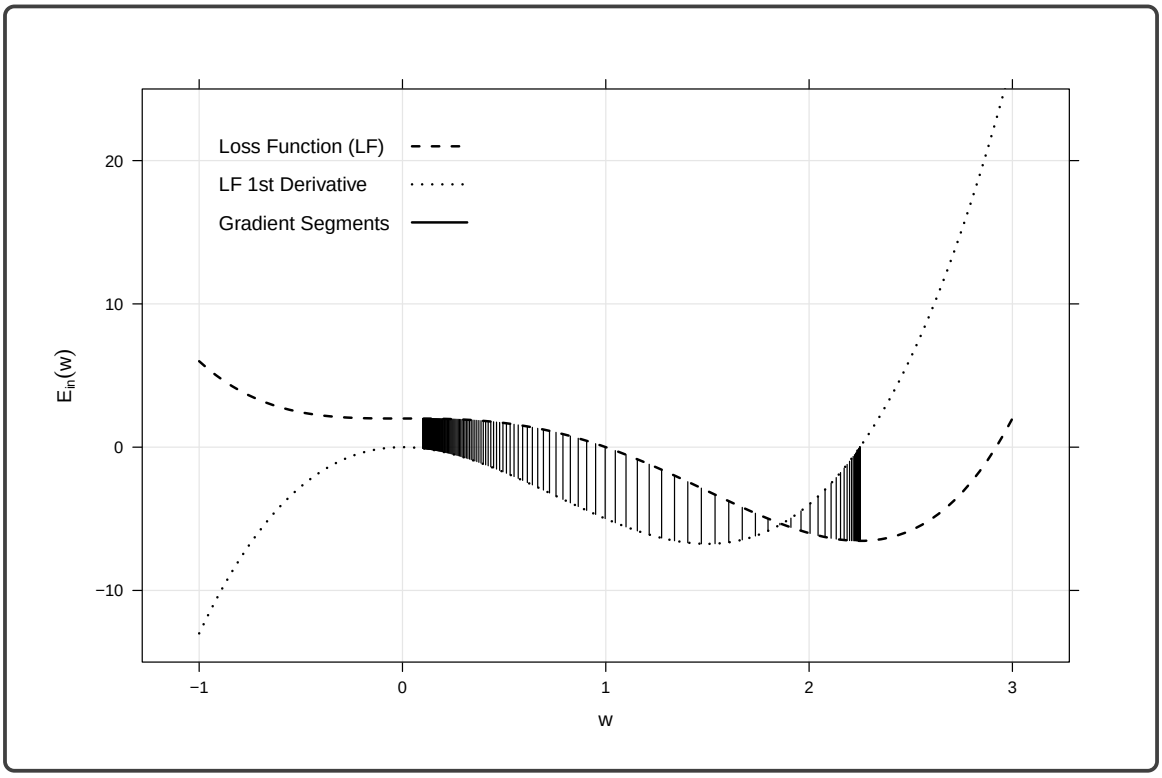


Figure 3.1: *Gradient Descent on Equation (3.1).*

In practice, however, there are hundreds to millions of data points that need to be summarized, so that at each iteration, the parameters are updated *after* the presentation of *all* the training examples that constitute an *epoch* — one complete presentation of the entire training dataset during the learning process (Haykin, 1998). In this setting, GD is sometimes called *batch gradient descent* (BGD).

3.2 Stochastic Gradient Descent

An alternative to BGD is SGD or *stochastic gradient descent*. SGD updates the parameter using only one observation for every iteration, which is a lot faster. Further, BGD is prone to local minimum since GD does so. This is guaranteed for misspecified initial value especially for high dimensional nonlinear error surface function. The stochasticity of the SGD follows from the randomization of the dataset at each epoch, and contrary to BGD, the SGD is not expected to converge to the global minimum, instead it will only stay around the global solution (*see* Algorithm 2). Example 3.2.1 illustrates the application of mathematical programming in estimating the parameters of a simple linear regression model.

Algorithm 2 *Stochastic Gradient Descent*

1: Initialize $\hat{\mathbf{w}}^{(r)}, r = 0$
2: **while** $\|\hat{\mathbf{w}}^{(r)} - \hat{\mathbf{w}}^{(r+1)}\| > \nu$ **do**
3: Randomize the data set $(\mathbf{x}_i, \mathbf{y}_i)$ with respect to i.
4: **for** $i \in \{1, \cdots, n\}$ **do**
5: Update the parameters

$$\begin{aligned}\hat{\mathbf{w}}^{(r)} &\triangleq \hat{\mathbf{w}}^{(r)} - \gamma \nabla \mathrm{e}(h(\mathbf{x}_i, \hat{\mathbf{w}}^{(r)}), \mathbf{y}_i) \\ &= \hat{\mathbf{w}}^{(r)} - \gamma \frac{\partial}{\partial \mathbf{w}} \left\{ \frac{1}{2} [h(\mathbf{x}_i, \hat{\mathbf{w}}^{(r)}) - \mathbf{y}_i]^2 \right\}\end{aligned}$$

6: **end for**
7: $\hat{\mathbf{w}}^{(r+1)} \triangleq \hat{\mathbf{w}}^{(r)}$
8: **end while**
9: **return** $\hat{\mathbf{w}}^{r}$ and r.

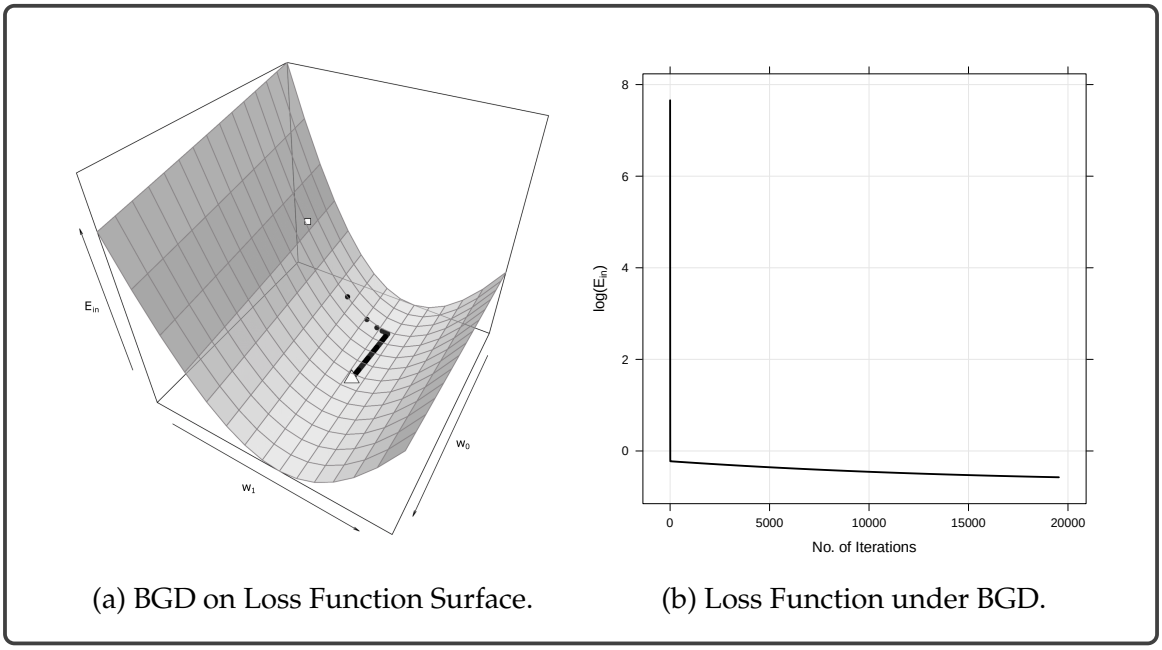
Example 3.2.1. Suppose the explanatory variable is simulated from a Gaussian distribution with $\mu \triangleq 15$ and $\sigma \triangleq 2$ for 100 samples. Let the true parameter be $w_0 \triangleq 3.4$ (intercept) and $w_1 \triangleq .75$, the precision parameter of the BGD be $\nu \triangleq .0001$ with learning rate $\gamma \triangleq .002$ and initial guess $w_0^{(0)} \triangleq -3$ and $w_1^{(0)} \triangleq -3$. The batch

gradient descent algorithm returns the following estimates: $\hat{w}_0^{(19,544)} = 0.2200854$ and $\hat{w}_1^{(19,544)} = 0.9549819$ at 19,544th iteration, *see* Figure 3.2.

In comparison, the stochastic gradient descent algorithm takes only 4,614 iterations to converge with the following final estimates: $\hat{w}_0^{(4,614)} = 2.732653$ and $\hat{w}_0^{(4,614)} = 0.814412$. Thus SGD performs well compared to BGD in terms of speed in convergence. Further, the ordinary least square (OLS) estimates, $\hat{w}_0^{(\text{OLS})} = 2.7130$ and $\hat{w}_1^{(\text{OLS})} = 0.7973$, are a lot closer to the SGD's solution compared to BGD. This is evident in the last 500 steps of the SGD which shows random fluctuations of the gradient vectors as seen in Figure 3.4. ⊸•

3.3 Laplace's Approximation

The simplest approximation to the posterior distribution of the parameter of interest is the Laplace's approximation. The idea behind this procedure is to use a Gaussian approximator, $\mathbb{G}(x)$, such that it is centered on the mode of the target



(a) BGD on Loss Function Surface. (b) Loss Function under BGD.

Figure 3.2: *Batch Gradient Descent on SLR Loss Function.*

distribution, $\mathbb{P}(x)$. To illustrate, suppose

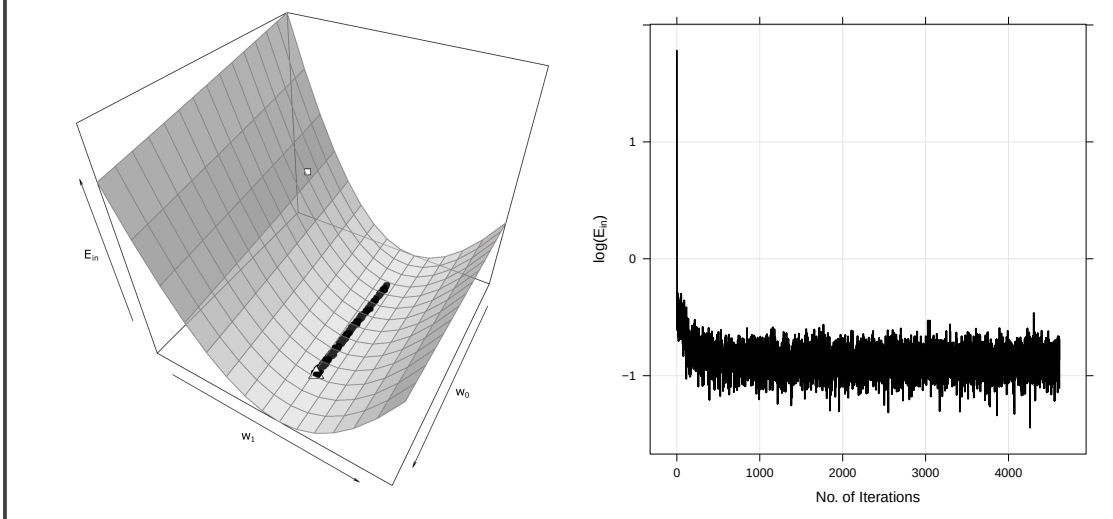
$$\mathbb{P}(x) \triangleq \frac{f(x)}{Z}, \quad \text{where } Z \triangleq \int f(x)\,\mathrm{d}\,x. \tag{3.2}$$

Using basic calculus, the mode[1] of the posterior distribution, say at $x = x_{\text{MAP}}$, is achieved by taking the derivative of the objective function with respect to the x-axis, such that the gradient of the function is 0 at $x = x_{\text{MAP}}$. That is,

$$\frac{\mathrm{d}\,f(x)}{\mathrm{d}\,x}\bigg|_{x=x_{\text{MAP}}} = 0. \tag{3.3}$$

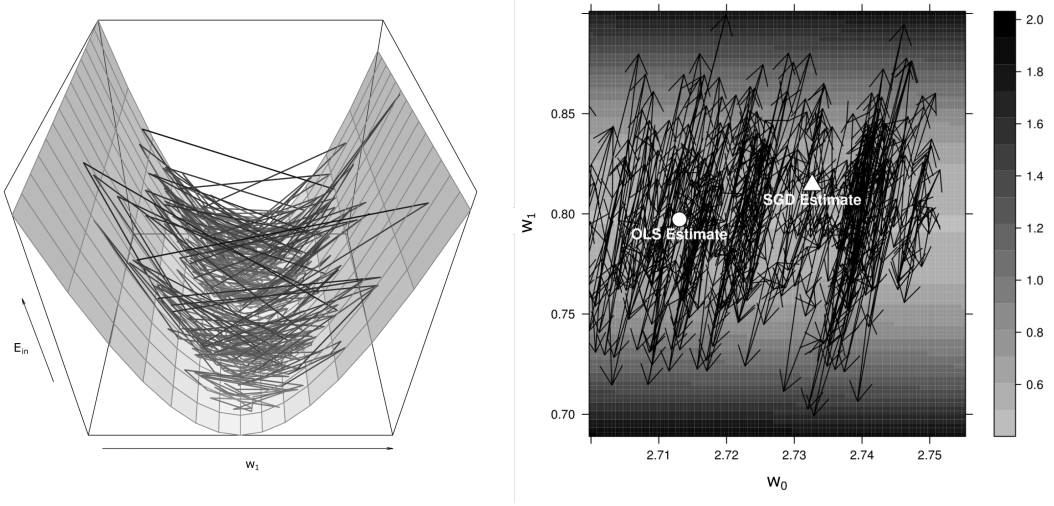
The Gaussian distribution has the property that its logarithm is a quadratic function of the variables (Bishop, 2006). Hence, the following is an approximation of the log of the objective function using second-order Taylor series expansion centered on

[1] obtained using maximum *a posteriori* (MAP)



(a) SGD on Loss Function Surface. (b) Loss Function under SGD.

Figure 3.3: *Stochastic Gradient Descent on SLR Loss Function.*



(a) SGD on Loss Function Surface. (b) SGD on Loss Function Contour.

Figure 3.4: *A Closer Look at SGD Gradient Vectors.*

the mode x_{MAP},

$$\log f(x) = \log f(x_{\text{MAP}}) + \frac{\mathrm{d}\log f(x)}{\mathrm{d}\,x}\bigg|_{x=x_{\text{MAP}}} (x - x_{\text{MAP}}) \tag{3.4}$$

$$+ \frac{\mathrm{d}^2 \log f(x)}{\mathrm{d}\,x^2}\bigg|_{x=x_{\text{MAP}}} \frac{(x - x_{\text{MAP}})^2}{2} + O(x) \tag{3.5}$$

$$\approx \log f(x_{\text{MAP}}) + \frac{\mathrm{d}^2 \log f(x)}{\mathrm{d}\,x^2}\bigg|_{x=x_{\text{MAP}}} \frac{(x - x_{\text{MAP}})^2}{2}. \tag{3.6}$$

Exponentiating both sides of the above equations becomes

$$f(x) \approx f(x_{\text{MAP}}) \exp\left[\frac{\mathrm{d}^2 \log f(x)}{\mathrm{d}\,x^2}\bigg|_{x=x_{\text{MAP}}} \frac{(x - x_{\text{MAP}})^2}{2}\right], \tag{3.7}$$

so that the normalized estimator $\mathbb{G}(x)$ is given by

$$\mathbb{G}(x) = \left(-\frac{1}{2\pi}\frac{\mathrm{d}^2 \log f(x)}{\mathrm{d}\,x^2}\bigg|_{x=x_{\text{MAP}}}\right)^{1/2} \exp\left[\frac{\mathrm{d}^2 \log f(x)}{\mathrm{d}\,x^2}\bigg|_{x=x_{\text{MAP}}} \frac{(x - x_{\text{MAP}})^2}{2}\right]. \tag{3.8}$$

Example 3.3.1. Suppose the posterior distribution is a chi-square of the form: $\mathbb{P}(x) \triangleq \frac{x^{k-1}\exp\left(-\frac{x^2}{2}\right)}{Z}$, with k degrees of freedom where $Z \triangleq 2^{\frac{k}{2}-1}\Gamma\left(\frac{k}{2}\right), \quad x > 0$. Using Laplace, the approximator to the posterior distribution is obtained as follows:

The log-likelihood of the density function is given by

$$\ell(x) \triangleq \log \mathbb{P}(x) = (k-1)\log x - \frac{x^2}{2} + C, \tag{3.9}$$

where C is the constant. The mode of this posterior is given by

$$\frac{\partial}{\partial x}\log \mathbb{P}(x) = \frac{k-1}{x_{\text{MAP}}} - x_{\text{MAP}} \overset{\text{set}}{=} 0 \tag{3.10}$$

$$x_{\text{MAP}} = \sqrt{k-1}. \tag{3.11}$$

The second partial derivative of the log-likelihood with respect to x evaluated at x_{MAP} is

$$-\frac{\partial^2}{\partial x^2}\log \mathbb{P}(x)\Big|_{x=x_{\text{MAP}}} = 1 - \frac{1-k}{x_{\text{MAP}}^2} = 2. \tag{3.12}$$

Thus the estimate of the posterior is given by a Gaussian distribution with mean $\mu = \sqrt{k-1}$ and variance $\sigma^2 = \frac{1}{2}$. Figure 3.5 visualizes the Gaussian distribution as an approximator to the posterior distribution. ⊸•

Now consider the case where $\mathbf{x} \in \mathbb{R}^d$, such that $\mathbb{P}(\mathbf{x}) \triangleq \frac{f(\mathbf{x})}{Z}$. Analogous to the univariate case, the Laplace's approximation for $\mathbb{P}(\mathbf{x})$ is obtained by aligning the mean of the multivariate Gaussian distribution to the mode of the posterior density, denoted by $\mathbf{x}_{\text{MAP}}$. As before, the log-likelihood of $f(\mathbf{z})$ is given by the following equation

$$\log f(\mathbf{x}) \approx \log f(\mathbf{x}_{\text{MAP}}) - \frac{1}{2}(\mathbf{x}-\mathbf{x}_{\text{MAP}})^{\mathrm{T}}\mathfrak{H}(\mathbf{x}-\mathbf{x}_{\text{MAP}}), \tag{3.13}$$

where $\mathfrak{H}$ is the Hessian matrix defined by

$$\mathfrak{H} \triangleq -\frac{\partial^2}{\partial \mathbf{x}^2}\log f(\mathbf{x})\Big|_{\mathbf{x}=\mathbf{x}_{\text{MAP}}}. \tag{3.14}$$

Exponentiating Equation (3.13) leads to the normalized approximator, $\mathbb{G}(\mathbf{x}) = \mathcal{N}_d(\mathbf{x}|\mathbf{x}_{\text{MAP}}, \mathfrak{H}^{-1})$.

3.4 Markov Chain Monte Carlo (MCMC)

Laplace has the advantage of being simple and easy to use. However, like any approximator, it has limitations especially on multimodal densities since it uses Gaussian as estimate to the posterior distribution. Most interesting high dimensional Bayesian models have multimodal *a posteriori*, which can't be captured through Laplace's method. To address this problem, sampling methods are instead used for approximating the *a posteriori*. These family of sampling methods are called Markov Chain Monte Carlos (MCMC) with *Metropolis-Hasting* (MH) and *Gibbs*

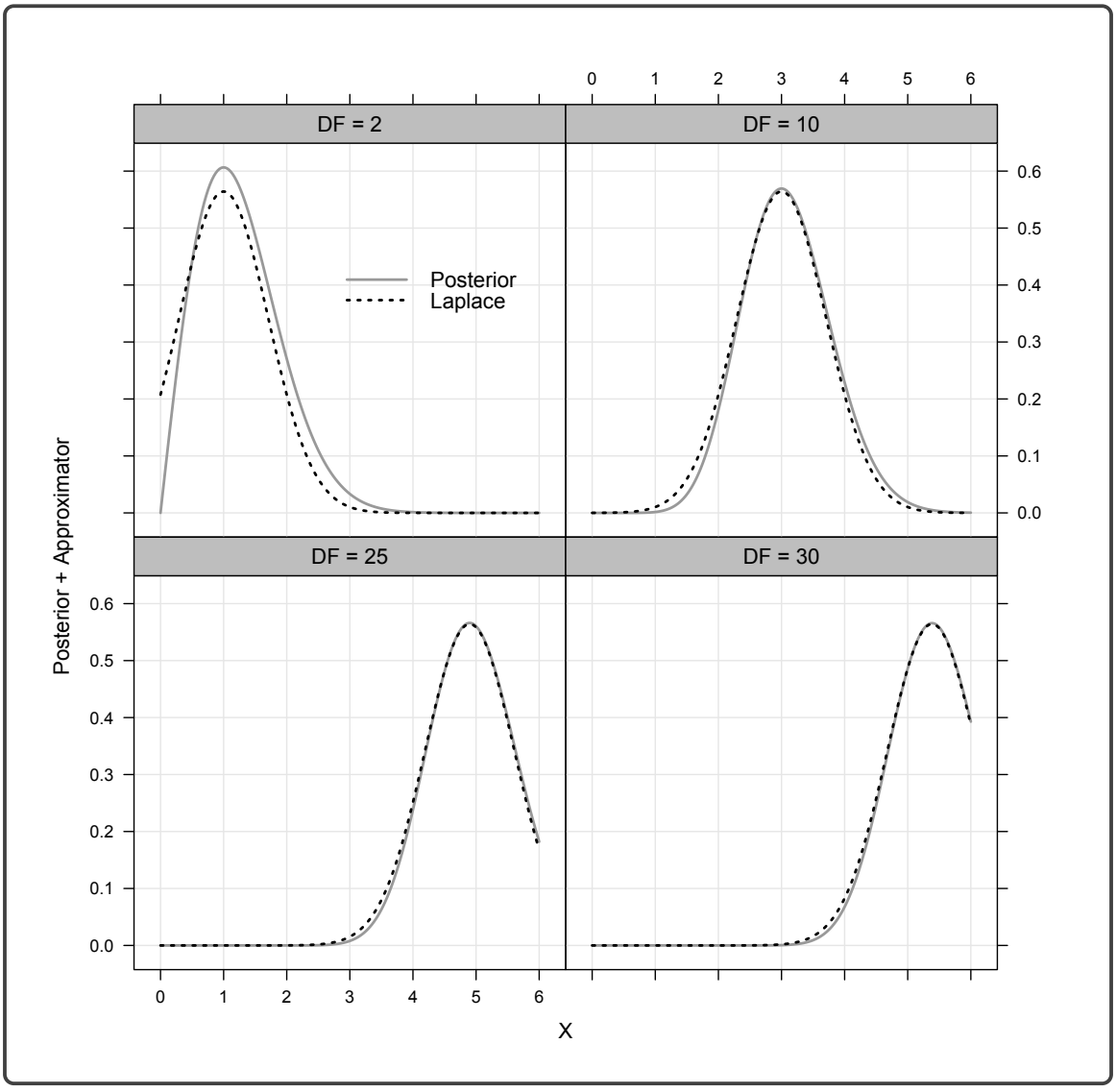


Figure 3.5: *Laplace's Approximation on χ^2-Distribution.*

Algorithm 3 *Metropolis-Hasting MCMC*

Initialize $\mathbf{w}_r \sim \mathbb{G}(\mathbf{w}), r = 0$
for $r \in \{1, \cdots, r_{\max}\}$ **do**
    Propose: $\mathbf{w}_{new} \sim \mathbb{G}(\mathbf{w}_{new}|\mathbf{w}_{r-1})$
    Acceptance: $\alpha(\mathbf{w}_{new}|\mathbf{w}_{r-1}) \triangleq \min\left\{1, \frac{\mathbb{P}(\mathbf{w}_{new}|\mathbf{w}_{r-1})\mathbb{G}(\mathbf{w}_{r-1}|\mathbf{w}_{new})}{\mathbb{P}(\mathbf{w}_{r-1}|\mathbf{w}_{new})\mathbb{G}(\mathbf{w}_{new}|\mathbf{w}_{r-1})}\right\}$
    Draw $x \sim \text{Unif}(0, 1)$
    if $x < \alpha(\mathbf{w}_{new}|\mathbf{w}_{r-1})$ **then**
        $\mathbf{w}_r \triangleq \mathbf{w}_{new}$
    else
        $\mathbf{w}_r \triangleq \mathbf{w}_{r-1}$
    end if
end for

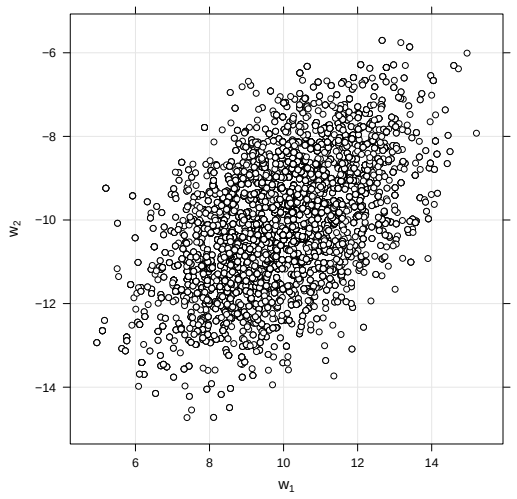
sampling as the popular MCMCs. Further, for sophisticated MCMCs, the algorithms available are not limited to *Hamiltonian Monte Carlo* (HMC) and *Stochastic Gradient HMC*.

3.4.1 Metropolis-Hasting

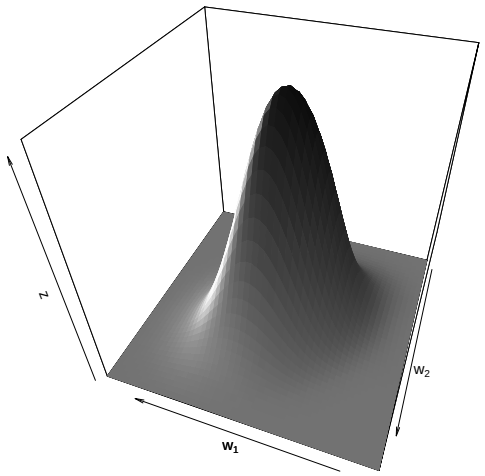
The idea of the MH algorithm is to randomly walk in the support of the target density such that the random step is governed by the proposal distribution $\mathbb{G}(\cdot)$. The assumption is that the posterior distribution has no closed-form solution, but the kernel, which is the unnormalized form of the target density is easy to evaluate. This is the advantage of the Metropolis-Hasting algorithm where the *a posteriori* is not necessarily be normalized — often the difficulty in simplifying the model evidence of the Bayes' rule. Let $\mathbb{P}(\cdot)$ be the *a posteriori*, then the Metropolis-Hasting algorithm is given in Algorithm 3.

Example 3.4.1. Consider the bivariate Gaussian distribution defined below:

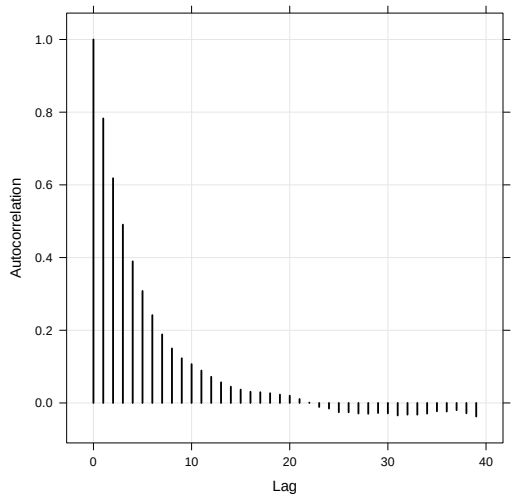
$$f(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) \triangleq \frac{1}{\sqrt{(2\pi)^d|\boldsymbol{\Sigma}|}} \exp\left[-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right]. \tag{3.15}$$



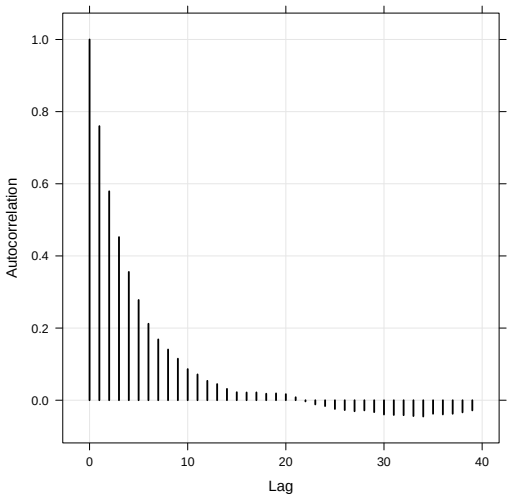
(a) Random Samples.



(b) Kernel Density Estimate.



(c) Autocorrelation of w_1.



(d) Autocorrelation of w_2.

Figure 3.6: *Metropolis-Hasting on Target Density after Burn-in.*

Suppose it has the following parameters:

$$\boldsymbol{\mu} \triangleq [10 \ -10]^{\mathrm{T}} \quad \text{and} \quad \boldsymbol{\Sigma} \triangleq \begin{bmatrix} 1.5^2 & \rho(1.5)(1.35) \\ \rho(1.5)(1.35) & 1.35^2 \end{bmatrix}$$

where $\rho \triangleq .5$; in order to draw samples from this model, let the proposal distribution defined to be the current step of the random walk plus an increment from a uniform distribution with parameters min = -5 and max = 5.

The random samples drawn by MH are not independent, this is due to

the design of the algorithm where the distribution of the candidate sample depends solely on the current sample[2]. To address this problem, diagnostics are applied using methods such as *thinning*, where every *i*th sample is taken and the rest are discarded; or using *burn-in* where first $n^\star$ samples are discarded. So that the plot of the random samples and its kernel density are depicted in Figure 3.6 using 10,000 iterations. Figures 3.6c and 3.6d depict the autocorrelations. ⊸•

3.4.2 Gibbs Sampling

MH is by far one of the easiest MCMC algorithm for drawing samples from *a posteriori* where direct sampling is not possible. It uses proposal distribution as drivers of the random walk in the support of the target density, which for high dimensional data, the choice of appropriate proposal function is sometimes difficult to specify. As an alternative, Gibbs sampler can be used for taking samples from the posterior distribution. The only requirement is that the joint distribution of the parameters (the *a posteriori*) can be decomposed into conditional distributions of each variable conditioned on other variables. Mathematically, suppose the multivariate distribution is given by $f(\mathbf{w}|\mathscr{D})$, where $\mathbf{w} \triangleq [w_1\ w_2 \cdots w_K]^{\mathrm{T}}$, then the Gibbs sampling algorithm is given in Algorithm 4.

Example 3.4.2. Using the same posterior distribution as in Example 3.4.1, the conditional distributions of the parameters conditioned on other parameters are also Gaussian with mean $\mu \triangleq \mu_1 + \left(\frac{\sigma_1}{\sigma_2}\right)\rho(x - \mu_2)$ and standard deviation $\sigma \triangleq \sqrt{(1-\rho^2)\sigma_1^2}$. Thus the Gibbs sampler for 10,000 iterations generates random samples shown in Figure 3.7. Analogous to MH, samples obtained using Gibbs are not independent but with lower autocorrelation compared to the former. As before,

[2]this is the property of the Markov Chain, hence the name MCMC.

the same diagnostics can be done. The autocorrelation is shown in Figures 3.7c and 3.7d, which suggest good mixing of the random samples. ⊸•

3.5 Bayesian Linear Regression

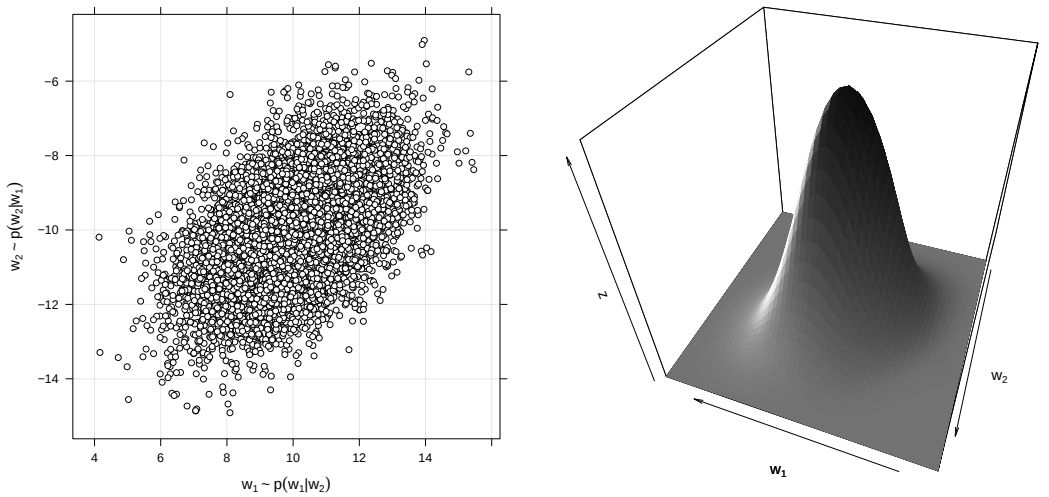
As an illustration of Bayesian inference to basic modeling, this section attempts to discuss the Bayesian approach to linear regression. Let $\mathscr{D} = \{(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_n, y_n)\}$ where $\mathbf{x}_i \in \mathbb{R}^d, y_i \in \mathbb{R}$ be the pairwised dataset. Suppose the response values, $y_1, \cdots, y_n$, are independent given the parameter $\mathbf{w}$, and is distributed as $y_i \sim \mathcal{N}(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i, \alpha^{-1})$, where α^{-1} is referred to as the *precision* parameter — useful for later derivation. In Bayesian perspective, the weights are assumed to be random and are governed by some *a priori* distribution. The choice of this distribution is subjective, but choosing arbitrary *a priori* can sometimes or often result to an intractable integration, especially for interesting models. For simplicity, a conjugate prior is used for the latent weights. Specifically, assume that $\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \beta^{-1}\mathbf{I})$ such that $\beta > 0$ is the hyperparameter supposed in this experiment

Algorithm 4 *Gibbs Sampling MCMC*

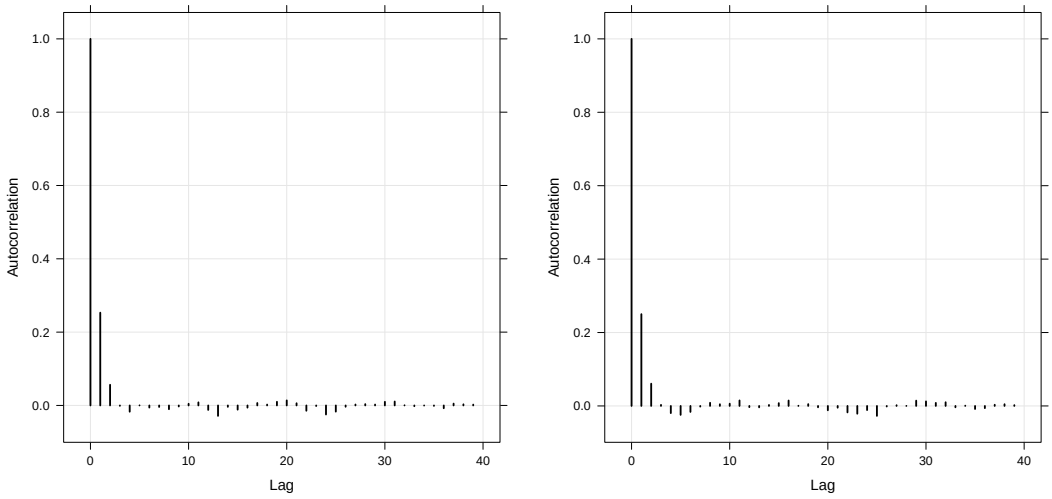
1: Initialize $\mathbf{w}_r, r = 0$
2: **for** $r \in \{1, \cdots, r_{\max}\}$ **do**
3: $\quad w_1^{(new)} \sim f(w_1 | w_2, \cdots, w_K)$
4: $\quad w_2^{(new)} \sim f(w_2 | w_1^{(new)}, \cdots, w_K)$
5: $\quad w_3^{(new)} \sim f(w_3 | w_1^{(new)}, w_2^{(new)}, \cdots, w_K)$
6: $\quad \vdots \qquad\qquad \vdots \qquad\qquad \vdots$
7: $\quad w_K^{(new)} \sim f(w_K | w_1^{(new)}, w_2^{(new)}, \cdots, w_{K-1}^{(new)})$
8: $\quad \mathbf{w}_r \triangleq [w_1^{(new)}, \cdots, w_K^{(new)}]^{\mathrm{T}}$
9: **end for**

as known value. The posterior distribution based on the Bayes' rule is given by

$$\mathbb{P}(\mathbf{w}|\mathbf{y}) = \frac{\mathbb{P}(\mathbf{w})\mathbb{P}(\mathbf{y}|\mathbf{w})}{\mathbb{P}(\mathbf{y})}, \tag{3.16}$$



(a) Random Samples. (b) Kernel Density Estimate.



