University of California
Santa Barbara

Novel Computational Methods for Bayesian Hierarchical Modeling in the Biomedical Domain

A dissertation submitted in partial satisfaction
of the requirements for the degree

Doctor of Philosophy
in
Computer Science

by

Arya Alexander Pourzanjani

Committee in charge:

Professor Linda R. Petzold, Chair
Professor John R. Gilbert
Professor Paul J. Atzberger
Professor Alexander Franks

March 2019

The Dissertation of Arya Alexander Pourzanjani is approved.

Professor John R. Gilbert

Professor Paul J. Atzberger

Professor Alexander Franks

Professor Linda R. Petzold, Committee Chair

March 2019

Novel Computational Methods for Bayesian Hierarchical Modeling in the Biomedical Domain

by

Arya Alexander Pourzanjani

To Mom, Dad, Sos, Sophie, and all of my pugs

Acknowledgements

I would like to first and foremost thank my mom (Zohreh) and my sister (Paradis). Their helpfulness and generosity allowed me to focus on my studies and being the best researcher I could be. Those two along with my father (Mehran), Sophie, and my close friends have always been available for me whenever I needed support or advice. I would also like to thank my parents for stressing the importance of education throughout my life.

Secondly, I would also like to thank my collaborators who, over the years, have provided me with the resources and motivation I needed to complete my thesis. In particular, I would like to thank Ben Bales and Richard Jiang for always being around the lab to bounce ideas off of. Our conversations and debates about statistics, machine learning, and even politics and economics were always lively and enriching. I hope that wherever I go next can provide the same level of intellectual stimulation. I am also thankful to Tie Bo Wu, who helped me interpret much of coagulopathy work, and was always an open-minded and supportive friend. I am thankful to YY Zhang who served as an early mentor to me. I am thankful to my biological and clinical collaborators over the years, who have made much of this work possible, in particular, Michael Harrington who served a mentor to me. Next, I am very thankful to my committee members, John Gilbert, Paul Atzberger, and Alexander Franks for their helpful advice over the years. Lastly, I am thankful to my advisor, Linda Petzold, who has always believed in me and treated me as an equal. Her belief in me and her patience has instilled me with great confidence. I also would like to thank Linda for teaching me a lot of the ins and outs of numerically solving ODEs. It has been an absolute privilege to have learned the field from one of the best.

Lastly, I would like to thank the U.S. Army Research Office, which funded a large

portion of this work under grant W911NF-10-2-0114.

Curriculum Vitæ

Arya Alexander Pourzanjani

Education

2019	Ph.D. in Computer Science (Expected), University of California, Santa Barbara.
2015	M.A. in Applied Statistics, University of California, Santa Barbara.
2013	B.S. in Computer Science, University of California, Santa Barbara.
2013	B.S. in Financial Mathematics and Statistics, University of California, Santa Barbara.

Publications

Pourzanjani, A. A., Petzold, L. R. (2019). Implicit Hamiltonian Monte Carlo for Sampling Multiscale Distributions. *Submitted to Springer Journal of Statistics and Computing.*

Pourzanjani, A. A., Wu T., Bales B.B., Petzold, L. R. (2018, August). Relating Disparate Measures of Coagulapathy Using Unorthodox Data: A Hybrid Mechanistic-Statistical Approach. *Proceedings of StanCon 2018 (Helsinki).*

Pourzanjani, A. A., Jiang, R. M., Mitchell, B., Atzberger, P. J., Petzold, L. R. (2017). General Bayesian Inference over the Stiefel Manifold via the Givens Representation. *Under Review, The Journal of Bayesian Analysis.*

Pourzanjani, A. A., Bales, B.B., Petzold, L.R., Harrington, M. (2018). Diagnosing Alzheimerś the Bayesian way. *Procdeedings of StanCon 2018 (Asilomar)*

Pourzanjani, A., Jiang, R. M., Petzold, L. R. (2017, December). Improving the Identifiability of Neural Networks for Bayesian Inference. *Proceedings of Bayesian Deep Learning Workshop at Advances in Neural Information Processing Systems (NIPS). 2017.*.

Pourzanjani, Bales B.B., Harrington M. Petzold, L. R. (2017, December). A Statistical Model for Automatic Staging and Prediction of Alzheimer's Disease. *Poster session presented at the Annual Finch Alzheimer's Disease Symposium.*

Pourzanjani, A. A., Wu T., Jiang, R. M., Cohen, M., Petzold, L. R. (2017). Understanding Coagulopathy using Multi-view Data in the Presence of Sub-Cohorts: A Hierarchical Subspace Approach. *Proceedings of Machine Learning for Healthcare 2017, JMLR W&C Track Volume 68.*

Pourzanjani, A., Quisel, T., Foschini, L. (2016). Adherent use of digital health trackers is associated with weight loss. *PloS one*, 11(4), e0152504.

Pourzanjani, A., Stuck, D., Sontag, D., Foschini, L. (2015, December). Fully Bayesian Unsupervised Disease Progression Modeling. *Proceedings of Machine Learning for Healthcare Workshop at Advances in Neural Information Processing Systems (NIPS).*

Pourzanjani, A., Herzog, E. D., Petzold, L. R. (2015). On the inference of functional Circadian networks using Granger Causality. *PloS One*, 10(9), e0137540.

Abstract

Novel Computational Methods for Bayesian Hierarchical Modeling in the Biomedical Domain

by

Arya Alexander Pourzanjani

The recent growth in the availability of biomedical data promises to reshape healthcare by ushering in an era of personalized medicine where data can be used to diagnose and treat patients with pinpoint accuracy. Truly realizing this goal requires building statistical models that individually model patient variations such as age, sex, and genetic makeup, which leads to a combinatorial growth in the number of parameters in a model and noisy estimates. Fortunately, Bayesian hierarchical models, along with recent computational advances, provide a solution to this issue. By naturally embedding the hierarchical structure that many datasets exhibit into the model, these models allow for separate estimates that capture population-level variation and simultaneously avoid noise via regularization to a population mean.

In this thesis, we describe novel models and computational methods for Bayesian hierarchical modeling of biomedical data. We begin by describing our contributions to various areas of Bayesian modeling, along with the problems from our applied work that motivated these contributions.

Specifically, we describe our disease progression model, which hierarchically models patient disease trajectories. The model, which was motivated by our applied work on Alzheimer's Disease, utilizes I-splines to capture the characteristic monotonic shape of dementia disease trajectories, along with Dirichlet distributions over the coefficients of these I-splines to hierarchically model these trajectories. Next we describe our work on

using the Givens Representation of orthogonal matrices to infer models with orthogonal matrix parameters, such as factor models, in a general Bayesian framework. We describe the innovations in our method along with our motivating hierarchical example based on the analysis of protein biomarkers of coagulopathic trauma patients. Next, we describe a mechanistic model of coagulopathy that relates clotting assay data to protein concentrations, effectively providing a fast and convenient way for clinicians to understand key protein markers involved in clotting.

Next we transition specifically to Hamiltonian Monte Carlo (HMC) and elucidate the connection between multiscale posterior distributions and the efficiency of HMC. We describe the issue of numerical stability inside HMC and present our implicit HMC algorithm for efficiently sampling non-Gaussian posterior distributions.

Lastly, we provide a summary of our contributions along with ideas for future work.

Contents

Chapter 1

Introduction

1.1 Motivation

Bayesian modeling, and Bayesian hierarchical modeling in particular, is an immensely powerful tool for the modeling and analysis of modern datasets, both large and small. In short, hierarchical models allow the modeler to embed their knowledge of natural hierarchies within data to effectively capture the heterogeneity of the data, while simultaneously regularizing unstable estimates to a common, group mean [1]. In many datasets, a hierarchical structure that can be utilized is often readily apparent. As an example, within a dataset containing biomarker data for individual hospital patients, we may naturally group the observations first by the hospital at which it was collected. Within each hospital we may group our observations by the type of injury the patient sustained. Within each injury group we may group observations based on relevant individual characteristics such as age or sex (Figure 1.1). While we expect variation within each group, we simultaneously expect that patients within a group can be informative of one another. For example, we may expect that two patients of similar age who sustained similar injuries and are being treated at the same hospital may exhibit unique biomarker

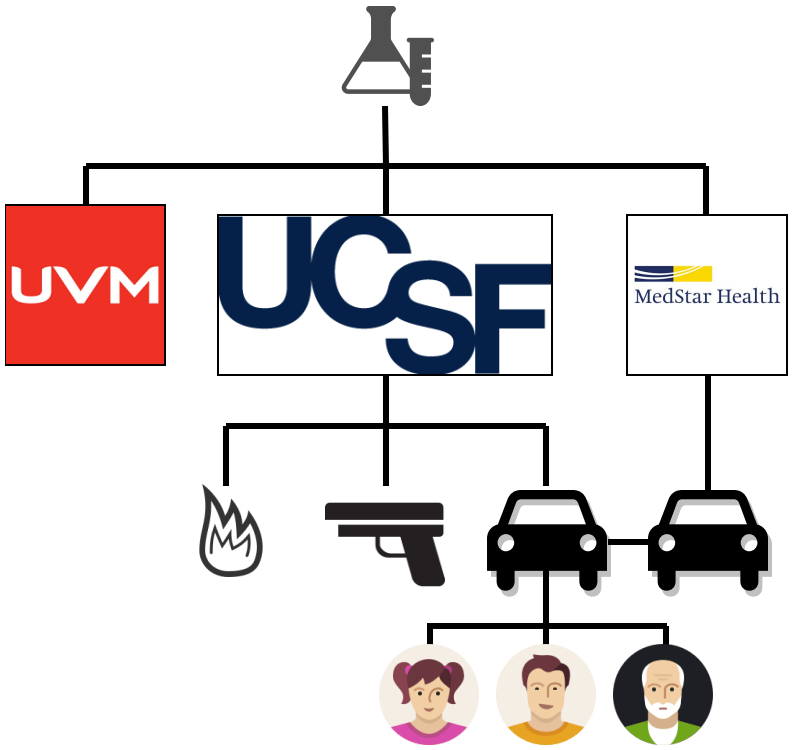

Figure 1.1: Natural hierarchies are readily apparent in many practical datasets, such as biomedical data which may be grouped by hospital, then injury type, and finally by patient type.

trajectories, however it would not be unreasonable for these observations to be informative of one another. Hierarchical modeling allows us to naturally encode this intuition within our models.

At the time of this writing, an explosion is occurring in the amount of data available in the medical domain. The price of genome sequencing has surpassed the pace of Moore's law, providing a wealth of individualized data to study every disease imaginable [2]. Large datasets with hierarchical structures, like the aforementioned examples, are becoming more and more available, and with them comes the promise of "personalized medicine", which many expect to immensely improve health outcomes and lives in the upcoming years [3, 4, 5]. The premise of personalized medicine is that with more data we will be able to capture individual patient characteristics, which will in turn allow us to provide

more targeted therapy. However, within this lies a contradiction, the solution of which hierarchical modeling provides. In particular, to truly personalize our decisions we must account for every unique characteristic of a patient e.g. their age, sex, genome, type and extent of injury etc. Unfortunately, once we group patients by all these relevant characteristics to conduct statistical inference, our once large dataset has been reduced to many small datasets which individually are difficult to learn from with such small sample sizes. Hierarchical modeling solves this challenge, by simultaneously grouping patients and treating them separately to a degree that the data can support.

At the same time that medical data has become more and more available, advances in the field of Bayesian computation have allowed for the modeling and inference of larger and larger models, including hierarchical models which were previously intractable to infer. In particular, the advent of the Hamiltonian Monte Carlo (HMC) sampling method of Duane et al. [6] and its subsequent introduction to the statistics community by Neal [7] has opened the flood gates for practical inference of high-dimensional models by practitioners. Unlike the Metropolis method [8] which was in wide use before HMC, HMC can scale to larger dimensions at an almost linear rate [9]. More recently, the No-U-Turn Sample (NUTS) extension of HMC by Hoffman and Gelman [10] and advances in automatic differentiation have led to practical software packages such as Stan that make the power of HMC available to wide audiences [11].

This work lies at the intersection of these recent advances in medical data availability and Bayesian computation, where I have conducted my research for the past few years. In particular, I describe novel models and computational techniques for inference that I have developed over the years in pursuit of my goal to better understand biology and medicine via data analysis.

1.2 Organization

In Chapters 2, 3, 4, and 5, I describe our advances in the respective areas of disease progression, factor modeling, and mechanistic modeling along with motivating examples from our applied work. In Chapter 6, I go over the relationship between Bayesian posteriors and the numerical solution of differential equations and I describe our Implicit HMC method for inference of multiscale posterior distributions. I close with a summary of our contributions along with future directions in Chapter 7.

Chapter 2

Disease Progression Models

Disease progression modeling is a rapidly growing area of data science that seeks to use patient data to quantify and forecast the severity of a disease [12, 13, 14]. This chapter summarizes our work titled "Diagnosing Alzheimer's the Bayesian Way" published in the Proceedings of StanCon, Asilomar (2018) [15]. We describe linear models of disease progression. Next we describe our nonlinear disease progression model that utilizes I-splines and hierarchical Dirichlet priors. Lastly, we describe the specific application of our model to Alzheimer's Disease.

2.1 Linear Models of Disease Progression

Several existing models of disease progression, particularly in Alzheimer's Disease, utilize monotonic functions to capture the continual worsening of biomarkers. Jedynak et al. posit that the deterioration of individual biomarkers is tied to a single latent state for each patient that increases monotonically over time [16]. This latent variable can thus be used as a disease progression score, representing how far in the disease an individual has progressed (Figure 2.1).

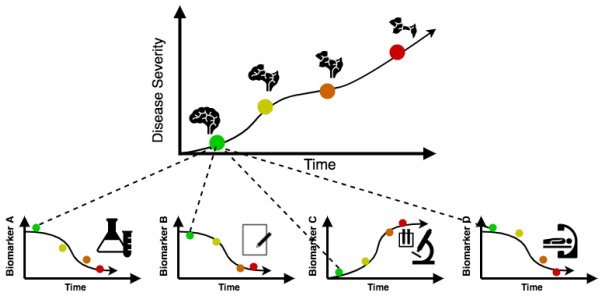

Figure 2.1: Diagram of a disease progression model with a single latent state representing disease severity.

While the model was originally presented in a frequentist least-squares framework, if we let $y_{ij}(t)$ denote the jth biomarker for the ith patient at age t, the original model can be translated to the probabilistic generative model

$$s_i(t) = \alpha_i t + \beta_i \tag{2.1}$$

$$y_{ij}(t) \sim \mathcal{N}(f(s_i(t) \,|\, a_j, b_j, c_j, d_j), \sigma_j), \tag{2.2}$$

where σ_j represents the measurement variability for the jth biomarker and $f(\cdot|a_j, b_j, c_j, d_j)$ represents a biomarker-specific 4-parameter sigmoid-curve for the jth biomarker

$$f(s \,|\, a_j, b_j, c_j, d_j) = \frac{a_j}{1 + e^{-b_j s - c_j}} + d_j. \tag{2.3}$$

The parameters c_j and b_j are of particular importance because they indicate when a biomarker starts deteriorating and how fast it deteriorates once it does. Figures 2.2 and 2.3 show posterior distributions of these parameters on the Bayesian version of this model fit to real dementia data using the priors

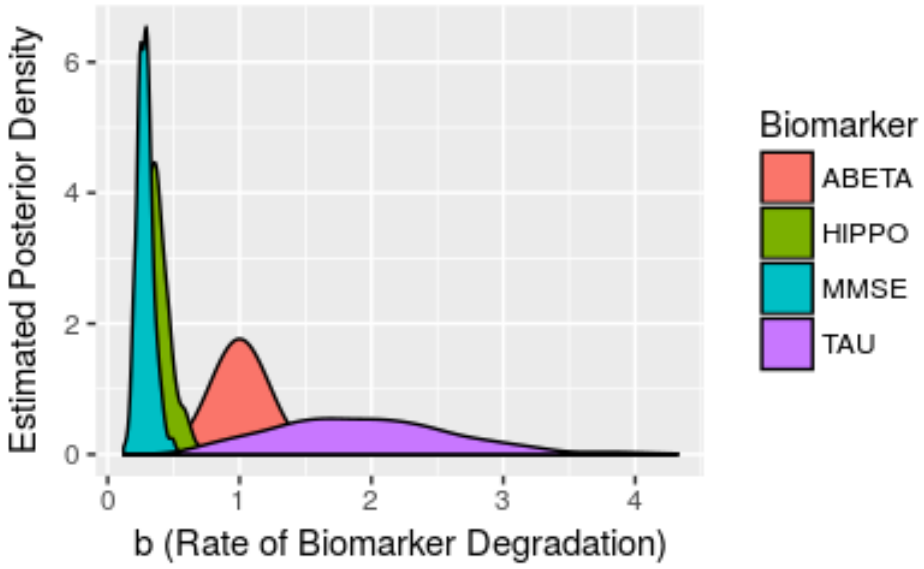

Figure 2.2: Posterior distribution of b_j parameters of the linear disease progression model fit to Alzheimer's data.

$$\gamma \sim \text{HalfNormal}(0, 10) \tag{2.4}$$

$$\alpha_i \sim \text{HalfNormal}(0, \gamma). \tag{2.5}$$

The values of the b_j and c_j parameters illustrate the relative speed of progression of the four biomarkers in this dementia disease progression model. The form of the degradation of the biomarkers takes the form of a logistic (or S-curve) because in practice, biomarkers often seem to get worse slowly at first, then rapidly, and eventually hit a saturation point. We note that one of the biomarkers has to have fixed values for b and c, to ensure identifiability of the parameters.

In this model, $s_i(t)$ is a monotonically increasing transformation of age that represents a continuous disease progression score for the ith individual. The individual specific parameters α_i and β_i determine the rate of progression and relative onset of disease

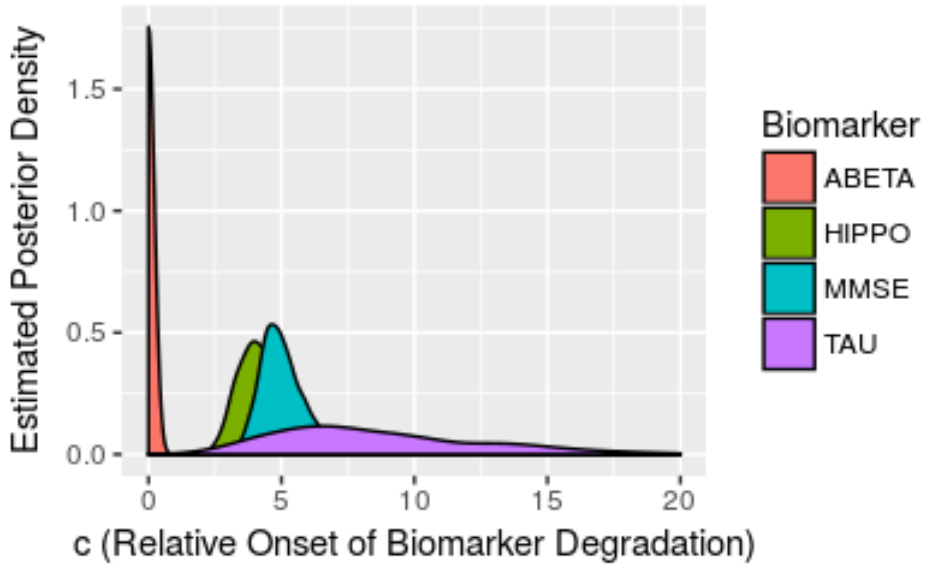

Figure 2.3: Posterior distribution of c_j parameters of the linear disease progression model fit to Alzheimer's data.

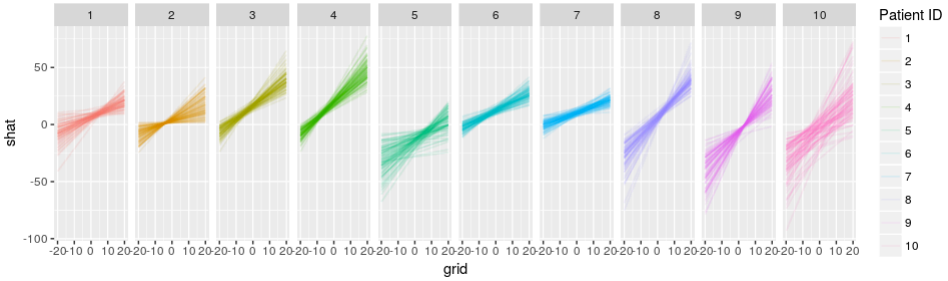

Figure 2.4: Posterior distribution of patient disease trajectories from the linear disease progression model fit to Alzheimer's data.

respectively for the ith individual. To ensure the progression variable $s_i(t)$ is monotonic in time, α_i is constrained to be positive. Figure 2.4 shows posterior draws of what these curves typically look like, while Figure 2.5, shows associated biomarker trajectories.

We note that while each patient has a unique set of parameters that describe their personal latent disease progression score, the parameters that describe individual biomarker evolution in the sigmoid function are constant across the population. This implies that the degradation of the biomarkers occurs in the same relative order and relative rate in all patients, an assumption that would have to be properly verified by examining the

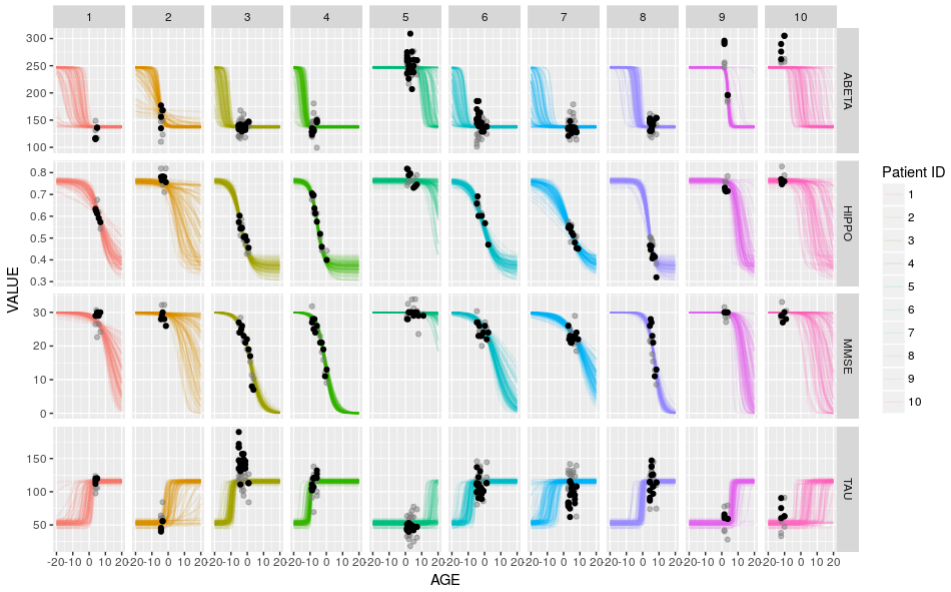

Figure 2.5: Posterior distribution of patient biomarker trajectories from the linear disease progression model fit to Alzheimer's data.

relative rates of deterioration in individuals.

2.2 Nonlinear Modeling of Disease Progression Using I-Splines

While a linear fit can do a reasonable job of capturing each individual's progression of biomarkers over time, it leaves much to be desired. The model assumes that the rate of progression of the disease is constant, i.e. that patients' disease status continuously progress at the same rate. In reality, it can often be the case that there are plateaus of progression, such as in vascular dementia, where progression can slow down then later pick back up, due either to endogenous or exogenous circumstances. To more flexibly represent disease progression curves requires a way to flexibly model monotonic functions of age. Lorenzi et al. [17] use a monotonic Gaussian Process to model how each biomarker deteriorates over time in the population, and use ordinary Gaussian Processes for each

individual, to describe random deviations from this general progression at the individual level. To achieve monotonicity in their Gaussian Processes, they use the method proposed in [18], and fit their model using Expectation Propagation (EP)[17]. To achieve a full Bayesian implementation that can be fit using standard inference methods, we instead turn to I-Splines.

2.2.1 I-Splines

I-Splines are a flexible and adjustable set of monotone basis functions used to model monotonic functions [19]. By taking a linear combination of these functions, and constraining the coefficients to be positive, one can flexibly model a wide class of monotone functions. Similarly, one can model increasing functions with a range from 0 to 1, by taking a linear combination of the I-Splines where the coefficients are constrained to sum to one.

To our knowledge, the only other known method for flexibly modeling monotone functions are monotone Gaussian Processes [18], which were used by [17]. This method relies on constraining the Gaussian Process to have a positive derivative at a set number of points, which then usually forces the function to monotone. While attractive, the method may not return true monotone functions and it is not clear how to select the points where the derivative should be positive.

Given a domain, and a set of nodes $t_1, \cdots, t_D$ on that domain, the I-Spline basis functions are obtained by integrating the piece-wise-defined M-Splines, themselves a set of spline functions defined on the same domain using the same nodes. M-Splines are defined recursively. The order one M-Spline functions are piece-wise constant functions. The ith M-Spline of order one is defined as

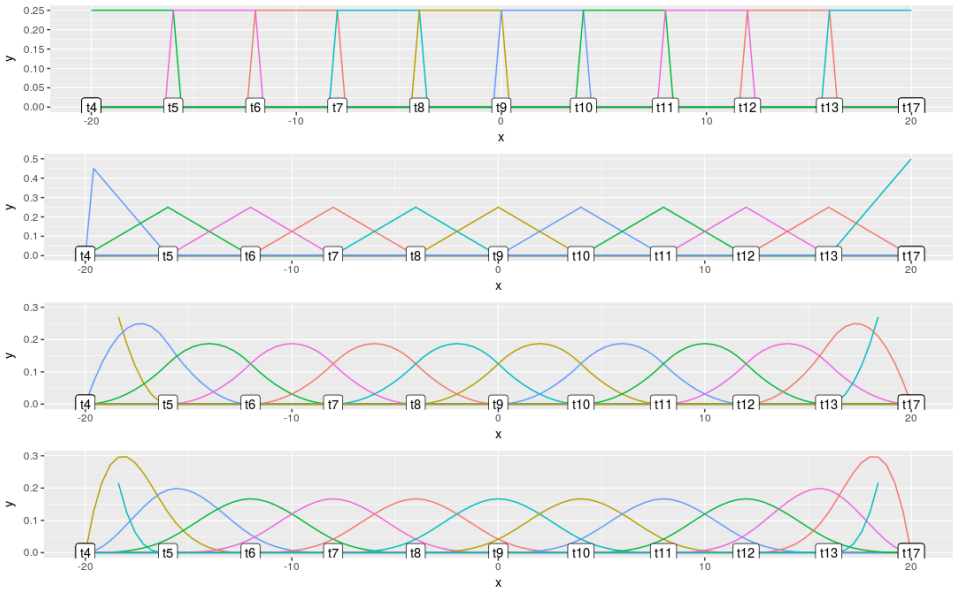

Figure 2.6: Various M-Spline bases.

$$M_{i1}(x) := \begin{cases} \frac{1}{(t_{i+1}-t_i)}, & t_i \leq x < t_{i+1} \\ 0, & \text{otherwise} \end{cases} \tag{2.6}$$

while the ith M-Spline of order k is defined recursively as

$$M_{ik}(x) := \left(\frac{k}{k-1}\right) \frac{(x-t_i)M_{i,k-1}(x) + (t_{i+k}-x)M_{i+1,k-1}(x)}{t_{i+k}-t_i}. \tag{2.7}$$

One can show that the M-Spline basis functions each integrate to one over the specified domain and are positive. Figure 2.6 shows the M-Spline basis functions for orders $k = 1, 2, 3, 4$ using custom code written to generate M-Splines in R.

Because M-Splines integrate to one over the specified domain and are positive, their integral will be a set of functions that monotonically increase from 0 to 1 over the domain. Since the M-Splines are piece-wise polynomials, this integral is easy to compute. Figure

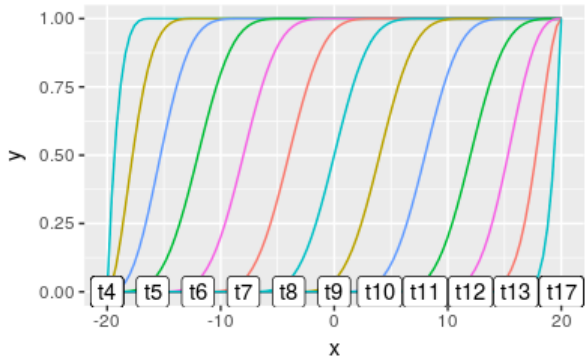

Figure 2.7: I-spline basis.

2.7 illustrates an I-spline basis over the domain $[-20, 20]$.

Positive linear combinations of I-Splines will be monotone, while convex linear combinations will be monotone and range from 0 to 1. We use this idea to expand the linear latent progression model to a more flexible latent progression using I-Splines. Specifically, we replace the latent disease progression function, $s_i(t)$, with a positive linear combination of I-Splines, rather than with a linear function.

