sBG-MCMC

Aaron D. Ramsey

April 2024

1 Introduction

In contractual settings, understanding customer attrition is crucial. Arguably, it is the duty of the manager to harness the full potential of attrition rates, given the privilege of its observation. Supplementing attrition is its counterpart, retention, commonly employed in calculating customer tenure and lifetime value. These metrics, vital for businesses, must be completely owned and understood.

Commonly, retention (the proportion of customers retained between successive periods) is assumed to be constant across a cohort. This is differentiated from what we will call the survival rate—the proportion of the total population alive in a given period. This *good-enough* approach, however, leads to a gross underestimation of tenure and value. At first glance, it may seem logical, but a more thorough examination reveals its weaknesses. A constant rate of retention requires a uniform attrition rate among a cohort's customers, irrespective of the observation period. This, however, defies intuition. It seems implausible that a customer who has resubscribed to Netflix for the 30th time would have the same likelihood of leaving in the 31st period as a new subscriber in their first. Empirically, it is even shown retention rates increase over time.

Accordingly, some would argue that these customers are *getting better*, but this contradicts the nature of human behavior. People don't aspire to become better consumers; they are simply guided by their own, seemingly random, desires. From our perspective, the *good customers* had always been good; identifying them, however, required observation. Customers are drawn to try new products; and, if in agreement with their preferences, they continue that relationship with the business. Such preferences remain unchanged (assuming stationarity) by the mere act of trying a product. Just as a coin landing on tails to infinity will never happen, all customers will churn, particularly if an alternative product better aligns with their preferences or current necessities.

How can we ascertain what actions our customers will take, or more aptly, how do we discern the assortment of customers we possess? Each customer, acting independently, obscures the underlying factors influencing their behavior. Yet, we endeavor to do our best. Fortunately, we have a proxy for determining customers with similar behavior patterns—the duration of our relationship.

To harness this relationship/churn data and our beliefs of customer behavior, we will employ the shifted beta-geometric (sBG) distribution (Fader and Hardie, 2007). The sBG has a hierarchical structure. On the individual level—the lowest—we model customer tenure

using the geometric distribution, particularly the shifted variant. To briefly summarize, geometric distribution aligns with our understanding of customer attrition. It is parameterized by θ (churn propensity), which encodes the various unobservable influences impacting a customer's decision to churn—product preferences, personality traits, occupational hazards, etc. Given we know an individual's true θ, we can calculate the probability of that customer surviving through period t: $p(T > t|\theta)$.

One level higher, the customer's theta is itself characterized by a beta distribution. This distribution accounts for our intuition regarding customer heterogeneity. It represents the spectrum of θ values among our customers and is characterized by two hyperparameters, α and β. The versatile shapes of this distribution, and the corresponding retention curves they imply, can be examined through the interactive graph here[1].

A result of modeling fixed churn rates are increasing retention rates. It's an artifact of the drop-out effect of bad customers. Accordingly, it would be ill-advised to only consider retention rates when analyzing one's customer base. This is evident in the survival curve, which reveals that customer cohorts with high churn rates rapidly vanish, despite achieving significant retention in later stages. In contrast, cohorts with a lower churn propensity retain a substantial portion of their initial base while still attaining high retention. Typically, companies excessively inclined towards the higher-churn segment of the distribution either cease to exist or adapt, attracting more high-value customers.

2 Methodology

The motivation behind this analysis is straightforward. Having previously implemented the sBG employing Empirical Bayes, a compelling desire to explore a more comprehensive approach emerged. My methodology features the fully Bayesian interpretation of the model; and, to benchmark, I will include results from the Empirical Bayes method (one whose simplicity allows for implementation in Excel).

2.1 Bayesian Interpretations

It's crucial to understand the implications that a Bayesian interpretation induces on a model's architecture. Generally, we consider all unknown parameters as random variables with assigned probabilities for their possible values. Parallel with observed data, we can choose to imbue these parameters with our subjective beliefs, codified by prior distributions. This produces an updated estimate of our *belief* in model parameters—the posterior distribution. Choosing priors that are themselves parametric distributions induces a hierarchical structure (such as our sBG) complicating the posterior distribution. Commonly, the posterior has no standard form. To manage this, we will use methods such as random-walk Metropolis (RWM) and, more recently, Hamiltonian Monte Carlo (HMC). Using these tools we can more efficiently generate samples from complex posterior distributions. An alternative approach involves loosening certain strict Bayesian principles, as seen in the empirical Bayes method. For example, the joint posterior distribution for an individual can be calculated using Bayes'

[1]For printed versions, the url is https://aarondaelramsey.com/academics/statistics/sBG/#betaDist

rule:

$$\begin{aligned} p(\theta, \alpha, \beta|y) &= \frac{p(y|\theta, \alpha, \beta) \cdot p(\theta, \alpha, \beta)}{p(y)} \\ &= \frac{p(y|\theta, \alpha, \beta) \cdot p(\theta|\alpha, \beta) \cdot p(\alpha, \beta)}{p(y)} \\ &= \frac{p(y|\theta) \cdot p(\theta|\alpha, \beta) \cdot p(\alpha, \beta)}{p(y)} \end{aligned}$$

In fact, this joint distribution is precisely what the fully Bayesian approach utilizes. Here, we also present the expanded form of the evidence—$p(y)$.

$$p(\theta, \alpha, \beta|y) = \frac{p(y|\theta) \cdot p(\theta|\alpha, \beta) \cdot p(\alpha, \beta)}{\int\int\int p(y|\theta) \cdot p(\theta|\alpha, \beta) \cdot p(\alpha, \beta)\, d\alpha\, d\beta\, d\theta}$$

The integrals in the denominator are often intractable; and, unfortunately, given our model's structure, it is impossible to find a closed-form solution. However, the desired result, $p(\theta, \alpha, \beta|y)$, depends exclusively on the values of θ, α, and β. Therefore, removing any terms without such dependencies ensures the estimated values remain proportional to the original distribution. Having integrated out these parameters, the posterior's bottom term, the evidence, solely concerns y and can be removed. The result is a distribution exhibiting the same qualities as the original.

$$p(\theta, \alpha, \beta|y) \propto p(y|\theta) \cdot p(\theta|\alpha, \beta) \cdot p(\alpha, \beta)$$

If our interests were solely in estimating the thetas (individual churn rates) of our model, we could reduce complexity by integrating our proportional posterior over the population parameters α and β:

$$\begin{aligned} \int\int p(\theta, \alpha, \beta|y)\, d\alpha\, d\beta &\propto \int\int p(y|\theta) \cdot p(\theta|\alpha, \beta) \cdot p(\alpha, \beta)\, d\alpha\, d\beta \\ p(\theta|y) &\propto p(y|\theta) \cdot \int\int p(\theta|\alpha, \beta) \cdot p(\alpha, \beta)\, d\alpha\, d\beta \end{aligned}$$

Unfortunately, we find ourselves back where we started: without a closed form solution for this integral. A common solution for such issues is the Gibbs sampler. The Gibbs sampler is an iterative algorithm relying on the model parameters' conditional distributions. This method, part the broader Markov chain Monte Carlo family, is both simple to understand and implement. Unfortunately, for our specific model, the conditional distribution of α and β do not have a standard form, a prerequisite for the Gibbs sampler. The most popular extension addressing such concessions, Random-Walk Metropolis, uses proposed samples from known distributions in place of a conditional distribution. The Gaussian is often chosen for its symmetry. Proposed samples are then accepted or rejected according to an acceptance criterion. Another class of algorithms, another subset of MCMC, is Hamiltonian Monte Carlo. We will explore and apply both methods.

