The Pennsylvania State University
The Graduate School

SCALABLE BAYESIAN INFERENCE FOR GENERALIZED MULTIVARIATE DYNAMIC LINEAR MODELS

A Thesis in
Informatics
by
Manan Saxena

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Master of Science

May 2024

The thesis of Manan Saxena was reviewed and approved by the following:

Abstract

Generalized Multivariate Dynamic Linear Models (GMDLMs) are a flexible class of multivariate time series models well-suited for non-Gaussian observations. They represent a special case within the more widely recognized multinomial logistic-normal (MLN) models. They are effective for analyzing sequence count data due to their ability to handle complex covariance structures and provide interpretability/control over the structure of the model. However, their current implementations are limited to small datasets, primarily because of computational inefficiency and increased variance in parameter estimates. Our work addresses the need for scalable Bayesian inference methods for these models. We develop an efficient method for obtaining a point estimate of our parameter by using the Kalman Filter and calculating closed-form gradients for our optimizer. Additionally, we provide uncertainty quantification of our parameter using Multinomial Dirichlet Bootstrap and refine these estimates further with Particle Refinement. We demonstrate that our inference scheme is considerably faster than STAN and provides a reliable approximation comparable to results obtained from MCMC.

Table of Contents

List of Figures

List of Tables

Acknowledgments

I would like to express my sincere gratitude to my advisor, Professor Justin Silverman, for his unmatched guidance. Through his courses, he introduced me to a new research area and helped me navigate it every step of the way. I really appreciate Professor Romit Maulik and Professor Aron Laszka for joining my committee. Their insights have been instrumental in the success of my research.

I am also immensely grateful to Professor James Wang for providing me with guidance and the opportunity to structure my master's program during the first year. I would also like to thank my peers in the Silverman Lab for helping and providing constructive criticism during discussions.

Finally, I am eternally grateful to my parents, brother, friends, and country for supporting me through the years and giving me the skills to succeed.

JDS and MS were supported in part by NIH 1-R01-GM148972-01. Any views and conclusions contained herein are those of the author and do not necessarily represent the official positions, express or implied, of the funders.

Chapter 1 Introduction

Our research aims to analyze and create models for data that can be represented as a Multinomial Time Series, as shown in Figure 1.1. We possess a dataset Y which contains T observations of D-dimensional count vectors. The counting process for each observation is modeled as a multinomial. Each count vector has a total of $n_t = \sum_i Y_{it}$ counts. This type of data is commonly found in various real-world applications, such as predicting the flow of students in the Italian school system (Cargnoni et al., 1997), constructing topic models in natural language processing (Glynn et al., 2019; Blei and Lafferty, 2006), and more recently for time series modeling of microbiome data (Silverman et al., 2018). Our research is specifically focused on microbial data, particularly microbial data in the human gut.

Human gut microbiota is dynamic, changing on daily time-scales in response to various factors such as disease, diet, or drugs (David et al., 2014). Because of its dynamic nature, there has been a substantial focus on longitudinal studies, leading to abundant time-series data. However, the sequencing-based measurement process by which this data is obtained is non-trivial: it can roughly be thought of as a random sample of the bacteria in the gut with the size of the sample being arbitrary and unrelated to total microbial load. As a result, many authors model this data as multinomial and focus on estimating dynamic changes in microbial proportional abundances (Silverman et al., 2018, 2022). As this data is zero-laden (often with over 50% zero values), Bayesian methods are increasingly preferred to account for the limited observations present for many taxa.

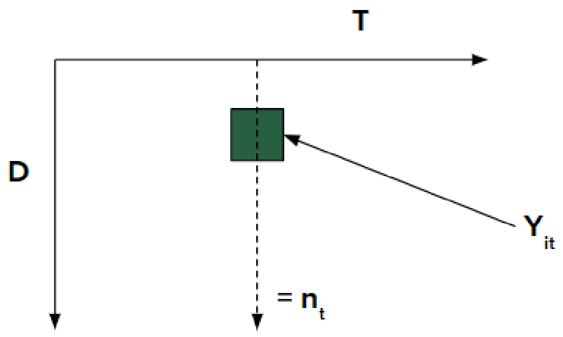


Figure 1.1. Multinomial Time Series Data

D: Number of taxa	Y_{it}: Count of i^{th} taxa at t^{th} timepoint
T: Number of timepoints	n_t: summation of all counts at t^{th} timepoint

While Bayesian Multinomial-Dirichlet models are the most common form of Bayesian multinomial model in many fields, these models can perform poorly when exist substantial correlations between multinomial categories. Especially in the human gut, phylogenetic relationships between bacteria can lead to substantial positive and negative correlations, making Multinomial-Dirichlet models sub-optimal (Silverman et al., 2022). An alternative to the Multinomial-Dirichlet is the Multinomial Logistic-Normal, the logistic-normal distribution relaxes the independence assumption inherent in the Dirichlet distribution. In application to multinomial time series, Multinomial Logistic-Normal Dynamic Linear Models have become increasingly popular (Silverman et al., 2018, 2022; Cargnoni et al., 1997). Yet the lack of conjugacy between the multinomial and the logistic-normal makes these models computationally intractable for many real-world applications. In this work, we aim to solve those issues by building an optimized and fast method to calculate Maximum A Posteriori (MAP) estimates of the parameters of the GMDLM and then use Multinomial Dirichlet Bootstrapping with parallelized particle refinement to produce an uncertainty quantification of the parameters in a computationally efficient manner.

Chapter 2 Background

2.1 Multinomial Logistic-Normal(MLN) Models

This class of models is extensively used for analyzing Multinomial Time Series data. Each count vector in the data is modeled as coming from a multinomial distribution, and any time-varying stochastic process with parameters can characterize the probability vector. We define the MLN model as a multinomial transformed-multivariate normal model as:

$$\begin{aligned} Y_{.t} &\sim \text{Multinomial}(n_t, \pi_{.t}) \\ \pi_{.t} &= \phi^{-1}(\eta_{.t}) \\ \eta_{.t} &\sim \mathcal{N}(\mu_{.t}, \Sigma) \end{aligned} \tag{2.1}$$

In our study, we represent ϕ^{-1} as an inverse log-ratio transform, such as the Inverse ALR (Inverse additive log-ratio) transform given by the following equation

$$\text{ALR}_D^{-1}(\eta_j) = \left(\frac{e^{\eta_{1j}}}{1 + \sum_{i=1}^{D-1} e^{\eta_{ij}}}, \dots, \frac{e^{\eta_{(D-1)j}}}{1 + \sum_{i=1}^{D-1} e^{\eta_{ij}}}, \frac{1}{1 + \sum_{i=1}^{D-1} e^{\eta_{ij}}} \right) \tag{2.2}$$

The parameter $\eta_{.t}$ is modeled as a multivariate normal distribution, providing the flexibility to utilize pre-existing multivariate normal models. This substitution allows for a seamless integration of well-established multivariate techniques within this compositional framework.

2.2 Generalized Multivariate Dynamic Linear Models(GMDLM)

Analyzing time series data often presents a challenge due to having only a single realization of the process, the properties of which may not be fully understood at a given time point.