(c) Autocorrelation of w_1. (d) Autocorrelation of w_2.

Figure 3.7: *Gibbs Sampling on Target Density after Burn-in.*

where $\mathbb{P}(\mathbf{w})$ is the *a priori* distribution of the parameter, $\mathbb{P}(\mathbf{y}|\mathbf{w})$ is the likelihood, and $\mathbb{P}(\mathbf{y})$ is the normalizing factor. The likelihood is given by

$$\mathbb{P}(\mathbf{y}|\mathbf{w}) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\alpha^{-1}}} \exp\left[-\frac{\alpha(y_i - \mathbf{w}^{\mathrm{T}}\mathbf{x}_i)^2}{2}\right] \tag{3.17}$$

$$= \left(\frac{\alpha}{2\pi}\right)^{n/2} \exp\left[-\sum_{i=1}^{n} \frac{\alpha(y_i - \mathbf{w}^{\mathrm{T}}\mathbf{x}_i)^2}{2}\right]. \tag{3.18}$$

In matrix form, this can be written as

$$\mathbb{P}(\mathbf{y}|\mathbf{w}) \propto \exp\left[-\frac{\alpha}{2}(\mathbf{y} - \mathfrak{A}\mathbf{w})^{\mathrm{T}}(\mathbf{y} - \mathfrak{A}\mathbf{w})\right] \tag{3.19}$$

where $\mathfrak{A} = \left[(\mathbf{x}_i^{\mathrm{T}})\right]$, i.e. $\mathfrak{A} \in (\mathbb{R}^n \times \mathbb{R}^d)$, this matrix is known as the *design matrix*. Given that $\mathbf{w}$ has the following prior distribution

$$\mathbb{P}(\mathbf{w}) = \frac{1}{\sqrt{(2\pi)^d|\beta^{-1}\mathbf{I}|}} \exp\left[-\frac{1}{2}\mathbf{w}^{\mathrm{T}}\beta\mathbf{I}\mathbf{w}\right], \tag{3.20}$$

implies that the posterior has the following form:

$$\mathbb{P}(\mathbf{w}|\mathbf{y}) \propto \exp\left[-\frac{\alpha}{2}(\mathbf{y} - \mathfrak{A}\mathbf{w})^{\mathrm{T}}(\mathbf{y} - \mathfrak{A}\mathbf{w})\right] \exp\left[-\frac{1}{2}\mathbf{w}^{\mathrm{T}}\beta\mathbf{I}\mathbf{w}\right] \tag{3.21}$$

$$= \exp\left\{-\frac{1}{2}\left[\alpha(\mathbf{y} - \mathfrak{A}\mathbf{w})^{\mathrm{T}}(\mathbf{y} - \mathfrak{A}\mathbf{w}) + \mathbf{w}^{\mathrm{T}}\beta\mathbf{I}\mathbf{w}\right]\right\}. \tag{3.22}$$

Expanding the terms in the exponent, becomes

$$\alpha\mathbf{y}^{\mathrm{T}}\mathbf{y} - 2\alpha\mathbf{w}^{\mathrm{T}}\mathfrak{A}^{\mathrm{T}}\mathbf{y} + \mathbf{w}^{\mathrm{T}}(\alpha\mathfrak{A}^{\mathrm{T}}\mathfrak{A} + \beta\mathbf{I})\mathbf{w}. \tag{3.23}$$

The next step is to complete the square of the above equation such that it resembles the inner terms of the exponential factor of the Gaussian distribution. The quadratic form of the exponential term of a $\mathcal{N}(\mathbf{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma}^{-1})$ is given by

$$(\mathbf{w} - \boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{w} - \boldsymbol{\mu}) = (\mathbf{w} - \boldsymbol{\mu})^{\mathrm{T}}(\boldsymbol{\Sigma}^{-1}\mathbf{w} - \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}) \tag{3.24}$$

$$= \mathbf{w}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\mathbf{w} - 2\mathbf{w}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu} + \boldsymbol{\mu}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}. \tag{3.25}$$

The terms in Equation (3.23) are matched up with that in (3.25), so that

$$\boldsymbol{\Sigma}^{-1} = \alpha \mathfrak{A}^{\mathrm{T}} \mathfrak{A} + \beta \mathbf{I} \tag{3.26}$$

and

$$\mathbf{w}^{\mathrm{T}} \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu} = \alpha \mathbf{w}^{\mathrm{T}} \mathfrak{A}^{\mathrm{T}} \mathbf{y} \tag{3.27}$$

$$\boldsymbol{\Sigma}^{-1} \boldsymbol{\mu} = \alpha \mathfrak{A}^{\mathrm{T}} \mathbf{y} \tag{3.28}$$

$$\boldsymbol{\mu} = \alpha \boldsymbol{\Sigma} \mathfrak{A}^{\mathrm{T}} \mathbf{y}. \tag{3.29}$$

Thus the *a posteriori* is a Gaussian distribution with location parameter in Equation (3.29) and scale parameter given by the inverse of Equation (3.26). The derivation above is not fully mathematical since the process of completing the square is similar to the proof of Theorem 5.1.1.

Example 3.5.1 (*Sequential Bayesian Learning by Simulation)***.** This example based from Bishop (2006) illustrates the sequential Bayesian learning to fitting a straight line to a simulated data. Consider the input variable x and a target variable y such that the true function is the simple linear regression with parameters $w_0 \triangleq -.3$ and $w_1 \triangleq -.5$. Suppose the random values of x are taken from a uniform distribution having domain $[-1, 1]$; then the target variable $y = h(x, \mathbf{w}) + \varepsilon = w_0 + w_1 x + \varepsilon$, where ε is a Gaussian noise having mean 0 and standard deviation 5. The goal is to recover the true value of w_0 and w_1 from the data. Using the *a priori* in Equation (3.20), the plot of this density is given in the first row, second column of Figure 3.8. This is the case in which no data point yet is observed. The white diamond point in the contour plot of the prior density is the true value of the parameters that needs to be estimated. The corresponding 20 samples of straight lines with weights sampled from the *a priori* are plotted in the far right hand side of the first row of

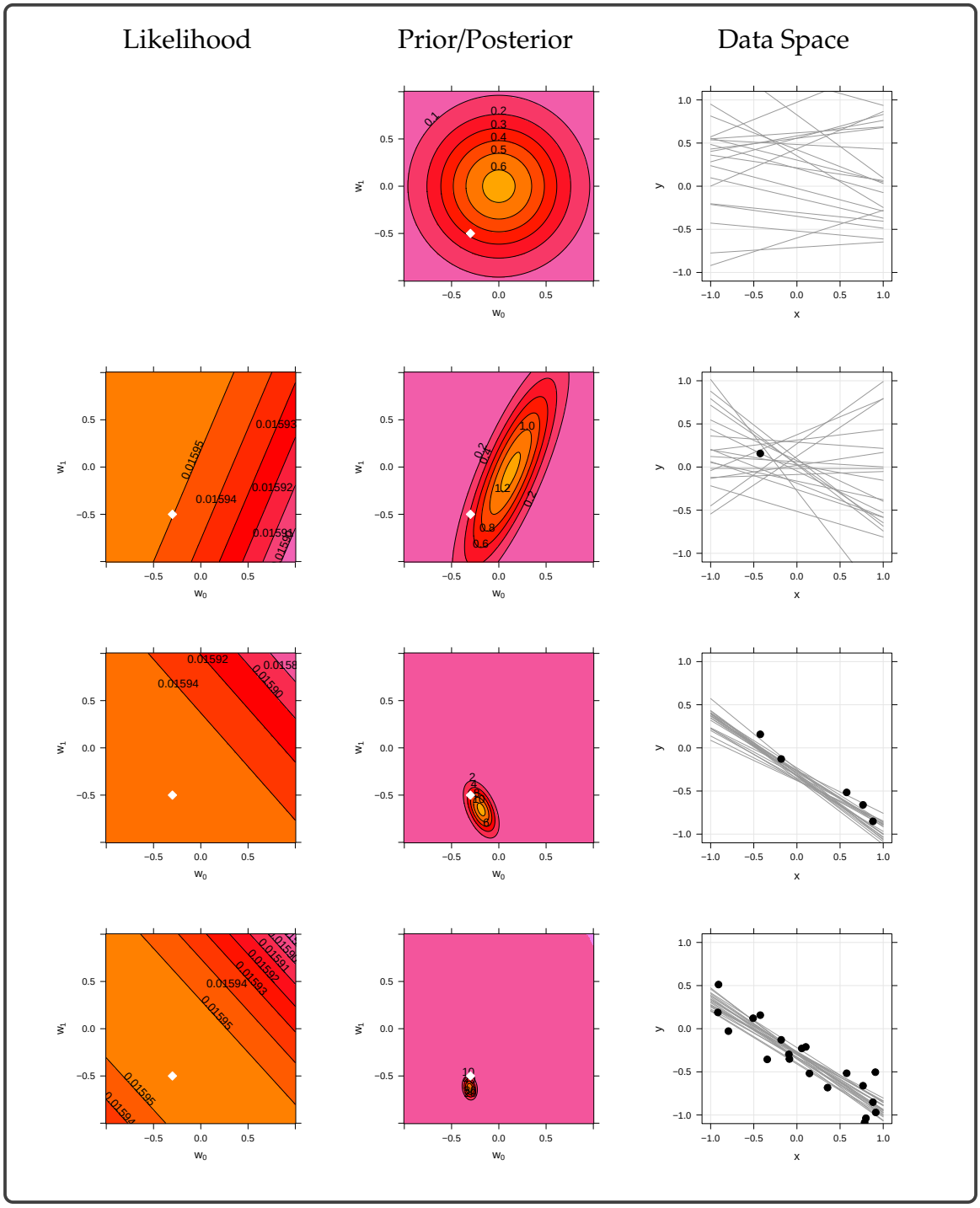


Figure 3.8: *Sequential Bayesian Learning of Fitting a Straight Line.*

the figure. The second row depicts the case where first observation is observed. The likelihood of this data point is shown in the left hand side of the row. Using this likelihood multiplied with the *a priori* in the previous row, and further multiply it with the normalizing constant, gives the *a posteriori* in the middle column of the second row. The corresponding fitted lines are placed in the right hand side of the row. Following the same procedure for the third row, with likelihood of the first five observations shown in the left hand side of the row, and *a priori* given by the *a posteriori* of the second row, returns a more concentrated posterior distribution of the weights with 20 sampled fitted lines in the third column of the third row. The process is repeated until the fourth row consisting of 20 observations. ⊸•

3.6 Markov Chain Diagnostics

The different techniques for *a posteriori* convergence analysis will be discussed in this section.

3.6.1 Convergence to the Stationary Distribution

In order to have a good estimate of the parameters, the chain must converge to the stationary distribution, in this case the target *a posteriori*. The samples taken from a single realization will likely not give assurance on the convergence unless, according to Robert and Casella (2010), there is some information about the support of the target density. Further, the authors emphasized that when looking at a single chain, this means acting as if the chain is already in its stationary regime at the start, meaning in practice that the initial sample at the start belongs to an area of likely (enough) values for the posterior. This assumption is harder to maintain in higher dimensions, however, when the support of $\mathbb{P}(\mathbf{w}|\mathbf{y})$ is generally unknown.

The recommendation of Robert and Casella (2010), which is implemented in most Bayesian software such as Stan[3], is to use several chains in parallel in order to compare the performance. For Stan, in particular, the default number of parallel chains is set to 4. To diagnose the stationarity, one can use cumulative plot to draw a smooth line on the trace of the samples. Or alternatively, do a nonparametric tests like Kolmogorov-Smirnov and Cramer-von Mises on two halves of the sample, that is, test whether the first half of the sample has the same distribution as the second half of the sample. The following are the tests on the stationarity of the chains.

Heidelberger-Welch

This non-parametric test makes use of Cramer-von Mises procedure by comparing two distributions through their approximate L_2 distance. If $\mathbb{F}$ is the reference cumulative distribution function (CDF) and $\mathbb{F}_m$ is the emprirical CDF, then the Cramer-von Mises statistic is

$$\mathbb{C} = \int (\mathbb{F}(w) - \mathbb{F}_t(w))^2 \, \mathrm{d}\, \mathbb{F}(w). \tag{3.30}$$

The null hypothesis is that the empirical CDF is equal to the reference CDF. The Heidelberger-Welch test can only be applied to single chain and is based on Brownian bridge theory and spectral analysis (*see* Cowles & Carlin, 1996).

3.6.2 Convergence of Average

Once the issue of stationarity is fixed, the next diagnostics is the convergence of the empirical average to the true value of the parameter, which is the expected value of the empirical average. That is,

$$\frac{1}{\tau} \sum_{t=1}^{\tau} h(w_i(t)) \to \mathbb{E}_{\mathbb{P}(w_i|\mathbf{y})}[h(w_i(t))], \tag{3.31}$$

[3]http://mc-stan.org/

where h is any arbitrary function. One way to test this is through the Gelman-Rubin method discussed below:

Gelman-Rubin

This test assesses the convergence based on multiple realization of the markov chain, by comparing the estimated between-variances and within-variances of the chains. In particular, the between-variances is the variance of the averages of the independent chains. While the within-variances is the average of the variances of the independent chains. Hence, if the markov chains converged, then the two statistics mentioned must approximately equate to each other. Otherwise, if there is large differences between these variances, then that is an indication of nonconvergence.

3.6.3 Approximating IID Samples

Using the fact that the MCMC approximates the *a posteriori*, then ideally, samples taken by MCMC should extend to the (approximate) production of IID samples from $\mathbb{P}(\mathbf{w}|\mathbf{y})$. As already illustrated on Examples 3.4.1 and 3.4.2, this requirement can be addressed by applying burn-in and thinning on the samples, that way the autocorrelation of the chain decreases over time.

3.7 Economic Indicators

As mentioned in the objective of this paper, the application of the proposed model (BADL-SGHMC) is on forecasting the Philippine's economic growth. Hence, the following are the descriptions of the indicators used in modeling:

1. **Peso/US Dollar Exchange Rate (ERATE)**

Monthly average of Philippine Peso-US Dollar exchange rate.

2. **Stock Price Index (SPI)**

   Philippine stock price index (SPI) serves as a measure of the changes in, and the movements of, the average prices of company shares of stock traded in the Philippine Stock Exchange (PSE).

3. **Gross International Reserves (GIR)**

   Foreign assets that are readily available to and controlled by the BSP for direct financing of payments imbalances and for managing the magnitude of such imbalances.

4. **Balance of Payments - Current Account (BOP)**

   Consists of the aggregate balance of goods, services, income and current transfers. This account measures the net transfer of real resources between the domestic economy and the rest of the world.

These indicators are freely available from the Bangko Sentral ng Pilipinas website, accessible through the following link:

`http://www.bsp.gov.ph/statistics/statistics_metadata.asp`

The time points of these quarterly economic series starts from 1999 first quarter to 2016 third quarter.

Chapter 4

METHODOLOGY

The theory of the algorithm and the model used in the main results are the main subject of this chapter. Specifically, Section 4.1 will discuss the Stochastic Gradient Hamiltonian Monte Carlo (SGHMC), proposed in this study as the Markov Chain Monte Carlo for estimating the Bayesian Autoregressive Distributed Lag (BADL) model — the objective model of this thesis. The presentation of the SGHMC proceeds with the introduction of the Hamiltonian dynamics, its extension to MCMC (known as Hamiltonian Monte Carlo), and the Langevin dynamics. Finally, the theory of the BADL is given in Section 4.2.

4.1 Stochastic Gradient Hamiltonian Monte Carlo

Classical or frequentist approach to modeling assumes the parameters $\mathscr{P}$ to be nonrandom and unknown, and inference is done using *maximum likelihood estimation* (MLE). In Bayesian statistics, these parameters are assumed to exhibit randomness that can be described by a probability distribution called the *posterior* distribution. This distribution is proportional to the *likelihood of the data* and the prior knowledge of the parameters governed by the *a priori*. From Casella and Berger (2002), this is a subjective distribution based on the experimenter's belief, and is formulated before the data are seen. A sample is then taken from a population indexed by $\mathscr{P}$ and the prior distribution is updated with this sample information. The updated prior becomes the posterior distribution. This updating is done with

the use of Bayes' Rule given below

$$\mathbb{P}(\mathscr{P}|\mathscr{D}) = \frac{\mathbb{P}(\mathscr{D}|\mathscr{P})\mathbb{P}(\mathscr{P})}{\mathbb{P}(\mathscr{D})}. \tag{4.1}$$

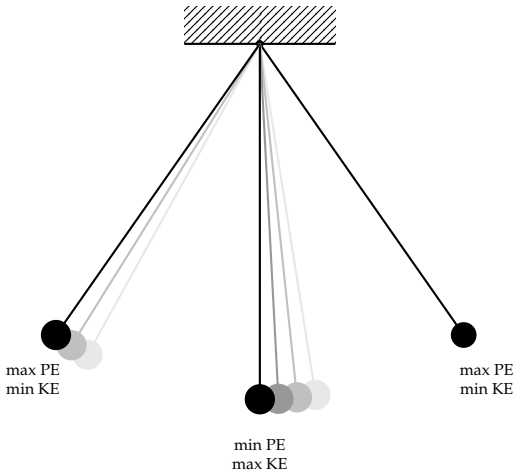
The only factor that is often difficult to derive, especially for interesting models, is the normalizing factor $\mathbb{P}(\mathscr{D}) = \int \mathbb{P}(\mathscr{D}, \mathscr{P})\, \mathrm{d}\, \mathscr{P}$. This probability is often intractable due to high dimensional integration. As a result, different approximation techniques were proposed for characterizing the *a posteriori*, $\mathbb{P}(\mathscr{P}|\mathscr{D})$.

4.1.1 Hamiltonian Dynamics

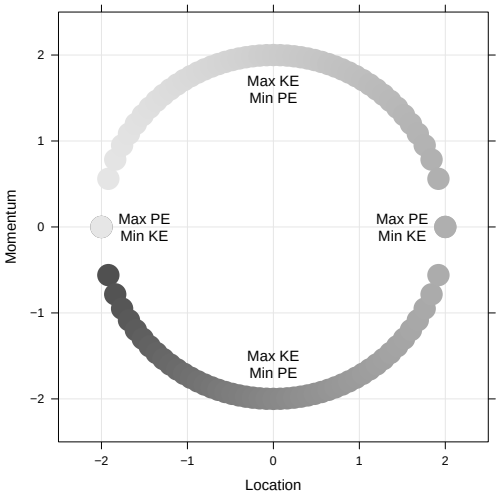
One limitation of the Metropolis-Hasting algorithm is that, the samples drawn are based on the rejection criterion where the decision to accept depends on the proposal distribution. The simplest proposal is the random walk, and it works by sampling the next step from the Uniform distribution. The problem with this nature of computation is that the distance traversed through the state space grows only as the square root of the number of steps (Bishop, 2006). However, if the step size is increased by expanding the min and max parameters in the case of Uniform proposal function, this will shorten the distance traversed but will lead to high rejection rate.

The *Hamiltonian Monte Carlo* originally known as *Hybrid Monte Carlo* in the paper by Duane et al. (1987), addresses the issue of the Metropolis-Hasting by considering auxiliary variable for describing the physical system in drawing samples from the target distribution. To understand the process, a brief review in Physics is provided for *Hamiltonian dynamics*. *Dynamic* in Physics deals with the study of the causes of motion, that is, the physical factors that can affect the object's motion in the *system*, which is the event or phenomenon being studied.

In particular, Hamiltonian dynamics describe the system using *location* parameter notated as **w** and *momentum* parameter **p**. As an example, consider a ball attached to a *frictionless* pendulum swinging on a vertical plane (*see* Figure 4.1a). For each location of the ball given by **w**, there is a corresponding *potential energy* (PE), denoted by $\mathbb{U}(\mathbf{w})$, and for each momentum **p**, there is an associated *kinetic energy* (KE) $\mathbb{K}(\mathbf{p})$. So that at the extreme trajectory of the pendulum, the PE is maximum and KE is minimum; and at the equilibrium point, the KE is maximum and PE is minimum. The system is a function of time, hence the Hamiltonian dynamics evolve in a continuous *space* called *phase space* (*see* Figure 4.1b). Another example which is being used in many literature (*see* Chen et al., 2014; Neal, 2011) is described as follows: imagine a hockey puck sliding over a frictionless ice surface of varying height. The PE term is based on the height of the surface at the current puck position, **w**, while the KE is based on the momentum of the puck, **p**, and its mass, **Σ**. If the surface is flat, the puck moves at a constant velocity. For positive slopes,



(a) Energies in Physical Pendulum.



(b) Energies in Phase Space.

Figure 4.1: *Conversion of Energies in Physical Pendulum and Phase Space.*

the KE decreases as the PE increases until the KE is at its minimum. The puck then slides back down the hill increasing its KE and decreasing its PE.

The total energy of the system is characterized by the Hamiltonian $\mathbb{H}(\mathbf{w}, \mathbf{p}) \triangleq \mathbb{U}(\mathbf{w}) + \mathbb{K}(\mathbf{p})$, and therefore describes the conversion of the two energies as the object moves throughout a system in time. So that the following are the Hamiltonian equations:

$$\begin{aligned} \frac{\mathrm{d}\,\mathbf{w}}{\mathrm{d}\,t} &= \frac{\partial \mathbb{H}(\mathbf{w}, \mathbf{p})}{\partial \mathbf{p}} = \frac{\mathrm{d}\,\mathbb{K}(\mathbf{p})}{\mathrm{d}\,\mathbf{p}} \\ \frac{\mathrm{d}\,\mathbf{p}}{\mathrm{d}\,t} &= -\frac{\partial \mathbb{H}(\mathbf{w}, \mathbf{p})}{\partial \mathbf{w}} = -\frac{\mathrm{d}\,\mathbb{U}(\mathbf{w})}{\mathrm{d}\,\mathbf{w}}. \end{aligned} \tag{4.2}$$

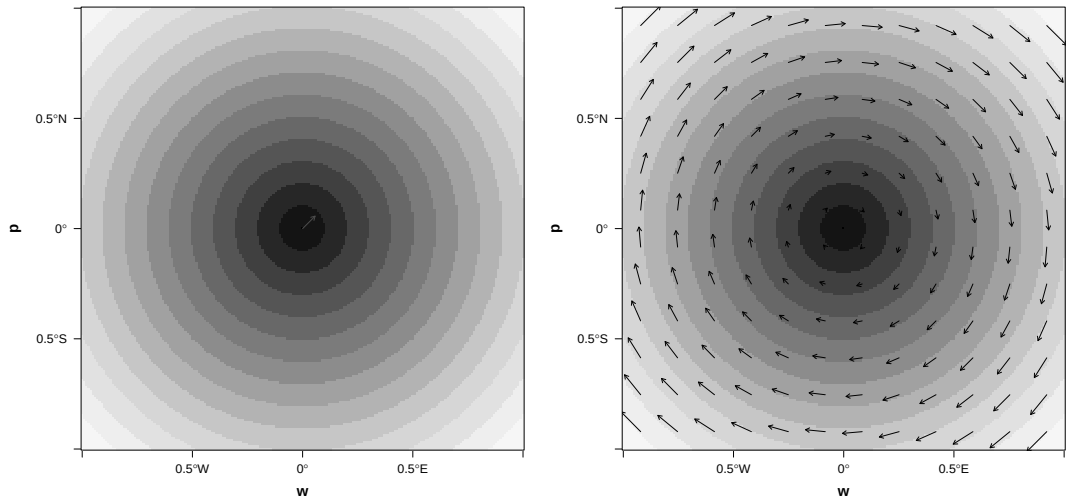
The succeeding discussion will prove the above equations by illustration. The Hamiltonian energy is a scalar function, so the levels of the contour in Figure 4.2 corresponds to the values of $\mathbb{H}(\mathbf{w}, \mathbf{p})$. The change in the total energy is given by the following equation:

$$\nabla \mathbb{H}(\mathbf{w}, \mathbf{p}) = \begin{bmatrix} \dfrac{\partial \mathbb{H}(\mathbf{w}, \mathbf{p})}{\partial \mathbf{w}} \\ \dfrac{\partial \mathbb{H}(\mathbf{w}, \mathbf{p})}{\partial \mathbf{p}} \end{bmatrix} \neq \begin{bmatrix} \dfrac{\mathrm{d}\,\mathbf{w}}{\mathrm{d}\,t} \\ \dfrac{\mathrm{d}\,\mathbf{p}}{\mathrm{d}\,t} \end{bmatrix}. \tag{4.3}$$

The reason for the above inequality is because the trajectories of the gradient of $\mathbb{H}$ is perpendicular to the contour lines. The final turn therefore, is to rotate the gradient of $\mathbb{H}$ 90° clockwise, so that the vector field of $\nabla \mathbb{H}(\mathbf{w}, \mathbf{p})$ is now tangent to the contour of $\mathbb{H}$, which is parallel to the field lines of the vector $\begin{bmatrix} \dfrac{\mathrm{d}\,\mathbf{w}}{\mathrm{d}\,t} & \dfrac{\mathrm{d}\,\mathbf{p}}{\mathrm{d}\,t} \end{bmatrix}^{\mathrm{T}}$, *see* Figure 4.2b. This in effect is equivalent to simply rotating the axes 90° clockwise, so that $\mathbf{p}$ becomes $\mathbf{w}$, and $\mathbf{w}$ becomes $-\mathbf{p}$, and their corresponding placeholder in the vector will also interchange. Thus Equation (4.3) becomes

$$\mathrm{rot}^{90^\circ} \nabla \mathbb{H}(\mathbf{w}, \mathbf{p}) = \begin{bmatrix} \dfrac{\partial \mathbb{H}(\mathbf{w}, \mathbf{p})}{\partial \mathbf{p}} \\ -\dfrac{\partial \mathbb{H}(\mathbf{w}, \mathbf{p})}{\partial \mathbf{w}} \end{bmatrix}. \tag{4.4}$$

Therefore Equation (4.4) is equivalent to Equation (4.2).



(a) Contour Lines of $\mathbb{H}(\mathbf{w}, \mathbf{p})$. (b) $\nabla[\mathbf{w}(t)\ \mathbf{p}(t)]$ over $\mathbb{H}(\mathbf{w}, \mathbf{p})$.

Figure 4.2: *Contour Lines of* $\mathbb{H}(\mathbf{w}, \mathbf{p})$ *and Gradient of* $[\mathbf{w}(t)\ \mathbf{p}(t)]$.

There are three properties that makes Hamiltonian dynamics good for sampling. The first one is the *conservation* of the energy, that is, a change in time for both position and momentum won't affect the Hamiltonian total energy. This can be seen from the following equation

$$\frac{\mathrm{d}\,\mathbb{H}}{\mathrm{d}\,t} = \sum_i \left[\frac{\partial \mathbb{H}}{\partial \mathbf{w}} \frac{\mathrm{d}\,\mathbf{w}}{\mathrm{d}\,t} + \frac{\partial \mathbb{H}}{\partial \mathbf{p}} \frac{\mathrm{d}\,\mathbf{p}}{\mathrm{d}\,t} \right] \tag{4.5}$$

$$= \sum_i \left[\frac{\partial \mathbb{H}}{\partial \mathbf{w}} \frac{\partial \mathbb{H}}{\partial \mathbf{p}} - \frac{\partial \mathbb{H}}{\partial \mathbf{p}} \frac{\partial \mathbb{H}}{\partial \mathbf{w}} \right] = 0. \tag{4.6}$$

The second property follows from the *Liouville's theorem*, which claims that the system *preserves* the volume of the phase space. The last property is *reversibility*, all details can be found in Neal (1996); Bishop (2006); Neal (2011).

4.1.2 Leapfrog Method

Since the phase space changes over time which is a continuous variable, then in order to simulate the Hamiltonian dynamics under numerical computations, the time has to be discretized. There are several ways to do this, one such solution

Algorithm 5 *Hamiltonian MCMC*

Initialize Leapfrog parameters: γ and τ;
Set initial location $\mathbf{w}^{(r=0)}(t=0)$;
for $r \in \{0, \cdots, r_{\max}\}$ **do**
    Draw initial momentum, $\mathbf{p}^{(r)}(t=0) \sim \frac{\exp\left[-\mathbb{K}(\mathbf{p})\right]}{Z}$;
    Compute $\mathbb{H}(\mathbf{w}^{(r)}(0), \mathbf{p}^{(r)}(0)) \triangleq \mathbb{U}(\mathbf{w}^{(r)}(0)) + \mathbb{K}(\mathbf{p}^{(r)}(0))$;
    Simulate Hamiltonian dynamics using Leapfrog:
    for $t \in \{0, \cdots, \tau\}$ **do**

$$\mathbf{p}^{(r)}(t+\gamma/2) \triangleq \mathbf{p}^{(r)}(t) - (\gamma/2)\frac{\partial \mathbb{U}(\mathbf{w}^{(r)}(t))}{\partial \mathbf{w}^{(r)}(t)} \tag{4.7}$$

$$\mathbf{w}^{(r)}(t+\gamma) \triangleq \mathbf{w}^{(r)}(t) + \gamma\frac{\partial \mathbb{K}(\mathbf{p}^{(r)}(t+1))}{\partial \mathbf{p}^{(r)}(t+1)}, \tag{4.8}$$

$$\mathbf{p}^{(r)}(t+\gamma) \triangleq \mathbf{p}^{(r)}(t+\gamma/2) - (\gamma/2)\frac{\partial \mathbb{U}(\mathbf{w}^{(r)}(t+\gamma))}{\partial \mathbf{w}^{(r)}(t+\gamma)} \tag{4.9}$$

    end for
    if $\Delta\mathbb{H} < 0$ **then**
        $\mathbf{w}^{(r+1)}(0) \triangleq \mathbf{w}^{(r)}(\tau+\gamma)$
    else
        if $a < \exp(\Delta\mathbb{H}), a \sim \text{Unif}(0,1)$ **then**
            $\mathbf{w}^{(r+1)}(0) \triangleq \mathbf{w}^{(r)}(\tau+\gamma)$
        else
            $\mathbf{w}^{(r+1)}(0) \triangleq \mathbf{w}^{(r)}(0)$
        end if
    end if
end for

is to consider the *leapfrog method*, which works as follows:

$$\mathbf{p}(t+\gamma/2) = \mathbf{p}(t) - (\gamma/2)\frac{\partial \mathbb{U}(\mathbf{w}(t))}{\partial \mathbf{w}(t)} \tag{4.10}$$

$$\mathbf{w}(t+\gamma) = \mathbf{w}(t) + \gamma\frac{\partial \mathbb{K}(\mathbf{p}(t))}{\partial \mathbf{p}(t)}, \tag{4.11}$$

$$\mathbf{p}(t+\gamma) = \mathbf{p}(t+\gamma/2) - (\gamma/2)\frac{\partial \mathbb{U}(\mathbf{w}(t+\gamma))}{\partial \mathbf{w}(t)} \tag{4.12}$$

where $\gamma > 0$.