A more straightforward method is Empirical Bayes. It's facilitates statistical analysis/inference on data where the prior distribution is determined by the observed data—hence, empirical. This contrasts the traditional Bayesian approach where prior beliefs are

fixed before any data are observed. It is considered a good-enough approximation to the fully Bayesian treatment of a hierarchical model.

Rather than permitting the top-level model parameters to vary, they are instead fixed to their most likely values. Maximizing the marginal likelihood $p(y|\alpha, \beta)$ with respect to the population-level parameters allows one to find these values. Such procedure contributes to Empirical Bayes' other name—maximum marginal likelihood. This structure greatly reduces computational complexity allowing analysis to be easily performed in Excel. With the advent of efficient and effective computational techniques, fully Bayesian approaches have mostly replaced Empirical Bayes methods; however, its ease of use, interpretability, and predictive power can neither be overstated nor overlooked.

Once found, the values of α and β allow for the calculation of the conditional posterior distribution of θ. The parameters α and β are fixed points. From a Bayesian perspective, this is the same as conditioning on these parameters.

$$\begin{aligned} p(\theta|y, \alpha, \beta) &= \frac{p(y|\theta) \cdot p(\theta|\alpha, \beta)}{p(y|\alpha, \beta)} \\ &= \frac{p(y|\theta) \cdot p(\theta|\alpha, \beta)}{\int p(y|\theta) \cdot p(\theta|\alpha, \beta) d\theta} \end{aligned}$$

2.2 Data and Model

The example data consist of 1,000 customers and their corresponding churn periods, which have been censored at the 7^{th} period. That is, it is unknown when each customer surviving through the 7^{th} period churned. Each customer's observed churn period, denoted y_i, is assumed to follow a Geometric distribution with a corresponding parameter representing churn propensity, θ_i. This structure implies an independent, constant retention rate for each customer. Each θ_i is assumed to be drawn from the same Beta distribution, described by parameters, α and β. Such structure can be represented by the following formulas:

$$\begin{aligned} y_i &\sim \text{Geom}(\theta_i) \\ \theta_i &\sim \text{Beta}(\alpha, \beta) \end{aligned}$$

A helpful analogy for internalizing this structure is to imagine each customer within a cohort (in our example the original 1000 customers acquired in the same period) reaching into a bag of coins and selecting one. The probability of this coin landing on heads is their propensity to churn (θ_i). The mix of coins within the bag follows a Beta distribution; and, coin selection is only done once, at the beginning of the customer relationship. At each period, the customers flip their coins, and those whose coin lands on heads, churn. These constraints again underscore the two main assumptions of the model: each period, a customer decides to leave the firm with a personal, constant probability θ_i (stationarity) where each θ_i was generated by a Beta distribution at the cohorst inception. The Beta distribution is a convenient and intuitive choice as it exhibits conjugacy with the Geometric distribution and is bounded between 0 and 1.

In order for us to estimate these parameters, we need to incorporate the data with our model. Normally, it would involve calculating the product of likelihoods and prior distributions; however, recall that our data are right-censored at the 7th period. This structure

necessitates the use of two different likelihood formulas: one based on the geometric's pdf; the other, its survival function:

$$\begin{aligned}
p(\boldsymbol{\theta}, \alpha, \beta|\boldsymbol{y}) &\propto p(\boldsymbol{y}|\boldsymbol{\theta}, \alpha, \beta) \cdot p(\boldsymbol{\theta}, \alpha, \beta) \\
&= p(\boldsymbol{y}|\boldsymbol{\theta}, \alpha, \beta) \cdot p(\boldsymbol{\theta}|\alpha, \beta) \cdot p(\alpha, \beta) \quad \text{where } p(\alpha, \beta) \propto 1 \\
&= \left(\prod_{i=1}^{1000} (p(y_i|\theta_i, \alpha, \beta) p(\theta_i|\alpha, \beta)) \right) \cdot p(\alpha, \beta) \\
&\propto \prod_{i=1}^{1000} (p(y_i|\theta_i) p(\theta_i|\alpha, \beta)) \\
&= \prod_{i=1}^{759} \left(\theta_i (1-\theta_i)^{y_i - 1} \frac{\theta_i^{\alpha-1}(1-\theta_i)^{\beta-1}}{B(\alpha, \beta)} \right) \prod_{i=760}^{1000} \left((1-\theta_i)^7 \frac{\theta_i^{\alpha-1}(1-\theta_i)^{\beta-1}}{B(\alpha, \beta)} \right) \\
&= \left(\frac{1}{B(\alpha, \beta)} \right)^{1000} \left(\prod_{i=1}^{759} \left(\theta_i^{\alpha} (1-\theta_i)^{\beta + y_i - 2} \right) \prod_{i=760}^{1000} \left(\theta_i^{\alpha-1} (1-\theta_i)^{\beta+6} \right) \right)
\end{aligned}$$

It should be noted that I have chosen α and β to have a flat prior for simplicity. The result is $p(\alpha, \beta) \propto 1$. Additionally, it's important to note that I have hard-coded y_i for the censored cell at $t = 7$ in the second product. This is specific to this dataset, and I want to make it clear that this value doesnt change in the second product. Generally, the the survival function for the geometric is $p(T > t) = (1-\theta_i)^t$; however, this may cause confusion for some users. One can reparameterize the survival function to be one more than the last observed date (8 in our case) to make implementation in Python more straightforward. Additionally, the indices of the products are also hardcoded to values specific to our dataset. To generalize the posterior, a data-agnostic formula can be given by:

$$p(\boldsymbol{\theta}, \alpha, \beta|\boldsymbol{y}) \propto \left(\frac{1}{B(\alpha, \beta)} \right)^{N} \left(\prod_{i=1}^{N_u} \left(\theta_i^{\alpha} (1-\theta_i)^{\beta + y_i - 2} \right) \prod_{i=N_u+1}^{N} \left(\theta_i^{\alpha-1} (1-\theta_i)^{\beta + y_i - 1} \right) \right)$$

$$p(\boldsymbol{\theta}, \alpha, \beta|\boldsymbol{y}) \propto \left(\frac{1}{B(\alpha, \beta)} \right)^{N} \left(\prod_{i=1}^{N_u} \left(\theta_i^{\alpha} (1-\theta_i)^{\beta + y_i - 2} \right) \prod_{i=N_u+1}^{N} \left(\theta_i^{\alpha-1} (1-\theta_i)^{\beta + y_i - 2} \right) \right)$$

N represents the total number of customers in the dataset, N_u represents the number of customers existing in the uncensored cells, and the top and bottom equations respectively represent the two interpretations of the survival function. For clarity, I will be using the former definition in all subsequent formulae.