This limitation leads to the assumption that certain distribution properties of the process generating the observations remain constant over time, making it difficult to pinpoint the exact causes of observed data variability. To address these challenges, one can adopt a state space framework, where observations are generated by applying a function to the hidden state of the system. This hidden state evolves over time with each new data point, offering increased flexibility in modeling data variability.

In this framework, the focus is on the underlying model itself, represented as a dynamic linear model (DLM) when the system's operators are linear. The model captures the process's evolution and its dependency on previous states, allowing for a detailed analysis of time series data. DLMs are categorized based on the dimensionality of the observations: univariate DLMs for 1-dimensional vectors and multivariate DLMs for higher dimensions (Laine, 2019).

Generalized Multivariate models are an extension and a flexible class of multivariate DLMs for non-Gaussian observations. These were first introduced in Quintana and West (1987) and developed further in West and Harrison (2006). As per the notations in Silverman et al. (2022); West and Harrison (2006) these models are defined as

$$\begin{aligned}
Y_{.t} &\sim f(\pi_{.t}) \\
\pi_{.t} &= \phi^{-1}(\eta_{.t}) \\
\eta_t^\intercal &= F_t^\intercal \Theta_t + v_t \qquad v \sim \mathcal{N}(0, \gamma_t \Sigma) \\
\Theta_t &= G_t \Theta_{t-1} + \Omega_t, \quad \Omega_t \sim \mathcal{N}(0, W_t \Sigma) \\
\Theta_0 &\sim \mathcal{N}(M_0, C_0, \Sigma) \\
\Sigma &\sim \mathcal{IW}(\Xi, \nu)
\end{aligned} \tag{2.3}$$

Y is a $D \times N$ matrix, and f is any function that maps probability vectors to Y. In our experiments, Y represents count data; hence, f is a multinomial distribution. ϕ is any log-ratio transform to convert from simplex space of π to a real valued space of η. Each η_t is a P-dimensional vector, F_t is a $Q \times 1$ vector called the observation matrix describing a linear model connecting the latent state space to the parameter η_t, θ_t is a $Q \times P$ matrix describing the state of our time series at time t, G_t is the state transition matrix of size $Q \times Q$, Σ is a $P \times P$ covariance matrix depicting the covariation between the P-dimensional time series, W_t is $Q \times Q$ covariance matrix describing the covariation between the perturbations influencing the latent states, and γ_t is just a scalar hyperparameter to change the importance of the observation at time t (usually set to 1).

2.3 Collapse-Uncollapse (CU) Sampler

In Silverman et al. (2022), they proposed this sampling algorithm to sample from the posterior of relevant parameters of any probabilistic model represented by $p(\Psi, \eta, Y)$ where Ψ and η are the parameters of the models and can be decomposed into marginals of the form:

$$p(\eta, \Psi|Y) = p(\Psi|\eta, Y)\frac{p(\eta, Y)}{p(Y)} \tag{2.4}$$

With this decomposition, we can sample first from the posterior of the collapsed $p(\eta, Y)$ and then use that sample of η and Y; we can get a sample of Ψ using the conditional distribution of $p(\Psi|\eta, Y)$. Together the sample of η and Ψ then represents a single sample from our model. Also, since the sampling process is split into two separate processes, after generating samples from the collapsed form, we can generate samples of the uncollapsed part in a parallelized paradigm to further speed up the entire process. For most common models, we have an exact solution to map samples from the collapsed part to the other parameters in the uncollapsed part of the CU sampler, again helping to make sampling more efficient. The unique two-step structure of this sampler makes it particularly useful when partial conjugates exist in the probabilistic models similar to the partial conjugacy found in the Multinomial-Logistic Normal defined before.

Chapter 3 Related Work

Early work of modeling multinomial time series data as GMDLM used Metropolis within Gibbs samplers (Cargnoni et al., 1997) but could scale to a few multinomial categories. Polya-Gamma data augmentation and Variational Inference have been used recently, but Silverman et al. (2022) have found them to be computationally expensive and perform poorly. Silverman et al. (2018) have found Hamiltonian Monte Carlo(HMC) provides a more scalable approach for modeling the data, but they also suffer from scaling beyond ten multinomial categories and 800 time points. Most recently, Silverman et al. (2022) proposes a class of models called Latent Matrix-T Processes(LTPs), which are shown to generalize Matrix-T processes to handle a wider variety of data types like count data. It is proven that many popular MLN models are special cases of these models, including GMDLMs. Being the current state-of-the-art, we delve deeper into these models, their problems, and how they can be solved.

3.1 Latent Matrix-T Processes (LTP)

In this paradigm, Y is presented as a hierarchical process formulated by process f, which has parameters that, after log-ratio transformations, follow a matrix-t process. The model form of LTPs looks like this :

$$\begin{aligned} Y &\sim f(\pi) \\ \pi &= \phi^{-1}(\eta) \\ \eta &\sim T(\nu, B, K, A). \end{aligned} \tag{3.1}$$

Where B, K, and A are real-valued matrices representing the mean and covariance structure of the Matrix-T processes. ϕ^{-1} is usually a log-ratio transformation from the real-valued space to the probability space Π.

In their research, Silverman et al. (2022) introduced a larger class of models called Marginally LTP models. These models are essentially those whose marginal is an LTP. They derived this form for common MLN models like the Generalized Multivariate Conjugate Linear model (GMCL), Generalized Multivariate Gaussian Processes (GMGP), and GMDLMs. Representing these in the Marginal LTP form enables us to use the Collapse-Uncollapse sampler for efficient posterior inference. This is made possible because all of these models have a closed-form posterior conditional (facilitating the 'uncollapse' phase) and the joint marginal (enabling the 'collapsed' phase) of the sampler.

For GMDLM, we can write it as a model with joint distribution $p(\Theta, \Sigma, \eta, Y)$ and factored as $p(\Theta, \Sigma|\eta, Y)p(\eta, Y)$ with $\Psi = \{\Theta, \Sigma\}$. Then $p(\eta, Y)$ could be presented as an LTP with the following B, K, and A matrices -

$$B = \begin{bmatrix} | & | & & | \\ \alpha_1 & \alpha_t & \cdots & \alpha_T \\ | & | & & | \end{bmatrix}$$

$$\alpha_t = (F_t^\intercal G_{t:1}^\intercal M_0)^\intercal$$

$$K = \Xi$$

$$A_{t,t-k} = \begin{cases} \gamma_t + F_t^\intercal \left[W_t + \sum_{l=t}^{2} G_{t:l} W_{l-1} G_{l:t}^\intercal + G_{t:1} C_0 G_{1:t}^\intercal\right] F_t & \text{if } k = 0 \\ F_t^\intercal \left[G_{t:t-k+1}\, W_{t-k} + \sum_{l=t-k}^{2} G_{t:l}\, W_{l-1}\, G_{l:l-k}^\intercal + G_{t:1}\, C_0\, G_{1:t-k}^\intercal\right] F_{t-k} & \text{if } k > 0 \end{cases} \tag{3.2}$$

But to make the sampling from $p(\eta|Y)$ possible due to the computationally expensive nature of the problem, they also provide a Laplace approximation for the density. The approximation is defined as $q(\eta|Y) = N(\text{vec}\,\hat{\eta}, H^{-1}(\text{vec}\,\hat{\eta}))$ where $H^{-1}(\text{vec}\,\hat{\eta})$ denotes the inverse Hessian matrix of $\log p(\eta|Y)$ evaluated at the point $\text{vec}\,\hat{\eta}$ and $\hat{\eta}$ denotes the MAP estimate of $p(\eta|Y)$.