2.3 Application to Alzheimer's Disease and Dementia

The progression of Alzheimer's Disease (AD) is characterized by the gradual deterioration of biomarkers and eventual loss of basic memory and decision-making functions, which makes the I-spline model ideal. Using these biomarker values and other tests to estimate how far an individual has progressed in the disease is valuable in diagnosis as well as in assessing the efficacy of interventions. Additionally, prediction of how the individual will continue to progress is critical in decision making. While it is known that AD only gets worse over time, it is believed that patients with the disease progress at different rates and at different stages of their lives. There is no standard path of progres-

sion for people with the disease, which makes estimation of disease severity and future progression difficult. In addition to estimating these paths for an individual given their measurements, it is of clinical and biological significance to be able to understand the order in which certain biomarkers begin to deteriorate, and what their distribution might look like for various stages of the disease.

Currently, staging of AD is accomplished by the thorough review of a patient by a panel of expert doctors. While this diagnosis serves as a gold-standard, the diagnosis process is cumbersome, prone to subjectivity, and typically only includes three discrete levels, which does not represent the continuously progressing nature of AD. These three stages are, in order of severity: Normal (NL), Mild Cognitive Impairment (MCI), and Dementia.

2.3.1 Data

For our analysis we used a pre-cleaned table of biomarker data from the publicly available Alzheimer's Disease Neuroimaging Initiative (ADNI) dataset. The dataset contains various biomarker measurements for 1,737 patients, as well as their age at the time of measurement. The biomarkers measured include the following:

- ABETA: Concentration of the amyloid-beta protein in cerebrospinal fluid (CSF), measured in pg/ml. In healthy individuals this harmful protein is actively cleared from the brain in to the CSF. In individuals with AD, the amyloid-beta protein concentrates in the brain to form harmful plaques. Low levels in the CSF indicate the protein is not being cleared from the brain, and is thus an indication of AD.

- HIPPO: Volume of the individual's hippocampal brain region as measured by MRI and normalized to their baseline brain volume. In AD the hippocampus is known

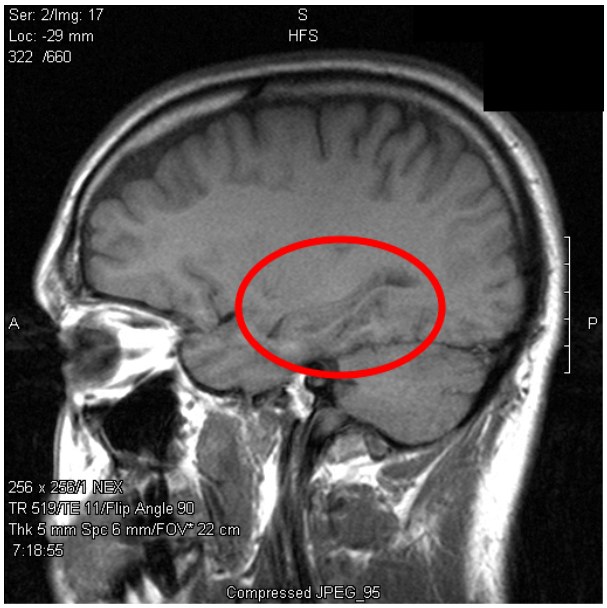

Figure 2.8: MRI imaging of a human brain.

to shrink in size as the disease progresses. Volume is typically estimated using software applied to images like Figure 2.8.

- MMSE: Mini-Mental State Examination (MMSE) test. A 30-point questionnaire administered by psychologists to assess working memory and brain function in an individual. Patients with AD show lower scores in MMSE, but unfortunately lowered MMSE scores typically only show once the disease has progressed significantly.

- TAU: Concentration of tau protein in the CSF, measured in pg/ml. Tau is a protein that shows up as a byproduct of dead neurons. High tau in the CSF indicates an abundance of dying neurons in the brain.

Population histograms of these biomarkers during the three stages of the disease reveal a monotonic pattern in the four biomarkers we are considering (Figure 2.9). For example, ABETA appears to be a biomodal distribution where the lower mode becomes more common as the disease worsens. The distribution of HIPPO appears to shift to

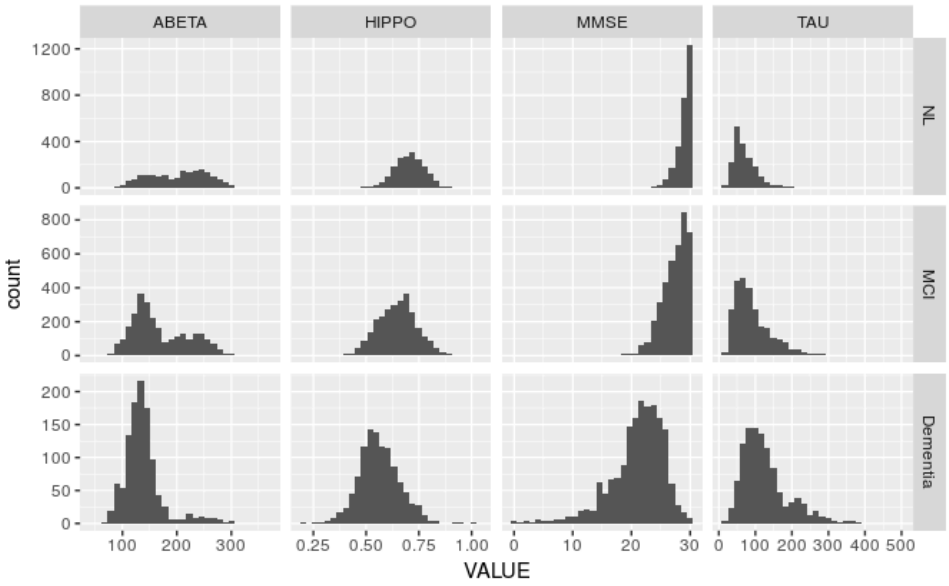

Figure 2.9: Histograms of patient biomarker data over the three stages of Alzheimer's Disease.

the left with increased worsening of AD. These observations lead us to believe that these biomarkers can be modeled as coming from a single continuous latent variable that we can interpret as being a disease progression score.

Figure 2.10 shows biomarker trajectories for ten different patients over time.

2.3.2 Model

In addition to replacing the linear disease progression curves with I-splines, we let the measurement variability of the ABETA measurements be different for each individual, and place a hierarchical prior over these parameters. The biomarker plots suggest that the random variation in these measurements may differ in magnitude from person to person.

2.3.3 Results

Once fit using Stan, posterior draws of individual disease progression curves imme-

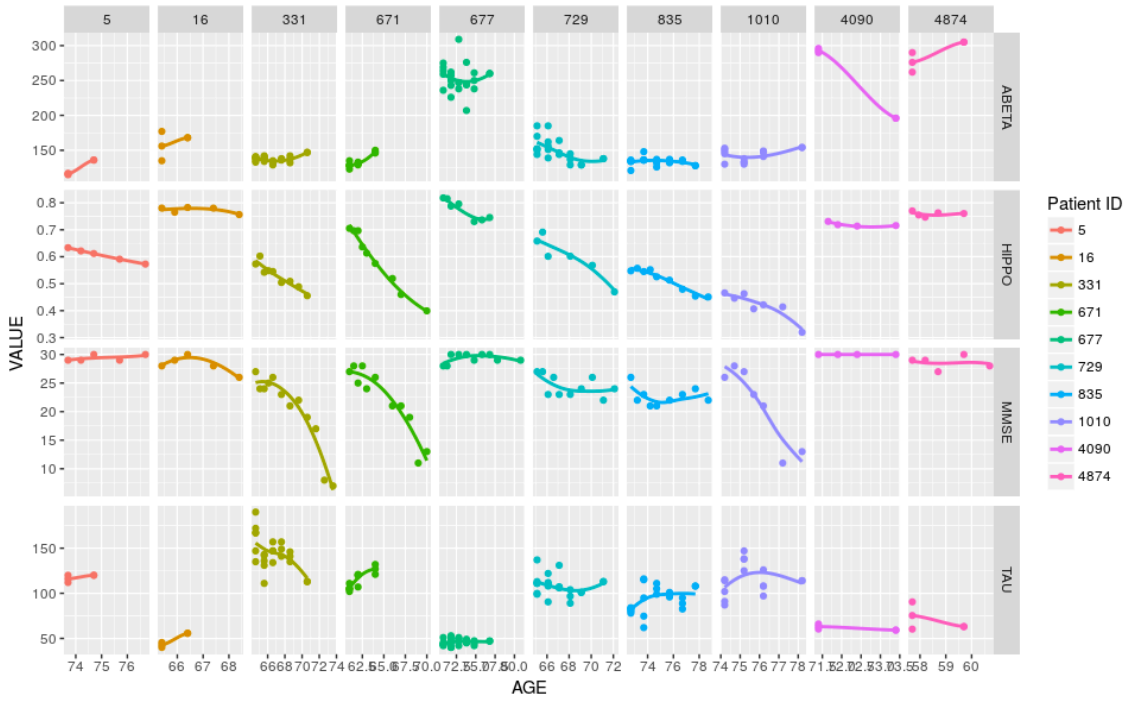

Figure 2.10: Longitudinal trajectories of biomarker data for ten separate patients.

diately reveal possible non-linearities in disease progression (Figure 2.11). While some patients, such as patient 4, still maintain approximately linear disease progression, others such as patient 5 or 9 display slow initial progression then a sharp worsening in the disease. Others, such as patient 7, seem to worsen then have a flattened prior where progression is slow, followed by another quickly progressing epoch. These more flexible disease progression curves also manifest in seemingly more accurate biomarker progression curves (Figure 2.12), leaving us with an overall more trustworthy and robust model.

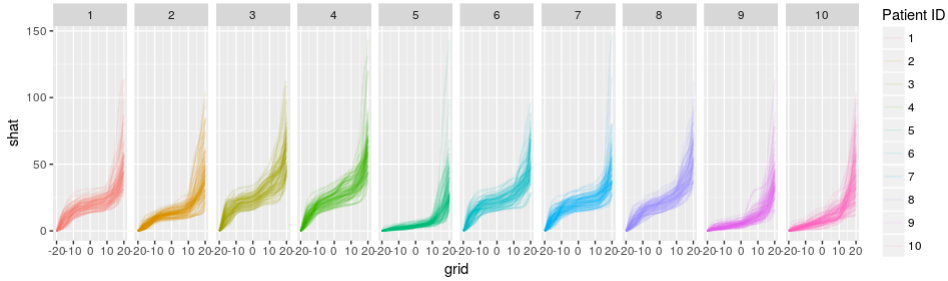

Figure 2.11: Posterior draws of nonlinear trajectories from the I-spline model.

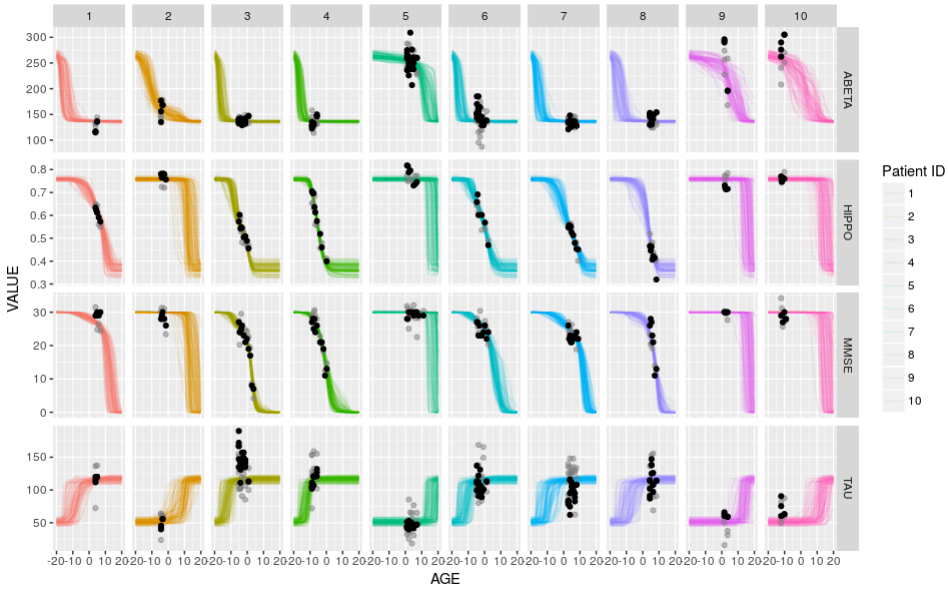

Figure 2.12: Posterior draws of nonlinear biomarker trajectories from the I-spline model.

Chapter 3

Factor Models and other Models with Orthogonal Matrix Parameters

In this chapter we present our work titled "General Bayesian Inference over the Stiefel Manifold via the Givens Transform" [20] where we introduce an approach based on the Givens representation that allows for a routine, reliable, and flexible way to infer Bayesian models with orthogonal matrix parameters. This class of models most notably includes models from multivariate statistics such as factor models and probabilistic principal component analysis (PPCA). Our approach overcomes several of the practical barriers to using the Givens representation in a general Bayesian inference framework. In particular, we show how to inexpensively compute the change-of-measure term necessary for transformations of random variables. We also show how to overcome specific topological pathologies that arise when representing circular random variables in an unconstrained space. In addition, we discuss how the alternative parameterization can be used to define new distributions over orthogonal matrices as well as to constrain parameter space to eliminate superfluous posterior modes in models such as PPCA. While previous approaches to this inference problem involved specialized updates to the orthogonal matrix

parameters, our approach lets us represent these constrained parameters in an unconstrained form. Unlike previous approaches, this allows for the inference of models with orthogonal matrix parameters using any modern inference algorithm, including those available in modern Bayesian modeling frameworks such as Stan, Edward, or PyMC3. We illustrate with examples how our approach can be used in practice in Stan to infer models with orthogonal matrix parameters, and we compare to existing methods.

3.1 Introduction

Statistical models parameterized in terms of orthogonal matrices are ubiquitous, particularly in the treatment of multivariate data. This class of models includes certain multivariate time series models [21], factor models [22], and a swath of recently developed probabilistic dimensionality reduction models such as Probabilistic PCA (PPCA), Exponential Family PPCA (BXPCA), mixture of PPCA [23], and Canonical Correlation Analysis (CCA) [24, Chapt. 12.5]. These sorts of models have not only enjoyed extensive use in fields such as psychology [25], but are also gaining traction in diverse applications including biology [26], finance [27], materials science [28], and robotics [29].

Despite their ubiquity, their remains no quick, routine, and flexible options for fitting models with orthogonal matrix parameters. Existing methods for inferring these models are either insufficiently general or too complicated to implement and tune in isolation. Modern probabilistic programming frameworks, such as Stan, Edward, and PyMC3 [30, 31, 32], try to abstract their users away from the details of inference and implementation, but none offer support for orthogonal matrix parameters. The reason is that rather than using a specialized inference algorithm for orthogonal matrices, which existing approaches do, these software frameworks typically handle constrained parameters such as orthogonal matrices by transforming them to an unconstrained space [30, 33]. For example, if a

model contains a parameter $\sigma > 0$ that is constrained to be positive, these frameworks typically take the log of this parameter and conduct inference over $\tilde{\sigma} = \log \sigma$, which is unconstrained.

An unconstrained parameterization of orthogonal matrices would allow for general Bayesian inference in any software framework without having to change its inner-workings, but because of the complexities in dealing with the space of orthogonal matrices, otherwise known as the Stiefel manifold, several challenges remain in the way of this approach. While many parameterizations of orthogonal matrices exist [34, 35], only smooth representations, such as the Givens representation, can be practically considered, as inference methods such as Hamiltonian Monte Carlo (HMC) typically require continuous and differentiable likelihoods. Furthermore, any such transformation of a random variable typically requires computing a change-of-measure adjustment term that is often unknown or expensive to compute. A further complication is that the Stiefel manifold has a fundamentally different topology than Euclidean space, which can lead to biased inference if particular care is not taken in implementation. Lastly, while not strictly necessary, any representation would ideally have an intuitive interpretation that would allow practitioners to work with and even define useful distributions in terms of the new representation.

We introduce a general approach to the posterior inference of statistical models with orthogonal matrix parameters based on the Givens representation of orthogonal matrices. We address several practical implementation issues such as computation of the change-of-measure adjustment, as well as proper handling of transformed coordinates to ensure unbiased samples. Our approach enables the application of any general inference algorithm to models containing orthogonal matrix parameters, allowing inference of these models by any commonly available inference algorithm such as HMC [7], the No-U-Turn Sampler (NUTS) [10], Automatic Differentiation Variational Inference (ADVI) [33] or

Black Box Variational Inference [36]. Unlike existing approaches, our approach is easy to implement and does not require any specialized inference algorithms or modifications to existing algorithms or software. This allows users to rapidly build and prototype complex probabilistic models with orthogonal matrix parameters in any common software framework such as Stan, Edward, or PyMC3 without the worry of messy implementation details.

In Section 3.2 we discuss existing methods for Bayesian inference over the Stiefel manifold and the difficulty in implementing these methods in a general Bayesian inference framework. In Section 3.3 we describe the Givens representation by first introducing the Givens reduction algorithm and then connecting it to a geometric perspective of the Stiefel manifold, providing an approachable intuition to the transform. We go on to describe practical solutions for using the Givens representation in a general Bayesian inference setting in Section 3.4. In Section 3.5 we illustrate with statistical examples the use of the Givens representation and how it compares to existing methods in practice. Lastly, we conclude with a brief discussion in Section 3.6 where we summarize our contributions.

3.2 Related Work

Hoff et al. [37] introduces a Gibbs sampling approach to update unknown orthogonal matrix parameters from a collection of known conditional distributions. Unfortunately, this requires that the conditional distribution of the orthogonal matrix parameter given other model parameters belongs to a known parametric distribution that is easy to sample. In practice, this limits the approach to a specific class of models.

More general HMC methods have been devised, but their use of specialized update rules makes them difficult to implement and tune in practice. In particular, these methods

infer orthogonal matrix parameters by using different HMC update rules for constrained and unconstrained parameters. This separation of constrained and unconstrained parameters requires additional book-keeping to know which update rules to use on which parameter. Unfortunately, many probabilistic programming languages do not keep track of this as they treat parameters agnostically by transforming to an unconstrained space. The specialization of these methods to HMC also makes them difficult to generalize to other inference algorithms based on VI or optimization which an unconstrained parameterization approach would have no trouble with.

Specifically, Brubaker et al. [38] proposed a modified HMC, which uses a different update rule for constrained parameters based on the symplectic SHAKE integrator [39]. For unconstrained parameters, the method uses a standard Leapfrog update rule. For constrained parameters, the method first takes a Leapfrog step which usually moves the parameter to a value that does not obey constraints. The method then uses Newton's method to "project" the parameter value back down to the manifold where the desired constraints are satisfied.

Byrne and Girolami [40] as well as Holbrook et al. [41] also utilize a separate HMC update rule to deal with constrained parameters. Specifically, they utilize analytic results and the matrix exponential to update the parameters in such a way that guarantees constraints are still satisfied in the embedded matrix coordinates. More precisely, they use the fact that analytic solutions for the geodesic equations on the Stiefel manifold in the embedded coordinates are known. This gives rise to their Embedded Manifold HMC (EMHMC) algorithm. Like the method of [38], the use of separate update rules in EMHMC makes the algorithm difficult to implement in more general settings.

3.3 The Givens Representation of Orthogonal Matrices

We motivate and introduce the Givens representation by first describing the related Givens reduction algorithm of numerical analysis and then tying this to the geometric aspects of the Stiefel manifold.

3.3.1 Givens Rotations and Reductions

Given any $n \times p$ matrix, A, the Givens reduction algorithm is a numerical algorithm for finding the QR-factorization of A, i.e. an $n \times p$ orthogonal matrix Q and an upper-triangular $p \times p$ matrix R such that $A = QR$. The algorithm works by successively applying a series of Givens rotations so as to "zero-out" elements of A below the diagonal. These Givens rotations are simply $n \times n$ matrices, $R_{ij}(\theta_{ij})$, that take the form of an identity matrix except for the (i,i) and (j,j) positions which are replaced by $\cos\theta_{ij}$ and the (i,j) and (j,i) positions which are replaced by $-\sin\theta_{ij}$ and $\sin\theta_{ij}$ respectively.

When applied to a vector, $R_{ij}(\theta_{ij})$ has the effect of rotating the vector counter-clockwise in the (i,j)-plane, while leaving other elements fixed. Intuitively, its inverse, $R_{ij}^{-1}(\theta_{ij})$, has the same effect, but clockwise. Thus one can "zero-out" the jth element, u_j, of a vector u, by first using the arctan function to find the angle θ_{ij} formed in the (i,j)-plane by u_i and u_j, and then multiplying by the matrix $R_{ij}^{-1}(\theta_{ij})$ (Figure 3.1, inset).

In the Givens reduction algorithm, these rotation matrices are applied one-by-one to A in this way to eliminate all elements below the diagonal. First, all elements in the first column below the first row are eliminated by successively applying the rotation matrices $R_{12}^{-1}(\theta_{12}), R_{13}^{-1}(\theta_{13}), \cdots, R_{1n}^{-1}(\theta_{1n})$ (Figure 3.2). Because multiplication by $R_{ij}(\theta_{ij})$ only affects elements i and j of a vector, once the jth element is zeroed out, the subsequent

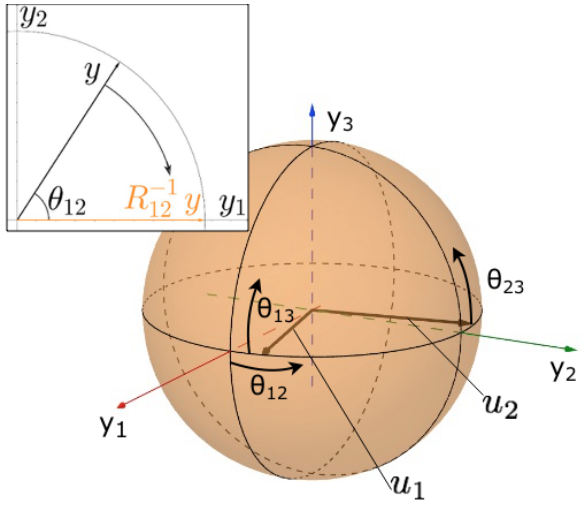

Figure 3.1: (Inset) Givens rotations can be used to rotate a vector so as to eliminate its component in a certain direction. (Main Figure) A p-frame on the Stiefel manifold can be visualized as a set of rigidly connected orthogonal basis vectors, u_1 and u_2, shown here in black. One can move about the Stiefel manifold and describe any p-frame by simultaneously applying rotations matrices of a prescribed angle to these basis vectors. Applying the rotation matrix $R_{12}(\theta_{12})$ corresponds to rotating the two basis vectors toegher in the (1,2)-plane, which by our convention is the (x, y)-plane. Similarly, simultaneously apply $R_{13}(\theta_{13})$ corresponds to a rotation of the 2-frame in the $(1, 3)$ or (x, z)-plane, while $R_{23}(\theta_{23})$ corresponds to rotating u_2 about u_1.

rotations, $R_{13}^{-1}(\theta_{13}), \cdots, R_{1n}^{-1}(\theta_{1n})$, will leave the initial changes unaffected. Similarly, once the first column of A is zeroed out below the first element, the subsequent rotations, which do not involve the first element will leave the column unaffected. The rotations $R_{23}^{-1}(\theta_{23}), \cdots, R_{2n}^{-1}(\theta_{2n})$ can thus be applied to zero out the second column, while leaving the first column unaffected. This results in the upper triangular matrix

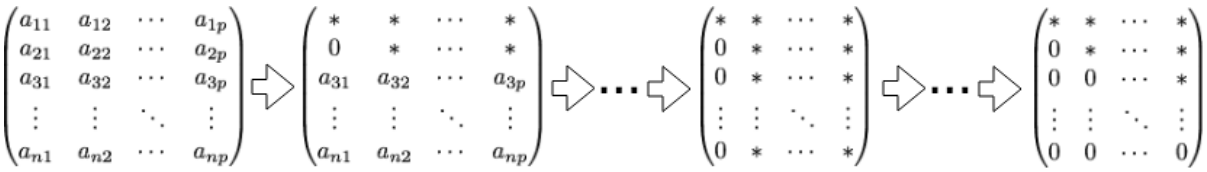

Figure 3.2: The Givens reduction eliminates lower diagonal elements of an $n \times p$ matrix one column at a time. Because each rotation, $R_{ij}(\theta_{ij})$, only affects rows i and j, previously zeroed out elements do not change.

$$R_* := \underbrace{R_{pn}^{-1}(\theta_{pn}) \cdots R_{p,p+1}^{-1}(\theta_{p,p+1}) \cdots R_{2n}^{-1}(\theta_{2n}) \cdots R_{23}^{-1}(\theta_{23}) \cdots R_{1n}^{-1}(\theta_{1n}) \cdots R_{12}^{-1}(\theta_{12})}_{Q_*^{-1}} A. \tag{3.1}$$

Crucially, the product of rotations, which we call Q_*^{-1}, is orthogonal since it is simply the product of rotation matrices which are themselves orthogonal. Thus its inverse can be applied to both sides of Equation 3.1 to obtain

$$Q_* R_* = A. \tag{3.2}$$

The familiar QR form can be obtained by setting Q equal to the first p columns of Q_* and setting R equal to the first p rows of R_*. The Givens reduction is summarized in Algorithm 1.

Input: A
Result: Q, R
$Q_*^{-1} = I$ $R_* = A$
for *i in 1:p* **do**
  for *j in (i+1):n* **do**
    $\theta_{ij} = \arctan(Y[j,i]/Y[i,i])$
    $Q_*^{-1} = R_{ij}^{-1}(\theta_{ij}) Q_*^{-1}$
    $R_* = R_{ij}^{-1}(\theta_{ij}) R_*$
  end
end
return $Q_*[, 1:p], R_*[1:p, 1:p]$

Algorithm 1: Psuedo-code for the Givens reduction algorithm for obtaining the QR factorization of a matrix A.

3.3.2 The Geometry of Orthogonal Matrices

The Stiefel manifold, $V_{p,n}$, consists of p-frames: ordered sets of p n-dimensional unit-length vectors, where $p \leq n$. p-frames naturally correspond to $n \times p$ orthogonal matrices

which can be used to define the Stiefel manifold succinctly as

$$V_{p,n} := \{Y \in \mathbb{R}^{n\times p} : Y^T Y = I\}. \tag{3.3}$$

Geometrically, an element of the Stiefel manifold can be pictured as a set of orthogonal, unit-length vectors that are rigidly connected to one another. A simple case is $V_{1,3}$, which consists of a single vector, u_1, on the unit sphere. This single vector can be represented by two polar coordinates that we naturally think of as longitude and latitude, but can also be thought of simply as subsequent rotations of the standard basis vector $e_1 := (1,0,0)^T$ in the (x,y) and (x,z) planes, which we refer to as the $(1,2)$ and $(1,3)$ planes for generality. In mathematical terms, u_1 can be represented as $u_1 = R_{12}(\theta_{12})R_{13}(\theta_{13})e_1$ (Figure 3.1).

Continuing without geometric interpretation, $V_{2,3}$ can be pictured as a vector in $V_{1,3}$ that has a second orthogonal vector, u_2, that is rigidly attached to it as it moves about the unit sphere. Because this second vector is constrained to be orthogonal to the first, its position can be described by a single rotation about the first vector. Thus elements of $V_{2,3}$ can be represented by three angles: two angles, θ_{12} and θ_{13}, that represent how much to rotate the first vector, and a third angle, θ_{23} that controls how much the second vector is rotated about the first (Figure 3.1). Mathematically this can be represented as the 3×2 orthogonal matrix $R_{12}(\theta_{12})R_{13}(\theta_{13})R_{23}(\theta_{23})(e_1, e_2)$.

Although elements of the Stiefel manifold can be represented by $n \times p$ matrices, their inherent dimension is less than np because of the constraints that the matrices must satisfy. The first column must satisfy a single constraint: the unit-length constraint. The second column must satisfy two constraints: not only must it be unit length, but it must also be orthogonal to the first column. The third column must additionally be orthogonal to the second column, giving it a total of three constraints. Continuing in

this way reveals the inherent dimensionality of the Stiefel manifold to be

$$d := np - 1 - 2 - \cdots p = np - \frac{p(p+1)}{2}. \tag{3.4}$$

3.3.3 Obtaining the Givens Representation

The Givens reduction applied to an orthogonal matrix gives rise to a representation of the Stiefel manifold that generalizes the intuitive geometric interpretation described above. When applied to an $n \times p$ orthogonal matrix Y, the Givens reduction yields

$$R_{pn}^{-1}(\theta_{pn}) \cdots R_{p,p+1}^{-1}(\theta_{p,p+1}) \cdots R_{2n}^{-1}(\theta_{2n}) \cdots R_{23}^{-1}(\theta_{23}) \cdots R_{1n}^{-1}(\theta_{1n}) \cdots R_{12}^{-1}(\theta_{12}) Y = I_{n,p} \tag{3.5}$$

where $I_{p,n}$ is defined to be the first p columns of the $n \times n$ identity matrix, i.e. the matrix consisting of the first p standard basis vectors $e_1, \cdots, e_p$. The first $n-1$ rotations transform the first column into e_1, since it zeros out all elements below the first and the orthogonal rotations do not affect the length of the vector which by hypothesis is unit length. Similarly, the next $n-2$ rotations will leave the length of the second column and its orthogonality to the first column intact because again, the rotation matrices are orthogonal. Because the second column must be zero below its second element it must be e_2. Continuing in this way explains the relationship in Equation 3.5.

