MODERN MONTE CARLO METHODS AND THEIR APPLICATION IN SEMIPARAMETRIC REGRESSION

Samuel Joseph Thomas

Submitted to the faculty of the University Graduate School
in partial fulfillment of the requirements
for the degree
Doctor of Philosophy
in the Department of Biostatistics,
Indiana University
May 2021

Accepted by the Graduate Faculty, Indiana University, in partial
fulfillment of the requirements for the degree of Doctor of Philosophy.

Doctoral Committee

Wanzhu Tu, Ph.D., Chair

Ben Boukai, Ph.D.

March 12, 2021

Xiaochen Li, Ph.D.

Fengguang Song, Ph.D.

© 2021

Samuel Joseph Thomas

DEDICATION

To my wife, Patricia Anne Thomas, for her love and support through the years of study and research required for this degree, and to my children Alana and Sam Jr (Bo) for their understanding and encouragement all along the way. I also dedicate this work to my father, Samuel G. Thomas, the first Ph.D. in our family and inspiration for the degree, to my mother, Elizabeth Thomas, whose faith in my abilities kept me on the path, and to my brother and sister Andy and Mary Thomas for all of their support.

ACKNOWLEDGMENTS

Funding for this research was provided by Capital Group Companies, for which I am extremely grateful. I want to thank my managers and colleagues at Capital Group for their patience, and for accommodating my varied schedule through the years.

I would like to express my gratitude to my Ph.D. supervisor, Professor Wanzhu Tu, for his support and guidance throughout my research. The lessons that I have learned from the depth and breadth of Wanzhu's statistical knowledge and research acumen, as well as his classroom instruction, will stay with me throughout my career. I could not imagine a better mentor throughout my research.

I would also like to thank Professor Ben Boukai, my first advisor in this program, for his patience and for his belief in me despite my prior hiatus from academic studies. In addition, I would like to thank Ben for his classroom instruction in those first few years. His knowledge and passion for statistics deepened my appreciation for this field.

I want to thank Professor Xiaochun Li and Professor Fengguang Song as well for serving on my committee. Their classroom instruction on statistical and high performance computation provided me a foundation in the early days of my research.

I want to express a special thank you to Associate Director of Graduate Affairs, Shawne Mathis, for patiently helping me through all the complicated logistics, from my application to this program through the conclusion of my Ph.D. studies.

Samuel Joseph Thomas

MODERN MONTE CARLO METHODS AND THEIR APPLICATION IN SEMIPARAMETRIC REGRESSION

The essence of Bayesian data analysis is to ascertain posterior distributions. Posteriors generally do not have closed-form expressions for direct computation in practical applications. Analysts, therefore, resort to Markov Chain Monte Carlo (MCMC) methods for the generation of sample observations that approximate the desired posterior distribution. Standard MCMC methods simulate sample values from the desired posterior distribution via random proposals. As a result, the mechanism used to generate the proposals inevitably determines the efficiency of the algorithm. One of the modern MCMC techniques designed to explore the high-dimensional space more efficiently is Hamiltonian Monte Carlo (HMC), based on the Hamiltonian differential equations. Inspired by classical mechanics, these equations incorporate a latent variable to generate MCMC proposals that are likely to be accepted. This dissertation discusses how such a powerful computational approach can be used for implementing statistical models. Along this line, I created a unified computational procedure for using HMC to fit various types of statistical models. The procedure that I proposed can be applied to a broad class of models, including linear models, generalized linear models, mixed-effects models, and various types of semiparametric regression models. To facilitate the fitting of a diverse set of models, I incorporated new parameterization and decomposition schemes to ensure the numerical performance of Bayesian model fitting without sacrificing the procedure's general applicability. As a concrete application, I demonstrate how to use the proposed procedure to fit a multivariate generalized additive model (GAM), a nonstandard statistical model with a complex covariance structure and numerous parameters. Byproducts of the research include two software packages that all practical data analysts to use the proposed computational method to fit their own models. The research's main methodological contribution is the unified computational approach that it presents for Bayesian model fitting that can be used for standard and nonstandard statistical models. Availability of such a procedure has greatly enhanced statistical modelers' toolbox for implementing new and nonstandard statistical models.

Wanzhu Tu, Ph.D., Chair

TABLE OF CONTENTS

LIST OF TABLES ix

LIST OF FIGURES x

CHAPTER 1 A Review of MCMC Methods 1

- 1.1 Introduction 1
- 1.2 MCMC in Bayesian Statistical Analysis 2
  - 1.2.1 Metropolis-Hastings 3
  - 1.2.2 Gibbs Sampling 6
  - 1.2.3 Variations of Metropolis-Hastings and Gibbs Sampling 8
  - 1.2.4 Limitations of Standard MCMC 9
- 1.3 Motivation for Hamiltonian Monte Carlo 12

CHAPTER 2 A Bayesian Framework for Statistical Model Estimation 16

- 2.1 Making Hamiltonian Monte Carlo Accessible to Statisticians 16
- 2.2 Introduction to Hamiltonian Monte Carlo 17
  - 2.2.1 The Idea 17
  - 2.2.2 Fundamental Concepts of Hamiltonian Monte Carlo 20
  - 2.2.3 Euclidean Hamiltonian Monte Carlo 22
- 2.3 Modern Variants and Adaptations of HMC 31
  - 2.3.1 Riemannian Manifold Hamiltonian Monte Carlo 31
  - 2.3.2 Algorithmic Variations of HMC 36
- 2.4 A General Process for Fitting Statistical Models using HMC 42
  - 2.4.1 hmclearn: A Flexible Computational Tool 44
  - 2.4.2 Fitting a Mixed Effects Model in HMC 46

CHAPTER 3 Fitting Non-Standard Statistical Models Using HMC 57

3.1 Challenges using HMC to Fit Complex Statistical Models 58

3.2 A New Approach to Fitting Covariance Matrices Using HMC 60

3.3 Data Driven Prior Specification for Efficient HMC Estimation 65

3.4 Fitting Multivariate Response Generalized Additive Models Using HMC 69

3.4.1 Generalized Additive Model Formulation 71

3.4.2 Bivariate smoothing parameterization 72

3.4.3 Extending Generalized Additive Models to Multiple Responses . 74

3.4.4 Connecting Random Intercepts from Multiple Responses 74

3.5 Modeling Framework for Semiparametric Regression 75

3.5.1 Framework for Multivariate Response Modeling 79

3.5.2 bayesGAM: a General Purpose Package for Modeling 80

3.6 Fitting a Bivariate Response GAM Using HMC 87

3.7 Extending the Application of HMC in Statistics 90

REFERENCES 92

CURRICULUM VITAE

LIST OF TABLES

Table 3.1 GAM model fitting functionality from R packages 83

LIST OF FIGURES

Figure 1.1 Illustration of an expected value computation with the standard normal distribution 15

Figure 2.1 Convergence of the example MH simulation 18

Figure 2.2 Main Steps of the Hamiltonian Monte Carlo Method 30

Figure 2.3 Major steps of HMC implementation 43

Figure 2.4 Comparison of frequentist and Bayesian fit of the fixed effect parameters from the mixed effects model example from hmclearn 55

Figure 2.5 Comparison of frequentist and Bayesian fit of the random effect parameters from the mixed effects model example from hmclearn . . . 56

Figure 3.1 Contour plot of the bivariate response of diastolic and systolic blood pressure by bayesGAM 89

Figure 3.2 Correlation of the bivariate response of diastolic and systolic blood pressure to weight and age 90

CHAPTER 1

A Review of MCMC Methods

1.1 Introduction

Markov Chain Monte Carlo (MCMC) methods simulate sample values from the desired posterior distribution via random proposals. The computational efficiency of such algorithms depends on the construction of the proposal generating process. For example, Metropolis-Hastings proposals are generated by random walks. While the proposals have good theoretical properties, an unguided random walk is known to be inefficient in covering the support of the target density function. Gibbs Sampling can be inefficient as well, particularly when the posterior density has regions with narrow support. In high dimensional parameter spaces such as those encountered in semiparametric regression analysis, the convergence of these standard MCMC methods can be prohibitively slow for practical use.

One of the modern MCMC techniques designed to address exploring high-dimensional is called Hamiltonian Monte Carlo (HMC). This extension to Metropolis-Hastings is based on the Hamiltonian differential equations. Inspired by statistical physics, these equations incorporate a latent variable to develop more efficient MCMC proposals. Along this line, further improvements appear feasible by using the posterior density's inherent Riemannian geometry to better inform the MCMC proposals. A focus of this research is to investigate the possibility of applying HMC to improve the standard MCMC algorithms in complex semiparametric analysis. In addition to algorithmic development, this research implements newly developed algorithms into R packages **hmclearn** and **bayesGAM** for fitting of a broad class of statistical models. The end products of this research, including both computational algorithms and software packages, are intended to promote HMC to fit statistical models that are difficult or impossible using standard optimization techniques.

1.2 MCMC in Bayesian Statistical Analysis

Estimation and inference are made based on the observed data $\mathcal{D}$ together with *a priori* information from the parameters of interest $\boldsymbol{\theta} = (\theta_1, ..., \theta_k)^T \in \mathbb{R}^k$. The posterior distribution $f(\boldsymbol{\theta}|\mathcal{D})$ combines the data and prior information according to Bayes formula. The posterior is shown to be proportional to the product of the likelihood function $f(\mathcal{D}|\boldsymbol{\theta})$ and the prior density $f(\boldsymbol{\theta})$ (Carlin and Louis 2008),

$$f(\boldsymbol{\theta}|\mathcal{D}) = \frac{f(\mathcal{D}|\boldsymbol{\theta})f(\boldsymbol{\theta})}{\int f(\mathcal{D}|\boldsymbol{\theta})f(\boldsymbol{\theta})d\boldsymbol{\theta}} \propto f(\mathcal{D}|\boldsymbol{\theta})f(\boldsymbol{\theta}) \tag{1.1}$$

When $f(\boldsymbol{\theta}|\mathcal{D})$ cannot be analytically derived, MCMC methods such as *Gibbs Sampling* (Geman and Geman 1984) and the *Metropolis-Hastings* algorithm (Metropolis et al. 1953)(Hastings 1970) become the only options outside of large sample approximations.

In MCMC, a sequence of random samples is generated where each simulated value is dependent only on the previous sample. This generates a sequence of correlated samples that are more likely to concentrate around the areas of high probability density. While an adjustment is often made to the number of simulations to get a more accurate estimate of the sample size (i.e. often called the effective sample size), this method is still typically more efficient than uncorrelated random sampling.

The particular differences between MCMC methods are based on the construction of the transition probabilities. The posterior density is always the same, but the mechanism by which the Markov chain explores this density differs by the method, and by parameter selections within a method.

The mathematics of these transitions must be setup in such a way to ensure that the Markov chain is ergodic, and that the steady-state distribution of $f(\boldsymbol{\theta}|\mathcal{D})$ is sampled appropriately. One important property of Markov chains with a stationary distribution is that the chain is reversible, often called detailed balance (Brooks 1998). I abbreviate the posterior notation to $f(\boldsymbol{\theta})$ for brevity,

$$f(\boldsymbol{\theta}^{(t)})T(\boldsymbol{\theta}^{(t)}, \boldsymbol{\theta}^{(t+1)}) = f(\boldsymbol{\theta}^{(t+1)})T(\boldsymbol{\theta}^{(t+1)}, \boldsymbol{\theta}^{(t)}). \tag{1.2}$$

If the transition kernel T satisfies detailed balance, then $f(\boldsymbol{\theta})$ must be the steady-state distribution (Tierney 1994).

I integrate to show that the steady-state probability is achieved when detailed balance is satisfied (Kelly 2011).

$$\begin{aligned} f(\boldsymbol{\theta}^{(t+1)}) &= \int f(\boldsymbol{\theta}^{(t)})T(\boldsymbol{\theta}^{(t)}, \boldsymbol{\theta}^{(t+1)})d\boldsymbol{\theta} \\ &= \int f(\boldsymbol{\theta}^{(t+1)})T(\boldsymbol{\theta}^{(t+1)}, \boldsymbol{\theta}^{(t)})d\boldsymbol{\theta} \\ &= f(\boldsymbol{\theta}^{(t+1)}) \int T(\boldsymbol{\theta}^{(t+1)}, \boldsymbol{\theta}^{(t)})d\boldsymbol{\theta} \\ &= f(\boldsymbol{\theta}^{(t+1)}). \end{aligned} \tag{1.3}$$

In the discussion of MCMC, I will attempt to demonstrate how each of the MCMC methods is designed to achieve detailed balance, consequently resulting in an ergodic Markov chain with stationary distribution $f(\boldsymbol{\theta})$.

1.2.1 Metropolis-Hastings

Metropolis-Hastings (MH) defines a transition probability that produces a Markov chain that is ergodic and satisfies detailed balance (Gilks, Richardson, and Spiegelhalter 1995). Values of $\theta^{(t)}$ in the chain are defined in part by a proposal density, which I will define as

$q(\theta^{\text{PROP}}|\theta^{t-1})$. Here, θ^{PROP} is a proposal for the next value in the chain. This proposal density is conditioned on the previously stored value $\theta^{(t-1)}$. A variety of proposal functions can be used, with random walk proposals being a common choice (Gareth O. Roberts, Gelman, and Gilks 1997).

A Markov chain with this proposal function defined as the transition probability would satisfy detailed balance if the following relation holds,

$$f(\theta^{\text{PROP}})q(\theta^{(t-1)}|\theta^{\text{PROP}}) \stackrel{?}{=} f(\theta^{(t-1)})q(\theta^{\text{PROP}}|\theta^{(t-1)}). \tag{1.4}$$

However, there is no guarantee that this relationship is always true. It is likely that one side of the equation will be greater than the other, and detailed balance will not hold (Chib and Greenberg 1995).

In order to ensure detailed balance, the proposal is accepted with a probability α, defined by the ratio of both sides of (1.4),

$$\alpha(\theta^{\text{PROP}}|\theta^{(t-1)}) = \min\left(1, \frac{f(\theta^{\text{PROP}})q(\theta^{(t-1)}|\theta^{\text{PROP}})}{f(\theta^{(t-1)})q(\theta^{\text{PROP}}|\theta^{t-1})}\right). \tag{1.5}$$

Note that if the proposal density is symmetric, such that $q(\theta^{\text{PROP}}|\theta^{t-1}) = q(\theta^{(t-1)}|\theta^{\text{PROP}})$, the ratio simplifies to

$$\alpha(\theta^{\text{PROP}}|\theta^{(t-1)}) = \min\left(1, \frac{f(\theta^{\text{PROP}})}{f(\theta^{(t-1)})}\right). \tag{1.6}$$

The simplified form in (1.6) is the Metropolis algorithm, while the generalization in (1.5) was provided by Hastings. Now it is possible to define a transition probability

$$T(\theta^{\text{PROP}}|\theta^{(t-1)}) := q(\theta^{\text{PROP}}|\theta^{(t-1)})\alpha(\theta^{\text{PROP}}|\theta^{(t-1)}) \tag{1.7}$$

which satisfies detailed balance for MH (Chib and Greenberg 1995),

$$\begin{aligned}
f(\theta^{(t-1)})T(\theta^{(t-1)}, \theta^{\text{PROP}} &= f(\theta^{\text{PROP}})T(\theta^{\text{PROP}}, \theta^{(t-1)}) \\
f(\theta^{(t-1)})q(\theta^{(t-1)}, \theta^{\text{PROP}})\alpha(\theta^{(t-1)}, \theta^{\text{PROP}}) &= f(\theta^{\text{PROP}})q(\theta^{(t-1)}|\theta^{\text{PROP}})\alpha(\theta^{\text{PROP}}, \theta^{(t-1)}) \\
f(\theta^{(t-1)})q(\theta^{\text{PROP}}|\theta^{(t-1)}) \min\left(1, \frac{f(\theta^{\text{PROP}})q(\theta^{(t-1)}|\theta^{\text{PROP}})}{f(\theta^{(t-1)})q(\theta^{\text{PROP}}|\theta^{(t-1)})}\right) &= f(\theta^{\text{PROP}})q(\theta^{(t-1)}|\theta^{\text{PROP}}) \\
&\quad \min\left(1, \frac{f(\theta^{(t-1)})q(\theta^{\text{PROP}}|\theta^{(t-1)})}{f(\theta^{\text{PROP}})q(\theta^{(t-1)}|\theta^{\text{PROP}})}\right) \\
\min\left(f(\theta^{(t-1)})q(\theta^{\text{PROP}}|\theta^{(t-1)}), f(\theta^{\text{PROP}})q(\theta^{(t-1)}|\theta^{\text{PROP}})\right) &= \min\left(f(\theta^{\text{PROP}})q(\theta^{(t-1)}|\theta^{\text{PROP}}),\right. \\
&\quad \left. f(\theta^{(t-1)})q(\theta^{\text{PROP}}|\theta^{(t-1)})\right).
\end{aligned} \tag{1.8}$$

The final step is true due to the symmetry of the min operation.

One advantage of MH is that the algorithm does not require that the target density be fully specified, but only up to a constant which does not depend on θ. As such, these constants cancel when applied in the transition probability (1.5).

The steps in the MH algorithm can be specified.

Algorithm 1 Metropolis-Hastings

```
procedure MH($\theta^{(0)}, f^*(\theta), q(\theta^{(x)}|\theta^{(y)}), N$)
    Calculate $f^*(\theta^{(0)})$
    for $t = 1, ..., N$ do
        $\theta^{\text{PROP}} \leftarrow q(\theta^{\text{PROP}}|\theta^{(t-1)})$
        $u \leftarrow U(0,1)$
        $\alpha = \min\left(1, \frac{f(\theta^{\text{PROP}})q(\theta^{(t-1)}, \theta^{\text{PROP}})}{f(\theta^{(t-1)})q(\theta^{\text{PROP}}, \theta^{(t-1)})}\right)$
        If $\alpha < u$, then $\theta^{(t)} \leftarrow \theta^{\text{PROP}}$. Otherwise, $\theta^{(t)} \leftarrow \theta^{(t-1)}$
    end for
    return $\theta^{(1)}...\theta^{(N)}$
end procedure
```

For random walk proposal densities and other proper density functions, MH will be irreducible and aperiodic (Gelman et al. 2013). Under these conditions, the simulated results will be correlated samples from the posterior density after some number of initial steps away from the initial value (Chib and Greenberg 1995). Efficient proposals may be developed based on the structure of the posterior density (G. O. Roberts and Tweedie 1996).

1.2.2 Gibbs Sampling

Another traditional MCMC method is called Gibbs Sampling, which can be interpreted as a specific implementation of MH (Gelman et al. 2013). Most statistical models in practical application have multiple parameters. The posterior density for a model with k parameters $\boldsymbol{\theta} = (\theta_1, ..., \theta_k)^T$ from n observations of a continuous distribution $\mathbf{y} = (y_1, ..., y_n)^T$ would again be determined from Bayes formula,

$$\begin{aligned} f(\boldsymbol{\theta}|\mathbf{y}) &= \frac{f(\mathbf{y}|\boldsymbol{\theta})f(\boldsymbol{\theta})}{\int f(\mathbf{y}|\boldsymbol{\theta})f(\boldsymbol{\theta})d\boldsymbol{\theta}} \\ f(\theta_1, ...\theta_k|\mathbf{y}) &= \frac{f(\mathbf{y}|\theta_1, ..., \theta_k)f(\theta_1, ..., \theta_k)}{\int ... \int f(\mathbf{y}|\theta_1, ..., \theta_k)f(\theta_1, ..., \theta_k)d\theta_1, ..., d\theta_k}. \end{aligned} \tag{1.9}$$

A joint distribution can be derived from its conditional distributions, provided that the joint distribution exists (Arnold and Press 1989). This property is advantageous for statistical models where the conditional posterior densities can be fully specified. Statistical models based on the exponential family of distributions (e.g. linear regression, logistic regression) are among the many popular types of models where it is possible to specify the conditional distributions (Diaconis, Khare, and Saloff-Coste 2008).

The first step in Gibbs sampling is the mathematical derivation of the conditional posterior densities. The Gibbs sampler then proceeds to draw N samples from these densities in a sequential fashion with initial values for all but one of the parameters specified (Carlin and Louis 2008).

Algorithm 2 Gibbs Sampling

```
procedure GIBBS($\theta_2^{(0)}, ..., \theta_k^{(0)}, N$)
    for $t = 1, ..., N$ do
        $\theta_1^{(t)} \leftarrow f(\theta_1|\theta_{-1}^{(t-1)}, y)$
        $\theta_2^{(t)} \leftarrow f(\theta_2|\theta_1^{(t)}, \theta_{(-1,-2)}^{(t-1)}, y)$
        ...
        $\theta_k^{(t)} \leftarrow f(\theta_k|\theta_{-k}^{(t)}, y)$
    end for
    return $\boldsymbol{\theta}^{(1)}...\boldsymbol{\theta}^{(N)}$
end procedure
```

The transition kernel for Gibbs sampling is then the product of the conditional densities of the posterior (Schervish and Carlin 1992),

$$K(\boldsymbol{\theta}^{(t)}|\boldsymbol{\theta}^{(t-1)}) = f(\theta_1^{(t)}|\theta_{-1}^{(t-1)})f(\theta_2^{(t)}|\theta_1^{(t)}, \theta_{-1,-2}^{(t-1)})...f(\theta_k^{(t)}|\theta_{-k}^{(t-1)}). \tag{1.10}$$

Provided $f(\theta_i|\theta_{-i}, y)$ is well-defined for the data $\mathbf{y} \in \mathcal{D}$, then $K(\boldsymbol{\theta}^{(t)}|\boldsymbol{\theta}^{(t-1)})$ produces a Markov chain that is irreducible and aperiodic (G. O. Roberts and Smith 1994). One characteristic of Gibbs sampling is that the Markov chain will produce samples that approximate the marginal densities (Casella and George 1992),

$$\begin{aligned} f(\theta_i|\mathbf{y}) &= \int f(\boldsymbol{\theta}|\mathbf{y})d\boldsymbol{\theta}_{-i} \\ &= \int f(\boldsymbol{\theta}_{-i}, \theta_i|\mathbf{y})d\boldsymbol{\theta}_{-i} \\ &= \int f(\theta_i|\boldsymbol{\theta}_{-i}, \mathbf{y})f(\boldsymbol{\theta}_{-i}|\mathbf{y})d\boldsymbol{\theta}_{-i} \\ &= E\left[f(\theta_i|\boldsymbol{\theta}_{-i}, \mathbf{y})\right]. \end{aligned} \tag{1.11}$$

The estimated marginal distributions then come directly from the simulations for each $\theta_i \in \boldsymbol{\theta}$ (Casella and George 1992),

$$\hat{f}(\theta_i) = \frac{1}{N}\sum_{i=1}^{N} f(\theta_i|\boldsymbol{\theta}_{-i}). \tag{1.12}$$

The ergodicity of the Markov chain assures convergence to the conditional expectation of the posterior as $N \to \infty$ (Besag et al. 1995). Sampling from the conditional density $f(\theta_i|\boldsymbol{\theta}_{-i})$ provides more information than sampling just θ_i and provides a better estimate, known as the Rao-Blackwellization property of the Gibbs sampler (J. S. Liu, Wong, and Kong 1994). This can be seen from the law of total variance,

$$\begin{aligned} Var(\theta_i) &= E[Var(\theta_i|\boldsymbol{\theta}_{-i})] + Var[E(\theta_i|\boldsymbol{\theta}_{-i})] \\ Var[E(\theta_i|\boldsymbol{\theta}_{-i})] &= Var(\theta_i) - E[Var(\theta_i|\boldsymbol{\theta}_{-i})] \qquad (1.13) \\ &\leq Var(\theta_i). \end{aligned}$$

When the derivation of conditional posterior densities is possible, the Gibbs sampler offers an advantage of accepting the proposal for every iteration (Bonamente 2016). In contrast, standard Metropolis-Hastings simulations require the selection of a proposal density (e.g. a random walk) that could require tuning to achieve convergence in a reasonable period of time. The transition kernel in Gibbs does not have this requirement, which removes several implementation burdens from the analyst.

1.2.3 Variations of Metropolis-Hastings and Gibbs Sampling

The Metropolis-Hastings and Gibbs algorithms are sometimes used jointly to fit certain statistical models that would be difficult with just one or the other technique (Carlin and Louis 2008). This flexibility can be advantageous to fit complex models with many parameters, particular those models with parameters whose full conditional posterior distribution cannot be analytically derived. Gibbs is typically effective in exploring the distribution near the starting point, while random-walk Metropolis-Hastings may be more effective for multimodal or other complex distributions (Robert 2007). Software packages focused on MCMC simulation often incorporate elements of both algorithms in model fitting.

Adaptive Rejection Metropolis Sampling (ARMS) is an example of a specific method that combines the two approaches, including a Metropolis-Hastings step within a Gibbs sampling chain (Gilks, Best, and Tan 1995). The Metropolis-Hastings step is used for conditional posterior densities that may be difficult to sample efficiently using Gibbs. The proposal for this MH step is based on adaptive rejection sampling (Ripley 1987), which uses an envelope function of the target conditional distribution to create samples. The intent of this proposal is to reduce the probability of rejection, thereby optimizing the efficiency of the algorithm.

An alternative to Metropolis-Hastings and Gibbs sampling that is often used in conjunction with these standard MCMC techniques is called slice sampling (Radford M. Neal 2003). This method implements an auxiliary variable ν to sample from the posterior density based on horizontal slices (Carlin and Louis 2008). Slice sampling iterates via repeated sampling of the uniform distribution (Gelman et al. 2013). This method has the advantages of generality to a wide variety of distributions and not requiring the analyst to tune a proposal density, which leads itself well-suited for automated software implementations (Radford M. Neal 2003). One disadvantage of slice sampling is that it does not perform well on multi-modal distributions (Thompson and Neal 2010).

1.2.4 Limitations of Standard MCMC

Ergodic Markov chains require irreducible and aperiodic transition probabilities. From a theoretical standpoint, these restrictions are fairly minimal. Based on these aspects of MCMC theory, Gibbs and Metropolis-Hastings should be sufficient to fit any reasonably well-behaved statistical model. However, each of these methods have characteristics that limit their application for difficult problems.

The transition probabilities of Gibbs sampling are based on the conditional posterior densities. When the conditional densities are available, Gibbs sampling will almost always meet the theoretical requirements of irreducibility or aperiodicity (G. O. Roberts and Smith 1994). The main limitation of Gibbs is that the full conditional densities cannot always be derived. When the full conditionals are not available, Metropolis-Hastings or other techniques must be used for the analysis.

Metropolis-Hastings also meets the theoretical requirements of MCMC, but without the restrictions of fully specified conditional posterior densities. The main limitation of Metropolis-Hastings is the inherent computational inefficiency of random walk proposals. One possible difficulty in the selection of a proposal function is a mismatch between the domain of the proposal and the support of the posterior density (G. O. Roberts and Smith 1994). One can imagine an example where the posterior is a standard Normal $\pi \sim N(0, 1)$, but a random-walk transition is used with a high variance $q(\theta^{(t)}|\theta^{(t-1)}) \sim f(\theta^{(t-1)}) + Unif(-1e6, +1e6)$. The theoretical properties of irreducibility and aperiodicity would be satisfied, but the convergence rate of such a proposal would be extremely slow. The more common research questions regarding these standard MCMC methods are the rates of convergence and detection of convergence, rather than the theoretical properties of convergence (Gelman and Rubin 1992)(Brooks and Gelman 1998)(G. O. Roberts and Tweedie 1996).

An ergodic Markov chain will explore the entire posterior density given infinite time (Radford M. Neal 1993). However, MCMC chains must have a finite termination point to be practically useful. The principal risk of MCMC methods in practice is a chain that has not had sufficient time to cover the target density (Gilks, Richardson, and Spiegelhalter 1995). The challenge is to determine how many simulations must be run before an MCMC simulation can adequately represent the posterior distribution (Cowles and Carlin 1996).

Many convergence detection approaches have been developed. Some of the methods are heavily theoretical. For example, Rosenthal (1993) determined that the rate of convergence of a particular data augmentation algorithm similar to Gibbs sampling was approximately $O(\log n)$. However, such a a result is based on a special case of Gibbs sampling. The difficulty of developing the mathematics for many Gibbs applications limits the practical application of such results (Cowles and Carlin 1996).

In practice, diagnostic software such as CODA (Best, Cowles, and Vines 1995) is used to computationally detect convergence of MCMC simulations. The Gelman and Rubin (1992) test for convergence based on multiple independent Markov chains remains one of the most popular diagnostics. The idea behind Gelman and Rubin's test is that each independent chain should converge to the same distribution. This statistic, called the Potential Scale Reduction Factor (PSRF) $\hat{R}$ (Brooks and Gelman 1998) calculates the ratio of the variance between the means of each individual chain and average within-chain variances. If each chain converges, then the within-chain variance should dominate the between-chain variance (Cowles and Carlin 1996), creating a ratio close to zero and a PSRF close to one.