After generating the samples from the collapsed LTP form, such as by sampling from the Laplace approximation, they can be conditioned upon and samples of the remaining parameters Σ and Θ can be obtained from the posterior by performing the uncollapse step, which involves sampling from the posterior conditional $p(\Theta, \Sigma|\eta, Y)$. For GMDLMs, Silverman et al. (2022) have provided an efficient algorithm for uncollapsing through filtering and smoothing recursions mentioned below.

Filtering recursion:

(1) Posterior at $t-1$:

$$
\begin{aligned}
p(\Sigma|H_{t-1}^{\intercal}) &\sim IW(\Xi_{t-1}, \nu_{t-1}) \\
p(\Theta_{t-1}|\Sigma, H_{t-1}^{\top}) &\sim \mathcal{N}(M_{t-1}, C_{t-1}, \Sigma)
\end{aligned}
\tag{3.3}
$$

(2) Prior at t:

$$
\begin{aligned}
A_t &= G_t M_{t-1} \\
R_t &= G_t C_{t-1} G_t^{\top} + W_t \\
p(\Sigma|H_{t-1}^{\intercal}) &\sim IW(\Xi_{t-1}, \nu_{t-1}) \\
p(\Theta_t|\Sigma, H_{t-1}^{\top}) &\sim \mathcal{N}(A_t, R_t, \Sigma)
\end{aligned}
\tag{3.4}
$$

(3) One-step ahead forecast at t:

$$
\begin{aligned}
f_t^{\intercal} &= F_t^{\intercal} A_t \\
q_t &= \gamma_t + F_t^{\intercal} R_T F_t \\
p(\Sigma|H_{t-1}^{\intercal}) &\sim IW(\Xi_{t-1}, \nu_{t-1}) \\
p(\eta_t|\Sigma, H_{t-1}^{\intercal}) &\sim N(f_t, q_t \Sigma)
\end{aligned}
\tag{3.5}
$$

(4) Posterior at t:

$$
\begin{aligned}
e_t^{\intercal} &= \eta_t^{\intercal} - f_t^{\intercal} \\
S_t &= \frac{R_t F_t}{q_t} \\
M_t &= A_t + S_t e_t^{\intercal} \\
C_t &= R_t - q_t S_t S_t^{\intercal} \\
\nu_t &= \nu_{t-1} + 1 \\
\Xi_t &= \Xi_{t-1} + \frac{e_t e_t^{\intercal}}{q_t} \\
p(\Sigma|H_{t-1}^{\intercal}) &\sim IW(\Xi_t, \nu_t) \\
p(\Theta_t|\Sigma, H_t^{\intercal}) &\sim \mathcal{N}(M_t, C_t, \Sigma)
\end{aligned}
\tag{3.6}
$$

Smoothing Recursion:

1. Sample $\Sigma \sim IW(\Xi_T, \nu_T)$ and then $\Theta_T \sim N(M_t, C_t, \Sigma)$.

2. For each time t from $T-1$ to 0, sample $p(\Theta_t|\Theta_{t+1}, H^{\intercal})$ from $N(M_t^*, C_t^*, \Sigma)$ where

$$Z_t = C_t G_{t+1}^{\intercal} R_{t+1}^{-1}$$

$$\begin{aligned} M_t^* &= M_t + Z_t(\Theta_{t+1} - a_{t+1}) \\ C_t^* &= C_t - Z_t R_{t+1} Z_t^\intercal. \end{aligned}$$

3.2 Issues with GMDLM modeled as Latent Matrix-T Process

The approach of Modeling GMDLM as a Marginal LTP is mathematically sound and allows for accurate parameter inference. However, it faces scalability issues when dealing with a large number of multinomial categories and time points. This problem arises due to the non-stationary priors frequently specified in GMDLM. To fit these priors in the Marginal LTP form, we need to precompute the joint prior for the entire time series at the initial time point. This precomputation, in the form of A parameter matrix of the LTP, results in significant uncertainty in the states, which causes a numeric overflow. Therefore, this approach is not practical for larger real-world datasets. We call this phenomenon the "Bayesian Whip" and the following model shows a small simulation to prove our point.

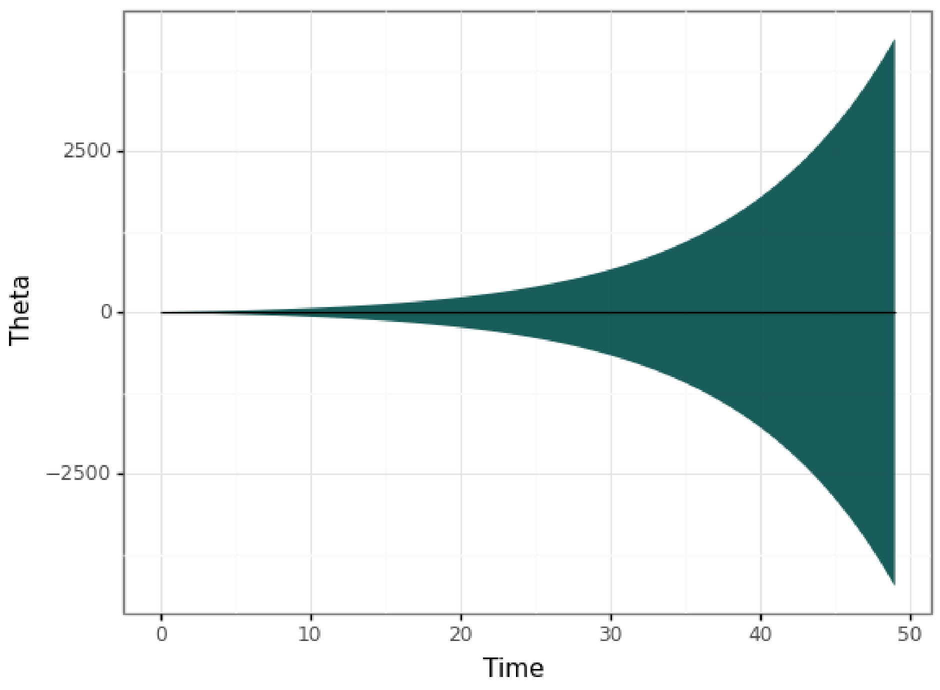
For simplicity purposes, if we consider Θ_0 as a univariate variable with a value of 0, the simulation produces Figure 3.1.

$$\begin{aligned} &\Theta_0 \sim \mathcal{N}(0,1) \\ &\Theta_t = G\Theta_{t-1} + W_t \quad W_t \sim \mathcal{N}(0,1) \end{aligned} \tag{3.7}$$