Because Y was taken to be an arbitrary orthogonal matrix, then it is clear from Equation 3.5 that any orthogonal matrix Y can be factored as

$$Y = R_{12}(\theta_{12}) \cdots R_{1n}(\theta_{1n}) \cdots R_{23}(\theta_{23}) \cdots R_{2n}(\theta_{2n}) \cdots R_{p,p+1}(\theta_{p,p+1}) \cdots R_{pn}(\theta_{pn}) I_{n,p}. \tag{3.6}$$

Defining $\Theta := (\theta_{12} \cdots \theta_{1n} \cdots \theta_{23} \cdots \theta_{2n} \theta_{p,p+1} \cdots \theta_{pn})$ we can consider any orthogonal matrix as a function, $Y(\Theta)$, of these angles, effectively parameterizing the Stiefel manifold and yielding the Givens representation. The Givens representation is a smooth representation with respect to the angles Θ [35], and lines up with our geometric insight discussed in the previous subsection.

3.4 The Givens Representation for Bayesian Inference of Orthogonal Matrix Parameters

Practical use of the Givens representation in a general Bayesian inference framework involves solving several practical challenges. In addition to the standard change of measure term required in any transformation of a random variable, care must be taken to address certain pathological cases of the Givens representation that occur due to the different topologies of the Stiefel manifold and Euclidean space. We further describe these challenges and explain how we overcome them in practice. We also briefly remark on how the Givens representation can be leveraged in practice to solve issues with identifiability, and define new and useful distributions over the Stiefel manifold. We conclude the section by describing how the computation of the Givens representation scales in theory, particularly in comparison to EMHMC.

3.4.1 Transformation of Measure Under the Givens Representation

As is usual in any transformation of random variables, care must be taken to include a Jacobian determinant term in the transformed density to account for a change of measure under the transformation. For a posterior density over orthogonal matrices that takes

the form $p_Y(Y)$, the proper density over the transformed random variable, $\Theta(Y)$, takes the form $p_\Theta(\theta) = p_Y(Y(\Theta))|J_{Y(\Theta)}(\Theta)|$ [42]. Intuitively, this extra Jacobian determinant term accounts for how probability measures are distorted by the transformation (Figure 3.3). Unfortunately, the Givens representation, $Y(\Theta)$, is a map from a space of dimension $d := np - p(p+1)/2$ to a space of dimension np. Hence the determinant is non-square and thus undefined.

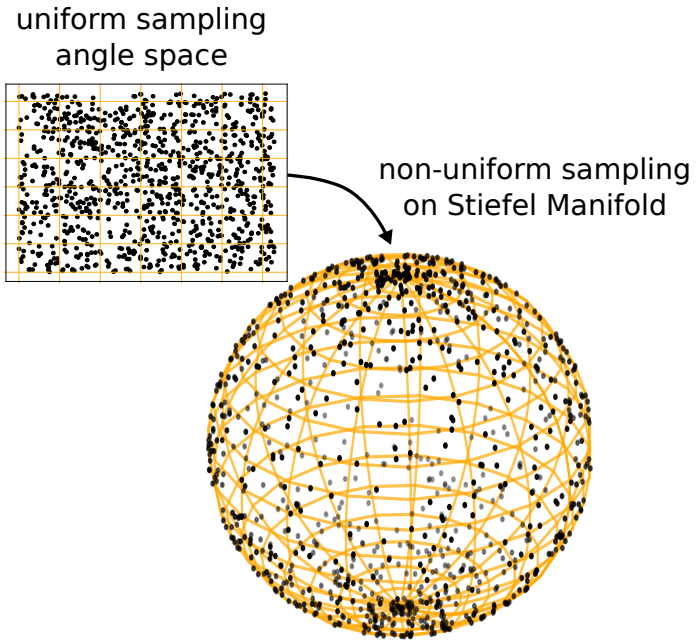

Figure 3.3: Uniform sampling in the Givens representation coordinates does not necessarily lead to uniform sampling over the Stiefel manifold without the proper measure adjustment term. Under the mapping, regions near the pole are shrunk to regions on the sphere with little area, as opposed to regions near to the equator which the transform maps to much larger areas on the sphere. Intuitively, the change-of-measure term quantifies this proportion of shrinkage in area.

To compute the change of measure term analogous to the Jacobian determinant, one must appeal to the algebra of differential forms. We denote the product of $n \times n$ rotation

matrices in the Givens representation by G, i.e.

$$G := R_{12}(\theta_{12}) \cdots R_{1n}(\theta_{1n}) \cdots R_{23}(\theta_{23}) \cdots R_{pn}(\theta_{pn}) \cdots R_{p,p+1}(\theta_{p,p+1}) \cdots R_{pn}(\theta_{pn}), \quad (3.7)$$

and its jth column by G_j. The paper [43] shows that the proper measure form for a signed surface element of $V_{p,n}$ is the differential form

$$\bigwedge_{i=1}^{p} \bigwedge_{j=i+1}^{n} G_j^T \, dY_i. \quad (3.8)$$

Letting $J_{Y_i(\Theta)}(\Theta)$ be the Jacobian of the ith column of Y with respect to the angle coordinates of the Givens representation, this differential form can be written in the coordinates of the Givens representation as

$$\bigwedge_{i=1}^{p} \bigwedge_{j=i+1}^{n} G_j^T \, J_{Y_i(\Theta)}(\Theta) d\Theta. \quad (3.9)$$

Because this is a wedge product of d d-dimensional elements, Equation 3.9 can be conveniently written as the determinant of the $d \times d$ matrix

$$\begin{pmatrix} G_{2:n}^T J_{Y_1(\Theta)}(\Theta) \\ G_{3:n}^T J_{Y_2(\Theta)}(\Theta) \\ \vdots \\ G_{p:n}^T J_{Y_p(\Theta)}(\Theta) \end{pmatrix}, \quad (3.10)$$

where $G_{k:l}$ denote columns k through l of G. As we show in the Appendix, this term can be analytically simplified to the following simple product whose absolute value serves as

our measure adjustment term:

$$\prod_{i=1}^{p} \prod_{j=i+1}^{n} \cos^{j-i-1} \theta_{ij}. \tag{3.11}$$

3.4.2 Implementation of Angle Coordinates

When using the Givens representation for general Bayesian inference in practice, care must be taken to properly account for pathologies that arise from mapping the Stiefel manifold to Euclidean space. We let $\theta_{12}, \theta_{23}, \cdots \theta_{p,p+1}$ range from $-\pi$ to π and we refer to these specific coordinates as the latidinal coordinates, to evoke the analogy for the simple spherical case. Similarly, we let the remaining coordinates range from $-\pi/2$ to $\pi/2$ and we refer to these coordinates as longitudinal coordinates. This choice of intervals defines a coordinate chart from Euclidean space to the Stiefel manifold, i.e. a mapping between the two spaces. As is inevitable with any coordinate chart between differing topological spaces, there is a subset of the Stiefel manifold of measure zero that the Givens representation will be unable to represent because the topologies of the Stiefel manifold and Euclidean space differ. For $V_{1,3}$ this corresponds to a sliver of the sphere (Figure 3.4). Furthermore, the coordinate chart will contain singularities where the adjustment term (Equation 3.11) becomes zero, possibly biasing any distributional calculations. On the sphere, this corresponds to areas on the unconstrained space being mapped to smaller and smaller areas near the pole (Figure 3.4). We further discuss these pathologies and introduce techniques to overcome them in practice.

As is routinely done in practice, a logistic transform can be used to map the interval $[-\pi, \pi]$ to the unconstrained interval $(-\infty, \infty)$. Unfortunately, this leaves regions of parameter space that should otherwise be connected, disconnected by the aforementioned set of measure zero. In practice, this can lead to biased sampling where regions of

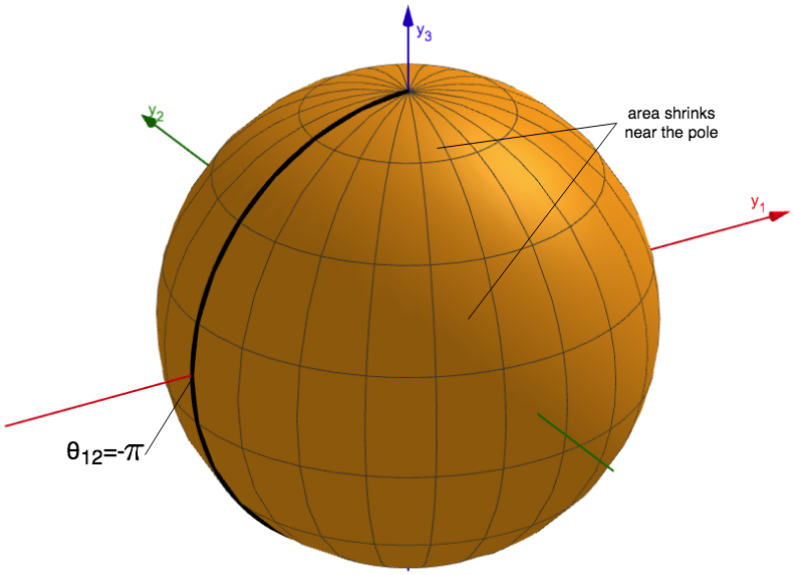

Figure 3.4: The angular coordinates chart has an infinitesimal sliver of measure zero that lies between $\theta_{12} = -\pi$ and $\theta_{12} = \pi$ that separates the two parts of the sphere in the Givens representation. The grid over the sphere reveals how the Givens representation maps areas that are the same size in the Θ coordinates to smaller and smaller regions on the sphere the closer they are to the poles.

parameter space with equal mass are not visited for equal amounts of time if the posterior is not sufficiently concentrated (Figure 3.5, upper).

To overcome this, we create for each longitudinal angle, θ, a pair of coordinates x and y then set $\theta = \arctan(y/x)$. Introducing this auxilary dimension connects otherwise separate regions of parameter space. Furthermore, we let $r = \sqrt{x^2 + y^2} \sim \mathcal{N}(1, 0.1)$. This helps in practice to avoid regions of parameter space where arctan is ill-defined, while leaving the marginal distribution of θ untouched (Figure 3.5, lower).

For the latitudinal angles, we can use the standard technique of constraining the parameters over an interval, then using the logistic transform. However, to avoid singularities in the measure adjustment term, we set the interval to the slightly smaller interval $[-\pi/2+\epsilon, \pi/2-\epsilon]$ rather than the full interval $[-\pi/2, \pi/2]$. Here ϵ is a small value (on the order of 10^{-5} in our experiments) that effectively blocks off a small portion of parameter

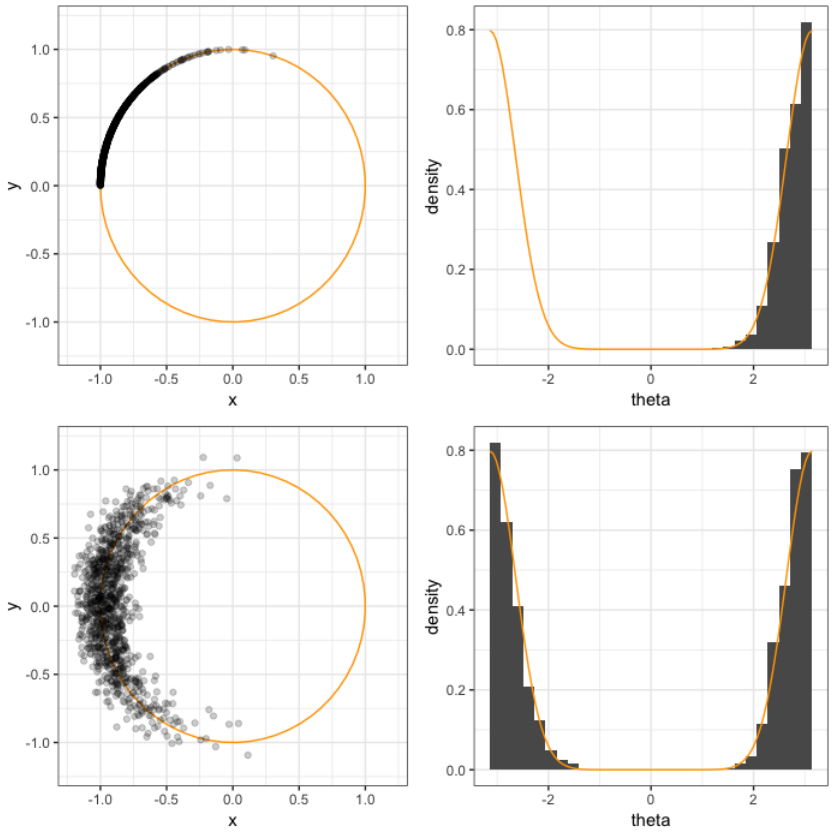

Figure 3.5: (Upper) When posterior mass is not sufficiently bounded away from the edges of the interval, regions of posterior mass that are separated by massless regions may arise. Because these relatively massless regions are difficult for a sampler to traverse, biased sampling can often occur, as the sampler is only able to visit one mode of the posterior distribution. (Lower) By introducing an auxiliary coordinate, one can effecitvely replicate the topology of a circle, effectively "wrapping" the two ends of the interval, leading to unbiased sampling.

space surrounding the singularities of the change of measure term. In the spherical case, this is equivalent to a small patch on either pole that is blocked off. In practice, blocking off this small region avoid issues such as divergences that occur in HMC in such regions of high curvature, while not meaningfully affecting the results of posterior inference.

3.4.3 Coordinate Charts and Identifiability

For certain applications such as PPCA, it may be desirable to further limit parameter space to avoid symmetries that lead to identifiability issues in the posterior. In the Givens representation coordinates this is simply a matter of constraining the range of the longitudinal angles to the interval $[-\pi/2, \pi/2]$ (Figure 3.6). Unfortunately, as in the case of the full interval, this can lead to biased sampling due to regions of low mass separating regions of high mass in parameter space. However, this issue can be resolved by carefully connecting these regions via a simple mirroring technique which we describe next.

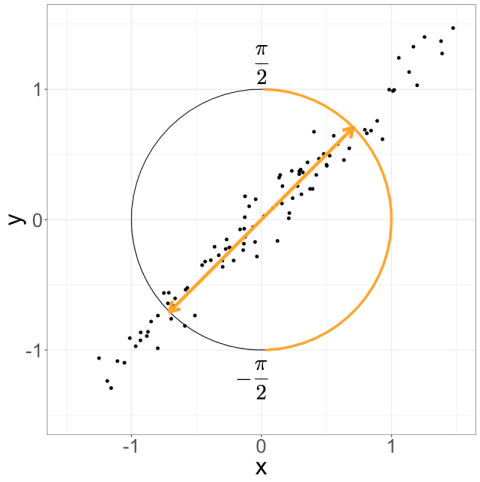

Figure 3.6: PPCA seeks to find the best lower dimensional p-frame to describe a high-dimensional set of points. For $n = 2$ and $p = 1$, this corresponds to the vector that most closely describes a set of two-dimensional points that lie close to flat line. Since a p-frame and its negative can describe the data equally well, a multi-modal posterior over the Stiefel manifold results. By limiting the longitudinal angle to lie in the interval $[-\pi/2, \pi/2]$ the sampler does not consider this redundant mode.

We can allow the original longitudinal and latitudinal coordinates, θ_{lon} and θ_{lat} to freely roam the Stiefel manifold using the aforementioned approach then define the new transformed parameters θ^*_{lon} and θ^*_{lat} to essentially be mirrored versions of these original

coordinates. Specifically, we can define

$$\theta^*_{\text{lon}} = \begin{cases} \theta_{\text{lon}}, & |\theta_{\text{lon}}| \leq \frac{\pi}{2} \\ -\frac{\pi}{2} + (\theta_{\text{lon}} - \frac{\pi}{2}), & \theta_{\text{lon}} > \frac{\pi}{2} \\ \frac{\pi}{2} + (\theta_{\text{lon}} + \frac{\pi}{2}), & \theta_{\text{lon}} < -\frac{\pi}{2} \end{cases} \tag{3.12}$$

and

$$\theta^*_{\text{lat}} = \begin{cases} \theta_{\text{lat}}, & |\theta_{\text{lon}}| \leq \frac{\pi}{2} \\ -\theta_{\text{lat}}, & |\theta_{\text{lon}}| > \frac{\pi}{2}. \end{cases} \tag{3.13}$$

These transformed coordinates essentially mirror and reflect the original coordinates so that once the hemisphere is crossed, the path taken continues on the opposite side of the Stiefel manifold, where there would naturally be an area of high posterior mass (Figure 3.7). In fact, one can check that the PPCA likelihood (Equation 3.16) is continuous with respect to these new coordinates, allowing for efficient sampling even when there is appreciable posterior mass near the edge of the hemisphere.

3.4.4 New Distributions Using the Givens Representation

Rather than placing priors over standard orthogonal matrix coordinates, Y, one can place priors over the coordinates of the Givens representation Θ. In practice this leads to new classes of possible distributions. [44] utilize sparsity promoting priors over the coordinates of the Givens representation to produce a distribution over the Stiefel manifold that favors sparse matrices. They apply this distribution to the estimation of normal mixture classification probabilities. [45] make use of a different parameterization of orthogonal matrices to define a distribution over orthogonal matrices.

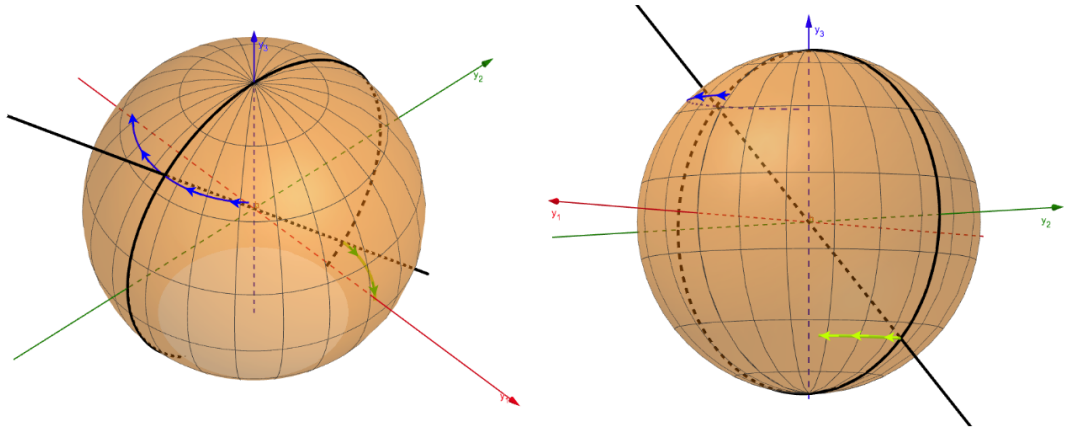

Figure 3.7: Occasionally, the direction that best describes a high-dimensional dataset in PPCA (black line) is near the boundary of the longitudinal coordinate (thick black border). In this case, its negative will have an appreciable probability mass near it and the same density. Because of this continuity in the density, these areas of parameter space can be smoothly connected. Specifically, once the border is crossed (blue path) the coordinates now describe a point on the opposite end of the Stiefel manifold (green path).

3.4.5 Computational Scaling of the Givens Representation

The primary computational cost in using the Givens representation is the series of d $n \times n$ matrix multiplications applied to $I_{n,p}$ in Equation 3.6. Fortunately, unlike dense matrix multiplication, applying a Givens rotation to an $n \times p$ matrix involves only two vector additions of size p (Algorithm 2). Thus, since d scales on the order of np, computation of the Givens representation in aggregate scales as $\mathcal{O}(np^2)$.

In comparison, EMHMC involves an orthogonalization of an $n \times p$ matrix which scales as $\mathcal{O}(np^2)$ and a matrix exponential computation that scales as $\mathcal{O}(p^3)$ [40]. In practice, we find that EMHMC scales better when p is much smaller than n, whereas the Givens representation scales better when p is large and closer to n. We present benchmarks in the following section.

Input: θ
Result: Y
$Y = I_{n,p}$; idx $= d$
for *i in p:1* **do**
  for *j in n:(i+1)* **do**
    $Y_i = \cos(\theta_{\text{idx}})Y[i,] - \sin(\theta_{\text{idx}})Y[j,]$
    $Y_j = \sin(\theta_{\text{idx}})Y[i,] + \cos(\theta_{\text{idx}})Y[j,]$
    $Y[i,] = Y_i$
    $Y[j,] = Y_j$
    $idx = idx - 1$
    log density $+= (j - i - 1) \log \cos \theta_{\text{idx}}$
  end
end
return Y

Algorithm 2: Psuedo-code for obtaining the orthogonal matrix Y from the Givens Representation as well as appropriately adjusting the log of the posterior density.

3.5 Results and Examples

We demonstrate the use of the Givens representation and compare it with EMHMC for three common statistical examples from the literature. All Givens representation experiments were conducted in Stan using Stan's automatic warm-up and tuning options. For all Stan experiments, we ensured that there were no divergences during post-warmup sampling and that all $\hat{R}$ were 1.01 or below. All timing experiments were conducted on a 2016 Macbook Pro.

3.5.1 Uniform Sampling on the Stiefel Manifold

We sample uniformly from the Stiefel manifold of various sizes to assess the practical scalability of the Givens representation. We compare its sampling efficiency and $\hat{R}$ values to EMHMC on 500 post-warmup samples from each method (Table 3.1).

As mentioned in Section 3.4.1, to uniformly sample the Stiefel manifold in the Givens representation, the change of measure term, Equation 3.11, must be computed as part of the likelihood. Meanwhile, uniform sampling over the Stiefel manifold is achieved

		EMHMC		Givens	
p	n	$\hat{R}$	n_{eff}	$\hat{R}$	n_{eff}
1	10	1.00	231	1.00	496
1	100	1.00	317	1.00	488
1	1000	1.00	238	1.00	487
10	10	1.00	408	1.00	390
10	100	1.00	473	1.00	487
10	1000	1.00	454	1.00	488
100	100	1.00	484	1.00	479

Table 3.1: $\hat{R}$ and n_{eff} values averaged over all elements of the matrix parameter Y.

in EMHMC simply using a constant likelihood because the method uses the original matrix coordinates. However, as mentioned in Section 3.4.5, this comes at the cost of an expensive HMC update to ensure that the updated parameter still satisfies the constraints.. In practice, we find that EMHMC scales better as n is increased, although the approach using the Givens representation in Stan remains competitive (Figure 3.8).

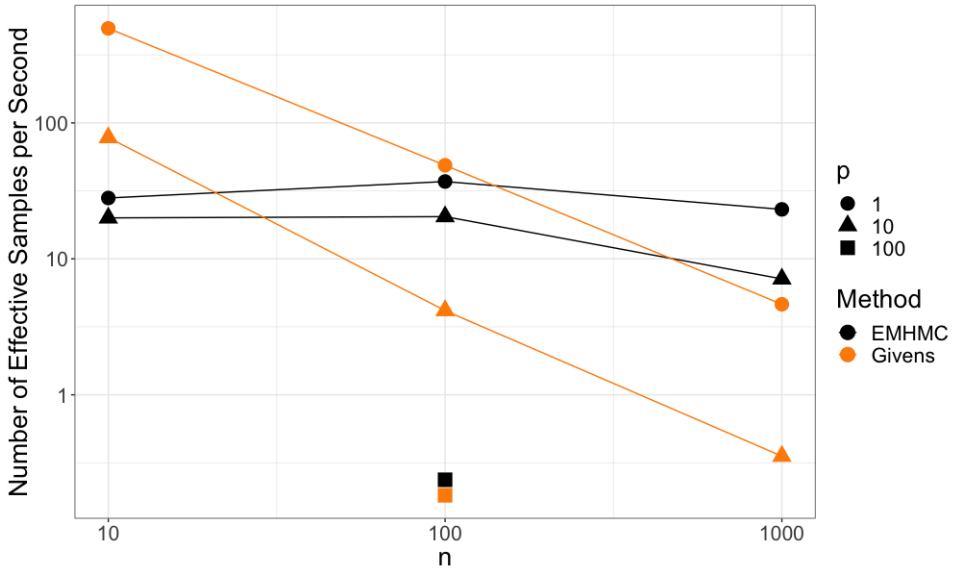

Figure 3.8: For small values of n the Givens representation approach in Stan produces more effective samplers per second while for larger values the EMHMC scales better since the primary cost of the matrix exponential remains constant.

3.5.2 Probabilistic PCA (PPCA)

Factor Analysis (FA) and Probabilistic PCA (PPCA) [46] posit a probabilistic generative model where high-dimensional data is determined by a linear function of some low-dimensional latent state [24, Chapt. 12]. Geometrically, for a three-dimensional set of points forming a flat pancake-like cloud, PCA can be thought of as finding the best 2-frame that aligns with this cloud (Figure 3.9). Formally, PPCA posits the following generative process for how a sequence of high-dimensional data vectors $\mathbf{x}_i \in \mathbb{R}^n$, $i = 1, \cdots, N$ arise from some low dimensional latent representations $\mathbf{z}_i \in \mathbb{R}^p$ $(p < n)$:

$$\begin{aligned} \mathbf{z}_i &\sim \mathcal{N}_p(0, I) \\ \mathbf{x}_i|\mathbf{z}_i, W, \Lambda, \sigma^2 &\sim \mathcal{N}_n(W\Lambda\mathbf{z}_i, \sigma^2 I). \end{aligned} \tag{3.14}$$

To ensure identifiability, W is constrained to be an orthogonal $n \times p$ matrix, while Λ is a diagonal matrix with positive, ordered elements. Because $\mathbf{x}_i$ is a linear transformation of a multivariate Gaussian, its distribution is also multivariate Gaussian with mean zero and covariance $\mathbf{C} := W\lambda^2 W^T + \sigma^2$ [24]. Letting $\hat{\Sigma} := (1/N)\sum_{i=1}^N \mathbf{x}_i\mathbf{x}_i^T$ denote the empirical covariance matrix, this gives us the simplified PPCA likelihood

$$p(\mathbf{x}_1, \cdots, \mathbf{x}_N|W, \sigma^2) = -\frac{N}{2}\ln|\mathbf{C}| - \frac{1}{2}\sum_{i=1}^N \mathbf{x}_i^T\mathbf{C}^{-1}\mathbf{x}_i \tag{3.15}$$

$$= -\frac{N}{2}\ln|\mathbf{C}| - \frac{N}{2}\mathrm{tr}(\mathbf{C}^{-1}\hat{\Sigma}). \tag{3.16}$$

Traditional PCA corresponds to the closed-form maximum likelihood estimator for W in the limit as $\sigma^2 \to 0$, providing no measure of uncertainty for this point-estimate.

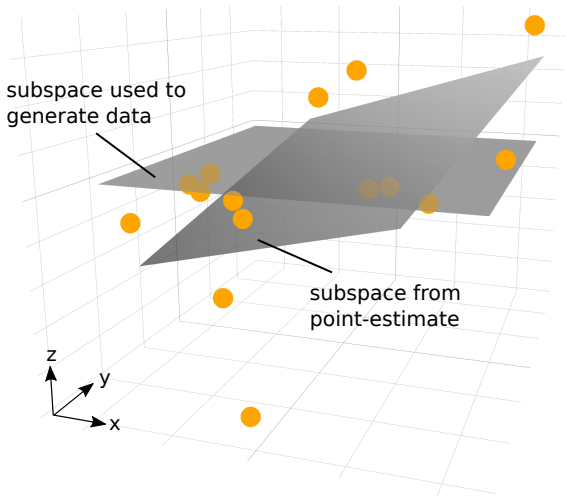

Figure 3.9: PCA finds a single orthogonal matrix in the Stiefel Manifold that is closest, in terms of average squared distance, to the set of points. This point estimate can often mislead us from the true subspace, which in this case is the horizontal (x, y)-plane which was used to generate the noisy data. The data shown here is within the three--dimensional space parameterized by x, y, and z. Alternatively, in Probabilistic PCA (PPCA) a posterior distribution is used to estimate the approximating subspace and also to quantify the uncertainty of the result.

Furthermore, for more elaborate models, the analytical form of the maximum-likelihood estimator is rarely known.