2.3 Random-Walk Metropolis

We now need to sample from our joint posterior. Given its complexity, we employ the Metropolis algorithm, a Markov chain Monte Carlo method, which simplifies sampling by conditioning on current parameter samples. In addition, it loosens the requirements of the Gibbs sampler—conditional distributions are allowed to have non-standard form.

The initial step is to determine the conditional posteriors of all our parameters. This is achieved by disregarding terms dependent on the parameters being conditioned on. Consequently, the resulting distribution is proportional to its true value. For example, the parameters $\boldsymbol{\theta}$ have conditional posterior distributions proportional to the following equations:

$$p(\theta_i|\theta_{j\neq i},\alpha,\beta,y) \propto \begin{cases} \theta_i^{\alpha}(1-\theta_i)^{\beta+y_i-2} & \text{for } 1 \leq i < 760 \\ \theta_i^{\alpha-1}(1-\theta_i)^{\beta+6} & \text{for } i \geq 760 \end{cases}$$

We recognize these as the functional forms of Beta distributions. We will see these again later when discussing empirical Bayes:

$$\theta_i|\theta_{j\neq i},\alpha,\beta,\boldsymbol{y} \sim \begin{cases} \text{Beta}(\alpha+1,\beta+y_i-1) & \text{for } 1 \leq i < 760 \\ \text{Beta}(\alpha,\beta+7) & \text{for } i \geq 760 \end{cases}$$

Similarly, by ignoring elements of the joint distribution not reliant on α and β, a joint conditional distribution can be calculated. Unfortunately, this results in a non-standard distribution, which will require the Metropolis step of our algorithm to find samples from:

$$p(\alpha,\beta|\boldsymbol{y},\boldsymbol{\theta}) \propto \prod_{i=1}^{1000}\left(\frac{\theta_i^{\alpha}(1-\theta_i)^{\beta}}{B(\alpha,\beta)}\right)$$

In an iteration of the Metropolis algorithm, samples are drawn from each conditional distribution based on existing samples—commonly referred to as current or previous samples, depending on perspective. Each time a sample for a parameter is accepted, it becomes the current sample for that parameter, which is then used in subsequent samplings for that iteration. The follow scheme illustrates the process of sampling using this algorithm.

1. Initialize the parameters $\boldsymbol{\theta}$, α, and β, the starting points in the state space, and choose a symmetric proposal distribution $q(x^*|x^{t-1})$ for your parameters generated by non-standard conditional distributions.

2. For each iteration $t = 1, 2, \ldots, T$:

   (a) Sample the parameters $\boldsymbol{\theta}$ using
   $$\theta_i|\theta_{j\neq i},\alpha,\beta,y_i \sim \begin{cases} \text{Beta}(\alpha+1,\beta+y_i-1) & \text{for } 1 \leq i < 760, \\ \text{Beta}(\alpha,\beta+7) & \text{for } i \geq 760 \end{cases}$$

   (b) Generate a proposal x^* from the proposal distribution $q(x^*|x^{t-1})$ for both α and β. Where σ_α and σ_β are tuning parameters for the algorithm. (See Appendix A.1 for information about non-symmetric proposal distributions)

      i. $\alpha^* \sim \text{Normal}(\alpha^{t-1}, \sigma_\alpha)$

      ii. $\beta^* \sim \text{Normal}(\beta^{t-1}, \sigma_\beta)$

   (c) Input the proposed values into their non-standard joint distribution and compare to the current values using a ratio r.

      i. $r = \frac{p(\alpha^*,\beta^*|\boldsymbol{y},\boldsymbol{\theta})}{p(\alpha^{t-1},\beta^{t-1}|\boldsymbol{y},\boldsymbol{\theta})}$

(d) We accept the proposed values with probability r. (i.e. when r is greater than u where u $\sim$ Uniform$(0, 1)$)

(e) If accepted, our current sample values are updated to the proposed values. Otherwise, the old samples remain.

Generally it it is computationally useful to work with the logarithms of our distribution. Utilizing the properties of logarithms, r can be represented as a difference of logarithms and then compared to $\log(u)$:

$$\begin{aligned}
p(\alpha, \beta|\boldsymbol{y}, \boldsymbol{\theta}) &\propto \prod_{i=1}^{1000} \left(\frac{\theta_i^{\alpha}(1-\theta_i)^{\beta}}{B(\alpha, \beta)} \right) \\
&= B(\alpha, \beta)^{-1000} \cdot \prod_{i=1}^{1000} \left(\theta_i^{\alpha}(1-\theta_i)^{\beta} \right) \\
\log(p(\alpha, \beta|\boldsymbol{y}, \boldsymbol{\theta})) &\propto \log(B(\alpha, \beta)^{-1000}) + \log\left(\prod_{i=1}^{1000} \theta_i^{\alpha}(1-\theta_i)^{\beta} \right) \\
&= -1000 \log(B(\alpha, \beta)) + \log(\theta_1^{\alpha} \cdot \ldots \cdot \theta_{1000}^{\alpha} \cdot (1-\theta_1)^{\beta} \cdot \ldots \cdot (1-\theta_{1000})^{\beta}) \\
&= -1000 \log(B(\alpha, \beta)) + \log(\theta_1^{\alpha}) + \ldots + \log(\theta_{1000}^{\alpha}) \\
&\quad + \log((1-\theta_1)^{\beta}) + \ldots + \log((1-\theta_{1000})^{\beta}) \\
&= -1000 \log(B(\alpha, \beta)) + \alpha \log(\theta_1) + \ldots + \alpha \log(\theta_{1000}) \\
&\quad + \beta \log(1-\theta_1) + \ldots + \beta \log(1-\theta_{1000}) \\
&= -1000 \log(B(\alpha, \beta)) + \alpha \sum_{i=1}^{1000} (\log(\theta_i)) + \beta \sum_{i=1}^{1000} (\log(1-\theta_i)) \\
&= \alpha \sum_{i=1}^{1000} (\log(\theta_i)) + \beta \sum_{i=1}^{1000} (\log(1-\theta_i)) - 1000 \log(B(\alpha, \beta))
\end{aligned}$$

$$\log(r) = \log(p(\alpha^*, \beta^*|\boldsymbol{y}, \boldsymbol{\theta})) - \log(p(\alpha^{t-1}, \beta^{t-1}|\boldsymbol{y}, \boldsymbol{\theta}))$$