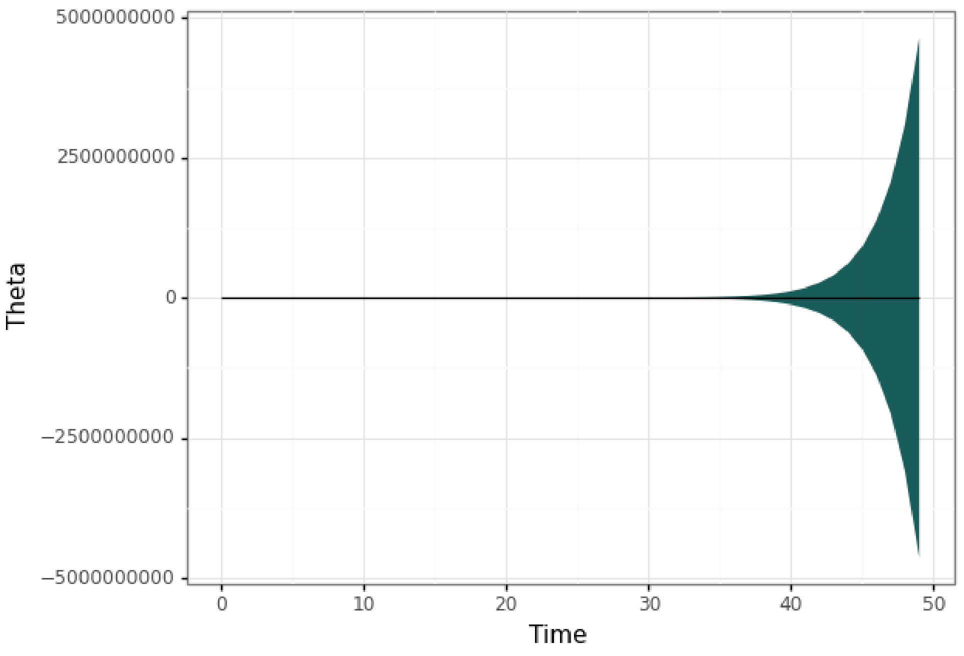
$$\overset{\text{recursive maths}}{\Downarrow}$$

$$\Theta_t = G^t\Theta_0 + \sum_{i=1}^{t} G^{t-i} W_i \tag{3.8}$$

As we see in the figure, as time increases, the variance of our state explodes, leading to numerical overflow. As G increases, the variance blowup is more significant and occurs for much smaller time points than 50.



$G = 1.1$



$G = 1.5$

Figure 3.1. Bayesian Whip phenomenon seen in the simulation

Chapter 4
Proposed Method

4.1 General Idea

As mentioned above, the Marginal Latent Matrix-T process form for GMDLMs (Silverman et al., 2022) can experience numerical instability and an explosion in state variance as time points increase. This is primarily due to GMDLM often being specified with non-stationary priors. To tackle these issues, our methodology differs from the practice of reducing the entire time series into a Matrix-T process represented by pre-calculated matrix parameters for the time series.

We utilize the concept of sequential filtering, which is popular in dynamic linear models, to estimate our parameters η and Θ. This is done with the aid of Kalman filtering and smoothing equations. By opting for sequential filtering, we avoid pre-computing the entire joint prior and instead, we only compute one-step-ahead conditional priors, which are based on the prior observations. Even for a non-stationary model, these one-step-ahead priors are usually straightforward to represent numerically and are not prone to numerical instability.

Our proposed method aims to get a feasible, computationally efficient way to find the MAP estimate of η, build approximate posterior credible intervals around that estimate, improve it using particle refinement, and then use closed-form posterior conditionals to get estimates of Θ parameter. Hence, our proposed method is broken down into four parts:

- Find the $\hat{\eta}$ (MAP estimate)
- Generate samples of η by performing uncertainty quantification around $\hat{\eta}$
- Improve estimate of η by performing particle refinement on samples

- Uncollapse to get estimates of the state Θ

4.2 MAP estimation

To obtain the Maximum A Posteriori (MAP) estimate of η, we solve the following optimization problem:

$$\hat{\eta} = \underset{\eta \in \mathbb{R}^{P \times N}}{\operatorname{argmin}} \left[-\log p(\eta \mid Y) \right]. \tag{4.1}$$

This problem can be partitioned into two components: the log-likelihood contribution from Y given the log-ratio transform of η and the prior contribution of η. This relationship is expressed as:

$$-\log p(\eta|Y) \propto -\log f(Y|\phi^{-1}(\eta)) - \log p(\eta) \tag{4.2}$$

The initial step in this process involves computing the distribution of η_t dependent only on previously occurred η values in the time series and marginalizing out the state Θ. We achieve this by using the filtering recursive equations in Silverman et al. (2022). These equations represent the three stages of the Kalman Filter, and are repeated for every time point in the series :

- Prior of the state is forecasted one step ahead
- Forecast the observation based on the forecasted state
- Capture the observation and update our beliefs about the state at the time t

Leveraging the equations from the one-step ahead forecast, we can marginalize out Σ to derive the distribution of η_t solely based on $H_{t-1}^{\intercal}$(η values till $t-1$). By resolving this relation, we obtain:

$$p(\eta_t|H_{t-1}^{\intercal}) \sim T(\nu_{t-1}, f_t, q_t\Xi_{t-1}) \tag{4.3}$$

Now, we can calculate the log probability of η and the gradients to be fed into our optimizer. Since η follows a multivariate T-distribution, its log probability is:

$$\log\ p(\eta_t|H_{t-1}^{\intercal})) \propto -\frac{(\nu_{t-1}+p)}{2}\ \log\left(1+\frac{1}{\nu_{t-1}}(\eta_t-f_t)^{\intercal}(q_t\Xi_{t-1})^{-1}(\eta_t-f_t)\right) \tag{4.4}$$

And we get the following gradients derived in Appendix A :

$$\begin{aligned}\frac{d\log p(\eta_t|H_{t-1}^{\intercal})}{d\eta_t} &\propto \frac{-(\nu_{t-1}+p)}{2}\frac{1}{S}\frac{dS}{d\eta_t} \\ \frac{dS}{d\eta_t} &= \frac{2}{\nu_{t-1}}(\eta_t^{\intercal}-f_t^{\intercal})(q_t\Xi_{t-1})^{-1} \\ S &= 1+\frac{1}{\nu_{t-1}}(\eta_t-f_t)^{\intercal}(q_t\Xi_{t-1})^{-1}(\eta_t-f_t)\end{aligned} \tag{4.5}$$

4.3 Uncertainty quantification

In Silverman et al. (2022), a Laplace approximation is provided for uncertainty quantification around the MAP estimate $\hat{\eta}$, but calculating the Hessian matrix is complex and computationally expensive. Hence, we utilize Multinomial Dirichlet Bootstrap in this work to provide uncertainty quantification. It provides samples around our estimate, which helps us get a mean estimate and 95% CI around the mean. We use the following equation to generate the samples $\hat{\eta}$.

$$\eta_t^{\mathbf{S}} \sim \phi(\mathrm{Dir}(\phi^{-1}(\hat{\eta}_t) * \sum Y_t + \alpha)) \tag{4.6}$$

Here, $\hat{\eta}_t$ is the t^{th} vector of $\hat{\eta}$ matrix, $\mathbf{S}$ defines the number of Dirichlet samples generated, α is the pseudocount to deal with 0 values and ϕ represents any log-ratio transform. In our case, we have used the ALR transform.