Ultimately, practical limitations of standard MCMC methods are due in greater part to numerical and computational issues than the theoretical basis of Bayesian statistics. For any of the MCMC methods discussed to this point, analysts must make decisions related to computation. At a minimum, analysts must determine how long a simulation must run before termination. A MCMC simulation that converges quickly and efficiently to the target density is imperative for Bayesian inference to be practically useful.

1.3 Motivation for Hamiltonian Monte Carlo

Modern MCMC algorithms extend standard methods to more efficiently explore the posterior density. These extensions replace the purely random proposals of traditional Metropolis-Hastings with problem-specific information. The intent of this additional information is to produce a more efficient Markov chain, while preserving the essential requirements for ergodicity. The theoretical basis for these informed proposals is derived from a number of fields outside of traditional statistics, including computer science, information theory, differential geometry, and physics. The fundamental concepts are introduced regarding these methods and provide references for those interested in exploring further.

The principal Metropolis-Hastings extension introduced here is sometimes called Hybrid Monte Carlo (Duane et al. 1987), but is more commonly referred to as Hamiltonian Monte Carlo (HMC) today (MacKay 2003) (R. Neal 2011). HMC expands the traditional Metropolis-Hastings technique by providing additional information from the target density to guide the chain.

The original HMC method is based geometrically on Euclidean space, and is more specifically described as Euclidean Hamiltonian Monte Carlo (EHMC) (Michael Betancourt et al. 2017). This designation identifies a specific type of HMC from a broader from a broader geometry called Riemannian HMC (RHMC) (Girolami and Calderhead 2011). Finally, I will provide an overview of recent variations of these methods, along with computational and software considerations.

In MCMC simulation, calculating expected values with respect to the posterior density is frequently the objective. Calculating expected values involves computing or estimating an integral (Michael Betancourt 2017). The expected value of $g(\theta)$ with respect to the density

$f(\theta)$ is described

$$\mathbf{E}_f(g(\theta)) = \int_{\boldsymbol{\theta}} g(\theta) f(\theta) d\theta. \tag{1.14}$$

Direct calculations of these integrals are not always possible (Voss 2013). When a direct calculation is not possible, numerical or simulation methods must be used to approximate these integrals. The high-dimensional challenge of computing such an integral can be illustrated by thinking of the expected value integral as the product of volume and density,

$$\begin{aligned} \mathbf{E}_f(g(\theta)) &= \mathit{volume} \cdot \mathit{density} \\ &= \int_{\Theta} d\theta \cdot g(\theta) f(\theta). \end{aligned} \tag{1.15}$$

The contributions of *volume* and *density* can be used to define a *typical set* in the estimation of such an integral (MacKay 2003). In the continuous case, the typical set is expressed in terms of the density function and differential entropy (Vasicek 1976). The differential entropy $H(f)$ is defined as

$$\begin{aligned} H(x) &= -\int f(x) \log f(x) dx \\ &= \mathbf{E}_f[-\log f(x)], \end{aligned} \tag{1.16}$$

and can be interpreted as measure of the average information content(MacKay 2003).

The typical set can then be defined for a sequence of N random draws from a probability density $f(x)$ for some small $\epsilon > 0$

$$\left| \frac{1}{N} \log f(x_1, ..., x_N) - H(x) \right| \leq, \epsilon \tag{1.17}$$

where $\frac{1}{N} \log f(x_1, ..., x_N) \xrightarrow{p} H(x)$. In other words, a set of N draws is considered a typical set if the average probability is close to its differential entropy (Cover and Thomas 2012). An efficient MCMC simulation will sample many of the points in this typical set since this region contributes the most information to expectations (Michael Betancourt 2017).

Example: Typical set of the standard normal density

These concepts can be illustrated with a standard normal density.

$$f(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}, \tag{1.18}$$

with the logarithm of the density

$$\log f(x) = -\frac{1}{2}\left[\log 2\pi + x^2\right]. \tag{1.19}$$

Next, I determine the differential entropy by calculating the expectation from equation 1.16,

$$\begin{aligned} H(x) &= -\int f(x)\log f(x)dx \\ &= \int f(x)\frac{1}{2}\left[\log 2\pi + x^2\right]dx \\ &= \frac{1}{2}\log 2\pi + \frac{1}{2}\mathbf{E}(\mathbf{x})^{\mathbf{2}} \\ &= \frac{1}{2}\left[1 + \log 2\pi\right]. \end{aligned} \tag{1.20}$$

As x moves further away from areas of high density, the negative log of the density increases with distance away from the typical set. In calculating an expectation of the standard normal, I note in Figure 1.1 that, as the distance increases from the mode, the volume contributes more to the expectation than the density. I note the rapidly diverging contributions of the density and volume, even in this one-dimensional case.

This divergence becomes particularly problematic in high-dimensional space; the typical set resides in a small portion of the total volume (Michael Betancourt 2017). This "curse of dimensionality"(Bellman 1957) is not unique to MCMC, but is universally challenging to any estimation process in statistical computing. For example, quadrature methods (Whittaker and Robinson 1967) are popular in frequentist statistics, but tend to scale poorly for high dimensional problems (Evans and Swartz 2000).

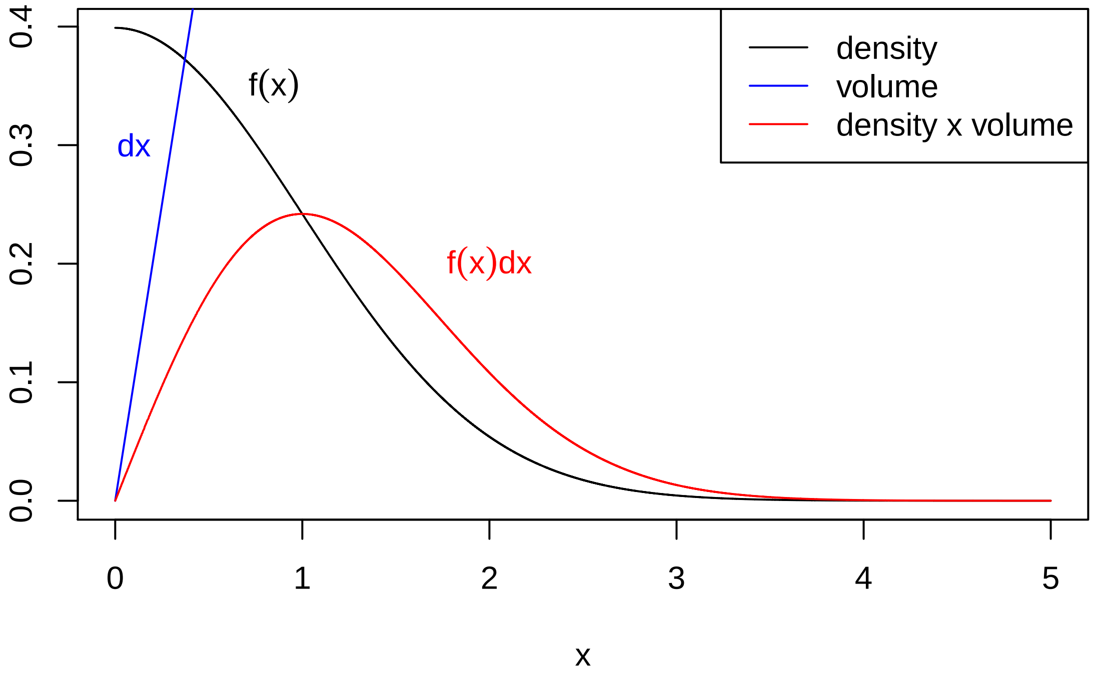

Figure 1.1: Illustration of an expected value computation with the standard normal distribution

The standard MCMC algorithms random-walk Metropolis-Hastings and Gibbs sampling can be inefficient when the posterior density is unfamiliar to the analyst (Turner et al. 2013). The inherent inefficiency of uninformed, random proposals of Metropolis-Hastings and the local exploration of Gibbs sampling (Robert 2007) is especially challenging in high-dimensions. Modern MCMC methods leverage information about the target density and the inherent geometry of probability distributions (Efron 1978) to produce more efficient estimations (Michael Betancourt et al. 2017).

CHAPTER 2

A Bayesian Framework for Statistical Model Estimation

2.1 Making Hamiltonian Monte Carlo Accessible to Statisticians

Hamiltonian Monte Carlo (HMC) is one of the newer Markov Chain Monte Carlo (MCMC) methods for Bayesian computation. An essential advantage of HMC over the traditional MCMC methods, such as the Metropolis-Hastings algorithm, is its greatly improved computational efficiency, especially in higher-dimensional and more complex models. But despite the method's computational prowess and the existence of excellent introductions (R. Neal 2011)(Michael Betancourt 2017), practitioners still face daunting challenges in applying the method to their own applications. Difficulties mainly arise in three areas: (1) unfamiliarity with the theory behind the algorithm, (2) lack of understanding of how the existing software works, (3) inability to tune the HMC parameters. These difficulties have limited the use of HMC to those who understand the theory and have the programming skills to implement the algorithm. But it does not have to be so.

A major challenge to understanding HMC is the algorithm's basis in fields unfamiliar to statisticians. The mathematics behind HMC is based on differential geometry, which is an abstract and challenging subject to understand (Michael Betancourt 2017). Some of this geometry is covered in a discussion on modern variants of HMC. However, this initial introduction relies on concrete analogies from physical laws of motion. Physics provides a less abstract interpretation of the Hamiltonian equations which can be helpful for some in learning the essential concepts.

In this chapter, I present a general framework for analysts to fit statistical models on their own. To that end, a comprehensive overview of the HMC algorithm and the intuition behind the method are provided. This overview is intended to demystify the HMC algorithm for statisticians who are reluctant to use methods they do not understand. The differentiating characteristics of this overview compared to other HMC introductions include,

1. An introduction to the HMC algorithm from a more familiar *statistical* perspective,
2. A concise step-by-step process for statisticians to fit their own models, and
3. A general purpose software package in the familiar `R` language to implement models in practice.

To summarize, the intent of this section is to provide statisticians with sufficient theoretical background to confidently use HMC in practice, a structured approach to model fitting with HMC, and the tools necessary for research and data analysis.

2.2 Introduction to Hamiltonian Monte Carlo

2.2.1 The Idea

One way to improve the efficient convergence of MCMC is to adopt a proposal generating mechanism that samples more frequently in parts of the parameter space that are more likely to be accepted. Hamiltonian equations in classical mechanics turn out to be a perfect tool to do so. In 1987, Duane and colleagues described for the first time one such procedure, which they called the "Hybrid Monte Carlo'' (Duane et al. 1987), abbreviated as HMC. Because the method is based on the Hamiltonian dynamics of physics, it later acquired the name Hamiltonian Monte Carlo, retaining the abbreviation of HMC (R. Neal 2011).

The idea of HMC is quite intuitive: To generate samples that mimic the behavior of a target function $f(\boldsymbol{\theta})$, one should sample the high density areas more frequently than the low density areas. Or, if one works with $-\log f(\boldsymbol{\theta})$, one should frequent the areas of low values instead of high values of the negative log density. Such a sampling process can be carried out in a more guided way, by mimicking the movement of objects in a simplified physical environment.

I consider the movement of an object on a frictionless curve of varying height (R. Neal 2011). There are two dimensions in this example, although only one of the two dimensions is of actual interest. The horizontal position of the object represents the value of $\boldsymbol{\theta}$, the parameter of interest. The vertical position of the object is related to the auxiliary variable, the momentum, whose purpose is to help guide the object along the path. The shape of the curve itself is based on the particular negative log posterior function over which I am sampling.

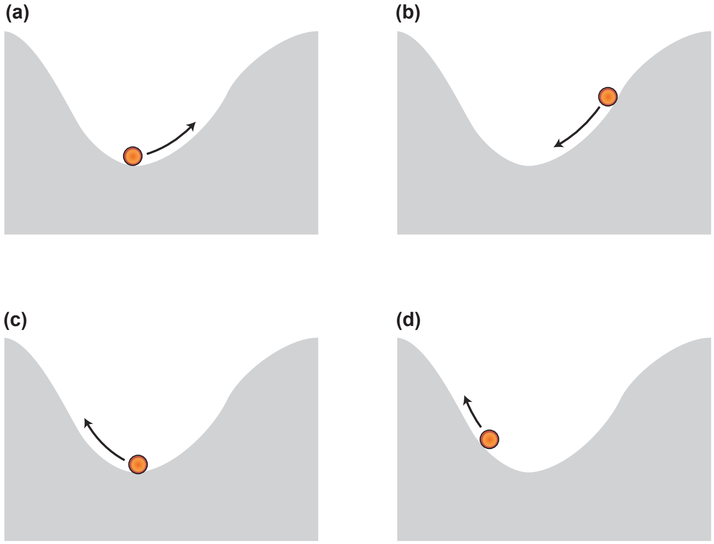

Figure 2.1: Convergence of the example MH simulation

This example is illustrated in Figure 2.1: (a) I apply a force with randomly generated direction and strength to the object. This object acquires a certain amount of kinetic energy, which makes it move in the direction of the applied force. The momentum, proportional to the object's velocity, changes throughout the path of the curve. (b) When the object moves up along the curve, the velocity of the object, and therefore, its momentum, decreases. Its kinetic energy decreases and potential energy increases, while the total energy remains constant. The object will stop at a point when all of its kinetic energy is converted to potential energy. (c) The potential energy then makes the object move in the opposite direction, converting its potential energy back to kinetic energy. At the lowest point of the curve, all of the energy is in the kinetic form (peak momentum), which would push the object up to the left side of the curve. (d) As the object goes up on the curve, its kinetic energy is converted to potential energy, until all is in the form of potential energy. Then the object would stop and then slide back as guided by its potential energy. Since the surface is frictionless, the total energy remains constant throughout these repeated movements.

Suppose I randomly select a stopping point for the object along this curve. The horizontal position would represent a sampled value of the parameter of interest. This would represent a single proposal in the MCMC chain.

Next, suppose I repeat this example along the same curve, but with a lower force (and, therefore, a lower peak velocity). The object will have a lower momentum and traverse a smaller portion of the curve, concentrated in the middle. Thus, the next sample would have to be in the region of higher posterior density, regardless of the selected stopping point. If I repeat this process with randomly applied momenta, one could obtain samples following the target distribution. Most of the samples will be concentrated in the middle of the curve, while fewer will be in the tails. Samples in the tails would only be possible when the randomly applied force (and momentum) is high, while samples in the middle would be possible for most randomly selected momenta.

But how does this moving object remain on the curve? Hamiltonian equations provide an answer. Hamiltonian equations are differential equations that govern the conversion of kinetic and potential energy over the surface of a given function. One therefore could use these equations to guide the generation of samples following a given distribution. To do so, one only needs randomly generated momenta to keep the object moving back and forth along the target function; the momenta themselves, however, are never the endpoints of this exercise.

2.2.2 Fundamental Concepts of Hamiltonian Monte Carlo

From statistical mechanics, the canonical probability distribution (or Boltzmann distribution) is defined

$$P(x) = \frac{1}{Z} \exp\{-E(x)/T\}, \tag{2.1}$$

where $E(x)$ is an energy function for state x, T is the temperature of the system and Z is a positive normalizing constant (Gibbs 1902).

The canonical distribution can be re-written as an invariant Hamiltonian function (Michael Betancourt 2017) which describes total mechanical energy (Arnol'd 2013). I will call the auxiliary variable p based on the notation that comes from physics equations describing total energy, where p is commonly used to denote physical momentum. This variable is not of interest itself, but is used in the proposal density to assist in the simulation of the target posterior density $f(\theta|\mathcal{D})$.

In the univariate case, the inputs to the Hamiltonian $H(\theta, p)$ are position $\theta \in \mathbb{R}$ and momentum $p \in \mathbb{R}$ based on their physical interpretations. From a statistical perspective, θ is the parameter of interest and p is a latent variable. The canonical distribution can then be more specifically written for the application as

$$\begin{aligned} f(\theta, p) &= \frac{1}{Z} \exp\{-H(\theta, p)/T\} \\ f(\theta, p) &\propto \exp\{-H(\theta, p)\}. \end{aligned} \tag{2.2}$$

Based on position and momentum, total energy is defined in the Hamiltonian as the sum of kinetic $K(\theta, p)$ and potential $U(\theta)$ energy (Nakahara 2003) (2.3),

$$H(\theta, p) = K(\theta, p) + U(\theta). \tag{2.3}$$

Continuing the physical analogy, I describe the potential and kinetic energy as first-order differential equations with respect to time (Arnol'd 2013). These relationships of the Hamiltonian to time will be instrumental in the MCMC implementation,

$$\begin{aligned} \frac{dp}{dt} &= -\frac{\partial H}{\partial \theta} \\ \frac{d\theta}{dt} &= \frac{\partial H}{\partial p}. \end{aligned} \tag{2.4}$$

I show that the total rate of change of the Hamiltonian with respect to time is zero. From a physical perspective, this is analogous to the conservation of energy (Abraham and Marsden 1978). By the chain rule (2.5), I demonstrate the invariance of the Hamiltonian function,

$$\begin{aligned} \frac{dH(\theta, p)}{dt} &= \frac{\partial H}{\partial \theta}\frac{\partial \theta}{\partial t} + \frac{\partial H}{\partial p}\frac{\partial p}{\partial t} \\ &= -\frac{\partial p}{\partial t}\frac{\partial \theta}{\partial t} + \frac{\partial \theta}{\partial t}\frac{\partial p}{\partial t} \\ &= 0. \end{aligned} \tag{2.5}$$

This conservation implies an acceptance probability of unity in the continuous case (R. Neal 2011). I will need to provide a discrete approximation of these differential equations in the HMC implementation.

Another important property of the Hamiltonian function is the conservation of volume in (θ, p) space (R. Neal 2011). By Liouville's theorem, the Hamiltonian system can be shown to preserve volume over time (Abraham and Marsden 1978). The conservation of volume can be demonstrated by showing that the divergence of the vector field described by (2.4) is zero

(R. Neal 2011)(Arnol'd 2013). For $(\theta_1, ..., \theta_k) \in \boldsymbol{\theta}$ and a momentum vector $(p_1, ..., p_k) \in \mathbf{p}$,

$$\begin{aligned}\sum_{j=1}^{k}\left[\frac{\partial}{\partial\theta_j}\frac{d\theta_j}{dt}+\frac{\partial}{\partial p_j}\frac{dp_j}{dt}\right] &= \sum_{j=1}^{k}\left[\frac{\partial}{\partial\theta_j}\frac{dH(\theta,p)}{dp_j}-\frac{\partial}{\partial p_j}\frac{dH(\theta,p)}{d\theta_j}\right]\\ &= \sum_{j=1}^{k}\left[\frac{\partial^2 H(\theta,p)}{\partial\theta_j\partial p_j}-\frac{\partial^2 H(\theta,p)}{\partial p_j\partial\theta_j}\right] \qquad (2.6)\\ &= 0.\end{aligned}$$

This aspect of Hamiltonian dynamics indicates that the probability space defined by $(\boldsymbol{\theta}, \mathbf{p})$ does not change over time.

In summary, the Hamiltonian function is invariant to transformation of parameters, thereby preserving the geometry of the typical set (Michael Betancourt 2017). The flow described by (2.4) incorporates information about the target density $f(\boldsymbol{\theta})$. In contrast with uninformed random-walk proposals, HMC proposals will be based no problem-specific information. By incorporating this information, HMC can improve on the efficiency of traditional MCMC algorithms (R. Neal 2011).

2.2.3 Euclidean Hamiltonian Monte Carlo

In applying HMC to a generic MCMC setting, $\boldsymbol{\theta}$ follows the posterior density $f(\boldsymbol{\theta})$, and the momentum $\mathbf{p}$ is generated from a parametric distribution. The momentum matches the dimensionality of $\boldsymbol{\theta}$ as a vector of length k. The Hamiltonian function in the multivariate case is similar to the univariate function $H(\boldsymbol{\theta}, \mathbf{p}) = U(\boldsymbol{\theta}) + K(\mathbf{p})$ where $\mathbf{p} \in \mathbb{R}^k$ and $\boldsymbol{\theta} \in \mathbb{R}^k$.

2.2.3.1 Algorithm Development

Since I am primarily interested in generating $\boldsymbol{\theta}$ from a given distribution $f(\boldsymbol{\theta})$, I let $U(\boldsymbol{\theta}) := -\log f(\boldsymbol{\theta})$. Such a designation ensures that MCMC samples of $\boldsymbol{\theta}$ generated from the Hamiltonian function follows the desired distribution. For momentum, I assume $\mathbf{p} \sim$

$N_k(0, \mathbf{M})$, where $\mathbf{M}$ is a user-specified covariance matrix. Under this formulation, I have

$$H(\boldsymbol{\theta}, \mathbf{p}) = -\log f(\boldsymbol{\theta}) + \frac{1}{2}\mathbf{p}^T \mathbf{M}^{-1} \mathbf{p}. \tag{2.7}$$

Over time, HMC travels on trajectories that are governed by the following first-order differential equations, known as the *Hamiltonian equations*

$$\begin{aligned} \frac{d\mathbf{p}}{dt} &= -\frac{\partial H(\boldsymbol{\theta}, \mathbf{p})}{\partial \boldsymbol{\theta}} = -\frac{\partial U(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \nabla_{\boldsymbol{\theta}} \log f(\boldsymbol{\theta}), \\ \frac{d\boldsymbol{\theta}}{dt} &= \frac{\partial H(\boldsymbol{\theta}, \mathbf{p})}{\partial \mathbf{p}} = \frac{\partial K(\mathbf{p})}{\partial \mathbf{p}} = \mathbf{M}^{-1}\mathbf{p}, \end{aligned} \tag{2.8}$$

where $\nabla_{\boldsymbol{\theta}} \log f(\boldsymbol{\theta})$ is the gradient of the log posterior density. A solution to the Hamiltonian equations is a function that defines the path of $(\boldsymbol{\theta}, \mathbf{p})$ from which specific values of $\boldsymbol{\theta}$ could be sampled. Within an MCMC iteration, I sample a value $\boldsymbol{\theta}$ from this path. The randomness of the MCMC samples comes from the momentum $\mathbf{p} \sim N_k(0, \mathbf{M})$ and the specific $\boldsymbol{\theta}$ value that is chosen.

As with other valid MCMC algorithms, HMC's transition probability is designed to meet the theoretical requirements for detailed balance and reversibility. These conditions ensure that the HMC samples provide a valid representation of the posterior distribution. If I denote the transition probability from $\boldsymbol{\theta}^{(1)}$ to $\boldsymbol{\theta}^{(2)}$ as $T(\boldsymbol{\theta}^{(1)}, \boldsymbol{\theta}^{(2)})$, then detailed balance requires that $f(\boldsymbol{\theta}^{(1)})T(\boldsymbol{\theta}^{(1)}, \boldsymbol{\theta}^{(2)}) = f(\boldsymbol{\theta}^{(2)})T(\boldsymbol{\theta}^{(2)}, \boldsymbol{\theta}^{(1)})$. The HMC transition probability includes two components to ensure that detailed balance and reversibility hold true:

1. The accept/reject step, and
2. The negation of the momentum after the final leapfrog step.

The negated momentum illustrates the reversibility of HMC transitions, which can be demonstrated by stepping through the leapfrog from the proposed state to the original state. (Tierney 1994) develops the theoretical requirements for MCMC algorithms in general, while (Michael Betancourt 2017) provides a detailed exposition specific to HMC.

From the canonical distribution (2.2), I will extend the posterior density $f(\boldsymbol{\theta})$ with an auxiliary momentum variable $\mathbf{p}$ to define a joint posterior density $f(\boldsymbol{\theta}, \mathbf{p})$. The constants for this distribution are set to unity,

$$f(\boldsymbol{\theta}, \mathbf{p}) = e^{-H(\boldsymbol{\theta}, \mathbf{p})}. \tag{2.9}$$

By calculating the logarithm of the canonical density, I formulate the relationship between the Hamiltonian function and the log posterior density,

$$\begin{aligned} H(\boldsymbol{\theta}, \mathbf{p}) &= -\log f(\boldsymbol{\theta}, \mathbf{p}) \\ &= -\log f(\mathbf{p}|\boldsymbol{\theta}) - \log f(\boldsymbol{\theta}) \\ &:= K(\boldsymbol{\theta}, \mathbf{p}) + U(\boldsymbol{\theta}). \end{aligned} \tag{2.10}$$

From the physical analogy, the potential energy function $U(\boldsymbol{\theta})$ is the negative log posterior, and the kinetic energy function $K(\boldsymbol{\theta}, \mathbf{p})$ is the conditional density of the latent momentum variable. In a standard EHMC algorithm, the momentum $\mathbf{p}$ is defined to be independent of the target $\boldsymbol{\theta}$, such that

$$f(\mathbf{p}|\boldsymbol{\theta}) = f(\mathbf{p}), \tag{2.11}$$

and

$$K(\boldsymbol{\theta}, \mathbf{p}) = K(\mathbf{p}). \tag{2.12}$$

The resulting Hamiltonian equation reflects the independence of the potential and kinetic energy functions,

$$\begin{aligned} H(\boldsymbol{\theta}, \mathbf{p}) &= K(\mathbf{p}) + U(\boldsymbol{\theta}) \\ &= -\log f(\mathbf{p}) - \log f(\boldsymbol{\theta}). \end{aligned} \tag{2.13}$$

I define the distribution of the momentum $\mathbf{p}$ as a multivariate Normal for the k parameters in $(\theta_1, ..., \theta_k)^T \in \boldsymbol{\theta}$ where $\mathbf{M}$ is the covariance matrix, often chosen to be the identity matrix

I,

$$\begin{aligned} f(\mathbf{p}) &= (2\pi)^{-k/2}|\mathbf{M}|e^{-\frac{1}{2}\mathbf{p}^T\mathbf{M}^{-1}\mathbf{p}} \\ &\propto e^{-\frac{1}{2}\mathbf{p}^T\mathbf{M}^{-1}\mathbf{p}} \end{aligned} \tag{2.14}$$

$$\log f(\mathbf{p}) \propto -\frac{1}{2}\mathbf{p}^T\mathbf{M}^{-1}\mathbf{p}.$$

The kinetic energy function becomes

$$\begin{aligned} K(\mathbf{p}) &= -\log f(\mathbf{p}) \\ &= \frac{1}{2}\mathbf{p}^T\mathbf{M}^{-1}\mathbf{p}. \end{aligned} \tag{2.15}$$

The kinetic energy function reduces to the analogous formula for kinetic energy in Newtonian physics $k = \frac{p^2}{2M}$ (Serway and Vuille 2012). Also, the kinetic energy is a quadratic function of p, which will become important in demonstrating the reversibility of the proposal (i.e. $K(\mathbf{p}) = K(-\mathbf{p})$). Now that the Hamiltonian function is defined for the target density and chosen momentum, I apply the flow equations (2.4) that will define the MCMC exploration,

$$\begin{aligned} \frac{d\mathbf{p}}{dt} &= -\frac{\partial U(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \nabla_{\boldsymbol{\theta}} \log f(\boldsymbol{\theta}) \\ \frac{d\boldsymbol{\theta}}{dt} &= \frac{\partial K(\mathbf{p})}{\partial \mathbf{p}} = \mathbf{M}^{-1}\mathbf{p}. \end{aligned} \tag{2.16}$$

The differential equations will be used to develop a proposal function for new values of $(\boldsymbol{\theta}^t, \mathbf{p}^t)$. However, these differential equations do not have a closed form solution with the exception of trivial examples. Therefore, I need to provide a discrete approximation. The simplest approximation is from Euler's method. If an approximate solution to the above differential equations is found using Euler's method,

$$\begin{aligned} \frac{d\mathbf{p}_i(t)}{dt} &\approx \frac{\mathbf{p}_i(t+\epsilon) - \mathbf{p}_i(t)}{\epsilon} \\ \frac{d\boldsymbol{\theta}_i(t)}{dt} &\approx \frac{\boldsymbol{\theta}_i(t+\epsilon) - \boldsymbol{\theta}_i(t)}{\epsilon}, \end{aligned} \tag{2.17}$$

such that

$$\begin{aligned} \mathbf{p}_i(t+\epsilon) &\approx \mathbf{p}_i(t) + \epsilon\frac{d\mathbf{p}_i(t)}{dt} \\ &\approx \mathbf{p}_i(t) + \epsilon\nabla_{\boldsymbol{\theta}} \log f(\boldsymbol{\theta}) \\ \boldsymbol{\theta}_i(t+\epsilon) &\approx \boldsymbol{\theta}_i(t) + \epsilon\frac{d\boldsymbol{\theta}_i(t)}{dt} \\ &\approx \boldsymbol{\theta}_i(t) + \epsilon\mathbf{M}^{-1}\mathbf{p}_i(t). \end{aligned} \tag{2.18}$$

This approximation is problematic for HMC, because the discretization does not preserve volume. I need to use a special type of approximation called a symplectic integrator, which has the property of preserving volume (Channell and Scovel 1990). The approximation that is commonly used in EHMC is called the Newton-Stormer-Verlet or "Leapfrog" integrator (Hairer, Lubich, and Wanner 2003). The Leapfrog method preserves volume at a cost of being slightly more complex than Euler's method. Leapfrog is defined to move in discrete steps of size ϵ,

$$\begin{aligned}\mathbf{p}(t+\epsilon/2) &= \mathbf{p}(t) + (\epsilon/2)\nabla_{\boldsymbol{\theta}} \log f(\boldsymbol{\theta}(t)) \\ \boldsymbol{\theta}(t+\epsilon) &= \boldsymbol{\theta}(t) + \epsilon \mathbf{M}^{-1}\mathbf{p}(t+\epsilon/2) \\ \mathbf{p}(t+\epsilon) &= \mathbf{p}(t+\epsilon/2) + (\epsilon/2)\nabla_{\boldsymbol{\theta}} \log f(\boldsymbol{\theta}(t+\epsilon)).\end{aligned} \tag{2.19}$$

I demonstrate that the Leapfrog is reversible by reversing the direction of the stepsize, where $\epsilon^* = -\epsilon$,

$$\begin{aligned}\mathbf{p}_i(t+\epsilon+\epsilon^*/2) &= \mathbf{p}_i(t+\epsilon) + \frac{\epsilon^*}{2}\nabla_{\boldsymbol{\theta}} \log f(\boldsymbol{\theta}_i(t+\epsilon)) \\ \boldsymbol{\theta}_i(t+\epsilon+\epsilon^*) &= \boldsymbol{\theta}_i(t+\epsilon) + \epsilon^* \mathbf{M}^{-1}\mathbf{p}_i(t+\epsilon+\epsilon^*/2) \\ \mathbf{p}_i(t+\epsilon+\epsilon^*) &= \mathbf{p}_i(t+\epsilon+\epsilon^*/2) + \frac{\epsilon^*}{2}\nabla_{\boldsymbol{\theta}} \log f(\boldsymbol{\theta}_i(t+\epsilon+\epsilon^*)).\end{aligned} \tag{2.20}$$

The course has reversed back to the original point,

$$\begin{aligned}\boldsymbol{\theta}_i(t+\epsilon+\epsilon^*) &= \boldsymbol{\theta}_i(t) \\ \mathbf{p}_i(t+\epsilon+\epsilon^*) &= \mathbf{p}_i(t).\end{aligned} \tag{2.21}$$

The analyst is responsible for setting several parameters for EHMC: the stepsize ϵ, the number of steps L for each proposal, and the mass matrix $\mathbf{M}$. Some guidance on the selection of these parameters is provided by R. Neal (2011) and Gelman et al. (2013). Automated algorithms are also available that automatically select these parameters for the user (Hoffman and Gelman 2014).