We used the Givens representation to infer this model using simulated data. Specifically, we generated a three-dimensional dataset that lies on a two-dimensional plane with $N = 15$ observations according to the above generative process. The data is plotted in Figure 3.9). We chose $\text{diag}(\Lambda) = \text{diag}(2, 1)$, $\sigma^2 = 1$, and W to be $I_{3,2}$, which in the Givens representation corresponds to $\theta_{12} = \theta_{13} = \theta_{23} = 0$ i.e. the horizontal plane. We point out how this horizontal plane differs from the slanted plane obtained from the classical PCA maximum likelihood estimate (Figure 3.9). In this case, the advantage of the full posterior estimate that the Bayesian framework affords is clear. Posterior samples of θ_{13}, which if we recall from Figure 3.1 is the Givens representation angle that controls the upwards tilt of the plane, reveal a wide posterior which cautions us against the spurious

maximum likelihood estimate of $\hat{\theta}_{13} = -0.15$ (Figure 3.10, right). Note also that by naturally constraining the set of angles considered as in Section 3.4.3, the superfluous modes that EMHMC visits are avoided. Likewise, posterior distributions of Λ are more informative than point estimates for quantifying the inherent dimensionality of the data (Figure 3.10, left).

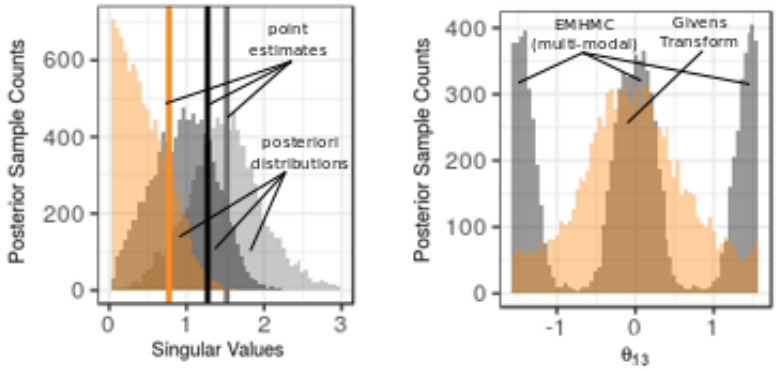

Figure 3.10: PPCA inference for three-dimensional synthetic data. (Left) Posterior draws of the Λ parameter are more informative in dimensionality selection than point-estimates. The posterior distributions for Λ_1 and Λ_2 (dark grey, and grey) contain almost no mass near zero, suggesting that the data probably contains significant variation in those directions. Meanwhile, the posterior Λ_3 (orange) has it posterior mode at zero, suggesting there is a high probability that this parameter is close to zero, which the point estimate by itself neglects to convey. (Right) By limiting the angles of rotation in the Givens Transform, we can further avoid unidentifiability in our problem and eliminate multi-modal posteriors that show up in other methods such as EMHMC.

3.5.3 The Network Eigenmodel

To illustrate the Givens representation on a more elaborate model with orthogonal matrix parameters, we used it to infer the network eigenmodel of [37] on real data and compared it to EMHMC. The same model was inferred using EMHMC by [40]. The data, which was originally described in [47] and freely available in the R package *eigenmodel*,

consists of a symmetric 230×230 graph matrix, Y, which encodes whether the proteins in a protein network of size $n = 230$ interact with one another.

The probability of a connection between all combinations of proteins can be described by the lower-triangular portion of a symmetric matrix of probabilities, however the network eigenmodel uses a much lower dimensional representation to represent this connectivity matrix. Specifically, given an orthogonal matrix U, a diagonal matrix Λ, and a scalar c, then letting $\Phi(\cdot)$ represent the probit link function, the model is described as follows:

$$
\begin{aligned}
c &\sim \mathcal{N}(0, 10^2) && (3.17)\\
\Lambda_i &\sim \mathcal{N}(0, n),\ \forall i && (3.18)\\
Y_{ij} &\sim \text{Bernoulli}\left(\Phi([U\Lambda U^T]_{ij} + c)\right),\ \forall i > j. && (3.19)
\end{aligned}
$$

The Stan implementation using the Givens representation took approximately 300 seconds to collect 1000 samples, 500 of which were warmup. In contrast, EMHMC took 812 seconds to run the same 1000 samples using the hyperparameter values specified in [40]. Figure 3.11 compares traceplots for c, Λ, and the elements of the top row U for the 500 post warmup samples from each sampler. As mentioned in [40] the non-ordering of the Λ parameters results in a multimodality in the posterior whereby values of Λ can be "flipped". Computed $\hat{R}$ and n_{eff} for these parameters are shown in Table 3.2.

3.6 Discussion

We have introduced a systematic approach to incorporating the Givens representation into a general Bayesian inference framework for the purpose of inferring general Bayesian

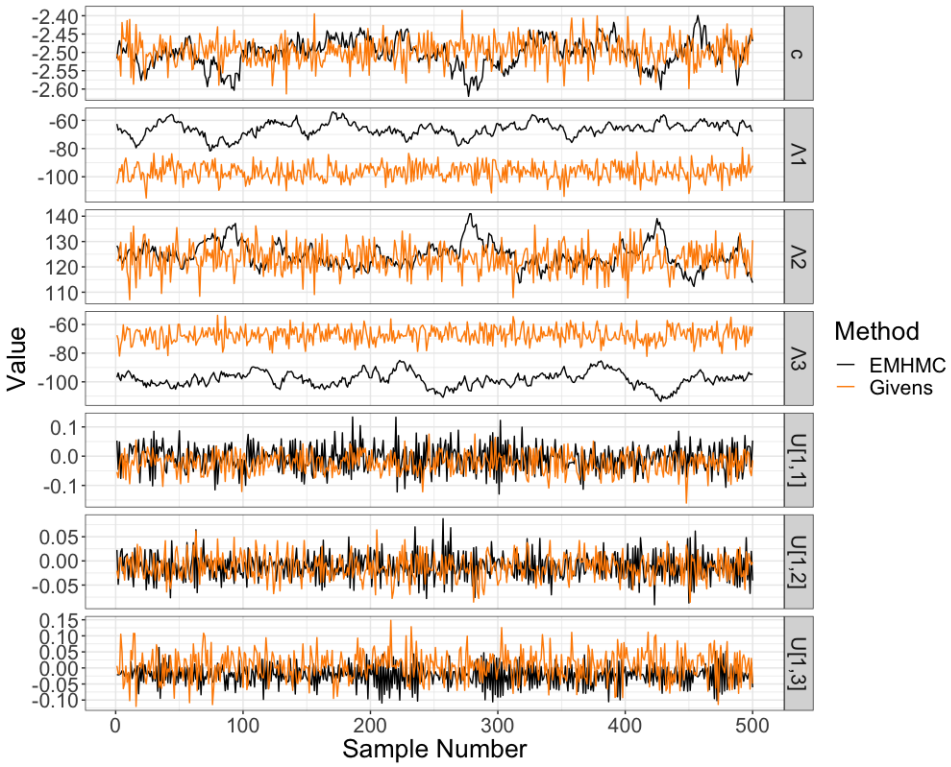

Figure 3.11: Traceplots of samples from the Givens representation implementation in Stan and EMHMC reveal the multimodality in the elements of Λ. For brevity, only the top three elements of U are shown.

models with orthogonal matrix parameters. Our approach overcomes practical barriers to using the Givens representation in such a setting, including having to efficiently compute the measure adjustment term and dealing with singularities caused by differences in topology. Furthermore, we also provided an intuitive explanation behind the Givens representation that is accessible to statisticians and followed with practical examples for which we provide code. We expect our approach can be used quite widely in practice by a variety of practitioners.

	EMHMC		Givens	
Parameter	$\hat{R}$	n_{eff}	$\hat{R}$	n_{eff}
c	1.00	22	1.00	496
Λ_1	1.00	19	1.00	500
Λ_2	1.00	23	1.00	500
Λ_3	1.10	18	1.00	500
$U[1,1]$	1.01	500	1.00	500
$U[2,1]$	1.00	500	1.00	500
$U[3,1]$	1.02	500	1.00	500

Table 3.2: $\hat{R}$ and n_{eff} values for the parameters in the network eigenmodel. For brevity, only three of the matrix parameters are shown.

Chapter 4

Hierarchical Factor Analysis in Coagulopathy

In this chapter, we present our work titled "Understanding Coagulopathy using Multi-view Data in the Presence of Sub-Cohorts: A Hierarchical Subspace Approach" [48] published in the proceedings of the 2018 Machine Learning for Healthcare Conference in the Journal of Machine Learning Research (JMLR). Death from trauma is most often the result of uncontrollable bleeding as a result of Acute Traumatic Coagulopathy (ATC), a disease that manifests itself differently in different sub-cohorts of trauma patients. Understanding the mechanisms of ATC and how existing patient tests can inform us about these mechanisms is key to treating the disease. We introduce a hierarchical Canonical Correlation Analysis (CCA) model that captures a lower dimensional representation of the coagulation system based on blood protein and other tests. The hierarchial nature of the model is ideal in the setting where multiple sub-cohorts are present, but statistical strength can reasonably be borrowed from similar groups. We illustrate how the model may be useful in understanding and treating ATC.

4.1 Introduction

Trauma is the leading cause of death between the ages of 1 and 44 [49]. Major trauma often induces a coagulopathic state known as Acute Traumatic Coagulopathy (ATC) that manifests in increased bleeding and resultant mortality [50]. Despite the coagulation cascade being a well-studied field in biochemistry, the causes and mechanisms behind ATC are still uncertain [51].

One of the primary difficulties in treating ATC is information scarcity. During trauma care, the state of the coagulation system is often assessed by thromboelastography (TEG). This assay allows for quick assessment of the patient's clotting potential using the viscoelastic properties of the patient's blood such as time to clot formation, clotting rate, clot strength, and rate of clot lysis. Unfortunately, TEG does not provide much insight on specific clotting factor levels which can highlight the abnormalities causing ATC. Assays that test for plasma protein levels can take much longer than TEG or may not be available. Thus, treatment decisions must be made with limited information. If we can use historical data and other patient data to ascertain the relationship between plasma protein levels and TEG, decision making can be improved at the point of care.

While some protein data has been collected and analyzed [52], a consensus mechanism for ATC remains to be found. One possible reason for this is that there may not a single mechanism behind ATC. Currently, ATC is characterized primarily by clot times, but assuming a single mechanism can obfuscate the more likely heterogeneous nature of the disease that has been alluded to previously in the literature [52]. The coagulation system is a complex network of reactions with many positive and negative feedback mechanisms [53]. Because of this, there are many possible mechanisms that can cause the system to fail. Distilling these mechanisms into a small collection of easily-interpretable variables would go a long way towards better understanding of ATC. This could potentially enable

recognition of phenotypes using limited information such as TEG results, in order to inform decision making in practice.

Further complicating the issue is the inherent heterogeneity in patients and injury types. Coagulopathy is not relegated to any particular form of trauma or injury mechanism, e.g. car accidents or gunshot wounds. A-priori, we would expect that patients with different types of injury might have different coagulopathic phenotypes and cannot be grouped together in an analysis.

We propose a probabilistic graphical model that extends canonical correlation analysis (CCA) to distill the complex network of interactions in trauma-induced coagulopathy into a small set of easy-to-understand variables. The model extends CCA in such a way as to allow different sub-cohorts (based on injury mechanism) to be represented by different latent variables, i.e. subspaces. At the same time, the model is able to "borrow" information between similar sub-cohorts when data in a particular sub-cohort is sparse. This latter feature is accomplished by placing a hierarchical prior over the orthornormal matrices of weights used in CCA, via the recently introduced Givens representation. To our knowledge this is the first application of hierarchical modeling (in the Bayesian sense) to CCA, PCA, or any kind of model with orthornormal matrix parameters, as this task was previously intractable for even small problems without a change of representation that the Givens transform provides.

We show how, when applied to blood protein assays and TEG measurements, our model can find different low-dimensional descriptions of the coagulation system for different sub-cohorts of injured patients, and how the model can share information across sub-cohorts, via hierarchical modeling. We then illustrate how the shared subspace properties of CCA may be used in a clinical setting to phenotype patients and guide treatment using TEG measurements (i.e. without expensive and time-consuming protein assays).

4.2 Cohort

Our dataset consists of whole blood collected from 174 patients upon arrival to the emergency department of a Level I trauma center between 2005 to 2015 as part of an ongoing study [54].

4.2.1 Cohort and Sub-Cohort Selection

To remove any bias arising from age and differing injury severity, we selected patients below the age of 45 and with an injury severity score greater than or equal to 25, a common threshold used to mark severe injury in past studies. We examined patients suffering from gun shot wounds (GSW), motor vehicle collisions (MVC), stab wounds (SW) and assault, representing a diverse set of possible injury mechanisms. These divisions are also representative of groups with larger sample sizes (GSW) and smaller sample sizes (assault) to illustrate the inferential power of our hierarchical model. A summary of our sub-cohorts is given in Table B.1.

4.2.2 Feature Choices

For each patient, we examined the TEG and blood protein assays. These consist of values of R, K, MA, and Ly30 for TEG and FactorII, FactorX, Protein C, D-Dimer, Fibrinogen, and Platelets for blood protein. The values are collectively referred to as x_{teg} and x_{prot} for the remainder of this work. Table B.2 in the Appendix summarizes the number of missing entries for these values amongst the 174 total patients. We describe in the methods section how we account for these missing values.

4.3 Methods

We start with a description of how we handle missing values in our analysis, and give a short introduction into Canonical Correlation Analysis. Next, we provide a description of how we conduct full Bayesian posterior inference over the orthornormal matrix parameters present in CCA using the Givens transform and Stan [30]. Finally we describe our novel extension to CCA that takes into account the heterogeneity between sub-cohorts present in our dataset via Bayesian hierarchical modeling.

4.3.1 Missing Values

We use multiple imputation as described in [55] to conduct inference while properly accounting for the extra uncertainty that arises from missing data. Specifically, we draw five imputed datasets, conduct Hamiltonian Monte Carlo (HMC) inference in Stan separately on these imputed datasets, and then combine all posterior samples (after sampling is complete) into a single pool of samples. In this way our posterior samples represent a mixture of posteriors under various imputation possibilities. For actual imputation we use a nonparametric random imputation method based on CART and described in [55].

4.3.2 CCA and Probabilistic CCA (PCCA)

Canonical correlation analysis (CCA) is an extension of PCA used to extract cross-covariance information about a pair of related datasets, or views, that are tied to a common set of samples [24]. The classical formulation of CCA finds a pair of linear projections that maximize the shared variance between the two views. Inspecting the obtained projections allows one to observe the level of commonality across views.

The probabilistic formulation of CCA (PCCA), [56], posits a generative process for the same task. Formally, if we have two sets of different data types, or views, $x_{\text{prot}} \in \mathbb{R}^{D_{\text{prot}}}$

and $x_{\text{teg}} \in \mathbb{R}^{D_{\text{teg}}}$, we can define for each view a set of orthonormal loading matrices W and B and corresponding diagonal matrices, Λ and Γ, that describe the variance explained by each latent dimension. Then for each sample $i \in (1, ...N)$, we can define three lower-dimensional latent variables: one for each view, $z^{(i)}_{\text{prot}} \in \mathbb{R}^{L_{\text{prot}}}$ and $z^{(i)}_{\text{teg}} \in \mathbb{R}^{L_{\text{teg}}}$, and a shared latent variable that connects both views: $z^{(i)}_s \in \mathbb{R}^{L_s}$. We define $z^{(i)} := (z^{(i)}_{\text{prot}}, z^{(i)}_{\text{teg}}, z^{(i)}_s)$. A single data point for sample i is generated according to the following specification:

$$z^{(i)} \sim \mathcal{N}(z^{(i)}_{\text{prot}}|0, I_{L_{\text{prot}}})\mathcal{N}(z^{(i)}_s|0, I_{L_s})\mathcal{N}(z^{(i)}_{\text{teg}}|0, I_{L_{\text{teg}}}) \tag{4.1}$$

$$x^{(i)}_{\text{prot}} \sim \mathcal{N}(x^{(i)}_{\text{prot}}|B_x\Gamma_x z^{(i)}_{\text{prot}} + W_{\text{prot}}\Lambda_{\text{prot}} z^{(i)}_s + \mu_{\text{prot}}, \sigma^2_{\text{prot}} I_{D_{\text{prot}}}) \tag{4.2}$$

$$x^{(i)}_{\text{teg}} \sim \mathcal{N}(x^{(i)}_{\text{teg}}|B_y\Gamma_{\text{teg}} z^{(i)}_{\text{teg}} + W_{\text{teg}}\Lambda_{\text{teg}} z^{(i)}_s + \mu_{\text{teg}}, \sigma^2_{\text{teg}} I_{D_{\text{teg}}}). \tag{4.3}$$

The maximum likelihood estimate (MLE) for this model converges to the solution of classical CCA up to an invariant rotation of the axis, as shown in [56]. The intuition behind this generative process is that by introducing a shared latent variable z_s, the learned W matrices capture information that is shared between the two views, while the B matrices capture information that is not contained in the other view. A graphical model of this process is shown in Figure 4.1.

Automatic Dimensionality Selection We note that the diagonal matrices Λ and Γ serve as importance weights, in the sense that they describe the weight given to each latent dimension in predicting the higher dimensional data. When these weights are close to zero, they indicate that a latent dimension has no predictive relationship to the data. Thus, examining these weights can yield insight into the inherent dimensionality of the data. Specifically, using full posterior draws of Λ and Γ allows us to make probabilistic

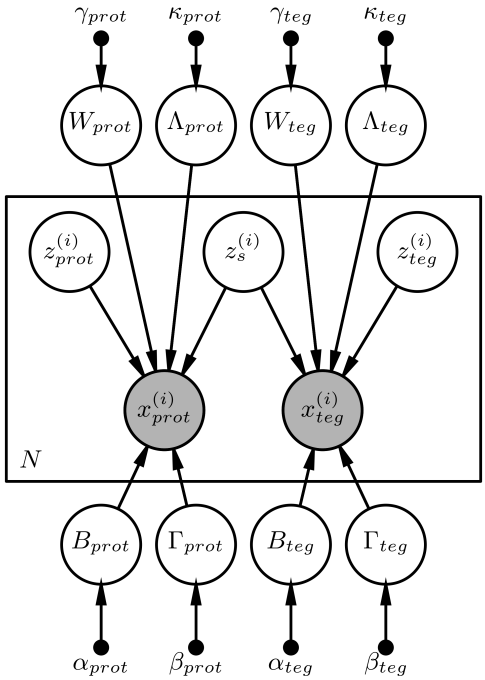

Figure 4.1: Graphical Model of a Non-Hierarchical CCA for our dataset. We attach various zero-centered Cauchy priors on the angles of the orthonormal matrices under the Givens parameterization to induce sparsity in our solutions.

statements about the inherent dimensionality of our data, e.g. if posterior draws show that a weight is below some value, then with high probability we can conclude that the latent dimension is unnecessary. We use this type of analysis for selection of the latent dimension in our results. We place Cauchy priors over importance weights as well as matrix coefficients to induce sparsity, as described in the Appendix.

4.3.3 Hierarchical Modeling and Hierarchical CCA

Treating dissimilar patients, such as patients with different types of injuries, as a single homogeneous group will lead to statistical bias in any inference. On the other hand, a full model where separate parameters are allocated and estimated for each of a multitude of sub-cohorts can lead to large models with many parameters that usually

result in high uncertainty estimates in the fully Bayesian case, and overfit estimates in the maximum likelihood case (lest we provide an inordinate amount of data). Even in the age of big data, this can be an issue, as once we have appropriately sub-cohorted by all relevant features, e.g. injury type, sex, race, age, weight, etc. we are left with small sample sizes within each sub-cohort. Furthermore, certain sub-cohorts are prone to having less data, e.g. severely injured patients usually have fewer test values available because there was no time to collect data for those patients!

Hierarchical models are well known in Bayesian data analysis as an elegant solution to this conundrum. This approach proceeds by treating sub-cohorts as separate entities with their own respective parameters, but additionally modeling the dependence between the groups with a common prior distribution over the parameters [57]. Hierarchical models are very flexible because intuitively they can "adapt" to data by "shrinking" the value of similar groups to a common value when data in a group is sparse, but they allow parameter estimates to approach the value given by the data as more data is made available. The hierarchical model includes as a special case the "full" model, where each group gets its own parameter that is estimated separately, and a "null" model where each group is treated as one big homogeneous group. Hierarchical models allow us to have a model somewhere between these two extremes.

Hierarchical CCA In our probabilistic graphical model, we desire separate CCA parameters for each sub-cohort of injured patients. This calls for separate orthonormal matrices for each sub-cohort, and thus for hierarchical modeling, a prior distribution over orthornormal matrices. At least two distributions over orthornormal matrices exist in the statistics literature [43]. One such distribution is the Matrix Langevin distribution. Incorporating these distributions into a probabilistic graphical model and conducting inference is difficult however, because evaluating their density functions involves computing

the hypergeometric function of a matrix argument, a problem shown to be difficult even for small matrices [58]. We detail in the following section how we use the Givens representation to build a hierarchical CCA model and perform joint analysis of the sub-cohorts.

4.3.4 Givens Representation for Graphical Models

Training of the model was done in Stan using Hamiltonian Monte Carlo (HMC) sampling of posteriors for all latent variables and unknown parameters.

In order to conduct full Bayesian inference on weight matrices and set hierarchical priors on such matrices, we require a way to sample on the space of orthonormal matrices, preferably using a robust sampling method like HMC, a difficult problem in general due to the constraint the samples of matrices must satisfy [40]. While methods exist for HMC sampling of posteriors of general constrained parameters [38, 40], these methods treat constrained and unconstrained parameters separately and require separate numerical integrators for each type of parameter, making them difficult to implement in larger probabilistic graphical models, with complicated priors, such as the model in described in the preceeding section.

In order to sample posteriors of orthonormal matrices, we instead appeal to the Givens representation introduced earlier. Because the Givens representation represents orthornormal matrices as angles, we can place hierarchical priors over orthornormal matrices in a straight-forward manner by simply placing priors over the angles. In our model we do just that, placing truncated normal priors over the respective angles of each group of injured patients. Our full probabilistic graphical model is similar to the model shown earlier in Figure 4.1, but differs in that it contains four times as many group-level parameters, for the four different sub-cohorts described earlier. Furthermore, the truncated normal prior acts as a hierarchical prior over the orthornormal matrices.

4.4 Results

In the first sub-section we apply our model to plasma protein assays and TEG measurements to find low dimensional descriptions of the coagulation system for each respective sub-cohort of injured patients. We illustrate how in a probabilistic framework, we can assess with confidence the differences in sub-cohorts, as well as assess how much information our model captures, using posterior information. We then briefly describe the affect of our hierarchical inference.

In the second subsection we explain how the shared subspace properties of CCA may be used in a clinical setting to phenotype patients and guide treatment using TEG measurements, without protein assays.

4.4.1 Finding Distinct Phenotypes of Coagulopathy

Table 4.1 shows point estimates of the orthornormal matrices of weights, W_{prot} and W_{teg}, that relate the common latent variable, z_s, to measured data, x_{prot} and x_{teg}, for both the gun shot wound (GSW) and motor vehicle collision (MVC) sub-cohorts. Both groups of patients share commonalities in their latent variables. For example, their first latent variables correspond to high FactorII, FactorX, and platelets, but also to low TEG K (speed of clot formation). On the other hand, the model captures differences between the different groups, e.g. FactorX is a smaller component of the first latent variable in the GSW sub-cohort than in the MVC sub-cohort. Full posterior inference provided by Stan and the Givens transform allows us to compare these relationships probabilistically using full posteriors rather than relying on point estimates (see Figure 4.2 for an example). From full posterior draws we can simply count the number of posterior draws where the FactorX weight parameter is higher in the GSW sub-cohort than the MVC sub-cohort, giving us a posterior probability of 0.6, in this case of the statement being true, given

Table 4.1: Point estimates for the orthornormal matrix of weights relating the common (between the protein and TEG data) latent variable to data for each respective sub-cohort. Point estimates were obtained by taking the sample with the largest log probability amongst all 20,000 posterior samples. Cells are colored by intensity of estimated values. Red indicates a strong positive correlation, light red a weak positive correlation, blue a strong negative correlation, and light blue a weak negative correlation.

	GSW		**MVC**	
Proteins	Latent 1	Latent 2	Latent 1	Latent 2
FactorII	0.47	-0.22	0.46	-0.38
FactorX	0.56	-0.09	0.82	0.05
ATIII	0.13	0.06	-0.02	0.57
D-Dimer	-0.15	-0.95	-0.05	0.51
Fibrinogen	0.10	0.17	0.22	0.30
Platelets	0.64	-0.02	0.26	0.41
TEG				
R	-0.74	0.06	0.00	-0.03
K	-0.42	-0.07	-0.84	0.14
MA	0.51	0.14	0.53	0.21
Ly30	-0.06	0.98	0.01	0.97

the data we observed.

Other differences between the two sub-cohorts are apparent in the point estimate of the weight matrix. The model reveals that the second latent variable in the GSW sub-cohort is tied strongly to low D-Dimer and low Ly30, while in the MVC sub-cohort the second latent variable is actually tied to high D-Dimer and low Ly30. Figure 4.3 visually illustrates this sub-cohort specific relationship our model was able to find.

In Figure 4.4 we compare posterior distributions of the "importance" parameters Λ and Γ of each of the latent dimensions. The plot can be suggestive of different inherent dimensionalities across the different subcohorts. For example the assault group has a posterior for the weight of the second shared dimension that is closer to zero while the GSW group's weight for that same weight is concentrated more around a smaller positive value, possibly indicating that of the GSW victims there is an extra dimension to be kept

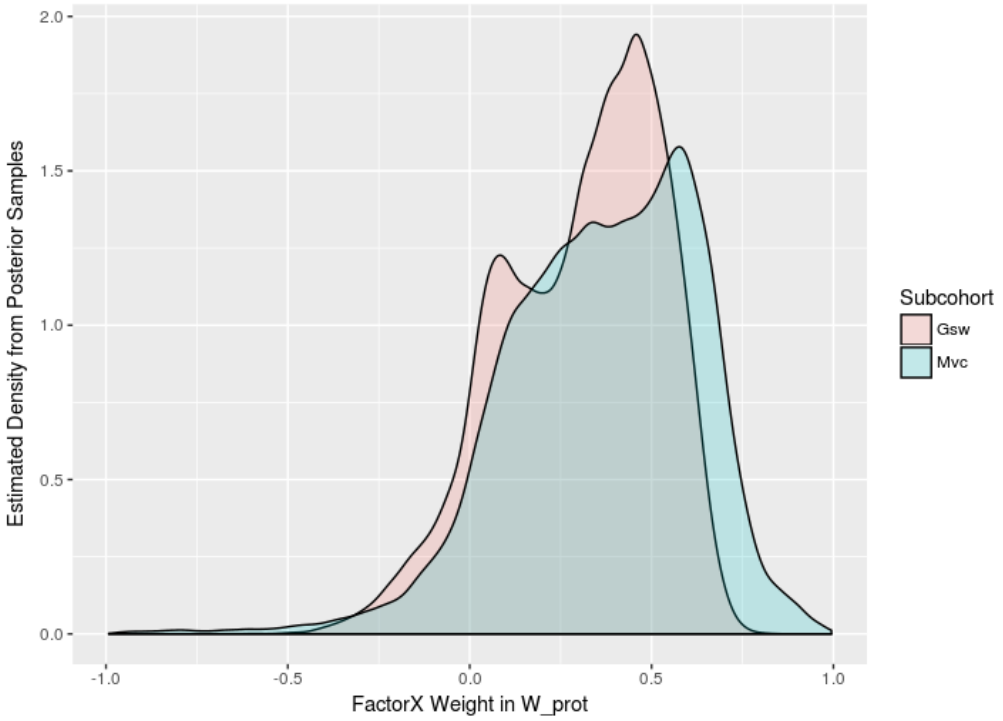

Figure 4.2: Densities estimated from posterior samples of the FactorX weight in the matrix W_{prot}, in the GSW and MVC sub-cohorts respectively. These posterior results indicate the MVC sub-cohort are represented by a latent dimension that places more weigh on FactorX.

track of in their shared latent space.

Lastly, we make note of the shrinkage properties of the hierarchical prior. We conducted separate, non-hierarchical inferences (not shown here) for the GSW group and MVC groups and found that the posterior distributions of their parameters, in particular the W_{prot} and W_{teg} matrices, were left relatively unchanged from the overall hierarchical estimates shown here. These groups had 85 and 52 patients in them respectively. On the other hand, we found much wider posteriors on parameters when estimating the assault group, which consisted of only 16 patients, whereas the hierarchical estimate yields narrower posteriors that are closer to the estimates of the other sub-cohorts.