Hamiltonian/Hybrid Monte Carlo

The Hamiltonian dynamics is related to MCMC using the fact that the total energy is related to the probability distribution of the parameter of interest using the concept of *canonical distribution* from the Statistical Mechanics. That is,

$$\mathbb{P}(\mathscr{P}) = \frac{1}{Z}\exp\left[-E(\mathscr{P})\right], \tag{4.13}$$

where E is the total energy. Therefore, E in this case, is $\mathbb{H}(\mathbf{w}, \mathbf{p})$. Further, the three properties of the Hamiltonian dynamics mentioned above will make the canonical distribution *invariant*. So that the equation becomes

$$\mathbb{P}(\mathbf{w}, \mathbf{p}) \propto \exp\left[-\mathbb{H}(\mathbf{w}, \mathbf{p})\right] \tag{4.14}$$

$$= \exp\left[-\mathbb{U}(\mathbf{w}) - \mathbb{K}(\mathbf{p})\right] \tag{4.15}$$

$$= \exp\left[-\mathbb{U}(\mathbf{w})\right]\exp\left[-\mathbb{K}(\mathbf{p})\right] \tag{4.16}$$

$$\propto \mathbb{P}(\mathbf{w})\mathbb{P}(\mathbf{p}). \tag{4.17}$$

Therefore the joint canonical distribution of the location parameter $\mathbf{w}$ and the momentum parameter $\mathbf{p}$ factors into the products of its marginal density, implying independence. In this case, the momentum parameter serves as the *auxiliary variable* for the parameter of interest, $\mathbf{w}$, and because $\mathbf{p}$ is used

for simulating Hamiltonian dynamics in phase space, making it random plus the gradient of its distribution leads to the exploration of high probability region, and thus it is also considered as the proposal distribution with certainty of acceptance on the proposed samples. However due to the discretization of the phase space, the true samples proposed from the joint distribution of the location and momentum parameters are accepted using some adjustment. This adjustment is the introduction of the Metropolis-Hasting criterion as the decision rule for accepting the sample. Finally, the kinetic energy is often assumed to be standard Gaussian distributed, and thus

$$\mathbb{K}(\mathbf{p}, \mu \triangleq \mathbf{0}, \mathbf{\Sigma} \triangleq \mathbf{I}) = \frac{(\mathbf{p}-\mu)^{\mathrm{T}}\mathbf{\Sigma}^{-1}(\mathbf{p}-\mu)}{2} = \frac{\mathbf{p}^{\mathrm{T}}\mathbf{p}}{2}. \tag{4.18}$$

where μ and $\mathbf{\Sigma}$ are the mean vector and the variance-covariance matrix, respectively. The $\mathbf{\Sigma}$ of the kinetic energy can be assigned to any positive definite matrix if additional information about the target density is available.

Hence to summarize the parameter of interest $\mathbf{w}$, the following is the target distribution,

$$\mathbb{U}(\mathbf{w}) = -\log \mathbb{P}(\mathbf{w}|\mathbf{y}) \tag{4.19}$$

$$= -\log[\mathbb{P}(\mathbf{w})\mathcal{L}(\mathbf{w}|\mathbf{y})] - C, \tag{4.20}$$

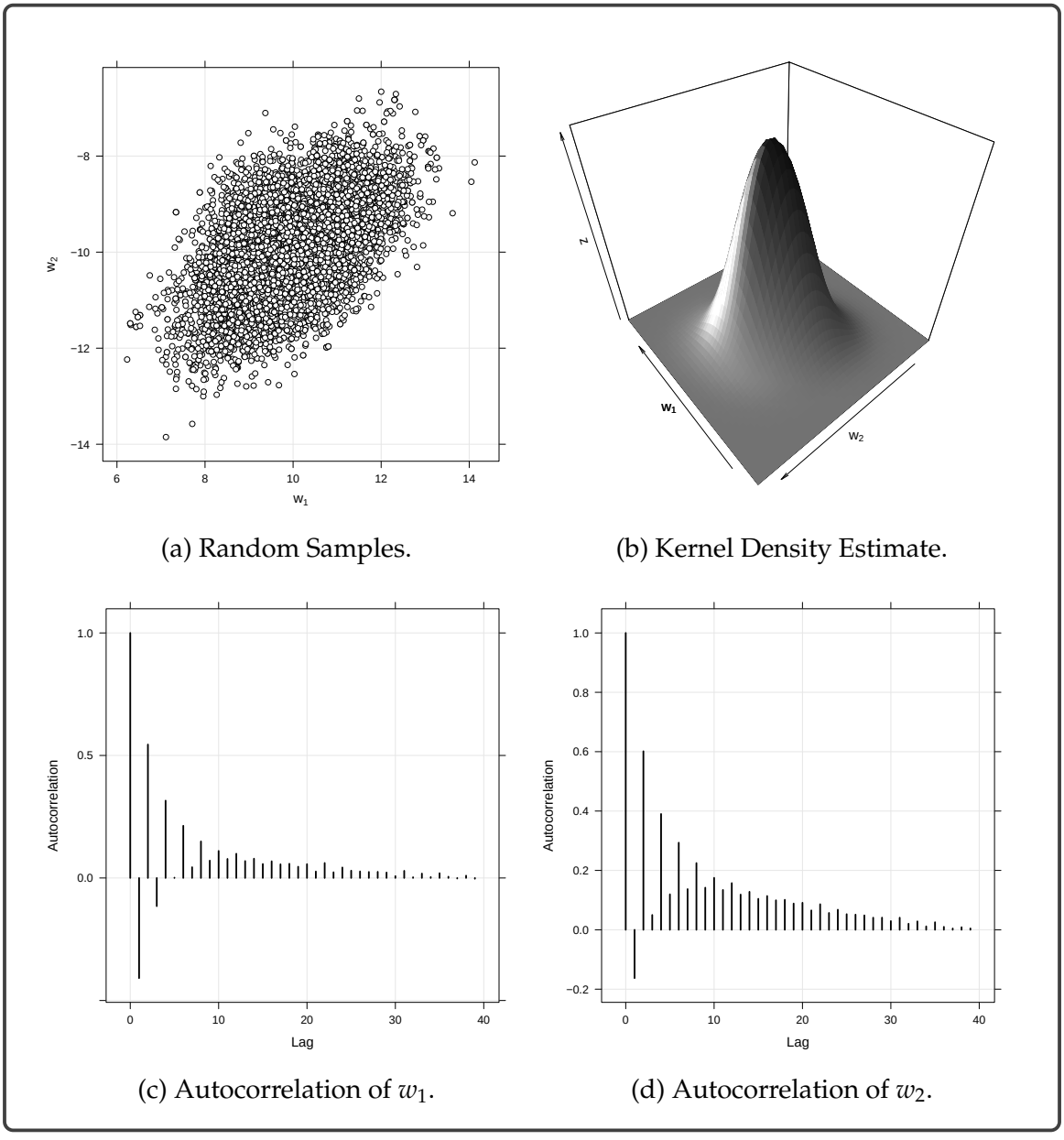
where $C \triangleq \log \mathbb{P}(\mathbf{y})$, which will be cancelled out at line 9 of Algorithm 5.

Example 4.1.1. The samples drawn from the bivariate Gaussian distribution defined in Example 3.4.1 using Hamiltonian MCMC are depicted in Figure 4.3. The corresponding autocorrelations for both parameters using "burn-in" method are depicted in Figures 4.3c and 4.3d. ⊸

4.1.3 Langevin Dynamics

The Stochastic Gradient HMC works by considering Langevin dynamics on its momentum. The dynamics extend the idea of the Newton's second law of motion, which orignially proceeds as follows: let **f** be the force, **p** be the momentum, m be the mass, **v** be the veclocity, and **a** be the acceleration, then

$$\mathbf{f} = \frac{\mathrm{d}\,\mathbf{p}}{\mathrm{d}\,t} = \frac{\mathrm{d}(m\mathbf{v})}{\mathrm{d}\,t} = m\frac{\mathrm{d}\,\mathbf{v}}{\mathrm{d}\,t} = m\mathbf{a}. \tag{4.21}$$



(a) Random Samples.

(b) Kernel Density Estimate.

(c) Autocorrelation of w_1.

(d) Autocorrelation of w_2.

Figure 4.3: *MCMC Hamiltonian on Target Density.*

The idea of Langevin dynamics is to take into account or at least approximate the effect of neglected degrees of freedom, and this is achieved by adding two force terms: one represents the frictional force, $\eta\mathbf{v}^{\blacklozenge}$; and the other represents the random force, $\mathbf{e}$. So that the Langevin equation is given below:

$$\frac{\mathrm{d}\,\mathbf{p}}{\mathrm{d}\,t} - \eta\mathbf{v}^{\blacklozenge} + \mathbf{e} = m\mathbf{a}, \tag{4.22}$$

where the random force is assumed to have zero mean and is uncorrelated, i.e. $\mathbf{e} \sim \mathcal{N}(\mathbf{0}, \xi\mathbf{I})$.

4.1.4 Stochastic Gradient HMC

The discussion in this section is based on Chen et al. (2014). The idea is to apply the common practice in machine learning for speeding up the numerical computations in optimization problems. For very large dataset, especially in the era of Big data, the use of batch gradient descent can be very slow even for 100 observations only, the convergence can take time as illustrated in Example 3.2.1. The solution as presented in § 3.2, is to consider stochastic gradient or minibatch stochastic gradient approach. That instead of taking single observation for updating the differential equation in the case of pure stochastic gradient approach or using all observations in the case of batch gradient, why not use only samples of the full dataset? The computational advantage of minibatch gradient over the two approaches is vectorization. Vectorization is computationally fast if it is possible for the algorithm. Let $\tilde{\mathscr{D}}$ be the minibatch or sample of the full dataset $\mathscr{D}$, then $\tilde{\mathscr{D}} \subseteq \mathscr{D}$, implies that

$$\nabla\tilde{\mathbb{U}}(\mathbf{w}) = -\frac{|\mathscr{D}|}{|\tilde{\mathscr{D}}|}\sum_{\mathbf{x}\in\tilde{\mathscr{D}}} \nabla \log \mathbb{P}(\mathbf{x}|\mathbf{w}) - \nabla \log \mathbb{P}(\mathbf{w}). \tag{4.23}$$

The minibatch above is uniformly sampled from $\mathscr{D}$, so that the weight $\frac{|\mathscr{D}|}{|\tilde{\mathscr{D}}|}$ makes $\nabla\tilde{\mathbb{U}}(\mathbf{w})$ an estimate to $\nabla\mathbb{U}(\mathbf{w})$. The error of this estimate is known as the *stochastic gradient noise*, and is given by

$$\nabla\tilde{\mathbb{U}}(\mathbf{w}) - \nabla\mathbb{U}(\mathbf{w}) = \xi. \tag{4.24}$$

Implying $\mathbb{E}[\nabla\tilde{\mathbb{U}}(\mathbf{w})] = \nabla\mathbb{U}(\mathbf{w})$, and hence

$$\mathbb{E}[\nabla\tilde{\mathbb{U}}(\mathbf{w}) - \nabla\mathbb{U}(\mathbf{w})] = \mathbb{E}[\xi] = \mathbf{0}. \tag{4.25}$$

Let $\mathbb{V}\mathrm{ar}[\xi] = \mathfrak{A}(\mathbf{w})$ be the variance-covariance matrix of the stochastic gradient noise, then by central limit theorem (CLT), $\xi \sim \mathcal{N}(\mathbf{0}, \mathfrak{A}(\mathbf{w}))$. Therefore $\tilde{\mathbb{U}}(\mathbf{w})$ is approximated as follows:

$$\nabla\tilde{\mathbb{U}}(\mathbf{w}) \approx \nabla\mathbb{U}(\mathbf{w}) + \xi, \quad \xi \sim \mathcal{N}(\mathbf{0}, \mathfrak{A}(\mathbf{w})). \tag{4.26}$$

The equality in Equation (4.24) is replaced with approximation in Equation (4.26) since ξ is now taken as a sample from a defined distribution, the Gaussian. In effect, the momentum update of the HMC algorithm now has a noise term added to it. That is, $\Delta\mathbf{p} = -\gamma\nabla\tilde{\mathbb{U}}(\mathbf{w})$, so that $\mathbb{V}\mathrm{ar}[-\gamma\xi] = \gamma^2\mathfrak{A}(\mathbf{w})$ or $2\mathfrak{B}(\mathbf{w})$ where $\mathfrak{B}(\mathbf{w}) = \frac{1}{2}\gamma\mathfrak{A}(\mathbf{w})$ is the diffusion matrix. γ^2 is redefined as γ, since it is a constant. Thus if the batch size, $|\tilde{\mathscr{D}}|$, becomes small, then the variability of ξ becomes large. The resulting discrete time system can be viewed as a γ-discretization of the following continuous stochastic differential equation:

$$\frac{\mathrm{d}\,\mathbf{w}}{\mathrm{d}\,t} = \boldsymbol{\Sigma}^{-1}\mathbf{p} \quad \text{and} \quad \frac{\mathrm{d}\,\mathbf{p}}{\mathrm{d}\,t} \approx -\nabla\mathbb{U}(\mathbf{w}) + \xi^{\blacklozenge}, \tag{4.27}$$

where $\xi^{\blacklozenge} \sim \mathcal{N}(\mathbf{0}, 2\mathfrak{B})$. For brevity, $\mathfrak{B}(\mathbf{w})$ is now notated as $\mathfrak{B}$ for the rest of the chapter. To gain some intuition, consider again the hockey puck analogy of § 4.1.1. The dynamics is still the same but this time there is a presence of some

random wind blowing. The wind may blow the puck further away than expected. This contribution of the randomness is one of the degrees of freedom mentioned in Langevin dynamics (*see* § 4.1.3), which affects the preservation of the entropy under the Hamiltonian dynamics, and is shown in one of the results of Chen et al. (2014), refer to the article for the theoretical results of the entropy of the target density.

To address the problem presented above, a frictional force is added to the equation. This introduces a correction step even before considering errors introduced by the discretization of the dynamical system. So that Equation (4.27) becomes

$$\frac{\mathrm{d}\,\mathbf{w}}{\mathrm{d}\,t} = \boldsymbol{\Sigma}^{-1}\mathbf{p} \quad \text{and} \quad \frac{\mathrm{d}\,\mathbf{p}}{\mathrm{d}\,t} = -\nabla\mathbb{U}(\mathbf{w}) - \mathfrak{B}\boldsymbol{\Sigma}^{-1}\mathbf{p} + \xi^{\blacklozenge}, \tag{4.28}$$

where $\xi^{\blacklozenge} \sim \mathcal{N}(\mathbf{0}, 2\mathfrak{B})$. According to Chen et al. (2014), the posterior distribution described by the dynamics in the above equation is the unique stationary distribution. Equation (4.28) is also referred to us the *second-order Langevin dynamics*, *see* § 4.1.3.

SGHMC in Practice

The parameter $\mathfrak{B}$ up to this point is assumed to be known. However, this is not the case in practice. A remedy is to consider an estimate of $\mathfrak{B}$ instead, denoted as $\hat{\mathfrak{B}}$, and define a user-specified frictional term $\mathfrak{C} \geq \hat{\mathfrak{B}}$. That is $\mathfrak{C} - \hat{\mathfrak{B}} \geq 0$ suggests that the matrix $\mathfrak{C} - \hat{\mathfrak{B}}$ is positive-semidefinite. So that the dynamics becomes

$$\frac{\mathrm{d}\,\mathbf{w}}{\mathrm{d}\,t} = \boldsymbol{\Sigma}^{-1}\mathbf{p} \quad \text{and} \quad \frac{\mathrm{d}\,\mathbf{p}}{\mathrm{d}\,t} = -\nabla\tilde{\mathbb{U}}(\mathbf{w}) - \mathfrak{C}\boldsymbol{\Sigma}^{-1}\mathbf{p} + \xi^{\bigstar}, \tag{4.29}$$

where $\xi^{\bigstar} \sim \mathcal{N}(\mathbf{0}, 2(\mathfrak{C}-\hat{\mathfrak{B}}))$. The result is known as the *Stochastic Gradient Hamiltonian Monte Carlo* (SGHMC) algorithm defined in Algorithm 6.

Algorithm 6 *Stochastic Hamiltonian MCMC*

Initialize Leapfrog parameters: γ and τ;
Initialize estimate for $\hat{\mathfrak{B}}(\mathbf{w}) = \frac{\gamma}{2}\hat{\mathfrak{A}}(\mathbf{w})$, and specify the matrix $\mathfrak{C}$;
Set initial location $\mathbf{w}^{(r=0)}(t=0)$;
for $r \in \{0, \cdots, r_{\max}\}$ **do**
    Draw initial momentum, $\mathbf{p}^{(r)}(t=0) \sim \frac{\exp\left[-\mathbb{K}(\mathbf{p})\right]}{Z}$;
    Simulate Hamiltonian dynamics using Leapfrog:
    for $t \in \{0, \cdots, \tau\}$ **do**

$$\Delta\mathbf{w}^{(r)}(t+\gamma) \triangleq \gamma \nabla_{\mathbf{p}^{(r)}(t+1)} \mathbb{K}(\mathbf{p}^{(r)}(t+1)), \tag{4.30}$$

$$\Delta\mathbf{p}^{(r)}(t+\gamma) \triangleq -\gamma \nabla_{\mathbf{w}^{(r)}(t)} \tilde{\mathbb{U}}(\mathbf{w}^{(r)}(t)) - \mathfrak{C}(\mathbf{w}^{(r)}(t))\boldsymbol{\Sigma}^{-1}\mathbf{p} + \xi^{\star} \tag{4.31}$$

$$\text{where } \xi^{\star} \sim \mathcal{N}(\mathbf{0}, 2\gamma(\mathfrak{C} - \mathfrak{B})) \tag{4.32}$$

    end for
    $\mathbf{w}^{(r+1)} \triangleq \mathbf{w}^{(r)}$
end for

As mentioned above, $\hat{\mathfrak{B}}$ is an estimate for $\mathfrak{B}$ since the latter is not known in practice. The recommendation of Chen et al. (2014), is to set $\hat{\mathfrak{B}}$ to $\mathbf{0}_{n\times n}$ as its simplest estimate, and in effect, Equation (4.29) will still preserve the entropy since the expression is now governed by a controllable matrix $\mathfrak{C}$. Further, using the fact that $\mathfrak{B} = \frac{1}{2}\gamma\mathfrak{A}$ then $\hat{\mathfrak{B}} = \frac{1}{2}\gamma\hat{\mathfrak{A}}$. Finally, as $\gamma \to 0$, $\mathfrak{B} = \mathbf{0}_{n\times n}$. The other option is to estimate $\mathfrak{A}$ using *empirical* Fisher information as in Ahn et al. (2012).

4.2 Autoregressive Distributed Lag Model

The popular time series models often used in econometrics is the Autoregressive (AR) expectation function, where the response variable $y(t)$ is dependent on its lagged values. Further, if the *distributed lag* term, which is composed of other explanatory variables and including its lags, are to be added to the model, then this leads to *Autoregressive Distributed Lag* (ADL). In

this paper, the ADL as emphasized in the first chapter will be the objective model. The Stochastic Gradient Hamiltonian Monte Carlo (SGHMC) discussed in the preceding section will be integrated into the estimation of the parameters of the ADL which will be discussed in the next section.

4.2.1 The Model: ADL(p, q)

The simplest ADL model is of order $p = 1$ and $q = 0$, denoted by ADL$(1, 0)$ and is given by

$$y(t) = w_0 + w_1 y(t-1) + w_2 x(t) + \varepsilon(t), \quad \varepsilon \sim \mathcal{N}(0, \sigma). \tag{4.33}$$

If there are m other explanatory variables, then the updated model is

$$y(t) = w_0 + w_1 y(t-1) + \sum_{i=1}^{m} w_{2+i} x_i(t) + \varepsilon(t), \quad \varepsilon \sim \mathcal{N}(0, \sigma). \tag{4.34}$$

Thus for ADL$(1, 1)$, the following is the form

$$y(t) = w_0 + w_1 y(t-1) + \sum_{i=1}^{m} \sum_{l=0}^{1} w_{(2+l\cdot m)+i} x_i(t-l) + \varepsilon(t), \quad \varepsilon \sim \mathcal{N}(0, \sigma). \tag{4.35}$$

For general ADL(p, q), the model can be written as

$$y(t) = w_0 + \sum_{k=1}^{p} w_k y(t-k) + \sum_{i=1}^{m} \sum_{l=0}^{q} w_{\kappa(p,l,m,i)} x_i(t-l) + \varepsilon(t), \quad \varepsilon \sim \mathcal{N}(0, \sigma). \tag{4.36}$$

where $\kappa(p, l, m, i) \triangleq [(p + 1) + l \cdot m] + i$. The first summation, $\sum_{k=1}^{p} w_k y(t-k)$, denotes the *autoregressive* term, while the second double-summation, $\sum_{i=1}^{m} \sum_{l=0}^{q} w_{\kappa(p,l,m,i)} x_i(t-l)$, denotes the *distributed lag* term, and w_0 and $\varepsilon(t)$ are the constant and the error term, respectively. If the error term is centered on 0, then

$$\mathbb{E}[y(t)] = w_0 + \sum_{k=1}^{p} w_k y(t-k) + \sum_{i=1}^{m} \sum_{l=0}^{q} w_{\kappa(p,l,m,i)} x_i(t-l). \tag{4.37}$$

4.2.2 Identification

The identification of the ADL model is done using Vector Autoregressive (VAR) for lag order p and q. The theory of this subject will not be discussed since it is beyond the scope of this paper. In Bayesian perspective, on the other hand, the order can be modeled by specifying *a priori* on these parameters.

4.2.3 Estimation

Estimation in frequentist's approach is done using either of the two procedures. The first method is through ordinary least squares which does not assume any constraints on the parameters, and that the error term of the model is Gaussian distributed with mean 0 and with constant variance. The other approach follows from constraining the values of the parameters, these procedures include polynomial constraints and spline constraints (Welty et al., 2009).

In this thesis, the Bayesian approach to modeling is done by setting *a priori* on the parameters of the model. The Bayesian principle of setting *a priori* is analogous to constraining the estimate in the classical statistics, and this specification is based on expert knowledge of the researcher.

Chapter 5

MAIN RESULTS

The theoretical results and application of this thesis are the main subject of this chapter. In particular, Section 5.1 presents the main propositions used for modeling BADL(1, 1), where Proposition 5.1.1 aims to derive the posterior distribution of the parameter, Proposition 5.1.2 is the gradient of the potential energy needed in the SGHMC algorithm, and Proposition 5.1.3 is an immediate result from the preceding proposition and is needed in the HMC algorithm. Further, Section 5.2 is the application of this thesis, which aims to forecast the Philippines' economic growth using BADL(1, 1).

5.1 Theoretical Results

Proposition 5.1.1. *Let* $\mathscr{D} = \{[\mathbf{x}(t), y(t)], \forall t \in \mathbb{Z}_+^\tau\}$ *be the data such that* $y(t)$ *is modeled by a Gaussian function with mean given in Equation (4.37) and constant variance* $\alpha^{-1} \in \mathbb{R}_+$. *If* $\mathbf{w}$ *is the vector of coefficients of ADL(*p, q*) such that* $\mathbf{w} \sim \mathcal{N}_d(\mathbf{0}, \beta^{-1}\mathbf{I})$, *where* $\beta^{-1} \in \mathbb{R}_+$, *then the posterior is a multivariate Gaussian distribution with covariance matrix* $\boldsymbol{\Sigma} = (\alpha\mathfrak{G}^{\mathrm{T}}\mathfrak{G} + \beta\mathbf{I})^{-1}$ *and mean vector* $\boldsymbol{\mu} = \alpha\boldsymbol{\Sigma}\mathfrak{G}^{\mathrm{T}}\mathbf{y}$, *where* $\mathfrak{G}$ *is the design matrix.*

Proof. Let $\mathbf{w} \triangleq [w_0 \ w_1 \ \cdots \ w_{\kappa(p,q,m,m)}]^{\mathrm{T}}$, $\kappa(p, l, m, m) \triangleq [(p + 1) + l \cdot m] + m$ and let $\mathbf{z}(t) \triangleq [1 \ y(t-1) \ \cdots \ x_m(t-q)]^{\mathrm{T}}$, then the ADL$(p, q)$ can be written as

$$y(t) = \mathbf{w}^{\mathrm{T}}\mathbf{z}(t) + \varepsilon(t), \quad \varepsilon(t) \sim \mathcal{N}(0, \alpha^{-1}). \tag{5.1}$$

The likelihood is therefore given by

$$\mathcal{L}(\mathbf{w}|\mathbf{y}) \triangleq \left(\frac{\alpha}{2\pi}\right)^{\tau/2} \exp\left\{-\sum_{t=1}^{\tau} \frac{\alpha[y(t) - \mathbf{w}^{\mathrm{T}}\mathbf{z}(t)]^2}{2}\right\}. \tag{5.2}$$

Let $\mathbf{y} \triangleq [y(1)\ y(2)\ \cdots\ y(\tau)]^{\mathrm{T}}$ and let $\mathfrak{G} \triangleq [(\mathbf{z}(t)^{\mathrm{T}})]$, i.e. $\mathfrak{G} \in \mathbb{R}^{\tau} \times \mathbb{R}^{d}$. Thus in matrix form

$$\mathcal{L}(\mathbf{w}|\mathbf{y}) \propto \exp\left[-\frac{\alpha}{2}(\mathbf{y} - \mathfrak{G}\mathbf{w})^{\mathrm{T}}(\mathbf{y} - \mathfrak{G}\mathbf{w})\right]. \tag{5.3}$$

The prior is given by

$$\mathbb{P}(\mathbf{w}) = \frac{1}{\sqrt{(2\pi)^{d}|\beta^{-1}\mathbf{I}|}} \exp\left[-\frac{1}{2}\mathbf{w}^{\mathrm{T}}\beta\mathbf{I}\mathbf{w}\right], \tag{5.4}$$

so that the posterior would be

$$\mathbb{P}(\mathbf{w}|\mathbf{y}) \propto \exp\left[-\frac{\alpha}{2}(\mathbf{y} - \mathfrak{G}\mathbf{w})^{\mathrm{T}}(\mathbf{y} - \mathfrak{G}\mathbf{w})\right] \exp\left[-\frac{1}{2}\mathbf{w}^{\mathrm{T}}\beta\mathbf{I}\mathbf{w}\right] \tag{5.5}$$

$$= \exp\left\{-\frac{1}{2}\left[\alpha(\mathbf{y} - \mathfrak{G}\mathbf{w})^{\mathrm{T}}(\mathbf{y} - \mathfrak{G}\mathbf{w}) + \mathbf{w}^{\mathrm{T}}\beta\mathbf{I}\mathbf{w}\right]\right\}. \tag{5.6}$$

Expanding the terms in the exponential factor becomes

$$\alpha\mathbf{y}^{\mathrm{T}}\mathbf{y} - 2\alpha\mathbf{w}^{\mathrm{T}}\mathfrak{G}^{\mathrm{T}}\mathbf{y} + \mathbf{w}^{\mathrm{T}}(\alpha\mathfrak{G}^{\mathrm{T}}\mathfrak{G} + \beta\mathbf{I})\mathbf{w}. \tag{5.7}$$

Hence

$$\mathbb{P}(\mathbf{w}|\mathbf{y}) \propto C \exp\left\{-\frac{1}{2}\left[\mathbf{w}^{\mathrm{T}}(\alpha\mathfrak{G}^{\mathrm{T}}\mathfrak{G} + \beta\mathbf{I})\mathbf{w} - 2\alpha\mathbf{w}^{\mathrm{T}}\mathfrak{G}^{\mathrm{T}}\mathbf{y}\right]\right\}. \tag{5.8}$$

The terms in the exponential factor is now of the form $ax^2 - 2bx$. This suggest a quadratic equation and therefore can be factored by completing the square. To do so, let $\mathbf{D} \triangleq \alpha\mathfrak{G}^{\mathrm{T}}\mathfrak{G} + \beta\mathbf{I}$ and $\mathbf{b} \triangleq \alpha\mathfrak{G}^{\mathrm{T}}\mathbf{y}$, then

$$\mathbb{P}(\mathbf{w}|\mathbf{y}) \propto C \exp\left\{-\frac{1}{2}\left[\mathbf{w}^{\mathrm{T}}\mathbf{D}\mathbf{w} - 2\mathbf{w}^{\mathrm{T}}\mathbf{b}\right]\right\} \tag{5.9}$$

$$= C \exp\left\{-\frac{1}{2}\left[\mathbf{w}^{\mathrm{T}}\mathbf{D}\mathbf{w} - \mathbf{w}^{\mathrm{T}}\mathbf{b} - \mathbf{b}^{\mathrm{T}}\mathbf{w}\right]\right\}. \tag{5.10}$$

Next is to add a term that is not a function of $\mathbf{w}$ which can be assumed to be part of the constant C. Let this term be $\mathbf{b}^{\mathrm{T}}\mathbf{D}^{-1}\mathbf{b}$, then

$$\mathbb{P}(\mathbf{w}|\mathbf{y}) \propto C \exp\left\{-\frac{1}{2}\left[\mathbf{w}^{\mathrm{T}}\mathbf{D}\mathbf{w} - \mathbf{w}^{\mathrm{T}}\mathbf{b} - \mathbf{b}^{\mathrm{T}}\mathbf{w} + \mathbf{b}^{\mathrm{T}}\mathbf{D}^{-1}\mathbf{b}\right]\right\}. \tag{5.11}$$

In order to proceed, the matrix $\mathbf{D}$ must be symmetric and invertible since later this will be the covariance matrix of the posterior which requires such property. If satisfied, then $\mathbf{I} = \mathbf{D}\mathbf{D}^{-1} = \mathbf{D}^{-1}\mathbf{D}$, so that

$$\mathbb{P}(\mathbf{w}|\mathbf{y}) \propto C \exp\left\{-\frac{1}{2}\left[\mathbf{w}^{\mathrm{T}}\mathbf{D}\mathbf{w} - \mathbf{w}^{\mathrm{T}}\mathbf{D}\mathbf{D}^{-1}\mathbf{b} - \mathbf{b}^{\mathrm{T}}\mathbf{D}^{-1}\mathbf{D}\mathbf{w} + \mathbf{b}^{\mathrm{T}}\mathbf{D}^{-1}\mathbf{D}\mathbf{D}^{-1}\mathbf{b}\right]\right\}.$$

Finally, let $\boldsymbol{\Sigma} \triangleq \mathbf{D}^{-1}$ and $\boldsymbol{\mu} \triangleq \mathbf{D}^{-1}\mathbf{b}$, then

$$\begin{aligned}\mathbb{P}(\mathbf{w}|\mathbf{y}) &\propto C \exp\left\{-\frac{1}{2}\left[\mathbf{w}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\mathbf{w} - \mathbf{w}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu} - \boldsymbol{\mu}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\mathbf{w} + \boldsymbol{\mu}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}\right]\right\} &(5.12)\\ &= C \exp\left\{-\frac{1}{2}\left[(\mathbf{w}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{w}-\boldsymbol{\mu})\right]\right\}. &(5.13)\end{aligned}$$

Thus $C = \frac{C_0}{\mathbb{P}(\mathbf{y})}$, where C is the constant of the Guassian kernel in Equation (5.13). Therefore,

$$\mathbb{P}(\mathbf{w}|\mathbf{y}) = \mathcal{N}_d(\mathbf{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma}), \tag{5.14}$$

where $\boldsymbol{\Sigma} = (\alpha\mathfrak{G}^{\mathrm{T}}\mathfrak{G} + \beta\mathbf{I})^{-1}$ and $\boldsymbol{\mu} = \alpha\boldsymbol{\Sigma}\mathfrak{G}^{\mathrm{T}}\mathbf{y}$. □

Proposition 5.1.2. *Let the posterior of the parameters be* $\mathbb{P}(\mathbf{w}|\mathbf{y})$ *given in Proposition 5.1.1, with* $\mathbf{y} = [y(1)\ y(2)\ \cdots\ y(\tau)]^{\mathrm{T}}$. *Further, let* $\mathbf{w} \sim \mathcal{N}_d(\mathbf{0}, \beta^{-1}\mathbf{I})$, *then the gradient noise of* $-\log\mathbb{P}(\mathbf{w}|\mathbf{y})$, *needed for SGHMC's computation is given below:*

$$-\alpha\sum_{t=1}^{\tau}(y(t) - \mathbf{w}^T\mathbf{z}(t))\mathbf{z}(t) + \beta\mathbf{w} + \xi, \quad \xi \sim \mathcal{N}(\mathbf{0}, \mathfrak{A}(\mathbf{w})). \tag{5.15}$$

Proof. Again, let $\mathbf{w} \triangleq [w_0\ w_1\ \cdots\ w_{\kappa(p,q,m,m)}]^{\mathrm{T}}$, $\kappa(p,l,m,m) \triangleq [(p+1) + l\cdot m] + m$ and let $\mathbf{z}(t) \triangleq [1\ y(t-1)\ \cdots\ x_m(t-q)]^{\mathrm{T}}$, then

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}\,\mathbf{w}}[-\log\mathbb{P}(\mathbf{w}|\mathbf{y})] &= -\frac{\mathrm{d}}{\mathrm{d}\,\mathbf{w}}\left[\ell(\mathbf{w}|\mathbf{y}) + \log\mathbb{P}(\mathbf{w}) - \log\mathbb{P}(\mathbf{y})\right] &(5.16)\\ &= -\left[\frac{\mathrm{d}}{\mathrm{d}\,\mathbf{w}}\ell(\mathbf{w}|\mathbf{y}) + \frac{\mathrm{d}}{\mathrm{d}\,\mathbf{w}}\log\mathbb{P}(\mathbf{w})\right] &(5.17)\end{aligned}$$

where analogous to Equation (3.18)

$$\frac{\mathrm{d}}{\mathrm{d}\,\mathbf{w}}\ell(\mathbf{w}|\mathbf{y}) = \frac{\mathrm{d}}{\mathrm{d}\,\mathbf{w}}\log\left\{\left(\frac{\alpha}{2\pi}\right)^{\tau/2}\exp\left[-\sum_{t=1}^{\tau}\frac{\alpha(y(t)-\mathbf{w}^{\mathrm{T}}\mathbf{z}(t))^2}{2}\right]\right\} \tag{5.18}$$

$$= -\frac{\mathrm{d}}{\mathrm{d}\,\mathbf{w}}\sum_{t=1}^{\tau}\frac{\alpha(y(t)-\mathbf{w}^{\mathrm{T}}\mathbf{z}(t))^2}{2} = \alpha\sum_{t=1}^{\tau}(y(t)-\mathbf{w}^{\mathrm{T}}\mathbf{z}(t))\mathbf{z}(t), \tag{5.19}$$

and the derivative of the prior with log transformation is given by

$$\frac{\mathrm{d}}{\mathrm{d}\,\mathbf{w}}\log\mathbb{P}(\mathbf{w}) = -\frac{\beta}{2}\frac{\mathrm{d}}{\mathrm{d}\,\mathbf{w}}\mathbf{w}^{\mathrm{T}}\mathbf{w} = -\beta\mathbf{w}. \tag{5.20}$$

Equation (5.15) then follows from Equation (4.26). □

Proposition 5.1.3. *The gradient of the potential energy needed for Hamiltonian Monte Carlo is given by*

$$-\alpha\sum_{t=1}^{\tau}(y(t)-\mathbf{w}^{T}\mathbf{z}(t))\mathbf{z}(t)+\beta\mathbf{w}. \tag{5.21}$$

Proof. The proof follows from the previous result. □

5.2 Forecasting Philippine's Economic Growth

As already mentioned in the first chapter of this paper, the Autoregressive Distributed Lag (ADL) model has been applied to different time series problems across disciplines. Most of these, however, are non-Bayesians. Hence, this thesis attempts to contribute to the application of the Bayesian ADL (BADL) using Stochastic Gradient Hamiltonian Monte Carlo (SGHMC). In particular, the model is applied to forecasting the Philippine's economic growth, or the year-over-year GDP's growth rate. The following are the economic indicators used for forecasting the GDP's growth rate:

1. Peso/US Dollar Exchange Rate (ERATE)

2. Stock Price Index (SPI)

3. Gross International Reserves (GIR)

4. Balance of Payments - Current Account (BOP)

The indicators are known to have correlation with the gross domestic product of the Philippines, for example the first three indicators above were tested by Mapa, Del Prado, Poliquit, and Asaad (2016). The Balance of Payments, on the other hand, is also an important predictor since according to the International Monetary Fund (2005) that, the flows reflected in the Balance of Payments affect, in important ways, the total economy's activities associated with production, generation and distribution of income, consumption, and accumulation activities. For instance, credit and debit entries for goods and services in balance of payments accounts are equivalent to flows of exports and imports of goods and services. These flows are refected in the economy's account for goods and services and consequently affect the measurement of gross domestic product (GDP) and its composition in terms of final demand components.