2.4 Hamiltonian Monte Carlo

In most cases, using RWM to generate samples from a posterior distribution is effective, yet it isn't perfect. Namely, RWM can and will propose a disproportionate number of samples outside the typical set, relative to those points' actual likelihoods. Thankfully, the acceptance criteon in RWM ensures that these deviations do not distort the characteristics of the posterior. Higher rejection, however, does decrease the algorithm's ability to effectively explore the typical set. In multidimensional settings, the tuning parameter is set to achieve an approxiamate 23% acceptance rate. We see this first hand in the sBG. The algorithm struggles to diffuse through the posterior space; and, in fact, the auto-correlation is so high, roughly 800,000 samples are needed across our four chains to achieve an effective sample

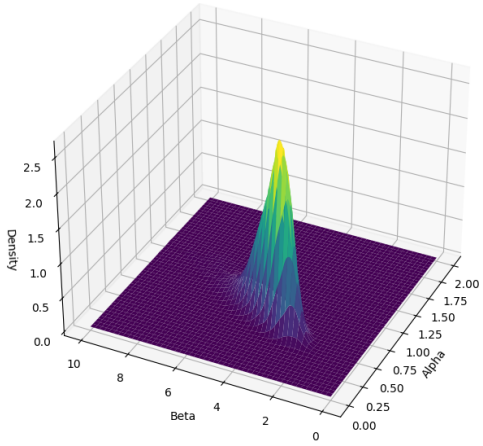
size of 1,000. This is neither reasonable nor practical, prompting the need to consider an alternative solution—Hamiltonian Monte Carlo.

Hamiltonian Monte Carlo (HMC) is a class of algorithms which uses Hamiltonian trajectories to generate samples from a posterior distribution. One of earliest implementations, Hybrid Monte Carlo, generated samples using Hamiltonian trajectories integrated over a fixed duration ($L * \epsilon$) to generate Metropolis proposals in place of a proposal distribution. While Hybrid Monte Carlo outperformed RWM, it still presented numerous inefficiencies and complications. Among those were the tuning of model parameters L (number of leapfrog steps) and ϵ (step size), supplying gradients for trajectory calculations, and the underutilization of information contained within the trajectory's intermediate points. Without significant HMC experience, tuning these parameters optimally is a non-trivial task. Moreover, many posterior distributions exhibit pathological behaviors with ill-set tuning parameters, especially in high-dimensional settings—an irony given such settings are one of its strengths.

In Gelman and Hoffman (2012) a new HMC variant named the No U-Turn Sampler (NUTS) was proposed. It featured automated control of tuning parameters, included information from trajectories' intermediate points, and offered a solution for the provision of gradients required by the algorithm. The specifics of implementing NUTS will be discussed later, so for now lets try to understand how this class of algorithms works.

2.4.1 Intuition

Intuition for NUTS/HMC can be more easily distilled from the construction of a physical system. Imagine a small particle—or, a ball for ease of visualization—positioned in $\mathbb{R}^3$ on a frictionless surface. The surface's height corresponds to our posterior distribution. Accordingly, the projection of this particle onto the xz plane represents the position in parameter space—known as the configuration in space mechanics literature. This visualization can affectionately be called Mount Likelihood.



In machine learning algorithms, the goal is to ascend/descend using a function's gradients to guide the algorithm to some maximum or minimum. That, however, is not our goal. We

want to incorporate information from the posterior's gradients to more efficiently explore all of the typical set rather than converging to the mode of the distribution. In simple terms, we want to integrate physics into our system—hence the construction of an analogous physical system.

As it stands, our physical construction does not afford us any conveniences. Placing our ball on the constructed surface would have the ball slide (no rolling without friction) away, having never explored regions of higher density. One solution could be to imbue the ball with some momentum in the direction of higher density; however, there are two considerations. Firstly, there would need to be a determined direction in which to give the momentum. Secondly, assuming the ball were sent towards a high density region, it would not remain there for a proportional length of time. Similar to before and without one's intervention, it would slide down the slope to disparate regions outside the typical set. As a result a stopping criterion of some sophistication would need to be applied. In either case, such interventions would introduce signicant bias into the system.

The solution, at a high level, is to invert the posterior; so, instead of looking at Mount Likelihood, we observe the Ravine of Rareness. The more likely a set of parameters, the more negative the height. This ensures, at any time, the particle tends toward a local minimum. There need not be any concern of the particle sliding to distant points in space. With this addressed, we need only place a particle on the surface and it will tend towards high-density regions. To simulate its path, Hamilton's equations need solved:

$$\frac{dq_i}{dt} = \frac{\partial H}{\partial p_i} \quad \text{and} \quad \frac{dp_i}{dt} = -\frac{\partial H}{\partial q_i}$$

$$\text{where} \quad H(\boldsymbol{q}, \boldsymbol{p}) = U(\boldsymbol{q}) + \frac{1}{2}\boldsymbol{p}^T M^{-1} \boldsymbol{p}$$

The daunting task of solving these equations must be achieved through numerical integration. That aside, we currently have only a naive understanding of what *physics* we are inducing on the system. By incorporating Hamiltonian dynamics, we have implicitly augmented our space with D additional parameters. For this example, there are now four total parameters—two for position ($\boldsymbol{q}$) and two for their corresponding momenta ($\boldsymbol{p}$). The bottom equation, known as the Hamiltonian, incorporates all four parameters in the augmented space. It is the sum of potential and kinetic energies contained within the system. The potential energy can be written as the height of the particle, which is generally given as the inverted log of the posterior. The choice of kinetic enegry is somewhat up to us, though care must be taken in its choice as not all kinetic energies are useful or valid. The standard choice being the Euclidean-Gaussian kinetic energy.

To further describe the physical implications of HMC, the sole forces acting upon the particle are gravity and the reaction from the posterior's surface—again no friction. Consequently, the total energy of the system is conserved along the path defined by these equations. From the perspective of our augmented system, these paths of constant energy define level sets of H. In our two-dimensional parameter space, a level set could be called a level curve or a contour line:

$$H^{-1}(E) = \{\boldsymbol{q}, \boldsymbol{p} \mid H(\boldsymbol{q}, \boldsymbol{p}) = E\}$$

In simpler terms, this represents the set of all inputs of the ordered pair $(\boldsymbol{q}, \boldsymbol{p})$—points in position-momentum phase space—that, when used to evaluate the Hamiltonian, yield a constant energy level E. The particle's movement, confined to a particular energy level, is periodic—think of a swinging pendulum. This behavior limits its ability to thoroughly explore the posterior distribution; and, more importantly, without updating the level set the particle exists on, it will only continue to traverse the same path each time we compute its trajectory. We need a procedure to transition the particle between level sets.

Fortunately, our choice of kinetic energy is a result of constructing a probability distribution over the momenta. This allows us to transition between the energy levels easily. Each iteration, after generating a parameter sample, we can resample the momenta from a Gaussian distribution, refreshing the particle's path. For a more in-depth exploration of these topics, Betancourt (2017) provides incredible insights and explanations for anyone interested in a more rigorous introduction to HMC.