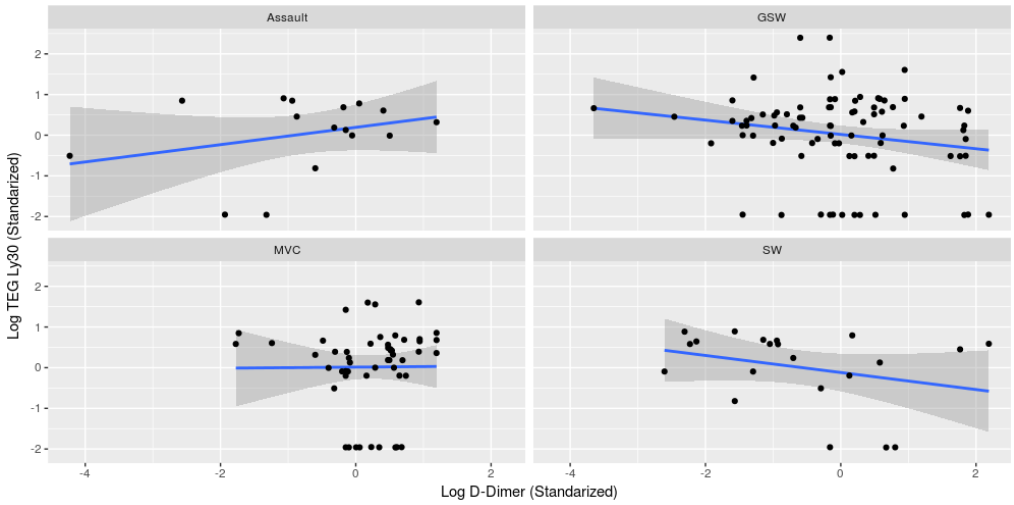

Figure 4.3: Relationship between D-Dimer and TEG Ly30 for different sub-cohorts of patients. For gun shot wounds victims and stab wound victims (both penetrating injuries) the two variables have a negative correlation where as for assault and motor vehicle collisions (both blunt injuries) the relationship is slightly positive.

4.4.2 Using TEG and Injury Data to Predict Phenotypes

Intuitively, the hierarchical CCA model distills the information available in protein and TEG data into shared latent components and ties together the relationship between proteins and TEG by specifying how much of the information carried in these measurements can be ascertained simply by knowing the values of the latent variables. For example, in Table 4.1 we see that latent variable 1 of the GSW sub-cohort is tied to high FactorII and FactorX levels in the protein category, and low R, low K, and high MA in the TEG category. Thus intuitively, given values from a TEG test of a patient and their injury type, we should be able to back out a posterior distribution of the underlying protein values of that patient. In the case of a GSW victim with low R, low K, and high MA, the CCA model suggests that there is a good chance that this patient could have high FactorII and FactorX levels. This information on possible values of the underlying protein levels could serve as valuable information in a real life trauma setting where interventions are guided based on the state of the coagulation system. For example, plasma

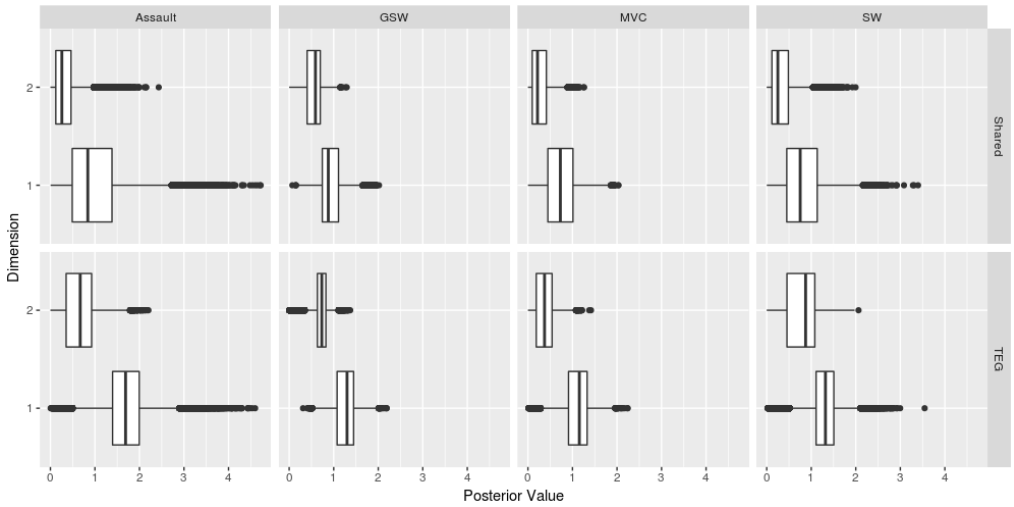

Figure 4.4: Posterior distributions of the "importance" parameters Λ and Γ of each of the latent dimensions. In each facet we show a boxplot for the importance parameters of latent dimension 1 and latent dimnension 2, using samples from their posterior distributions. These weights describe the exploratory power the latent variables have in describing our data (see methods section). The row of shared facets represent importance weights for the latent variable that is shared between the protein and TEG views while the row of TEG facets shows posteriors of importance weights for the TEG-only latent variable.

transfusions are administered when it is thought that a patient's Factor levels are low. Figure 4.5 shows posterior draws of what a GSW patient's FactorII will look like given a TEG reading with low R, low K, and high MA.

The diagonal weight matrix Λ can also roughly tell us how much coagulation protein information we can extract using TEG readings. High posterior values for the weights connecting the shared latent variable to the protein data signify that variance in the protein data can be readily explained by information that is shared between both the protein and TEG views. Low values of those weights or high values of the Γ weights, connecting the latent protein variables to the protein data, would signify that there is information in the protein data that cannot be captured by the shared latent variable and thus can not be captured by TEG.

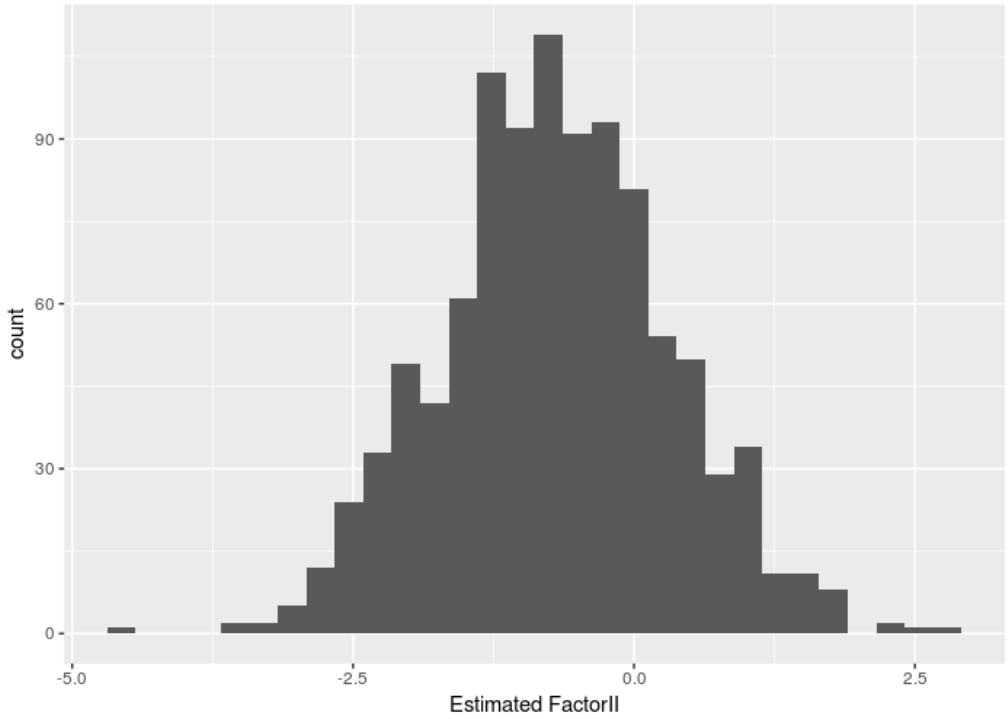

Figure 4.5: Posterior samples of estimated FactorII levels for a patient given their TEG values.

4.5 Discussion and Related Work

Our results show how trauma data can be distilled into low dimensional latent components in such a way as to respect the inherent difference in sub-cohorts of trauma patients. We show how differences in different groups can manifest themselves in different models, and how Bayesian posterior analysis of latent variable "weights" can be used to assess the inherent dimensionality of data in different groups. We then showed the benefits of hierarchical modeling as it pertains to our analysis, and why hierarchical modeling is an essential tool when sub-cohorting. We ended with a possible use case of how understanding the underlying relationship of coagulation proteins and TEG measurements via latent variables can omcrease our understanding of coagulopathy and suggest improvements in the treatment of trauma. While we acknowledge that sub-cohorting by injury type may not be the optimal sub-cohort strategy, our method can be used in more general settings

and especially in medicine where sub-cohorting is important to analysis.

Chapter 5

Mechanistic Models

In this chapter we present our work titled "Relating Disparate Measures of Coagulapathy Using Unorthodox Data: A Hybrid Mechanistic-Statistical Approach" from the proceedings of StanCon Helsinki 2018 [59]. We first describe the nature of coagulopathy and TEG measurements, and then describe our method of fitting ODE models to derived data. Finally, we show results from applying this estimation procedure in practice.

5.1 Coagulopathy and TEG Measurements

Traumatic injury is the leading cause of death for people under the age of 44 [49]. Many of these deaths are the result of uncontrolled bleeding due to a trauma-induced disorder called Acute Traumatic Coagulapathy, or known more simply as Coagulopathy [50]. How major trauma causes Coagulapathy and how to treat the disease is still a subject of ongoing research. There are various competing hypotheses for why so many trauma patients are coagulapathic. The Coagulation Cascade and the Fibrinolytic system are complex networks of dynamically interacting proteins in blood that are responsible for forming and breaking up clots [53].

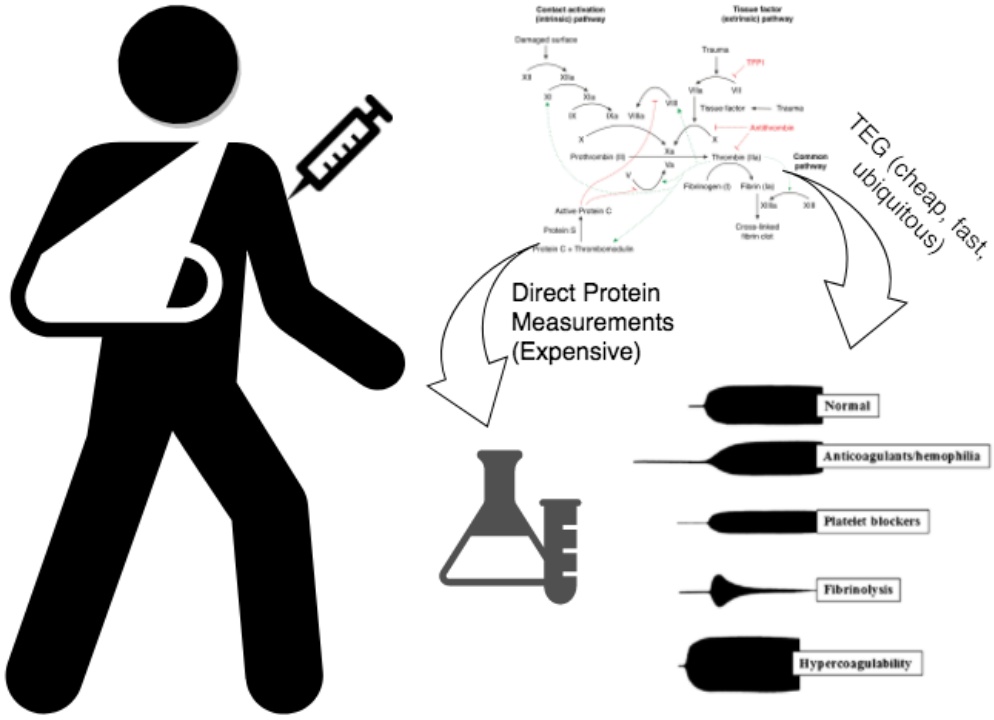

Figure 5.1: Direct protein concetration measurements and TEG are used to examine the state of the Coagulation Cascade of trauma patients using a small blood sample. While direct protein measurements are obviously more informative of the exact state of a patient's blood at any given time, they are slow and expensive to obtain, compared to TEG.

It is generally understood that Coagulopathy comes as a result of a malfunction in one or both of these systems. To better study these complex networks, and how they are affected during trauma, doctors and scientists have two major assays at their disposal: direct protein concentration measurements and Thromboelastography, known as TEG (Figure 5.1). Direct protein concentration measurements can tell us the concentration levels of key players in the body's coagulation system. Thus they can help us to understand why a patient's blood is not clotting and how they can be treated. Unfortunately, these tests are available only at a very select number of hospitals specializing in trauma, they are expensive to run, and most importantly they are slow to run in a setting where applying the correct treatment as quickly as possible is of the utmost importance.

In contrast to direct protein measurements, TEG measurements are ubiquitous, inexpensive to run, and can provide results in as little as 20 minutes. However, they do not measure protein concentrations directly. Instead, TEG works by placing a small sample of blood in a cup, chemically initiating the clotting process, then using a metal probe to measure the physical size of the resulting clot over time in millimeters (mm) (Figure 5.2). The resulting output is a measure of clot thickness over time for the patient that is indicative of several important features of their clotting state including:

- How long it takes for a patient's blood to start forming a clot
- How fast the clot grows once clotting is initiated
- How strong the patient's clots become
- How long the clots are able to maintain their integrity before being broken up

While TEG measurements clearly contain useful information regarding a patient's clotting state, they are simply a proxy for the latent system of clotting proteins in the blood that is much more difficult to measure. Ideally, we can use our mechanistic understanding of the coagulation system in the form of Ordinary Differential Equations (ODEs) along with a statistical model, to better understand what exactly TEG is telling us about the state of the underlying coagulation system, and furthermore to infer a patient's protein concentrations using solely their TEG measurements.

Typically, mechanistic ODE models are fit in Stan using data that consists of the states of the ODE over time, see e.g. [60] or [61]. In these settings one typically posits an error distribution for the data that is centered around the forward simulated values of the ODE. In contrast, our TEG data does not come in the form of clot thickness over time, but rather in the form of four quantities derived from the clot thickness curve that

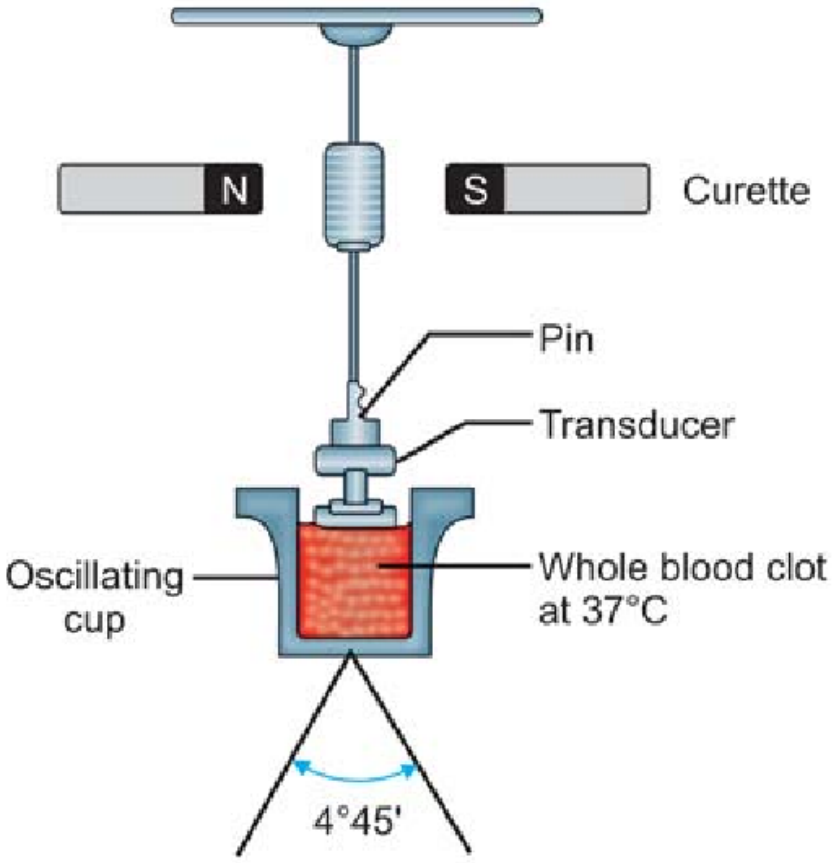

Figure 5.2: In a TEG assay a small sample of blood is placed in a cup which is spun around quickly to initiate the clotting process. A thin metal pin then measures the size over time in millimeters (mm) of the resulting clot.

are typically used in the medical community to summarize the most important properties of a TEG curve (Figure 5.3). These four quantities are described below:

1. R: The time (in minutes) for the clot to reach 2 mm. This quantity represents the time it takes for the clotting process to initiate.

2. K: The time (in minutes) for the clot to reach 20 mm, from the time it reached 2 mm. This quantity represents speed of clot formation.

3. MA: The maximum amplitude of the clot i.e. the size of the clot when it is at its largest. This quantity measures the strength of a patient's clot.

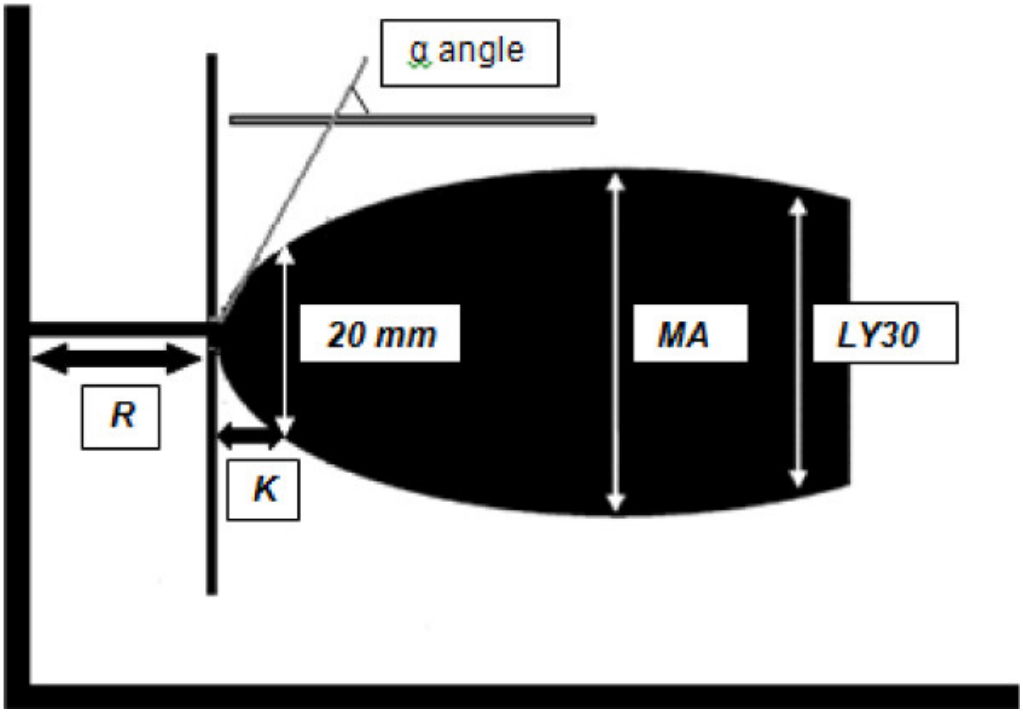

Figure 5.3: TEG curves represent the thickness of a clot over time and are summarized by the four key quantities R, K, MA, and Ly30.

4. Ly30: The percentage of the clot which has broken down after 30 minutes as compared to the maximum amplitude of the clot. This quantity measures how fast clots are being broken up.

5.1.1 A Mechanistic Model of the Coagulation System

The coagulation system is well-studied, with several mechanistic ODE models in the literature that describe how the system evolves dynamically over time. Models for the coagulation system vary widely in the number of states, reactions, and parameters they contain, including complex models with up to 80 states [62]. For our purposes we developed a simple reduced-order model based off of elements from both the work by [62] as well as [63] that captures the most important players in the coagulation system. The model includes the most basic components of the clotting process: the coagulation cas-

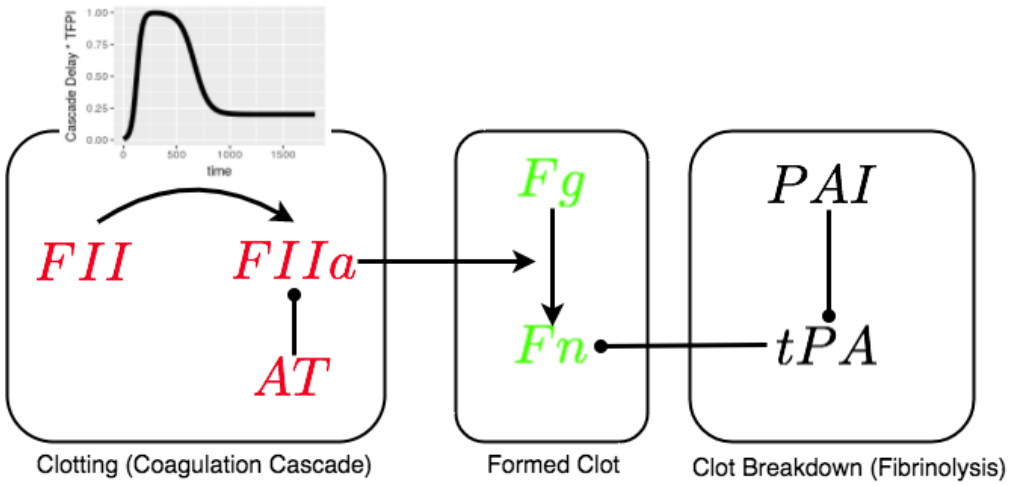

Figure 5.4: A seven-state model of coagulation that models the clotting process, actual clots, and the clot breakdown process. The rate of activation of FII in the coagulation cascade is summarized by a delay term that is governed by parameters that are specific to the patient.

cade responsible for forming clots, actual clot material, and clot breakdown or Fibrinolysis (Figure 5.4).

We model the coagulation cascade as primarily consisting of the activation of the blood protein FactorII (FII) to its activated form $FIIa$ via the action of a sigmoid delay function that is parameterized by parameters b and c. We model these parameter values as specific to the patient. Importantly, $FIIa$ can be blocked by antithrombin (AT). $FIIa$ once activated can then facilitate the conversion of raw clot material, Fibrinogen, or Fg in to an actual clot Fibrin, or Fn. Once a clot is formed, the clot can be broken up by the protein tPA, which itself can be blocked by the protein PAI. The differential equations for this model summarize this process and are shown below. Values for reaction constants represent how fast these respective reactions occur with respect to one another. The values we used for these constant were either gathered from the literature or fit using Maximum A-Posteriori (MAP) estimation.

$$\frac{dFII}{dt} = -\text{CascadeDelay(t, b, c)} \cdot \text{TFPI(t)} \cdot \frac{FII}{K_{FIIa} + FII} \tag{5.1}$$

$$\frac{dFIIa}{dt} = \text{CascadeDelay(t, b, c)} \cdot \text{TFPI(t)} \cdot \frac{FII}{K_{FIIa} + FII} - k_{AT} \cdot FIIa \cdot AT \tag{5.2}$$

$$\frac{dAT}{dt} = -k_{AT} \cdot FIIa \cdot AT \tag{5.3}$$

$$\frac{dFg}{dt} = -k_{clot} \cdot FIIa \cdot \frac{Fg}{K_{clot} + Fg} \tag{5.4}$$

$$\frac{dFn}{dt} = k_{clot} \cdot FIIa \cdot \frac{Fg}{K_{clot} + Fg} - k_{lys} \cdot tPA \cdot \frac{Fn}{K_{lys} + Fn} \tag{5.5}$$

$$\frac{dtPA}{dt} = -k_{PAI} \cdot tPA \cdot PAI \tag{5.6}$$

$$\frac{dPAI}{dt} = -k_{PAI} \cdot tPA \cdot PAI \tag{5.7}$$

$$\tag{5.8}$$

5.1.2 A Mechanistic Model for TEG

While the model includes the concentration of Fn (Fibrin) which is a criticial component of clots, the actual clot thickness which TEG measures is not measuring Fn per se, but some function of it. Following [63], we used the Hill function

$$\text{ClotThickness(t)} = k\frac{Fn^2}{K + Fn^2} \tag{5.9}$$

with $k = 64.0$ and $K = 100.0$ to translate Fn to clot thickness.

5.2 Inferring ODEs Using Hitting Time and Max Data

In the typical ODE estimation setting, where our data consists of the value of the ODE states over time, y_n, $n = 1, \cdots, N$ there typically is a likelihood of the form

$$\prod_{en=1}^{N} p(y_n | y_n^{(sim)}(y_0, \theta)), \tag{5.10}$$

where $y_n^{(sim)}(y_0, \theta)$ is a forward simulation of our ODE, given the initial conditions y_0 and the parameters θ. The Hamiltonian Monte Carlo (HMC) algorithm requires the gradient of the likelihood with respect to the quantities we are trying to estimate, in this case y_0 and θ.

In the case of TEG in coagulapathy, the data consists of hitting times, such as the quantity R, which represents the time it takes for a patient's clot to reach a size of 2 mm. Thus, in a probabilistic model, the likelihood must instead take the form

$$p(R | R^{(sim)}(y_0, \theta)). \tag{5.11}$$

For HMC to obtain the correct gradients of this likelihood, care must be taken in computing the hitting time $R^{sim}(y_0, \theta)$. First, note that a numerical ODE solver will return the value of the solution at discrete time points, which in the case of of TEG, are used to compute clot thickness, C_n at the discrete time points $t_1, \cdots, t_N$. R is a continuous value formally defined as

$$R := \inf\{t : C(t | y_0, \theta) > 2.0\}, \tag{5.12}$$

which unfortunately does not have a smooth derivative with respect to the initial condi-

tions and unknown parameters. In practice, this would cause problems for HMC, because the log-likelihood should be a smooth function of the unknowns, and ideally have smooth derivatives as well.

To ameliorate this, the discrete solution points can be utilized $\{C_n(y_0, \theta)\}$ to obtain a continuous function $C(t|y_0, \theta)$ by interpolating the solution points using cubic splines. The interpolated function will have two smooth derivatives, allowing HMC to run smoothly.

5.2.1 Using Cubic Splines to Obtain Continuous ODE Solutions

Fortunately, the Stan language is expressive enough to allow us to easily implement code for computing a smooth function $C(t)$ that interpolates through the discrete points C_n, $n = 0, \cdots, N$ defined at the points $t_0, \cdots, t_N$. In particular, we can write a function in Stan to fit a cubic interpolating spline through a given set of points. We provide a quick review of cubic smoothing splines and how to compute them loosely following the exposition in [64].

Our aim is to derive the functional form of a group of cubic splines $s_{3,i-1}(t)$ over the intervals $[t_{i-1}, t_i]$. The functions will be cubic polynomials and will be twice differentiable, even at the nodes $t_0, \cdots, t_N$. We first define $f_i = s_3(t_i)$, $m_i = s_3(t_i)$, and $M_i = s_3''(t_i)$ for $i = 0, \cdots, N$. Since $s_{3,i-1}$ is a cubic polynomial, its second derivative is linear. Since the cubic spline must have continuous second derivatives we have

$$s''_{3,i-1}(t) = M_{i-1}\frac{t_i - t}{h_i} + M_i\frac{t - t_{i-1}}{h_i} \tag{5.13}$$

for $t \in [t_{i-1}, t_i]$ where $h_i = t_i - t_{i-1}$. Integrating twice we obtain

$$s_{3,i-1}(t) = M_{i-1}\frac{(t_i - t)^3}{6h_i} + M_i\frac{(t - t_{i-1})^3}{6h_i} + C_{i-1}(t - t_{i-1}) + \widetilde{C}_{i-1} \tag{5.14}$$

The constants C_{i-1} and $\widetilde{C}_{i-1}$ are uniquely determined by imposing the end point values $s_3(t_{i-1}) = f_{i-1}$ and $s_3(t_i) = f_i$, yielding for $i = 1, \cdots, N-1$

$$\widetilde{C}_{i-1} = f_{i-1} - M_{i-1}\frac{h_i^2}{6} \tag{5.15}$$

$$C_{i-1} = \frac{f_i - f_{i-1}}{h_i} - \frac{h_i}{6}(M_i - M_{i-1}). \tag{5.16}$$

By imposing continuity of the first dervivatives at the nodes we arrive at the linear system

$$\mu_i M_{i-1} + 2M_i + \lambda_i M_{i+1} = d_i,\ i = 1, \cdots, N_1, \tag{5.17}$$

where

$$\mu_i = \frac{h_i}{h_i + h_{i+1}} \tag{5.18}$$

$$\lambda_i = \frac{h_{i+1}}{h_i + h_{i+1}} \tag{5.19}$$

$$d_i = \frac{6}{h_i + h_{i+1}}\left(\frac{f_{i+1} - f_i}{h_{i+1}} - \frac{f_i - f_{i-1}}{h_i}\right) \tag{5.20}$$

for $i = 1, \cdots, N-1$. Setting $\lambda_0 = \mu_N = 1$ and $d_0 = d_1$ leads to the following linear

system which defines our spline coefficients:

$$\begin{pmatrix} 2 & \lambda_0 & 0 & \cdots & 0 \\ \mu_1 & 2 & \lambda_1 & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \mu_{N-1} & 2 & \lambda_{N-1} \\ 0 & \cdots & 0 & \mu_N & 2 \end{pmatrix} \begin{pmatrix} M_0 \\ M_1 \\ \vdots \\ M_{N-1} \\ M_N \end{pmatrix} = \begin{pmatrix} d_0 \\ d_1 \\ \vdots \\ d_{N-1} \\ d_N \end{pmatrix} \tag{5.21}$$

Because this system is tridiagonal, we can solve it in $\mathcal{O}(N)$ time using the Thomas algorithm [64].