The Philippines' economic growth rate is quarterly released by the Philippine Statistics Authority (PSA), which uses a year-over-year growth rate computation. Thus in order to relate to the publication of the PSA, the time series used in this paper are also in terms of year-over-year growth rate. In particular, the rates are extracted from the deseasonalized time series. However, it should be noted that the study can also proceed without deseasonalizing since the year-over-year growth rate is not affected by the seasonality. The plot of these rates are given in Figure 5.1.

Analogous to classical statistics, the stationarity of the time series must be assessed before proceeding to modeling. The Augmented-Dickey Fuller

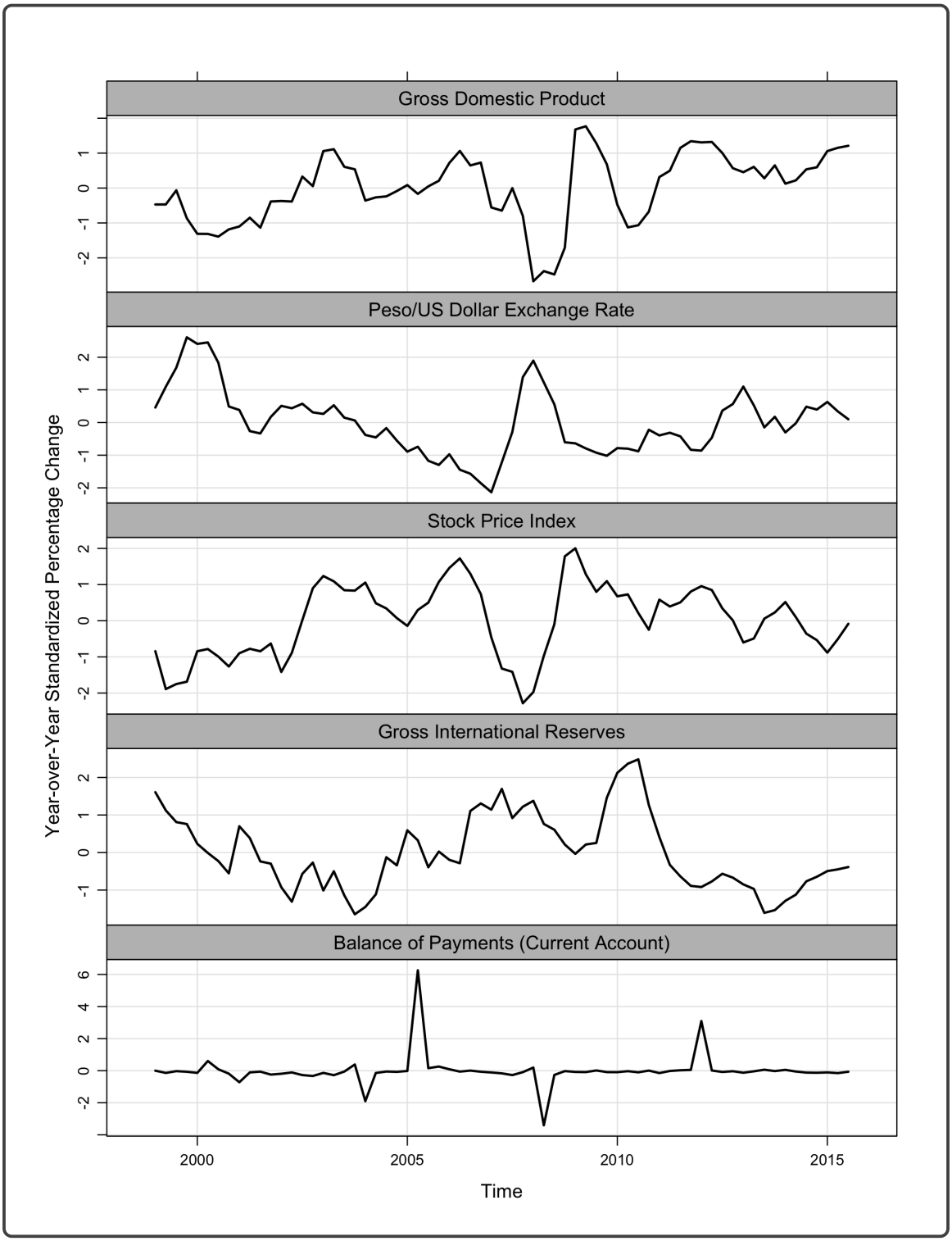


Figure 5.1: *Time Series of the Economic Indicators used in the Study.*

(ADF) stationary test suggests that all indicators are nonstationary. Hence first differentiation on the time series must be applied. However, the ADL is a specialized type of a dynamic regression (Welty et al., 2009), thus the assumption of stationarity can be relaxed according to Kumar and Maity (2008), and because of that, this thesis will proceed with the nonstationary series.

To test the performance of the proposed model, the data are partitioned into training and testing datasets. Specifically, 70% of the data points are allocated to the training dataset and the remaining 30% are reserved for the testing dataset.

5.2.1 BADL(1, 1)-SGHMC Posterior Inference

The ADL(1, 1) model derived from Equation (4.36) has the following form:

$$y(t) = w_0 + w_1 y(t-1) + \sum_{i=1}^{4}\sum_{l=0}^{1} w_{(2+l\cdot 4)+i} x_i(t) + \varepsilon(t), \quad \varepsilon(t) \sim \mathcal{N}(0, 1/\alpha),$$

where y is the dependent variable, which is the growth rate of the GDP (the reference series); the x_is, on the other hand, correspond to the economic indicators listed in Figure 5.1; the weights, $w_i, i \in \{0, 1, \cdots, 10\}$ are assumed to have the following prior distribution: let $\mathbf{w} \triangleq [w_0\ w_1\ \cdots\ w_{10}]^{\mathrm{T}}$, then

$$\mathbf{w} \sim \mathcal{N}_{10}(\mathbf{0}, \beta\mathbf{I}). \tag{5.22}$$

From Proposition 5.1.2, the gradient noise of the posterior distribution is given by

$$-\alpha \sum_{t=1}^{\tau} (y(t) - \mathbf{w}^{\mathrm{T}}\mathbf{z}(t))\mathbf{z}(t) + \beta\mathbf{w} + \boldsymbol{\xi}, \quad \boldsymbol{\xi} \sim \mathcal{N}(\mathbf{0}, \mathfrak{A}(\mathbf{w})). \tag{5.23}$$

Other parameters of the SGHMC which include $\mathfrak{C}$ and $\mathfrak{A}$ are set to identity matrix, while the kinetic energy given in Equation (4.18), which is another parameter of the SGHMC, has gradient $\frac{\mathbf{p}}{2}$. As discussed in Chapter 4, $\mathbf{p}$ is the momentum parameter sampled from the kinetic energy as indicated in Algorithm 6.

Posterior Summaries

The parameters of the model are estimated through Markov Chain Monte Carlo simulation, and for this paper, four markov chains were considered. Each of these chains has overdispersed starting position, which can be set to any value. However for this thesis, the initial positions are constrained to $[0.001, 1]$ interval. This interval is divided into partitions to make the starting values overdispersed. That is, the first markov chain takes random starts for 8 parameters (*see* Table 5.1) from the first-quartile of the said interval using uniform distribution, so that in general $\mathbf{w}_{\text{chain}} \sim \mathcal{U}(0.001 + (\text{chain} - 1)/4, \text{chain}/4)$. That is, if chain = 1, then $\mathbf{w}_1 \sim \mathcal{U}(0.001, 1/4)$; and so on. The MCMC simulation is done for three different cases, and these are the following:

- **Case 1** Leapfrog step size $\gamma = .09$ for 1,000 iterations;
- **Case 2** Leapfrog step size $\gamma = .009$ for 10,000 iterations;
- **Case 3** Leapfrog step size $\gamma = .0009$ for 100,000 iterations.

These scenarios are useful for preliminary evaluation on the behavior of the MCMC algorithms used in this paper, especially for the SGHMC since no paper yet has studied its optimal leapfrog parameter. It should be emphasized, however, that these cases explains a small perspective on the overall behavior of the SGHMC. The estimate of the parameters for the three cases are provided in Table 5.1. The estimates were obtained by averaging the four markov chains simulated. That is, the output of the simulation is four matrices (corresponding to the four markov chains), each with dimension 1,000 by 8 (e.g. for the first case). The columns are the corresponding parameters and the rows are the sampled estimate from

	Variables	**BADL**$(1,1)$-**MH**		**BADL**$(1,1)$-**HMC**		**BADL**$(1,1)$-**SGHMC**	
		Coefficient	Std. Error	Coefficient	Std. Error	Coefficient	Std. Error
1st Case	w_0	-0.127	0.123	-0.066	0.079	-0.068	0.103
	GDP$(t-1)$	0.439	0.144	0.261	0.052	0.434	0.140
	ERATE(t)	-0.102	0.087	0.333	0.154	0.072	0.199
	SPI(t)	0.159	0.264	0.155	0.086	0.097	0.165
	GIR(t)	-0.129	0.198	-0.288	0.081	-0.317	0.170
	BOP(t)	0.094	0.108	0.000	0.031	0.039	0.093
	ERATE$(t-1)$	0.323	0.223	-0.399	0.144	-0.204	0.186
	SPI$(t-1)$	0.247	0.235	0.359	0.092	0.225	0.192
	GIR$(t-1)$	0.133	0.131	0.013	0.084	0.064	0.166
	BOP$(t-1)$	0.162	0.085	0.062	0.033	0.040	0.095
2nd Case	w_0	-0.082	0.130	-0.076	0.022	-0.066	0.045
	GDP$(t-1)$	0.292	0.105	0.304	0.046	0.430	0.039
	ERATE(t)	0.099	0.091	0.500	0.055	0.080	0.068
	SPI(t)	-0.049	0.122	0.114	0.055	0.110	0.065
	GIR(t)	-0.260	0.102	-0.355	0.039	-0.294	0.071
	BOP(t)	0.030	0.115	-0.019	0.021	0.040	0.042
	ERATE$(t-1)$	-0.268	0.166	-0.548	0.077	-0.206	0.074
	SPI$(t-1)$	0.459	0.154	0.419	0.070	0.220	0.074
	GIR$(t-1)$	-0.033	0.101	0.098	0.023	0.049	0.052
	BOP$(t-1)$	-0.006	0.101	0.038	0.023	0.045	0.038
3rd Case	w_0	-0.095	0.094	-0.094	0.022	-0.042	0.098
	GDP$(t-1)$	0.273	0.142	0.304	0.027	0.446	0.039
	ERATE(t)	0.693	0.181	0.523	0.043	0.108	0.089
	SPI(t)	0.345	0.090	0.172	0.035	0.148	0.093
	GIR(t)	-0.343	0.196	-0.325	0.038	-0.202	0.144
	BOP(t)	0.018	0.049	-0.004	0.020	0.066	0.078
	ERATE$(t-1)$	-0.822	0.169	-0.586	0.056	-0.124	0.158
	SPI$(t-1)$	0.173	0.113	0.345	0.041	0.266	0.093
	GIR$(t-1)$	0.048	0.170	0.071	0.033	0.045	0.061
	BOP$(t-1)$	0.081	0.053	0.062	0.017	0.084	0.091

Table 5.1: *Estimated Coefficients of the BADL(1, 1) using Different MCMCs Across Cases.*

the posterior distribution. These four matrices are averaged to obtain single mean matrix with the same dimension as the four matrices. Then, the first ten rows are discarded since the burn-in is set to 10, so that the current dimension of the matrix is 990 by 8. Thinning is then applied, by taking every 10th row of the matrix. Thus, the final dimension of the matrix after burn-in and thinning is 99 by 8. From this matrix, the column means are computed to arrive at 1 by 8 vector. This vector is the estimate of the **w** and is given in Table 5.1.

The standard error of the coefficients serve as insight on the stability of the sampled estimates. A quick investigation suggest that the SGHMC has, for some instance, smaller standard error compared to MH and HMC. The kernel density estimate (KDE) of the SGHMC's chains are given in Figures 5.2, 5.3, and 5.4; and the corresponding traces of these chains are given in Figures 5.5, 5.6 and 5.7. These figures are "unfiltered" since these are the raw samples, that is, no burn-in and thinning were applied yet. Hence subjectively, the markov chains simulated using SGHMC seems to have converged since the KDEs exhibit the same scale and location, especially for the first two cases. The third case seems to have shifting on its KDE, that is, not all chains have the same or very close location parameter. And with regards to the traces, by observation, as the step size (the leapfrog parameter) γ decreases, the fluctuation of chains gets concentrated and moves very slowly to the stationarity. This is evident especially on the third case, where the four non-overlapping starting values of the markov chains are clearly exposed.

In comparison to the two MCMCs, the MH and HMC, the plot of the KDEs and traces are available in Appendix A, specifically refer to Figures A.1 to A.12. From these figures, the samples obtained by MH seems to be unstable across cases

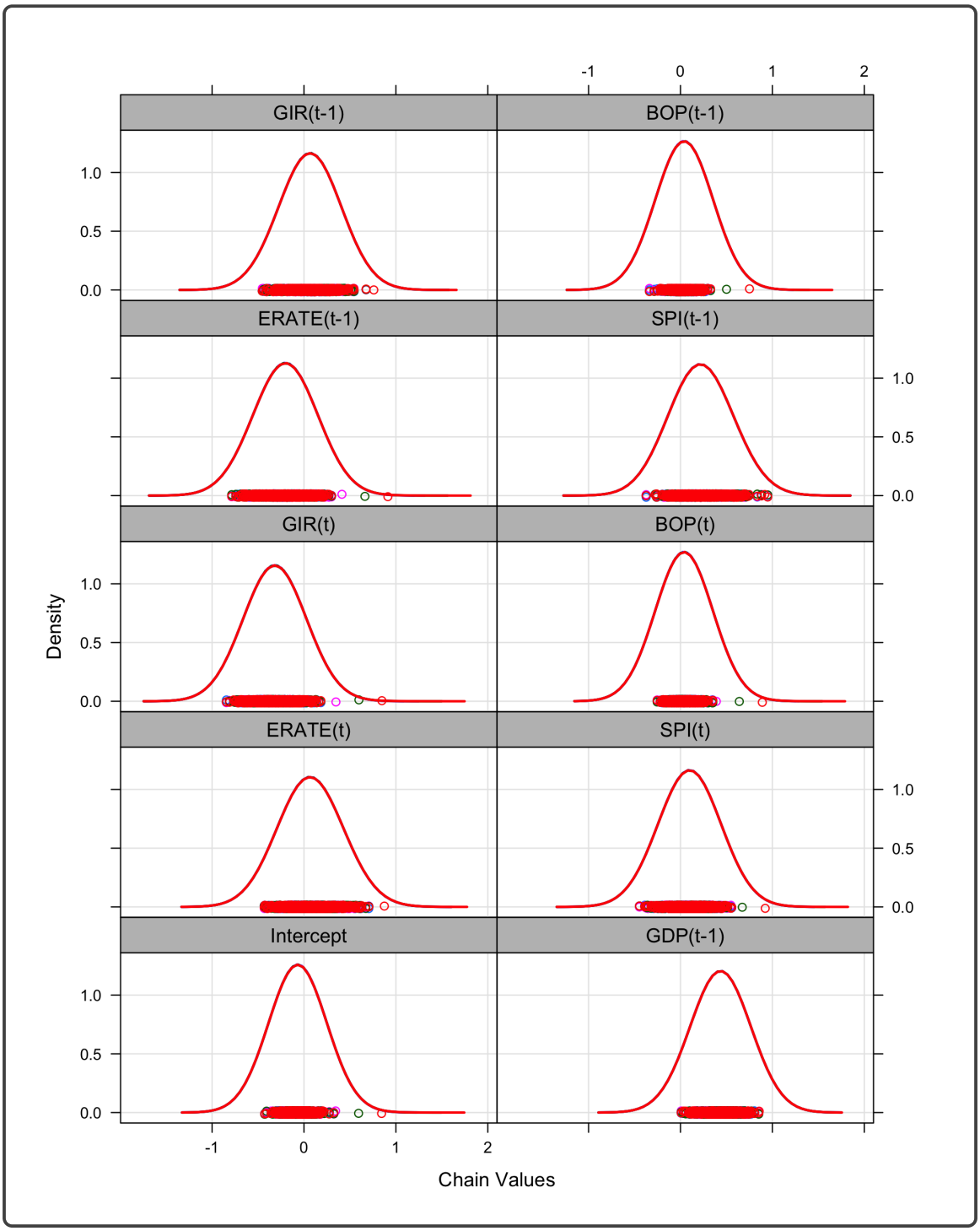


Figure 5.2: *Kernel Density Estimate of the Unfiltered Chains using Stochastic Gradient Hamiltonian Monte Carlo for Case 1 (γ = .09 for 1000 Iterations).*

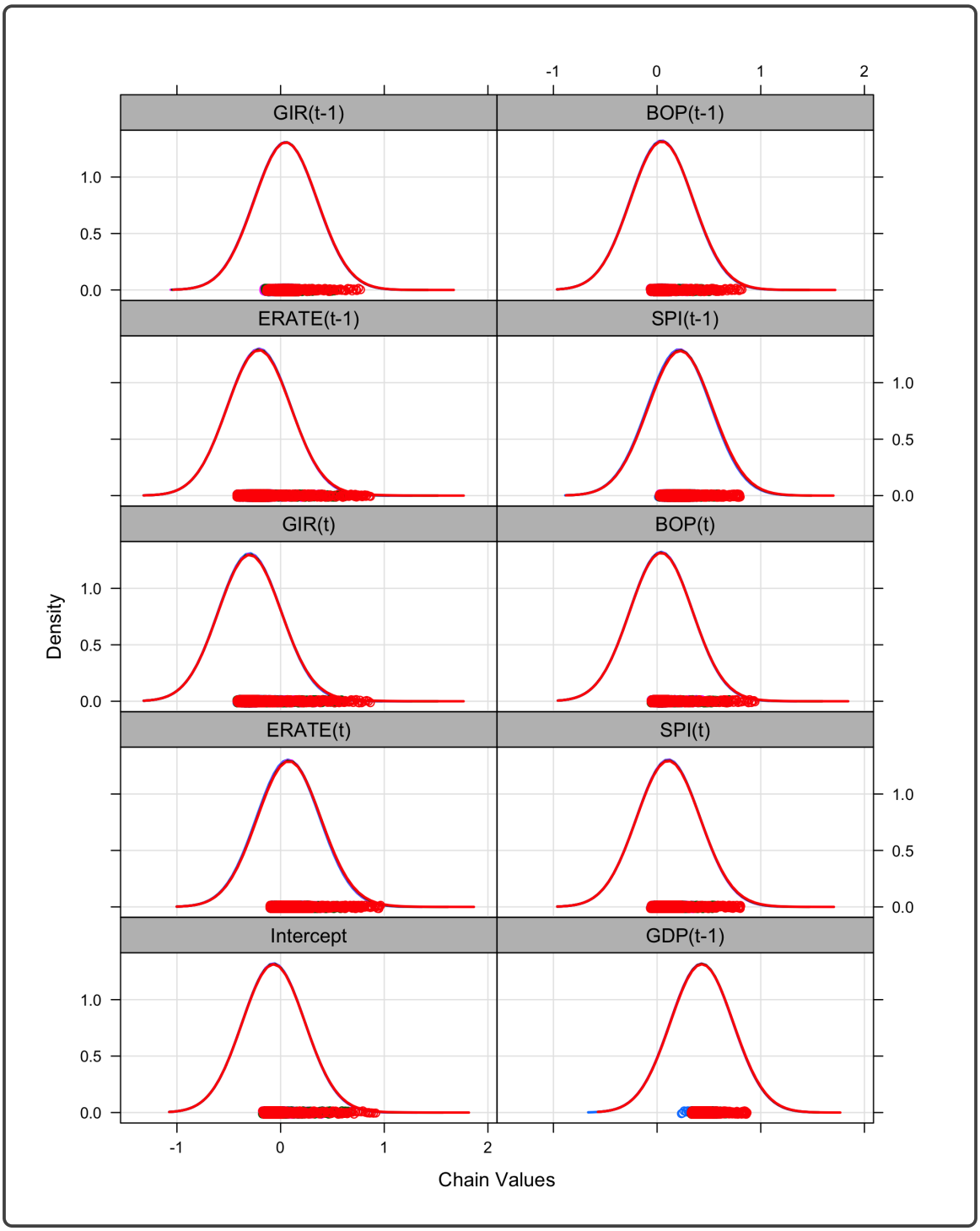


Figure 5.3: *Kernel Density Estimate of the Unfiltered Chains using Stochastic Gradient Hamiltonian Monte Carlo for Case 2 ($\gamma = .009$ for 10000 Iterations).*

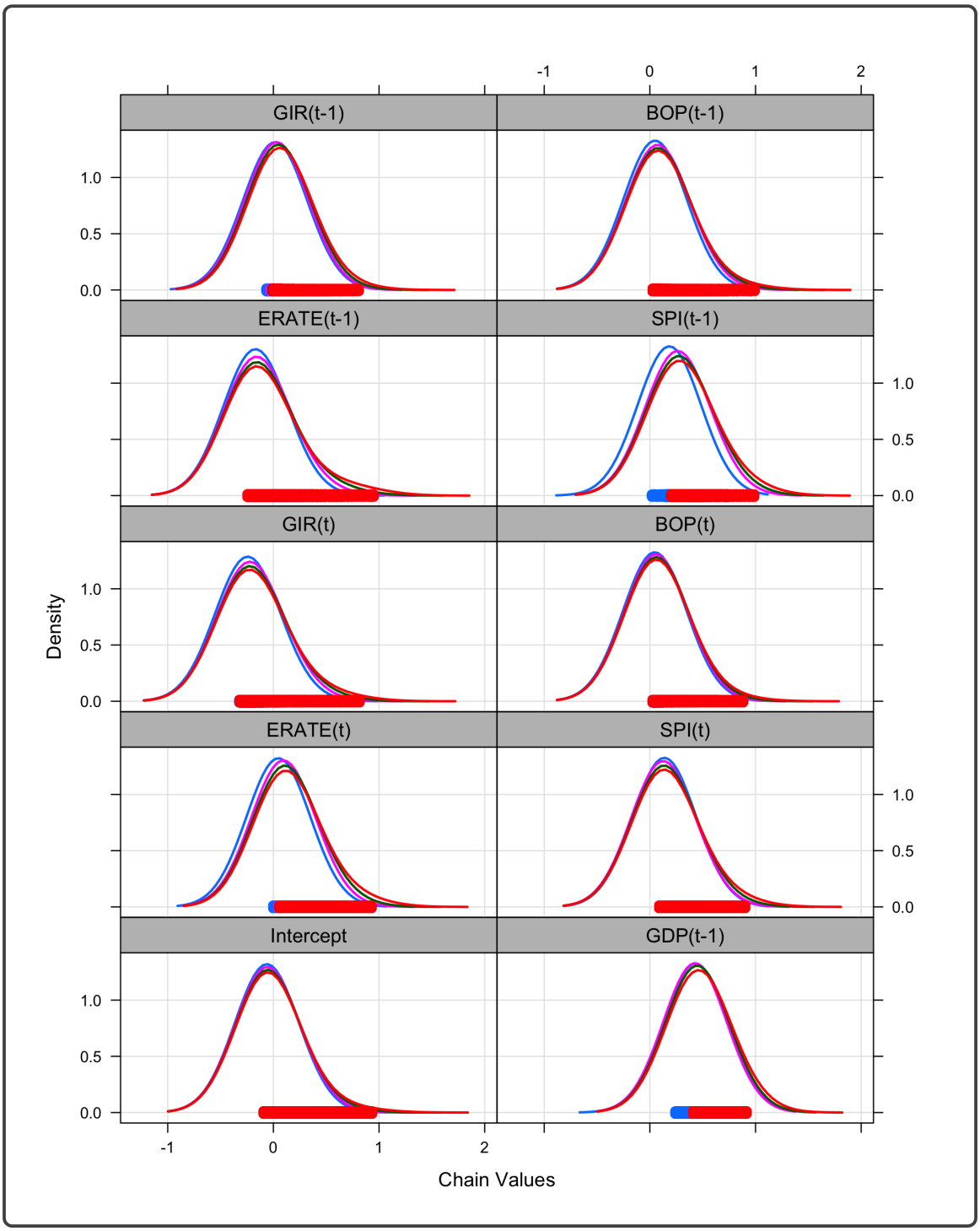


Figure 5.4: *Kernel Density Estimate of the Unfiltered Chains using Stochastic Gradient Hamiltonian Monte Carlo for Case 3 (γ = .0009 for 100000 Iterations).*

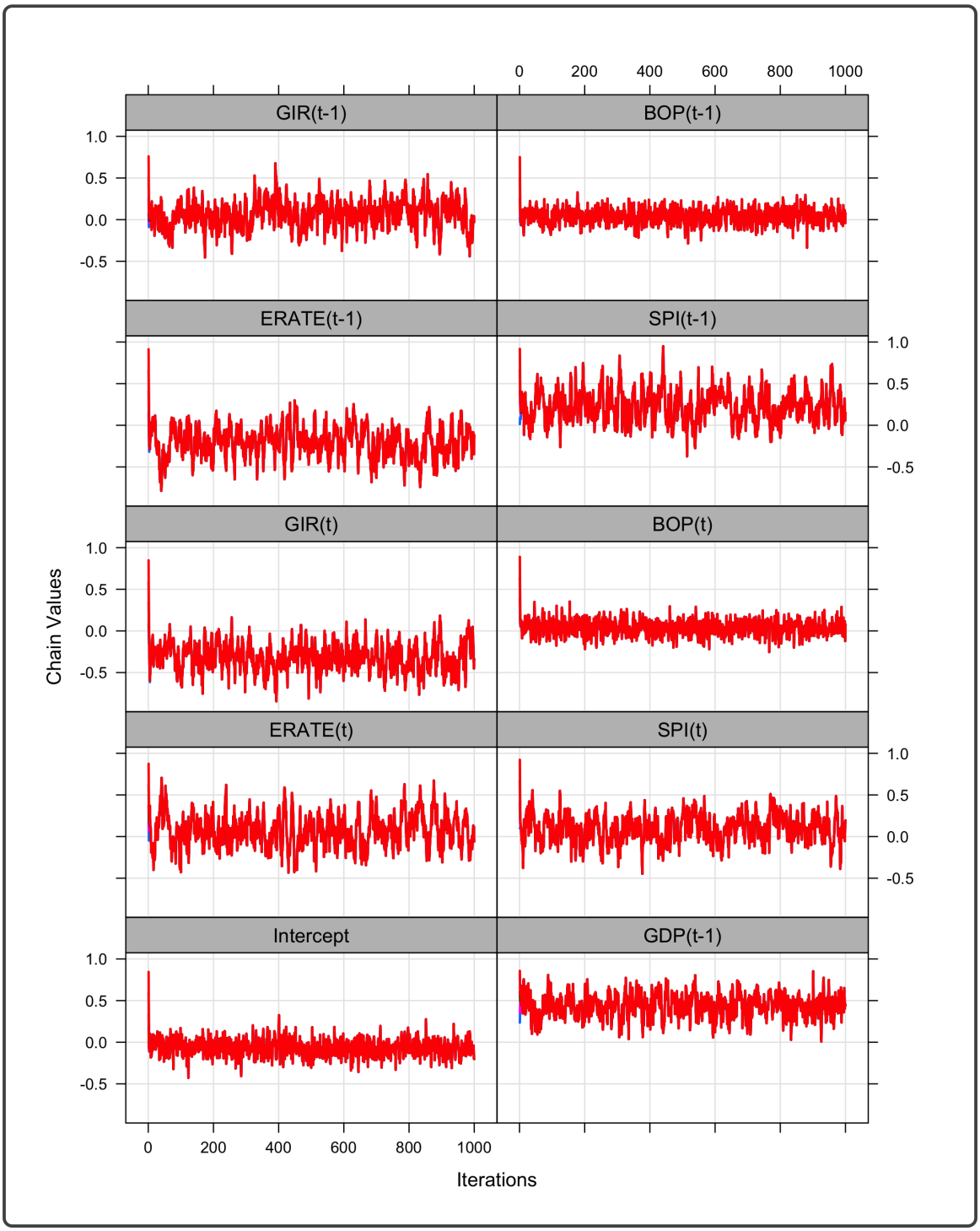


Figure 5.5: *Unfiltered Traces of the Chains using Stochastic Gradient Hamiltonian Monte Carlo for Case 1 ($\gamma = .09$ for 1000 Iterations).*

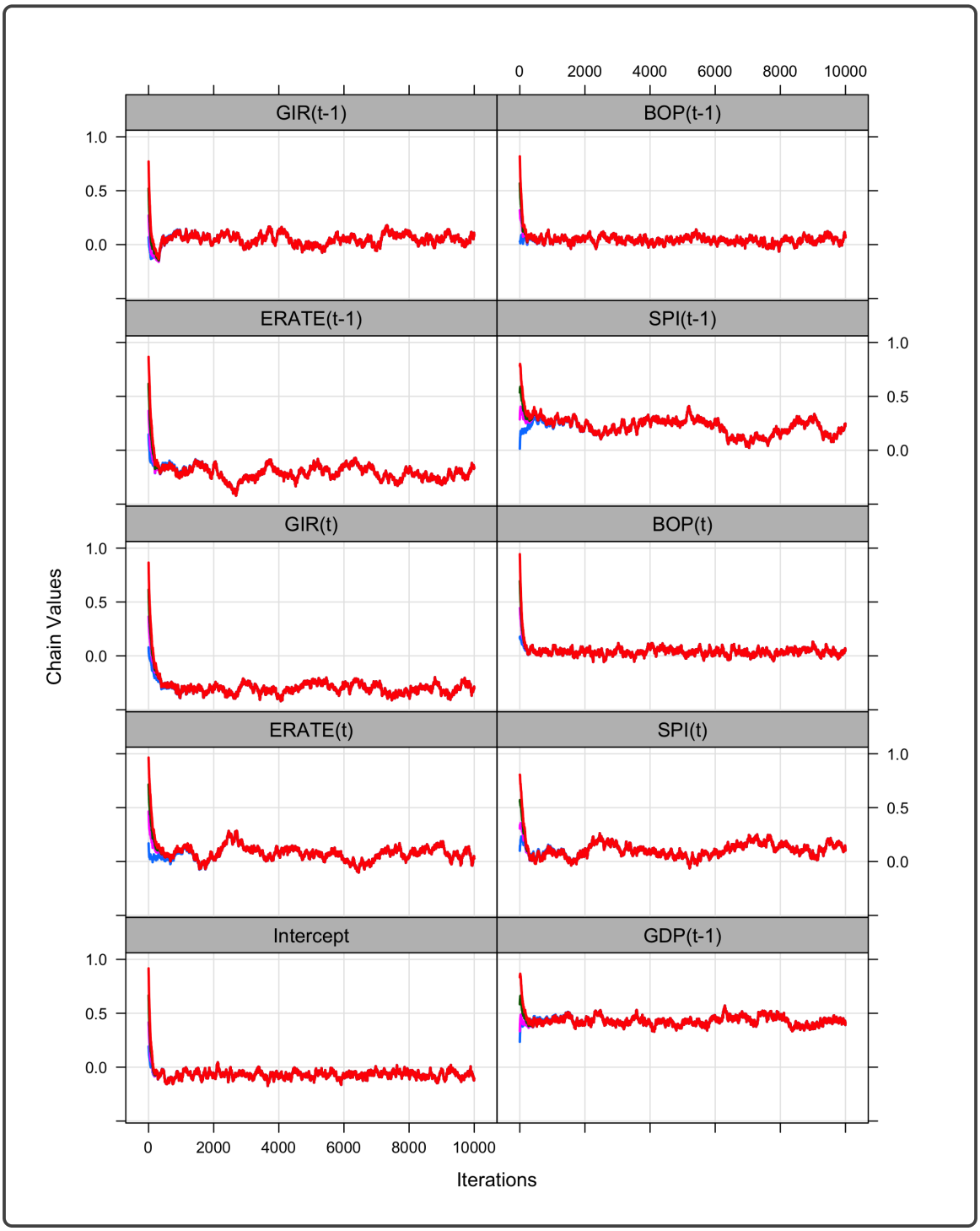


Figure 5.6: *Unfiltered Traces of the Chains using Stochastic Gradient Hamiltonian Monte Carlo for Case 2 (γ = .009 for 10000 Iterations).*

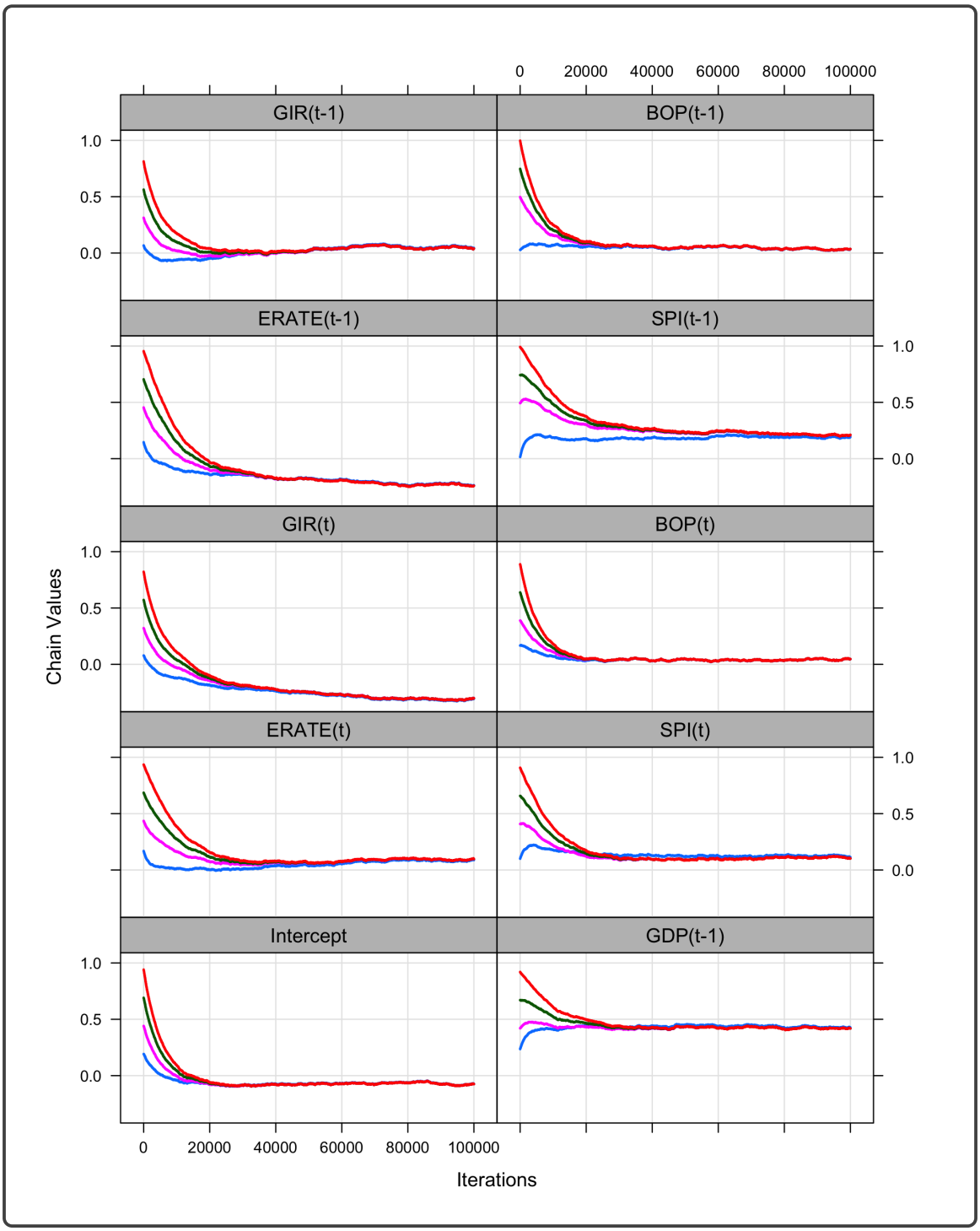


Figure 5.7: *Unfiltered Traces of the Chains using Stochastic Gradient Hamiltonian Monte Carlo for Case 3 ($\gamma = .0009$ for 100000 Iterations).*

since the KDE of the four chains have different scales and locations. Further, the traces of the samples under MH shows that most of the proposed samples were rejected. These findings are aligned with the discussion made in § 3.4.2, that the specification of the proposal distribution of the MH is difficult to tune under high dimensional parameter estimation. For this study, the proposal distribution used is a standard Gaussian centered on the samples drawn from the *a posteriori*. On the other hand, for the traces of the HMC, the first two cases seem to characterize the trace of the MH. That is, the rejection rate is also high. This follows from the fact that the HMC uses MH acceptance criterion due to γ discretization of the phase space. For the third case of the HMC's chains, the mixing seems fair since most of the proposed samples were accepted. Hence subjectively, the SGHMC is still on top of these contenders relative to all cases considered.