4.4 Particle Refinement

We can improve our posterior estimates of η by performing particle refinement of the Multinomial Dirichlet samples generated in the previous step. The particle refinement allows us to better explore the posterior of $p(\eta|Y)$, giving better mean estimates and credible intervals of the parameter.

In our experiments, we employ the Metropolis algorithm. Each sample of η from the set $\mathbf{S}$, generated in the previous step, serves as an initial state for our $\mathbf{S}$ chains. Given that our initial states are close to the MAP of the posterior of η, we only need to run each chain for a few iterations. Our target function is the probability density $p(\eta|Y)$, and for the proposal generation distribution, we use a normal distribution. This distribution's mean is the current state in our chain, while the covariance follows a mixture of isotropic normal Z and a normal with the covariance matrix over our Dirichlet samples:

$$\begin{aligned}\eta_{\text{proposal}} &= VZ + (1-V)U \\ Z &\sim N(\eta_{k-1}, \delta I) \\ U &\sim N(\eta_{k-1}, \delta\Sigma) \\ V &\sim \text{Bernoulli}(\pi)\end{aligned} \tag{4.7}$$

Here, Σ represents the covariance matrix across the **S** Dirichlet samples. The hyperparameter δ is used to scale the covariance matrices. The hyperparameter π determines the likelihood of choosing between the scaled Identity matrix **I** and **Σ**. k is simply the k^{th} step out of the total K steps in each chain. We calculate the **Σ** using the following equation:

$$\begin{aligned}\eta^S &= vec(\eta^s) \\ \Sigma &= Cov(\eta^1, ..., \eta^S)\end{aligned} \tag{4.8}$$

This setup allows for a nuanced exploration of the parameter space, balancing exploration and exploitation in the Metropolis algorithm. Further, since running MCMC chains is time-consuming, we can speed this up by running the chains together in a parallelized paradigm. In principle, the most considerable speed bump would be if we could run all **S** chains together simultaneously, but the number depends on the computational resource being used.

4.5 State Θ Estimation

After improving our Dirichlet samples by particle refinement, we have completed the collapsed part of the CU sampler of sampling from the posterior density $p(\eta|Y)$. Now, we can use them to complete the uncollapse part of getting the state estimates of the GMDLM model. We do that using the filtering and smoothing equations mentioned in Silverman et al. (2022) and Prado and West (2010).

Since this procedure would give us S samples of Θ, we can also get mean and credible intervals for Θ estimates.

4.6 Software Implementation

We implement GMDLM using C++ and utilize the Eigen and Boost libraries for optimized matrix algebraic calculations. To run our model's MAP estimation part, we code the gradients in C++, create an R binding using Rcpp, and employ the LBFG-S optimizer provided by RcppNumerical. We implement the Metropolis algorithm in C++ with parallelization using OpenMP for faster particle refinement. This has been packaged as a standalone general-purpose header library with R bindings, providing a more accessible interface. Importantly, the particle refinement library we developed is versatile and can be employed for a wide range of particle refinement or MCMC tasks, not limited to our work on the GMDLM model

Chapter 5 Experiments

5.1 Simulation Dataset

We conducted experiments using simulated data based on the GMDLM model, explained in Chapter 2. The data was generated by simulating a mean-reverting random walk. This mean-reverting property was used to ensure that the data did not move too far away from the long-term average, which is a common dynamic observed in natural and economic systems.

In our simulation, we set the number of time points to $N = 500$ and the number of rows in the state matrix to $Q = 1$. We varied the number of taxa (D) from 3 to 100. The observation vector was used to convert the state matrix to the mean of observations (F_t), which was set to 1. The state transition matrix (G_t) has a single element of 0.9 since our Q was 1. The covariance matrix ($\mathbf{\Sigma}$), which represents the variance of the model's states and observations over time, was generated by sampling from an Inverse-Wishart distribution. The scale matrix for this distribution was an identity matrix $\mathbf{I}$ with $D + 4$ degrees of freedom.

For the initial state (Θ_0), we used samples from a Matrix-Normal distribution, with a mean of zero and identity matrix $\mathbf{I}$, and $\mathbf{\Sigma}$ as the covariance matrices. The observation noise was generated from a normal distribution with a mean of 0 and $\mathbf{\Sigma}$ as the covariance matrix. The state noise was also from a Matrix-Normal distribution with zero mean. However, it used W_t along with $\mathbf{\Sigma}$ for the covariance matrices. W_t in this simulation was fixed and did not change over time. It was a single draw from the inverse gamma distribution with a scale parameter of 2 and a rate of 1.

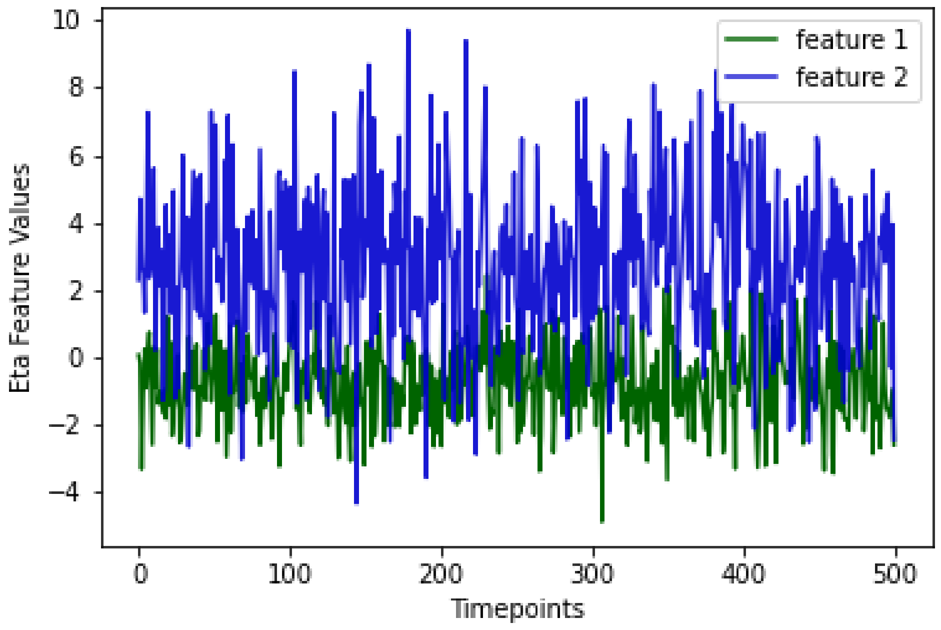


Figure 5.1. Plot of two randomly selected η features from simulated data

5.2 Results

The results are divided based on the sequential steps of our proposed approach in Chapter 4.

5.2.1 MAP estimation

Using the gradients and density function calculated for $p(\eta|Y)$ in the previous section, we find $\hat{\eta}$ (MAP estimate of η). As it is expected that the dimension of η would be significant, we decided to use gradient-based optimization methods like L-BFGS over Newtonian methods, which require computationally heavy and complex calculation of Hessian's.