5.2.2 Obtaining Hitting Times from the Spline Interpolation

With our continuous clot thickness function in hand, we are now able to compute stopping times that are a smooth function of our unknowns by using an algebraic solver on our continuous function and appealing to the implicit function theorem. Letting $C(t|y_0, \theta)$ represent our spline function that interpolates the discrete points $C_n(y_0, \theta)$, R is defined implicitly as

$$C(R(y_0, \theta)|y_0, \theta) = 2.0. \tag{5.22}$$

Note that R, the time that the clot hits 2.0 mm is dependent on the initial value of the system, as well as the parameter values of the system. Taking the partial derivative of both sides of the equation with respect to θ yields

$$\frac{\partial}{\partial R} C(R|y_0, \theta) \frac{\partial R}{\partial \theta} + \frac{\partial}{\partial \theta} C(R|y_0, \theta) = 0.0, \tag{5.23}$$

which then yields the correct partial derivative we need for HMC:

$$\frac{\partial R}{\partial \theta} = -\frac{\partial}{\partial \theta} C(R|y_0, \theta) \left(\frac{\partial}{\partial R} C(R|y_0, \theta) \right)^{-1}. \tag{5.24}$$

This solve can be accomplished and the correct partial derivative will be used in Stan's HMC implementation by simply passing the function $C(t)$ to the algebraic solver available in Stan which uses a modified version of Newton's method to find the solution of nonlinear systems of equations [65].

Because Newton Iterations may diverge with a poor starting guess, and at the time of this writing Stan's algebraic solver does not support variable initial guesses, we opted to implement a custom C++ solver based off of the bisection method.

5.2.3 Obtaining the Max of Our Spline Function

Note that the MA, or maximum amplitude TEG values also requires a nonlinear solve to obtain. To compute this value we also use custom C++ code based off of the bisection method on the derivative of the function. In this case the appropriate gradient can be computed similarly.

5.3 Inferring Hybrid Mechanistic-Statistical Models and the Unorthodox Nature of TEG Data

To use the derived quantities in TEG to infer unknown parameters and initial conditions, we must first forward simulate the ODE, compute the clot thickness as a function of fibrin concentration, Fn, then use the trajectory of clot of thickness over time to derive our simulated TEG data, which we finally can compare to our data by positing some statistical model. We describe the finer points of this process in the following section.

5.3.1 Inferring Protein Concentrations Using TEG Data

Recall, our goal of inferring patient initial concentration of tPA given their TEG measurements. In theory this should be possible because tPA is a protein primarily responsible for clot breakup and the Ly30 measurement of TEG measures the amount of clot breakup after 30 minutes. In this case, the patient has an unusually high Ly30 value of 1.7, indicating that 1.7% of their clot already broke up after only 30 minutes. In light of this, we should expect this patient to have a higher than average tPA concentration.

5.3.2 Prior Selection

Recall that in our mechanistic model every patient has "cascade delay" parameters b and c that describe the specific state of the coagulation cascade. Using our four pieces of TEG data, we must infer the two parameters and also the value of the unknown protein concentrations tPA and PAI for our patient. To do this reasonably, it helps to incorporate any prior knowledge we have about these parameters. For the protein concentrations, we set the prior distributions to exponential distributions with respective means 4e-10 and 9.3e-10. These are the distributions of these protein concentrations for general trauma patients. With the information given it's reasonable to assume our trauma patient's protein values are drawn form the distribution of protein values for trauma patients. For b and c, we use weakly informative priors that are representative of a wide variety of possible coagulation profiles that we would expect to see in trauma patients.

5.3.3 Posterior Analysis

With priors set, the model can be fit, integrating our prior knowledge and data to produce a posterior distribution of this patient's tPA values. Figure 5.5 compares prior and

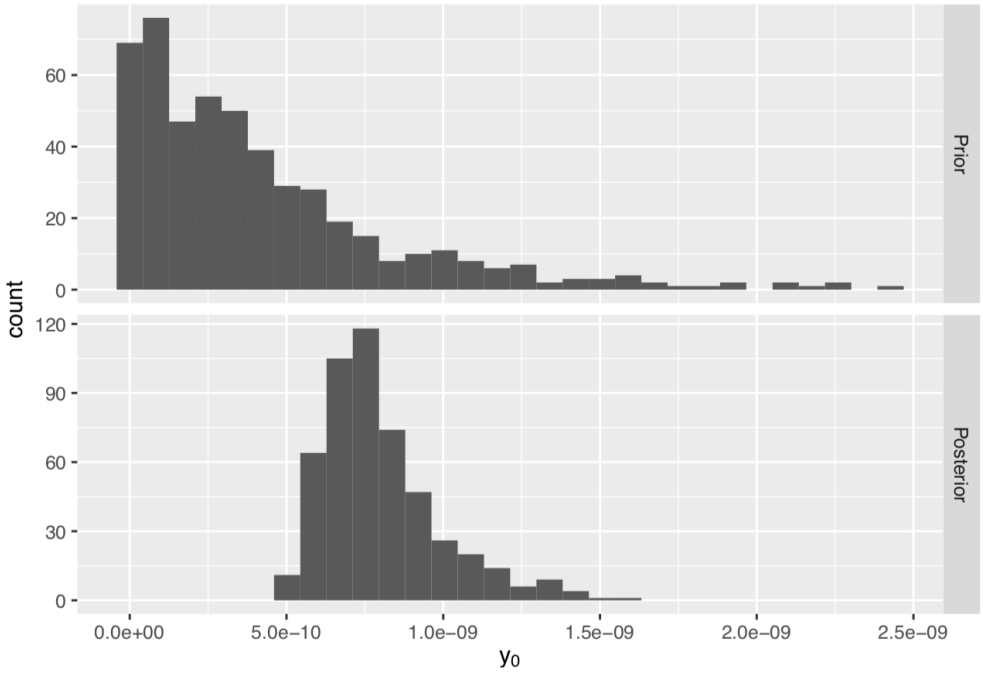

Figure 5.5: Posterior versus prior distribution of tPA values.

posterior tPA values for the example patient. This patient's tPA concentration is higher than the population average, which we used as our prior distribution. This is reflective of their relatively high Ly30 value, as expected. This posterior shrinkage is indicative of the information provided in the data as well as the population-level distributions. We note that the patient data also informs the parameters of the population-level distribution through the estimation of the individual patient parameters.

5.3.4 Discussion

We showed how one can tackle the very practical problem of tying together disparate measures of coagulopathy by incorporating mechanistic and statistical knowledge into Stan. In particular, we showed how TEG data, which consists of unorthodox measurement quantities, can be used to infer much less ubiquitous but more informative protein data with clever use of techniques from numerical analysis such as splines, root-finding,

and the inverse function theorem.

We fit a model to a single patient, first to illustrate our tying together of various numerical techniques, and second to show how a mechanistic model together with Bayesian analysis in Stan can be used in a practical clinical setting to infer useful information. In a larger study, for the purpose of more specific model callibration/checking it may be useful to fit multiple patients, forward simulate TEG data, and check callibration quantities with respect to uncertainty intervals to ensure that modeling and distributional assumptions are sound. We did not pursue this further as it was outside of our scope, but we mention it for completeness.

We also point out that our priors for the protein concentrations were selected using an empirical Bayes approach where we fit a point estimate to protein data from an entire population of patients we had data for. In a sense, this is a computationally convenient approximation to a hierarchical model which would simultaneously incorporate data from multiple patients and learn the population level distributions.

Chapter 6

Implicit Hamiltonian Monte Carlo

We describe our recent work on multiscale posterior distribution and HMC titled "Implicit Hamiltonian Monte Carlo for Sampling Multiscale Distributions" submitted the Springer Journal of Statistics and Computing [66]. Hamiltonian Monte Carlo (HMC) has been widely adopted in the statistics community because of its ability to sample high-dimensional distributions much more efficiently than other Metropolis-based methods. Despite this, HMC often performs sub-optimally on distributions with high correlations or marginal variances on multiple scales because the resulting stiffness forces the leapfrog integrator in HMC to take an unreasonably small stepsize. We provide intuition as well as a formal analysis showing how these multiscale distributions limit the stepsize of leapfrog and we show how the implicit midpoint method can be used, together with Newton-Krylov iteration, to circumvent this limitation and achieve major efficiency gains. Furthermore, we offer practical guidelines for when to choose between implicit midpoint and leapfrog and what stepsize to use for each method, depending on the distribution being sampled. Unlike previous modifications to HMC, our method is generally applicable to highly non-Gaussian distributions exhibiting multiple scales. We illustrate how our method can provide a dramatic speedup over leapfrog in the context

of the No-U-Turn sampler (NUTS) applied to several examples.

6.1 Introduction

The Hamiltonian Monte Carlo (HMC) algorithm [6] and its recent successor, the No-U-Turn (NUTS) sampler [10], have seen widespread use recently in the statistics community because of their proficiency in sampling high-dimensional distributions. In fact, [9] showed that as the dimension, D, of the distribution $p(q)$ being sampled tends to infinity, HMC requires only $\mathcal{O}(D^{5/4})$ samples to sufficiently explore the distribution, while the classic random-walk Metropolis algorithm [8] requires $\mathcal{O}(D^2)$. Roughly speaking, HMC and NUTS achieve this efficiency gain because rather than exploring parameter space in a random fashion, they systematically explore level-sets of Hamiltonian energy by using the leapfrog integrator to numerically simulate Hamiltonian dynamics over the potential energy surface defined by $U(q) = -\log p(q)$ [7]. While the Hamiltonian energy remains roughly constant over these level sets, when simulating Hamiltonian dynamics using the leapfrog integrator, the log-probability density and values of the random variables q often vary quite widely in practice, yielding samples that are much closer to independent than samples from the random-walk Metropolis algorithm [67].

The key to HMC reaching the correct stationary distribution lies in the reversibility of the leapfrog integrator that allows the Markov transitions to satisfy detailed-balance. Furthermore, the leapfrog integrator is volume-preserving in phase-space, which makes the computation of the Metropolis probabilities in HMC trivial [7]. In fact, the leapfrog integrator belongs to a class of numerical integrators known as symplectic integrators that exhibit both of these properties along with a more general property known as symplecticity. Symplectic integrators have been well-studied [39, 68]. However, little work has been done to characterize how the properties of a probability distribution relate to

the properties of the associated Hamiltonian system in HMC, and how these properties make certain integrators more advantageous than others, depending on the problem.

To this end, we provide an analysis of how and why the geometry of Bayesian posterior distributions with low variance components can lead to multiscale Hamiltonian systems, i.e. systems with rapidly oscillating components that force leapfrog to take a much smaller stepsize than what would otherwise be possible with an implicit integrator. We describe the implicit midpoint integrator as an alternative to leapfrog for these types of problems, and show how to practically implement it in an HMC context. Furthermore, we offer practical guidelines on the stepsize to choose when using either leapfrog or implicit midpoint in an HMC sampler, as well as heuristics for when to choose one algorithm over the other. Finally, we compare, using practical examples, the efficiency of leapfrog NUTS (lfNUTS) with an implicit NUTS implementation we introduce called iNUTS. Code for these experiments is available as an R package that can be obtained by emailing the authors.

In Section 6.2 we provide a brief introduction to HMC and then describe the concept of numerical stability of an integrator and how it connects to the geometries of multivariate distributions in HMC. In Section 6.3 we describe the symplectic implicit midpoint algorithm along with our practical custom implementation specific to HMC. We also derive the mathematical stability limit for the implicit midpoint algorithm, showing how it provides a clear advantage over leapfrog on multiscale systems. In Section 6.4 we show how in practice, using real examples, our iNUTS implementation leads to less computational work per effective sample than leapfrog-based NUTS on common multiscale systems. We conclude with a summary of our contributions and future directions in Section 6.5.

Related Work: While various works have explored modifications of the leapfrog integrator in HMC, the connection between posterior geometry and integrator choice as well as the multiscale problem has been sparsely examined. On the statistics side, both [69] and [70] adapted the leapfrog integrator in HMC by splitting the potential energy into a Gaussian term and a non-Gaussian term, which are integrated separately. While these methods can alleviate the multiscale problem for distributions that can be well-approximated by a Gaussian, they fail to offer an efficiency gain over leapfrog for more complicated distributions, as we describe in Section 6.2. Our implicit midpoint-based method is able to achieve efficiency gains over leapfrog on a more general class of problems.

Okudo et al. [71] modify the leapfrog integrator in HMC by adding an auxiliary variable that allows for online adaptation of the stepsize. The RMHMC method of [72] uses local Hessian information of the potential energy function to adaptively change the mass matrix of HMC, which is equivalent to adaptively changing leapfrog stepsizes. While both of these methods can effectively adapt the stepsize in leapfrog to the local geometries of non-Gaussian distributions, they are still subject to using a small integrator stepsize in a multiscale problem, just as standard leapfrog is. In contrast, our implicit midpoint-based approach has much less severe stepsize limitations [73].

6.2 Hamiltonian Monte Carlo and Numerical Stability in a Multiscale Problem

We provide a brief overview of the basic HMC algorithm and how it leads to a Hamiltonian system of ODEs. We then provide a short explanation and illustration of numerical stability of the leapfrog integrator on a linear Hamiltonian system, and

then extend this analysis to show how stability can be a crucial bottleneck in multiscale systems. Finally, using the linearization of arbitrary Hamiltonian systems, we extend the multiscale concept to non-Gaussian distributions and draw the connection between posterior geometry and the mass matrix of HMC. For a more comprehensive review of HMC, see [7]. For a more recent review that includes an exposition on NUTS, see [67]. For a more thorough review of stability analysis for the numerical solution of Hamiltonian systems see [39, Ch. 2.6].

6.2.1 Hamiltonian Monte Carlo

HMC provides samples from an arbitrary distribution over $q \in \mathbb{R}^D$ with density $p(q)$ by taking Markov transitions that satisfy detailed-balance with respect to $p(q)$. In HMC, a potential energy surface $U(q) = -\log p(q)$ is defined, along with a kinetic energy, $K(p) := p^T M^{-1} p$, and a Hamiltonian, $H(q,p) := U(q) + K(p)$. Given a starting point q_0 on this surface, an HMC transition begins by sampling a random momentum, $p_0 \in \mathbb{R}^D$, from a multivariate normal distribution with mean zero and covariance M [67]. Given these initial values q_0 and p_0, the Hamiltonian system

$$\begin{aligned} q' &= \frac{\partial H}{\partial p} = M^{-1} \\ p' &= \frac{\partial H}{\partial p} = -\nabla_q U(q) \end{aligned} \tag{6.1}$$

is solved, resulting in a new set of points q_1 and p_1 in phase-space, as shown in Figure 6.1. The point q_1 is then accepted or rejected as a new sample of the distribution, in typical Metropolis fashion using the acceptance probability defined by

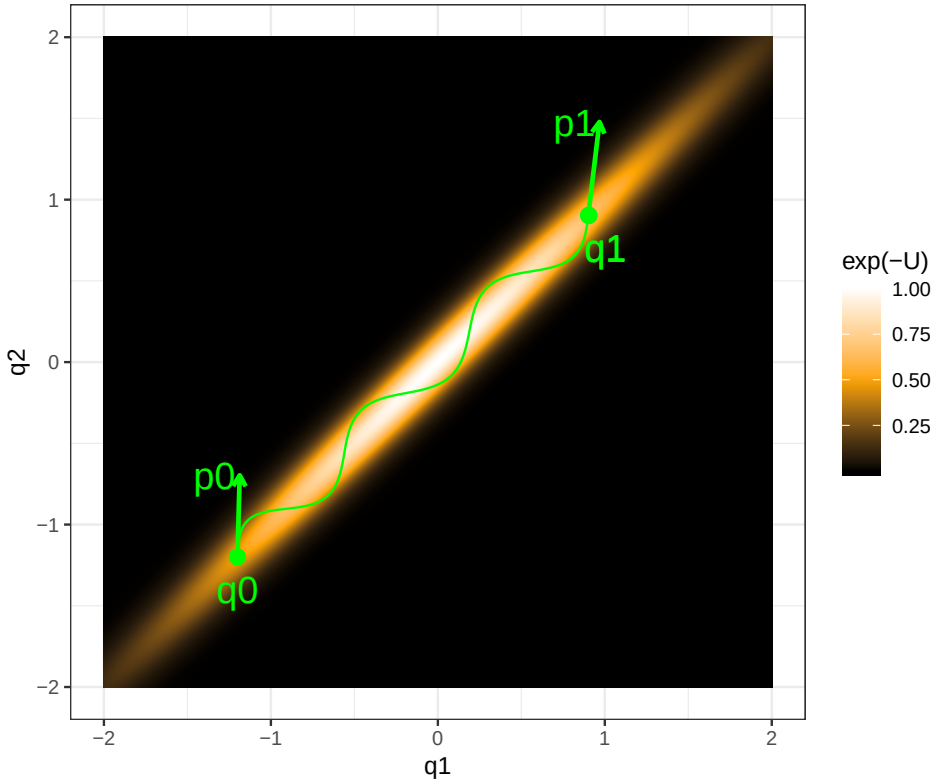

Figure 6.1: Given a starting point q_0, an HMC proposal starts by drawing a momentum, p_0, then simulating Hamiltonian dynamics for a fixed time to reach a new proposal, (q_1, p_1). A Metropolis accept-reject step is then used to accept the new proposal randomly, depending on the difference between the Hamiltonian at the original point in q-p space and the Hamiltonian at the proposed point.

$$\min\left\{1,\ \exp\left(H(q_0,p_0)-H(q_1,p_1)\right)\right\}. \tag{6.2}$$

In classical HMC, the dynamics defined in (6.1) are simulated by applying discrete steps of the leapfrog integrator for T steps. These leapfrog steps are discretized by a stepsize h, yielding

$$\begin{aligned} q_{n+1} &= q_n + hM^{-1}p_n - \frac{h^2}{2}M^{-1}\nabla_q U(q_n) \\ p_{n+1} &= p_n - \frac{h}{2}\nabla_q U(q_n) - \frac{h}{2}\nabla_q U(q_{n+1}). \end{aligned} \tag{6.3}$$

These update equations crucially provide a reversible and volume preserving transition due to the symplectic property of the leapfrog integrator [7]. Moreover, for a satisfactory stepsize the Hamiltonian, $H(q,p)$, remains nearly constant over the numerical simulation of the Hamiltonian dynamics.

6.2.2 Numerical Stability

In practice, the stepsize of the leapfrog integrator is limited by its numerical stability, which is problem-dependent. This concept is easiest to illustrate using a simple system derived from a univariate Gaussian. Specifically, for a univariate Gaussian distribution with mean zero and variance σ^2, the potential energy function used in HMC is $U(q) = q^T\Sigma^{-1}q$. For an identity mass matrix, this leads to the following leapfrog update rule:

$$\begin{aligned} q_{n+1} &= q_n + hp_n - \frac{h^2}{2}\sigma^2 q_n \\ p_{n+1} &= p_n - \frac{h}{2}\sigma^2 q_n - \frac{h}{2}\sigma^2 q_{n+1}, \end{aligned} \tag{6.4}$$

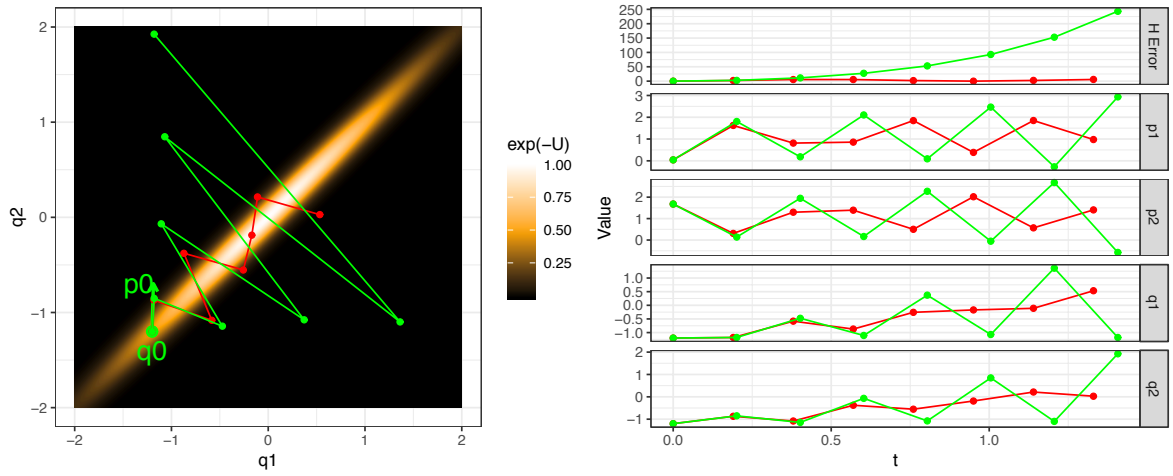

Figure 6.2: For stepsizes larger than the stability threshold of the problem, the leapfrog method goes unstable, leading to wild numerical solutions whose errors characteristically grow after each step (green trajectory). This is in contrast to numerical solutions below the stability threshold, for which the error stays bounded after a series of steps (red trajectory).

which can be compactly written in matrix form as

$$\begin{pmatrix} q_{n+1} \\ p_{n+1} \end{pmatrix} = \begin{pmatrix} 1 - \frac{h^2}{2}\sigma^2 & h \\ -h\sigma^2(1 - \frac{h^2}{4}\sigma^2) & 1 - \frac{h^2}{2}\sigma^2 \end{pmatrix} \begin{pmatrix} q_n \\ p_n \end{pmatrix}. \tag{6.5}$$

The update matrix in (6.5) describes how leapfrog advances forward one step, for a univariate Gaussian system with mass one. Because the leapfrog method is symplectic, the determinant of this matrix is always one. However, for $h < 2/\sigma$ the eigenvalues are complex and have modulus equal to one, while for $h > 2/\sigma$ they are real with one of the eigenvalues having modulus greater than one [39, Ch. 2.6]. The latter case results in numerical instability of the integrator [39]. Intuitively, this means that any small numerical errors that will inevitably arise in the numerical solutions (q_n, p_n) will be successively "magnified" by each application of the leapfrog update matrix, quickly resulting in a solution with unacceptably large error (Figure 6.2).

The univariate analysis can be readily extended to a multivariate Gaussian of dimen-

sion D with covariance matrix Σ. With a mass matrix equal to the identity, this results in the following Hamiltonian system:

$$\begin{aligned} q' &= p \\ p' &= \Sigma^{-1} q. \end{aligned} \tag{6.6}$$

In general, Σ^{-1} may have off-diagonal correlation terms that make this derived system coupled. However, the transformation defined by $u := V^{-1}q$ and $w := V^{-1}p$, where V is a matrix whose columns consist of the eigenvectors of Σ^{-1}, essentially uncouples the differential equations, yielding

$$\begin{aligned} u' &= w \\ w' &= \Lambda u, \end{aligned} \tag{6.7}$$

where Λ is a diagonal matrix composed of the eigenvalues, $\lambda_1, \cdots, \lambda_D$, of Σ^{-1} [39, Ch. 2.6]. This decoupling transformation can be thought of as reparameterizing the problem so that the potential energy surface exhibits no correlation (Figure 6.3).

Applying the leapfrog update rule (6.3) to the uncoupled equations (6.7) elucidates how the issue of numerical stability arises in the case of a multivariate Gaussian. Specifically, if one or more of the eigenvalues of Σ^{-1} do not satisfy the inequality $h < 2/\sqrt{\lambda_i}$, then the leapfrog update matrix will have a real eigenvalue with modulus greater than one, and instability in the numerical integration will arise. In practice this will lead to poor acceptance rates in the Markov chain as well as to divergences [67]. Formally, for leapfrog on a multivariate Gaussian with mass matrix I the stepsize h must satisfy the

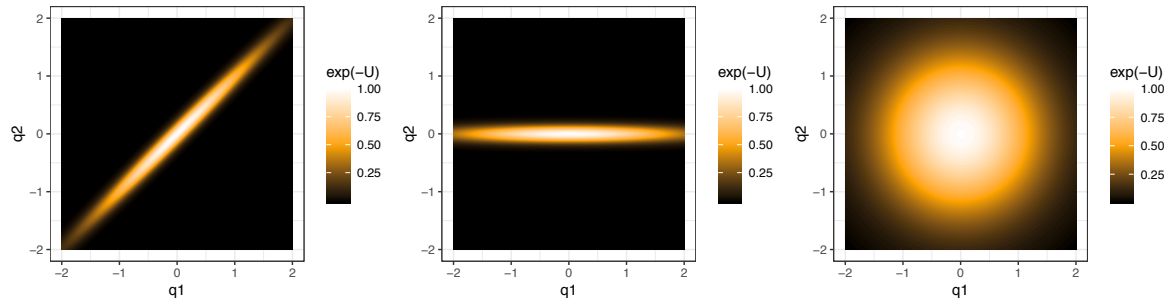

Figure 6.3: A decoupling transformation is equivalent to transforming a correlated distribution (left) to an uncorrelated one (middle). Similarly, a transformation can be used to make an arbitrary Gaussian isotropic (right).

inequality

$$h < 2/\sqrt{\rho(\Sigma^{-1})} \tag{6.8}$$

to ensure stability (here $\rho(\Sigma^{-1})$ denotes the spectral radius of Σ^{-1}, defined as the maximum of the absolute value of its eigenvalues).

This condition can severely limit the timestep of leapfrog. Intuitively, the dimensions of the distribution that exhibit high variance, and thus low curvature in the potential energy surface, will have slowly oscillating Hamiltonian trajectories that are very smooth and can be integrated by leapfrog with a reasonable stepsize. Meanwhile, dimensions with low variance, and thus high curvature, will lead to rapdily oscillating solutions that require a very small stepsize. Thus a system that has even one state with a dramatically smaller variance than the others will force leapfrog to take excessively small steps on the whole system. An ODE system that exhibits this sort of behavior is known as a multiscale system [74]. In practice, the telltale sign of a distribution that will lead to a multiscale Hamiltonian system in HMC is the inverse covariance matrix, Σ^{-1} having a large condition number. The condition number $\kappa(\Sigma^{-1})$ is defined as the ratio of the largest eigenvalue of Σ^{-1} to the smallest. Intuitively, it captures the ratio of the curvatures of the

dimensions of the potential energy surface. Practically speaking, a large condition number can arise either when an uncorrelated Gaussian has a large disparity in the variance of its largest and smallest dimensions, or alternatively when a multivariate Gaussian contains high correlations between dimensions. In engineering, the multiscale problem is often resolved by using an appropriate implicit integrator, which is typically able to take much larger steps on multiscale problems while preserving the accuracy of desired quantities of the system [73, 75]. We describe this approach and its application to HMC in the following section.

6.2.3 Nonlinear Posterior Geometry

While the stability analysis considered so far has been only for Gaussian distributions which result in linear Hamiltonian systems, most practical distributions being sampled by HMC are not Gaussian and thus result in nonlinear Hamiltonian systems. For nonlinear systems, one can locally identify a multiscale system by analyzing the condition number of the local Hessian of the potential energy surface, $\nabla_{qq}U(q)$. This local Hessian is equivalent to the inverse covariance matrix of the local Laplace approximation to the posterior [1, Chapter 4]. Thus the key difference between the posterior geometry of a Gaussian distribution and that of a more complicated distribution is that the former has a potential energy surface with a constant Hessian while the latter may contain vastly different local Hessians throughout the surface, which can lead to different multiscale properties of the associated Hamiltonian system (Figure 6.4). In practice, this means that different leapfrog stepsizes may be required, depending on where on the potential energy surface the sampler is currently located. Roughly speaking, RMHMC uses local Hessian evaluations of the potential energy surface to adaptively change this stepsize based on the local curvature [72].