Algorithm 3 Euclidean Hamiltonian Monte Carlo

```
procedure EHMC($\boldsymbol{\theta}^{(0)}, \log f(\boldsymbol{\theta}), \mathbf{M}, N, \epsilon, L$)
    Calculate $\log f(\boldsymbol{\theta}^{(0)})$
    for $t = 1, ..., N$ do
        $\mathbf{p} \leftarrow N(0, \mathbf{M})$
        $\boldsymbol{\theta}^{(t)} \leftarrow \boldsymbol{\theta}^{(t-1)}, \tilde{\boldsymbol{\theta}} \leftarrow \boldsymbol{\theta}^{(t-1)}, \tilde{\mathbf{p}} \leftarrow \mathbf{p}$
        for $i = 1, ..., L$ do
            $\tilde{\boldsymbol{\theta}}, \tilde{\mathbf{p}} \leftarrow$ Leapfrog($\tilde{\boldsymbol{\theta}}, \tilde{\mathbf{p}}, \epsilon, \mathbf{M}$)
        end for
        $\alpha \leftarrow \min\left(1, \frac{\exp(\log f(\tilde{\boldsymbol{\theta}}) - \frac{1}{2}\tilde{\mathbf{p}}^T\mathbf{M}^{-1}\tilde{\mathbf{p}})}{\exp(\log f(\tilde{\boldsymbol{\theta}}^{(t-1)}) - \frac{1}{2}\mathbf{p}^T\mathbf{M}^{-1}\mathbf{p})}\right)$
        With probability $\alpha$, $\boldsymbol{\theta}^{(t)} \leftarrow \tilde{\boldsymbol{\theta}}$ and $\mathbf{p}^{(t)} \leftarrow -\tilde{\mathbf{p}}$
    end for
    return $\boldsymbol{\theta}^{(1)}, ..., \boldsymbol{\theta}^{(N)}$
    function LEAPFROG($\boldsymbol{\theta}^*, \mathbf{p}^*, \epsilon, \mathbf{M}$)
        $\tilde{\mathbf{p}} \leftarrow \mathbf{p}^* + (\epsilon/2)\nabla_{\boldsymbol{\theta}} \log f(\boldsymbol{\theta}^*)$
        $\tilde{\boldsymbol{\theta}} \leftarrow \boldsymbol{\theta}^* + \epsilon\mathbf{M}^{-1}\tilde{\mathbf{p}}$
        $\tilde{\mathbf{p}} \leftarrow \tilde{\mathbf{p}} + (\epsilon/2)\nabla_{\boldsymbol{\theta}} \log f(\tilde{\boldsymbol{\theta}})$
        return $\tilde{\boldsymbol{\theta}}, \tilde{\mathbf{p}}$
    end function
end procedure
```

Although the theoretical basis of EHMC is heavily mathematical, the particulars of the implementation are not substantially more complex than the standard Metropolis-Hastings algorithm. The most mathematically challenging aspect of this algorithm is the calculation of the gradient of the log posterior. Tuning EHMC also requires setting two parameters ϵ and L instead of selecting a proposal density as in Metropolis-Hastings. In addition, the analyst may elect to select a covariance matrix M that is not identity. For example, the diagonal elements of $\mathbf{M}$ may be scaled relative to the individual parameters $(\theta_1, ..., \theta_k \in \boldsymbol{\theta})$ (R. Neal 2011).

Like other well-known MCMC algorithms, EHMC also satisfies detailed balance and is, in most cases, ergodic (R. Neal 2011). Michael Betancourt (2017) shows that detailed balance is satisfied by the Hamiltonian proposals. The transition proposal is designated Q with $(\boldsymbol{\theta}_0, \mathbf{p}_0)$ as the starting location and $(\boldsymbol{\theta}_L, \mathbf{p}_L)$ as the proposal after L Leapfrog steps,

$$f(\boldsymbol{\theta}_0, \mathbf{p}_0)Q(\boldsymbol{\theta}_L, -\mathbf{p}_L|\boldsymbol{\theta}_0, \mathbf{p}_0) = f(\boldsymbol{\theta}_L, -\mathbf{p}_L)Q(\boldsymbol{\theta}_0, \mathbf{p}_0|\boldsymbol{\theta}_L, -\mathbf{p}_L). \tag{2.22}$$

Unlike random-walk Metropolis-Hastings, the Leapfrog-based proposals in EHMC are deterministic. The proposal always move L steps in stepsize increments of ϵ. A formalized definition of this proposal with the δ function is

$$\begin{gathered}\delta(x) = 1 \text{ if } x = 0 \\ \delta(x) = 0 \text{ otherwise.}\end{gathered} \tag{2.23}$$

Given $(\boldsymbol{\theta}_0, \mathbf{p}_0)$ as the initial point and $(\boldsymbol{\theta}_L, \mathbf{p}_L)$ as the end point after L steps each of size ϵ, I specify the proposal function for $(\boldsymbol{\theta}^*, \mathbf{p}^*)$,

$$Q(\boldsymbol{\theta}^*, \mathbf{p}^* | \boldsymbol{\theta}_0, \mathbf{p}_0) = \delta(\mathbf{p}^* - (-\mathbf{p}_L))\delta(\boldsymbol{\theta}^* - \boldsymbol{\theta}_L). \tag{2.24}$$

The negated momentum $-\mathbf{p}_L$ is required to assure reversibility. The quadratic form of the normal distribution ensures the negative momentum does not affect the joint distribution $f(\boldsymbol{\theta}, \mathbf{p}) = f(\boldsymbol{\theta}, -\mathbf{p})$ in (2.14). Based on the proposal function (2.24), the transition probabilities at $(\boldsymbol{\theta}_0, \mathbf{p}_0)$ and $(\boldsymbol{\theta}_L, \mathbf{p}_L)$ are both unity,

$$\begin{gathered}Q(\theta_L, -\mathbf{p}_L | \theta_0, \mathbf{p}_0) = \delta(-\mathbf{p}_L + \mathbf{p}_L)\delta(\boldsymbol{\theta}_L - \boldsymbol{\theta}_L) = 1 \\ Q(\boldsymbol{\theta}_0, \mathbf{p}_0 | \boldsymbol{\theta}_L, -\mathbf{p}_L) = \delta(\mathbf{p}_0 - \mathbf{p}_0)\delta(\boldsymbol{\theta}_0 - \boldsymbol{\theta}_0) = 1.\end{gathered} \tag{2.25}$$

I incorporate a correction probability α for the discretization of the Hamiltonian equations (R. Neal 2011) to balance the equation,

$$\begin{aligned}\alpha \cdot f(\boldsymbol{\theta}_0, \mathbf{p}_0) Q(\boldsymbol{\theta}_L, -\mathbf{p}_L | \boldsymbol{\theta}_0, \mathbf{p}_0) &= f(\boldsymbol{\theta}_L, -\mathbf{p}_L) Q(\boldsymbol{\theta}_0, \mathbf{p}_0 | \boldsymbol{\theta}_L, -\mathbf{p}_L) \\ \alpha \cdot f(\boldsymbol{\theta}_0, \mathbf{p}_0) \cdot 1 &= f(\boldsymbol{\theta}_L, -\mathbf{p}_L) \cdot 1 \\ \alpha &= \min\left(1, \frac{f(\boldsymbol{\theta}_L, -\mathbf{p}_L)}{f(\boldsymbol{\theta}_0, \mathbf{p}_0)}\right) \\ \alpha &= \min\left(1, \frac{e^{-H(\boldsymbol{\theta}_L, -\mathbf{p}_L)}}{e^{-H(\boldsymbol{\theta}_0, \mathbf{p}_0)}}\right) \\ \alpha &= \min\left(1, \frac{e^{-H(\boldsymbol{\theta}_L, \mathbf{p}_L)}}{e^{-H(\boldsymbol{\theta}_0, \mathbf{p}_0)}}\right).\end{aligned} \tag{2.26}$$

This formulation from Michael Betancourt (2017) shows that EHMC satisfies detailed balance. The remaining conditions required to establish ergodicity and a stationary posterior distribution are irreducibility and aperiodicity (Tierney 1994). EHMC can be shown to be irreducible provided the posterior density is twice continuously differentiable (Durmus, Moulines, and Saksman 2017)(Cancès, Legoll, and Stoltz 2007). The log posterior must be at least differentiable to run EHMC since the target is based on this gradient. The twice differentiable condition will be satisfied for exponential family based models provided the link function is twice differentiable (Fahrmeir, Kaufmann, and others 1985). In addition, Livingstone et al. (2016) show that EHMC is ergodic for a wide variety of models where the gradient of the log posterior is well-behaved.

The final condition of aperiodicity may be violated if the combination of parameters stepsize ϵ, number of leapfrog steps L, and momentum covariance matrix $\mathbf{M}$ combine to produce a proposals that are exactly periodic (R. Neal 2011). Incorporating some random component in ϵ and L may be helpful in preventing periodic transitions (Mackenze 1989). EHMC is seen to be generally ergodic provided some minor regularity conditions are met. The potential for increased efficiency does have a cost in the number and variety of parameter settings for EHMC, particularly in comparison with Metropolis-Hastings and Gibbs sampling. Before considering automated selection of EHMC parameters, I will continue with model formulation based on manual parameter selection.

The flowchart in Figure 2.2 shows the key steps in the HMC algorithm. Initial values for $\boldsymbol{\theta}$ and $\mathbf{p}$ are required to start the algorithm. With $\boldsymbol{\theta}^{(0)}$ and $\mathbf{p}^{(0)}$ specified, the leapfrog algorithm is used to find approximate solutions to the Hamiltonian equations. The leapfrog solutions define the path of $(\boldsymbol{\theta}, \mathbf{p})$ over time within an iteration.

Typically, multiple steps, each of length ϵ, are taken to generate an HMC proposal. Parameter L represents the number of steps. While L is often fixed to a positive integer value, some randomness can be introduced to ensure a valid exploration of the space of $(\boldsymbol{\theta}, \mathbf{p})$.

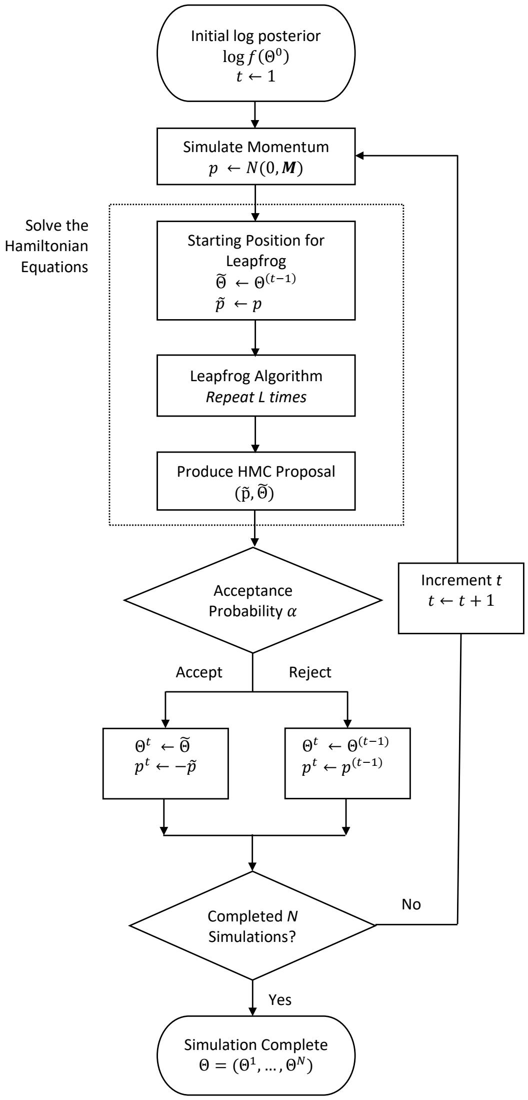
Initial log posterior
$\log f(\Theta^0)$
$t \leftarrow 1$
Simulate Momentum
$p \leftarrow N(0, \boldsymbol{M})$
Solve the Hamiltonian Equations
Starting Position for Leapfrog
$\tilde{\Theta} \leftarrow \Theta^{(t-1)}$
$\tilde{p} \leftarrow p$
Leapfrog Algorithm
Repeat L times
Produce HMC Proposal
$(\tilde{p}, \tilde{\Theta})$
Acceptance Probability α
Increment t
$t \leftarrow t+1$
Accept
Reject
$\Theta^t \leftarrow \tilde{\Theta}$
$p^t \leftarrow -\tilde{p}$
$\Theta^t \leftarrow \Theta^{(t-1)}$
$p^t \leftarrow p^{(t-1)}$
Completed N Simulations?
No
Yes
Simulation Complete
$\Theta = (\Theta^1, \dots, \Theta^N)$

Figure 2.2: Main Steps of the Hamiltonian Monte Carlo Method

The efficiency of an HMC algorithm can be improved through parameter tuning and reparameterization. HMC tuning involves selection and adjustment of the various HMC parameters. Two parameters that need to be specified are the step size ϵ and the number of leapfrog steps L. Elements in the covariance matrix $\mathbf{M}$ may also be adjusted from the default identity matrix for efficiency improvement.

Euclidean Hamiltonian Monte Carlo utilizes the gradient of $\log f(\boldsymbol{\theta})$ to generate more efficient proposals in a MCMC. The gradient introduces some additional computational burden, which, ideally, is offset by the benefit of proposals of larger jumps into higher probability space.

Practical challenges in tuning Euclidean Hamiltonian Monte Carlo include setting the stepsize parameter ϵ and number of Leapfrog steps L. Related to ϵ is the covariance matrix $\mathbf{M}$ specified for the latent variable $\mathbf{p}$.

A poorly selected stepsize ϵ can cause the MCMC to converge too slowly if too small, and miss narrow regions of the probability space if too large (Michael Betancourt 2017). An extension of EHMC that uses the second derivative of $\log f(\boldsymbol{\theta})$ as a replacement for the covariance matrix $\mathbf{M}$ is introduced in the next section.

2.3 Modern Variants and Adaptations of HMC

2.3.1 Riemannian Manifold Hamiltonian Monte Carlo

Euclidean Hamiltonian Monte Carlo provides a more informed proposal than random-walk Metropolis by using information from the target density (Gelman et al. 2013). The Hamiltonian Monte Carlo proposal is based on a combination of the step-size ϵ, the number of leapfrog steps L, the parameterization of the latent variable p, and the gradient of the target density. One way to consider Hamiltonian proposals is via the concept of distance traveled in the parameter space of θ. In Euclidean space, the distance is calculated via the L2-norm,

$$\begin{aligned} D(\theta, \theta + \gamma\theta) &= \sqrt{\gamma\theta \cdot \gamma\theta} \\ &= ||\gamma\theta||. \end{aligned} \tag{2.27}$$

If I consider a total distance of γ in (2.27), Euclidean Hamiltonian Monte Carlo assumes that the distance between the starting value and the proposal is approximately constant over a small distance. However, the gradients may change rapidly over even short distances (Calderhead 2011). One way to incorporate the rate of change in the gradient is to consider proposals using the *second* derivative.

Riemannian Manifold Hamiltonian Monte Carlo (RMHMC) extends EHMC by using a metric based on the second derivative to adjust the gradient based on the location in the parameter space (Girolami and Calderhead 2011). The key concepts regarding how the second derivative in Riemannian Hamiltonian Monte Carlo is linked to differential geometry are introduced here. In particular, the second order information is used to derive the shortest path through the parameter space based on the more general Riemannian geometry(Calderhead 2011).

2.3.1.1 Riemann Geometry

Bernhard Riemann was a German mathematician who made significant contributions to providing a general framework for geometry beyond Euclidean space (Laugwitz 2008). One of the principals of Riemannian geometry is that there exists a theory of surfaces that is independent of 3-dimensional Euclidean space(O'Neill 1997).

In $\mathcal{R}^3$ space, I define a dot product of two vectors $\mathbf{x}$ and $\mathbf{y}$ as

$$\mathbf{x} \cdot \mathbf{x} = |\mathbf{x}||\mathbf{x}| \cos \phi, \tag{2.28}$$

where ϕ is the angle between the two vectors. The dot product therefore provides information on the distance between points and the angle between them.

Abstract surfaces beyond 3-dimensions do not have dot products. However, Riemann conceived of a generalization of dot products called inner products that can be applied to abstract surfaces beyond 3-dimensional space (O'Neill 1997). An inner product between vectors $\mathbf{x}$ and $\mathbf{y}$ in arbitrary linear space M has the following properties (Jain, Ahmad, and Ahuja 1995):

1. Positive Definite: $< \mathbf{x}, \mathbf{x} > \ \geq 0$ and $< \mathbf{x}, \mathbf{x} >= 0$ iff $\mathbf{x} = \mathbf{0}$
2. Symmetry: $< \mathbf{x}, \mathbf{y} >=< \mathbf{y}, \mathbf{x} >$ for $\mathbf{x}, \mathbf{y} \in \mathbb{R}$
3. Linearity: $\alpha < \mathbf{x}, \mathbf{y} >= \alpha < \mathbf{x}, \mathbf{y} >$ and $< \mathbf{x} + \mathbf{z}, \mathbf{z} >=< \mathbf{x}, \mathbf{y} > + < \mathbf{y}, \mathbf{z} >$ where $\mathbf{x}, \mathbf{y}, \mathbf{z}$ are vectors in M and α is a constant.

The distance between two points $\boldsymbol{\theta}$ and $\boldsymbol{\theta} + d\boldsymbol{\theta}$ is considered in a k-dimensional space,

$$(\theta_1, ..., \theta_k), (\theta_1 + d\theta_1, ..., \theta_k + d\theta_k) \tag{2.29}$$

This distance is assumed to be in quadratic form,

$$g_{rs}\theta_r\theta_s, \tag{2.30}$$

with a summation following the Einstein convention (Barndorff-Nielsen, Cox, and Reid 1986)

$$\sum_{r,s=1}^{k} g_{rs} d\theta^r d\theta^s. \tag{2.31}$$

Here, g_{rs} is a collection of inner products in Riemannian space and is called a metric tensor (or simply, metric) (O'Neill 1997).

Although many metric tensors can be defined in the potentially high-dimensional space of $\boldsymbol{\theta}$ (Calderhead 2011), one natural consideration for use in HMC is the expected Fisher information matrix (Girolami and Calderhead 2011), where

$G(\boldsymbol{\theta}) = -E_{\boldsymbol{\theta}}\left[\nabla^2 \log f(\mathbf{y}|\boldsymbol{\theta})\right]$. Rao (1945) noted that $G(\boldsymbol{\theta})$ is a metric tensor in Riemannian space since it is positive definite and position-dependent. Further, Rao showed that the distance between two positions in the same probability space of θ, $f(\mathbf{y}|\boldsymbol{\theta})$ and $f(\mathbf{y}|\boldsymbol{\theta} + \delta\boldsymbol{\theta})$, is in quadratic form $\delta\boldsymbol{\theta}^T G(\boldsymbol{\theta})\delta\boldsymbol{\theta}$.

The general idea of Riemannian Hamiltonian Monte Carlo is to choose a metric $G(\boldsymbol{\theta})$ that provides information beyond the first-order gradient of the target density to produce more efficient proposals (Girolami and Calderhead 2011). In this introduction, I choose the expected Fisher information matrix to illustrate this method. However, in complicated models this particular metric may not be a feasible choose due to mathematical and computational complexities (M. J. Betancourt 2013).

2.3.1.2 Algorithm Development

In Euclidean Hamiltonian Monte Carlo, the distribution of the auxiliary momentum $\mathbf{p}$ is defined as a multivariate normal with covariance matrix $\mathbf{M}$ (2.16). The Hamiltonian equations specify the rate of change of $\boldsymbol{\theta}$ with respect to time,

$$\mathbf{p} = \mathbf{M}\frac{d\boldsymbol{\theta}}{dt} = \mathbf{M}\dot{\boldsymbol{\theta}}. \tag{2.32}$$

I calculate the L^2 norm of $\dot{\boldsymbol{\theta}}$

$$||\dot{\boldsymbol{\theta}}||_{\mathbf{M}} = \dot{\boldsymbol{\theta}}^T\mathbf{M}\dot{\boldsymbol{\theta}} = \mathbf{p}^T\mathbf{M}^{-1}\mathbf{p}. \tag{2.33}$$

The momentum $\mathbf{p}$ in Euclidean Hamiltonian Monte Carlo does not depend on $\boldsymbol{\theta}$ since $\mathbf{M}$ is constant. Riemannian Manifold Hamiltonian Monte Carlo replaces $\mathbf{M}$ with a position-dependent metric $G(\boldsymbol{\theta})$, such that $\mathbf{p} \sim N(0, G(\boldsymbol{\theta}))$. The distance of $\dot{\boldsymbol{\theta}}$ in Riemannian Manifold Hamiltonian Monte Carlo then depends on the position of $\boldsymbol{\theta}$ (Rao 1945)(Calderhead 2011),

$$||\dot{\boldsymbol{\theta}}||^2_{G(\boldsymbol{\theta})} = \dot{\boldsymbol{\theta}}^T G(\boldsymbol{\theta})\dot{\boldsymbol{\theta}} = \mathbf{p}^T G(\boldsymbol{\theta})^{-1}\mathbf{p}. \tag{2.34}$$

The Hamiltonian is revised to incorporate the metric $G(\boldsymbol{\theta})$(Girolami and Calderhead 2011),

$$\begin{aligned} H(\boldsymbol{\theta}, \mathbf{p}) &= -\log f(\mathbf{p}|\boldsymbol{\theta}) - \log f(\boldsymbol{\theta}) \\ &= \frac{1}{2}\log((2\pi)^k |G(\boldsymbol{\theta})|) + \frac{1}{2}\mathbf{p}^T G(\boldsymbol{\theta})^{-1}\mathbf{p} - \log f(\boldsymbol{\theta}). \end{aligned} \tag{2.35}$$

The dependence of momentum on $\boldsymbol{\theta}$ creates a more mathematically complex set of differential equations,

$$\begin{aligned} \frac{d\theta_i}{dt} &= \frac{\partial H}{\partial p_i} = \left[G(\boldsymbol{\theta})^{-1}\mathbf{p}\right]_i \\ \frac{dp_i}{dt} &= -\frac{\partial H}{\partial \theta_i} = \frac{\partial}{\partial \theta_i}\log f(\boldsymbol{\theta}) - \frac{1}{2}tr\left[G(\boldsymbol{\theta})^{-1}\frac{\partial G(\boldsymbol{\theta})}{\partial \theta_i}\right] + \frac{1}{2}\mathbf{p}^T G(\boldsymbol{\theta})^{-1}\frac{\partial G(\boldsymbol{\theta})}{\partial \theta_i}G(\boldsymbol{\theta})^{-1}\mathbf{p}. \end{aligned} \tag{2.36}$$

Since the momentum now depends on $\boldsymbol{\theta}$, the Newton-Stormer-Verlet Leapfrog integrator is no longer time reversible (i.e. $G(\boldsymbol{\theta}(t)) \neq G(\boldsymbol{\theta}(t+\epsilon))$) (Calderhead 2011). A generalized version of the Leapfrog integrator (Hairer, Lubich, and Wanner 2003) is able to preserve the reversibility required for MCMC. This generalized Leapfrog is used in Riemannian Manifold Hamiltonian Monte Carlo(Girolami and Calderhead 2011),

$$\begin{aligned} \mathbf{p}\left(t+\frac{\epsilon}{2}\right) &= \mathbf{p}(t) - \frac{\epsilon}{2}\nabla_{\boldsymbol{\theta}} H\left[\mathbf{p}\left(t+\frac{\epsilon}{2}\right), \boldsymbol{\theta}(t)\right] \\ \boldsymbol{\theta}(t+\epsilon) &= \boldsymbol{\theta}(t) + \frac{\epsilon}{2}\left\{\nabla_{\mathbf{p}} H\left[\mathbf{p}\left(t+\frac{\epsilon}{2}\right), \boldsymbol{\theta}(t)\right] + \nabla_{\mathbf{p}} H\left[\mathbf{p}\left(t+\frac{\epsilon}{2}\right), \boldsymbol{\theta}(t+\epsilon)\right]\right\} \\ \mathbf{p}(t+\epsilon) &= \mathbf{p}\left(t+\frac{\epsilon}{2}\right) - \frac{\epsilon}{2}\nabla_{\boldsymbol{\theta}} H\left[p\left(t+\frac{\epsilon}{2}\right), \boldsymbol{\theta}(t+\epsilon)\right]. \end{aligned} \tag{2.37}$$

Once a Riemannian metric $G(\boldsymbol{\theta})$ is defined, the generalized Leapfrog integrator is used to generate proposals for $\boldsymbol{\theta}$. The Metropolis-Hastings probability correction is retained from the Euclidean Hamiltonian Monte Carlo (2.26)(Girolami, Calderhead, and Chin 2009).

Compared to EHMC, RMHMC adds the overhead of additional computations including the Riemannian metric, its inverse, and implicit numerical methods for the generalized leapfrog. Fixed point iteration is one such numerical method that can be used for RMHMC (Girolami and Calderhead 2011).