Convergence Tests

As discussed in the preceding section, the standard error in Table 5.1 gives an explanation on the stability of the estimates. Further, the discussion centered on this topic by inspecting subjectively the KDE and trace plots of the chains. In this section, however, the chains will be assessed objectively using statistical test. To test the stationarity of the chain, the Heidelberger-Welch test is used, refer to Table 5.2 for the stationarity of the SGHMC's mean chain. Again, the mean chain refers to the average of the four markov chains. From the table, the estimates are all stationary as indicated by the ✓ in the third column, for all cases except for the GIR($t-1$) variable in the third scenario. The fourth column corresponds to the starting point of the stationary series in the iteration. For example, the chain of the estimate of BOP(t) follows stationarity starting at

the 1001st iteration under Case 2. Hence, this column can be used for burn-in. The fifth column is the corresponding p-value. From the R documentation of the `heidel.diag` function, the half-width test, calculates a 95% confidence interval for the mean, using the portion of the chain which have passed the stationarity test. Half the width of this interval is compared with the estimate of the mean. If the ratio between the half-width and the mean is lower than ϵ, the halfwidth test is passed. Otherwise the length of the sample is deemed not long enough to estimate the mean with sufficient accuracy. From Table 5.2, the trace of the markov chain did not pass the latter test. The Heidelberger-Welch test for the mean chains of the MH and HMC are given in Appendix B, refer to Tables B.1 and B.2. The NAs (Not Applicable) in Table B.1 is due to the computation of the spectral density at frequency $\lambda = 0$, which is a fraction, where the numerator is the variance of the half-width of the time series. If the variance of this half-width series is zero, then the succeeding computation of the half-width test is affected leading to NAs.

The next test to consider is the convergence of the four markov chains, and is done using the Gelman-Rubin's convergence test which returns the potential scale reduction factor (PSRF) or shrink factor statistics. Specifically, if the PSRF is close to 1, then the four markov chains have converged to the stationarity. The results are given in Table 5.3, where the UCI stands for Upper Confidence Interval, and the point estimate is the corresponding PSRF statistic. From this table, the SGHMC's markov chains have converged since all of its PSRFs are close to 1, which is also comparable to HMC's shrink factors. It should be noted that the result for MH is not available due to complications in the computation of the Gelman-Rubin statistic, and this is also the reason for the test statistic of

	Variables	Stationarity Test			Halfwidth Test		
		Status	Start	p-value	Status	Mean	Halfwidth
1st Case	w_0	✓	101	0.457	✗	-0.072	0.008
	GDP($t-1$)	✓	1	0.963	✓	0.434	0.017
	ERATE(t)	✓	1	0.363	✗	0.072	0.033
	SPI(t)	✓	1	0.530	✗	0.097	0.026
	GIR(t)	✓	1	0.336	✓	-0.317	0.025
	BOP(t)	✓	1	0.130	✗	0.039	0.007
	ERATE($t-1$)	✓	1	0.224	✗	-0.204	0.034
	SPI($t-1$)	✓	1	0.946	✗	0.225	0.031
	GIR($t-1$)	✓	1	0.056	✗	0.064	0.023
	BOP($t-1$)	✓	1	0.516	✗	0.040	0.007
2nd Case	w_0	✓	1	0.180	✗	-0.066	0.010
	GDP($t-1$)	✓	1	0.660	✓	0.430	0.011
	ERATE(t)	✓	1	0.200	✗	0.080	0.025
	SPI(t)	✓	1	0.410	✗	0.110	0.027
	GIR(t)	✓	1001	0.670	✓	-0.304	0.013
	BOP(t)	✓	1001	0.760	✗	0.036	0.006
	ERATE($t-1$)	✓	1	0.130	✗	-0.206	0.025
	SPI($t-1$)	✓	1	0.220	✗	0.220	0.038
	GIR($t-1$)	✓	1	0.640	✗	0.049	0.017
	BOP($t-1$)	✓	1001	0.410	✗	0.041	0.007
3rd Case	w_0	✓	1	0.063	✗	-0.042	0.062
	GDP($t-1$)	✓	20001	0.146	✓	0.429	0.006
	ERATE(t)	✓	10001	0.105	✗	0.082	0.023
	SPI(t)	✓	1	0.176	✗	0.148	0.089
	GIR(t)	✓	40001	0.052	✗	-0.285	0.039
	BOP(t)	✓	1	0.178	✗	0.066	0.053
	ERATE($t-1$)	✓	40001	0.079	✗	-0.210	0.030
	SPI($t-1$)	✓	30001	0.052	✓	0.221	0.014
	GIR($t-1$)	✗	NA	0.047	NA	NA	NA
	BOP($t-1$)	✓	20001	0.122	✗	0.050	0.012

Table 5.2: *Heidelberger-Welch's Stationarity and Halfwidth Tests of the Stochastic Gradient Hamiltonian Monte Carlos's Mean Chain Across Cases. The ✓ indicates stationarity, ✗ indicates nonstationarity, and NA means Not Applicable.*

	Parameters	**BADL**(1, 1)-**HMC**		**BADL**(1, 1)-**SGHMC**	
		Point Estimate	95% UCI	Point Estimate	95% UCI
2nd Case	w_0	NA	NA	0.999	0.999
	GDP($t-1$)	NA	NA	0.999	0.999
	ERATE(t)	NA	NA	0.999	0.999
	SPI(t)	NA	NA	0.999	0.999
	GIR(t)	NA	NA	0.999	0.999
	BOP(t)	NA	NA	0.999	0.999
	ERATE($t-1$)	NA	NA	0.999	0.999
	SPI($t-1$)	NA	NA	0.999	0.999
	GIR($t-1$)	NA	NA	0.999	0.999
	BOP($t-1$)	NA	NA	0.999	0.999
3rd Case	w_0	1.001	1.002	1.013	1.039
	GDP($t-1$)	1.004	1.012	1.510	2.177
	ERATE(t)	1.008	1.020	1.209	1.524
	SPI(t)	1.002	1.006	1.901	3.043
	GIR(t)	1.004	1.010	1.020	1.061
	BOP(t)	1.000	1.001	1.004	1.012
	ERATE($t-1$)	1.009	1.021	1.017	1.052
	SPI($t-1$)	1.001	1.003	1.979	3.168
	BOP($t-1$)	1.000	1.001	1.008	1.024

Table 5.3: *Gelman-Rubin's Estimated Potential Scale Reduction or Shrink Factor of the Four Markov Chains. The NA means Not Applicable.*

the three methods under the first case and the second case result of the HMC. The problem in the algorithm of the Gelman-Rubin can be traced back to its Cholesky's decomposition where the input variance-covariance from the four chains are not positive definite. Thus objectively, the SGHMC's and HMC's estimates have converged to stationarity.

The final test to check is the IID assumption of the samples from the markov chains, and this is done by checking the autocorrelation. The results are available in Appendix A, *see* Figures A.13 to A.21. For the first case, comparing Figures A.13, A.16 and A.19, the SGHMC has by far the ideal autocorrelation compared to MH and HMC. It has the lowest magnitude even at earlier lags. However, for the second and third cases, the results are far from the characteristics of the IID samples. Note

that the mean chains used in here are already filtered, that is, burn-in and thinning were applied. Therefore, for IID samples, the SGHMC has the best chains for $\gamma = .09$ and for minimum iteration of 1,000 runs only.

Forecasting

The forecasts in the training sample for all cases are shown in Figures 5.8, 5.9 and 5.10. In these figures, the sampled forecast values of the BADL(1, 1)-MH are aligned with the rejection rate of its estimates, which is evident on the wide gaps between its sampled forecasts in comparison to BADL(1, 1)-HMC and BADL(1, 1)-SGHMC. Another observation is that, as the value of γ decreases, the sampled forecasts for HMC and SGHMC become unstable (refer to Case 2 and 3) compared to the higher values of this parameter. The root mean squared error (RMSE) of the estimates are given in Table 5.4. In particular, the error mentioned here is the in-sample error, or the RMSE under the training dataset. From this table, the in-sample error of the proposed model is not far from the two contenders. Further, it should be emphasized that, the forecast plots provided are point estimates varying through the *a posteriori* of the parameters. The interval estimates of future observations are not shown in the plots but can be derived by marginalizing on the parameters. This is because, in Bayesian,

Models	Root Mean Squared Error			
	Error Type	1st Case	2nd Case	3rd Case
BADL(1, 1)-**MH**	In-Sample	0.774	0.612	0.567
	Out-Sample	0.952	0.982	0.985
BADL(1, 1)-**HMC**	In-Sample	0.585	0.574	0.538
	Out-Sample	0.979	0.984	0.981
BADL(1, 1)-**SGHMC**	In-Sample	0.606	0.605	0.557
	Out-Sample	0.906	0.911	0.935

Table 5.4: *Root Mean Squared Error of the Mean Forecasts of Bayesian ADL.*

there are two sources of uncertainty: first is the uncertainty associated with the observation (the response variable), and second is the uncertainty associated with the parameters. Therefore, to obtain the prediction interval of future observation, the variability on the parameters must be marginalized, and this topic is part of the recommendation.

The true performance of the models considered in this aplication is not based on the training dataset, but rather on the testing dataset. The forecast plots under the testing dataset are given in Figures 5.11, 5.12 and 5.13. The out-sample errors of the BADL(1, 1)-SGHMC in Table 5.4 are the lowest for all cases.

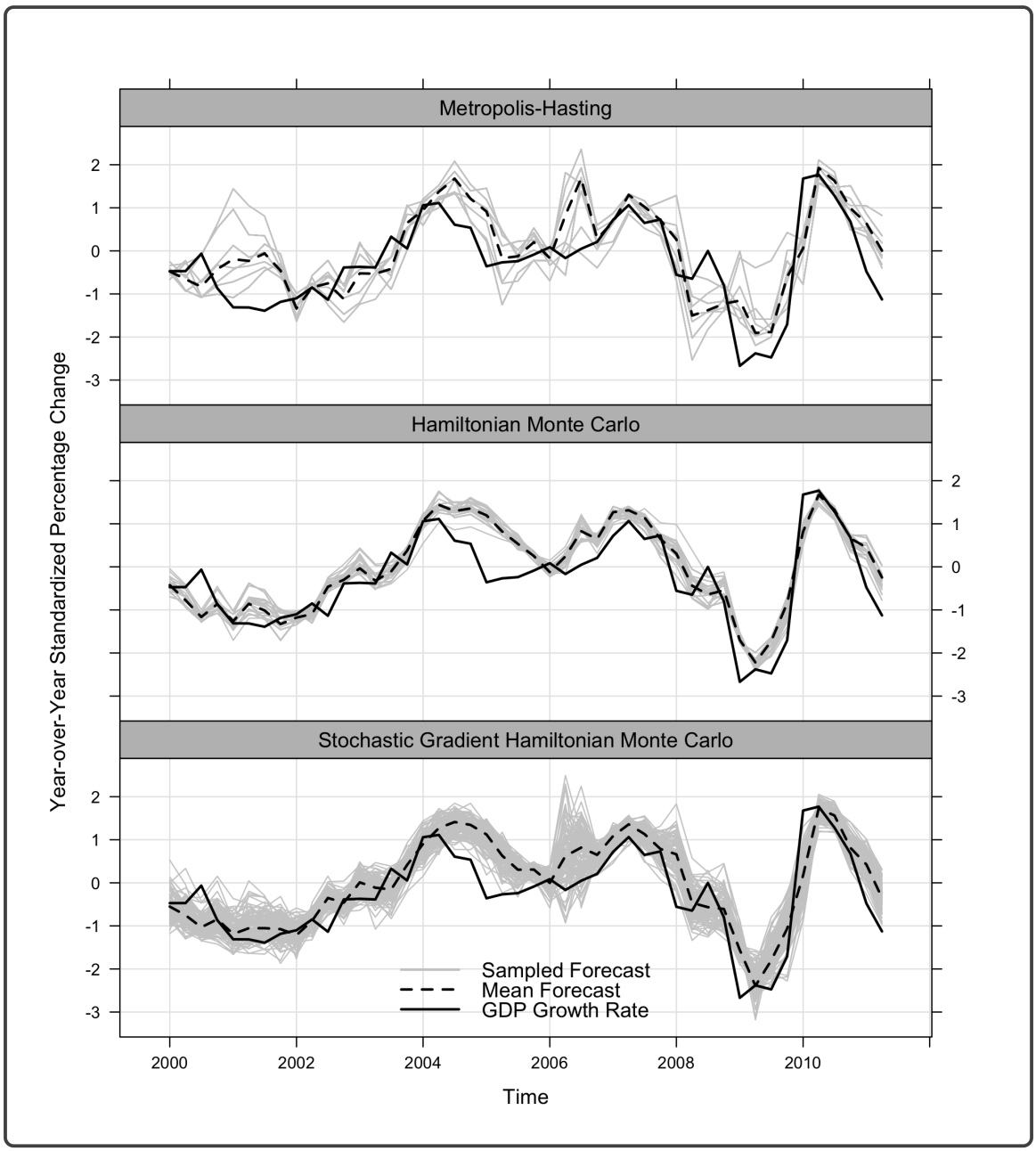


Figure 5.8: *Forecasts of BADL(1, 1) using different MCMCs under the Training Dataset Case 1 (γ = .09 for 1000 Iterations).*

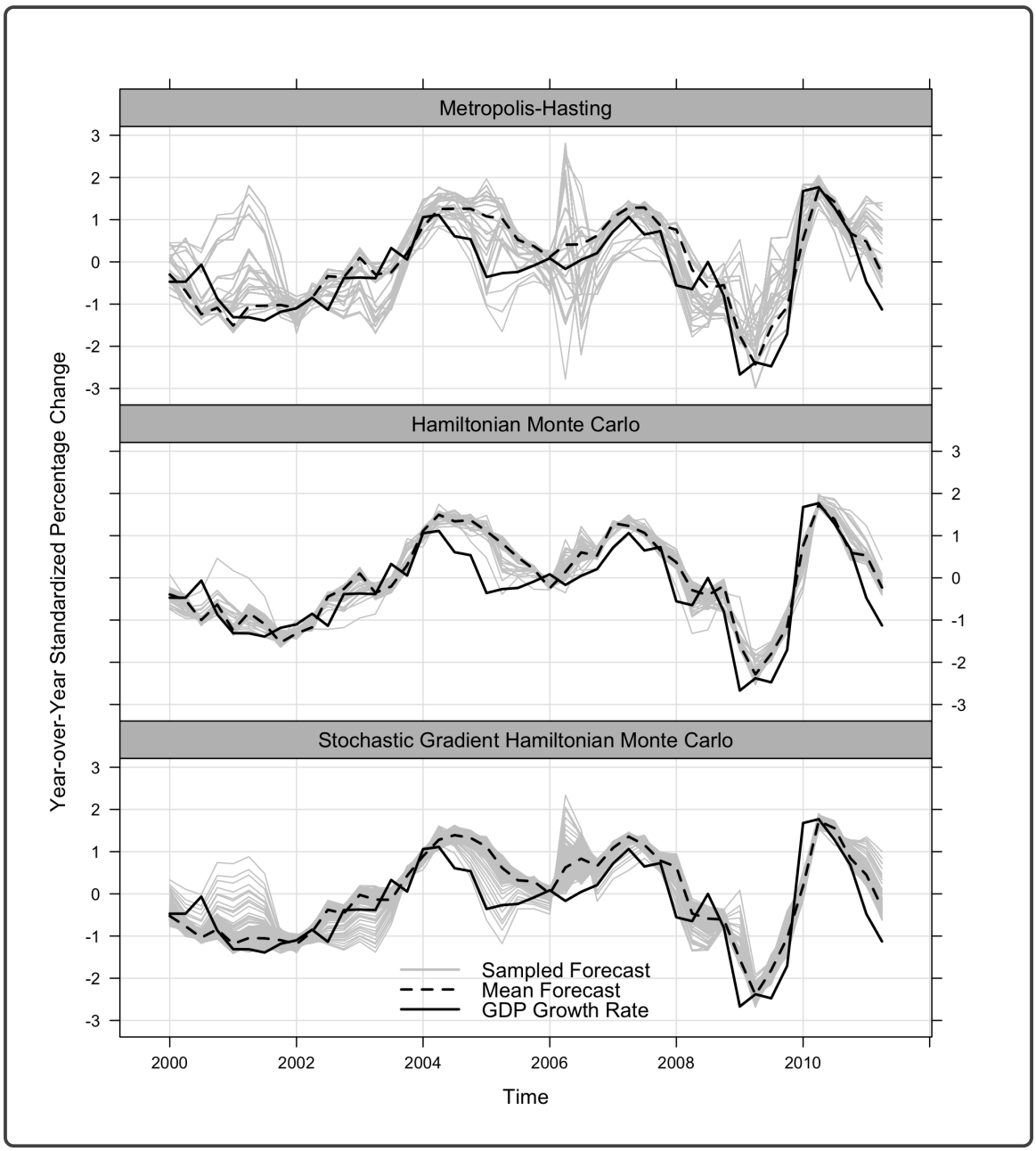


Figure 5.9: *Forecasts of BADL(1, 1) using different MCMCs under the Training Dataset Case 2 (γ = .009 for 10000 Iterations).*

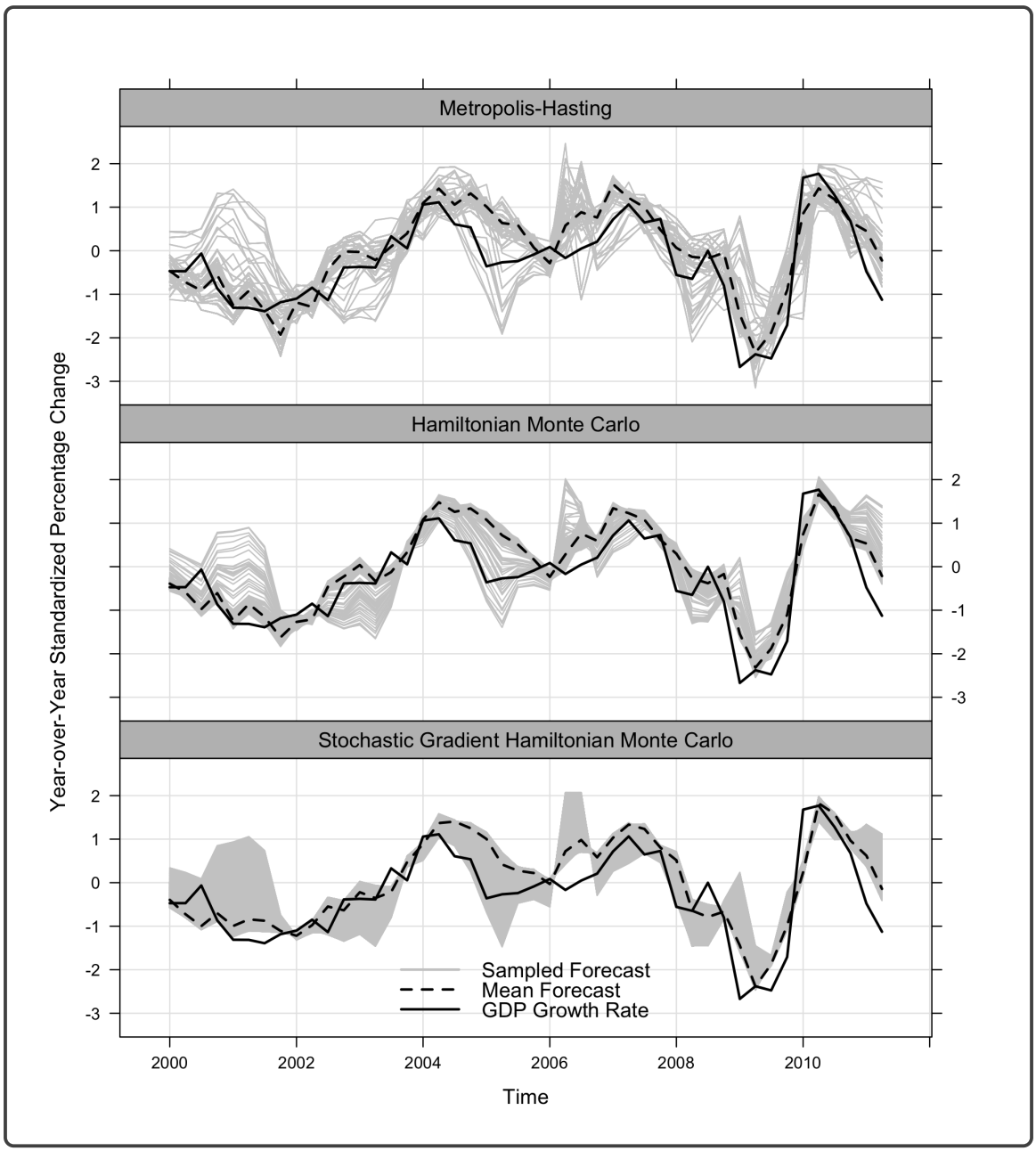


Figure 5.10: *Forecasts of BADL(1, 1) using different MCMCs under the Training Dataset Case 3 (γ = .0009 for 100000 Iterations).*

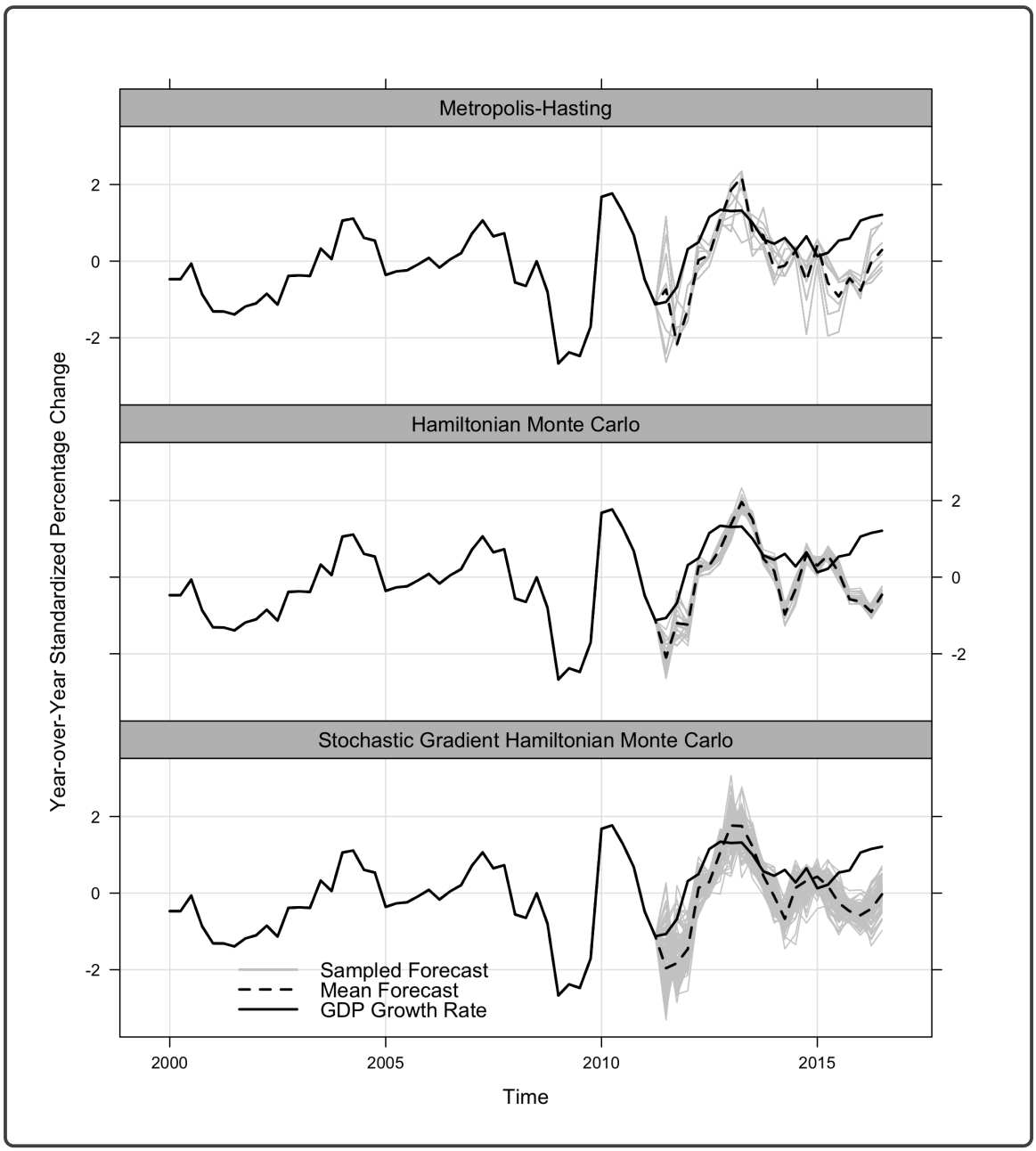


Figure 5.11: *Forecasts of BADL(1, 1) using different MCMCs under the Testing Dataset Case 1 (γ = .09 for 1000 Iterations).*

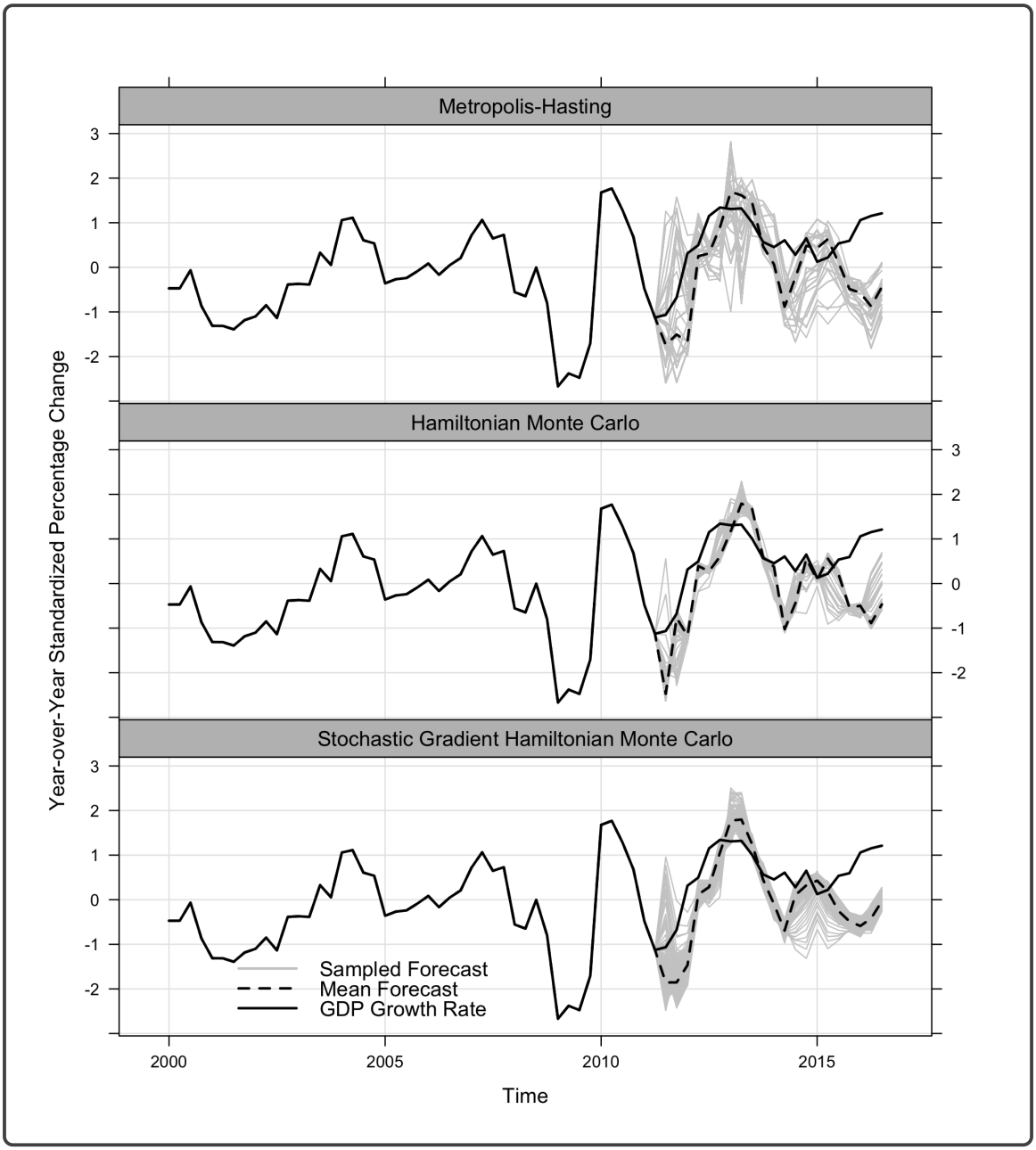


Figure 5.12: *Forecasts of BADL(1, 1) using different MCMCs under the Testing Dataset Case 2 (γ = .009 for 10000 Iterations).*

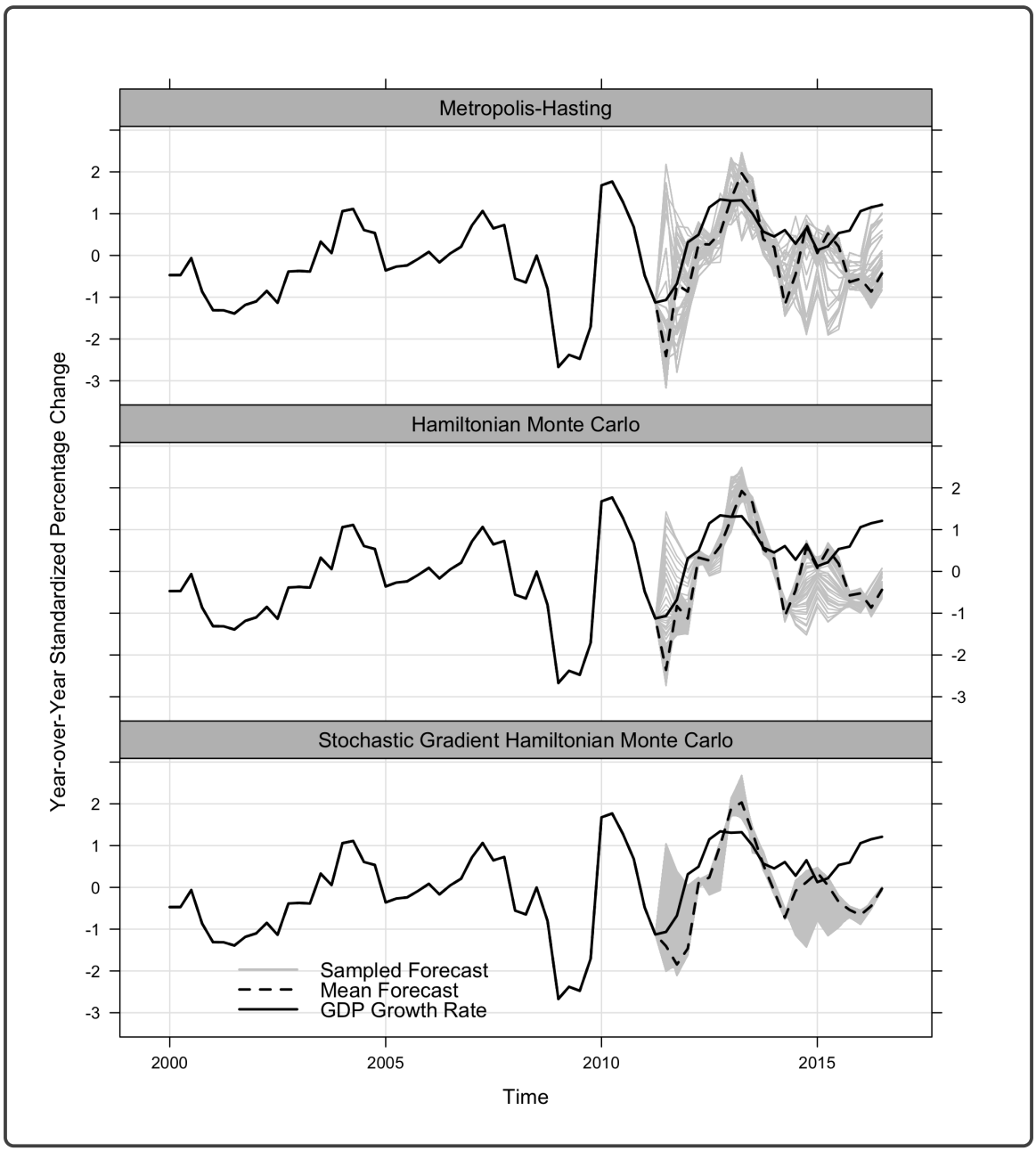


Figure 5.13: *Forecasts of BADL(1, 1) using different MCMCs under the Testing Dataset Case 3 (γ = .0009 for 100000 Iterations).*

Chapter 6

SUMMARY, CONCLUSION AND RECOMMENDATION

The findings of the study discussed in the previous chapter are summarized in Section 6.1. The conclusions are presented in Section 6.2, and the recommendations are provided in Section 6.3.