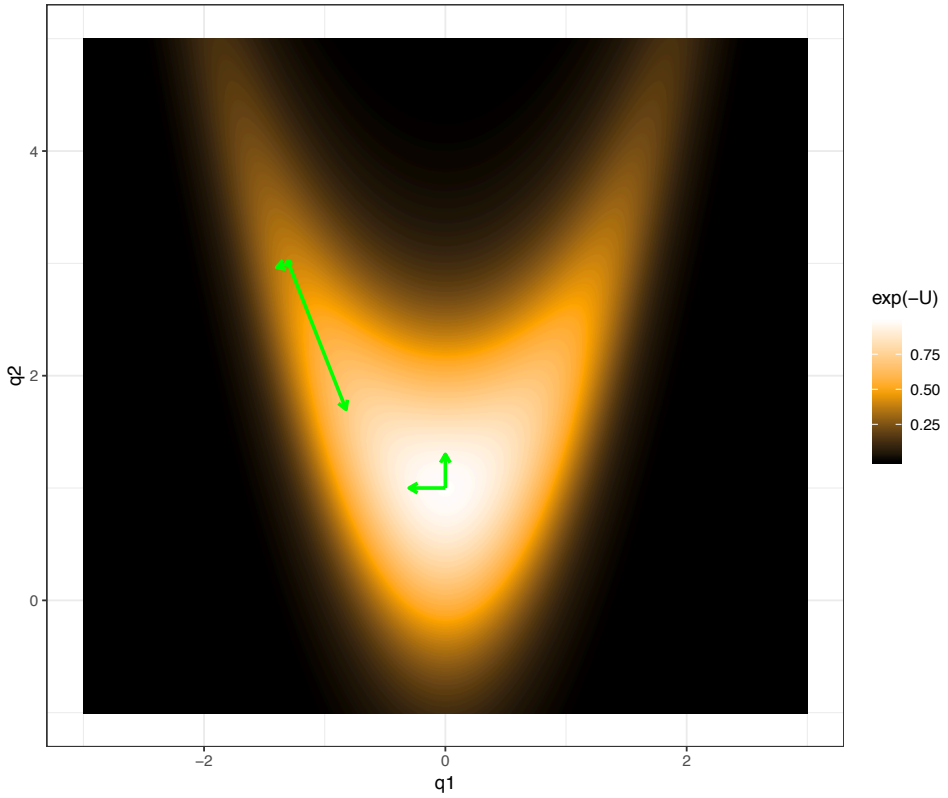

Figure 6.4: Unlike a Gaussian distribution, arbitrary non-Gaussian distributions have non-constant local Hessians (here characterized by the green eigenvectors of the local Hessian multiplied by their associated eigenvalues) that affect the oscillatory frequency of HMC trajectories and thus an acceptable leapfrog stepsize.

6.2.4 The Mass Matrix as a Reparameterization

Although a multivariate Gaussian system exhibiting multiple scales can significantly hinder the efficiency of the leapfrog method, this deficiency may be resolved by appropriately selecting the mass matrix M in (6.1). In fact, utilizing a mass matrix in HMC that is equal to the covariance Σ of the Gaussian is equivalent to reparameterizing the problem to an equivalent isotropic Gaussian [7] (Figure 6.3). From a numerical analysis perspective, M^{-1} can be viewed as a preconditioner that transforms the problem so as to give Σ^{-1} a better condition number $\kappa(\Sigma^{-1})$, i.e. it transforms the problem so that there is less discrepency between the largest and smallest eigenvalues of Σ^{-1}.

While a conveniently selected mass matrix can effectively eliminate the multiscale problem of leapfrog for Gaussian distributions, for distributions with more complicated geometry whose local curvature varies, a constant mass matrix is inherently much less effective in accounting for multiscale geometry. Splitting methods such as that of [69] as well as [70] which can handle high curvature and multiple scales by separating out a constant Gaussian approximation of the distribution unfortunately have the same limitation, as they cannot handle the varying curvature of non-Gaussian posteriors.

6.3 Implicit HMC

The stability bottleneck placed on leapfrog by a multiscale system is characteristic of explicit integrators like leapfrog. In the ODE community, implicit integrators have been used to essentially "skip" fast oscillations while accurately evolving in time quantities of interest in the system [73, 75]. We describe the implicit midpoint integrator: a symplectic alternative to leapfrog that is of the same order of accuracy as leapfrog, but is implicit, allowing it to take much bigger timesteps on multiscale problems than what would otherwise be possible with leapfrog. Unlike the approaches of [69] and [70],

the implicit midpoint integrator is applicable to arbitrary multiscale systems, not just Gaussian ones.

We explain how, unlike leapfrog, the implicit midpoint method has no stability limit on a linear system. We then describe a custom Newton-Krylov method for the solution of the nonlinear system that must be solved at each timestep by midpoint. Finally, we point out that while the implicit midpoint method is unconditionally stable on a linear system, in practice it can go unstable on nonlinear problems [73], although at a much larger stepsize than leapfrog is able to take. We discuss the practical implications of this and aspects of choosing between leapfrog and implicit midpoint for a specific problem, and we give practical guidelines for selecting a stepsize for both methods.

6.3.1 Implicit Midpoint and Stability

For general Hamiltonian systems of the form in equation 6.1, the timesteps of implicit midpoint are defined by

$$\begin{aligned} q_{n+1} &= q_n + hM^{-1}\left(\frac{p_n + p_{n+1}}{2}\right) \\ p_{n+1} &= p_n - h\nabla_q U\left(\frac{q_n + q_{n+1}}{2}\right). \end{aligned} \tag{6.9}$$

Because the point (q_{n+1}, p_{n+1}) is defined implicitly, as opposed to in leapfrog where an explicitly computable update equation is given, one must resort to either functional iteration or Newton's method to compute its value. However, in situations where implicit midpoint would be desirable over leapfrog, Newton's method or a more robust modification of it is typically preferred [74, ch. 3.4.2]. We elaborate on this point in the subsequent subsection.

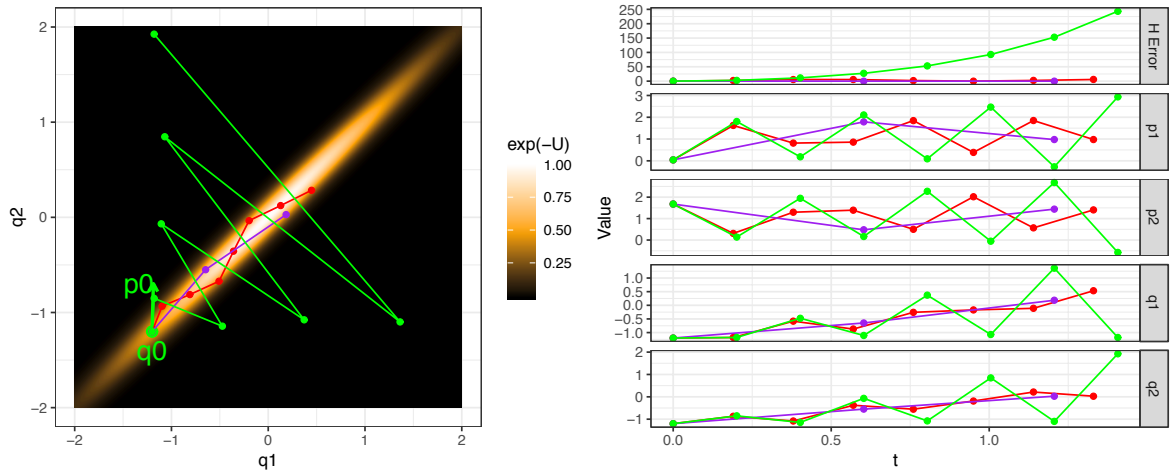

Figure 6.5: Unlike the leapfrog integrator which goes unstable for large enough stepsizes (green trajectory), the implicit midpoint integrator can remain stable at the same stepsize (red trajectory) and even at stepsizes much larger than the stability threshold for leapfrog (purple trajectory).

An analaysis that is similar to the one performed for leapfrog in Section 6.2.2 shows that unlike leapfrog, the implicit midpoint method does not have a stability limit on the linear system that arise from HMC sampling of a Gaussian distribution. In fact, like the update matrix for leapfrog, the update matrix for midpoint on a Gaussian distribution always has determinant one, however it can be shown that its eigenvalues are complex for any stepsize, h (see the Appendix for a full analysis of the stability of implicit midpoint for linear Hamiltonian systems). In other words, the implicit midpoint method is stable on Gaussian systems for any stepsize, i.e. it has not stability limit (Figure 6.5).

6.3.2 Fast Solution of the Nonlinear System

As previously mentioned, (6.9) must typically be solved for $(q_{n+1}\, p_{n+1})$ numerically. In practice this can be done by applying Newton's method to solve equation 6.9, however this can be costly for large systems. First, because the first D equations in 6.9 are linear, one can substitute the first D equations into the second D to obtain the nonlinear system

6.10 with D equations and the D unknowns, p_{n+1}. Once p_{n+1} is obtained, q_{n+1} can be obtained by simply plugging the result back into the first set of equations, yielding

$$p_{n+1} = p_n - h\nabla_q U\left(q_n + \frac{h}{4}M^{-1}(p_n + p_{n+1})\right). \tag{6.10}$$

Renaming p_{n+1} to x and noting that p_n is known from the previous timestep, equation (10) can be written as

$$g(x) := x - p_n + h\nabla_q U\left(q_n + \frac{h}{4}M^{-1}(p_n + x)\right). \tag{6.11}$$

Newton's method applied to this system consists of starting with an initial guess, x_0, and iteratively obtaining approximate solutions $x_1, x_2, \cdots$ using the following steps until a convergence criteria is reached:

1. Solve the linear system $J_g(x_n)\delta = -g(x_n)$
2. Set $x_{n+1} = x_n + \alpha\delta$,

where for the standard Newton iteration $\alpha = 1$. Here, $J_g(x_n)$ refers to the Jacobian of g evaluated at x_n and is given by

$$J_g(x) := I + \frac{h^2}{4}\nabla_{qq} U\left(q_n + \frac{h}{4}M^{-1}(p_n + x)\right)M^{-1}. \tag{6.12}$$

For large systems, exploitation of the structure of the system is critical for both efficiency and robustness. We introduce the Newton-Krylov method, which is particularly well-suited to this class of problems.

Newton-Krylov Methods

In the classic Newton method, the linear solve step requires explicitly computing $J_g(x_n)$ and carrying out a full linear solve of the equation $J_g(x_n)\delta = -g(x_n)$ for δ at each

Newton iteration. This requires evaluating the Hessian $\nabla_{qq}U(q)$ of the potential energy, which scales as $\mathcal{O}(D^2)$. A Newton-Krylov method [76] circumvents this expensive linear solve that occurs at every Newton iteration by replacing the linear solve with an approximate linear solve that is much cheaper to compute. Specifically, the Newton-Krylov method replaces the solution δ with an approximate solution $\tilde{\delta}$ such that $\|J_g(x_n)\tilde{\delta}+g(x_n)\|$ is less than some tolerance, η_n, otherwise known as a "forcing term". This forcing term is typically selected using a scheduling criteria that satisfies certain theoretical properties. For our experiments we found the most success with method 2.1 of [77], as it did not rely on manually selected "tuning parameters".

In our iNUTS implementation, we use the GMRES solver [78] to compute the approximate linear solves within the Newton-Krylov iterations. The GMRES algorithm works by iteratively computing approximate solutions to the linear system $J_g(x_n)\delta = -g(x_n)$ that reduce the norm $\|J_g(x_n)\delta + g(x_n)\|$, while needing only to evaluate the product of the matrix $J_g(x_n)$ with an arbitrary vector v.

Needing only to evaluate the product of the matrix $J_g(x_n)$ with an arbitrary vector v, rather than computing $J_g(x_n)$ and then computing its product with v, is particularly advantageous in HMC for two reasons. First, the Jacobian vector product $J_g(x_n) \cdot v$ can be evaluated using only a Hessian-vector product of the potential energy which, in an efficient automatic differentiation implementation such as the one available in Stan [11], scales as $\mathcal{O}(D)$ as opposed to computing a full Hessian, which scales as $\mathcal{O}(D^2)$. Second, the iterated solutions in a Krylov-based solver are known to converge faster when the matrix $J_g(x_n)$ has "clusters" of eigenvalues that are close together [79], as is the case when the system being solved comes from the posterior of a Bayesian model, particularly a multilevel Bayesian model. Specifically, in the case of implicit midpoint applied to HMC on the posterior of a Bayesian multilevel model, the matrix (6.12) will have clusters of eigenvalues that are all of similar order and correspond to the units in a multilevel model

that are are all at the same level. For example, for the famous "eight schools" model in [1], there is a variance parameter τ that is at the scale of 1, and eight separate school-level parameters at the scale of 10. Although there are nine parameters total, and thus a nine-by-nine system to solve, in practice the Newton-Krylov method can typically solve the linear system to numerical precision in only two iterations, because there are only two "clusters" of eigenvalues in the Hessian: the cluster at the scale of 1 and the cluster at the scale of 10.

Using a Line Search

The second modification we make to the classic Newton method is to change the steplength α of the Newton iteration to ensure that consecutive iterations of the Newton-Krylov method effectively reduce the residual error in the nonlinear system. In particular, we use a geometric line-search to continually halve the size of α until the new solution satisfies the Armijo-Goldstein condition [64].

Choosing an Initial Guess

While not a modification of Newton's method per se, in our iNUTS implementation we also use a unique method of setting the initial guess x_0 to the nonlinear solver. In practice, this greatly improves the number of Newton-Krylov iterations needed for convergence. In particular, we exploit the numerical properties of the implicit midpoint integrator on multiscale systems, to obtain a good initial guess. Specifically, when the stepsize h of the numerical integrator is small relative to the local frequency of a particular oscillating momentum coordinate $p^{(i)}$, the previous momentum, $p_n^{(i)}$, will serve as a good initial guess to $p_{n+1}^{(i)}$ in the system 6.10, as the numerical solution will not be changing much between consecutive steps. Similarly, the second to last momentum $p_{n-1}^{(i)}$ will also serve as a good initial guess, although not quite as good as the last value. On the other hand, when h is

much larger than the frequency of a particular oscillating variable $p^{(i)}$, which will be the case in a multiscale system, the numerical solution will approximately "alternate" about zero taking on the values $p_1^{(i)} = \gamma, p_2^{(i)} = -\gamma, p_3^{(i)} = \gamma, \cdots$. Thus in this case, the second to last momentum serves as a good initial guess. Thus we use p_{n-1} as an initial guess overall for the nonlinear equation solver.

6.3.3 Nonlinear Stability Limit and Choosing an Integrator and a Stepsize

While the implicit midpoint method is unconditionally stable for any stepsize h on a linear system, it can and will exhibit instability for certain nonlinear systems, although typically at a stepsize that is much larger than the stepsize at which leapfrog would go unstable on the same problem [73]. In practice, these instabilities can typically be identified by observing large growth in the Hamiltonian of the numerical numerical trajectory, or when Newton-Krylov iterations fail to converge. To find an appropriate stepsize for implicit midpoint in practice, we recommend running a sampler "warmup" period where the stepsize h is reduced by some factor like $1/2$ until these pathological behaviors are eliminated. This is akin to the current warmup method of Stan, where the stepsize of the integrator is reduced until an acceptable accept rate is reached [11].

To select between implicit midpoint and leapfrog on a specific problem, we suggest starting with implicit midpoint and periodically calculating an approximation to the largest eigenvalue of the local Hessian of the potential energy function during a warmup period. This eigenvalue approximation can be efficiently computed with only a few Hessian-vector products using the power method of numerical linear algebra [79]. This can in turn be used to get an approximation of the largest stepsize that leapfrog could take while maintaining stability, via equation 6.8. When the smallest of these approxi-

mate stepsizes divided by two is close to the implicit midpoint stepsize needed to maintain stability divided by the average number of gradient evaluations plus Hessian-vector products required by implicit midpoint, then the leapfrog method will be more efficient for the problem. This is because the leapfrog method requires two gradient evaluations per step, and a Hessian-vector product, like a gradient evaluation, scales linearly in the number of inputs to an automatic differentiation system such as the one used in Stan. In practice, we found that computation of the Hessian-vector product took 1.2-1.3 times the cost of a gradient evaluation in Stan.

6.4 Experiments

We illustrate on several examples how implicit midpoint can achieve superior efficiency over leapfrog on multiscale problems. For all of our examples we compare using a custom implementation of NUTS that either uses leapfrog (lfNUTS) or implicit midpoint (iNUTS). Our code is freely available as an R package, and at the time of writing can be obtained by emailing the authors.

6.4.1 2-D Gaussian

As discussed in Section 6.2.2, the leapfrog stability limit for a multivariate Gaussian is directly related to the largest eigenvalue of the inverse covariance matrix. One of the ways this eigenvalue can be large is when the multivariate Gaussian is highly correlated. To illustrate how this affects NUTS in practice, we compared the average tree depth per sample of lfNUTS and iNUTS for varying levels of ρ on the two-dimensional Gaussian

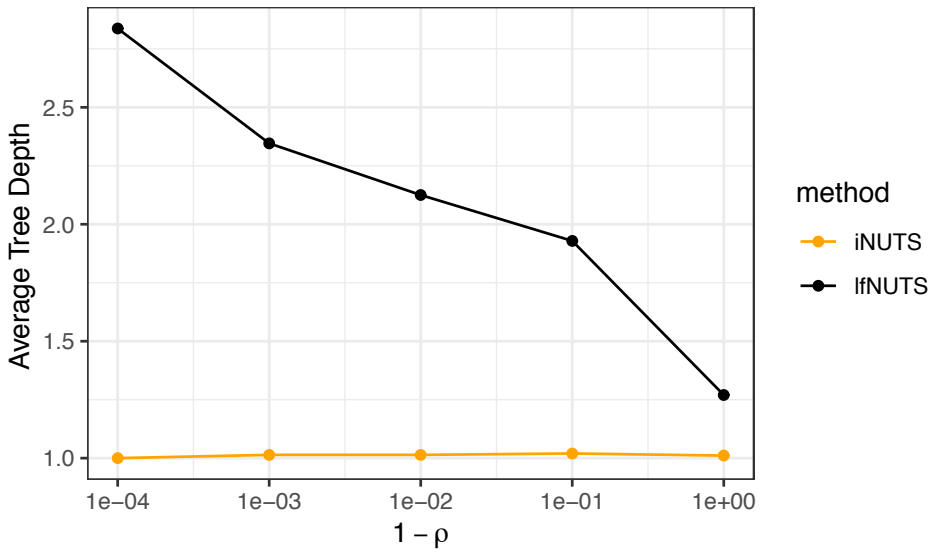

Figure 6.6: For highly correlated Gaussian distributions, the leapfrog integrator in lfNUTS is forced to take tiny stepsizes that result in larger average NUTS tree depths. As the correlation goes to zero, the average tree depth approaches that of iNUTS.

distribution with mean zero, and covariance matrix

$$\Sigma = \begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix}. \tag{6.13}$$

The empirical scaling of average tree depth over 1,000 NUTS samples versus $1 - \rho$ is shown in Figure 6.6. In the NUTS algorithm, tree depth is selected automatically, when a Hamiltonian trajectory has "U-turned". Two to the power of the tree depth represents how many steps in the trajectory were computed. For more correlated distributions, leapfrog is forced to take smaller stepsizes, which results in more steps having to be taken and thus larger tree depths. Note that for a Gaussian system, implicit midpoint has no stepsize limit for stability regardless of the covariance of the Gaussian. Thus the stepsize can be chosen large enough so that a tree depth of only one is required.

6.4.2 Banana Distribution

To illustrate how the implicit midpoint integrator is advantageous for distributions with nonlinear correlations that cannot be addressed by a mass matrix, we compared lfNUTS and iNUTS on a highly-correlated banana distribution [80] with parameters $a = 1$ and $b = 100$ whose probability density function (PDF) takes the following form:

$$\begin{aligned} & p(q_1, q_2) \\ & = \frac{1}{2\pi} \exp\left\{-\frac{1}{2}\left[a^2 q_1^2 + \frac{q_2 - ab(a^2 q_1^2 + a^2)}{a^2}\right]\right\}. \end{aligned} \tag{6.14}$$

The highly-nonlinear correlation structure of this distribution is typical for complicated non-Gaussian distributions. For HMC, the long direction across the length of the banana compared to its narrow width creates a prototypical multiscale problem, forcing leapfrog to take an excessively small stepsize that leads to insufficient exploration of the posterior in lfNUTS compared to iNUTS (Figure 6.7).

A summary of results comparing the two methods is presented in Table 1. Because the implicit midpoint steps within iNUTS require a nonlinear solve, there is much more overhead in each step compared to leapfrog. However, because implicit midpoint has a much larger stability limit than leapfrog, iNUTS is able to take a much larger step, leading to more efficient samples. In practice, iNUTS ends up needing to do only a third of the work as lfNUTS to obtain an effective sample. In terms of computer time, iNUTS took about 30% longer to obtain over ten times as many effective samples.

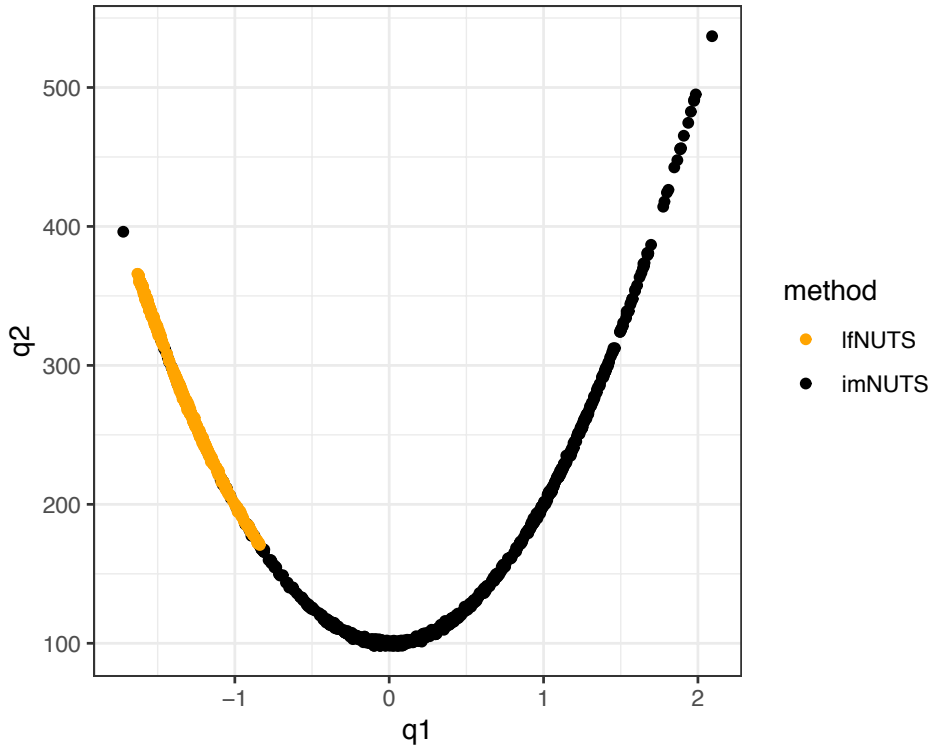

Figure 6.7: 1,000 samples of the banana distribution from lfNUTS and iNUTS. Because of the multiscale nature of the Hamiltonian system that arises when sampling the banana distribution, lfNUTS is forced to use an unreasonably small stepsize that nearly reduces its behavior to that of a random walk and renders it unable to explore the entire distribution (orange samples). Meanwhile, iNUTS can effectively take ten times as large of a stepsize which allows it to efficiently explore the entire posterior.

	lfNUTS	iNUTS
avg. grad evals/step	2.00	11.98
avg. hess-vec evals/step	0.00	9.88
avg. work/step	2.00	21.86
avg. tree depth	9.97	5.40
avg. effective samples	42.47	613.26
work/effective sample	47.09	16.11
total computer time (s)	181	231

Table 6.1: A comparison of the computation involved in the sampling experiment. Average gradient evaluations and Hessian-vector evaluations are per step of the integrator, and work is the sum of these. Average effective samples are over all 11 dimensions being sampled. While iNUTS requires more work per step, because it is able to take a much larger step than leapfrog, it can obtain many more effective samples in a similar time frame.

6.5 Discussion

We have shown that distributions exhibiting multiple scales limit the stepsize of the leapfrog integrator in HMC, and how this limitation can be effectively circumvented by utilizing the implicit midpoint integrator together with Newton-Krylov iteration. Furthermore, we offered a practical implementation of implicit midpoint that is applicable to Bayesian posterior sampling problems, and provided practical guidelines for choosing which integrator to use, as well as the stepsize to use in the integrator. As illustrated in our examples, using implicit midpoint together with Newton-Krylov instead of leapfrog can provide a practical and significant efficiency boost in the context of NUTS.

Chapter 7

Conclusion

We have described several novel models and computational methods for Bayesian hierarchical modeling of biomedical data. We described various contributions to computational Bayesian modeling, along with the problems from our applied work that motivated these contributions.

First we describe our hierarchical disease progression model, for modeling patient disease trajectories. We then described the intricacies of the nonlinear trajectories in the model and how we applied it to the progression of dementia in Alzheimer's disease. Next we described our work on the Givens Representation of orthogonal matrices to infer models with orthogonal matrix parameters. We described the innovations in our method along with our motivating hierarchical problem based on the analysis of protein biomarkers of coagulopathic trauma patients. Next, we described a mechanistic model of coagulopathy that relates clotting assay data to protein concentrations, effectively providing a fast and convenient way for clinicians to understand key protein markers involves in clotting.

Lastly we described our Implicit Hamiltonian Monte Carlo (HMC) method and illustrated the connection between multiscale posterior distributions and the efficiency of

HMC. We demonstrated the practical efficiency of our approach on several examples.

Appendix A

Deriving the Change of Measure Term in the Givens Representation

We start with the determinant of the matrix form of the change of measure term from Expression 3.10 (reproduced below):

$$\begin{pmatrix} G_{2:n}^T J_{Y_1(\Theta)}(\Theta) \\ G_{3:n}^T J_{Y_2(\Theta)}(\Theta) \\ \vdots \\ G_{p:n}^T J_{Y_p(\Theta)}(\Theta) \end{pmatrix} \tag{A.1}$$

For $l = 1, \cdots, n$, let us define the following shorthand notation

$$\partial_{i,i+l} Y_k := \frac{\partial}{\partial \theta_{i,i+l}} Y_k \tag{A.2}$$

and

$$\partial_i Y_k := \begin{pmatrix} \partial_{i,i+1} Y_k & \partial_{i,i+2} Y_k & \cdots & \partial_{in} Y_k. \end{pmatrix} \tag{A.3}$$

In the new notation Equation can be written in the following block matrix form:

$$\begin{pmatrix} G_{2:n}^T \partial_1 Y_1 & G_{2:n}^T \partial_2 Y_1 & \cdots & G_{2:n}^T \partial_p Y_1 \\ G_{3:n}^T \partial_1 Y_2 & G_{3:n}^T \partial_2 Y_2 & \cdots & G_{3:n}^T \partial_p Y_2 \\ \vdots & \vdots & \ddots & \vdots \\ G_{p:n}^T \partial_1 Y_p & G_{p:n}^T \partial_2 Y_p & \cdots & G_{p:n}^T \partial_p Y_p \end{pmatrix}. \tag{A.4}$$

First note that the block matrices above the diagonal are all zero. This can be seen by noting that the rotations in the Givens representation involving elements greater than i will not affect e_i, i.e. letting $R_i := R_{i,i+1} \cdots R_{in}$,

$$Y_i = R_1 R_2 \cdots R_p e_i = R_1 \cdots R_i e_i. \tag{A.5}$$

Thus for $j > i$, $\partial_j Y_i = 0$ and the determinant of Expression A.4 simplifies to the product of the determinant of the matrices on the diagonal i.e. the following expression:

$$\prod_{i=1}^{p} \det \left(G_{i+1:n}^T \partial_i Y_i \right). \tag{A.6}$$

A.1 Simplifying Diagonal Block Terms

Let I_i denote the first i columns of the $n \times n$ identity matrix and let I_{-i} represent the last $n - i$ columns. The term $G_{i+1:n}^T$ in Expression A.6 can be written as

$$G_{i+1:n}^T = I_{-i}^T G^T = I_{-i}^T R_p^T \cdots R_1^T. \tag{A.7}$$

To simplify the diagonal block determinant terms in Expression A.6 we take advantage of the following fact:

$$\det\left(G_{i+1:n}^T \partial_i Y_i\right) = \det\left(I_{-i}^T R_p^T \cdots R_1^T\right) = \det\left(I_{-i}^T R_i^T \cdots R_1^T \partial_i Y_i\right). \tag{A.8}$$

In other words, the terms $R_p^T \cdots R_{i+1}^T$ have no effect on the determinant. This can be shown by first separating terms so that

$$\begin{aligned} \det\left(G_{i+1:n}^T \partial_i Y_i\right) &= \det\left(\underbrace{I_{-i}^T}_{(n-i)\times n} R_p^T \cdots R_1^T \underbrace{\partial_i Y_i}_{n\times(n-i)}\right) && \text{(A.9)} \\ &= \det\left(I_{-i}^T \left[R_p^T \cdots R_{i+1}^T\right] \left[R_i^T \cdots R_1^T \partial_i Y_i\right]\right), && \text{(A.10)} \end{aligned}$$

and then noticing that $R_{i+1} \cdots R_p$ only effects the first i columns of the identity matrix, so that

$$I_{-i}^T \left[R_p^T \cdots R_{i+1}^T\right] = \left(R_{i+1} \cdots R_p \, I_{-i}\right)^T = \left(I_{-i}\right)^T. \tag{A.11}$$

Thus, Expression A.6 is equivalent to

$$\prod_{i=1}^{p} \det\left(I_{-i}^T R_i^T \cdots R_1^T \partial_i Y_i\right). \tag{A.12}$$

Now consider the k, l element of the $(n-i)\times(n-i)$ block matrix $I_{-i}^T R_i^T \cdots R_1^T \partial_i Y_i$. This can be written as

$$\begin{aligned} e_{i+k}^T R_i^T \cdots R_1^T \partial_{i,i+l} Y_i &= e_{i+k}^T R_i^T \cdots R_1^T \partial_{i,i+l} (R_1 \cdots R_i e_i) \\ &= e_{i+k}^T R_i^T \cdots R_1^T R_1 \cdots R_{i-1} (\partial_{i,i+l} R_i e_i) \\ &= e_{i+k}^T R_i^T (\partial_{i,i+l} R_i e_i). \end{aligned} \tag{A.13}$$

Since $e_{i+k}^T R_i^T R_i e_i = 0$, taking the derivatives of both sides and applying the product rule yields

$$\begin{aligned} & \partial_{i,i+l}(e_{i+k}^T R_i^T R_i e_i) = \partial_{i,i+l} 0 \\ \Rightarrow \quad & (\partial_{i,i+l} e_{i+k}^T R_i^T) R_i e_i + e_{i+k}^T R_i^T (\partial_{i,i+l} R_i e_i) = 0 \\ \Rightarrow \quad & e_{i+k}^T R_i^T (\partial_{i,i+l} R_i e_i) = -(\partial_{i,i+l} e_{i+k}^T R_i^T) R_i e_i. \end{aligned} \tag{A.14}$$

Combining (A.14) with A.13, the expression for the k, l element of $I_{-i}^T R_i^T \cdots R_1^T \partial_i Y_i$ becomes $-(\partial_{i,i+l} e_{i+k}^T R_i^T) R_i e_i$.