Algorithm 4 Riemannian Manifold Hamiltonian Monte Carlo

```
procedure RMHMC($\boldsymbol{\theta}^{(0)}, \log f(\boldsymbol{\theta}), G(\boldsymbol{\theta}), N, \epsilon, L$)
    Calculate $\log f(\boldsymbol{\theta}^{(0)})$
    for $t = 1, ..., N$ do
        $\mathbf{p}^0 \leftarrow N(0, G(\boldsymbol{\theta}))$
        $\boldsymbol{\theta}^{(t)} \leftarrow \boldsymbol{\theta}^{(t-1)}, \tilde{\boldsymbol{\theta}} \leftarrow \boldsymbol{\theta}^{(t-1)}, \tilde{\mathbf{p}} \leftarrow \mathbf{p}^{(0)}$
        for $i = 1, ..., L$ do
            $\tilde{\boldsymbol{\theta}}, \tilde{\mathbf{p}} \leftarrow$ Generalized Leapfrog($\boldsymbol{\theta}(t), \mathbf{p}(t), \epsilon$)
        end for
        $\alpha = \min\left(1, \frac{H(\tilde{\boldsymbol{\theta}}, \tilde{\mathbf{p}})}{H(\boldsymbol{\theta}(t-1), \mathbf{p}^0)}\right)$
        With probability $\alpha$, $\boldsymbol{\theta}^{(t)} \leftarrow \tilde{\boldsymbol{\theta}}$ and $\mathbf{p}^{(t)} \leftarrow -\tilde{\mathbf{p}}$
    end for
    return $\boldsymbol{\theta}^{(1)} ... \boldsymbol{\theta}^{(N)}$
    function GeneralizedLeapfrog($\boldsymbol{\theta}(t), \mathbf{p}(t), \epsilon$)
        $\tilde{\mathbf{p}} \leftarrow \mathbf{p}(t) - \frac{\epsilon}{2} \nabla_{\boldsymbol{\theta}} H \left[\mathbf{p}\left(t + \frac{\epsilon}{2}\right), \boldsymbol{\theta}(t)\right]$
        $\tilde{\boldsymbol{\theta}} \leftarrow \boldsymbol{\theta}(t) + \frac{\epsilon}{2} \left\{\nabla_{\mathbf{p}} H \left[\boldsymbol{\theta}(t), \tilde{\mathbf{p}}\right] + \nabla_{\mathbf{p}} H \left[\boldsymbol{\theta}(t + \epsilon), \tilde{\mathbf{p}}\right]\right\}$
        $\tilde{\mathbf{p}} \leftarrow \tilde{\mathbf{p}} - \frac{\epsilon}{2} \nabla_{\boldsymbol{\theta}} H \left[\boldsymbol{\theta}(t + \epsilon), \tilde{\mathbf{p}}\right]$
    end function
end procedure
```

2.3.2 Algorithmic Variations of HMC

EHMC is now a standard MCMC algorithm for Bayesian data analysis, with robust implementations in STAN and PyMC software packages. Several variations of EHMC have been proposed to improve the algorithm's efficiency and usability. Since RMHMC is not available as a general purpose modeling algorithm at the time of this writing, I focus methodological development on EHMC.

This section provides perspective on current areas of research derived from the EHMC algorithm. From this section on, the EHMC algorithm is abbreviate to HMC for brevity.

2.3.2.1 No-U-Turn Sampler (NUTS)

Perhaps the most impactful algorithmic variation of HMC uses the No-U-Turn Sampler (NUTS) (Hoffman and Gelman 2014). NUTS reduces the practical complexities of implementing HMC by automatically selecting the number of Leapfrog steps L. NUTS was originally implemented in Stan, and has since been adapted for **PyMC**. HMC with NUTS automated parameter selection is the primary MCMC algorithm for both software packages (Gelman, Lee, and Guo 2015)(Salvatier, Wiecki, and Fonnesbeck 2016).

The goal of NUTS is to automatically determine when a sufficient number of leapfrog steps have occurred for a proposal to maximize efficient sampling. One of the ways that NUTS makes this determination is by calculating the distance between proposals and the current position $\theta^{(t)}$. This development assumes a single parameter θ. This distance is determined by the derivative of half the squared difference between the current position $\theta^{(t)}$ and a new position $\theta^{(t+1)}$ with respect to time,

$$\frac{d}{dt}\frac{(\theta^{(t+1)}-\theta^{(t)})\cdot(\theta^{(t+1)}-\theta^{(t)})}{2} = (\theta^{(t+1)}-\theta^{(t)})\cdot\frac{d}{dt}(\theta^{(t+1)}-\theta^{(t)}) = (\theta^{(t+1)}-\theta^{(t)})\cdot p^{(t)}, \tag{2.38}$$

where $p^{(t)}$ is the current momentum (latent variable). This represents the distance away from the original $\theta^{(t)}$ given an infinitesimally small amount of time. NUTS therefore suggests an algorithm that progresses through leapfrog iterations until the proposal $\theta^{(t+1)}$ reverses in direction towards the original $\theta^{(t)}$ (i.e. no U-turn).

The EHMC model with momentum p is specified,

$$f(\theta,p) \propto \exp(\log f(\theta) - \frac{1}{2}p\cdot p). \tag{2.39}$$

NUTS introduces an additional latent variable u for slice sampling such that

$$f(\theta,p,u) \propto \mathcal{I}\left[u \in (0, \exp(\log f(\theta) - \frac{1}{2}p\cdot p))\right], \tag{2.40}$$

where $\mathcal{I}$ is the indicator function. The conditional probabilities $f(u|\theta,p)$ and $f(\theta,p|u)$ are each uniform if $u \leq \exp(\log f(\theta) - \frac{1}{2}p\cdot p)$. In addition to u, each NUTS iteration generates a set of position-momentum states $\mathcal{B}$ and a subset of these states $\mathcal{C}$ to which transitions may occur without violating detailed balance. Here, $\mathcal{B}$ is built by iteratively doubling the number of leapfrog sets in positive and negative time, at random. Integrating backwards and forward in time preserves time reversibility. This process is repeated until the distance between the original state θ and new θ' decreases (i.e. a U-turn).

The steps in NUTS are designed to leave the resulting joint distribution $f(\theta, p, u, \mathcal{B}, \mathcal{C}|\epsilon)$ invariant. Note that the first 3 steps constitute a Gibbs sampler.

1. Sample $p \sim N(0, 1)$
2. Sample new latent variable $u \sim U[0, \exp(\log f(\theta^{(t)}) - \frac{1}{2}p \cdot p)]$
3. Sample the number of leapfrog steps from the joint distribution $p(\mathcal{B}, \mathcal{C}|\theta^{(t)}, p, u, \epsilon)$
4. Sample $\theta^{(t+1)}, p \sim T(\theta^{(t)}, p, \mathcal{C})$

   where $T(\theta^{(t)}, p, \mathcal{C})$ is a transition kernel satisfying

$$\frac{1}{|\mathcal{C}|} \sum_{(\theta,p)\in\mathcal{C}} T(\theta', p'|\theta, p, \mathcal{C}) = \frac{\mathcal{I}[(\theta', p') \in \mathcal{C}]}{|\mathcal{C}|}. \tag{2.41}$$

As a result, the posterior distribution can be written,

$$\begin{aligned} p(\theta, p|\mathcal{B}, \mathcal{C}, u, \epsilon) &\propto p(\mathcal{B}, \mathcal{C}|\theta, p, u, \epsilon)p(\theta, p|u) \\ &\propto p(\mathcal{B}, \mathcal{C}|\theta, p, u, \epsilon)\mathcal{I}[u \leq \exp(\log f(\theta) - \frac{1}{2}p \cdot p)] \\ &\propto \mathcal{I}[(\theta, p) \in \mathcal{C}], \end{aligned} \tag{2.42}$$

which demonstrates that the joint distribution (θ, p) is uniform over the elements of $\mathcal{C}$. Note that NUTS places restrictions on the joint density of $\mathcal{B}, \mathcal{C}$ to satisfy detailed balance and to ensure the invariance of $p(\theta, p, u, \mathcal{B}, \mathcal{C}|\epsilon)$,

The end result of the NUTS algorithm is a transition kernel defined such that the joint density $p(\theta, p, u, \epsilon|\mathcal{B}, \mathcal{C})$ is invariant. This ensures that the target density is also invariant. Reversible time is preserved by resampling u and p forwards and backwards in time until the trajectory begins to reverse direction or a probability state that is extremely low occurs.

The second tuning parameter ϵ is selected using largely heuristic calculations based on the acceptance probability for a standard HMC. Hoffman and Gelman (2014) propose a target average acceptance probability of 0.65. However, NUTS does not have an accept/reject step. As such, an alternate statistic is devised called H^{NUTS}, which can be interpreted as an average acceptance probability that HMC would have given during the final doubling iteration,

$$H^{NUTS} := \sum_{\theta,p \in \mathcal{B}^{final}} \min\left[1, \frac{p(\theta, p)}{p(\theta^{(i-1)}), p^{i,0}}\right], \tag{2.43}$$

with expectation $h^{NUTS} := \mathcal{E}_{\sqcup}[H_t^{NUTS}]$.

Despite the heuristic nature of the selection of ϵ, there is justification for an acceptance probability of 0.65 in high-dimensions (Beskos et al. 2010). Further, software that implements HMC with NUTS provides methods of adjusting the step size manually if desired.

The NUTS algorithm enables analysts to perform HMC without the need to manually select tuning parameters L and ϵ. As the primary HMC implementation in Stan and **PyMC**, NUTS can be considered a standard in modern MCMC practice.

2.3.2.2 Automatic or manual differentiation

In the development of HMC so far, I have assumed that the gradient function of the log posterior has been derived and is provided for the purpose of HMC programming. In real practice, however, analytically evaluating a gradient is often quite cumbersome. The challenge is especially severe in higher dimensional spaces. To overcome this difficulty, analysts typically resort to numerical methods for gradient calculation. While gradient approximation methods such as finite differencing are readily available, the accumulation of discretization error creates challenges in an HMC algorithm, which requires $L + 1$ gradient calculations per proposal.

Gradient computation algorithms with an accuracy to the computer's floating point error have been developed for HMC (Carpenter et al. 2015) and for **machine learning** algorithms such as deep learning (Theano Development Team 2016). These algorithms use a computational graph to translate an often complex expression into a network of fundamental operations. The result is a representation of the original expression as the composition of many low-level functions (e.g. $f = f_1 \circ f_2 \circ ...$). Once this graph is derived, the chain rule is used to calculate the gradient exactly.

While automated differentiation algorithms provide the benefit of automated scalability, they are associated with a small increase in computation time. When programmed efficiently, functions that directly calculate an analytically derived gradient can outperform automated algorithms. However, the benefits of flexibility and scalability in automated methods often outweigh the added computational burden. Users with sufficient technical expertise may be able to add custom functionality since many software implementations of HMC are open-source.

2.3.2.3 Stochastic Gradient Hamiltonian Monte Carlo

While NUTS addresses difficulties in HMC parameter selection, Chen, Fox, and Guestrin (2014) propose Stochastic Gradient Monte Carlo (SGHMC) to address the computational burdens of HMC. SGHMC replaces the actual gradient of the log posterior with a noisy estimate derived from machine learning algorithms. This algorithmic variation of HMC addresses "big data" problems where gradient calculations are not computationally feasible.

Standard HMC calculates the potential energy function or log posterior in (2.10) based on the entire dataset $\mathcal{D}$. One way to decrease the computational burden is to reduce the size of the data. A "noisy" estimated gradient is computed from a simple random sample of the data $\widetilde{\mathcal{D}}$ called a mini-batch (Robbins and Monro 1951),

$$\begin{aligned}\nabla \widetilde{U}(\boldsymbol{\theta}) &= \frac{|\mathcal{D}|}{|\widetilde{\mathcal{D}}|}\nabla \log f(\boldsymbol{\theta}) \\ \nabla \widetilde{U}(\boldsymbol{\theta}) &\approx \nabla U(\boldsymbol{\theta}) + N(0, V(\boldsymbol{\theta})).\end{aligned} \tag{2.44}$$

The noisy gradient approximates the actual gradient via an appeal to the central limit theorem, where V is the covariance of the noise of the stochastic gradient. The most straightforward application of SGHMC replaces the gradient of the log posterior directly with (2.44). This algorithm requires an additional Metropolis-Hastings correction *before* discretization since the joint density $f(\boldsymbol{\theta}, \mathbf{p})$ is not invariant to this transformation. The MH step requires all of the data for computation, reducing the efficiency of this algorithm.

A balance between the efficiency of a less computationally intensive gradient and numerous, costly MH steps could not be found. Simulation studies of this version of SGHMC showed either poorly behaved trajectories or inefficient computations. A more robust algorithm is proposed based on second-order Langevin dynamics(Wang and Uhlenbeck 1945). This method adds a "friction" term $BM^{-1}\mathbf{p}dt$ to diminish the noise introduced by noisy gradient, where $B = \frac{1}{2}\epsilon V(\boldsymbol{\theta})$. The result is a revised set of Hamiltonian equations

$$\begin{aligned} d\boldsymbol{\theta} &= \mathbf{M}^{-1}\mathbf{p}\ dt \\ dp &= -\nabla U(\boldsymbol{\theta})dt + N(0, 2B(\boldsymbol{\theta})dt).\end{aligned} \tag{2.45}$$

This friction term offsets the impact of a large divergence in *dp*. The resulting time evolution of the Hamiltonian equation can be be described by the Fokker-Planck equation, which describes the time evolution of a system when subject to friction forces. The result is a stationary distribution of $H(\boldsymbol{\theta}, \mathbf{p})$, as desired. Although the stochastic gradient HMC produces a stationary distribution, the simulation is not time reversible. The effects of the lack of time reversibility have been left for future study. Initial simulation experiments on SGHMC focused on predictive applications common in machine learning. Additional research would need to be performed to determine the feasibility of using SGHMC in statistical inference.

2.4 A General Process for Fitting Statistical Models using HMC

The major steps required to fit a statistical model are summarized in Figure 2.3. Following the steps illustrated in the diagram, one could generate HMC samples with user-specified posterior and gradient functions, by using the `hmc` function in the **hmclearn** package.

The first set of steps require the specification of the log posterior as the sum of the log likelihood and log prior, at least to a numerical constant, $\log f(\boldsymbol{\theta}|\mathbf{y}) = \log f(\mathbf{y}|\boldsymbol{\theta}) + \log f(\boldsymbol{\theta})$, where $\boldsymbol{\theta}$ is the parameter of interest and $\mathbf{y}$ is the data.

If the support of $\boldsymbol{\theta}$ is restricted, a transformation must be applied. For example, if $\boldsymbol{\theta} \in (0, \infty)$, then a log transformation may be applied such that $\log \boldsymbol{\theta} \in \mathbb{R}$. The Jacobian must be derived and applied correctly in such cases.

Once the full log posterior function is specified (again, to a normalizing constant), an R function must be provided to return the result for simulated values of $\boldsymbol{\theta}$. The user may reference separate functions for the log likelihood and log prior, if so desired. Only the log posterior function must be provided in the software.

Similarly, a function must also be provided to calculate the gradient of the log posterior. The preferred approach is to manually derive the gradient, and then program the exact gradient function for the software. Alternatively, users may opt to use an automated gradient calculation. However, the gradients provided must be calculated exactly. Approximations such as finite differencing are inadequate for HMC.

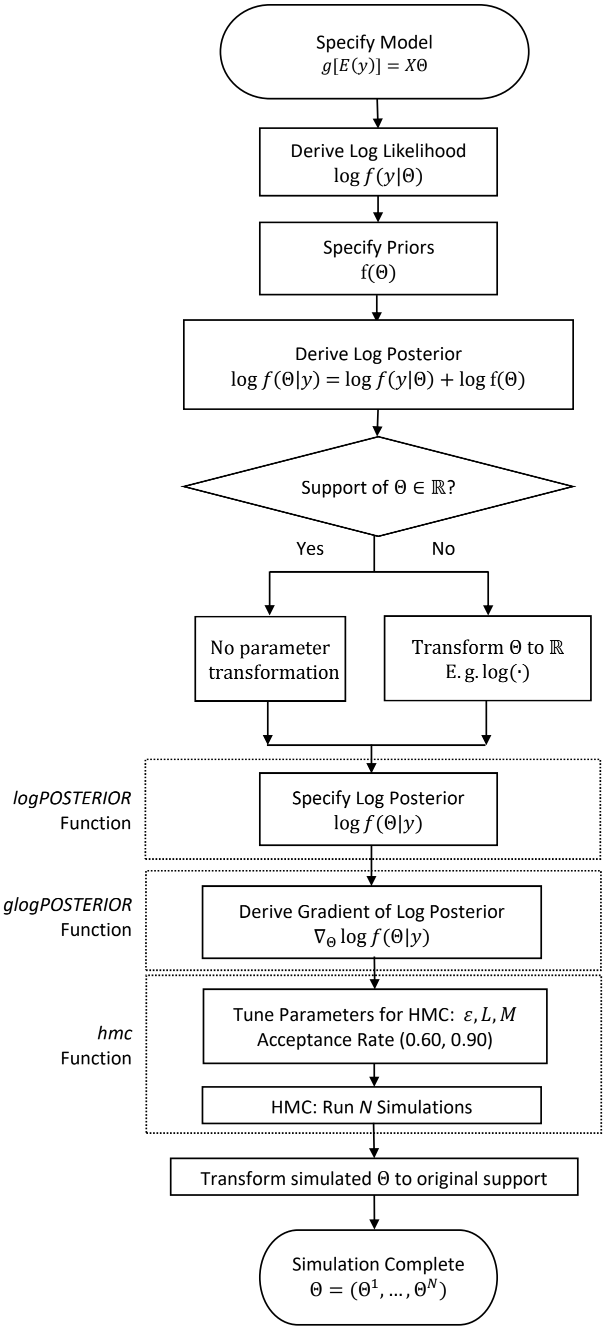
Specify Model
$g[E(y)] = X\Theta$
Derive Log Likelihood
$\log f(y|\Theta)$
Specify Priors
$f(\Theta)$
Derive Log Posterior
$\log f(\Theta|y) = \log f(y|\Theta) + \log f(\Theta)$
Support of $\Theta \in \mathbb{R}$?
Yes
No
No parameter transformation
Transform Θ to $\mathbb{R}$
E. g. $\log(\cdot)$
logPOSTERIOR Function
Specify Log Posterior
$\log f(\Theta|y)$
glogPOSTERIOR Function
Derive Gradient of Log Posterior
$\nabla_\Theta \log f(\Theta|y)$
hmc Function
Tune Parameters for HMC: ε, L, M
Acceptance Rate (0.60, 0.90)
HMC: Run N Simulations
Transform simulated Θ to original support
Simulation Complete
$\Theta = (\Theta^1, \ldots, \Theta^N)$

Figure 2.3: Major steps of HMC implementation

Once the data and functions are available, the HMC algorithm must be tuned to the application. It is generally a good practice to set ϵ to a smaller value relative to the magnitude of the parameter of interest. A smaller ϵ results in closer approximations and thus higher acceptance rates. But a small ϵ must be coupled with a large L to ensure the trajectory length ϵL is large enough to move the simulated parameter to a distant point in the distribution. On the other hand, if ϵL is too large the trajectory is likely to circle back, causing waste in simulation. To tune ϵ and L is to find the right combination of these values. One usually does so by monitoring the acceptance rate. (R. Neal 2011) suggested that the optimal acceptance rate is approximately 65%.

At the same time, it is often helpful to examine the trace plots of the MCMC samples for signs of autocorrelation. Slow-moving chains with stronger autocorrelation often indicate insufficient ϵL. While ϵ and L can be tuned jointly, most analysts choose to select the step size first, then under a given step size, they fine-tune the number of steps per leapfrog L.

Additional adjustments may be made to the tuning parameters beyond these basic steps. For example, different values of ϵ for each of the k parameters in $\boldsymbol{\theta}$ can be chosen to increase the sampling efficiency. The parameter for the number of steps L must be a natural number. However, randomly chosen L could be used to guard against periodicity of the Markov chain. The step size ϵ may also be randomized. A useful algorithm known as the No U-Turn Sampler (NUTS) automatically selects L for each sample; NUTS is a commonly used alternative to manual parameter tuning (Hoffman and Gelman 2014).

2.4.1 hmclearn: A Flexible Computational Tool

I present an R package **hmclearn** to provide users with the software tools to learn the intricacies of the HMC, through explicit specification of log posterior and gradient functions, as well as parameter tuning. It is designed to give user a hands-on experience for implementing HMC analysis for a broad class of statistical models. Once users have understood and mastered the essential HMC steps, they could go on to write their own code for specific applications. And while one of the major purposes of developing this package was to help teach HMC to new users, **hmclearn** facilitates fitting a wide variety of statistical models. As such, **hmclearn** can also be used as general purpose software to use HMC for research purposes.

The core function in **hmclearn** is `hmc`, which is a general-purpose function for MCMC sample generation using the EHMC method (shortened to simply HMC for the remainder of this chapter). This function takes user-defined log posterior and gradient functions as inputs and produces MCMC samples. Here, an explicit specification of prior $f(\boldsymbol{\theta})$ is not required as an input function. Instead, I provide users with the capability to define their log posterior $\log f(\boldsymbol{\theta}|\boldsymbol{y}) = \log f(\boldsymbol{y}|\boldsymbol{\theta}) + \log f(\boldsymbol{\theta})$, which includes $f(\boldsymbol{\theta})$. Such a design reduces the number of required input functions, while preserving users' flexibility in choosing different priors.

Other input parameters to `hmc` include the number of samples N, the step size ϵ, the number of leapfrog steps L, and the Mass matrix $\mathbf{M}$. These are the essential elements to start an HMC simulation, but the user will typically need to adjust at least some of these parameters to tailor the simulation to their specific applications. Users are required to provide their own starting values for $\boldsymbol{\theta}$ when using the `hmc` function for their own applications. Examples of log posterior and gradient functions are provided in **hmclearn** for various generalized linear mixed effect models, which can be used as templates for less standard models.

Running multiple MCMC chains is often desirable to determine if each chain converges to the same distribution of $\boldsymbol{\theta}$. Since modern computers almost universally have multiple core processors, parallel processing can be an efficient way to run multiple chains at the same time. To that end, **hmclearn** includes parameters to enable parallel processing as well as multiple chains.

Finally, a variety of Bayesian graphical functions are provided based on the `bayesplot` package (Gabry and Mahr 2016). Some of the functionality directly incorporated in **hmclearn** include trace plots, histograms, density plots, and credible interval plots. The integrated functions comprise the core diagnostic plotting functions typical for MCMC applications. Additional diagnostics can be programmed directly or called based on the output of the `hmc` function. This package along with source code and vignettes are available on CRAN at https://cran.r-project.org/web/packages/hmclearn/index.html.

2.4.2 Fitting a Mixed Effects Model in HMC

A mixed effects model is used to illustrate fitting standard statistical models using HMC. While standard, these models can be difficult to fit using standard methods. Further, more complex models including generalized additive models can be expressed in mixed effect model form.

A random intercept mixed effects model can be specified as

$$g[E(\mathbf{y}_i|u_i)] = \mathbf{X}_i\boldsymbol{\beta} + \mathbf{z}_i u_i,$$

for $i = 1, ..., n$ subjects, where each subject's response vector $\mathbf{y}_i = (y_{i1}, ..., y_{id})^T$ contains $j = 1, ..., d$ observations. Each subject has an individual random intercept parameter u_i, where $\mathbf{u} = (u_1, ..., u_n)^T$. The fixed effects design matrix $\mathbf{X}_i = (\mathbf{x}_{i1}^T, ..., \mathbf{x}_{id}^T)^T \in \mathbb{R}^{d\times(q+1)}$, where the jth row of $\mathbf{X}_i$ contains the $q+1$ covariate values of that observation, including a global intercept. The fixed effects regression coefficients for q covariates and a global intercept are a vector $\boldsymbol{\beta} = (\beta_0, ..., \beta_q)^T$. The random intercept vector is $\mathbf{z}_i = (z_{i1}, ..., z_{id})^T = \mathbf{1}_d$. The distribution of $\mathbf{y}_i$ conditional on u_i is Poisson with log link function, where $\log[E(\mathbf{y}_i|u_i)] = \mathbf{X}_i\boldsymbol{\beta} + \mathbf{z}_i u_j$.

The subject-level response vectors are combined in a single vector, $\mathbf{y} = (\mathbf{y}_i^T, ..., \mathbf{y}_n^T)^T \in \mathbb{R}^{nd\times 1}$. The full fixed effects design matrix for all subjects is $\mathbf{X} = (\mathbf{X}_1, ..., \mathbf{X}_n)^T \in \mathbb{R}^{nd\times(q+1)}$, and the random effects design matrix is $\mathbf{Z} = \mathbf{I}_n \otimes \mathbf{1}_d \in \mathbb{R}^{nd\times n}$. The log likelihood for the Poisson mixed effects model, omitting constants, is

$$\log f(\mathbf{y}|\mathbf{X}, \mathbf{Z}, \boldsymbol{\beta}, \mathbf{u}) \propto -\mathbf{1}_{nd}^T \left[e^{\mathbf{x}_{ij}^T\boldsymbol{\beta}+z_{ij}u_i}\right]_{nd\times 1} + \mathbf{y}^T(\mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\mathbf{u}),$$

where $\boldsymbol{\beta}$ is the fixed effects coefficient vector, u_i is the random intercept, and $\left[e^{\mathbf{x}_{ij}^T\boldsymbol{\beta}+z_{ij}u_i}\right]_{nd\times 1}$ is an $nd \times 1$ vector $\forall i = 1, \ldots, n$ and $j = 1, \ldots d$. I specify multivariate normal priors $\boldsymbol{\beta}|\sigma_\beta^2 \sim N(0, \sigma_\beta^2\mathbf{I})$ and $\mathbf{u} \sim N(0, \mathbf{G})$, where σ_β^2 is a hyperparameter set by the analyst and $\mathbf{G}$ is parameterized for efficient Bayesian computation.

Mixed effect models are types of hierarchical models (Gelman et al. 2013). I parameterize the covariance matrix of $\mathbf{G}$ for efficient sampling of hierarchical models such that $\mathbf{G}^{1/2} := \lambda\mathbf{I}\boldsymbol{\tau}$, where $\boldsymbol{\tau} = (\tau_1, ..., \tau_n)^T \sim N(0, \mathbf{I}_n)$ (M. J. Betancourt and Girolami 2013). For λ, I assign a 2-parameter half-t prior per the recommendation of (Gelman 2006) for hierarchical models.

One final parameter transformation is necessary before applying HMC. Since the support of λ is $(0, \infty)$, I apply a logarithmic transformation to expand the support to $\mathbb{R}$. The result is

$$\xi = \log\lambda, \quad \lambda = g^{-1}(\xi) = e^\xi,$$

$$f(\xi|a, b) \propto \left(1 + \frac{1}{\nu_\xi}\left(\frac{e^\xi}{A_\xi}\right)^2\right)^{-(\nu_\xi+1)/2} e^\xi,$$

$$\log f(\xi|a, b) \propto -\frac{\nu_\xi + 1}{2}\log\left(1 + \frac{1}{\nu_\xi}\left(\frac{e^\xi}{A_\xi}\right)^2\right) + \xi,$$

where ν_ξ and A_ξ are hyperparameters set by the analyst.

With the log likelihood and priors defined, I specify the log posterior, omitting constants, as

$$\log f(\boldsymbol{\beta}, \boldsymbol{\tau}, \xi | \mathbf{y}, \mathbf{X}, \mathbf{Z}, \sigma_\beta^2, \nu_\xi, A_\xi) \propto -\mathbf{1}_{nd}^T \left[e^{\mathbf{x}_{ij}^T \boldsymbol{\beta} + e^\xi z_{ij} \tau_i} \right] + \mathbf{y}^T (\mathbf{X}\boldsymbol{\beta} + e^\xi \mathbf{Z}\boldsymbol{\tau}) - \frac{\boldsymbol{\beta}^T \boldsymbol{\beta}}{2\sigma_\beta^2} - \frac{\nu_\xi + 1}{2} \log \left(1 + \frac{1}{\nu_\xi} \left(\frac{e^\xi}{A_\xi} \right)^2 \right) + \xi - \frac{1}{2} \boldsymbol{\tau}^T \boldsymbol{\tau}.$$

The parameters of interest are defined as $\boldsymbol{\theta} := (\beta_0, ..., \beta_q, \tau_1, ..., \tau_n, \xi)^T$, where $k = q + n + 2$. To fit this model using `hmc`, the user must provide a function for the log posterior where the first function parameter is a vector for the parameters of interest $\boldsymbol{\theta}$. Additional function parameters can be included for the data and hyperparameters. An example log posterior function for this model and specification of priors is included in **hmclearn**.

Writing the Hamiltonian function where $\mathbf{p} \sim N_k(0, \mathbf{M})$ for the mixed effects regression model is straightforward once the log posterior is developed,

$$H(\boldsymbol{\theta}, \mathbf{p}) = H(\boldsymbol{\beta}, \boldsymbol{\tau}, \xi, \mathbf{p}) \propto \log f(\boldsymbol{\beta}, \boldsymbol{\tau}, \xi | \mathbf{y}, \mathbf{X}, \mathbf{Z}, \sigma_\beta^2, \nu_\xi, A_\xi) + \frac{1}{2} \mathbf{p}^T \mathbf{M}^{-1} \mathbf{p}.$$

With the Hamiltonian function explicitly defined, I write the Hamiltonian equations for this particular model. The leapfrog algorithm is used to find a discrete approximation to the solutions of the Hamiltonian Equations. The steps of the leapfrog algorithm are integrated directly with `hmc` in a self-contained function. this function requires, as an input, a separate standalone function that returns a vector for the gradient of the log posterior. As with the log posterior function, the first function parameter must be a vector for the parameter of interest $\boldsymbol{\theta}$. An example gradient function for this model is also included in **hmclearn**,

$$\nabla_{\boldsymbol{\beta}} \log f(\boldsymbol{\beta}, \xi, \boldsymbol{\tau} | \mathbf{y}, \mathbf{X}, \mathbf{Z}, \sigma_\beta^2, \nu_\xi, A_\xi) \propto \mathbf{X}^T \left(- \left[e^{\mathbf{x}_{ij}^T \boldsymbol{\beta} + e^\xi z_{ij} \tau_i} \right]_{nd \times 1} + \mathbf{y} \right) - \boldsymbol{\beta} / \sigma_\beta^2,$$

$$\nabla_\xi \log f(\boldsymbol{\beta}, \xi, \boldsymbol{\tau} | \mathbf{y}, \mathbf{X}, \mathbf{Z}, \sigma_\beta^2, \nu_\xi, A_\xi) \propto e^\xi \boldsymbol{\tau}^T \mathbf{Z}^T \left(- \left[e^{\mathbf{x}_{ij}^T \boldsymbol{\beta} + e^\xi z_{ij} \tau_i} \right]_{nd \times 1} + \mathbf{y} \right) - \frac{\nu_\xi + 1}{1 + \nu_\xi A_\xi^2 e^{-2\xi}} + 1,$$

$$\nabla_{\boldsymbol{\tau}} \log f(\boldsymbol{\beta}, \xi, \boldsymbol{\tau} | \mathbf{y}, \mathbf{X}, \mathbf{Z}, \sigma_\beta^2, \nu_\xi, A_\xi) \propto e^\xi \mathbf{Z}^T \left(- \left[e^{\mathbf{x}_{ij}^T \boldsymbol{\beta} + e^\xi z_{ij} \tau_i} \right]_{nd \times 1} + \mathbf{y} \right) - \boldsymbol{\tau}.$$

Everything that is required to solve the Hamiltonian equations via the leapfrog algorithm and generate samples from the distribution of $f(\boldsymbol{\theta})$ is now available. The main `hmc` function handles the details of the HMC sample generation process for the user. Additional programming details are provided with the **hmclearn** package, including detailed vignettes with more examples.