To summarize the resultant structure, we begin by augmenting our model with auxiliary parameters representing the momentum, $\boldsymbol{p}$, of the particle in each dimension of the parameter space. We then separate the sampling process of the algorithm into two distinct steps. The first is a deterministic exploration of an energy level using Hamilton's equations. The second is a stochastic update or resampling of our momenta, positioning the particle on a new level set to explore. Instead of diffusing inefficiently through parameter space alone, we now only diffuse through the various level sets. After each sampling of momenta, the algorithm relies on a predetermined path to make large jumps from the initial point. The efficiency of the deterministic step can be controlled by tuning the L and ϵ parameters. Specifically, the NUTS algorithm dynamically sets L during trajectory construction. It uses a stopping criteon that halts the numerical integrator once the trajectory begins to turn around on itself—hence, the No U-Turn element of its name. Secondly, it employs a dual averaging algorithm (Nesterov, 2009) allowing for an optimal ϵ to be calculated during the warm-up period.

2.4.2 Implementation

Talk about using torch for autodiff. Discuss that I took a log transformation of the parameters to help with autocorr and ensuring alpha, beta > 0. Talk about how I used and implemented an optimal metric.

Using our structure and original NUTS implementation as a guide, I created a python program that estimates the posterior for the sBG. Aside from the number of chains and the desired samples per chain, the only required user input is the observed data given in the format of an array of churn numbers.

My implementation and all associated files can be found in this project's repository on GitHub contains my implementation. It deviates from the NUTS implementation and more closely resembles the scheme laid out by Betancourt—what is currently implemented by STAN.

Implementing HMC of any sort can be arduous. Before implementing, whether you are using bespoke code, STAN, pymc, etc. I would recommend in addition to the Betancourt and NUTS papers reading Neal (2011).

A result of allowing the sampler to move unrestricted in $\mathbb{R}^{D+1}$ using numerical integration

is the chance to generate samples outside the support of the Beta distribution. Namely, having α and $\beta < 0$. To avoid this I transformed the parameters using the natural logarithm. The logarithm is a non-linear transformation, so the volume of of the parameter space is distorted. We must account for this using a jacobian adjustment. In Appendix A.2 I cover my implementation of this reparameterization and the associated adjustments needed to be made to the posterior. A harmonius transformation choice can increase the efficiency of the exploration. By symmetry, a poor choice can lead to inefficient exploration. In any case, it is highly unlikely to always select a transformation that perfectly assists the algorithm. At times, as is the case for constrained variables, they are necessary but do not always allow for more efficient explorations. A solution is to change the metric we use to construct the parameter space.

Remaining loyal to our physical analogy, the metric is commonly referred to as the mass matrix. The metric essentially scales and twists a space. If well chosen, the it can decorrelate our parameters increasing the algorithms efficiency. I cover this implementation in Appendix A.3.

2.5 Empirical Bayes

My coverage of Empirical Bayes will be brief. I have distilled the vital components for the purpose of comparison. Fader and Hardie (2007) offers an in-depth derivation and handling of the Empirical Bayes implementation of the sBG within Excel.

Our goal is to find point estimates of α and β that maximize the likelihood of the data given the model. In our case, the model is the marginal likelhood function of the data—$p(y|\alpha, \beta)$.

$$\begin{aligned}
p(\theta|y, \alpha, \beta) &= \frac{p(y|\theta) \cdot p(\theta|\alpha, \beta)}{p(y|\alpha, \beta)} \\
&= \frac{p(y|\theta) \cdot p(\theta|\alpha, \beta)}{\int p(y|\theta) \cdot p(\theta|\alpha, \beta) d\theta} \\
\int_0^1 p(Y = y|\theta) \cdot p(\theta|\alpha, \beta) d\theta &= \int_0^1 \theta(1-\theta)^{y-1} \cdot \frac{\theta^{\alpha-1}(1-\theta)^{\beta-1}}{B(\alpha, \beta)} d\theta \\
p(Y = y|\alpha, \beta) &= \frac{1}{B(\alpha, \beta)} \cdot \int_0^1 \theta(1-\theta)^{\beta+y-2} d\theta \\
&= \frac{B(\alpha+1, \beta+y-1)}{B(\alpha, \beta)}
\end{aligned}$$

A similar process can be used to find $p(Y > y|\alpha, \beta)$

$$p(Y > y|\alpha, \beta) = \frac{B(\alpha, \beta+y)}{B(\alpha, \beta)}$$

These two formulas applied to the data will give us a resultant likelihood which we can maximize using the solver in excel. Specifically, we can apply $p(Y = y|\alpha, \beta)$ to the uncensored cells and $p(Y > y|\alpha, \beta)$ to the censored cell. Traditionally, the likelihood is a product of these

probabilities across the data. To ensure the lack of precision our computers have doesn't affect the calculations, it is common to take a log of the likelihood. This results in a sum of the log probabilities multiplied by the number of corresponding customers associated with it.

Additionally, if one were to track their customers and know when they return after churning as well as what their population parameters were i.e. what cohort they were from you can calculate the probability of churning in each subsequent period. Namely, our goal would be to calculate the value:

$$p(Y = y_{i,new}|y_i)$$

Knowing the conditional posteior of θ:

$$\begin{aligned}
p(\theta|y, \alpha, \beta) &= \frac{p(y|\theta) \cdot p(\theta|\alpha, \beta)}{p(y|\alpha, \beta)} \\
&= \frac{\theta_i \cdot (1-\theta_i)^{y_i - 1} \cdot \frac{\theta_i^{\alpha-1} \cdot (1-\theta_i)^{\beta-1}}{B(\alpha,\beta)}}{\frac{B(\alpha+1, \beta+y_i-1)}{B(\alpha,\beta)}} \\
&= \frac{\theta_i^{\alpha} \cdot (1-\theta_i)^{\beta+y_i-2} \cdot \cancel{B(\alpha, \beta)}}{B(\alpha+1,\, \beta+y_i-1) \cdot \cancel{B(\alpha, \beta)}} \\
&= \frac{\theta_i^{\alpha} \cdot (1-\theta_i)^{\beta+y_i-2}}{B(\alpha+1,\, \beta+y_i-1)}
\end{aligned}$$

We can calculate $p(Y = y_{i,new}|y_i)$:

$$\begin{aligned}
p(Y = y_{i,new}|y_i) &= \int_0^1 p(y_{i,new}|\theta_i) \cdot p(\theta_i|y_i) d\theta_i \\
&= \int_0^1 \theta_i \cdot (1-\theta_i)^{y_{i,new}-1} \cdot \frac{\theta_i^{\alpha} \cdot (1-\theta_i)^{\beta+y_i-2}}{B(\alpha+1,\, \beta+y_i-1)} d\theta_i \\
&= \frac{1}{B(\alpha+1,\, \beta+y_i-1)} \int_0^1 \theta_i^{\alpha+1} \cdot (1-\theta_i)^{\beta+y_i+y_{i,new}-3} d\theta_i \\
&= \frac{B(\alpha+2,\, \beta+y_i+y_{i,new}-2)}{B(\alpha+1,\, \beta+y_i-1)}
\end{aligned}$$