6.1 Summary

The main objective of this paper is the integration of the Stochastic Gradient Hamiltonian Monte Carlo (SGHMC) in estimating the parameters of the Bayesian Autoregressive Distributed Lag (BADL) model. In particular, two main propositions were derived for the theory of the proposed model, BADL-SGHMC, and these are: the posterior distribution of the weight vector of the model using standard Gaussian *a priori*, and the stochastic gradient of the potential energy. Further, to illustrate the application of the proposed model, the theoretical results were applied to forecasting the economic growth of the Philippines. The study considered a simple model of the BADL with order $p = q = 1$, and hence the following are the candidate Bayesian models fitted to the data: BADL(1, 1)-MH, BADL(1, 1)-HMC and BADL(1, 1)-SGHMC, where the first two models are estimated using Metropolis-Hasting and Hamiltonian Monte Carlo, respectively. The samples from the the markov chains were assessed using different statistical tests. The first test is the stationarity of the markov chains obtained by the three MCMCs (MH, HMC and SGHMC). The result suggests that the BADL(1, 1)-SGHMC performs well compared to BADL(1, 1)-MH and

BADL(1, 1)-HMC. The second test conducted assesses the convergence of the four markov chains using Gelman-Rubin, the statistics supported the initial findings from the trace plots that the chains converged to stationarity.

The discussions in the preceding chapter exclude the usage of the programming languages utilized for the MCMC simulation. As mentioned in the objective of the study, this paper uses two programming languages and these are R and Julia. In particular, the simulation of the MCMCs in forecasting the Philippine economic growth was done entirely in Julia, while the convergence tests were programmed in R. Moreover, Julia is introduced in this thesis as an alternative to R, since the former is a lot faster than the latter. Specifically, the 1.5 hour MH MCMC simulation in R is only 10.7 seconds in Julia, and since Julia has the advantage of using multiple processor, this elapsed time is expected to decrease if multiple processors were initialized. Finally, the author developed the first R and Julia packages for SGHMC, which is useful for exploring the proposed model using different datasets, and use SGHMC for other interesting models. The details are provided in Appendix C.

6.2 Conclusion

The proposed model, BADL(1, 1)-SGHMC, using multivariate standard Gaussian *a priori*, is a good alternative to BADL(1, 1)-MH and BADL(1, 1)-HMC. In particular, lower leapfrog step size is ideal for the proposed model even under 1,000 iterations only. Further, the training dataset used in modeling the economic growth of the Philippines uses 70% of the entire dataset. The unpartitioned dataset contains 67 total data points, and the first 46 observations accounts for the training dataset. Hence the remaining 21 observations are reserved for the testing dataset.

According to Chen et al. (2014), the central limit theorem (CLT) for the stochastic gradient noise of the potential energy is satisfied for at least 100 observations, which is far from the number of data points considered in the training dataset of this study. Interestingly, the SGHMC algorithm works well even for 46 observations only, this is evident on the performance of the BADL-SGHMC discussed in the precedings chapter compared to BADL-MH and BADL-HMC. The limitation of the data points to 67 observations depends heavily on the historical data of the indicators. Therefore, if at least 100 observations were used in the training dataset, the proposed BADL-SGHMC might perform even better compared to the contenders, and that would be part of the recommendation.

6.3 Recommendation

The main contribution of this paper is dedicated to the foundation of the BADL-SGHMC, hence the following are recommended for further study: explore higher orders of the BADL-SGHMC using at least 100 observations; set priors on the precision parameters of the model likelihood, which in this study is assumed to be known, fixed to constant α^{-1}; explore complicated BADL-SGHMC by setting priors on all hyperparameters; explore different values on the parameters of the SGHMC, especially for $\mathfrak{C}$ and $\mathfrak{A}$ which were considered as identity matrix for this study; consider other Stochastic Gradient MCMC algorithms, like Stochastic Gradient Langevin Monte Carlo (SGLMC) or Langevin Dynamics (SGLD) (*see* Welling & Teh, 2011), and Stochastic Gradient Fisher Scoring (*see* Ahn et al., 2012); and, obtain the prediction interval of the forecast by computing the predictive distribution for each items mentioned above.

References

Ahn, S., Korattikara, A., & Welling, M. (2012). Bayesian posterior sampling via stochastic gradient fisher scoring. In *Proceedings of the 29th international conference on machine learning*.

Bayes, T. (1763). An essay towards solving a problem in the doctrine of chances. *Philosophical Transactions*, *53*, 370-418. Retrieved from http://www.jstor.org/stable/105741

Bezanson, J., Edelman, A., Karpinski, S., & Shah, V. B. (2017). Julia: A fresh approach to numerical computing. *SIAM Review*, *59*(1), 65-98. Retrieved from http://dx.doi.org/10.1137/141000671 doi: 10.1137/141000671

Bishop, C. M. (2006). *Pattern recognition and machine learning (information science and statistics)*. Secaucus, NJ, USA: Springer-Verlag New York, Inc.

Buss, G. (2010). Economic forecasts with bayesian autoregressive distributed lag model: Choosing optimal prior in economic downturn. *Scientific Journal of Riga Technical University*, *42*(2), 100 - 105.

Casella, G., & Berger, R. L. (2002). *Statistical inference*. Pacific Grove (Calif.): Brooks Cole.

Chen, T., Fox, E., & Guestrin, C. (2014). Stochastic gradient hamiltonian monte carlo. In *Proceedings of the 31st international conference on machine learning* (Vol. 32). Beijing, China.

Cowles, M. K., & Carlin, B. P. (1996). Markov chain monte carlo convergence diagnostics: A comparative review. *Journal of the American Statistical Association*, *91*(434), 883-904. Retrieved from http://www.jstor.org/stable/2291683

Duane, S., Kennedy, A., Pendleton, B. J., & Roweth, D. (1987). Hybrid monte carlo. *Physics Letters B*, *195*(2), 216 - 222.

Hastings, W. K. (1970). Monte carlo sampling methods using markov chains and their applications. *Biometrika*, *57*(1), 97-109. Retrieved from http://www.jstor.org/stable/2334940

Haykin, S. (1998). *Neural networks: A comprehensive foundation* (2nd ed.). Upper Saddle River, NJ, USA: Prentice Hall PTR.

International Monetary Fund. (2005). *Balance of payments manual* (5th ed.). Retrieved May 26, 2017, from International Monetary Fund Web site: http://www.imf.org/external/pubs/cat/longres.aspx?sk=157.0.

Kumar, D. N., & Maity, R. (2008). Bayesian dynamic modelling for nonstationary hydroclimatic time series forecasting along with uncertainty quantification. *Hydrological Processes*, *22*(17), 3488–3499. Retrieved from http://dx.doi.org/10.1002/hyp.6951 doi: 10.1002/hyp.6951

Laplace, P. S. (1986, 08). Memoir on the probability of the causes of events. *Statist. Sci.*, *1*(3), 364–378. Retrieved from http://dx.doi.org/10.1214/ss/1177013621 doi: 10.1214/ss/1177013621

Mapa, C. D. S., Del Prado, D. G., Poliquit, I. A., & Asaad, A. B. (2016). Enhancement of the composite leading economic indicator system of the philippines. In *13th national convention on statistics*.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., & Teller, E. (1953). Equation of state calculations by fast computing machines. *The Journal*

of Chemical Physics, *21*(6), 1087-1092. Retrieved from http://dx.doi.org/10.1063/1.1699114 doi: 10.1063/1.1699114

Neal, R. M. (1996). *Bayesian learning for neural networks*. Secaucus, NJ, USA: Springer-Verlag New York, Inc.

Neal, R. M. (2011). Mcmc using hamiltonian dynamics. In G. L. J. Steve Brooks Andrew Gelman & X.-L. Meng (Eds.), *Handbook of markov chain monte carlo* (p. 113-162). Chapman and Hall/CRC.

Ravines, R. R., Schmidt, A. M., & Migon, H. S. (2006). Revisiting distributed lag models through a bayesian perspective. *Applied Stochastic Models in Business and Industry*, *22*(2), 193–210. Retrieved from http://dx.doi.org/10.1002/asmb.628 doi: 10.1002/asmb.628

Robert, C., & Casella, G. (2010). *Introducing monte carlo methods with r*. Springer-Verlag New York.

Welling, M., & Teh, Y. W. (2011). Bayesian learning via stochastic langevin dynamics. In *Proceedings of the 28th international conference on machine learning*.

Welty, L. J., Peng, R. D., Zeger, S. L., & Dominici, F. (2009). Bayesian distributed lag models: Estimating effects of particulate matter air pollution on daily mortality. *Biometrics*, *65*(1).

Appendix A

Statistical Figures

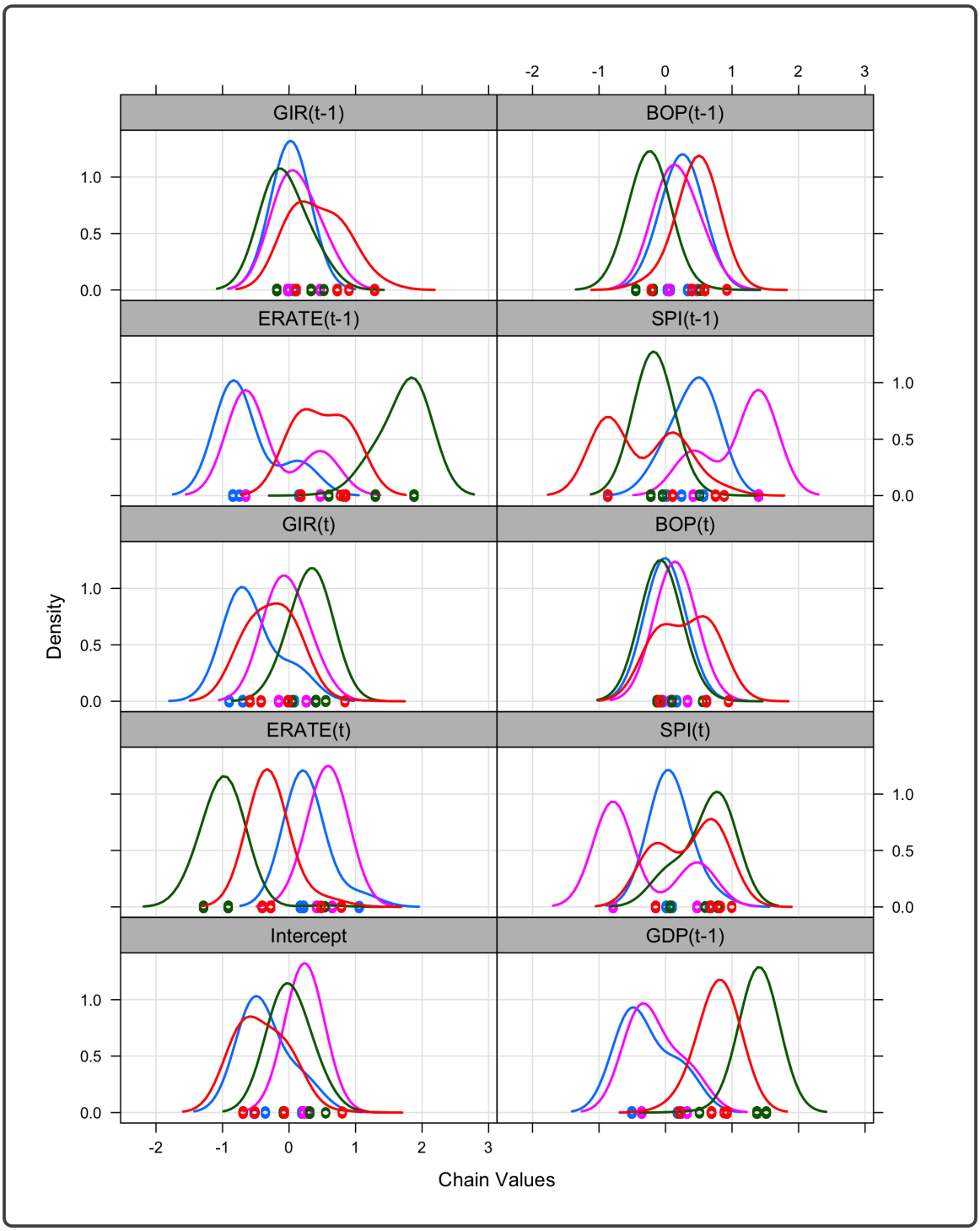


Figure A.1: *Kernel Density Estimate of the Unfiltered Chains using MH Case 1 (γ = .09 for 1000 Iterations).*

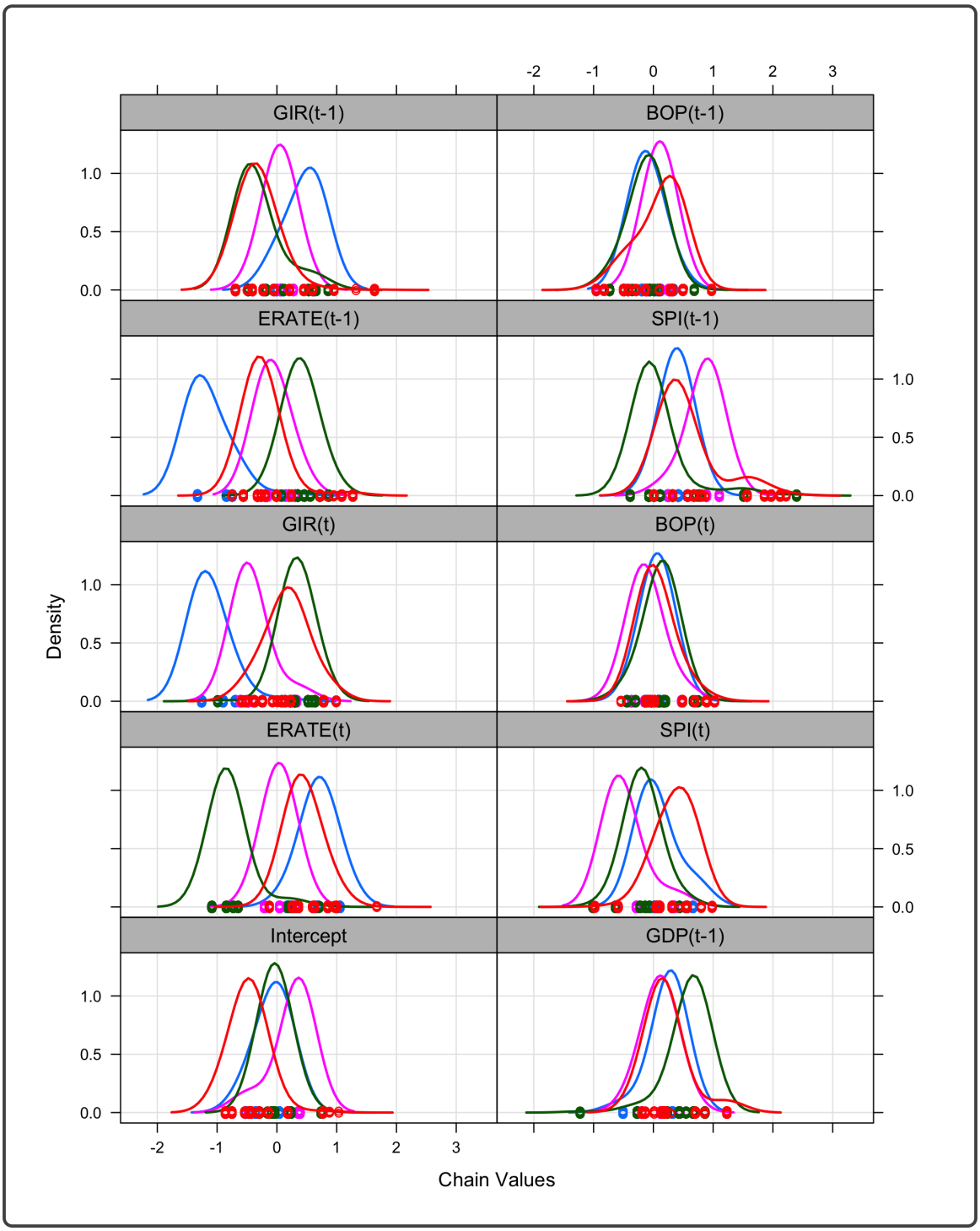


Figure A.2: *Kernel Density Estimate of the Unfiltered Chains using MH Case 2 (γ = .009 for 10000 Iterations).*

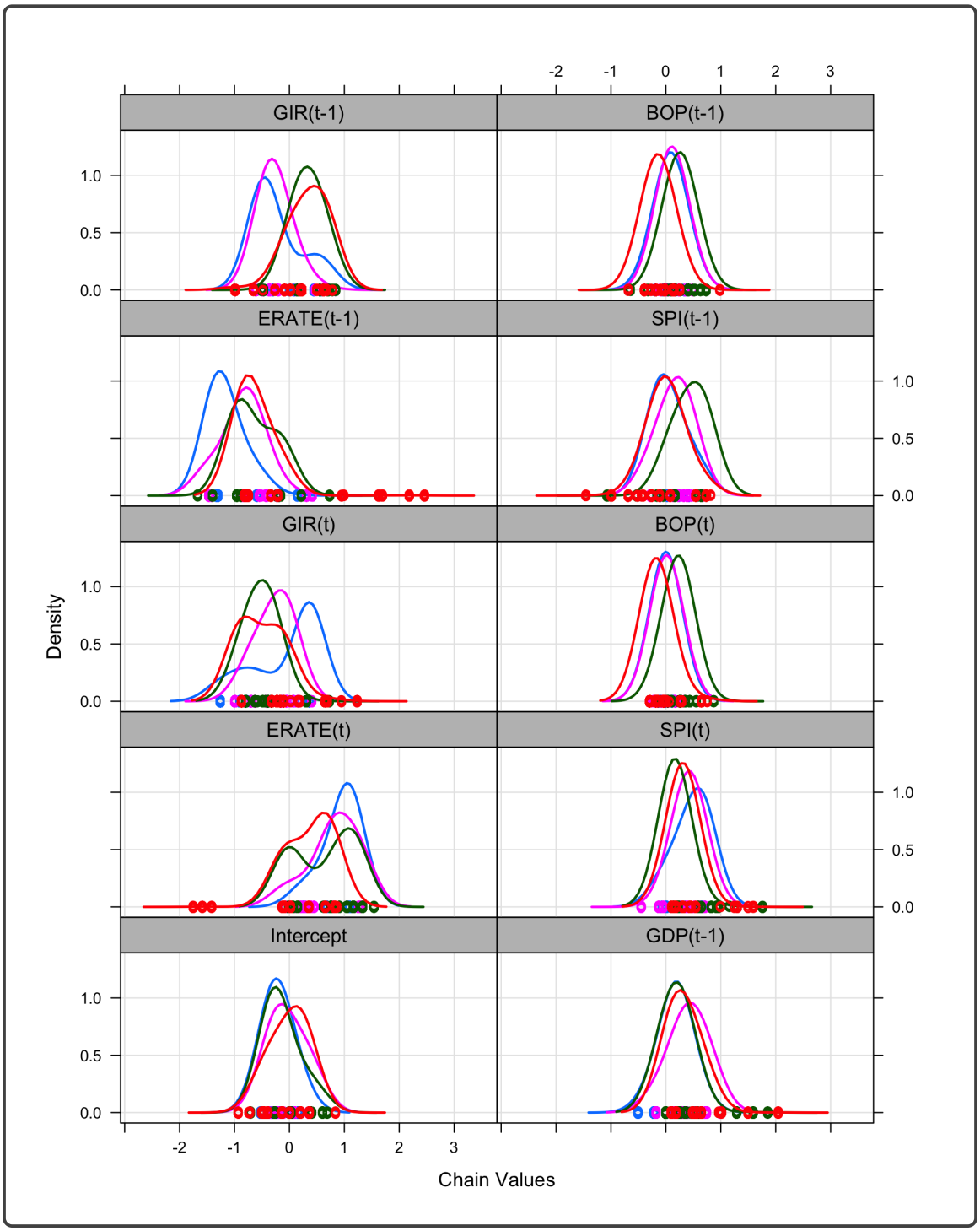


Figure A.3: *Kernel Density Estimate of the Unfiltered Chains using MH Case 3 (γ = .0009 for 100000 Iterations).*

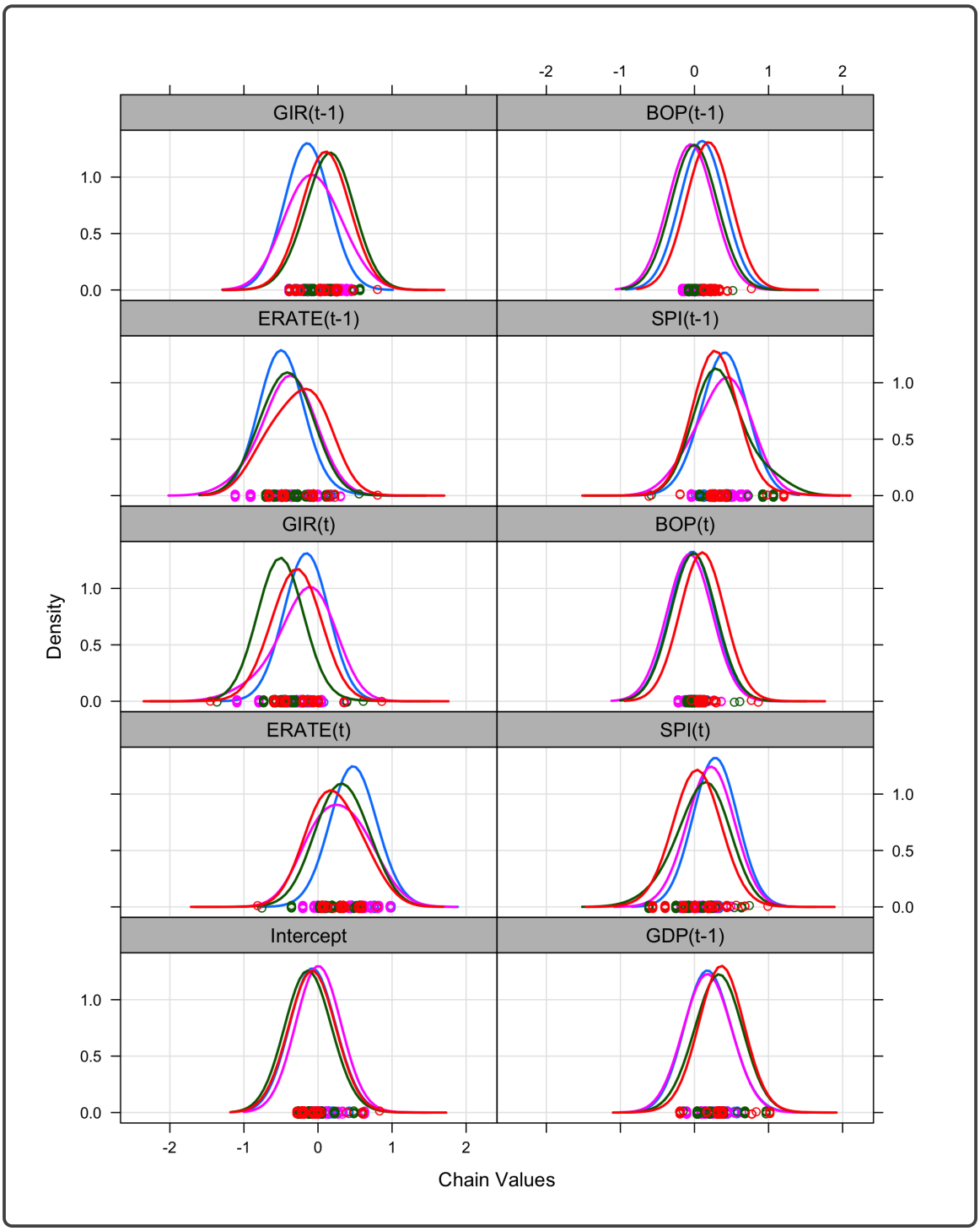


Figure A.4: *Kernel Density Estimate of the Unfiltered Chains using HMC Case 1 (γ = .09 for 1000 Iterations).*

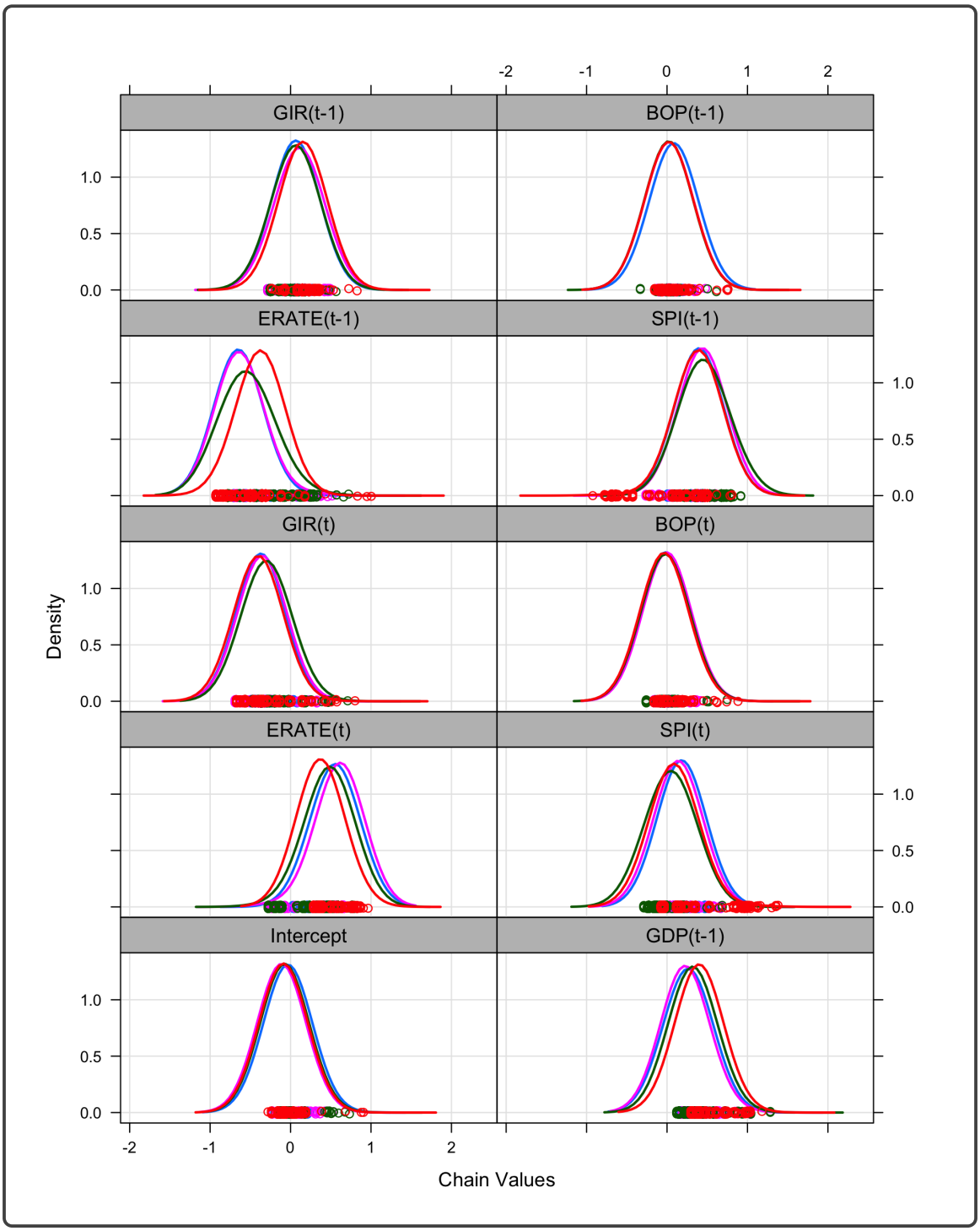


Figure A.5: *Kernel Density Estimate of the Unfiltered Chains using HMC Case 2 ($\gamma = .009$ for 10000 Iterations).*

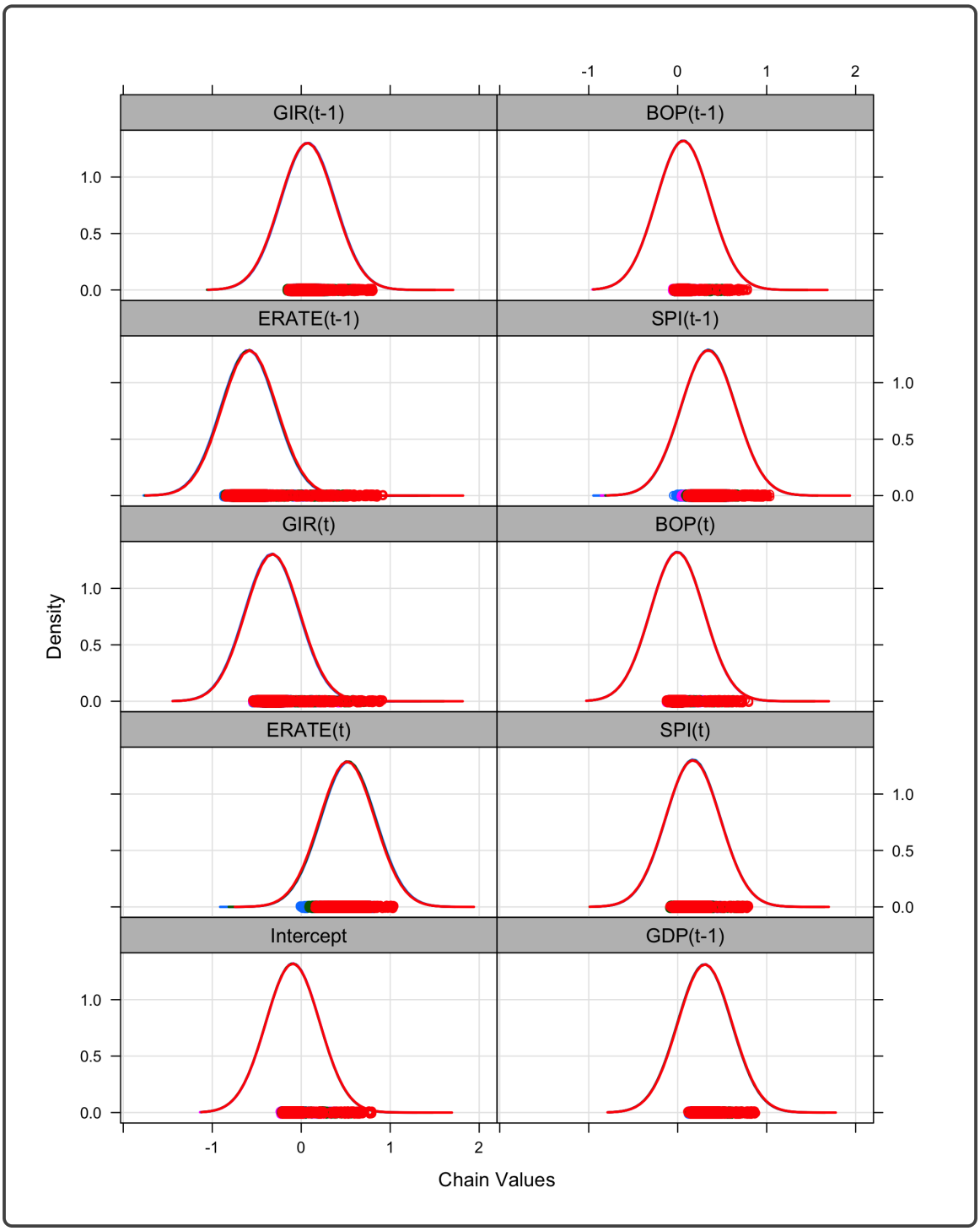


Figure A.6: *Kernel Density Estimate of the Unfiltered Chains using HMC Case 3 (γ = .0009 for 100000 Iterations).*