This example comes from a study on gopher tortoises (Ozgul et al. 2009) (Fox, Negrete-Yankelevich, and Sosa 2015) (Bolker 2018). The mortality of the tortoise populations is measured in the number of shells, the dependent variable. I estimate the association of the number of shells to year (2004, 2005, 2006) and seroprevalence to Mycoplasma agassizii. The random effects are intercepts for each of $n = 10$ sites in Florida. Each site has $d = 3$ observations, one for each year. The fixed effects are a global intercept, two indicator variables for the three years, and seroprevalence. The poisson mixed effects model formulation for this application is

$$\begin{aligned}\log[E(\mathbf{shells})] \propto \sum_{i=1}^{10}\sum_{j=1}^{3} \Big[-e^{[1, I(2005)_{ij}, I(2006)_{ij}, \text{prev}_{ij}]\boldsymbol{\beta} + e^{\xi} z_{ij}\tau_i} + \\ y_{ij}\left([1, I(2005)_{ij}, I(2006)_{ij}, \text{prev}_{ij}]\boldsymbol{\beta} + e^{\xi} z_{ij}\tau_i\right)\Big] - \\ \frac{\boldsymbol{\beta}^T\boldsymbol{\beta}}{2\sigma_\beta^2} - \frac{\nu_\xi + 1}{2}\log\left(1 + \frac{1}{\nu_\xi}\left(\frac{e^{\xi}}{A_\xi}\right)^2\right) + \xi - \frac{1}{2}\boldsymbol{\tau}^T\boldsymbol{\tau},\end{aligned} \tag{2.46}$$

where $\mathbf{y} := (\mathbf{shells}_1, ..., \mathbf{shells}_{10})^T$ and $\mathbf{shells}_i = (\text{shells}_1, \text{shells}_2, \text{shells}_3)^T$. The fixed effects design matrix is composed from $\mathbf{x}_{ij}^T = [1, I(2005)_{ij}, I(2006)_{ij}, \text{prev}_{ij}]$, and the random effects design matrix from $z_{ij} = 1$ for site i and 0 otherwise, for all observations $j = 1, 2, 3$.

To fit this model using `hmc`, the initial values of $\boldsymbol{\theta}$ must first be specified in a vector of length $k = 15$. I use the default hyperparameters for the sample log posterior and gradient functions in **hmclearn**, such that $\sigma_\beta^2 = 1e3, \nu_\xi = 1,$and $A_\xi = 25$. The step sizes are set to values for this particular model and data set as part of the tuning process. %The log posterior and gradient functions are based on the likelihood and prior choices in this example.

The model takes a few seconds to run on a modern pc. Users have a number of options to summarize and visualize the HMC samples. The generic `summary` function provides quantiles from the posterior samples in a table. Many data visualization options are available through direct integration with the `bayesplot` package (Gabry and Mahr 2016). Graphical options for visualizing the posterior samples include histograms, density plots, and credible interval plots. General MCMC diagnostics such as trace plots, autocorrelation plots, and $\hat{R}$ statistics (Gelman and Rubin 1992) are also readily available. Additional customized analyses can be performed using the posterior sample output from `hmc`.

The design matrices X and Z must be setup for *hmc*. The fixed effects matrix X contains a global intercept and covariates for year 2005 *factor.year.2005*, year 2006 *factor.year.2006*, and Seroprevalence *prev*.

A random intercept is generated for each of the 10 sites in the dataset and stored as a block diagonal matrix in Z.

```
R> data(Gdat)
R>
R> ##########
R> # block diagonal
R> Zi.lst <- split(rep(1, nrow(Gdat)), Gdat$Site)
R> Zi.lst <- lapply(Zi.lst, as.matrix)
R> Z <- Matrix::bdiag(Zi.lst)
R> Z <- as.matrix(Z)
R> X <- model.matrix(~ factor(year), data=Gdat)
R> X <- cbind(X, Gdat$prev)
R> colnames(X)[ncol(X)] <- "prev"
R> colnames(X) <- make.names(colnames(X))
R> colnames(X)[1] <- "intercept"
R> y <- Gdat$shells
```

The log posterior and gradient functions are based on the likelihood and prior choices in this example.

```
R> glmm_poisson_posterior <- function (theta, y, X, Z, n, nrandom = 1,
+                                        nuxi = 1, Axi = 25, sig2beta = 1000)
+  {
+    Z <- as.matrix(Z)
+    p <- ncol(X)
+    beta_param <- theta[1:p]
+    tau_param <- theta[(p + 1):(p + n * nrandom)]
+    xi_param <- theta[(p + n * nrandom + 1):(p + n * nrandom +
+        nrandom)]
+    Dhalf <- diag(exp(xi_param), nrandom, nrandom)
+    L <- diag(nrandom)
+    LDhalf <- L %*% Dhalf
+    LDhalf_block <- kronecker(diag(n), LDhalf)
+    u_param <- LDhalf_block %*% tau_param
+    XZbetau <- X %*% beta_param + Z %*% u_param
+    log_likelihood <- -sum(exp(XZbetau)) + y %*% XZbetau
+    log_beta_prior <- -1/2 * t(beta_param) %*% beta_param/sig2beta
+    log_tau_prior <- -1/2 * t(tau_param) %*% tau_param
+    log_xi_prior <- -(nuxi + 1)/2 * log(1 + 1/nuxi * exp(2 *
+        xi_param)/Axi^2)
+    result <- log_likelihood + log_beta_prior + log_tau_prior +
+        sum(log_xi_prior)
+    return(as.numeric(result))
+ }
R>
R> g_glmm_poisson_posterior <- function(theta, y, X, Z, n, nrandom=1,
+                                         nuxi=1, Axi=25, sig2beta=1e3) {
+   Z <- as.matrix(Z)
+   p <- ncol(X)
+
+   # extract parameters from theta vector
+   beta_param <- theta[1:p]
+   tau_param <- theta[(p+1):(p+n*nrandom)]
+
+   # diagonal of G matrix
+   xi_param <- theta[(p+n*nrandom+1):(p+n*nrandom+nrandom)]
+
+   # reconstruct G LDLT decomposition
+   Dhalf <- diag(exp(xi_param), nrandom, nrandom)
+
+   L <- diag(nrandom)
+
+   LDhalf <- L %*% Dhalf
+   LDhalf_block <- kronecker(diag(n), LDhalf)
+
+   # u is deterministic function of xi and tau
+   u_param <- LDhalf_block %*% tau_param
+
+   XZbetau <- X %*% beta_param + Z %*% u_param
+
+   # block L and xi for xi gradient
+   L_block <- kronecker(diag(n), L)
+   Dhalf_block <- kronecker(diag(n), Dhalf)
+
+   # gradient
+   g_beta <- -t(X) %*% (exp(XZbetau) - y)- (beta_param)/sig2beta
+
+   # tau gradient
+   g_tau <- -t(LDhalf_block) %*% t(Z) %*% (exp(XZbetau) - y) - tau_param
+
+   # gradient for xi using matrix algebra
+   zero_v <- rep(0, nrandom)
+   g_xi <- sapply(seq_along(1:nrandom), function(jj) {
+     zv <- zero_v
+     zv[jj] <- 1
+     bd <- kronecker(diag(n), diag(zv, nrandom, nrandom))
+     - t(L_block %*% bd %*% Dhalf_block %*% tau_param) %*% t(Z) %*%
+       (exp(XZbetau) - y)
+   })
+   g_xi <- g_xi - (nuxi + 1) / (1 + nuxi*Axi^2 * exp(-2*xi_param)) + 1
+
+   g_all <- c(as.numeric(g_beta),
+              as.numeric(g_tau),
+              g_xi)
+
+   return(g_all)
+ }
```

With the dependent variable and design matrices defined, I run HMC for the Poisson mixed effects model. Initial values are set to zero and default hyperparameters are selected.

```
R> N <- 2e3
R>
R> set.seed(412)
```

```
R> initvals <- c(rep(0, 4),
+                rep(0, 10),
+                0)
R>
R> eps_vals <- c(3e-2, 3e-2, 3e-2, 1e-3, rep(1e-1, 10), 3e-2)
R>
R> fm3_hmc <- hmc(N = N, theta.init = initvals,
+                    epsilon = eps_vals, L = 10,
+                    logPOSTERIOR  = glmm_poisson_posterior,
+                    glogPOSTERIOR  = g_glmm_poisson_posterior,
+                    varnames=c(colnames(X),
+                         paste0("tau", 1:ncol(Z)), "xi"),
+                    param=list(y = y, X=X, Z=Z, n=10),
+                    chains=2, parallel=FALSE)
```

```
R> summary(fm3_hmc, burnin=200, probs=c(0.025, 0.5, 0.975))
```

```
Summary of MCMC simulation

                          2.5%         50%        97.5%      rhat
intercept        -1.129909142 -0.08752398  0.692578094 0.9997424
factor.year.2005 -1.372014112 -0.66698791 -0.003595214 0.9998643
factor.year.2006 -1.012942877 -0.38332103  0.214769142 1.0010172
prev              0.006494779  0.02331707  0.039944877 0.9999076
tau1             -2.410554813 -0.77997020  0.817336772 0.9997693
tau2             -1.729833796 -0.18396657  1.220663444 0.9997559
tau3             -1.983026482 -0.56865490  0.806617926 1.0001245
tau4             -0.568423528  0.73061452  2.175718706 1.0024635
tau5             -1.610232724 -0.10697694  1.196162525 0.9997881
tau6             -0.560222899  1.11240129  2.487083132 1.0000637
tau7             -1.046858556  0.23629673  1.463913025 1.0001180
tau8             -1.660281510 -0.19001017  1.089851556 1.0005547
tau9             -0.752246407  0.92311776  2.312896544 1.0010893
tau10            -2.503686065 -0.99581556  0.489402086 0.9997290
xi               -2.587503507 -0.36255053  0.554016427 1.0032564
```

In this example, the posterior estimates are comparable to frequentist estimates. I use the `lme4` package (Bates et al. 2007) to provide frequentist parameter estimates as a comparison to HMC.

```
R> fm3 <- glmer(shells~prev+factor(year)+(1|Site),
+               family=poisson,data=Gdat,
+               control=glmerControl(optimizer="bobyqa",
+                           check.conv.grad=.makeCC("warning",0.05)))
R>
R> coef(summary(fm3))
```

```
                    Estimate  Std. Error    z value    Pr(>|z|)
(Intercept)      -0.05779059 0.397498868 -0.1453856 0.884406472
prev              0.02230054 0.007714978  2.8905518 0.003845661
factor(year)2005 -0.65368457 0.357270249 -1.8296642 0.067300174
factor(year)2006 -0.37351125 0.322775316 -1.1571865 0.247196154
```

Next, I store frequentist fixed effects estimates in R variables.

```
R> freqvals_fixed <- c(fixef(fm3))
R> freqvals_fixed <- freqvals_fixed[c(1, 3, 4, 2)]
```

I also compare the random effects parameter estimates with **lme4**. I apply the linear transformation back to **u** for comparison.

```
R> u.freq <- ranef(fm3)$Site[, 1]
R> lambda.freq <- sqrt(VarCorr(fm3)$Site[1])
R>
R> # transform parameters back to original scale
R> fm3_hmc$thetaCombined <- lapply(fm3_hmc$thetaCombined, function(xx) {
+    tau_mx <- as.matrix(xx[, grepl("tau", colnames(xx))])
+    u_mx <- tau_mx * exp(xx[, "xi"])
+    u_df <- as.data.frame(u_mx)
+    colnames(u_df) <- paste0("u", 1:ncol(u_df))
+    xx <- cbind(xx, u_df, exp(xx[, "xi"]))
+    colnames(xx)[ncol(xx)] <- "lambda"
+    xx
+ })
```

The frequentist estimates for fixed effects are close to HMC in this example, as shown in Figure 2.4.

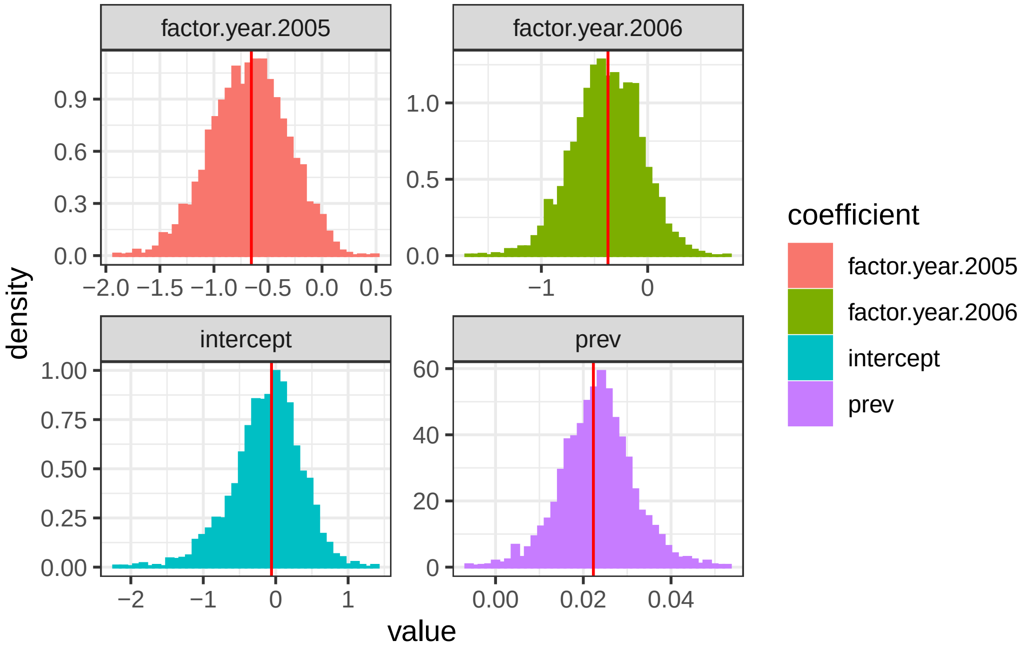

Figure 2.4: Comparison of frequentist and Bayesian fit of the fixed effect parameters from the mixed effects model example from hmclearn

```
R> diagplots(fm3_hmc, burnin=200, comparison.theta = freqvals_fixed,
+            cols=1:4)
```

The random effects parameters are also aligned with frequentist estimates, as shown in Figure 2.5.

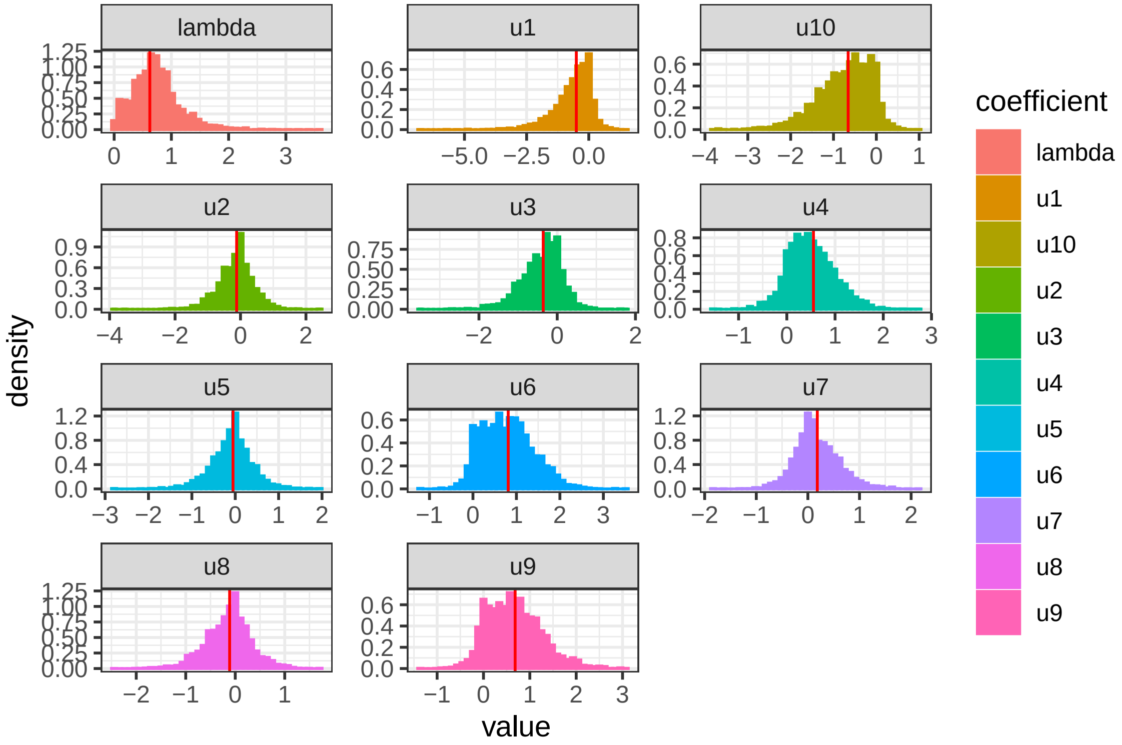

Figure 2.5: Comparison of frequentist and Bayesian fit of the random effect parameters from the mixed effects model example from hmclearn

CHAPTER 3

Fitting Non-Standard Statistical Models Using HMC

The Bayesian framework developed in the previous chapter is beneficial for standard statistical models as well as complex, non-standard models. For standard statistical models, this framework facilitates applications to high-dimensional datasets. Complex non-standard statistical models, however, present additional challenges to analysts who want to use HMC to fit such models. The nature of these challenges range from the practical, such as tuning and gradient derivations, to the theoretical, namely the mathematical parameterization development required to take advantage of HMC's strengths and minimize HMC's limitations.

In this chapter, a Bayesian parameterization is developed for a specific class of non-standard models called Generalized Additive Models (GAM). The variant of GAM's developed here estimates multiple responses as well as the subject-level correlation between responses. While multiple outcomes may be estimated independently, this approach neglects the potential significance of the correlation of measures within the same individual (H. Liu, Tu, and others 2012). This type of model is particularly relevant to biostatistics, where multiple outcomes are often measured simultaneously.

The innovations presented in this chapter provide statisticians with methodological development necessary for complex models, as well as the tools for applying HMC in practical research, including

1. A description of certain limitations of HMC in fitting complex models, particularly with respect to estimating correlated parameters,

2. Methodological development of parameterization specifically to take advantage of the strengths while accounting for limitations of HMC,
3. The empirical Bayesian nature of priors specified for computational efficiency and its implications for analysts ,
4. Example parameterization development for a class of complex, non-standard statistical models, and
5. A general purpose software package in the familiar `R` language to efficiently fit many standard and non-standard models using HMC with few technical barriers.

To summarize, the intent of this section is to provide statisticians interested in using HMC for complex models with strategies to design their own parameterization for efficient model fitting. These strategies include methodological development for parameterization as well as practical tools for fitting non-standard models in practice. While the focus on this chapter is on developing methods to fit a particular class of models, the results of this research can be applied to many other statistical models that present similar computational challenges.

3.1 Challenges using HMC to Fit Complex Statistical Models

For simpler models, the derivation of the gradient of the log posterior as well as manually setting tuning parameters is a feasible undertaking. However, for large and complex models, the derivation and tuning exercises present a substantial barrier to using HMC for practical applications. Automated tools to handle gradient derivation and computation as well as parameter tuning are needed for HMC to facilitate the adoption of this technique by applied statisticians.

It is important to note that exact gradient computation is critical to implement HMC in practice. The error introduced by approximation methods such as finite differencing accumulate through the multiple steps of the leapfrog algorithm. As a result, the discrete approximation error to the Hamiltonian equations increases to where the acceptance ratio drops to nearly zero in my computational experiments.

The most efficient gradient formulation possible for computation is a function coded to calculate the gradient directly, as presented in the previous chapter. When possible, programming such a function directly is the optimal approach. If this is not feasible, an exact automated gradient computation approach is required. Available software tools for automated gradient computation include Autodiff (Carpenter et al. 2015) and the Tensorflow API (Abadi et al. 2016).

Model tuning also becomes prohibitively challenging as the number of parameters grows. Some of these challenges can be mitigated using techniques presented in this chapter. However, the sheer volume of hundreds or even thousands of parameters can be daunting for even the most experienced Bayesian statisticians. A number of automated tuning methods are available for practical use today, including the popular No U-Turn Sampler (Hoffman and Gelman 2014) for automatic selection of ϵ and L.

In addition to the complexities of gradient computation and model tuning, the HMC algorithm itself has mathematical characteristics that present challenges to implementation. HMC proposals are based on a combination of the step-size ϵ, the number of leapfrog steps L, the parameterization of the latent variable $\mathbf{p}$, and the gradient of the log target density. One way to consider Hamiltonian proposals is via the concept of distance traveled in the parameter space of $\boldsymbol{\theta}$. In Euclidean space, the distance is calculated via the L2-norm, such that $D(\boldsymbol{\theta}, \boldsymbol{\theta} + \gamma\boldsymbol{\theta}) = ||\gamma\boldsymbol{\theta}||$.

If I consider a total distance of γ in (2.27), Euclidean Hamiltonian Monte Carlo assumes that the distance of in each direction of the parameter space is approximately constant over a small distance. However, the gradients may change rapidly over even short distances (Calderhead 2011). Since Euclidean HMC only includes first order information, parameterization adjustments may be essential when fitting complex statistical models. Correlated parameters are particularly challenging since the linear mapping of HMC proposals via $\mathbf{M}^{-1}\mathbf{p}$ may not efficiently explore the log posterior (Girolami and Calderhead 2011).

The development of a parameterization strategy to take advantage of the benefits of HMC, while minimizing the algorithm's limitations is required to maximize computational efficiency. Such computational challenges can be illustrated with the complexities of fitting covariance matrices using HMC. In particular, I focus on covariance matrix model fitting for Generalized Additive Models (GAM) with multiple response as an example of a complex non-standard statistical model.

3.2 A New Approach to Fitting Covariance Matrices Using HMC

To fit multiple response GAM's using HMC, a covariance matrix of the multiple outcomes must be simulated to estimate the subject-level correlation. Current parameterization for covariance matrix estimation with HMC uses the single parameter LKJ (Lewandowski, Kurowicka, and Joe 2009) prior. Although this approach is effective for many general applications, this single-parameter approach restricts the analyst's ability to incorporate prior covariance information in the estimation process. In particular, the prior specification of partial covariances between outcome pairs cannot be directly incorporated with an LKJ prior. A preferred parameterization would allow analysts to set priors for partial covariances directly. To that end, a novel parameterization for simulating covariance matrices in HMC is developed to provide that flexibility.

In the mixed effects model framework, the random effects are modeled as a multivariate Normal with covariance $\mathbf{G}$, where $\mathbf{u} \sim N(0, \mathbf{G})$. One of the most challenging aspects of using MCMC to estimate mixed effects models is parameterizing the covariance of the random effects $\mathbf{G}$. There are a number of options, with varying opinions.

The traditional approach from Gibbs sampling is to use an inverse Wishart prior on $\mathbf{G}$. With this prior, the conditional posterior distribution of $\mathbf{G}$ is also inverse Wishart. One disadvantage of the inverse Wishart is that the flexibility in parameterization is limited (Gelman et al. 2013). Another disadvantage is that the marginal distribution of the variance is an inverse gamma distribution. The inverse gamma can be especially problematic for $\mathbf{G}$ when the variance is close zero. Simply put, the distribution of the inverse gamma is not well-behaved at extremely low values (Gelman 2006). The density near zero is extremely low, which biases estimates to larger variance even if the true variance is low.

One alternative approach is the scaled inverse Wishart distribution. This approach incorporates additional parameters for the variance via a diagonal matrix. The most commonly used approach in today's hierarchical models divides the covariance matrix into diagonal matrices of standard deviations and a correlation matrix (Barnard, McCulloch, and Meng 2000).

Barnard, McCulloch, and Meng (2000) call this approach a separation strategy. Beyond the intuitive appeal of separating the standard deviations and correlation, MCMC chains can more easily navigate the separated parameters. The priors for the standard deviations can be any distribution with strictly positive values, such as a lognormal. The prior for the correlation matrix is typically chosen to be the LKJ prior (Lewandowski, Kurowicka, and Joe 2009). The LKJ distribution is defined for a correlation matrix with a single scaling parameter that is used to provide information on the strength of the individual partial correlations.

This separation strategy approach works well for many modeling applications. In particular, the single parameter prioritization is appealing in fitting high-dimensional correlation matrices. The individual parameterization of each partial correlation can be prohibitively expensive from a computational efficiency perspective. However, this single-parameter approach has a disadvantage in constraining the off-diagonal covariance parameters. For statistical models where the correlations are parameters of interest, a more flexible parameterization is needed.

Optimally, each off-diagonal parameter of a given covariance matrix would receive its own prior when fitting such a model with HMC. A particular complexity in designing an approach for HMC is that each parameter must be unconstrained. Further, each simulated covariance matrix must also be positive semi-definite.

I propose a modeling strategy based on the modified Cholesky decomposition to simultaneously allow unconstrained parameterization while ensuring positive definite results. In this approach, $\mathbf{L}$ is the lower triangular matrix with diagonal elements of 1 and $\mathbf{D}$ is a diagonal matrix of the variances and $\mathbf{D}^{1/2}$ a diagonal matrix of the standard deviations.

$$\begin{aligned} \mathbf{G} &= \mathbf{L}\mathbf{D}\mathbf{L}^T \\ &= \mathbf{L}\mathbf{D}^{1/2}\mathbf{D}^{1/2}\mathbf{L}^T \end{aligned}$$

An appealing aspect of this decomposition is that the off-diagonal parameters in $\mathbf{L}$ can be observed as autoregressive parameters on the random effects (Chan and Jeliazkov 2009). Provided the diagonal elements of $\mathbf{D}$ are strictly positive, the off-diagonal elements of $\mathbf{L}$ may be unconstrained.

This approach provides the analyst with the flexibility to incorporate prior information for individual partial correlations. A practical application for this flexibility is multiple response modeling. In biostatistical applications, multiple outcomes are often measured simultaneously. Prior information from previous biostatistical studies may be incorporated into such modeling using this structure. This direct specification of priors for particular correlations is not possible with the commonly used LKJ prior. In contrast, the modified Cholesky approach provides researchers with maximal flexibility for such applications.

The inverse gamma prior was first explored in researching variance parameters for the diagonal elements of $\mathbf{D}$, with a log transformation to allow the parameter to be unconstrained over all real numbers. This approach created significant problems in the MCMC simulations of mixed effects models due to the previously articulated issues with inverse gamma's behavior close to zero.

An alternative parameterization was explored based on the half-t family of distributions (Gelman 2006). These distributions behave well at zero, and provide the flexibility for a variety of parameterizations, including the half-cauchy and improper uniform distribution. The half-t priors applied to $\mathbf{D}$ substantially the computational efficiency of model fitting with HMC.

One of the difficulties identified by M. J. Betancourt and Girolami (2013) is the curvature of the distribution of the random effects with the covariance $\mathbf{G}$. At low variance values, the movement of the random effects in the chain can be severely restricted. These difficulties were observed in fitting linear mixed models and generalized linear mixed models with a logit link.

A remedy for the funnel distribution problem is to re-define $\mathbf{u}$ based on the product of a standard normal vector $\boldsymbol{\tau}$ and the modified Cholesky decomposition of $\mathbf{G}$. The HMC chain samples on $\boldsymbol{\tau}$, $\mathbf{D}^{1/2}$, and L. The random effects $\mathbf{u}$ are a deterministic function of the sampled parameters. This approach assures that the variance of $\boldsymbol{\tau}$ is always constant, while the covariance is sampled separately.