Additionally we could generalize this formula to take into account the entire history of a customer with us. Namely we can find $p(y_{n+1}|\vec{y})$. First we must find the posterior distribution of a customer after n (i.e. $y_n, y_{n-1}, \cdots, y_2, y_1 = \vec{y}$) observations of a reliationship with us

given we know what initial population they come from.

$$
\begin{aligned}
p(\theta|\vec{y}) &= \frac{p(\vec{y}|\theta) \cdot p(\theta|\alpha,\beta)}{\int_0^1 p(\vec{y}|\theta)p(\theta|\alpha,\beta)\,d\theta} \\
&= \frac{(\prod_{i=1}^n \theta(1-\theta)^{y_i-1}) \cdot \frac{\theta^{\alpha-1}(1-\theta)^{\beta-1}}{\cancel{B(\alpha,\beta)}}}{\int_0^1 (\prod_{i=1}^n \theta(1-\theta)^{y_i-1}) \cdot \frac{\theta^{\alpha-1}(1-\theta)^{\beta-1}}{\cancel{B(\alpha,\beta)}}\,d\theta} \\
&= \frac{\theta^n(1-\theta)^{\sum_{i=1}^n(y_i)-n} \cdot \theta^{\alpha-1}(1-\theta)^{\beta-1}}{\int_0^1 \theta^n(1-\theta)^{\sum_{i=1}^n(y_i)-n} \cdot \theta^{\alpha-1}(1-\theta)^{\beta-1}\,d\theta} \\
&= \frac{\theta^{\alpha+n-1}(1-\theta)^{\beta+\sum_{i=1}^n(y_i)-(n+1)}}{\int_0^1 \theta^{\alpha+n-1}(1-\theta)^{\beta+\sum_{i=1}^n(y_i)-(n+1)}\,d\theta} \\
&= \frac{\theta^{\alpha+n-1}(1-\theta)^{\beta+\sum_{i=1}^n(y_i)-(n+1)}}{B(\alpha+n,\beta+\sum_{i=1}^n(y_i)-n)} \\
&= Beta(\alpha+n,\beta+\sum_{i=1}^n(y_i)-n)
\end{aligned}
$$

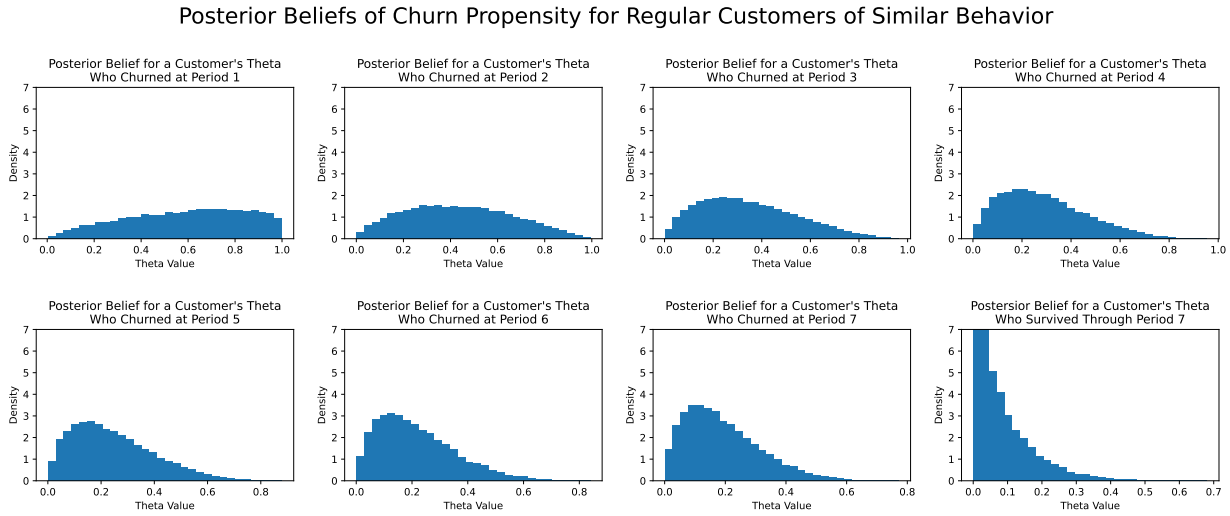
Accordingly, the predictive posterior for a customer with $\vec{y}$ observation where n is the length of that vector is

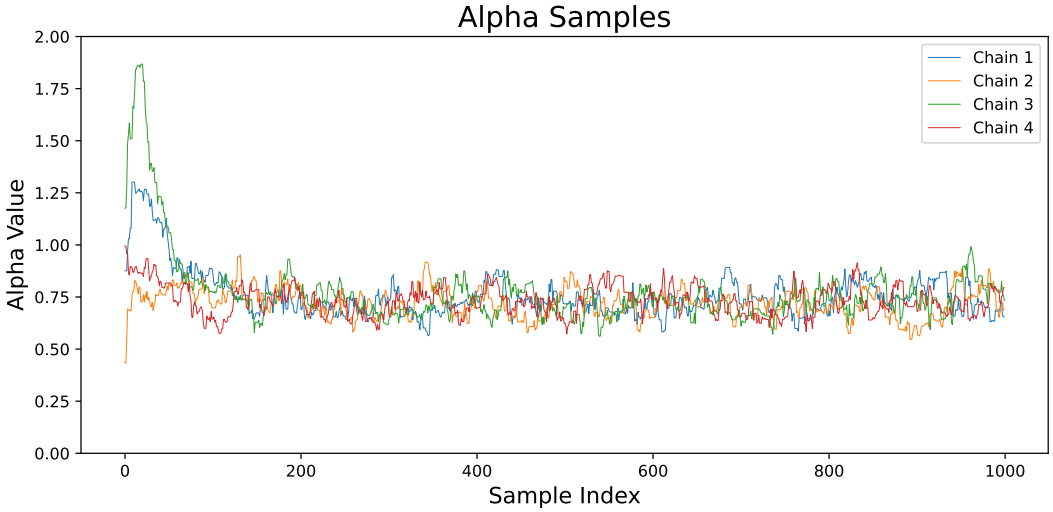
$$
\begin{aligned}
p(y_{n+1}|\vec{y}) &= \int_0^1 p(y_{n+1}|\theta) \cdot p(\theta|\vec{y})\,d\theta) \\
&= \int_0^1 \theta(1-\theta)^{y_{n+1}-1} \cdot Beta(\alpha+n,\beta+\sum_{i=1}^n(y_i)-n)\,d\theta \\
&= \frac{B(\alpha+n+1,\beta+y_{n+1}+\sum_{i=1}^n(y_i)-(n+1)}{B(\alpha+n,\beta+\sum_{i=1}^n(y_i)-n)}
\end{aligned}
$$