We ran the optimizer written in C++ and RcppNumerical with our gradients provided and compared it to our version of the GMDLM model written in the STAN programming language. To implement the STAN version, we replicated the model from the previous chapter, which features an updated conditional distribution for η_t. However, unlike our optimizer, STAN uses automatic differentiation to calculate gradients for its L-BFGS optimizer, so we did not provide our own calculated gradients for this implementation.

In Figure 5.2, we show that our optimizer reaches the same minimum as STAN's because the ratio of final log probabilities of the two models approaches 1. We ran three simulations for each value of D varying from 3 to 100, as depicted in the figure. Table 5.1 shows that our optimizer reaches the same optimal log probability as STAN implementation in much fewer iterations.

D	3			25			50			75			100		
Simulation	I	II	III	I	II	III	I	II	III	I	II	III	I	II	III
Ours (C++)	11	14	13	16	19	20	20	20	19	28	26	19	74	81	66
STAN	20	24	21	61	87	97	224	240	242	417	432	286	2121	2342	1343

Table 5.1. Number of Iterations for Optimizers

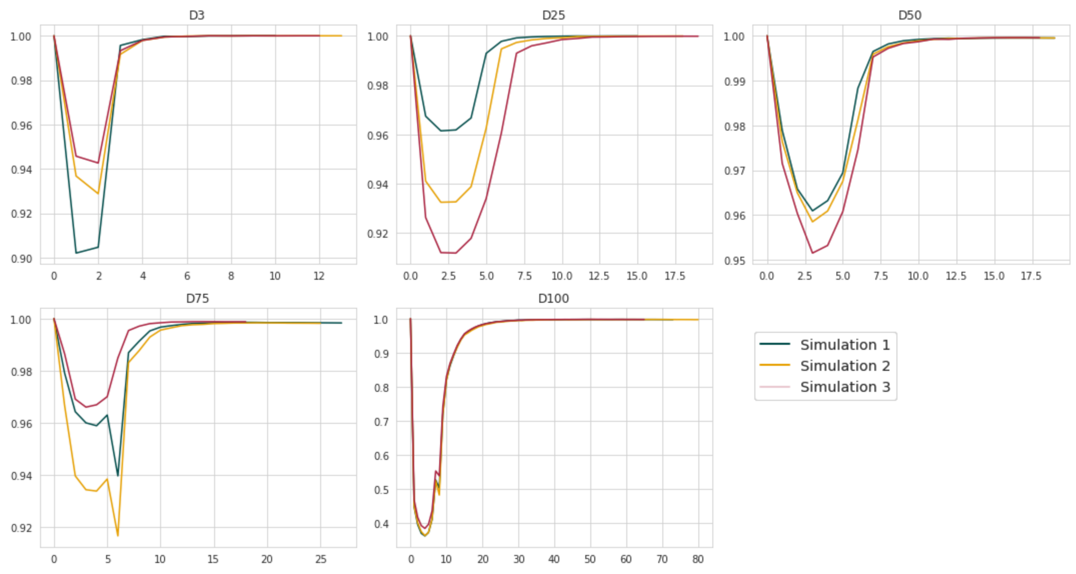


Figure 5.2. Ratio of STAN and Our C++ log probabilities vs Number of Iterations

Figure 5.3 depicts the average wall time per iteration for both optimizers on different D values. Our implementation takes much less wall time to optimize and is considerably faster for larger values of D.

5.2.2 Uncertainty Quantification

We use the multinomial Dirichlet bootstrap with the equation described before for uncertainty quantification. For our experiment, we take D as 10, N as 500, Q as 1, and pseudocount α as 0.1. As shown in Figure 5.4, we get a mean estimate of our η samples

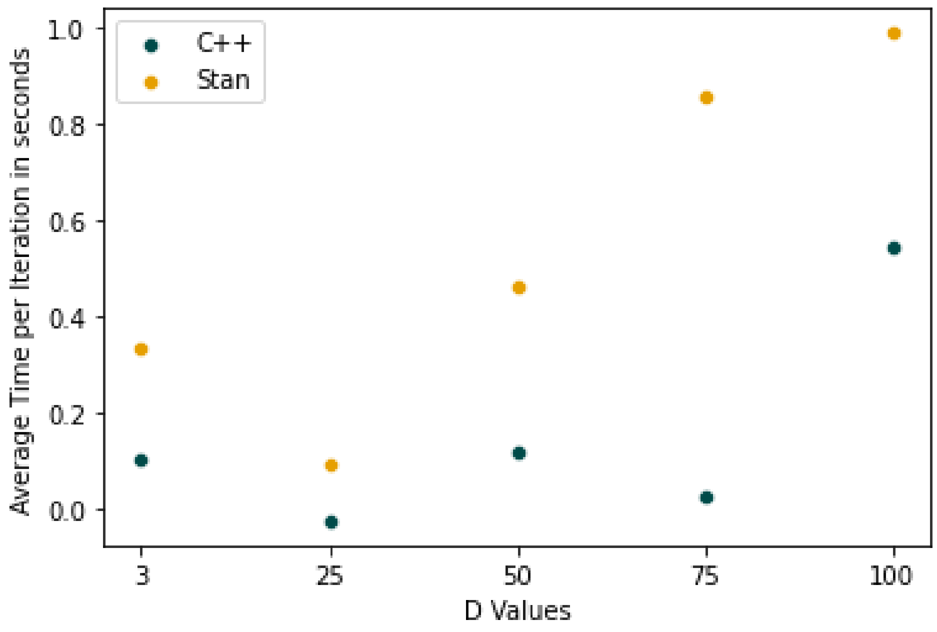


Figure 5.3. Average Time per Iteration vs D Values

along with a 95 % Confidence Interval. It depicts that our mean is similar to STAN's, and our error bound perfectly encapsulates the STAN mean. The STAN mean is generated from running 4 MCMC chains for 2000 steps, each with 2000 steps as warmup steps.

5.2.3 Particle Refinement

After obtaining our Multinomial Dirichlet Samples η_t^S, we further refine our estimates using the Metropolis algorithm. To assess the quality and efficiency of the samples produced, we calculate three key metrics: Effective Sample Size (ESS), Effective Samples per second (ESS/s), and Seconds per Effective Sample (SpES). We compare our results with those obtained from the MCMC run in STAN. The ESS metric determines the number of independent samples among the autocorrelated samples generated by the MCMC. It provides insights into the sampling efficiency and helps diagnose MCMC chains' convergence issues. ESS/s measures the rate at which independent samples are generated, indicating the computational efficiency of the sampling process. On the other hand, SpES highlights the computational effort required to achieve one effective sample (Carpenter et al., 2017). Table 5.2 shows the results of our experiment with $D = 10$, $N = 500$, and $Q = 1$. We adjust the hyperparameters of our proposal generating distribution to $\delta = 0.003$ and $\pi = 0.5$

As the table shows, we run more chains but only for a few steps. The reason is that