The distribution of the random effects are defined as normal with a mean of zero. Note that in this parameterization, I directly decompose $\mathbf{G}$ instead of $\mathbf{G}^{-1}$ as in Chan and Jeliazkov (2009),

$$\begin{aligned}\mathbf{u} &\sim N(0, \mathbf{G}) \\ \mathbf{G} &= \mathbf{L}\mathbf{D}\mathbf{L}^T \\ &= \mathbf{L}\mathbf{D}^{1/2}\mathbf{D}^{1/2}\mathbf{L}^T.\end{aligned}$$

Let λ_k where $k = 1, ...p$ denote the diagonal entries of $\mathbf{D}^{1/2}$ and let a_{kj} where $1 \leq j < k \leq p$ denote free elements of lower unitrangular matrix $\mathbf{L}$,

$$\mathbf{D}^{1/2} := \begin{pmatrix} \lambda_1 & 0 & ... & 0 \\ 0 & \lambda_2 & 0... & 0 \\ ... & ... & ... & ... \\ 0 & 0 & ... & \lambda_p \end{pmatrix} \quad \mathbf{L} := \begin{pmatrix} 1 & 0 & 0 & ... & 0 \\ a_{21} & 1 & 0 & ... & 0 \\ a_{31} & a_{32} & 1 & ... & ... \\ ... & ... & ... & ... & ... \\ a_{p1} & a_{p2} & ... & ... & 1 \end{pmatrix}.$$

Also define $\lambda := (\lambda_1, ..., \lambda_p)^T$ and $\mathbf{a}_k := (a_{k1}, ..., a_{k,k-1})^T$ and $\mathbf{a} := (\mathbf{a}_2^T, ..., \mathbf{a}_p^T)^T$.

I use a half-t prior for standard deviation λ_k and Normal priors for $\mathbf{a}$,

$$p(\lambda_k) \sim \left(1 + \frac{1}{\nu}\left(\frac{\lambda_k}{A}\right)^2\right)^{-(\nu+1)/2}$$

$$\mathbf{a}|\lambda \sim N(a_0, A_0).$$

The hyperparameter a_0 does not need to be zero, and A_0 can be correlated and may depend on $\boldsymbol{\lambda}$. In this model, I define $\mathbf{a}$ independent of $\boldsymbol{\lambda}$.

I re-parameterize u using a standard normal parameterization which I define as $\boldsymbol{\tau} = (\tau_1, ..., \tau_q)$. Here, $\mathbf{u}$ is a deterministic function of $\mathbf{G}$ and $\boldsymbol{\tau}$,

$$\begin{aligned} &\boldsymbol{\tau} \sim N(0, \mathbf{I}_q), \\ &\mathbf{u} := \mathbf{L}\mathbf{D}^{1/2}\boldsymbol{\tau} \\ &\quad \sim N(0, \mathbf{L}\mathbf{D}^{1/2}\mathbf{I}(\mathbf{L}\mathbf{D}^{1/2})^T) \\ &\quad \sim N(0, \mathbf{L}\mathbf{D}^{1/2}\mathbf{D}^{1/2}\mathbf{L}^T) \\ &\quad \sim N(0, \mathbf{G}). \end{aligned}$$

The distribution of $\mathbf{u}$ therefore does not change with this parameterization. This re-parameterization allows $\boldsymbol{\lambda}$ and $\boldsymbol{\tau}$ to be largely independent in HMC sampling.

3.3 Data Driven Prior Specification for Efficient HMC Estimation

The kinetic energy component in HMC is based on a mass matrix M, which effectively rotates and scales the target distribution (M. J. Betancourt and Girolami 2013),

$$\mathbf{K}(\boldsymbol{\theta}, \mathbf{p}) = \frac{1}{2}\mathbf{p}^T\mathbf{M}^{-1}\mathbf{p}.$$

The standard HMC algorithm defines $\mathbf{M}$ as a unit diagonal matrix. The HMC chain will converge (Gelman et al. 2013), but may be made more efficient by choosing a different $\mathbf{M}$ based on the data. The mass matrix is ideally based on the covariance of the parameters, which is unknown. An approach to tuning with this uncertainty is to transform the parameters via Cholesky decomposition (R. Neal 2011) or QR decomposition. Both approaches standardize the design matrix such that the mass matrix is then close to identity. I prefer QR decomposition as a standard design matrix transformation to minimize the computational impact of correlated parameters in HMC sampling.

Let $\boldsymbol{\theta} = \mathbf{R}^{*}\boldsymbol{\beta}$ for a model with n samples. The HMC estimates $\boldsymbol{\theta}$, from which I may use the deterministic formula to determine $\boldsymbol{\beta}$,

$$\begin{aligned}
\mathbf{X} &= \mathbf{Q}^{*}\mathbf{R}^{*} \\
\mathbf{Q}^{*} &= \mathbf{Q} \cdot \sqrt{n-1} \\
\mathbf{R}^{*} &= \frac{1}{\sqrt{n-1}}\mathbf{R} \\
\mathbf{X}\boldsymbol{\beta} &= \mathbf{Q}^{*}\mathbf{R}^{*}\boldsymbol{\beta} \\
\boldsymbol{\beta} &= \mathbf{R}^{*^{-1}}\boldsymbol{\theta}.
\end{aligned}$$

By assigning a prior on the transformed parameters $\boldsymbol{\beta}$, I effectively employ empirical Bayes to form the priors (M. Betancourt 2017). For a linear mixed effect model formulation with dependent variable $\mathbf{y}$,

$$\begin{aligned}
\mathbf{y} &= \mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\mathbf{u} + \boldsymbol{\epsilon} \\
&= \mathbf{Q}_{\beta}\mathbf{R}_{\beta}\boldsymbol{\beta} + \mathbf{Q}_{u}\mathbf{R}_{u}\mathbf{u} + \boldsymbol{\epsilon} \\
&= \mathbf{Q}_{\beta}\widetilde{\boldsymbol{\beta}} + \mathbf{Q}_{u}\widetilde{\mathbf{u}} + \boldsymbol{\epsilon},
\end{aligned}$$

where $\boldsymbol{\beta}$ and $\mathbf{u}$ are fixed and random effect parameters, $\mathbf{X}$ and $\mathbf{Z}$ are the related design matrices, and $\boldsymbol{\epsilon}$ is the error term. From QR decomposition, the transformed parameters are noted as $\widetilde{\boldsymbol{\beta}} = \mathbf{R}_{\beta}\boldsymbol{\beta}$ and $\widetilde{\mathbf{u}} = \mathbf{R}_{u}\mathbf{u}$.

Next, I define the parameter for the random effects parameter $\mathbf{u}$ via standard normal multiplied by another scale parameter, where

$$\begin{aligned}
\boldsymbol{\tau} &\sim N(0, \mathbf{I}) \\
\mathbf{u} &:= \boldsymbol{\tau}\lambda \\
&:= \boldsymbol{\tau}e^{\xi} \\
\mathbf{u} &\sim N(0, \lambda^{2}I) \\
&\sim N(0, e^{2\xi}\mathbf{I}).
\end{aligned}$$

The model equation then is adjusted to

$$\begin{aligned}\mathbf{y} &= \mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\mathbf{u} + \boldsymbol{\epsilon} \\ &= \mathbf{X}\boldsymbol{\beta} + e^{2\xi}\mathbf{Z}\boldsymbol{\tau} + \boldsymbol{\epsilon} \\ &= \mathbf{Q}_\beta \mathbf{R}_\beta \boldsymbol{\beta} + e^{2\xi}\mathbf{Q}_u \mathbf{R}_u \boldsymbol{\tau} + \boldsymbol{\epsilon}.\end{aligned}$$

In this methodology, I assign priors to the transformed parameters

$$\begin{aligned}&\widetilde{\boldsymbol{\tau}} \sim N(0, \mathbf{I}) \\ &\boldsymbol{\tau} = \mathbf{R}_u^{*-1}\widetilde{\boldsymbol{\tau}} \sim N(0, \mathbf{R}_u^{*-1}\mathbf{R}_u^{*-T}) \\ &\widetilde{\mathbf{u}} = e^{\boldsymbol{\xi}}\widetilde{\boldsymbol{\tau}} \sim N(0, e^{2\boldsymbol{\xi}}I) \\ &\mathbf{u} = e^{\boldsymbol{\xi}}\boldsymbol{\tau} \\ &\quad = e^{\boldsymbol{\xi}}\mathbf{R}_u^{*-1}\widetilde{\boldsymbol{\tau}} \\ &\mathbf{u} \sim N(0, \mathbf{R}_u^{*-1}(e^{2\boldsymbol{\xi}}\mathbf{I})\mathbf{R}_u^{*-T}).\end{aligned}$$

This differs from the standard prior $\mathbf{u} \sim N(0, e^{2\xi}\mathbf{I})$. Since this parameterization includes information from the design matrix $\mathbf{Z}$, I note that this approach is a form of empirical Bayes. I demonstrate that the prior for $\mathbf{u}$ incorporates all of the correlation information from the data,

$$\begin{aligned}\mathbf{Z} &= \mathbf{Q}\mathbf{R} \\ &= \mathbf{Q}\sqrt{n-1} \cdot \mathbf{R}/\sqrt{n-1} \\ &= \mathbf{Q}^*\mathbf{R}^* \\ \mathbf{Z}^T\mathbf{Z} &= (\mathbf{Q}^*\mathbf{R}^*)^T\mathbf{Q}^*\mathbf{R}^* \\ &= (\mathbf{R}^*)^T(\mathbf{Q}^*)^T\mathbf{Q}^*\mathbf{R}^* \\ &= (\mathbf{R}^*)^T(\mathbf{Q}\sqrt{n-1})^T(\mathbf{Q}\sqrt{n-1})\mathbf{R}^* \\ &= (n-1)(\mathbf{R}^*)^T\mathbf{Q}^T\mathbf{Q}\mathbf{R}^*.\end{aligned}$$

Since **Q** is a matrix of orthonormal columns, $Q^TQ = I$,

$$\begin{aligned}
\mathbf{Z}^T\mathbf{Z} &= (n-1)(\mathbf{R}^*)^T I\mathbf{R}^* \\
\mathbf{Z}^T\mathbf{Z} &= (n-1)(\mathbf{R}^*)^T\mathbf{R}^* \\
(\mathbf{Z}^T\mathbf{Z})^{-1} &= \frac{1}{n-1}\left((\mathbf{R}^*)^T\mathbf{R}^*\right)^{-1} \\
&= \frac{1}{n-1}(\mathbf{R}^*)^{-1}\mathbf{R}^{*-1^T} \\
(\mathbf{R}^*)^{-1}\mathbf{R}^{*-1^T} &= (n-1)(\mathbf{Z}^T\mathbf{Z})^{-1}.
\end{aligned}$$

Finally, it is apparent that the prior covariance for $\mathbf{u}$ can be written in terms of the design matrix $(\mathbf{Z}^T\mathbf{Z})^{-1}$

$$\begin{aligned}
\mathbf{u} &\sim N(0, \mathbf{R}_u^{*-1}(e^{2\xi}\mathbf{I})\mathbf{R}_u^{*-T}) \\
&\sim N(0, e^{2\xi}\mathbf{R}_u^{*-1}\mathbf{I}\mathbf{R}_u^{*-T}) \\
&\sim N(0, e^{2\xi}\mathbf{R}_u^{*-1}\mathbf{R}_u^{*-T}) \\
&\sim N(0, (n-1)e^{2\xi}(\mathbf{Z}^T\mathbf{Z})^{-1}).
\end{aligned}$$

Note that I also utilize the QR decomposition in the estimation of the fixed effect parameters $\boldsymbol{\beta}$. This prior, also multivariate Normal, is assigned to $\widetilde{\boldsymbol{\beta}}$, but with a fixed hyperprior on the variance,

$$\begin{aligned}
\widetilde{\boldsymbol{\beta}} &= \mathbf{R}_\beta^*\boldsymbol{\beta} \\
\widetilde{\boldsymbol{\beta}} &\sim N(0, \sigma_\beta^2 I) \\
\boldsymbol{\beta} &= \mathbf{R}_\beta^{*-1}\widetilde{\boldsymbol{\beta}} \\
\boldsymbol{\beta} &\sim N(0, \sigma_\beta^2\mathbf{R}_\beta^{*-1}\mathbf{R}_\beta^{*-1^T}) \\
&\sim N(0, (n-1)\sigma_\beta^2(\mathbf{X}^T\mathbf{X})^{-1}).
\end{aligned}$$

The hyperprior σ_β^2 is typically set large for a relatively uninformative prior when $\boldsymbol{\beta}$ has no prior information. Here also, the use of QR decomposition employs an empirical Bayes approach to model fitting with HMC.

While the parameterization approaches presented in this section are optimal in terms of computational efficiency, they do present statisticians with potentially difficult choices in assigning priors. For applications where vague priors are used for estimation, the empirical Bayesian nature of the priors presents little difficulty, as the information of the data on such priors is minimal.

In cases where an informed prior is desired, analysts have several options. One option is to fit the models in HMC directly, with the raw design matrices untransformed. For this option, a small step size may be necessary to accommodate the curved nature of the parameter space. The HMC chain will be ergodic and converge to the true posterior, but at a slower rate due to the small step sizes. In addition, more samples may be needed if the resulting MCMC chain has a high autocorrelation. The analyst may also elect to tune the mass matrix M to the expected covariance of the parameter of interest $\boldsymbol{\theta}$, although this can be difficult in practice. Finally, analysts may account for prior information while using the empirical Bayes priors from the design matrices transformations. For QR decomposition, the analyst would need to derive the transformed priors as assess the additional information provided by the data, as shown above.

3.4 Fitting Multivariate Response Generalized Additive Models Using HMC

The foundation of this modeling approach is based on *semiparametric regression* (Ruppert, Wand, and Carroll 2003), which incorporates both parametric and nonparametric components in a statistical model. Nonparametric smoothing may be handled by a variety of basis functions, including truncated polynomials and thin plate splines (TPS). The multiple outcomes are connected by a subject-specific random effect based on a multivariate normal distribution (H. Liu, Tu, and others 2012). Therefore, this complex modeling framework incorporates multidimensional responses, nonparametric effects, and the interaction of the nonlinear associations of bivariate independent variables to multiple outcomes.

H. Liu, Tu, and others (2012) apply this approach to modeling the association of the bivariate response of systolic and diastolic blood pressure and the joint nonlinear effects of height and weight. The multivariate response modeling framework is preferred for this application due to the high correlation of systolic and diastolic blood pressure as well as the significant interaction of the effects of height and weight. While weight was the major influence on both systolic and diastolic blood pressure, height showed a more significant association with diastolic blood pressure in comparison with a lesser effect of height on systolic blood pressure. This study demonstrates the capabilities of this modeling approach to differentiating the effects of weight and height on multivariate outcomes. In addition, these associations vary by sex and race, indicating different pathophysiologies by these factors.

This multivariate additive framework requires a computational approach capable of fitting the high-dimensional data produced by the univariate and bivariate smoothing functions, as well as the covariance structure of multiple responses. To the best of my knowledge, no current software packages are designed to handle this type of model directly. However, some model-fitting software can be used if the data is specially organized in a specific required format. For example, frequentist approaches for fitting these models rely on specifying fixed and random effect parameters in a mixed effect model framework (Ruppert, Wand, and Carroll 2003)(S. Wood 2017). This approach involves maximizing the log likelihood or a restricted log likelihood. The restricted maximized log likelihood (REML) approach accounts for the degrees of freedom in the fixed effects to reduce the bias in the covariance parameter selections. In either case, frequentist estimation requires the maximization of the log likelihood over a high dimensional space due to the substantial number of smoothing parameters. Approaches to fitting the maximum likelihood such as quadrature methods tend to scale poorly with high dimensions, limiting the practical potential to use existing software to fit multivariate generalized additive models.

3.4.1 Generalized Additive Model Formulation

A general formulation of the model is based on y_{ijk} for multiple responses $r = 1, \ldots, R$ from $i = 1, \ldots, m$ subjects at $j = 1, \ldots n_i$ time points for each subject. As an extension of Generalized Linear Model's (GLM), the model is from the exponential family of distributions,

$$f(y_{ijr}|\eta_{ijr}, \phi_r) = \exp\left(\frac{y_{ijr}\eta_{ijr} - b(\eta_{ijr})}{a(\phi)_r} + c(y_{ijr}; \phi_r)\right), \tag{3.1}$$

where η_{ijr} is the natural parameter of the distribution and ϕ_r is the nuisance or dispersion parameter (Agresti 2015). The mean of the response is parameterized as $\mu_{ijr} = E(y_{ijr})$, where μ_{ijr} is a function of the natural parameter. A monotonic, differentiable link function $g(\cdot)$ is applied to the natural parameter such that $\eta_i = g(\mu_i)$. Choosing sensible link functions depends on the distribution of the response. For example, the identity function $g(\mu_i) = \mu_i$ is used for normal data, and the logit function is often used for binomial data $g(\mu_i) = \log \frac{\mu_i}{1-\mu_i}$.

A multivariate response generalized additive model is specified,

$$\eta_{ijk} = g(\mu_{ijr}) = \mathbf{x}_{ij}^T \boldsymbol{\beta}_r + \mathbf{z}_{ij}^T \mathbf{b}_r + s_r(t_{1ij}, t_{2ij}), \tag{3.2}$$

where $\boldsymbol{\beta}_r = (\beta_r^{(1)}, \ldots, \beta_r^{(p)})^T$ is a vector of fixed effect parameters for covariates $\mathbf{x}_{ij}$, $\mathbf{b}_r = (b_r^{(1)}, \ldots, b_r^{(q)})^T$ is a vector of random effects parameters for subject-level covariates $\mathbf{z}_{ij}$ and $s_r(\cdot)$ is a nonparametric smooth function for independent variables t_{1ij} and t_{2ij}.

The smooth function can be expressed generally as $s(t_{1ij}, t_{2ij}) = \sum_{k=1}^{K} \gamma_{rk} h_k(t_{1ij}, t_{2ij})$ with coefficients γ_{rl} (Li, Liu, and Tu 2017). Further, the smoothing function can be written in the form $s(t_{1ij}, t_{2ij}) = \mathbf{T}_{ij}^T \boldsymbol{\gamma}_{ij}$, where $\boldsymbol{\gamma}_r = (\gamma_{r1}^T, \cdots, \gamma_{rK}^T)$ and

$\mathbf{T}_{ij} = [h_1(t_{1ij}, t_{2ij}), \ldots, h_K(t_{1ij}, t_{2ij})]^T$. The result of the smooth function $s_r(t_{1ij}, t_{2ij})$ where $j = 1, \ldots, n_i$ and $i = 1, \ldots, m$ can be expressed as a vector $\mathbf{s}_r$. This can be written in matrix form,

$$\mathbf{s}_r = \mathbf{T}_r \boldsymbol{\gamma}_r, \tag{3.3}$$

where $\mathbf{s}_r$ is the vector of coefficients. The basis function matrix $\mathbf{T}_r$ includes the row vectors $\mathbf{T}_{ij}^T \ \forall \ i, j$.

The response is expressed in matrix form, where

$$\begin{aligned} \mathbf{y}_r^T &= (y_{11r}, \ldots, y_{1n_ir}, \ldots, y_{m1r}, \ldots, y_{mn_ir})^T, \\ \mathbf{Y} &= (\mathbf{y}_1^T, \ldots, \mathbf{y}_R^T)^T, \end{aligned} \tag{3.4}$$

for $r = 1, \ldots, R$. The fixed effects design matrix for subject i is defined $\mathbf{X}_i^{(\beta)} = (\mathbf{x}_{i1}^T, \ldots, \mathbf{x}_{in_i}^T)^T$, and the random effects design matrix for subject i is $\mathbf{Z}_i^{(b)} = (\mathbf{z}_{i1}^T, \ldots, \mathbf{z}_{in_i}^T)^T$. For all subjects, $\mathbf{X}^{(\beta)} = (\mathbf{X}_1^{(\beta)^T}, \ldots, \mathbf{X}_m^{(\beta)^T})^T$ with fixed effect parameters $\widetilde{\boldsymbol{\beta}} = (\boldsymbol{\beta}_1^T, \ldots, \boldsymbol{\beta}_R^T)^T$ and $\mathbf{Z}^{(b)} = (\mathbf{Z}_1^{(b)^T}, \ldots, \mathbf{Z}_m^{(b)^T})^T$ with random effect parameters $\widetilde{\mathbf{b}} = (\mathbf{b}_1^T, \ldots, \mathbf{b}_R^T)^T$. Similarly, I define $\mathbf{s} = (\mathbf{s}_1^T, \ldots, \mathbf{s}_R^T)^T$, $\boldsymbol{\gamma} = (\boldsymbol{\gamma}_1^T, \ldots, \boldsymbol{\gamma}_R^T)^T$, and $\mathbf{T} = (\mathbf{T}_1^T, \ldots, \mathbf{T}_R^T)^T$. Equation (3.2) can then be written for the full dataset,

$$\boldsymbol{\eta} = \mathbf{X}^{(\beta)} \widetilde{\boldsymbol{\beta}} + \mathbf{Z}^{(b)} \widetilde{\mathbf{b}} + \mathbf{T}\boldsymbol{\gamma}. \tag{3.5}$$

The random effect parameters follow a multivariate normal distribution, $\widetilde{\mathbf{b}} \sim N(\mathbf{0}, \boldsymbol{\Sigma}_b \otimes \mathbf{I}_m)$.

3.4.2 Bivariate smoothing parameterization

Thin plate splines are a commonly used option to model s_r in (3.2). I let $\boldsymbol{\psi} = (\widetilde{\boldsymbol{\beta}}^T, \widetilde{\boldsymbol{\lambda}}^T, \boldsymbol{\gamma}^T)^T$ be a vector of parameters, where $\widetilde{\boldsymbol{\lambda}}$ represent the variance components of $\boldsymbol{\Sigma}_b$. One approach to fitting this model would consist of maximizing a penalized log-likelihood function (Hastie and Tibshirani 1990),

$$p\ell(\boldsymbol{\psi}) = \ell(\boldsymbol{\psi}) - \sum_{r=1}^{R} \lambda_r J(s_r), \tag{3.6}$$

where $\ell(\boldsymbol{\psi})$ is the log-likelihood function. The function $J(s_r)$ applies a penalty to s_r with smoothing parameter λ_r. A common choice for the penalty function is $J(s_r) = \int\int_{\mathbb{R}^2} s_r''(t_1, t_2)^2 dt_1 dt_2$, which can be written in quadratic form $J(s_r) = \boldsymbol{\gamma}_r^T \mathbf{S}_r \boldsymbol{\gamma}$ (S. N. Wood 2003). Here, $\mathbf{S}_r$ is a positive-definite matrix of known coefficients which can be divided into penalized and unpenalized components for the smoothing function s_r. The coefficient vector γ_r can be split into fixed and random effect coefficients based on the eigen decomposition of the $\mathbf{S}_r$ (Hastie and Tibshirani 1990).

The vector γ_r can be divided into fixed and random effects coefficients, such that $\boldsymbol{\gamma}_r^T \mathbf{S}_r \boldsymbol{\gamma}_r = \boldsymbol{\gamma}_{r,E}^T \mathbf{S}_{r,E} \boldsymbol{\gamma}_{r,E}$, where $\mathbf{S}_{r,E}$ is diagonal with positive eigenvalues of $\mathbf{S}_r$, $\boldsymbol{\gamma}_{r,E}$ are the random effect coefficients, and $\boldsymbol{\gamma}_{r,F}$ are the unpenalized fixed effects coefficients (Li, Liu, and Tu 2017). A mixed effect model formulation of (3.3) can be specified,

$$\mathbf{s}_r = \mathbf{T}_{r,F}\boldsymbol{\gamma}_{r,F} + \mathbf{T}_{r,E}\boldsymbol{\gamma}_{r,E}, \tag{3.7}$$

with fixed effects design matrix $\mathbf{T}_{r,F}$ and random effects design matrix $\mathbf{T}_{r,E}$ and $\boldsymbol{\gamma}_{r,E} \sim N\left(0, \frac{\lambda_r}{\mathbf{S}_{r,E}}\right)$. The mixed model form of the full model is expressed by substituting (3.7) into (3.5),

$$\boldsymbol{\eta} = \mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\mathbf{u}, \tag{3.8}$$

where $\mathbf{X} = (\mathbf{X}^{(\beta)}, \text{diag}(\mathbf{T}_{1,F}, \ldots, \mathbf{T}_{R,F}))$ and $\mathbf{Z} = (\mathbf{Z}^{(b)}, \text{diag}(\mathbf{T}_{1,E}, \ldots, \mathbf{T}_{R,E}))$ are the fixed and random effects matrices. The fixed effects parameters are combined, $\boldsymbol{\beta} = (\widetilde{\boldsymbol{\beta}}^T, \boldsymbol{\gamma}_{1,F}^T, \ldots, \boldsymbol{\gamma}_{R,F}^T)^T$, and the random effects parameters are combined, $\mathbf{u} = (\widetilde{\mathbf{b}}^T, \boldsymbol{\gamma}_{1,E}^T, \ldots, \boldsymbol{\gamma}_{R,E}^T)^T$. The distribution of the random effects parameters is multivariate normal, $\mathbf{u} \sim N(\mathbf{0}, \boldsymbol{\Sigma}_u)$ where $\boldsymbol{\Sigma}_u = \text{diag}(\boldsymbol{\Sigma}_b \otimes \mathbf{I}_m, \mathbf{S}_{1,E}^{-1}/\lambda_1, \ldots, \mathbf{S}_{R,E}^{-1}/\lambda_R)$. The variance components of $\boldsymbol{\Sigma}_u$ are $\boldsymbol{\lambda}_u = (\widetilde{\boldsymbol{\lambda}}^T, \lambda_1, \ldots, \lambda_R)^T$.

3.4.3 Extending Generalized Additive Models to Multiple Responses

I summarize some of the numerous complexities required for multivariate generalized additive models:

- Multiple outcomes,
- Correlation of multiple outcomes on individual subjects,
- Nonlinear effects of independent measures, and
- Interaction of two nonlinear effects on the outcomes of interest.

The combination of these factors necessitates a sophisticated modeling approach beyond what is commonly used by analysts today. The framework that is proposed for such applications is a multivariate additive model that was originally proposed by H. Liu, Tu, and others (2012) and generalized by Li, Liu, and Tu (2017).

3.4.4 Connecting Random Intercepts from Multiple Responses

Subject-level random intercepts may be specified as one of the random effects parameters. The random intercepts of the multiple responses are assigned to $\widetilde{\mathbf{b}}$, the first parameters in $\mathbf{u}$. The variance-covariance matrix $\boldsymbol{\Sigma}_b$ captures the correlation between repeated measurements of the same subject, as well as the correlation between multiple response variables (H. Liu, Tu, and others 2012).

I apply modified Cholesky decomposition similar to the recommendation by Chan and Jeliazkov (2009) for simulated estimation of $\boldsymbol{\Sigma}_b$. The modified Cholesky decomposition is applied to the covariance matrix $\boldsymbol{\Sigma}_b = \mathbf{L}\mathbf{D}\mathbf{L}^T$ where $\mathbf{L}$ is a $R \times R$ lower triangular matrix, and $\mathbf{D}$ is a $R \times R$ diagonal matrix containing the variance parameters of $\boldsymbol{\Sigma}_b$.

The lower triangular matrix **L** contains 1's on the diagonal and off-diagonal elements $a_{gh}, \quad 1 \leq h \leq g \leq R-1$. For $R > 2$,

$$\mathbf{L} = \begin{pmatrix} 1 & 0 & \dots & 0 \\ a_{21} & 1 & 0 & 0 \\ \dots & \dots & \dots & \dots \\ a_{R1} & a_{R2} & \dots & 1 \end{pmatrix} \quad \text{and} \quad \mathbf{D} = \begin{pmatrix} \widetilde{\lambda}_1 & 0 & \dots & 0 \\ 0 & \widetilde{\lambda}_2 & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & \widetilde{\lambda}_R \end{pmatrix}.$$

Further, I define the off-diagonal parameters of **L** as $\mathbf{a}_g = (a_{g1}, \dots, a_{g,g-1})^T$ and $\mathbf{a} = (\mathbf{a}_2^T, \dots, \mathbf{a}_{R-1}^T)^T$. The modified Cholesky decomposition with strictly positive restrictions on **D** ensure that simulated covariance matrices $\mathbf{\Sigma}_b$ will be positive definite with **a** unconstrained. (Newton 1988)(Pourahmadi 1999).