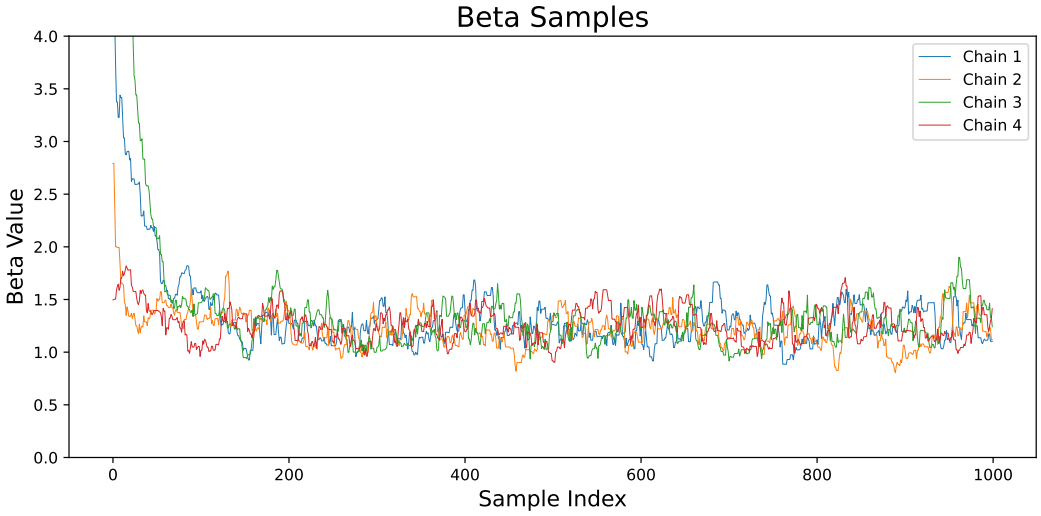
So, for a customer with observed behavior $\vec{y}$ coming from an initial population whose distribution of churn propensities is given by $Beta(\alpha,\beta)$ has a behavior during the $n+1^{th}$ relationship with our firm described by $p(y_{n+1}|\vec{y})$

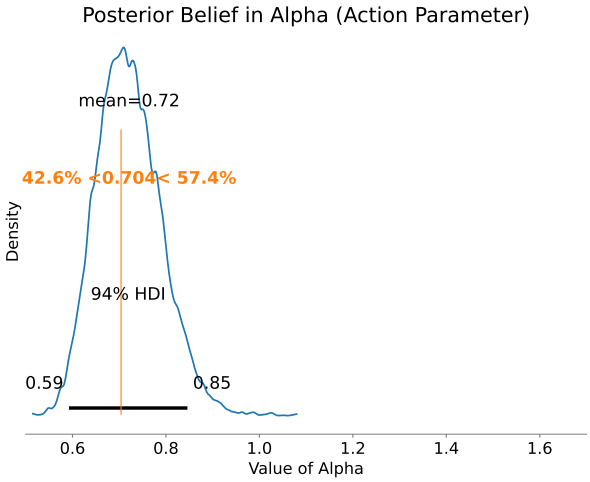
3 Results and Analysis

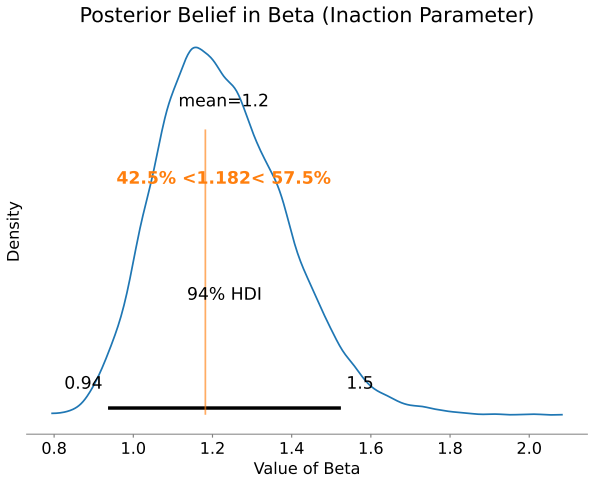
Theta posertiors are cool but mostly useless for churned customers. Lets see if typing more words allows me to rectify this error.

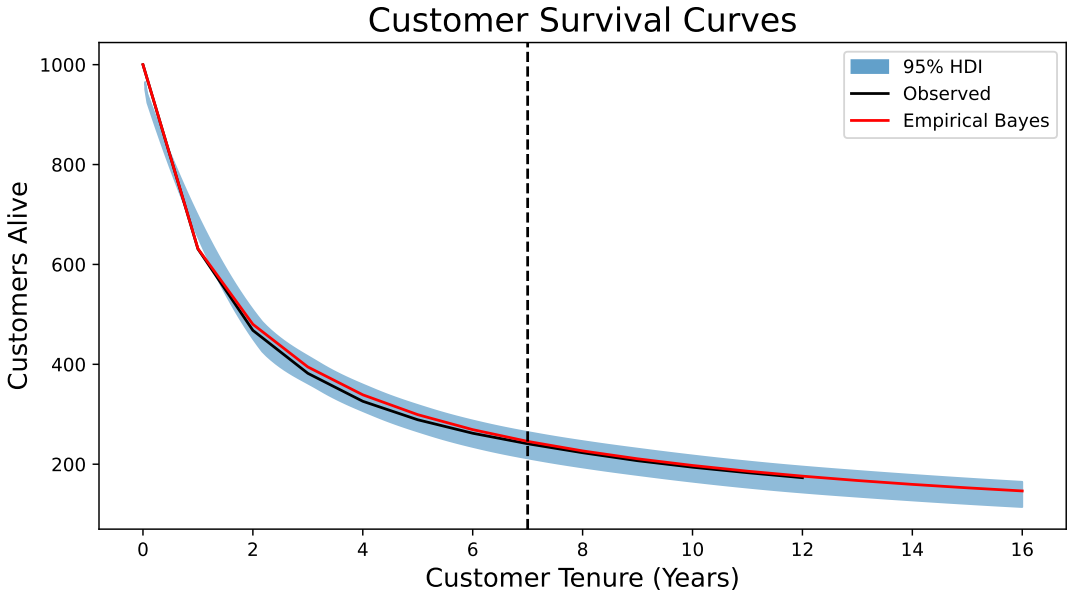

Posterior Beliefs of Churn Propensity for Regular Customers of Similar Behavior
Posterior Belief for a Customer's Theta Who Churned at Period 1
Posterior Belief for a Customer's Theta Who Churned at Period 2
Posterior Belief for a Customer's Theta Who Churned at Period 3
Posterior Belief for a Customer's Theta Who Churned at Period 4
Posterior Belief for a Customer's Theta Who Churned at Period 5
Posterior Belief for a Customer's Theta Who Churned at Period 6
Posterior Belief for a Customer's Theta Who Churned at Period 7
Postersior Belief for a Customer's Theta Who Survived Through Period 7
Density
Theta Value