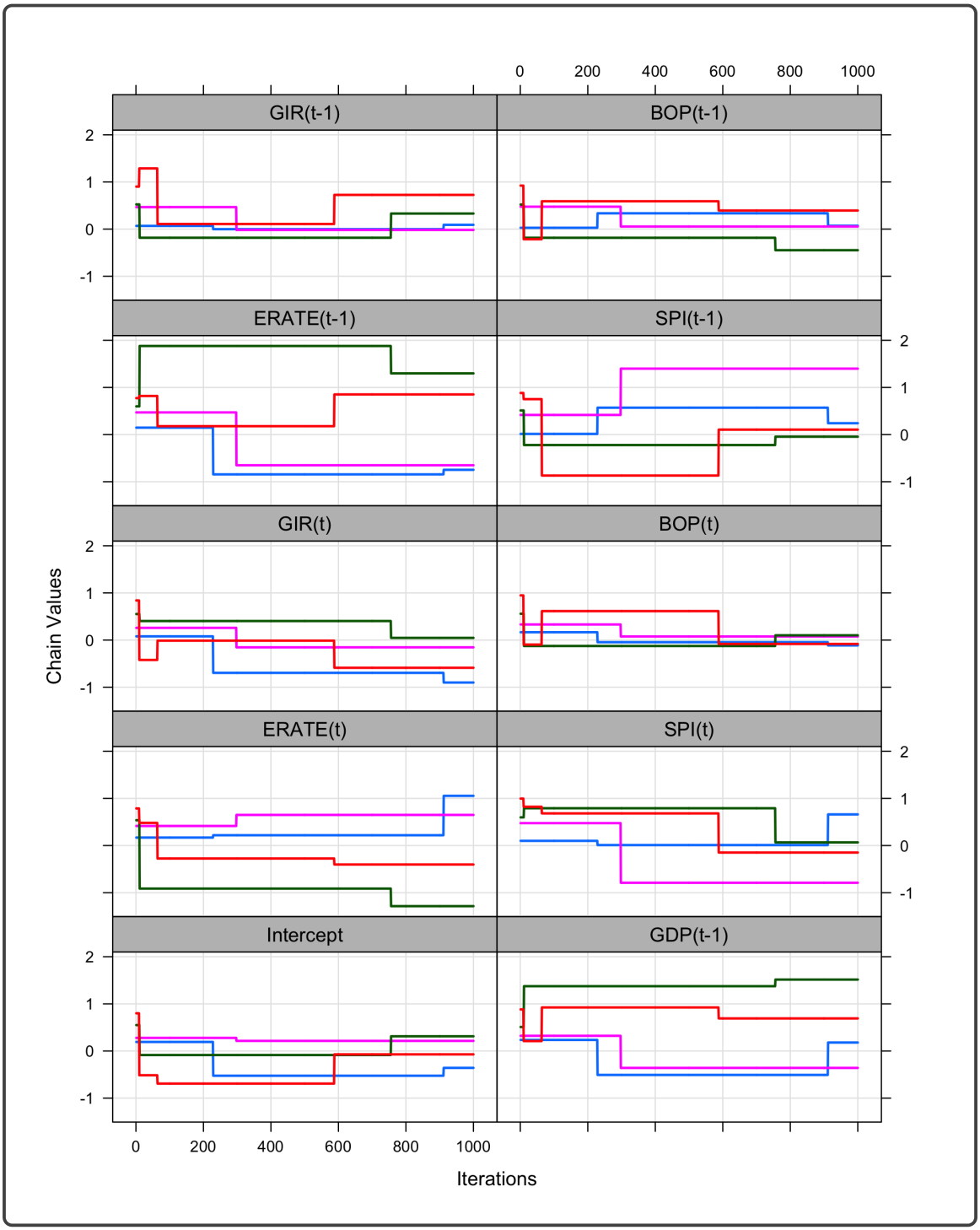


Figure A.7: *Unfiltered Traces of the Chains using MH Case 1 ($\gamma = .09$ for 1000 Iterations).*

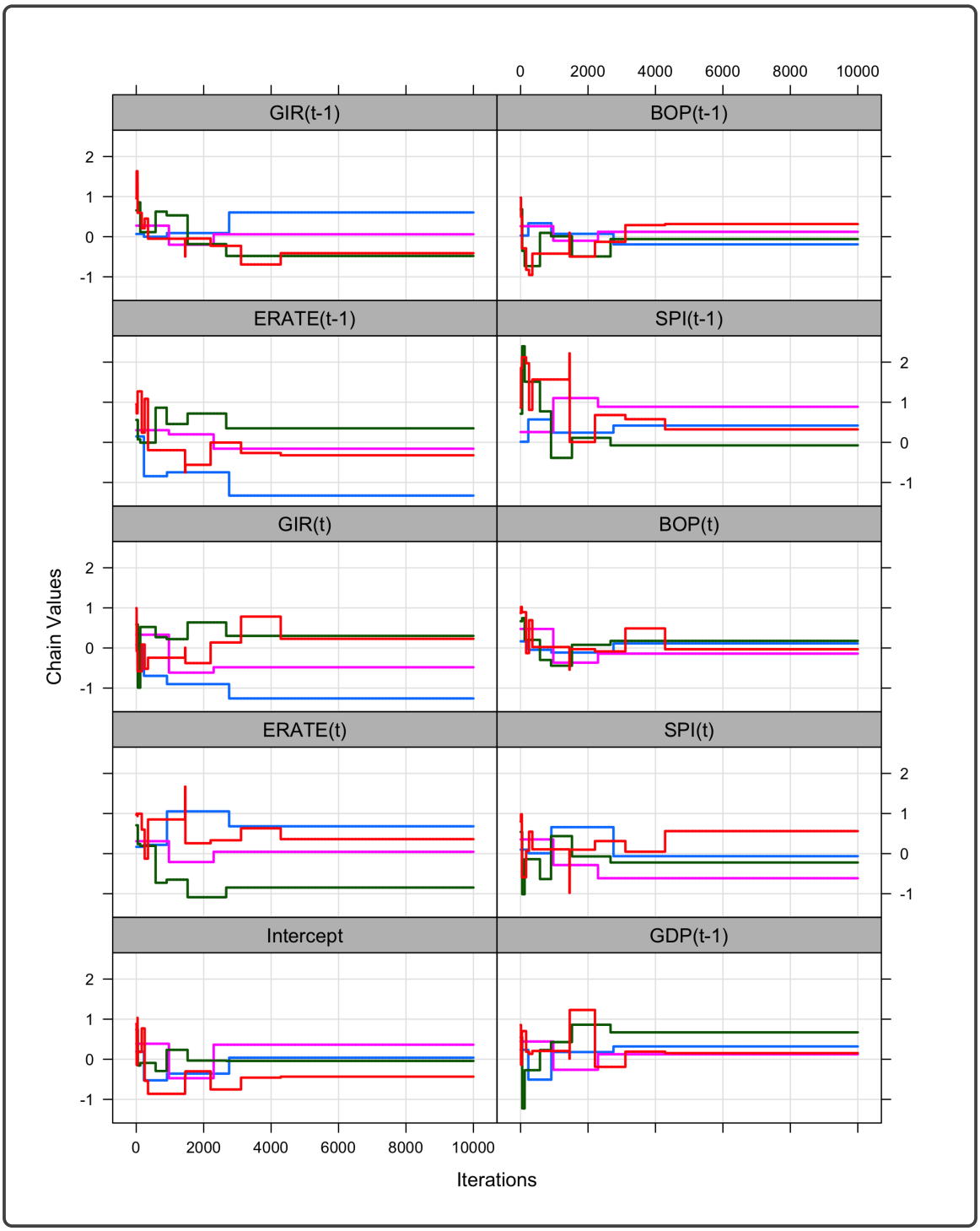


Figure A.8: *Unfiltered Traces of the Chains using MH Case 2 (γ = .009 for 10000 Iterations).*

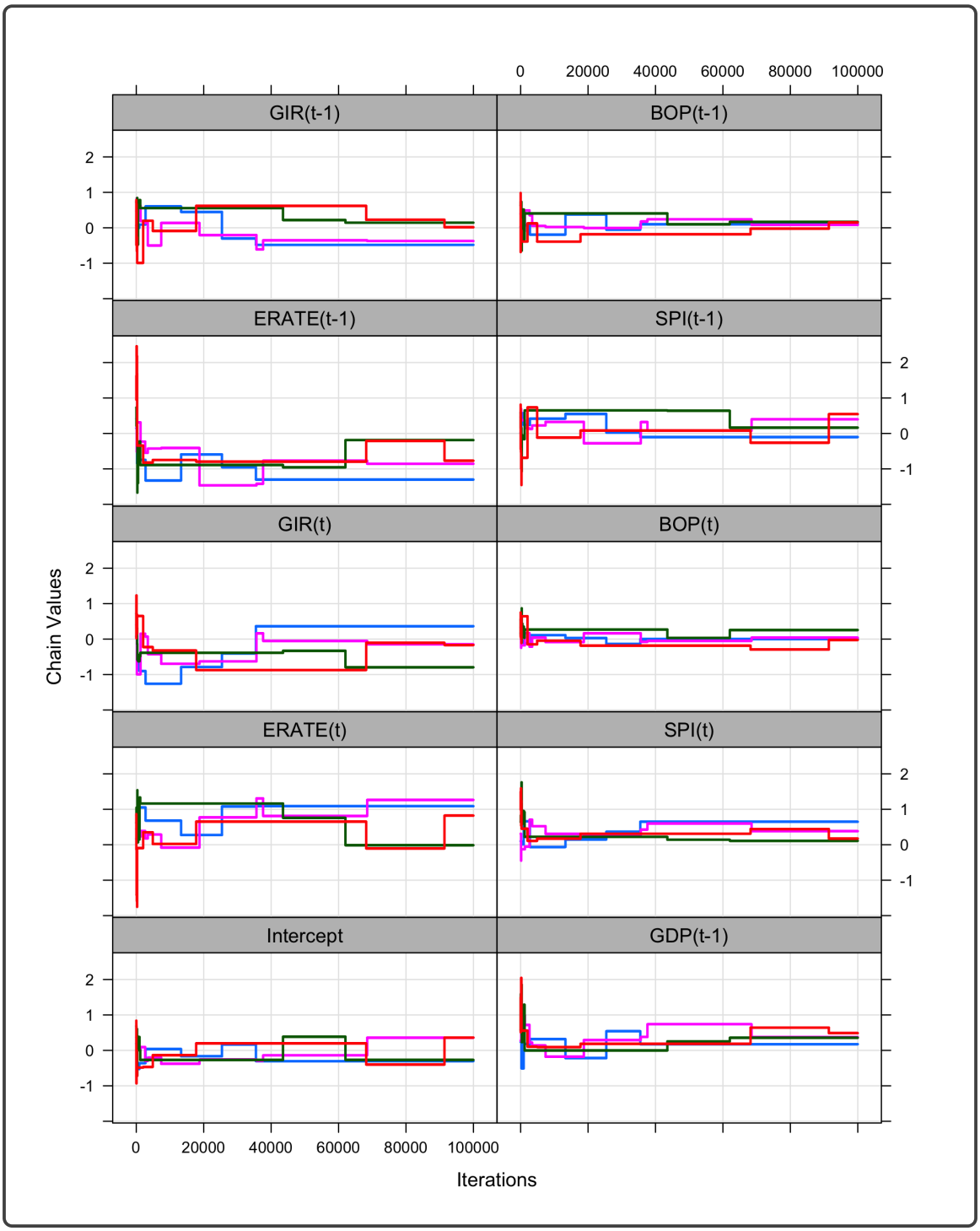


Figure A.9: *Unfiltered Traces of the Chains using MH Case 3 (γ = .0009 for 100000 Iterations).*

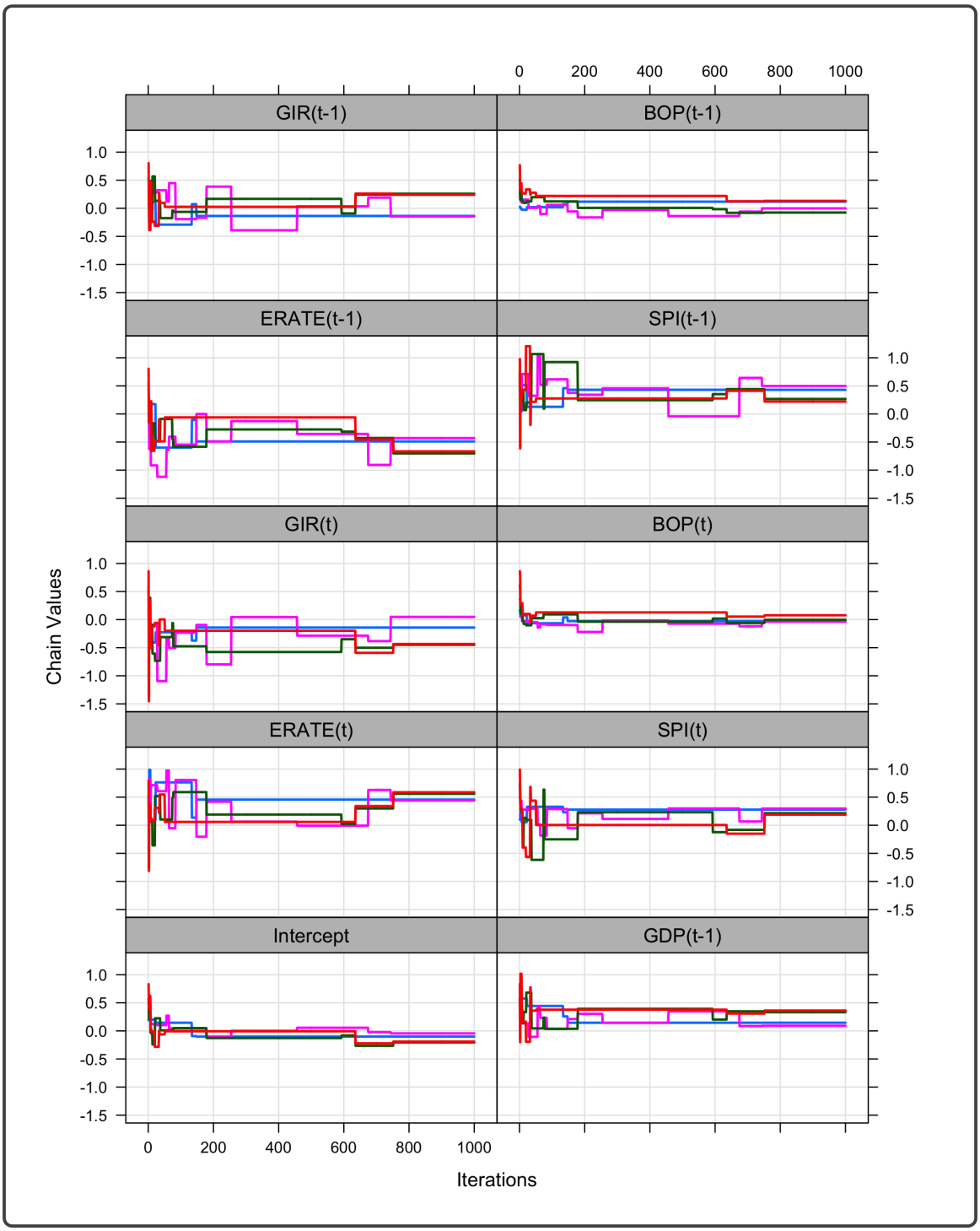


Figure A.10: *Unfiltered Traces of the Chains using HMC Case 1 (γ = .09 for 1000 Iterations).*

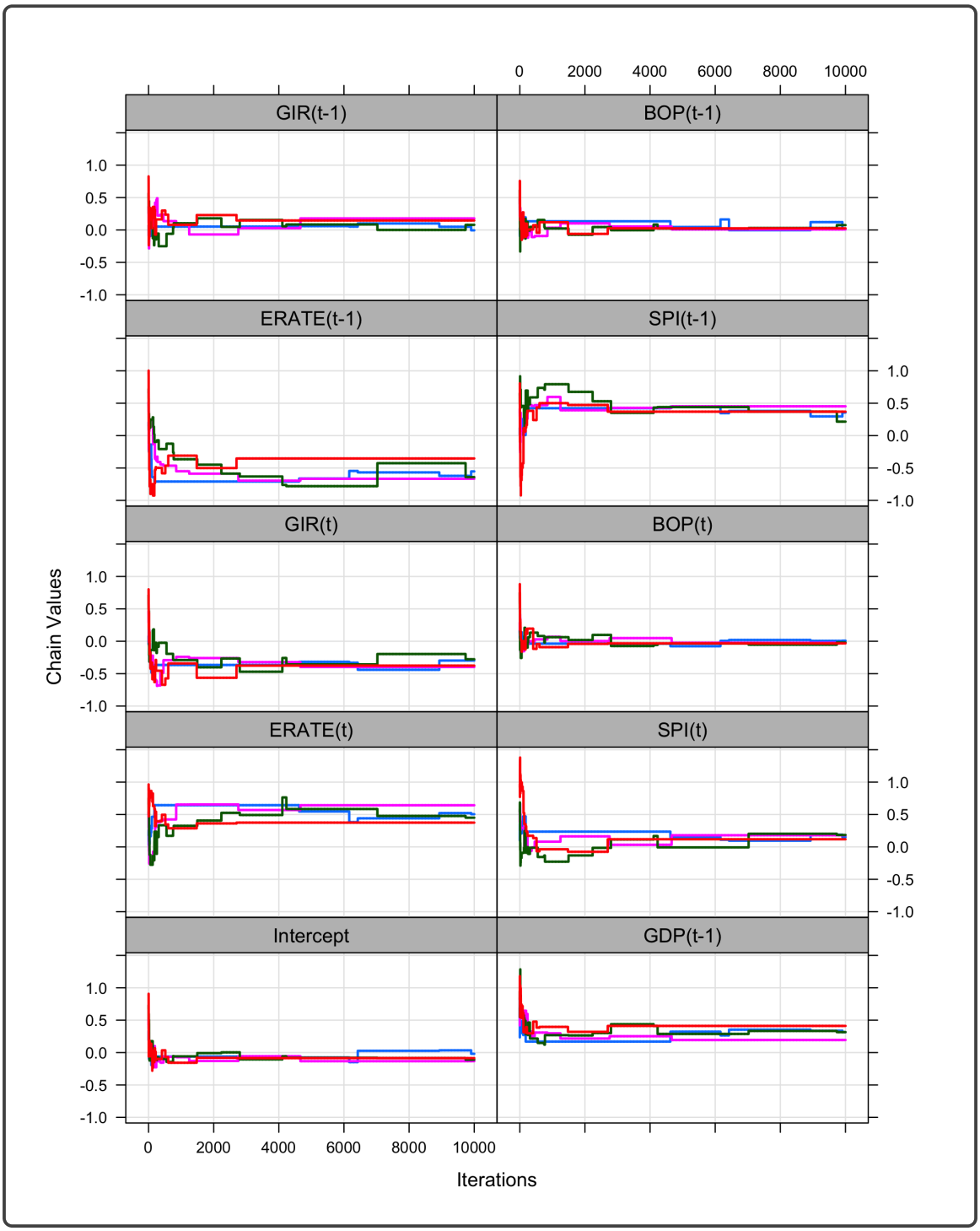


Figure A.11: *Unfiltered Traces of the Chains using HMC Case 2 (γ = .009 for 10000 Iterations).*

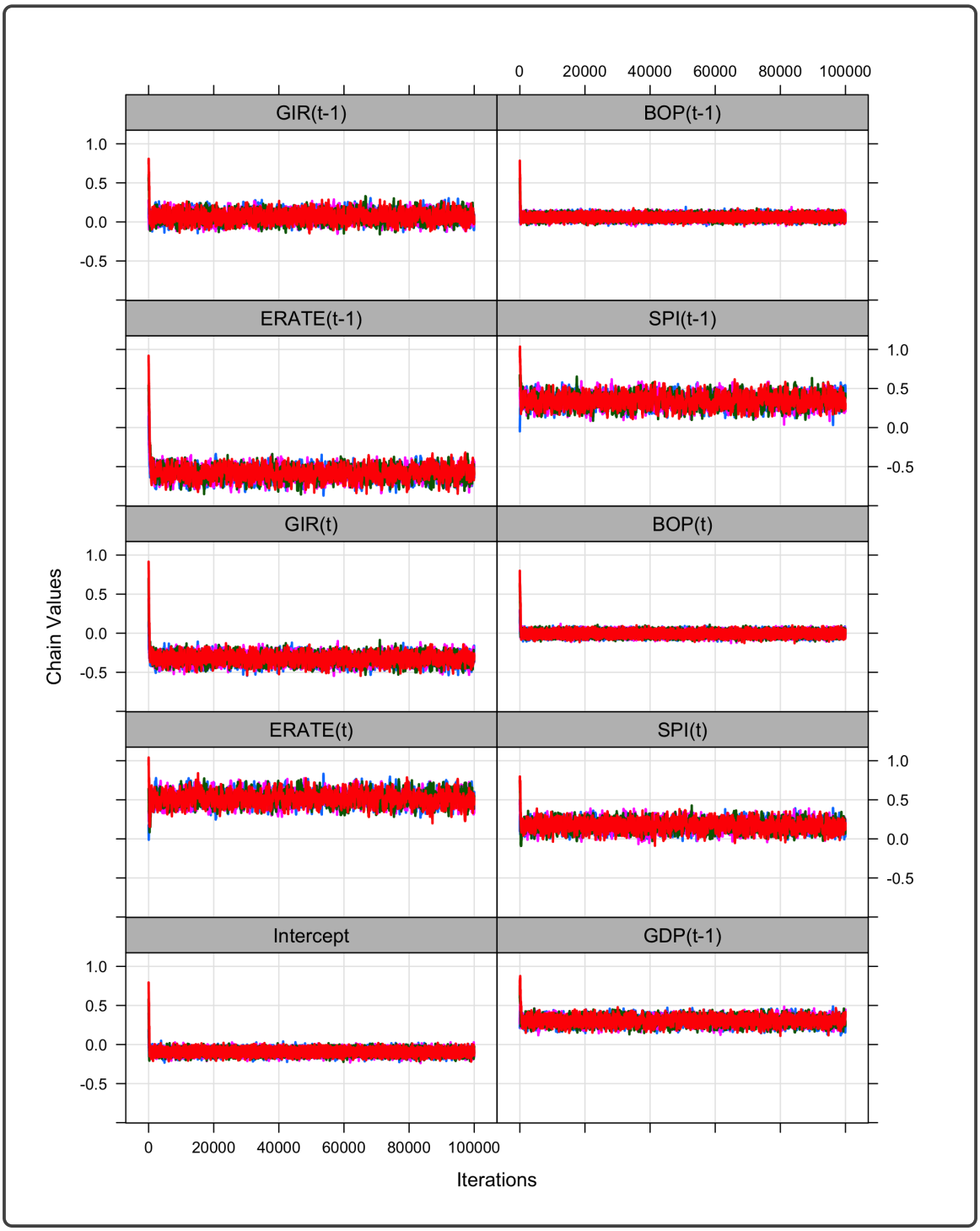


Figure A.12: *Unfiltered Traces of the Chains using HMC Case 3 (γ = .0009 for 100000 Iterations).*

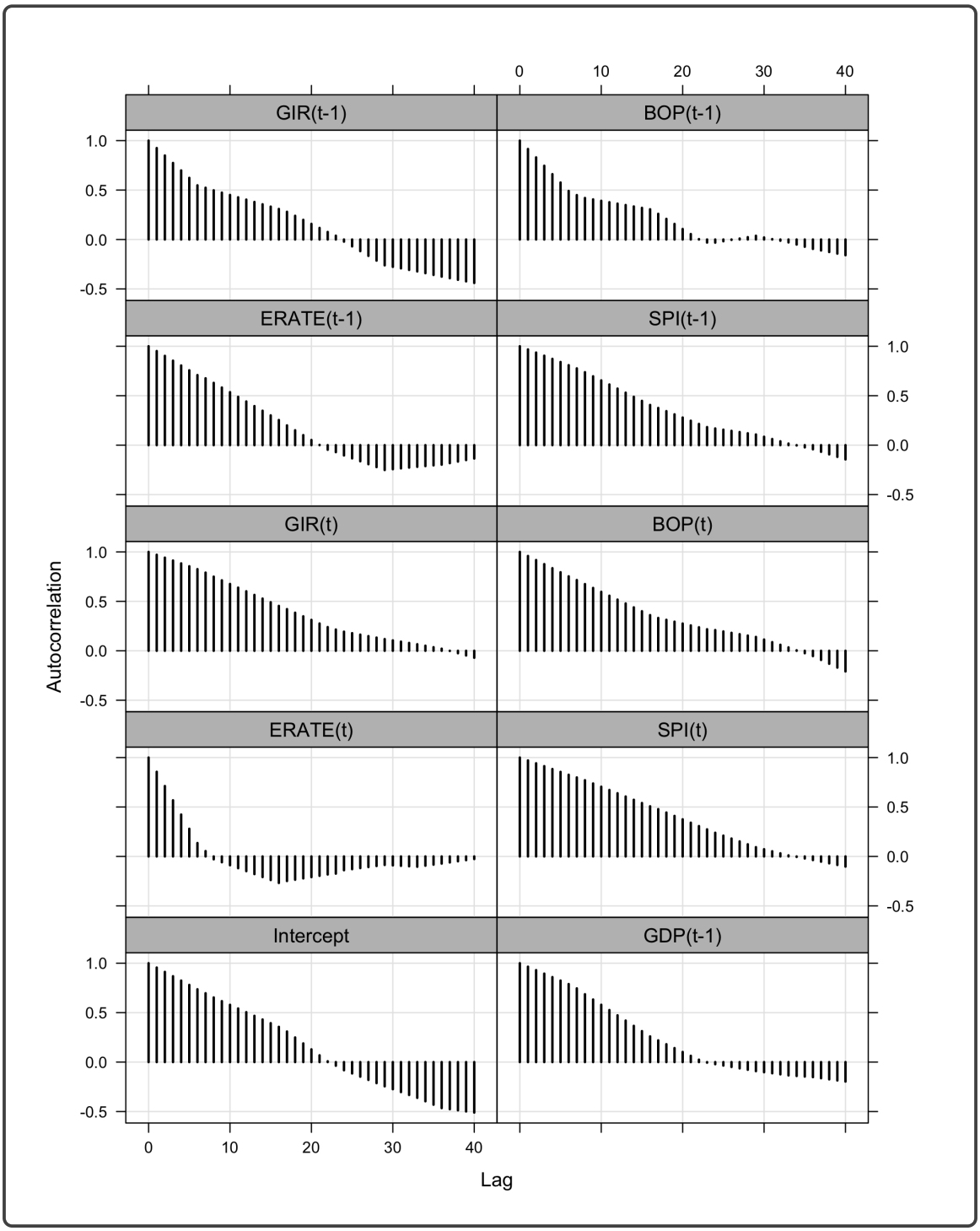


Figure A.13: *Filtered Mean Chain Autocorrelation using MH Case 1 (γ = .09 for 1000 Iterations).*

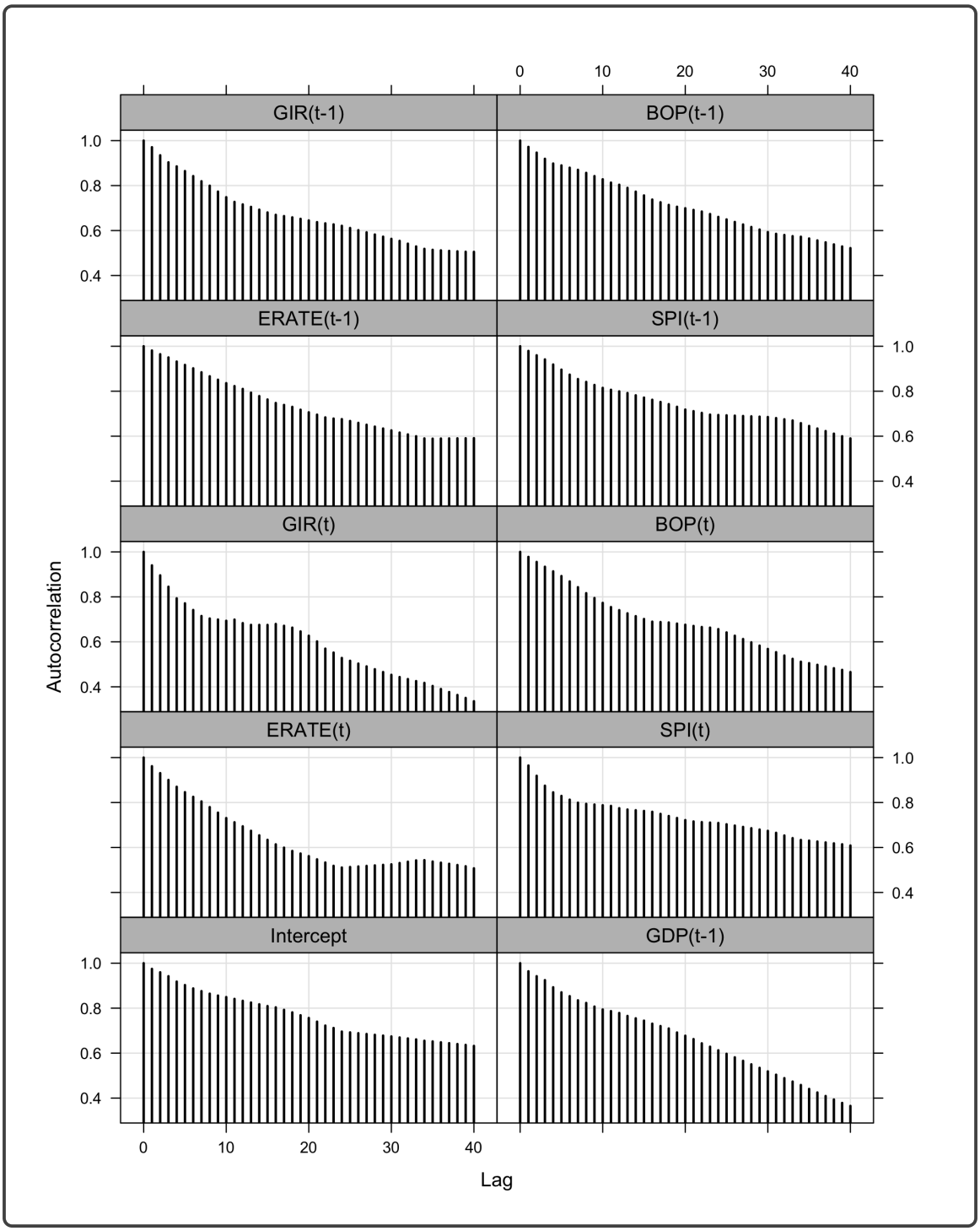


Figure A.14: *Filtered Mean Chain Autocorrelation using MH Case 2 (γ = .009 for 10000 Iterations).*

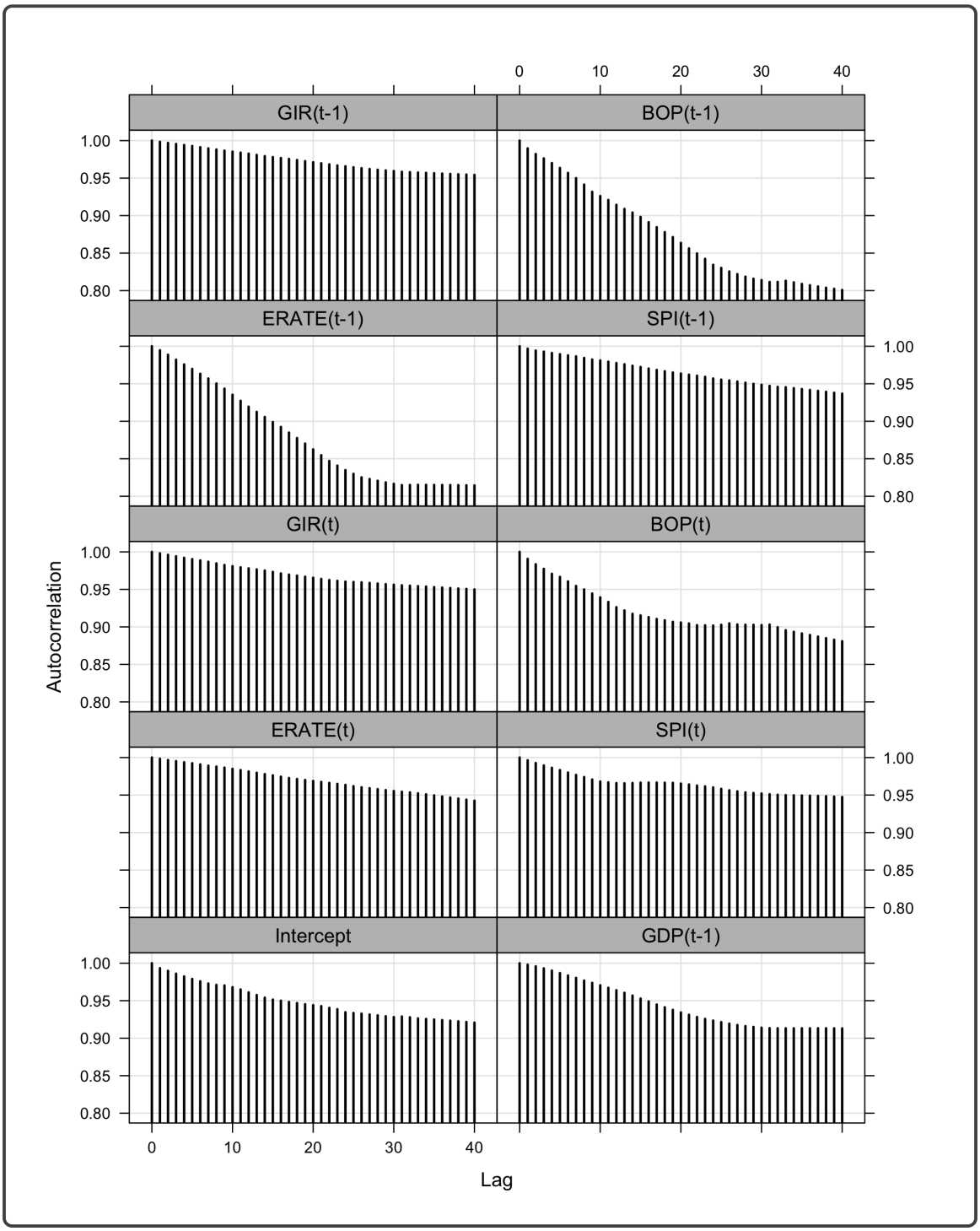


Figure A.15: *Filtered Mean Chain Autocorrelation using MH Case 3 (γ = .0009 for 100000 Iterations).*