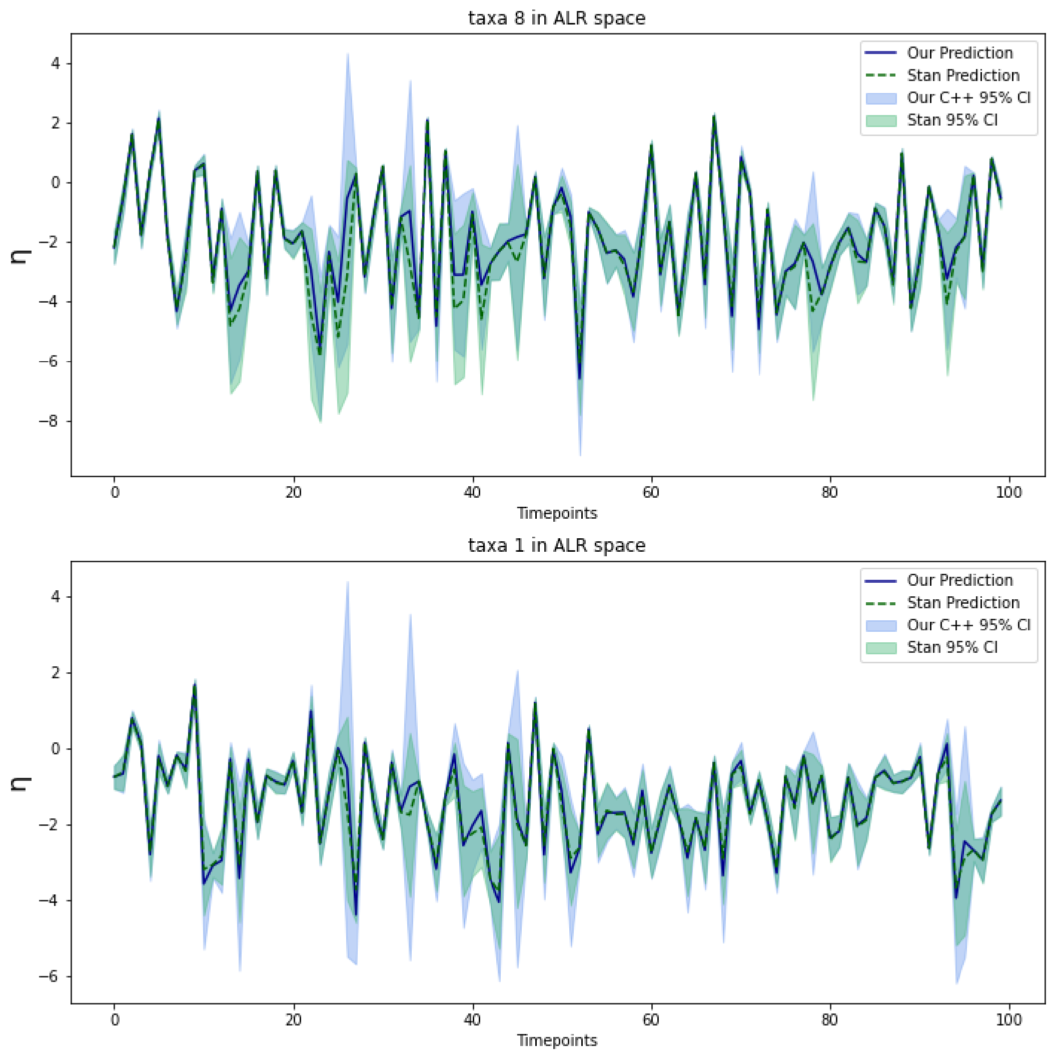


Figure 5.4. η Prediction vs Timepoints. The plot shows the eta prediction of two randomly selected taxa from D-1 η features evolving as the time series progresses.

our initial states for the chain are close to the MAP of the posterior of η, so we don't need to run many steps to get a better estimate. We note that our SpES is considerably lower than STANs, which firstly makes our method fast and also gives us the chance to run chains for more steps if required for the problem at hand.

Method	Num Chains	Num Steps	ESS	ESS/s	SpES
Our C++	2000	40	1139.18	91.32	0.01
STAN	4	2000	6773.19	0.62	1.61

Table 5.2. Comparison of MCMC metrics

5.2.4 State Θ estimation

Once we have the particle refined samples of η, we use the uncollapsing equations to get state Θ estimates. Below in Figure 5.5, we get a mean estimate of our state Θ samples, along with a 95% Credible Interval, and we compare our results with STAN's mean and 95% Credible Interval. But we note that our model massively underestimates the uncertainty as compared to STAN and we plan to look further into the possible reasons for this in our future work.

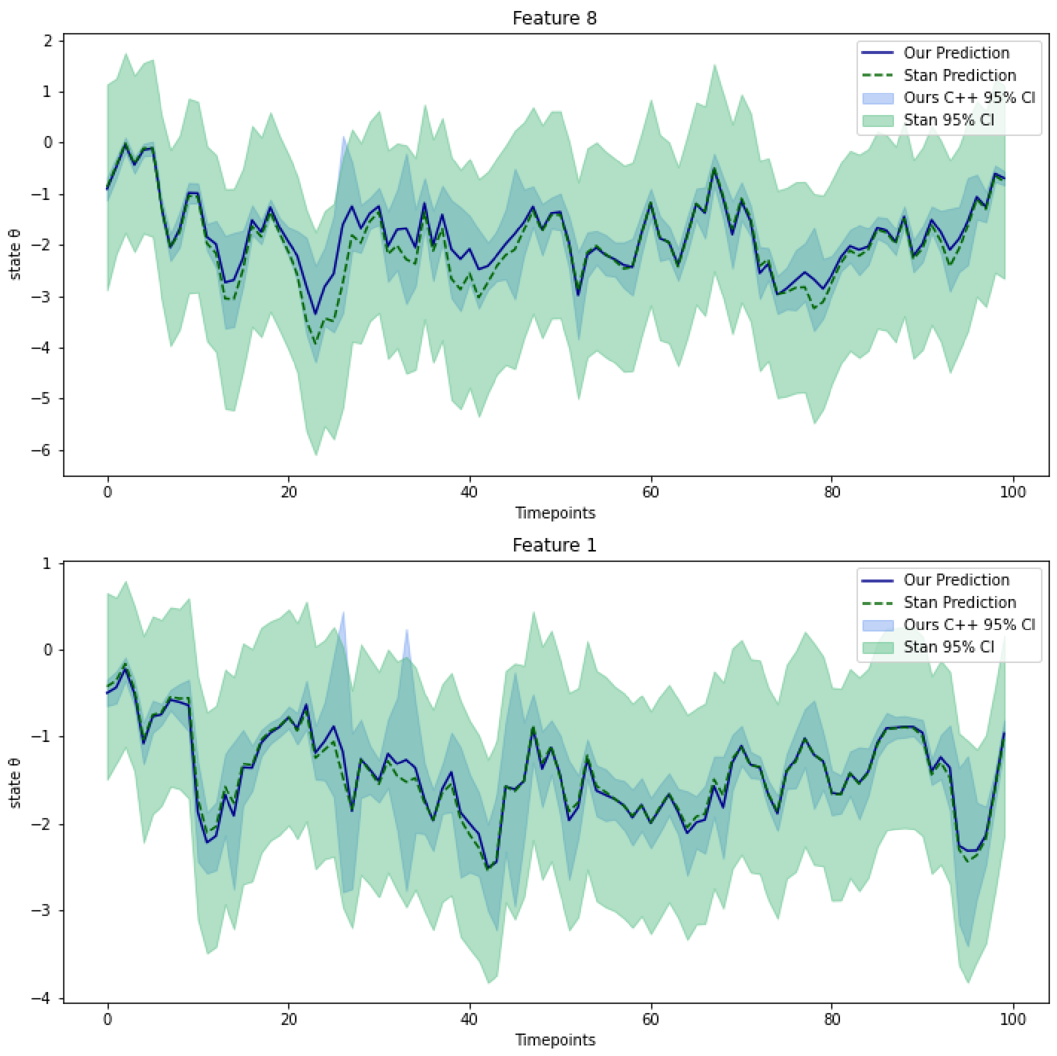


Figure 5.5. State Θ Prediction vs Timepoints. The plot shows the prediction of two randomly selected features of state Θ evolving as the time series progresses.