3.5 Modeling Framework for Semiparametric Regression

I begin the development of the modeling framework with a single dependent variable. The formulation of this model is similar to mixed effect models familiar to statisticians,

$$g[E(\mathbf{y}|\mathbf{X}, \mathbf{Z}, \boldsymbol{\beta}, \mathbf{u})] = \mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\mathbf{u}. \tag{3.9}$$

I define $g(\cdot)$ as a link function to the linear predictors as expressed in (3.9). The dependent variable $\mathbf{y} = y_1, \dots, y_n$ has n observations. The fixed effects design matrix is $\mathbf{X} = (\mathbf{x}_1^T, \dots, \mathbf{x}_p^T)^T$ for p fixed effect parameters where $\boldsymbol{\beta} = (\beta_1, ..., \beta_p)^T$. As applied to GAMs, **X** and $\boldsymbol{\beta}$ are the expressions for the parametric portion of the model. The default in **bayesGAM** is to assign β_1 as a global intercept, although this can be overridden by the user. The random effects design matrix **Z** captures the smoothing functions for the nonparametric portion of the model, while **u** are the corresponding nonparametric parameters.

Further, the nonparametric parameters and error parameters are assumed to be normally distributed, such that $\mathbf{u} \sim N(0, \mathbf{G})$. The covariance of the random effects **G** is specially structured to facilitate efficient sampling in HMC (M. J. Betancourt and Girolami

2013). I define $\mathbf{G} := \boldsymbol{\lambda}^T \mathbf{I}_q \boldsymbol{\tau}$, where $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_q)^T$ are strictly positive parameters and $\boldsymbol{\tau} = (\tau_1, \ldots \tau_q)^T$. The distribution of $\boldsymbol{\tau} \sim N(0, \mathbf{I}_q)$ is standard normal with no hyperparameters.

Semiparametric models express nonlinear relationships using a set of basis functions. These basis functions include *knots* at selected points in the span of a given independent variable. The individual parameters for the knots $\mathbf{u}$ provide the weights for the nonlinear function. The random effect design matrix $\mathbf{Z} = (\mathbf{Z}_1, ..., \mathbf{Z}_q)$ combines the submatrices for the $j = 1, \ldots, q$ variables modeled by nonlinear smoothing functions. The dimensions of each submatrix is dependent on the number of knots K_j for each variable. The individual knot locations for each set of nonlinear parameters j are defined as $\kappa_1^{(j)}, \ldots, \kappa_{K_j}^{(j)}$ such that $\mathbf{Z}_j \in \mathbb{R}^{n \times K_j}$.

For example, a linear spline basis function can be specified for a simple semiparametric model with normal response. In this example, $q = 1$ for a single nonparametric smoothing function, such that

$$\mathbf{y} = \beta_1 + \beta_2 x + \sum_{k=1}^{K_1} u_k^{(1)} (x - \kappa_k^{(1)})_+ + \boldsymbol{\epsilon}, \tag{3.10}$$

where $\boldsymbol{\beta} = (\beta_1, \beta_2)^T$ are the fixed effects parameters, $\kappa_1^{(1)}, \ldots, \kappa_{K_1}^{(1)}$ are the knots of the truncated line basis for x, $\mathbf{u} = (u_1^{(1)}, \ldots, u_{K_1}^{(1)})^T$ are the nonparametric parameters, and $\boldsymbol{\epsilon} = (\epsilon_1, \ldots, \epsilon_n)^T$ are the error parameters. The truncated line functions $(x - \kappa_k)_+$ are strictly positive, where

$$\begin{aligned} (x - \kappa_k^{(1)})_+ &= x - \kappa_k^{(1)} && \forall (x - \kappa_k^{(1)}) > 0, \\ &= 0 && \forall (x - \kappa_k^{(1)}) \leq 0. \end{aligned}$$

The design matrices are defined,

$$\mathbf{X} = \begin{bmatrix} 1 & x_1 \\ . & . \\ . & . \\ . & . \\ 1 & x_n \end{bmatrix} \qquad \text{and} \qquad \mathbf{Z} = \begin{bmatrix} (x_1 - \kappa_1^{(1)})_+ & \dots & (x_1 - \kappa_{K_1}^{(1)})_+ \\ . & \dots & . \\ . & \dots & . \\ . & \dots & . \\ (x_n - \kappa_K^{(1)})_+ & \dots & (x_n - \kappa_{K_1}^{(1)})_+ \end{bmatrix}.$$

One might consider estimating the nonparametric coefficients $\mathbf{u}$ directly along with the fixed effects parameters $\boldsymbol{\beta}$. While this approach would simplify the estimation process, this specification will typically overfit the nonlinear relationship, overweighting $\mathbf{u}$ and producing a non-smooth curve when plotting the fitted response values vs.~$f(x)$. Therefore, **bayesGAM** always estimates $\mathbf{u}$ as random effect parameters with more restricted priors.

Continuing with the simple semiparametric model example, the likelihood and log likelihood for (3.10) are specified omitting constants,

$$f(\mathbf{y}|\mathbf{X}, \mathbf{Z}, \boldsymbol{\beta}, \mathbf{u}, \sigma_\epsilon^2) \propto (\sigma_\epsilon^2)^{-n/2} e^{-\frac{1}{2\sigma_\epsilon^2}(\mathbf{y}-\mathbf{X}\boldsymbol{\beta}-\mathbf{Z}\mathbf{u})^T(\mathbf{y}-\mathbf{X}\boldsymbol{\beta}-\mathbf{Z}\mathbf{u})},$$

$$\log f(\mathbf{y}|\mathbf{X}, \mathbf{Z}, \boldsymbol{\beta}, \mathbf{u}, \sigma_\epsilon^2) \propto -\frac{n}{2}\log(\sigma_\epsilon^2) - \frac{1}{2\sigma_\epsilon^2}(\mathbf{y} - \mathbf{X}\boldsymbol{\beta} - \mathbf{Z}\mathbf{u})^T(\mathbf{y} - \mathbf{X}\boldsymbol{\beta} - \mathbf{Z}\mathbf{u}).$$

For a Bayesian approach to fitting these models, I assign normal or student-t priors to $\boldsymbol{\beta}$. For example, $\boldsymbol{\beta} \sim N(0, \sigma_\beta^2)$ where σ_β^2 is a hyperparameter set automatically by the software or manually by the user. The error parameters are assigned a prior restricted to strictly positive values, such as a half-normal or half-t distribution. For example, $\boldsymbol{\epsilon} \sim N(0, \sigma_\epsilon^2 \mathbf{I}_n)$ where $\boldsymbol{\epsilon} \in (0, \infty)$. The distribution of the random effects $\mathbf{u} \sim N(0, \mathbf{G})$ is also treated as a prior. The diagonal hyperparameters $\boldsymbol{\lambda}$ are assigned either half-normal or half-t priors to ensure strictly positive estimations. %Finally, $(\epsilon_1, \dots, \epsilon_n)^T$ are also assigned half-normal or half-t priors, with hyperparameters set automatically by the software or manually by the user. Note that Stan automatically applies transformations for constrained parameters to correctly sample in HMC.

From Bayes formula, the log posterior is proportional to the log likelihood plus the log prior,

$$\log f(\boldsymbol{\beta}, \mathbf{u}|\mathbf{y}, \mathbf{X}, \mathbf{Z}, \sigma_\epsilon^2, \sigma_\beta^2, \mathbf{G}) \propto \log f(\mathbf{y}|\mathbf{X}, \mathbf{Z}, \boldsymbol{\beta}, \mathbf{u}, \sigma_\epsilon^2) - \frac{\boldsymbol{\beta}^T\boldsymbol{\beta}}{2\sigma_\beta^2} - \frac{1}{2}\mathbf{u}^T\mathbf{G}^{-1}\mathbf{u}. \tag{3.11}$$

Intuitively, the log prior for $\mathbf{u}$ functions to lower the log posterior for high values of $||\mathbf{u}||$. The structure of this model therefore *penalizes* the log posterior for overfitting $\mathbf{u}$. This produces a smoother nonlinear relationship between y and $f(x)$.

The TPS basis function can be extended to bivariate smoothing, which I apply in **bayesGAM**. The automated bivariate spline smoother to obtain K knots $\boldsymbol{\kappa}_1, ..., \boldsymbol{\kappa}_K \in \mathbf{R}^2$ is adopted from Ruppert, Wand, and Carroll (2003) 's algorithm in section 13.5, which is designed to be computationally efficient for semiparametric regression and easily implemented using R software. The steps of the bivariate smoothing algorithm as applied in **bayesGAM** are

1. Automatically choose the number of knots $K = \max[20, \min(n/4, 150)]$.
2. Apply a clustering algorithm designed for large applications to automatically select the knots.
3. Compute the design matrices for the bivariate smoothing (biv),
    - $\mathbf{X}^{\text{biv}} = [1 \ \ \mathbf{x}_i]$
    - $\mathbf{Z}_K^{\text{biv}} = [||\mathbf{x}_i - \boldsymbol{\kappa}_k||^2 \log||\mathbf{x}_i - \boldsymbol{\kappa}_k||]_{1 \le i \le n}$
    - $\boldsymbol{\Omega} = [||\boldsymbol{\kappa}_k - \boldsymbol{\kappa}_k'||^2 \log||\boldsymbol{\kappa}_k - \boldsymbol{\kappa}_{k'}||]_{1 \le k,\, k' \le K}$
4. Compute the singular value decomposition (SVD) of $\boldsymbol{\Omega}$.
5. Use the result from SVD to obtain $\boldsymbol{\Omega}^{1/2}$.
6. Compute the random effects design matrix $\mathbf{Z}^{\text{biv}} = \mathbf{Z}_K^{\text{biv}}\boldsymbol{\Omega}^{1/2}$.

With the development of the modeling framework for univariate response models complete, I continue with the extension to multivariate responses.

3.5.1 Framework for Multivariate Response Modeling

The **bayesGAM** package fits multivariate response models based on a generalized design from H. Liu, Tu, and others (2012) and Li, Liu, and Tu (2017). Given $i = 1, \ldots, m$ subjects, the $r > 1$ responses for the ith subject and $l = 1, \ldots, n_i$ observations for subject i can be specified in matrix form,

$$\mathbf{Y}_i = (\mathbf{y}_i^{(1)}, \ldots, \mathbf{y}_i^{(r)}) = \begin{pmatrix} y_{i1}^{(1)} & \ldots & y_{i1}^{(r)} \\ y_{i2}^{(1)} & \ldots & y_{i2}^{(r)} \\ . & \ldots & . \\ y_{in_i}^{(1)} & \ldots & y_{in_i}^{(r)} \end{pmatrix}, \tag{3.12}$$

where $\mathbf{Y}_i \in \mathbb{R}^{n_i \times r}$.

The multiple outcomes are modeled,

$$\begin{aligned} g\left[y_{il}^{(1)} | \mathbf{x}_{il}, \mathbf{z}_{il}, \boldsymbol{\beta}^{(1)}, \mathbf{u}_i^{(1)}, U_i^{(1)}\right] &= U_i^{(1)} + \mathbf{x}_{il}^T \boldsymbol{\beta}^{(1)} + \mathbf{z}_{il}^T \mathbf{u}_i^{(1)} \\ &\cdots \\ g\left[y_{il}^{(r)} | \mathbf{x}_{il}, \mathbf{z}_{il}, \boldsymbol{\beta}^{(r)}, \mathbf{u}_i^{(r)}, U_i^{(r)}\right] &= U_i^{(r)} + \mathbf{x}_{il}^T \boldsymbol{\beta}^{(r)} + \mathbf{z}_{il}^T \mathbf{u}_i^{(r)}, \end{aligned} \tag{3.13}$$

where $\widetilde{\mathbf{U}}_i = (U_i^{(1)}, ..., U_i^{(r)})^T$ is the random subject effect vector, $\boldsymbol{\beta}^{(1)}, \ldots, \boldsymbol{\beta}^{(r)}$ are each $p \times 1$ vectors of fixed parameters, and $\mathbf{u}_i^{(1)}, \ldots, \mathbf{u}_i^{(r)}$ are each $q \times 1$ vectors for the nonparametric smoothing parameters for the ith subject.

The r outcomes therefore share the same covariates, but include different fixed effect parameters, random subject effect parameters, and nonparametric smoothing parameters for each response. The random effect for each subject is assumed to be normally distributed, such that $\widetilde{\mathbf{U}}_i \sim N(\mathbf{0}, \boldsymbol{\Sigma}_u)$. This software application uses modified Cholesky decomposition as recommended by Chan and Jeliazkov (Chan and Jeliazkov 2009) for simulated estimation of $\mathbf{\Sigma}_u$, but with two key differences. First, the modified Cholesky decomposition is applied directly to the covariance matrix $\mathbf{\Sigma}_u = \mathbf{L}\mathbf{D}\mathbf{L}^T$ since a known distribution (e.g.~Wishart) is not strictly necessary for the model. Second, the priors for $\widetilde{\boldsymbol{\lambda}} = (\widetilde{\lambda}_1, ..., \widetilde{\lambda}_r)^T$, the diagonal elements of $\mathbf{D}$, use half-t or half-normal priors instead of inverse gamma. These priors ensure that the posterior marginal distribution of $\widetilde{\boldsymbol{\lambda}}$ is strictly positive.

The lower triangular matrix $\mathbf{L}$ contains 1's on the diagonal and off-diagonal elements $a_{gh}, \quad 1 \leq h \leq g \leq r-1$. For $r > 2$,

$$\mathbf{L} = \begin{pmatrix} 1 & 0 & \dots & 0 \\ a_{21} & 1 & 0 & 0 \\ \dots & \dots & \dots & \dots \\ a_{r1} & a_{r2} & \dots & 1 \end{pmatrix} \quad \text{and} \quad \mathbf{D} = \begin{pmatrix} \widetilde{\lambda}_1 & 0 & \dots & 0 \\ 0 & \widetilde{\lambda}_2 & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & \widetilde{\lambda}_r \end{pmatrix}.$$

Further, I define the off-diagonal parameters of $\mathbf{L}$ as $\mathbf{a}_g = (a_{g1}, \dots, a_{g,g-1})^T$ and $\mathbf{a} = (\mathbf{a}_2^T, \dots, \mathbf{a}_{r-1}^T)^T$. The parameters $\mathbf{a}$ are unconstrained, and may use student-t or normal distribution priors centered at zero in the software. The modified Cholesky decomposition with the restrictions on $\mathbf{D}$ ensure that the simulated covariance matrix values for $\mathbf{\Sigma}_u$ will be positive definite (Newton 1988)(Pourahmadi 1999).

3.5.2 bayesGAM: a General Purpose Package for Modeling

A few software packages are currently available to fit semiparametric models. The **SemiPar** package (Wand et al. 2005) fits semiparametric models using frequentist techniques from the **nlme** package (Pinheiro et al. 2017). The **mgcv** (Simon Wood 2011) and **gamm4** (Simon Wood and Scheipl 2020) packages fit GAMs using frequentist techniques as well. These packages are reliable, robust, and computationally efficient for many applications of GAMs.

Traditional Bayesian techniques such MH and Gibbs sampling are also used to fit GAMs. The R (R Core Team 2017) packages **DPPackage** (Jara et al. 2011) and **BNSP**

(Papageorgiou and Marshall 2020) fit semiparametric models using MH, while **Bspmma** (Burr and others 2012) uses Gibbs sampling (Geman and Geman 1984), another traditional but more restrictive MCMC technique, to fit semiparametric models based on Dirichlet priors.

Fewer options for fitting generalized additive models are available for HMC. General purpose HMC software such as Stan (Gelman, Lee, and Guo 2015) and **PyMC** (Salvatier, Wiecki, and Fonnesbeck 2016) can be programmed to fit custom models such as semiparametric regression. Stan is a BUGS-like (Spiegelhalter et al. 1999) language for probabilistic Bayesian programming which uses HMC as the principal algorithm for fitting statistical models. **PyMC** is based on Python, which is popular in computer science but is typically less familiar to statisticians. All of these software packages are powerful options to analysts who have strong technical expertise in the requisite languages and sufficient understanding of the methodology to translate GAMs to these applications.

The R packages **MCMCglmm** (Hadfield and others 2010) and **brms** (Burkner 2017) are designed to fit multilevel models using MCMC. **MCMCglmm** uses MH and other traditional MCMC algorithms to fit these models. **brms** creates and compiles Stan code based on inputs provided by the user in R. Analysts can fit GAMs using these packages provided they have a strong background in the methodology and understand how to translate their models to Bayesian multilevel models to use the software.

The R package **rstanarm** uses pre-compiled Stan code to fit a wide variety of GAMs based on the exponential family of models. The types of models that can be fit using **rstanarm** closely match those available in the frequentist **gamm4** package. Analysts are also able to choose from a set of priors that are provided in the software. While **rstanarm** provides a wealth of capabilities for fitting GAMs, one feature not present is the ability to fit multivariate response GAMs. The flexible and powerful **brms** package provides the flexibility to fit many models, including multivariate response models, but requires a compiler to be build each individual model in `Stan` before running.

Analysts have powerful, readily available options to fit GAMs using HMC provided they have sufficiently high technical expertise. To program statistical models with `Stan` directly, for example, analysts must also be comfortable with installing and working with `C++` compilers on their pc's or servers. A case can certainly be made that learning to work with compilers is valuable for statisticians and data analysts. However, not everyone has the time or inclination to make this time investment, particularly if there is uncertainty whether newer computational techniques will offer practical benefits over more familiar software. Further, while the developers of HMC software provide comprehensive documentation on setup and installation and make themselves readily available for questions, the variety of operating systems and hardware frequently make installation time-consuming and difficult. Such challenges will inevitably continue to occur through no fault of the HMC software developers. For example, the recent `macOS` release `Catalina` has presented technical difficulties for application users and developers alike.

The **bayesGAM** package is designed to provide an easy-to-use option to fit univariate and multivariate response GAMs using HMC with few technical burdens. The `R` functions in this package use **rstan** (The Stan Development Team 2020) to call `Stan` routines that run the HMC simulations. The `Stan` code for these models is already translated to `C++` and pre-compiled for the user. The programming formulation for models in **bayesGAM** is designed to be familiar to statisticians and analysts who fit statistical models in `R` using base and contributed packages. Table 3.1 compares the functionality of `R` packages that use HMC to fit GAMs.

Table 3.1: GAM model fitting functionality from R packages

Feature	**rstanarm** stan_gamm4	**rstanarm** stan_mvmer	**brms**	**walker**	**bayesGAM**
Parametric models	y	y	y	y	y
Nonparametric models	y	n	y	n	y
Pre-compiled Stan code	y	y	n	n	y
Multivariate GLM	y	y	y	n	y
Multivariate nonparametric models	n	n	y	n	y
Autoregressive models	n	n	n	y	y

As will be detailed later, the decomposition and parameterization of the covariance matrix for the multivariate response is structured to maximize flexibility in prior specification and simulation. This approach enables modeling unstructured covariance matrices, which is often difficult to fit using available software packages due to the high dimensionality. Modern Bayesian graphics from the **bayesplot** package (Gabry and Mahr 2016) are directly integrated with the software, in addition to custom plotting functions for multivariate responses and nonparametric associations. My hope is that this software helps analysts with their current models and promotes the use of HMC for more general adoption in the statistical community. This package along with source code with examples are available on CRAN at https://cran.r-project.org/web/packages/bayesGAM/index.html.

The main function in the **bayesGAM** package is also called `bayesGAM`. Similar to many other **R** packages, this function supports model specification via formulas, denoted with the dependent variable(s) on the left-hand side and the independent variable(s) on the right hand side, separated by ~. An optional random intercept model can be specified with the `random` argument, similar to the **nlme** package (Pinheiro et al. 2017).

```
R> bayesGAM <- function (formula, random = NULL,
+   family = gaussian, data, offset,
+   beta = list(), eps = list(), lambda = list(),
+   spcontrol = list(qr = TRUE, mvindep=FALSE, ...),
+   method = "bayesGAMfit", ...)
```

Multivariate response models can be specified using the base `cbind` function, with each of the dependent variable names separated by commas. This specification is all that is needed for the software to recognize a multivariate model. When a random intercept is specified, the covariance between the multiple responses per subject is automatically modeled. Simulated values for the covariance matrix and its corresponding correlation matrix are stored in the results.

3.5.2.1 family: Currently supported distributions and link functions for the response

Currently, this software includes the capability to fit some of the most common of the exponential family of distributions: gaussian, binomial, and poisson. The link functions for these distributions are the same as those supported in the `family` function in base R (Agresti 2015) (Venables and Ripley 2002).

- Gaussian family supports the `identity`, `log`, and `identity` link functions.
- Binomial family supports logistic, normal, and cauchy cumulative distribution functions, labeled `logit`, `probit`, and `cauchit`, respectively.
- Poisson family supports the `log`, `identity`, and `sqrt` link functions

3.5.2.2 np: Nonparametric smoothing function

Smoothing functions can be specified by the `np` function. For univariate smoothing, truncated cubic polynomial basis functions are the default with automated knot selection. The user may optionally set a different degree for truncated polynomial basis functions or TPS basis functions as desired. Automated knot selection is available as the default option in `bayesGAM`.

The automated knot selection can be overridden by the user by providing the number of knots, in which case the software automatically selects the knots based on quantiles of the distribution. Alternatively, the user may provide a numeric vector of knot locations for univariate smoothing.

Bivariate smoothing based on thin-plate splines is employed when two variables are passed to `np`. By default, bivariate function knots are automatically selected by the software. The user may override the number of knots in the bivariate case by parameter.

3.5.2.3 L: Create lagged variables

A convenience function `L` is provided with the package to specify autoregressive models directly in the formula. Multiple lags can be created at once as specified by the user. This function assumes that the data is pre-sorted in ascending time order. Optionally, lag functions can be applied at a group level (e.g., for each subject id). To use this feature, the data must be sorted by id first, then time. The design matrices are automatically adjusted to remove cases with missing values after applying the `L` function.

3.5.2.4 prior: Specifying distributions for the priors

Prior distributions are specified for the model parameters $\boldsymbol{\beta}$, $\boldsymbol{\epsilon}$, $\boldsymbol{\lambda}$, and $\mathbf{a}$ with function parameters `beta`, `eps`, `lambda`, and `a`, respectively. The $\boldsymbol{\beta}$ parameters are the fixed effects parameters. The $\boldsymbol{\epsilon}$ parameter(s) are for the error terms of gaussian response models. The $\boldsymbol{\lambda}$ parameter(s) are included for models with nonparametric smoothing functions or random intercepts. When both random intercepts and v nonparametric smoothing are specified in univariate response models, the first parameter $\lambda_1 \in \boldsymbol{\lambda}$ applies to random intercepts and all other $\lambda_2, ..., \lambda_{v+1} \in \boldsymbol{\lambda}$ apply to the smoothing function. In multivariate response models, the first parameters in $\boldsymbol{\lambda}$ are applied to $\widetilde{\lambda}_1, \ldots, \widetilde{\lambda}_r \in \boldsymbol{\lambda}$, while all remaining parameters $\lambda_{r+1}, \ldots, \lambda_{v+r} \in \boldsymbol{\lambda}$ are applied to the nonparametric smoothing functions.

The prior distributions that are currently available are the normal and central (2-parameter) student-t distributions, specified with `normal` and `st` inputs, respectively. For $\boldsymbol{\beta}$ and $\mathbf{a}$, these priors assume support of $\mathbb{R}$. The prior distributions for $\boldsymbol{\epsilon}$ are $\boldsymbol{\lambda}$ automatically constrained to the support of $(0, \infty)$, corresponding to half-normal and half-t distributions. These constraints are passed to `Stan`, which automatically handles parameter transformations required for HMC. When prior distributions are not specified explicitly, `bayesGAM` uses vague normal priors for $\boldsymbol{\beta}$ and $\mathbf{a}$, and student-t priors for $\boldsymbol{\epsilon}$ and $\boldsymbol{\lambda}$.

3.5.2.5 Additional controls for fitting GAMs

Special controls are currently provided for QR decomposition via a logical functional parameter set in a list passed `spcontrol`. To facilitate efficient simulation, QR decomposition is used for the design matrices $\mathbf{X}$ and $\mathbf{Z}$ when `QR = TRUE`. Since the computational efficiency gained by using QR decomposition for HMC can be significant, QR decomposition is the default setting for `bayesGAM`. The method for fitting models using `bayesGAM` is called *bayesGAMfit*, which prepares the data for model fitting using the `R` interface to `Stan`, **rstan** (The Stan Development Team 2020). For multivariate response models with random intercepts, `mvindep` determines whether to model the responses with an unconstrained covariance $\boldsymbol{\Sigma}_u$, the default `FALSE`, or a diagonal covariance by setting `mvindep=TRUE`.

Additional parameters can be passed to the `rstan::sampling` function through the `...` argument in `bayesGAM`. This can be used to fine tune technical parameters used in `Stan`, such as those supporting the No U-turn Sampler (NUTS) algorithm (Hoffman and Gelman 2014). For example, specifying

```
control=list(adapt_delta=0.98), cores=4
```

in `bayesGAM` will pass both the `control` list and the `adapt_delta` parameters to `rstan::sampling`, as well as the number of CPU cores for parallel processing.

The default `summary` method displays posterior quantiles, $\hat{R}$ statistics, and effective sample sizes for each of the simulated parameters based on the method in **rstan** (The Stan Development Team 2020). The prior distributions used to fit the model can be displayed via the `showPrior` function.

The default `plot` method for objects created by `bayesGAM` uses functionality from **ggplot2** (Wickham 2016). The mean response is plotted against the independent variable(s), with smoothing automatically displayed for nonparametric functions. Credible intervals are also displayed based on a subsample of the MCMC results. In the bivariate smoothing case, a contour plot is displayed for each dependent variable. Additional plotting is available through the plot functions available for `stanfit` objects. Available plots include posterior intervals, trace plots, histograms, $\hat{R}$ statistics, effective sample size, and autocorrelation. Functions from the **bayesplot** package (Gabry and Mahr 2016) can also be used for objects created by `bayesGAM`.

3.6 Fitting a Bivariate Response GAM Using HMC

This example models the joint distribution of *diastolic* and *systolic* blood pressure of children and young adults. The modeling of the covariance matrix of the joint response $\mathbf{\Sigma}_b$ is unconstrained, such that the correlation of the response variables can be evaluated. The model specification for this data is

$$\mathrm{DBP}_{ij} = U_i^d + \beta_0^d + \beta_1^d \mathrm{SexM}_i + \beta_2^d \mathrm{RaceW}_i + f^d(\mathrm{WEIGHT}_{ij}, \mathrm{age}_{ij}) + \epsilon_{ij}^d$$

$$\mathrm{SBP}_{ij} = U_i^s + \beta_0^s + \beta_1^s \mathrm{SexM}_i + \beta_2^s \mathrm{RaceW}_i + f^s(\mathrm{WEIGHT}_{ij}, \mathrm{age}_{ij}) + \epsilon_{ij}^s$$

for observations $i = 1, ..., m$ subjects and $j = 1, ..., n_j$ observations per subject. The multivariate response is indicated using the base R function `cbind`. The random intercepts are generated from the subject ID's. Note that the random intercepts must be specified as factors. Normal priors are selected for $\boldsymbol{\beta}$, half-normal priors are set for $\boldsymbol{\lambda}$, and half-t priors for $\boldsymbol{\epsilon}$. The `spcontrol` parameter `mvindep` is set to `FALSE` to indicate that $\boldsymbol{\Sigma}_b$ is unconstrained.

```
R>  fbp_corr <- bayesGAM(cbind(dias, sys) ~
+                       SexM + RaceW + np(WEIGHT, age),
+                       random = ~ factor(ID),
+                       data = bpdatakeep,
+                       beta = normal(c(0, 50)),
+                       lambda = normal(c(0, 1)),
+                       eps = st(c(4, 0, 3)),
+                       spcontrol=list(mvindep = FALSE),
+                       family = "gaussian",
+                       cores=4, chains=4, iter=2000)
```

A normal prior is specified for $\boldsymbol{\beta}$, half-normal for λ, and half-t for ϵ. The prior for a is a vague normal prior default from the software. A summary view is displayed showing the quantiles of the posterior distribution of the parameters. The $\hat{R}$ statistic is shown for each of the parameters are close to 1 for all parameters indicating that the chains mixed well.

```
R> summary(fbp_corr, probs=c(0.025, 0.50, 0.975))
```

```
Inference for Stan model: multresponse_continuous.
4 chains, each with iter=2000; warmup=1000; thin=1;
post-warmup draws per chain=1000, total post-warmup draws=4000.

                     mean se_mean   sd  2.5%  50%  97.5% n_eff Rhat
eps[1]               7.87    0.01 0.30  7.31  7.87  8.50  3224    1
eps[2]               7.59    0.01 0.29  7.06  7.57  8.19  3317    1
a[1]                 0.58    0.00 0.20  0.21  0.58  0.98  1590    1
```

```
beta_(Intercept)[1] 55.43    0.10 4.28 46.19 55.49 64.57  1703    1
beta_SexM[1]        -0.40    0.05 1.64 -3.70  0.71  2.83  1168    1
beta_RaceW[1]        1.20    0.05 1.67 -2.06  1.25  4.52  1362    1
beta_WEIGHT[1]       0.19    0.00 0.05  0.09  0.18  0.28  2492    1
beta_age[1]         -0.25    0.00 0.21 -0.67 -0.25  0.17  2940    1
beta_(Intercept)[2] 89.73    0.18 5.25 77.95 90.08 99.52   826    1
beta_SexM[2]         1.07    0.05 1.72 -2.30  1.05  4.38  1411    1
beta_RaceW[2]        0.41    0.05 1.70 -2.86  0.40  3.90  1254    1
beta_WEIGHT[2]       0.39    0.00 0.05  0.29  0.39  0.48  2070    1
beta_age[2]         -0.59    0.00 0.21 -0.99 -0.59 -0.18  2588    1

Samples were drawn using NUTS(diag_e) at Tue Mar  9 20:29:52 2021.
For each parameter, n_eff is a crude measure of effective sample size,
and Rhat is the potential scale reduction factor on split chains (at
convergence, Rhat=1).
```

The nonparametric relationship of *WEIGHT* and *age* is displayed using the default `plot` method.

```
R> plot(fbp_corr)
```

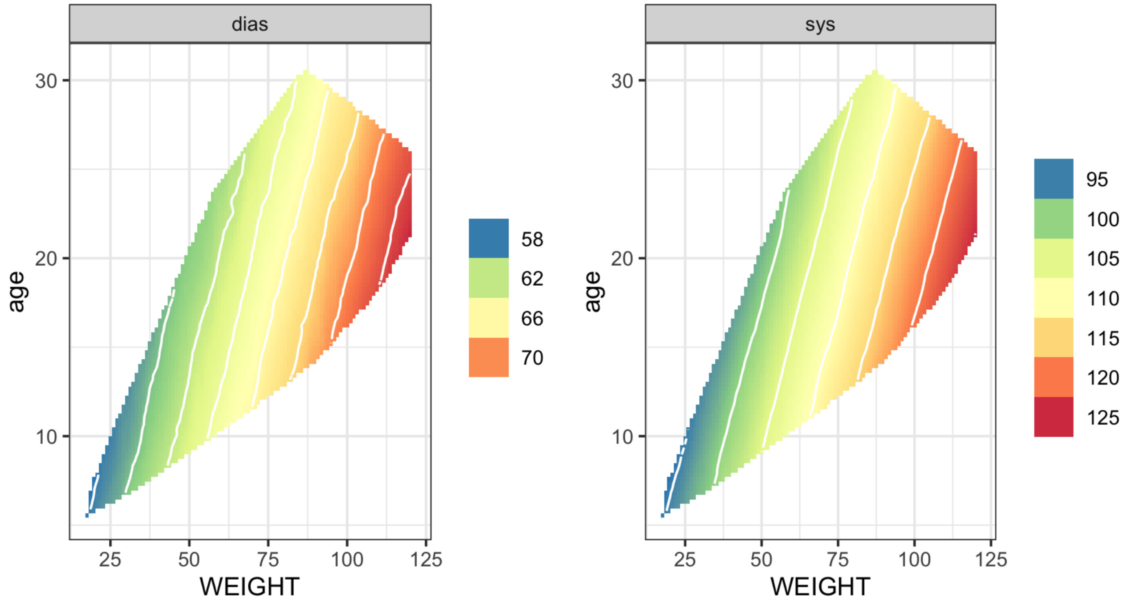

Figure 3.1: Contour plot of the bivariate response of diastolic and systolic blood pressure by bayesGAM

Figure 3.1 shows that blood pressure has a nonlinear positive association with weight and a slight negative association with age.