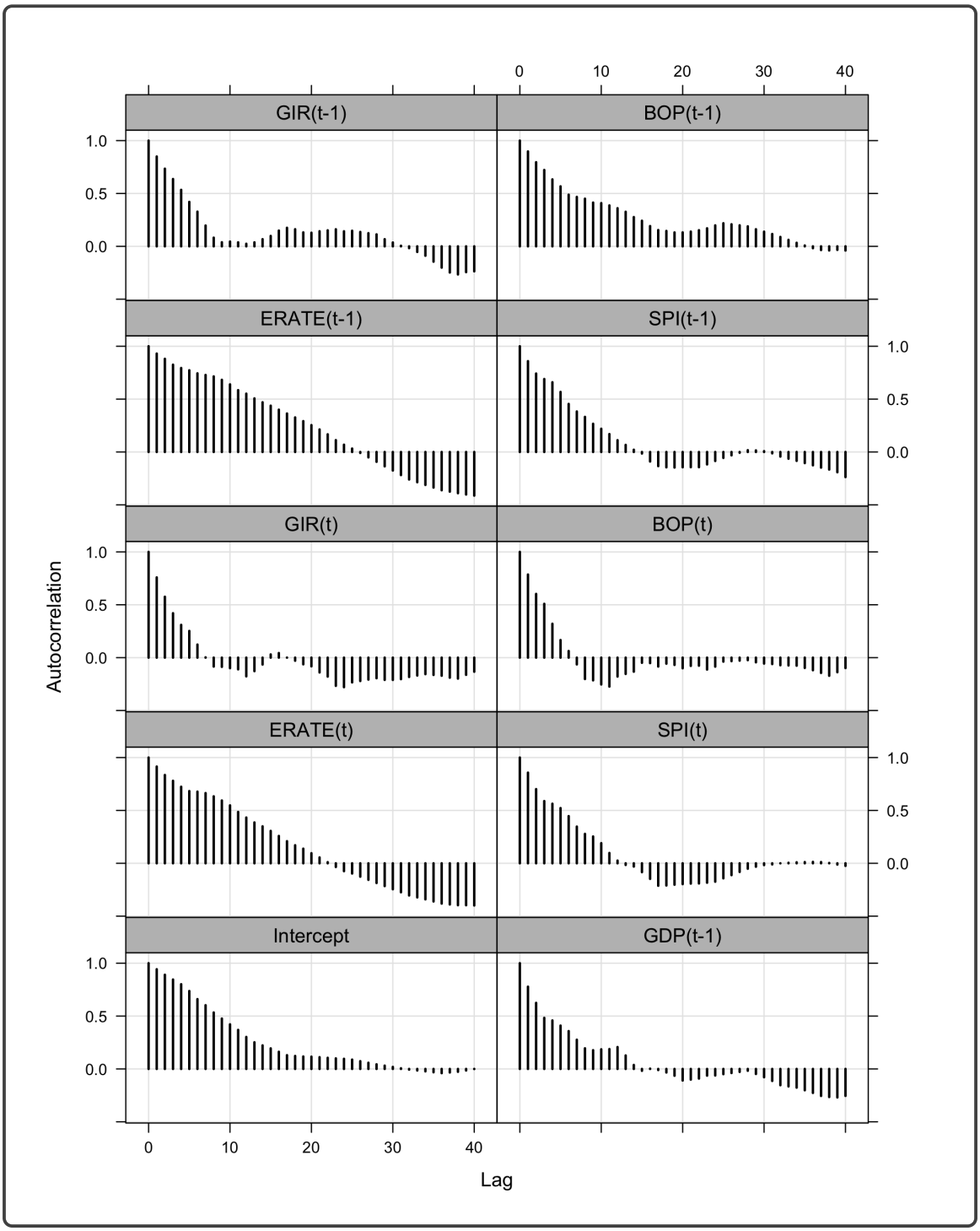


Figure A.16: *Filtered Mean Chain Autocorrelation using HMC Case 1 (γ = .09 for 1000 Iterations).*

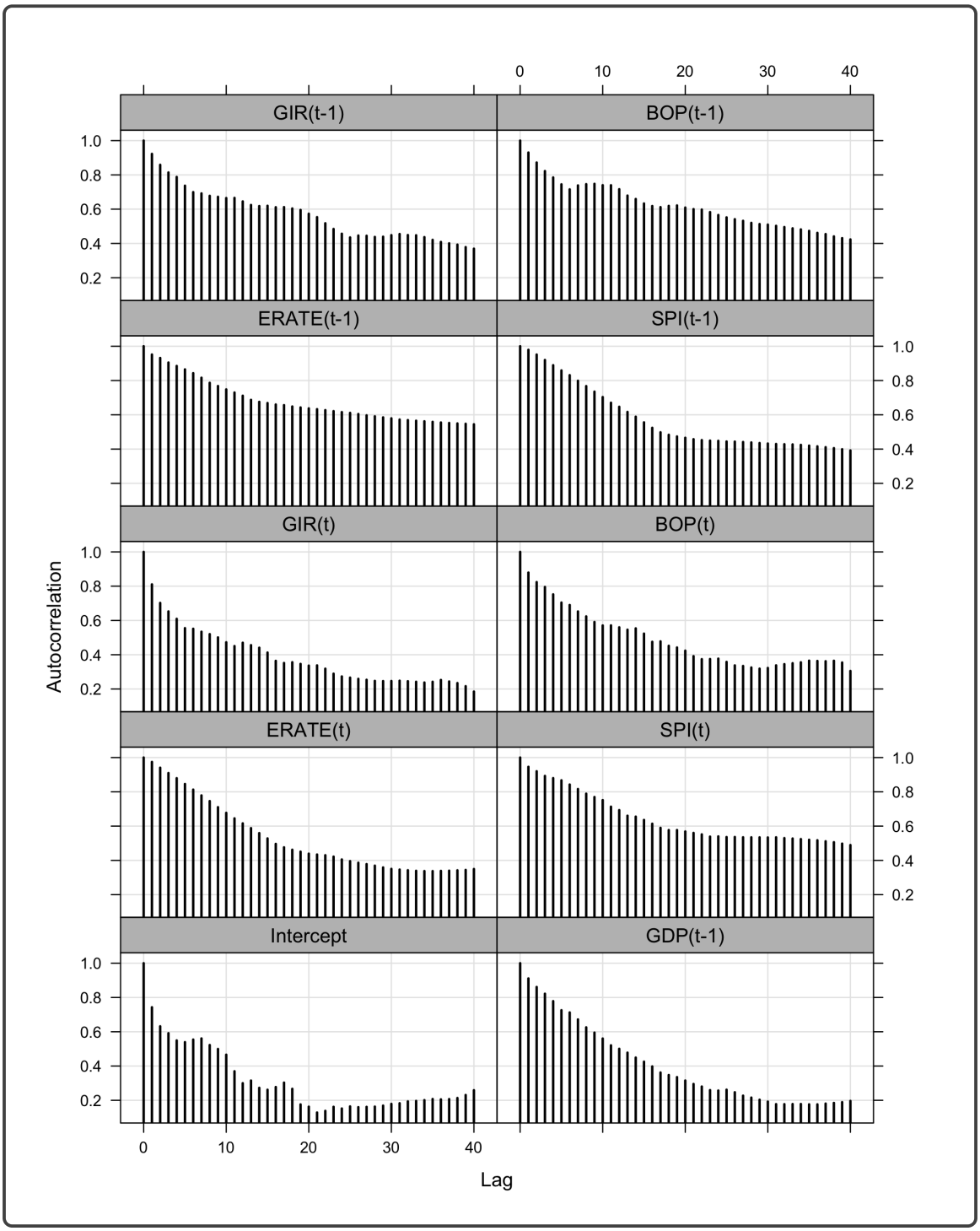


Figure A.17: *Filtered Mean Chain Autocorrelation using HMC Case 2 ($\gamma = .009$ for 10000 Iterations).*

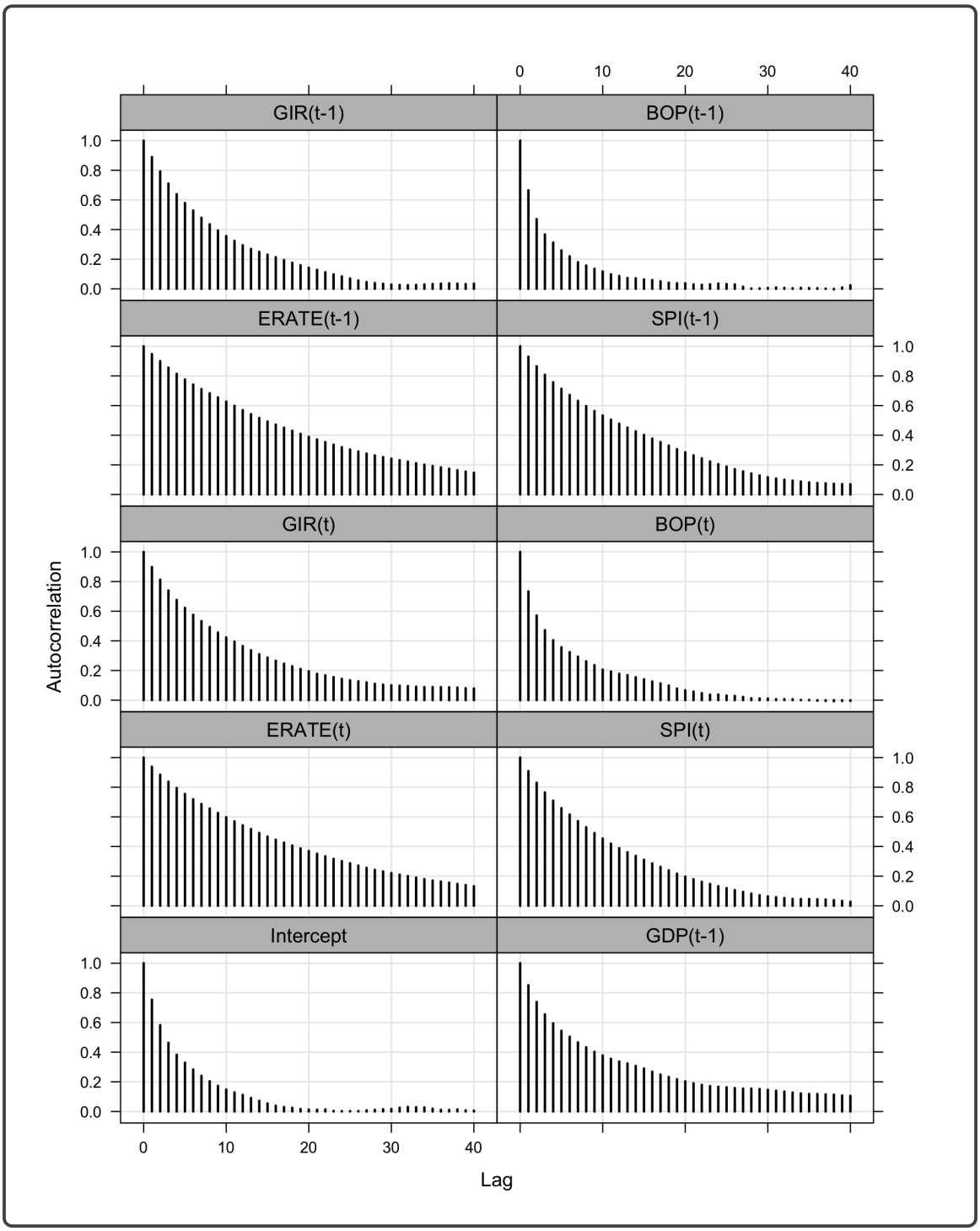


Figure A.18: *Filtered Mean Chain Autocorrelation using HMC Case 3 (γ = .0009 for 100000 Iterations).*

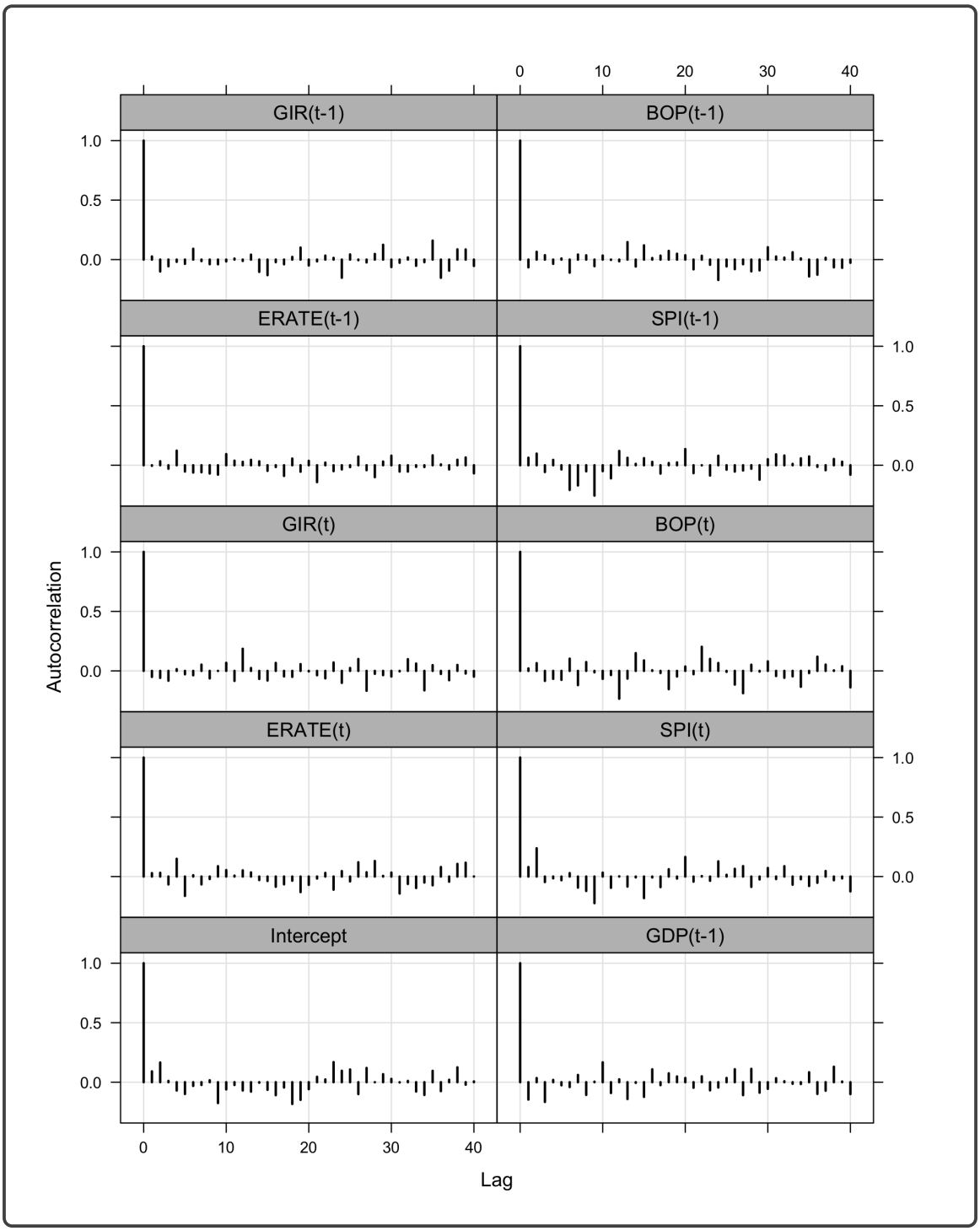


Figure A.19: *Filtered Mean Chain Autocorrelation using SGHMC Case 1 (γ = .09 for 1000 Iterations).*

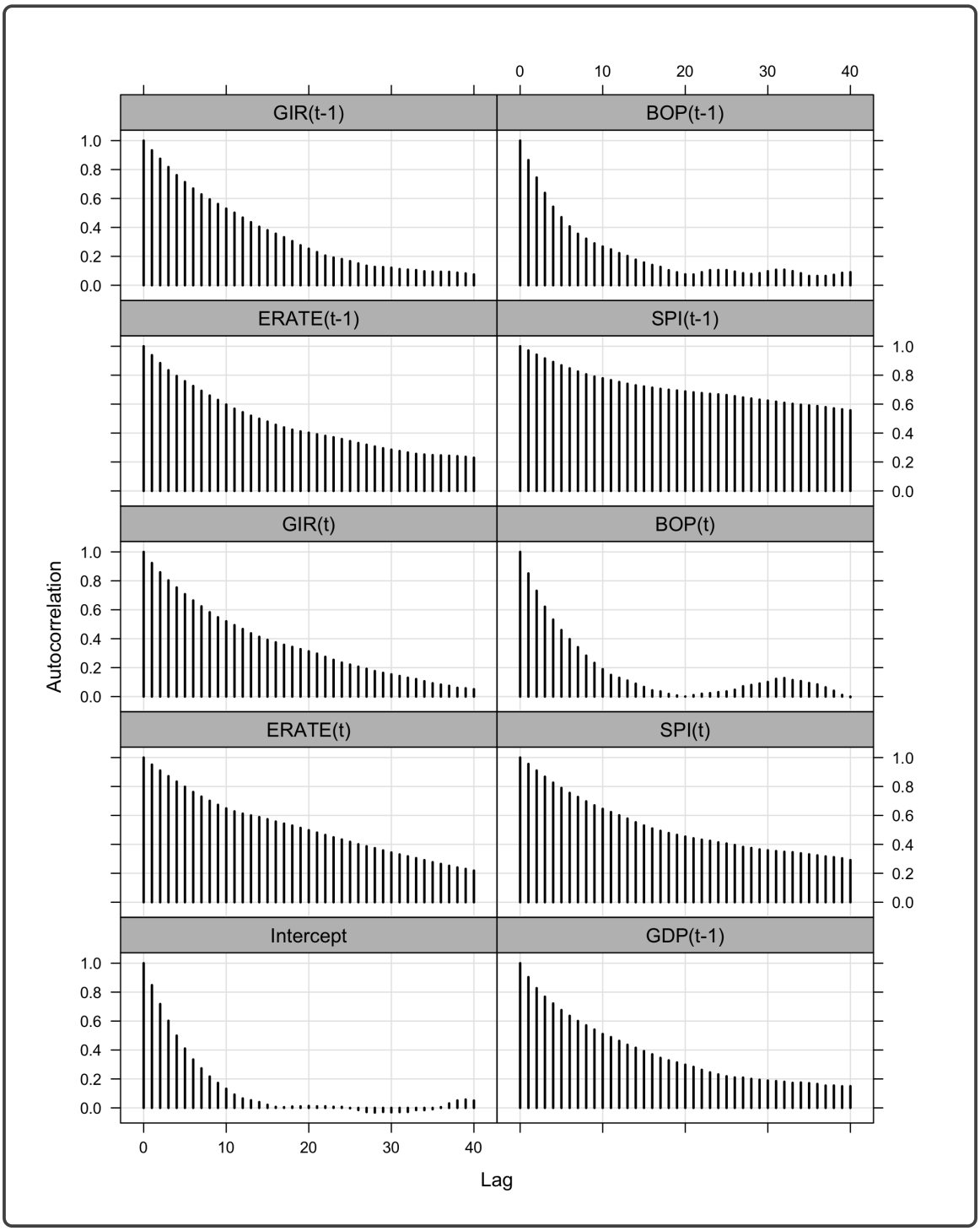


Figure A.20: *Filtered Mean Chain Autocorrelation using SGHMC Case 2 (γ = .009 for 10000 Iterations).*

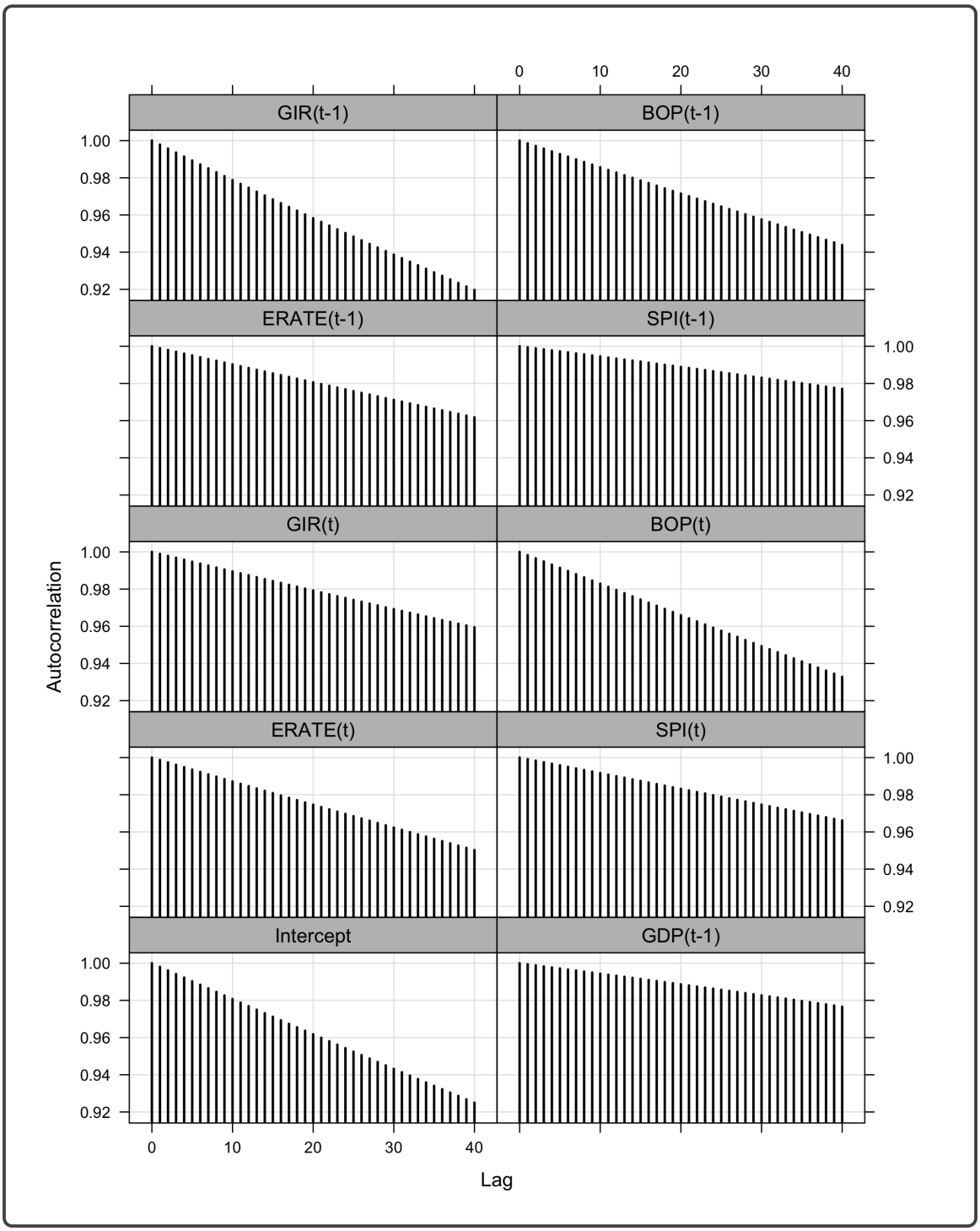


Figure A.21: *Filtered Mean Chain Autocorrelation using SGHMC Case 3 (γ = .0009 for 100000 Iterations).*

Appendix B

Statistical Tables

	Parameters	Stationarity Test			Halfwidth Test		
		Status	Start	p-value	Status	Mean	Halfwidth
1st Case	w_0	✓	1	0.370	✗	-0.127	0.123
	GDP($t-1$)	✓	201	0.354	✗	0.385	0.085
	ERATE(t)	✓	1	0.832	✗	-0.102	0.051
	SPI(t)	✓	301	0.106	✗	0.008	0.249
	GIR(t)	✓	301	0.082	✗	-0.238	0.180
	BOP(t)	✓	301	0.052	✗	0.045	0.102
	ERATE($t-1$)	✓	201	0.248	✗	0.228	0.101
	SPI($t-1$)	✓	401	0.114	✗	0.393	0.178
	GIR($t-1$)	✓	1	0.433	✗	0.133	0.127
	BOP($t-1$)	✓	1	0.102	✗	0.162	0.056
2nd Case	w_0	NA	NA	NA	NA	NA	NA
	GDP($t-1$)	NA	NA	NA	NA	NA	NA
	ERATE(t)	NA	NA	NA	NA	NA	NA
	SPI(t)	NA	NA	NA	NA	NA	NA
	GIR(t)	NA	NA	NA	NA	NA	NA
	BOP(t)	NA	NA	NA	NA	NA	NA
	ERATE($t-1$)	NA	NA	NA	NA	NA	NA
	SPI($t-1$)	NA	NA	NA	NA	NA	NA
	GIR($t-1$)	NA	NA	NA	NA	NA	NA
	BOP($t-1$)	NA	NA	NA	NA	NA	NA
3rd Case	w_0	✓	1	0.475	✗	-0.095	0.043
	GDP($t-1$)	✓	1	0.100	✗	0.273	0.069
	ERATE(t)	✓	1	0.299	✗	0.693	0.138
	SPI(t)	✓	20001	0.051	✗	0.377	0.070
	GIR(t)	✓	40001	0.425	✗	-0.211	0.042
	BOP(t)	✓	1	0.156	✗	0.018	0.019
	ERATE($t-1$)	✓	1	0.390	✓	-0.822	0.061
	SPI($t-1$)	✓	10001	0.117	✗	0.158	0.086
	GIR($t-1$)	✓	30001	0.051	✗	-0.042	0.122
	BOP($t-1$)	✓	1	0.133	✗	0.081	0.018

Table B.1: *Heidelberger-Welch's Stationarity and Halfwidth Tests of the Metropolis-Hasting's Mean Chain Across Cases. The ✓ indicates stationarity, ✗ indicates nonstationarity, and NA means Not Applicable.*

	Parameters	Stationarity Test			Halfwidth Test		
		Status	Start	p-value	Status	Mean	Halfwidth
1st Case	w_0	✓	101	0.10	✗	-0.083	0.0521
	GDP($t-1$)	✓	1	0.650	✓	0.261	0.025
	ERATE(t)	✓	1	0.670	✗	0.333	0.141
	SPI(t)	✓	1	0.390	✗	0.155	0.063
	GIR(t)	✓	1	0.900	✗	-0.288	0.030
	BOP(t)	✓	1	0.550	✗	0.001	0.010
	ERATE($t-1$)	✓	1	0.180	✗	-0.399	0.130
	SPI($t-1$)	✓	1	0.620	✗	0.359	0.091
	GIR($t-1$)	✓	401	0.170	✗	0.041	0.055
	BOP($t-1$)	✓	401	0.210	✗	0.044	0.016
2nd Case	w_0	✓	1	0.863	✓	-0.076	0.0047
	GDP($t-1$)	✓	3001	0.335	✓	0.312	0.009
	ERATE(t)	✓	1001	0.289	✓	0.513	0.018
	SPI(t)	✓	1	0.083	✗	0.114	0.030
	GIR(t)	✓	1	0.612	✓	-0.355	0.010
	BOP(t)	✓	1001	0.225	✗	-0.022	0.009
	ERATE($t-1$)	✓	1	0.414	✓	-0.548	0.038
	SPI($t-1$)	✓	2001	0.113	✓	0.406	0.023
	GIR($t-1$)	✓	2001	0.054	✗	0.102	0.014
	BOP($t-1$)	✓	2001	0.059	✗	0.035	0.015
3rd Case	w_0	✓	30001	0.084	✓	-0.095	0.001
	GDP($t-1$)	✓	1	0.110	✓	0.304	0.002
	ERATE(t)	✓	1	0.189	✓	0.523	0.005
	SPI(t)	✓	1	0.420	✓	0.172	0.003
	GIR(t)	✓	30001	0.661	✓	-0.329	0.003
	BOP(t)	✓	10001	0.437	✗	-0.005	0.001
	ERATE($t-1$)	✓	10001	0.652	✓	-0.591	0.004
	SPI($t-1$)	✓	1	0.520	✓	0.345	0.004
	GIR($t-1$)	✓	30001	0.636	✓	0.073	0.003
	BOP($t-1$)	✓	10001	0.126	✓	0.062	0.001

Table B.2: *Heidelberger-Welch's Stationarity and Halfwidth Tests of the Hamiltonian Monte Carlo's Mean Chain Across Cases. The ✓ indicates stationarity and ✗ indicates nonstationarity.*

Appendix C

R and Julia Packages for SGHMC

C.1 StochMCMC.r

The SGHMC R package developed for this thesis is hosted on Github.com and can be accessed through the following link:

`https://github.com/alstat/StochMCMC.r`

C.1.1 Installation

To install the package, run the following codes:

```
library(devtools)
install_github("alstat/StochMCMC.r")
```

and to load the package, simply run the following:

```
library(StochMCMC)
```

C.1.2 Documentations

For documentation of this package, please refer to the following link:

`http://stochmcmcr.readthedocs.io/en/latest/`

If there are concerns or problems regarding the package, please file an issue on the following link

`https://github.com/alstat/StochMCMC.r/issues`

C.1.3 SGHMC Source Code

The following is the implementation of the Algorithm 6 in R.

```
setClass("SGHMC",
  representation(
    dU       = "function",
    dK       = "function",
    dKSigma  = "array",
```

```
    C_mat    = "array",
    V_mat    = "array",
    init_est = "array",
    d        = "numeric"
  )
)

SGHMC <- function(dU, dK, dKSigma, C_mat, V_mat, init_est, d) {
  new("SGHMC", dU = dU, dK = dK, dKSigma = dKSigma,
     C_mat = C_mat, V_mat = V_mat, init_est = init_est, d = d)
}

setGeneric("mcmc", function(object, ...) {
  standardGeneric("mcmc")
})

setMethod("mcmc", signature(object = "SGHMC"), function(object,
  leapfrog_params = c(eps = .05, tau = 20),
  set_seed = 123,
  r = 1e+3) {

  dU <- object@dU;
  dK <- object@dK;
  dKSigma <- object@dKSigma
  C <- object@C_mat;
  V <- object@V_mat;
  w <- object@init_est;
  d <- object@d
  eps <- leapfrog_params["eps"];
  tau <- leapfrog_params["tau"]

  if (is.null(w)) {
    w = matrix(0, d, 1)
  } else {
    w = w
  }

  x <- matrix(0, r, d)
  x[1, ] <- w
  B <- .5 * V * eps
  D <- sqrt(2 * (C - B) * eps)

  if ((dim(B)[1] != dim(C)[1]) & (dim(B)[2] != dim(C)[2])) {
    error("C and V should have the same dimension.")
  } else {
    if (sum(dim(B)) > 1) {
      if (det(B) > det(C)) {
        error("eps is too big. Consider decreasing it.")
      }
    } else {
      if (B > C) {
        error("eps is too big. Consider decreasing it.")
      }
    }
  }

  for (i in 1:r) {
    p <- matrix(rnorm(d))
```

```
    for (j in 1:tau) {
      p <- p - dU(w) * eps - C %*% solve(dKSigma) %*%
        p + D %*% matrix(rnorm(d))
      w <- w + dK(p) * eps
    }

    x[i, ] <- w
  }

  return(x)
}
)
```

C.2 StochMCMC.jl

The Julia package developed for this thesis is hosted on Github.com and can be accessed through the following link:

`https://github.com/alstat/StochMCMC.jl`

C.2.1 Installation

To install the package, simply run the following codes:

```
Pkg.clone("https://github.com/alstat/StochMCMC.jl");
```

and to load the package, run the following:

```
using StochMCMC
```

C.2.2 Documentations

For documentation of this package, please refer to the following link:

`http://stochmcmcjl.readthedocs.io/en/latest/`

If there are concerns or problems regarding the package, please file an issue on the following link

`https://github.com/alstat/StochMCMC.jl/issues`

C.2.3 SGHMC Source Code

The following is the implementation of the Algorithm 6 in Julia.

```
immutable SGHMC
  dU      ::Function
  dK      ::Function
  dKSigma ::Array{Float64}
  C       ::Array{Float64}
  V       ::Array{Float64}
  init_est::Array{Float64}
```

```
  d       ::Int64
end

function mcmc(object::SGHMC;
  leapfrog_params::Dict{Symbol, Real} = Dict([:eps => .05, :tau => 20]),
  set_seed::Int64 = 123,
  r::Int64 = 1000)

  dU, dK, dKSigma = object.dU, object.dK, object.dKSigma
  C, V, w, d = object.C, object.V, object.init_est, object.d
  eps, tau = leapfrog_params[:eps], leapfrog_params[:tau]

  if typeof(set_seed) == Int64
    srand(set_seed)
  end

  chain = zeros(r, d);
  chain[1, :] = w
  B = .5 * V * eps
  D = sqrt(2 * (C - B) * eps)

  if size(B) != size(C)
    error("C and V should have the same dimension.")
  else
    if sum(size(B)) > 1
      if det(B) > det(C)
        error("eps is too big. Consider decreasing it.")
      end
    else
      if det(B[1]) > det(C[1])
        error("eps is too big. Consider decreasing it.")
      end
    end
  end

  for i in 1:r
    p = randn(d, 1)

    for j in 1:tau
      p = p - dU(w) * eps - C * inv(dKSigma) * p + D * randn(d, 1);
      w = w + dK(p) * eps;
    end

    chain[i, :] = w
  end

  return chain
end
```