Chapter 6
Conclusion and Future Work

Our work tackles the scaling problems faced by previous approaches when modeling multinomial time series data. In particular, we solve the "Bayesian Whip" issue of exploding variance and numerical overflow prevalent in GMDLMs representation as LTP models using sequential filtering algorithms like Kalman Filter.

To form a complete solution for inferencing parameters of the GMDLM model, we first develop a fast and efficient way to calculate the MAP estimate of the parameter η. Our results show that we reach the same optimum as STAN with reduced computational requirements. To provide an uncertainty quantification around our parameter estimate $\hat{\eta}$, we use the Multinomial Dirichlet bootstrap to generate samples around the MAP estimate and further improve those samples by parallelized particle refinement. Using the uncollapsing smoothing and filtering equations, we can map those samples to samples depicting the estimates of the states Θ. With comparison to STAN's result, we show that our estimates are comparable and a good approximation to the true values.

In future work, we aim to refine the Multinomial Dirichlet bootstrap and particle refinement components of our solution. Our current model tends to significantly underestimate the uncertainty in the posterior distribution of the Θ parameter. This underestimation could be attributed to the low variability in the samples generated from the Multinomial Dirichlet process, which are too close to the MAP estimate of the η parameter, leading to insufficient exploration of the posterior during particle refinement. To address this, increasing the pseudocount parameter α may introduce more variability into the samples. Additionally, the inefficacy in posterior estimation might stem from the inadequacy of our particle refinement's proposal-generating function. If the function produces proposals too close to the current state, it restricts the exploration to a nearby region of the posterior, failing to capture its full variability. Conversely, if proposals are generated from regions too distant from the current state, the Metropolis algorithm may reject most proposals, again limiting the exploration of the posterior. Tuning the hyperparameters δ and π of our proposal generating function, coupled with monitoring the acceptance rate of proposals, could help remove the cause of inadequate variability accounting and improve the exploration of the posterior landscape. Should these adjustments prove insufficient, we will consider alternative methodologies for leveraging the MAP estimate to achieve efficient approximate inference of our parameters.

Appendix | Gradient Calculation

Gradient for T-distribution of $p(\eta_t|H_{t-1}^{\intercal})$:

$$\log p(\eta_t|H_{t-1}^{\intercal})) \propto \frac{-(\nu_{t-1}+p)}{2} \log\left(1+\frac{1}{\nu_{t-1}}(\eta_t - f_t)^{\intercal}(q_t\Xi_{t-1})^{-1}(\eta_t - f_t)\right) \tag{.1}$$

Letting $L = 1+\frac{1}{\nu_{t-1}}(\eta_t - f_t)^{\intercal}(q_t\Xi_{t-1})^{-1}(\eta_t - f_t)$, we would like to calculate $\frac{d\ log|L|}{d\eta_t}$:

$$\frac{d\ log|L|}{d\eta_t} = \frac{1}{L}\ \frac{dL}{d\eta_t} \tag{.2}$$

$$\begin{aligned} dL &= d\left(\frac{1}{\nu_{t-1}}\left[(\eta_t - f_t)^{\intercal}\ X^{-1}\ (\eta_t - f_t)\right]\right) \\ &= \frac{1}{\nu_{t-1}}\left[d\eta_t^{\intercal}(X^{-1}\eta_t - X^{-1}f_t) + (\eta_t^{\intercal}X^{-1} - f_t^{\intercal}X^{-1})d\eta_t\right] \\ &= \frac{1}{\nu_{t-1}}\left[(\eta_t^{\intercal}X^{-1} - f_t^{\intercal}X^{-1})d\eta_t + (\eta_t^{\intercal}X^{-1} - f_t^{\intercal}X^{-1})d\eta_t\right] \end{aligned} \tag{.3}$$

$$\frac{dL}{d\eta_t} = \frac{2}{\nu_{t-1}}(\eta_t^{\intercal} - f_t^{\intercal})X^{-1} \tag{.4}$$

where $X = (q_t\Xi_{t-1})$

Bibliography

Blei, D. M. and Lafferty, J. D. (2006). Dynamic topic models. In *Proceedings of the 23rd international conference on Machine learning*, pages 113–120.

Cargnoni, C., Müller, P., and West, M. (1997). Bayesian forecasting of multinomial time series through conditionally gaussian dynamic models. *Journal of the American Statistical Association*, 92(438):640–647.

Carpenter, B., Gelman, A., Hoffman, M. D., Lee, D., Goodrich, B., Betancourt, M., Brubaker, M. A., Guo, J., Li, P., and Riddell, A. (2017). Stan: A probabilistic programming language. *Journal of statistical software*, 76.

David, L. A., Materna, A. C., Friedman, J., Campos-Baptista, M. I., Blackburn, M. C., Perrotta, A., Erdman, S. E., and Alm, E. J. (2014). Host lifestyle affects human microbiota on daily timescales. *Genome biology*, 15:1–15.

Glynn, C., Tokdar, S. T., Howard, B., and Banks, D. L. (2019). Bayesian analysis of dynamic linear topic models.

Laine, M. (2019). *Introduction to Dynamic Linear Models for Time Series Analysis*, page 139–156. Springer International Publishing.

Prado, R. and West, M. (2010). Time series modelling, inference and forecasting. *AMS, University of California, Santa Cruz ISDS, Duke Universit tsbook. dvi (hughchristensen. com)*.

Quintana, J. M. and West, M. (1987). An analysis of international exchange rates using multivariate dlm's. *Journal of the Royal Statistical Society Series D: The Statistician*, 36(2-3):275–281.

Silverman, J. D., Durand, H. K., Bloom, R. J., Mukherjee, S., and David, L. A. (2018). Dynamic linear models guide design and analysis of microbiota studies within artificial human guts. *Microbiome*, 6:1–20.

Silverman, J. D., Roche, K., Holmes, Z. C., David, L. A., and Mukherjee, S. (2022). Bayesian multinomial logistic normal models through marginally latent matrix-t processes. *The Journal of Machine Learning Research*, 23(1):255–296.

West, M. and Harrison, J. (2006). *Bayesian forecasting and dynamic models.* Springer Science & Business Media.