The correlation of diastolic and systolic blood pressure is 0.56, similar to the results from H. Liu, Tu, and others (2012), as shown in Figure 3.2.

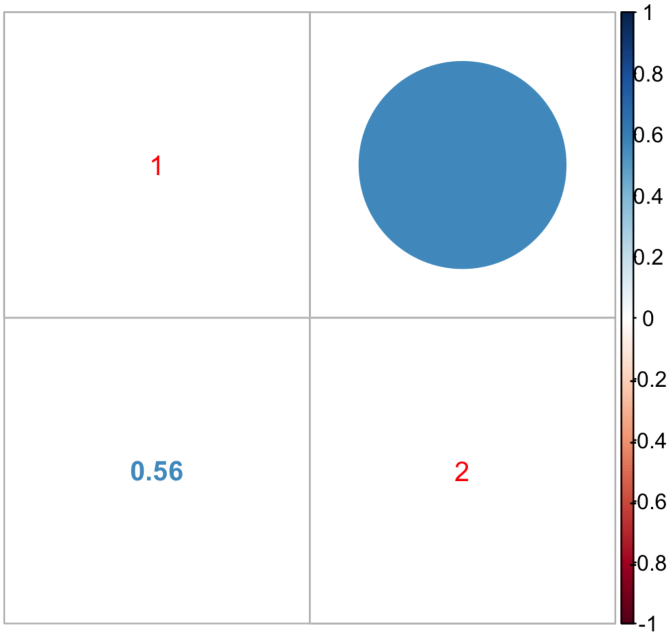

Figure 3.2: Correlation of the bivariate response of diastolic and systolic blood pressure to weight and age

3.7 Extending the Application of HMC in Statistics

This research develops a Bayesian computational framework for statisticians to fit standard and non-standard statistical models. This framework includes a comprehensive description of HMC from a statistical point of view, a process to fit a wide variety of models, and the tools to use these methods. The benefit of this research lies in the flexibility and computational efficiency of this framework when applied to complex models with high-dimensional data.

An example application of this framework was applied to fit a non-standard multiple response GAM. This approach was applied to a real biostatistical application of a study on blood pressure. The results of this study and other applications demonstrate the power and validity of these computational methods. Numerous opportunities for expanded applications of this framework to grow as methodological research continues to produce increasingly complex statistical models.

While multiple response GAM's have a relatively narrow application, the methods developed in this research are widely applicable to many types of statistical models. The application of this Bayesian framework to multiple response GAM's provides strategies for estimating correlating parameters, setting effective priors, and parameterization considerations. These strategies offer contributions to statistical computation beyond fitting a single class of non-standard models. As the design and development of statistical models continue to evolve, the HMC algorithm and methods to apply HMC effectively will be required to support the rapidly evolving field of biostatistical research.

REFERENCES

Abadi, Martín, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, et al. 2016. "TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems." *arXiv:1603.04467*, March. http://arxiv.org/abs/1603.04467.

Abraham, Ralph, and Jerrold E. Marsden. 1978. *Foundations of Mechanics.* American Mathematical Soc.

Agresti, Alan. 2015. *Foundations of Linear and Generalized Linear Models.* Hoboken, NJ: Wiley.

Arnol'd, V. I. 2013. *Mathematical Methods of Classical Mechanics.* Springer Science & Business Media.

Arnold, Barry C., and S. James Press. 1989. "Compatible Conditional Distributions." *Journal of the American Statistical Association* 84 (405): 152–56. https://doi.org/10.1080/01621459.1989.10478750.

Barnard, John, Robert McCulloch, and Xiao-Li Meng. 2000. "Modeling Covariance Matrices in Terms of Standard Deviations and Correlations, with Application to Shrinkage." *Statistica Sinica*, 1281–311.

Barndorff-Nielsen, O. E., D. R. Cox, and N. Reid. 1986. "The Role of Differential Geometry in Statistical Theory." *International Statistical Review / Revue Internationale de Statistique* 54 (1): 83–96. https://doi.org/10.2307/1403260.

Bates, Douglas, Deepayan Sarkar, Maintainer Douglas Bates, and L Matrix. 2007. "The Lme4 Package." *R Package Version* 2 (1): 74.

Bellman, Richard Ernest. 1957. *Dynamic Programming.* Princeton University Press.

Besag, Julian, Peter Green, David Higdon, and Kerrie Mengersen. 1995. "Bayesian Computation and Stochastic Systems." *Statistical Science* 10 (1): 3–41. https://doi.org/10.1214/ss/1177010123.

Beskos, A., N. S. Pillai, G. O. Roberts, J. M. Sanz-Serna, and A. M. Stuart. 2010. "The Acceptance Probability of the Hybrid Monte Carlo Method in High-dimensional Problems." *AIP Conference Proceedings* 1281 (1): 23–26. https://doi.org/10.1063/1.3498436.

Best, Nicky, Mary Kathryn Cowles, and Karen Vines. 1995. "CODA* Convergence Diagnosis and Output Analysis Software for Gibbs Sampling Output Version 0.30." *MRC Biostatistics Unit, Cambridge* 52.

Betancourt, M. 2017. "The QR Decomposition for Regression Models." http://mc-stan.org/users/documentation/case-studies/qr_regression.html.

Betancourt, M. J. 2013. "Generalizing the No-u-Turn Sampler to Riemannian Manifolds." *arXiv:1304.1920 [Stat]*, April. http://arxiv.org/abs/1304.1920.

Betancourt, M. J., and Mark Girolami. 2013. "Hamiltonian Monte Carlo for Hierarchical Models." *arXiv:1312.0906 [Stat]*, December. http://arxiv.org/abs/1312.0906.

Betancourt, Michael. 2017. "A Conceptual Introduction to Hamiltonian Monte Carlo." *arXiv:1701.02434 [Stat]*, January, 1–60. http://arxiv.org/abs/1701.02434.

Betancourt, Michael, Simon Byrne, Sam Livingstone, and Mark Girolami. 2017. "The Geometric Foundations of Hamiltonian Monte Carlo." *Bernoulli* 23 (4): 2257–98. https://doi.org/10.3150/16-BEJ810.

Bolker, Ben. 2018. "GLMM Worked Examples." https://bbolker.github.io/mixedmodels-misc/ecostats_chap.html.

Bonamente, Massimiliano. 2016. *Statistics and Analysis of Scientific Data*. Springer.

Brooks, Stephen P. 1998. "Markov Chain Monte Carlo Method and Its Application." *Journal of the Royal Statistical Society. Series D (The Statistician)* 47 (1): 69–100. http://www.jstor.org/stable/2988428.

Brooks, Stephen P., and Andrew Gelman. 1998. "General Methods for Monitoring Convergence of Iterative Simulations." *Journal of Computational and Graphical Statistics* 7 (4): 434–55. https://doi.org/10.1080/10618600.1998.10474787.

Burkner, Paul-Christian. 2017. "Brms: An R Package for Bayesian Multilevel Models Using Stan." *Journal of Statistical Software* 80 (1): 1–28. https://doi.org/10.18637/jss.v080.i01.

Burr, Deborah, and others. 2012. "Bspmma: An r Package for Bayesian Semiparametric Models for Meta-Analysis." *Journal of Statistical Software* 50 (4): 1–23.

Calderhead, Ben. 2011. "Differential Geometric MCMC Methods and Applications." PhD thesis, University of Glasgow. http://encore.lib.gla.ac.uk/iii/encore/record/C___Rb2922509.

Cancès, Eric, Frédéric Legoll, and Gabriel Stoltz. 2007. "Theoretical and Numerical Comparison of Some Sampling Methods for Molecular Dynamics." *ESAIM: Mathematical*

Modelling and Numerical Analysis 41 (2): 351–89. https://doi.org/10.1051/m2an:2007014.

Carlin, Bradley P., and Thomas A. Louis. 2008. *Bayesian Methods for Data Analysis*. Boca Raton, FL: CRC Press.

Carpenter, Bob, Matthew D Hoffman, Marcus Brubaker, Daniel Lee, Peter Li, and Michael Betancourt. 2015. "The Stan Math Library: Reverse-Mode Automatic Differentiation in C++." *arXiv Preprint arXiv:1509.07164*.

Casella, George, and Edward I. George. 1992. "Explaining the Gibbs Sampler." *The American Statistician* 46 (3): 167–74. https://doi.org/10.2307/2685208.

Chan, Joshua Chi-Chun, and Ivan Jeliazkov. 2009. "MCMC Estimation of Restricted Covariance Matrices." *Journal of Computational and Graphical Statistics* 18 (2): 457–80.

Channell, P. J., and C. Scovel. 1990. "Symplectic Integration of Hamiltonian Systems." *Nonlinearity* 3 (2): 231. https://doi.org/10.1088/0951-7715/3/2/001.

Chen, Tianqi, Emily B. Fox, and Carlos Guestrin. 2014. "Stochastic Gradient Hamiltonian Monte Carlo." In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, II-1683-II-1691. ICML'14. Beijing, China: JMLR.org. http://dl.acm.org/citation.cfm?id=3044805.3045080.

Chib, Siddhartha, and Edward Greenberg. 1995. "Understanding the Metropolis-Hastings Algorithm." *The American Statistician* 49 (4): 327–35.

Cover, Thomas M., and Joy A. Thomas. 2012. *Elements of Information Theory*. John Wiley & Sons.

Cowles, Mary Kathryn, and Bradley P. Carlin. 1996. "Markov Chain Monte Carlo Convergence Diagnostics: A Comparative Review." *Journal of the American Statistical Association* 91 (434): 883–904. http://www.jstor.org/stable/2291683.

Diaconis, Persi, Kshitij Khare, and Laurent Saloff-Coste. 2008. "Gibbs Sampling, Exponential Families and Orthogonal Polynomials." *Statistical Science* 23 (2): 151–78. https://doi.org/10.1214/07-STS252.

Duane, Simon, A. D. Kennedy, Brian J. Pendleton, and Duncan Roweth. 1987. "Hybrid Monte Carlo." *Physics Letters B* 195 (2): 216–22. https://doi.org/https://doi.org/10.1016/0370-2693(87)91197-X.

Durmus, Alain, Eric Moulines, and Eero Saksman. 2017. "On the Convergence of Hamiltonian Monte Carlo." *arXiv Preprint arXiv:1705.00166*, 1–45.

Efron, Bradley. 1978. "The Geometry of Exponential Families." *The Annals of Statistics* 6 (2): 362–76. https://doi.org/10.1214/aos/1176344130.

Evans, Michael, and Timothy Swartz. 2000. *Approximating Integrals via Monte Carlo and Deterministic Methods*. OUP Oxford.

Fahrmeir, Ludwig, Heinz Kaufmann, and others. 1985. "Consistency and Asymptotic Normality of the Maximum Likelihood Estimator in Generalized Linear Models." *The Annals of Statistics* 13 (1): 342–68.

Fox, Gordon A, Simoneta Negrete-Yankelevich, and Vinicio J Sosa. 2015. *Ecological Statistics: Contemporary Theory and Application*. Oxford University Press, USA.

Gabry, Jonah, and T Mahr. 2016. “Bayesplot: Plotting for Bayesian Models. R Package Version 1.1.0.”

Gelman, Andrew. 2006. “Prior Distributions for Variance Parameters in Hierarchical Models (Comment on Article by Browne and Draper).” *Bayesian Analysis* 1 (3): 515–34.

Gelman, Andrew, John B. Carlin, Hal S. Stern, David B. Dunson, Aki Vehtari, and Donald B. Rubin. 2013. *Bayesian Data Analysis.* CRC Press.

Gelman, Andrew, Daniel Lee, and Jiqiang Guo. 2015. “Stan: A Probabilistic Programming Language for Bayesian Inference and Optimization.” *Journal of Educational and Behavioral Statistics* 40 (5): 530–43.

Gelman, Andrew, and Donald B. Rubin. 1992. “Inference from Iterative Simulation Using Multiple Sequences.” *Statistical Science* 7 (4): 457–72. https://doi.org/10.1214/ss/1177011136.

Geman, Stuart, and Donald Geman. 1984. “Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images.” *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-6 (6): 721–41.

Gibbs, Josiah Willard. 1902. *Elementary Principles in Statistical Mechanics: Developed with Especial Reference to the Rational Foundations of Thermodynamics.* C. Scribner’s sons.

Gilks, W. R., N. G. Best, and K. K. C. Tan. 1995. “Adaptive Rejection Metropolis Sampling Within Gibbs Sampling.” *Journal of the Royal Statistical Society. Series C (Applied Statistics)* 44 (4): 455–72. http://www.jstor.org/stable/2986138.

Gilks, W. R., S. Richardson, and David Spiegelhalter. 1995. *Markov Chain Monte Carlo in Practice*. CRC Press.

Girolami, Mark, and Ben Calderhead. 2011. "Riemann Manifold Langevin and Hamiltonian Monte Carlo Methods." *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 73 (2): 123–214.

Girolami, Mark, Ben Calderhead, and Siu A. Chin. 2009. "Riemannian Manifold Hamiltonian Monte Carlo." *arXiv:0907.1100 [Math, Stat]*, July. http://arxiv.org/abs/0907.1100.

Hadfield, Jarrod D, and others. 2010. "MCMC Methods for Multi-Response Generalized Linear Mixed Models: The MCMCglmm r Package." *Journal of Statistical Software* 33 (2): 1–22.

Hairer, Ernst, Christian Lubich, and Gerhard Wanner. 2003. "Geometric Numerical Integration Illustrated by the Störmer–Verlet Method." *Acta Numerica* 12 (May): 399–450. https://doi.org/10.1017/S0962492902000144.

Hastie, Trevor J, and Robert J Tibshirani. 1990. *Generalized Additive Models*. Vol. 43. CRC press.

Hastings, W. K. 1970. "Monte Carlo Sampling Methods Using Markov Chains and Their Applications." *Biometrika* 57 (1): 97–109. https://doi.org/10.1093/biomet/57.1.97.

Hoffman, Matthew D, and Andrew Gelman. 2014. "The No-U-Turn Sampler: Adaptively Setting Path Lengths in Hamiltonian Monte Carlo." *Journal of Machine Learning Research* 15 (1): 1593–623.

Jain, P. K., Khalil Ahmad, and Om P. Ahuja. 1995. *Functional Analysis.* New Age International.

Jara, Alejandro, Timothy E Hanson, Fernando A Quintana, Peter Müller, and Gary L Rosner. 2011. "DPpackage: Bayesian Semi-and Nonparametric Modeling in r." *Journal of Statistical Software* 40 (5): 1.

Kelly, F. P. 2011. *Reversibility and Stochastic Networks.* New York, NY, USA: Cambridge University Press.

Laugwitz, Detlef. 2008. *Bernhard Riemann 1826-1866: Turning Points in the Conception of Mathematics.* Springer Science & Business Media.

Lewandowski, Daniel, Dorota Kurowicka, and Harry Joe. 2009. "Generating Random Correlation Matrices Based on Vines and Extended Onion Method." *Journal of Multivariate Analysis* 100 (9): 1989–2001.

Li, Zhuokai, Hai Liu, and Wanzhu Tu. 2017. "A Generalized Semiparametric Mixed Model for Analysis of Multivariate Health Care Utilization Data." *Statistical Methods in Medical Research* 26 (6): 2909–18.

Liu, Hai, Wanzhu Tu, and others. 2012. "A Semiparametric Regression Model for Paired Longitudinal Outcomes with Application in Childhood Blood Pressure Development." *The Annals of Applied Statistics* 6 (4): 1861–82.

Liu, Jun S., Wing Hung Wong, and Augustine Kong. 1994. "Covariance Structure of the Gibbs Sampler with Applications to the Comparisons of Estimators and Augmentation Schemes." *Biometrika* 81 (1): 27–40. https://doi.org/10.1093/biomet/81.1.27.

Livingstone, Samuel, Michael Betancourt, Simon Byrne, and Mark Girolami. 2016. "On the Geometric Ergodicity of Hamiltonian Monte Carlo." *arXiv:1601.08057 [Stat]*, January, 1–29. http://arxiv.org/abs/1601.08057.

MacKay, David J. C. 2003. *Information Theory, Inference and Learning Algorithms.* Cambridge University Press.

Mackenze, Paul B. 1989. "An Improved Hybrid Monte Carlo Method." *Physics Letters B* 226 (3-4): 369–71.

Metropolis, Nicholas, Arianna W Rosenbluth, Marshall N Rosenbluth, Augusta H Teller, and Edward Teller. 1953. "Equation of State Calculations by Fast Computing Machines." *The Journal of Chemical Physics* 21 (6): 1087–92.

Nakahara, Mikio. 2003. *Geometry, Topology and Physics, Second Edition.* CRC Press.

Neal, Radford. 2011. *Handbook of Markov Chain Monte Carlo.* Boca Raton, FL: CRC Press.

Neal, Radford M. 1993. "Probabilistic Inference Using Markov Chain Monte Carlo Methods." Technical Report CRG-TR-93-1. University of Toronto, Department of Computer Science.

Neal, Radford M. 2003. "Slice Sampling." *Annals of Statistics*, 705–41.

Newton, H Joseph. 1988. *Timeslab: A Time Series Analysis Laboratory.* Wadsworth Publ. Co.

O'Neill, Barrett. 1997. *Elementary Differential Geometry.* Academic Press.

Ozgul, Arpat, Madan K Oli, Benjamin M Bolker, and Carolina Perez-Heydrich. 2009. "Upper Respiratory Tract Disease, Force of Infection, and Effects on Survival of Gopher Tortoises." *Ecological Applications* 19 (3): 786–98.

Papageorgiou, Georgios, and Benjamin C Marshall. 2020. "Bayesian Semiparametric Analysis of Multivariate Continuous Responses, with Variable Selection." *Journal of Computational and Graphical Statistics*, 1–14.

Pinheiro, José, Douglas Bates, Saikat DebRoy, Deepayan Sarkar, Siem Heisterkamp, Bert Van Willigen, and R Maintainer. 2017. "Package 'Nlme'." *Linear and Nonlinear Mixed Effects Models, Version* 3.

Pourahmadi, Mohsen. 1999. "Joint Mean-Covariance Models with Applications to Longitudinal Data: Unconstrained Parameterisation." *Biometrika* 86 (3): 677–90.

R Core Team. 2017. "R: A Language and Environment for Statistical Computing." R Foundation for Statistical Computing; Vienna, Austria. https://www.R-project.org.

Rao, C. Radhakrishna. 1945. "Information and the Accuracy Attainable in the Estimation of Statistical Parameters" 37 (3): 81–91.

Ripley, Brian D. 1987. *Stochastic Simulation.* J. Wiley.

Robbins, Herbert, and Sutton Monro. 1951. "A Stochastic Approximation Method." *The Annals of Mathematical Statistics* 22 (3): 400–407. https://doi.org/10.1214/aoms/1177729586.

Robert, Christian. 2007. *The Bayesian Choice: From Decision-Theoretic Foundations to Computational Implementation.* New York: Springer Science & Business Media.

Roberts, G. O., and A. F. M. Smith. 1994. "Simple Conditions for the Convergence of the Gibbs Sampler and Metropolis-Hastings Algorithms." *Stochastic Processes and Their Applications* 49 (2): 207–16. https://doi.org/10.1016/0304-4149(94)90134-1.

Roberts, G. O., and R. L. Tweedie. 1996. "Geometric Convergence and Central Limit Theorems for Multidimensional Hastings and Metropolis Algorithms." *Biometrika* 83 (1): 95–110. https://doi.org/10.1093/biomet/83.1.95.

Roberts, Gareth O, Andrew Gelman, and Walter R Gilks. 1997. "Weak Convergence and Optimal Scaling of Random Walk Metropolis Algorithms." *The Annals of Applied Probability* 7 (1): 110–20.

Rosenthal, Jeffrey S. 1993. "Rates of Convergence for Data Augmentation on Finite Sample Spaces." *The Annals of Applied Probability* 3 (3): 819–39. https://doi.org/10.1214/aoap/1177005366.

Ruppert, David, Matt P Wand, and Raymond J Carroll. 2003. *Semiparametric Regression*. 12. Cambridge university press.

Salvatier, John, Thomas V Wiecki, and Christopher Fonnesbeck. 2016. "Probabilistic Programming in Python Using PyMC3." *PeerJ Computer Science* 2: e55.

Schervish, Mark J., and Bradley P. Carlin. 1992. "On the Convergence of Successive Substitution Sampling." *Journal of Computational and Graphical Statistics* 1 (2): 111–27. https://doi.org/10.1080/10618600.1992.10477008.

Serway, Raymond A., and Chris Vuille. 2012. *College Physics*. Cengage Learning.

Spiegelhalter, David, Andrew Thomas, Nicky Best, and Wally Gilks. 1999. "BUGS: Bayesian Inference Using Gibbs Sampling, Version 0.5 (Version Ii)." Medical Research Council Biostatistics Unit.

The Stan Development Team. 2020. "RStan: The R Interface to Stan." http://mc-stan.org/.

Theano Development Team. 2016. "Theano: A Python Framework for Fast Computation of Mathematical Expressions." *arXiv Preprint arXiv:1605.02688*, 1–19.

Thompson, Madeleine, and Radford M. Neal. 2010. "Covariance-Adaptive Slice Sampling." *arXiv:1003.3201 [Stat]*, March. http://arxiv.org/abs/1003.3201.

Tierney, Luke. 1994. "Markov Chains for Exploring Posterior Distributions." *Ann. Statist.* 22 (4): 1701–28. https://doi.org/10.1214/aos/1176325750.

Turner, Brandon M., Per B. Sederberg, Scott D. Brown, and Mark Steyvers. 2013. "A Method for Efficiently Sampling from Distributions with Correlated Dimensions." *Psychological Methods* 18 (3): 368–84. https://doi.org/10.1037/a0032222.

Vasicek, Oldrich. 1976. "A Test for Normality Based on Sample Entropy." *Journal of the Royal Statistical Society. Series B (Methodological)* 38 (1): 54–59. https://doi.org/10.2307/2984828.

Venables, William N, and Brian D Ripley. 2002. *Modern Applied Statistics with s-PLUS*. New York: Springer Science & Business Media.

Voss, Jochen. 2013. *An Introduction to Statistical Computing: A Simulation-Based Approach*. Hoboken, NJ: John Wiley & Sons.

Wand, MP, BA Coull, JL French, B Ganguli, EE Kammann, J Staudenmayer, and A Zanobetti. 2005. "SemiPar 1.0. R Package." *URL: Http://Cran. R-Project. Org.*

Wang, Ming Chen, and G. E. Uhlenbeck. 1945. "On the Theory of the Brownian Motion II." *Reviews of Modern Physics* 17 (2): 323–42. https://doi.org/10.1103/RevModPhys.17.323.

Whittaker, E. T., and G. Robinson. 1967. *The Calculus of Observations: An Introduction to Numerical Analysis*. 4th edition. Dover Publications.

Wickham, H. 2016. *Ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York. https://ggplot2.tidyverse.org.

Wood, S. 2017. *Generalized Additive Models: An Introduction with r*. CRC press.

Wood, Simon. 2011. "Fast Stable Restricted Maximum Likelihood and Marginal Likelihood Estimation of Semiparametric Generalized Linear Models." *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 73 (1): 3–36.

Wood, Simon N. 2003. "Thin Plate Regression Splines." *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 65 (1): 95–114.

Wood, Simon, and Fabian Scheipl. 2020. *Gamm4: Generalized Additive Mixed Models Using 'Mgcv' and 'Lme4'*. https://CRAN.R-project.org/package=gamm4.

CURRICULUM VITAE

Samuel Joseph Thomas

EDUCATION

- Ph.D. in Biostatistics, Indiana University, May 2021
- M.S. in Mathematics, Purdue University, Dec 2006
- B.S. in Electrical Engineering, University of Notre Dame, May 1995

PROFESSIONAL EXPERIENCE

- Data Scientist Senior, Capital Group Companies, Los Angeles, CA (Sep 2008 - Present)
- Adjunct Professor, Ivy Tech University, Department of Mathematics, Indianapolis, Indiana (Jan 2007 - May 2014)

SKILLS

- R, Python, SQL, Stan, LateX, Linux

PUBLICATIONS

Thomas, S. and Tu, W. (2020). *Hamiltonian Monte Carlo.* In Wiley StatsRef: Statistics Reference Online (eds N. Balakrishnan, T. Colton, B. Everitt, W. Piegorsch, F. Ruggeri and J.L. Teugels). doi:10.1002/9781118445112.stat08243

Thomas, S., and Tu, W. (2021). *Learning Hamiltonian Monte Carlo in R.* The American Statistician, published online Jan 2021.

Green, Brice and Thomas, Samuel, *Inference and Prediction of Stock Returns using Multilevel Models* (August 31, 2019). Available at SSRN: https://ssrn.com/abstract=3411358 or http://dx.doi.org/10.2139/ssrn.3411358

SOFTWARE PACKAGES

hmclearn: An R package to fit statistical models with Hamiltonian Monte Carlo. https://cran.r-project.org/web/packages/hmclearn/index.html

bayesGAM: An R package to fit semiparametric regression models using Hamiltonian Monte Carlo. https://cran.r-project.org/web/packages/bayesGAM/index.html

mlts: An R package to automatically develop forecasts and perform cross-validation for bottoms-up forecast models. Internal package for Capital Group Companies.

PRESENTATIONS

A Bayesian Analytical Software Based on Hamiltonian Monte Carlo. Regenstrief Institute, 12/4/2019. https://www.youtube.com/watch?v=sBA3lA0Nht0

Using Fourier Series to Model Daily Seasonal Patterns of Redemptions. Capital Group Companies, Data Science Interest Group, 2018

Improving Capacity and Financial Planning, a Guide to Business Forecasting with Alteryx. Inspire 2016 Alteryx Conference, San Diego, CA.

Predicting At-Risk Plans Using the C5 Algorithm. Capital Group Companies, Data Science Interest Group, 2015

UseR 2012 at Vanderbilt University. UseR 2012 Vanderbilt University.