However, note that

$$e_{i+k}^T R_i^T = e_{i+k}^T R_{in}^T \cdots R_{i,i+1}^T = e_{i+k}^T R_{i,i+k}^T \cdots R_{i,i+1}^T, \tag{A.15}$$

and that the partial derivative of this expression with respect to $i, i+l$ is zero when $k > l$. Thus it is apparent that $I_{-i}^T R_i^T \cdots R_1^T \partial_i Y_i$ contains zeros above the diagonal and that $\det\left(I_{-i}^T R_i^T \cdots R_1^T \partial_i Y_i\right)$ is simply the product of the diagonal elements of the matrix.

A.2 Diagonal Elements of the Block Matrices

To obtain the diagonal terms of the block matrices we directly compute $-\partial_{i,i+l} e_{i+k}^T R_i^T$ for $l = k$, $R_i e_i$, and their inner-product. Defining $D_{ij} := \partial_{ij} R_{ij}$,

$$
\begin{aligned}
-\partial_{i,i+k} R_i e_{i+k} &= -\partial_{i,i+k}(R_{i,i+1} \cdots R_{i,i+k} e_{i+k}) && \text{(A.16)} \\
&= -R_{i,i+1} \cdots R_{i,i+k-1} D_{i,i+k} e_{i+k} && \text{(A.17)} \\
& && \text{(A.18)} \\
&= R_{i,i+1} \cdots R_{i,i+k-1} \begin{pmatrix} 0 \\ \vdots \\ 0 \\ \cos\theta_{i,i+k} \\ 0 \\ \vdots \\ 0 \\ \sin\theta_{i,i+k} \\ 0 \\ \vdots \\ 0 \end{pmatrix} && \text{(A.19)}
\end{aligned}
$$

$$= R_{i,i+1} \cdots R_{i,i+k-2} \begin{pmatrix} 0 \\ \vdots \\ 0 \\ \cos\theta_{i,i+k-1} \cos\theta_{i,i+k} \\ 0 \\ \vdots \\ 0 \\ \sin\theta_{i,i+k-1} \cos\theta_{i,i+k} \\ \sin\theta_{i,i+k} \\ 0 \\ \vdots \\ 0 \end{pmatrix} \tag{A.20}$$

$$= \begin{pmatrix} 0 \\ \vdots \\ 0 \\ \cos\theta_{i,i+1} \cos\theta_{i,i+2} \cdots \cos\theta_{i,i+k-1} \cos\theta_{i,i+k} \\ \sin\theta_{i,i+1} \cos\theta_{i,i+2} \cdots \cos\theta_{i,i+k-1} \cos\theta_{i,i+k} \\ \vdots \\ \sin\theta_{i,i+k-1} \cos\theta_{i,i+k} \\ \sin\theta_{i,i+k} \\ 0 \\ \vdots \\ 0 \end{pmatrix} \tag{A.21}$$

which is zero up to the ith spot and after the $i+k$th spot. Note that the rotation matrix R_i applied to the standard basis vector e_i can be written as follows:

$$\begin{aligned} R_i e_i &= R_{i,i+1} \cdots R_{in} e_i && \text{(A.22)} \\ & && \text{(A.23)} \\ &= \begin{pmatrix} 0 \\ \vdots \\ 0 \\ \cos\theta_{i,i+1}\cos\theta_{i,i+2}\cdots\cos\theta_{i,n-1}\cos\theta_{in} \\ \sin\theta_{i,i+1}\cos\theta_{i,i+2}\cdots\cos\theta_{i,n-1}\cos\theta_{in} \\ \vdots \\ \sin\theta_{i,n-1}\cos\theta_{in} \\ \sin\theta_{in} \end{pmatrix}. && \text{(A.24)} \end{aligned}$$

Finally, directly computing the inner-product of $-\partial_{i,i+l} e_{i+k}^T R_i^T$ and $R_i e_i$ we obtain

$$\begin{aligned} -(\partial_{i,i+l} e_{i+k}^T R_i^T)(R_i e_i) &= \cos^2\theta_{i,i+1}\cos^2\theta_{i,i+2}\cdots\cos^2\theta_{i,i+k}\cos\theta_{i,i+k+1}\cdots\cos\theta_{in} \\ &+ \sin^2\theta_{i,i+1}\cos^2\theta_{i,i+2}\cdots\cos^2\theta_{i,i+k}\cos\theta_{i,i+k+1}\cdots\cos\theta_{in} \\ &+ \sin^2\theta_{i,i+2}\cos^2\theta_{i,i+3}\cdots\cos^2\theta_{i,i+k}\cos\theta_{i,i+k+1}\cdots\cos\theta_{in} \\ &\vdots \\ &+ \sin^2\theta_{i,i+k}\cos\theta_{i,i+k+1}\cdots\cos\theta_{in} \end{aligned}$$

$$
\begin{aligned}
&= \cos^2\theta_{i,i+2}\cos^2\theta_{i,i+3}\cdots\cos^2\theta_{i,i+k}\cos\theta_{i,i+k+1}\cdots\cos\theta_{in} \\
&+ \sin^2\theta_{i,i+2}\cos^2\theta_{i,i+3}\cdots\cos^2\theta_{i,i+k}\cos\theta_{i,i+k+1}\cdots\cos\theta_{in} \\
&\vdots \\
&+ \sin^2\theta_{i,i+k}\cos\theta_{i,i+k+1}\cdots\cos\theta_{in} \\
&= \cdots \\
&= \cos\theta_{i,i+k+1}\cdots\cos\theta_{in} \\
&= \prod_{k=i+1}^{n}\cos\theta_{ik}. \qquad \text{(A.25)}
\end{aligned}
$$

Thus the determinant of the entire block matrix $I_{-i}^T R_i^T \cdots R_1^T \partial_i Y_i$ simplifies to

$$
\prod_{k=i+1}^{n}\left(\prod_{j=k+1}^{n}\cos\theta_{ik}\right) = \prod_{j=i+1}^{n}\cos^{j-i-1}\theta_{ij}. \qquad \text{(A.26)}
$$

Finally, combining this with Expression A.12 yields

$$
\prod_{i=1}^{p}\det\left(I_{-i}^T R_i^T \cdots R_1^T \partial_i Y_i\right) = \prod_{i=1}^{p}\prod_{j=i+1}^{n}\cos^{j-i-1}\theta_{ij}. \qquad \text{(A.27)}
$$

Appendix B

Summary of Patient Demographics and Missing Values for the Application of the Givens Representation

Injury Mech.	N	% Male	Mean Age	Std. Age	% TBI	Med. Blood Units
GSW	85	90	27	8	40	5.0
MVC	52	90	27	7	75	0.5
SW	21	71	29	8	14	3.0
Assault	16	100	27	7	94	0.0

Table B.1: Brief sub-cohort description including number of samples in each sub-cohort (N), percentage of the sub-cohort that is male, average and standard deviation of age, percent of patients with traumatic brain injury, and median blood units recieved in the sub-cohort.

Table of Missing Values

Measurement	Number of Missing Value (out of 174)
Proteins	
FactorII	49
FactorX	50
Protein C	44
D-Dimer	54
Fibrinogen	123
Platelets	8
TEG	
R	135
K	135
MA	133
Ly30	133

Table B.2: Number of missing values for each measurement type.

Appendix C

Use of Cauchy Priors in Application of the Givens Representation

We note that in the CCA model described in Chapter 4, we place Cauchy priors over the scaling variables, Λ and Γ, which induce a sparsity property over latent dimensionality, forcing Λ and Γ to go to zero unless the signal in the data is sufficiently strong. Similarly we place Cauchy priors on the entries of the weight matrices W and B to induce sparsity in the matrix entries for interpretability purposes. These Cauchy priors serve a purpose that is akin to Laplace priors (or the L1 regularization in frequentist analysis), because just like the Laplace prior, the Cauchy prior places most of its mass near zero, with large mass in the tails of the distribution, allowing for appropriately large values when statistical signal is strong enough. However, we prefer the Cauchy prior, as its continuity property makes more sense in the context of our analysis (see [57] for a discussion on using Cauchy priors over Laplace priors). Finally, we mention that the hyper-parameter values of the Cauchy priors are chosen by leaving them as an unknown in our Stan model and conducting full Bayesian inference to sample from their posterior distribution.

Appendix D

Eigenvalues of the Update Matrix for Implicit Midpoint on a Linear System

For the simple univariate Gaussian system with mass matrix one defined in Section 6.2.2, the update equations (6.9) reduce to

$$\begin{aligned} q_{n+1} &= q_n + h\left(\frac{p_n + p_{n+1}}{2}\right) \\ p_{n+1} &= p_n - h\sigma^2\left(\frac{q_n + q_{n+1}}{2}\right). \end{aligned} \tag{D.1}$$

In matrix notation, these update equations can be rewritten as

$$\begin{pmatrix} q_{n+1} \\ p_{n+1} \end{pmatrix} = \begin{pmatrix} 1 & \frac{h}{2} \\ -\frac{h}{2}\sigma^2 & 1 \end{pmatrix} \begin{pmatrix} q_n \\ p_n \end{pmatrix} + \begin{pmatrix} 0 & \frac{h}{2} \\ -\frac{h}{2}\sigma^2 & 0 \end{pmatrix} \begin{pmatrix} q_{n+1} \\ p_{n+1} \end{pmatrix}. \tag{D.2}$$

Collecting the $n+1$ terms on the right yields

$$\begin{pmatrix} 1 & -\frac{h}{2} \\ \frac{h}{2}\sigma^2 & 1 \end{pmatrix} \begin{pmatrix} q_{n+1} \\ p_{n+1} \end{pmatrix} = \begin{pmatrix} 1 & \frac{h}{2} \\ -\frac{h}{2}\sigma^2 & 1 \end{pmatrix} \begin{pmatrix} q_n \\ p_n \end{pmatrix}, \tag{D.3}$$

which yields the update equation

$$\begin{aligned} \begin{pmatrix} q_{n+1} \\ p_{n+1} \end{pmatrix} &= \begin{pmatrix} 1 & -\frac{h}{2} \\ \frac{h}{2}\sigma^2 & 1 \end{pmatrix}^{-1} \begin{pmatrix} 1 & \frac{h}{2} \\ -\frac{h}{2}\sigma^2 & 1 \end{pmatrix} \begin{pmatrix} q_n \\ p_n \end{pmatrix} \\ &= \frac{1}{1+\frac{h^2}{4}\sigma^2} \begin{pmatrix} 1 & \frac{h}{2} \\ -\frac{h}{2}\sigma^2 & 1 \end{pmatrix} \begin{pmatrix} 1 & \frac{h}{2} \\ -\frac{h}{2}\sigma^2 & 1 \end{pmatrix} \begin{pmatrix} q_n \\ p_n \end{pmatrix} \\ &= \frac{1}{1+\frac{h^2}{4}\sigma^2} \begin{pmatrix} 1-\frac{h^2}{4}\sigma^2 & h \\ -h\sigma^2 & 1-\frac{h^2}{4}\sigma^2 \end{pmatrix} \begin{pmatrix} q_n \\ p_n \end{pmatrix}. \end{aligned} \tag{D.4}$$

Note that the eigenvalues of the scalar times the matrix in this equation are simply the eigenvalues of the matrix times the scalar in front. Defining $\mu := (h^2/4)\sigma^2$ these eigenvalues are the solutions of the quadratic equation

$$\lambda^2 - 2(1-\mu) + (1-\mu)^2 + h^2\sigma^2 = 0 \tag{D.5}$$

which by the quadratic formula are

$$\lambda_{1,2} = (1-\mu) \pm h\sigma i. \tag{D.6}$$

Thus the moduli of the eigenvalues of the overall update matrix are one.

Bibliography

[1] A. Gelman, H. S. Stern, J. B. Carlin, D. B. Dunson, A. Vehtari, and D. B. Rubin, *Bayesian Data Analysis*. Chapman and Hall/CRC, 2013.

[2] E. C. Hayden, *Is the $1,000 genome for real?*, *Nature News* (2014).

[3] M. A. Hamburg and F. S. Collins, *The Path to Personalized Medicine*, *New England Journal of Medicine* **363** (2010), no. 4 301–304.

[4] M. Whirl-Carrillo, E. M. McDonagh, J. Hebert, L. Gong, K. Sangkuhl, C. Thorn, R. B. Altman, and T. E. Klein, *Pharmacogenomics Knowledge for Personalized Medicine*, *Clinical Pharmacology & Therapeutics* **92** (2012), no. 4 414–417.

[5] L. Hood and S. H. Friend, *Predictive, Personalized, Preventive, Participatory (P4) Cancer Medicine*, *Nature reviews Clinical oncology* **8** (2011), no. 3 184.

[6] S. Duane, A. D. Kennedy, B. J. Pendleton, and D. Roweth, *Hybrid Monte Carlo*, *Physics Letters B* **195** (1987), no. 2 216–222.

[7] R. M. Neal *et. al.*, *MCMC using Hamiltonian dynamics*, *Handbook of Markov Chain Monte Carlo* **2** (2011), no. 11 2.

[8] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, *Equation of state calculations by fast computing machines*, *The Journal of Chemical Physics* **21** (1953), no. 6 1087–1092.

[9] A. Beskos, N. Pillai, G. Roberts, J.-M. Sanz-Serna, A. Stuart, *et. al.*, *Optimal tuning of the hybrid Monte Carlo algorithm*, *Bernoulli* **19** (2013), no. 5A 1501–1534.

[10] M. D. Hoffman and A. Gelman, *The no-u-turn sampler: adaptively setting path lengths in hamiltonian monte carlo.*, *Journal of Machine Learning Research* **15** (2014), no. 1 1593–1623.

[11] B. Carpenter, A. Gelman, M. D. Hoffman, D. Lee, B. Goodrich, M. Betancourt, M. Brubaker, J. Guo, P. Li, and A. Riddell, *Stan: A probabilistic programming language*, *Journal of Statistical Software* **76** (2017), no. 1.

[12] J. Zhou, J. Liu, V. A. Narayan, J. Ye, A. D. N. Initiative, *et. al.*, *Modeling Disease Progression via Multi-task Learning*, *NeuroImage* **78** (2013) 233–248.

[13] J. Zhou, J. Liu, V. A. Narayan, and J. Ye, *Modeling Disease Progression via Fused Sparse Group LASSO*, in *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 1095–1103, ACM, 2012.

[14] R. Sukkar, E. Katz, Y. Zhang, D. Raunig, and B. T. Wyman, *Disease Progression Modeling Using Hidden Markov Models*, in *Engineering in Medicine and Biology Society (EMBC), 2012 Annual International Conference of the IEEE*, pp. 2845–2848, IEEE, 2012.

[15] A. Pourzanjani, B. Bales, M. Harrington, and P. L, *Diagnosing alzheimer's the bayesian way*, *Proceedings of StanCon 2018, Asilomar* **2** (2018).

[16] B. M. Jedynak, A. Lang, B. Liu, E. Katz, Y. Zhang, B. T. Wyman, D. Raunig, C. P. Jedynak, B. Caffo, J. L. Prince, *et. al.*, *A Computational Neurodegenerative Disease Progression Score: Method and Results with the Alzheimer's Disease Neuroimaging Initiative Cohort*, *Neuroimage* **63** (2012), no. 3 1478–1486.

[17] M. Lorenzi, M. Filippone, D. C. Alexander, and S. Ourselin, *Disease Progression Modeling and Prediction through Random Effect Gaussian Processes and Time Transformation*, *arXiv preprint arXiv:1701.01668* (2017).

[18] J. Riihimäki and A. Vehtari, *Gaussian Processes with Monotonicity Information*, in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pp. 645–652, 2010.

[19] J. O. Ramsay, *Monotone regression splines in action*, *Statistical science* (1988) 425–441.

[20] A. A. Pourzanjani, R. M. Jiang, B. Mitchell, P. J. Atzberger, and L. R. Petzold, *General bayesian inference over the stiefel manifold via the givens transform*, *arXiv preprint arXiv:1710.09443* (2017).

[21] P. J. Brockwell, R. A. Davis, and M. V. Calder, *Introduction to time series and forecasting*, vol. 2. Springer, 2002.

[22] R. A. Johnson and D. W. Wichern, *Multivariate analysis*, *Encyclopedia of Statistical Sciences* **8** (2004).

[23] Z. Ghahramani, G. E. Hinton, *et. al.*, *The em algorithm for mixtures of factor analyzers*, tech. rep., Technical Report CRG-TR-96-1, University of Toronto, 1996.

[24] K. P. Murphy, *Machine learning: a probabilistic perspective*. MIT press, 2012.

[25] J. K. Ford, R. C. MacCallum, and M. Tait, *The application of exploratory factor analysis in applied psychology: A critical review and analysis*, *Personnel psychology* **39** (1986), no. 2 291–314.

[26] T. Hamelryck, J. T. Kent, and A. Krogh, *Sampling realistic protein conformations using local structural bias*, *PLoS Computational Biology* **2** (2006), no. 9 e131.

[27] S.-Y. Lee, W.-Y. Poon, and X.-Y. Song, *Bayesian analysis of the factor model with finance applications*, *Quantitative Finance* **7** (2007), no. 3 343–356.

[28] S.-H. Oh, L. Staveley-Smith, K. Spekkens, P. Kamphuis, and B. S. Koribalski, *2d bayesian automated tilted-ring fitting of disk galaxies in large hi galaxy surveys: 2dbat*, *Monthly Notices of the Royal Astronomical Society* (2017).

[29] F. Lu and E. Milios, *Robot pose estimation in unknown environments by matching 2d range scans*, *Journal of Intelligent and Robotic systems* **18** (1997), no. 3 249–275.

[30] B. Carpenter, A. Gelman, M. Hoffman, D. Lee, B. Goodrich, M. Betancourt, M. A. Brubaker, J. Guo, P. Li, and A. Riddell, *Stan: A probabilistic programming language*, *Journal of Statistical Software* **20** (2016).

[31] D. Tran, A. Kucukelbir, A. B. Dieng, M. Rudolph, D. Liang, and D. M. Blei, *Edward: A library for probabilistic modeling, inference, and criticism*, *arXiv preprint arXiv:1610.09787* (2016).

[32] J. Salvatier, T. V. Wiecki, and C. Fonnesbeck, *Probabilistic programming in python using pymc3*, *PeerJ Computer Science* **2** (2016) e55.

[33] A. Kucukelbir, R. Ranganath, A. Gelman, and D. Blei, *Fully automatic variational inference of differentiable probability models*, in *NIPS Workshop on Probabilistic Programming*, 2014.

[34] T. W. Anderson, I. Olkin, and L. G. Underhill, *Generation of random orthogonal matrices*, *SIAM Journal on Scientific and Statistical Computing* **8** (1987), no. 4 625–629.

[35] R. Shepard, S. R. Brozell, and G. Gidofalvi, *The representation and parametrization of orthogonal matrices*, *The Journal of Physical Chemistry A* **119** (2015), no. 28 7924–7939.

[36] R. Ranganath, S. Gerrish, and D. Blei, *Black box variational inference*, in *Artificial Intelligence and Statistics*, pp. 814–822, 2014.

[37] P. D. Hoff, *Simulation of the matrix bingham–von mises–fisher distribution, with applications to multivariate and relational data*, *Journal of Computational and Graphical Statistics* **18** (2009), no. 2 438–456.

[38] M. Brubaker, M. Salzmann, and R. Urtasun, *A family of mcmc methods on implicitly defined manifolds*, in *Artificial Intelligence and Statistics*, pp. 161–172, 2012.

[39] B. Leimkuhler and S. Reich, *Simulating hamiltonian dynamics*, vol. 14. Cambridge University Press, 2004.

[40] S. Byrne and M. Girolami, *Geodesic monte carlo on embedded manifolds*, *Scandinavian Journal of Statistics* **40** (2013), no. 4 825–845.

[41] A. Holbrook, A. Vandenberg-Rodes, and B. Shahbaba, *Bayesian inference on matrix manifolds for linear dimensionality reduction*, *arXiv preprint arXiv:1606.04478* (2016).

[42] R. W. Keener, *Theoretical statistics: Topics for a core course*. Springer, 2011.

[43] R. J. Muirhead, *Aspects of multivariate statistical theory*, vol. 197. John Wiley & Sons, 2009.

[44] A. Cron and M. West, *Models of random sparse eigenmatrices and bayesian analysis of multivariate structure*, in *Statistical Analysis for High-Dimensional Data*, pp. 125–153. Springer, 2016.

[45] C. A. León, J.-C. Massé, and L.-P. Rivest, *A statistical model for random rotations*, *Journal of Multivariate Analysis* **97** (2006), no. 2 412–430.

[46] M. E. Tipping and C. M. Bishop, *Probabilistic principal component analysis*, *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* **61** (1999), no. 3 611–622.

[47] G. Butland, J. M. Peregrín-Alvarez, J. Li, W. Yang, X. Yang, V. Canadien, A. Starostine, D. Richards, B. Beattie, N. Krogan, *et. al.*, *Interaction network containing conserved and essential protein complexes in escherichia coli*, *Nature* **433** (2005), no. 7025 531.

[48] A. A. Pourzanjani, T. B. Wu, R. M. Jiang, M. J. Cohen, and L. R. Petzold, *Understanding coagulopathy using multi-view data in the presence of sub-cohorts: A hierarchical subspace approach*, in *Machine Learning for Healthcare Conference*, pp. 338–351, 2017.

[49] D. Hoyert and J. Xu, *National Vital Statistics Reports: Deaths: Preliminary Data for 2011*, *National vital statistics reports* **61** (2012), no. 6.

[50] K. Brohi, J. Singh, M. Heron, and T. Coats, *Acute traumatic coagulopathy.*, *The Journal of trauma* **54** (2003) 1127–1130.

[51] J. R. Hess, K. Brohi, R. P. Dutton, C. J. Hauser, J. B. Holcomb, Y. Kluger, K. Mackway-Jones, M. J. Parr, S. B. Rizoli, T. Yukioka, *et. al.*, *The coagulopathy of trauma: a review of mechanisms*, *Journal of Trauma and Acute Care Surgery* **65** (2008), no. 4 748–754.

[52] M. E. Kutcher, A. R. Ferguson, and M. J. Cohen, *A principal component analysis of coagulation after trauma*, *The journal of trauma and acute care surgery* **74** (2013), no. 5 1223.

[53] E. Gonzalez, E. Moore, H. Moore, M. Chapman, C. Silliman, and A. Banerjee, *Trauma-induced coagulopathy: an institution?s 35 year perspective on practice and research*, *Scandinavian Journal of Surgery* **103** (2014), no. 2 89–103.

[54] K. Brohi, M. J. Cohen, M. T. Ganter, M. A. Matthay, R. C. Mackersie, and J.-F. Pittet, *Acute traumatic coagulopathy: initiated by hypoperfusion: modulated through the protein c pathway?*, *Annals of surgery* **245** (2007), no. 5 812.

[55] S. Van Buuren, *Flexible imputation of missing data*. CRC press, 2012.

[56] F. R. Bach and M. I. Jordan, *A probabilistic interpretation of canonical correlation analysis*, .

[57] A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin, *Bayesian data analysis*, vol. 2. Chapman & Hall/CRC Boca Raton, FL, USA, 2014.

[58] P. Koev and A. Edelman, *The efficient evaluation of the hypergeometric function of a matrix argument*, *Mathematics of Computation* **75** (2006), no. 254 833–846.

[59] A. Pourzanjani, T. Wu, B. Bales, and L. Petzold, *Relating disparate measures of coagulapathy using unorthodox data: A hybrid mechanistic-statistical approach*, *Proceedings of StanCon 2018, Helsinki* **2** (2018).

[60] B. Carpenter, *Predator-prey population dynamics: the lotka-volterra model in stan*, 2018.

[61] C. Margossian and B. Gillespie, *Differential equations based models in stan*, 2017.

[62] A. Y. Mitrophanov, A. S. Wolberg, and J. Reifman, *Kinetic model facilitates analysis of fibrin generation and its modulation by clotting factors: implications for hemostasis-enhancing therapies*, *Molecular BioSystems* **10** (2014), no. 9 2347–2357.

[63] A. Sagar and J. D. Varner, *Dynamic modeling of the human coagulation cascade using reduced order effective kinetic models*, *Processes* **3** (2015), no. 1 178–203.

[64] A. Quarteroni, R. Sacco, and F. Saleri, *Numerical mathematics*, vol. 37. Springer Science & Business Media, 2010.

[65] C. Margossian, *Computing steady states with stan's nonlinear algebraic solver*, 2018.

[66] A. Pourzanjani and L. Petzold, *Implicit hamiltonian monte carlo for sampling multiscale distributions*, *Submitted to the Springer Journal of Statistics and Computing* (2019).

[67] M. Betancourt, *A conceptual introduction to Hamiltonian Monte Carlo*, *arXiv preprint arXiv:1701.02434* (2017).

[68] E. Hairer, C. Lubich, and G. Wanner, *Geometric numerical integration: structure-preserving algorithms for ordinary differential equations*, vol. 31. Springer Science & Business Media, 2006.

[69] B. Shahbaba, S. Lan, W. O. Johnson, and R. M. Neal, *Split Hamiltonian Monte Carlo*, *Statistics and Computing* **24** (2014), no. 3 339–349.

[70] W.-L. Chao, J. Solomon, D. Michels, and F. Sha, *Exponential integration for Hamiltonian Monte Carlo*, in *International Conference on Machine Learning*, pp. 1142–1151, 2015.

[71] M. Okudo and H. Suzuki, *Hamiltonian Monte Carlo with explicit, reversible, and volume-preserving adaptive step size control*, *JSIAM Letters* **9** (2015) 33–36.

[72] M. Betancourt, *A general metric for Riemannian manifold Hamiltonian Monte Carlo*, in *Geometric science of information*, pp. 327–334. Springer, 2013.

[73] U. M. Ascher and S. Reich, *On some difficulties in integrating highly oscillatory Hamiltonian systems*, in *Computational Molecular Dynamics: Challenges, Methods, Ideas*, pp. 281–296. Springer, 1999.

[74] U. M. Ascher and L. R. Petzold, *Computer methods for ordinary differential equations and differential-algebraic equations*, vol. 61. SIAM, 1998.

[75] B. Leimkuhler and C. Matthews, *Molecular Dyamics.* Springer, 2016.

[76] D. A. Knoll and D. E. Keyes, *Jacobian-free Newton–Krylov methods: a survey of approaches and applications*, *Journal of Computational Physics* **193** (2004), no. 2 357–397.

[77] S. C. Eisenstat and H. F. Walker, *Choosing the forcing terms in an inexact newton method*, *SIAM Journal on Scientific Computing* **17** (1996), no. 1 16–32.

[78] Y. Saad and M. H. Schultz, *GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems*, *SIAM Journal on Scientific and Statistical Computing* **7** (1986), no. 3 856–869.

[79] C. D. Meyer, *Matrix analysis and applied linear algebra*, vol. 2. Siam, 2000.

[80] A. W. Long, K. C. Wolfe, M. J. Mashner, and G. S. Chirikjian, *The banana distribution is Gaussian: A localization study with exponential coordinates*, *Robotics: Science and Systems VIII; MIT Press: Cambridge, MA, USA* (2013) 265–272.