Alpha Samples
Alpha Value
Sample Index
Chain 1
Chain 2
Chain 3
Chain 4

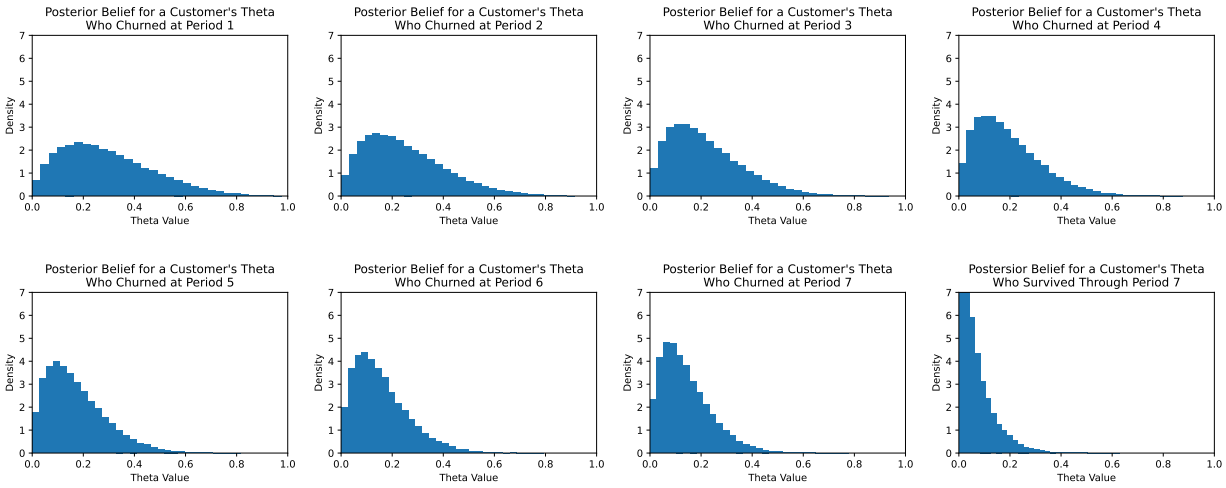

Beta Samples
Beta Value
Sample Index
Chain 1
Chain 2
Chain 3
Chain 4

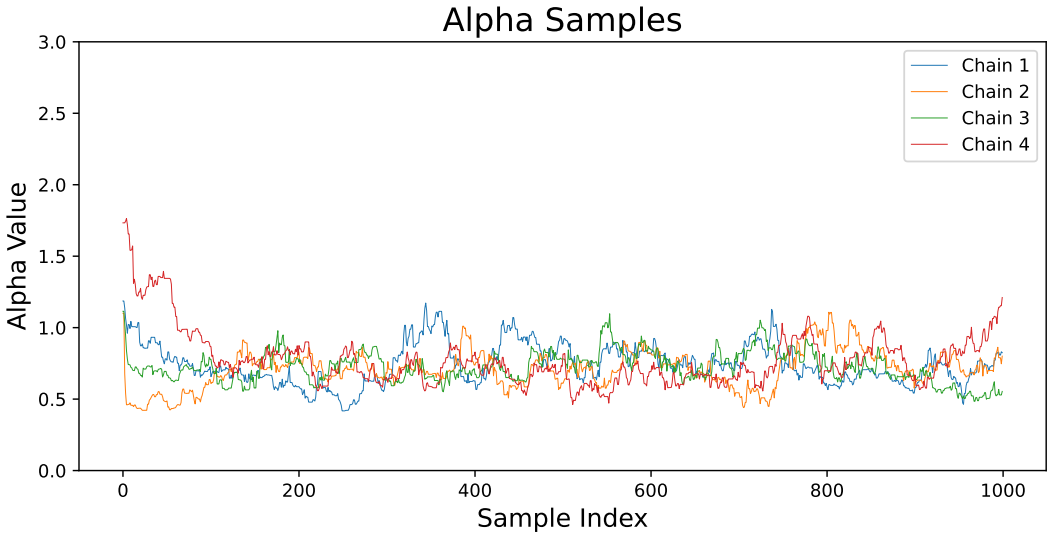


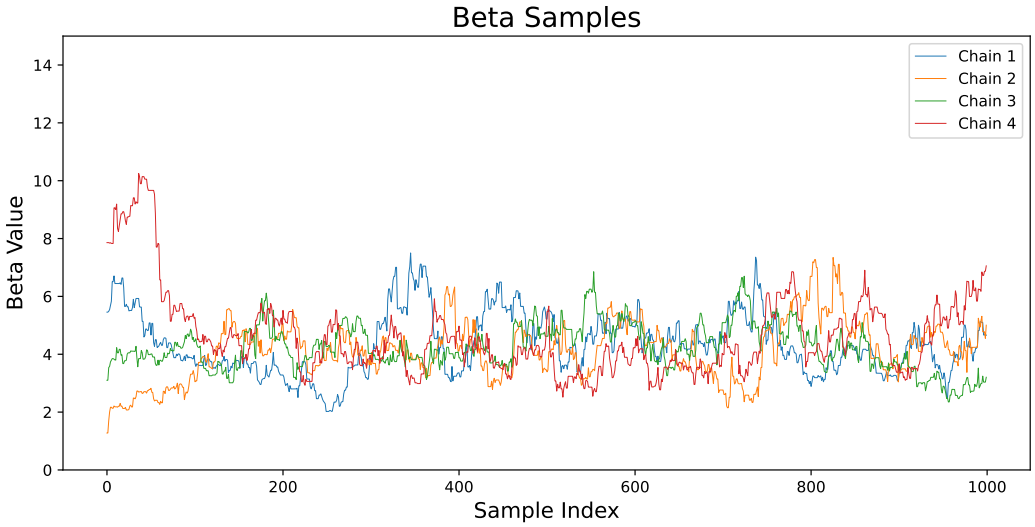


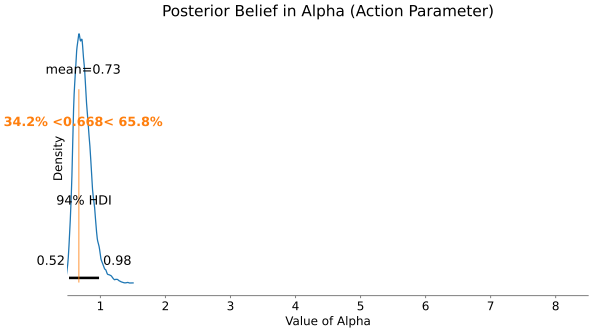


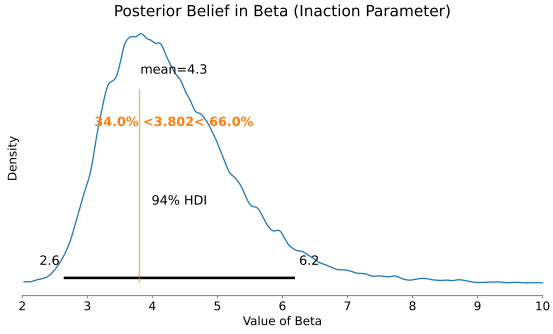
Posterior Beliefs of Churn Propensity for Regular Customers of Similar Behavior

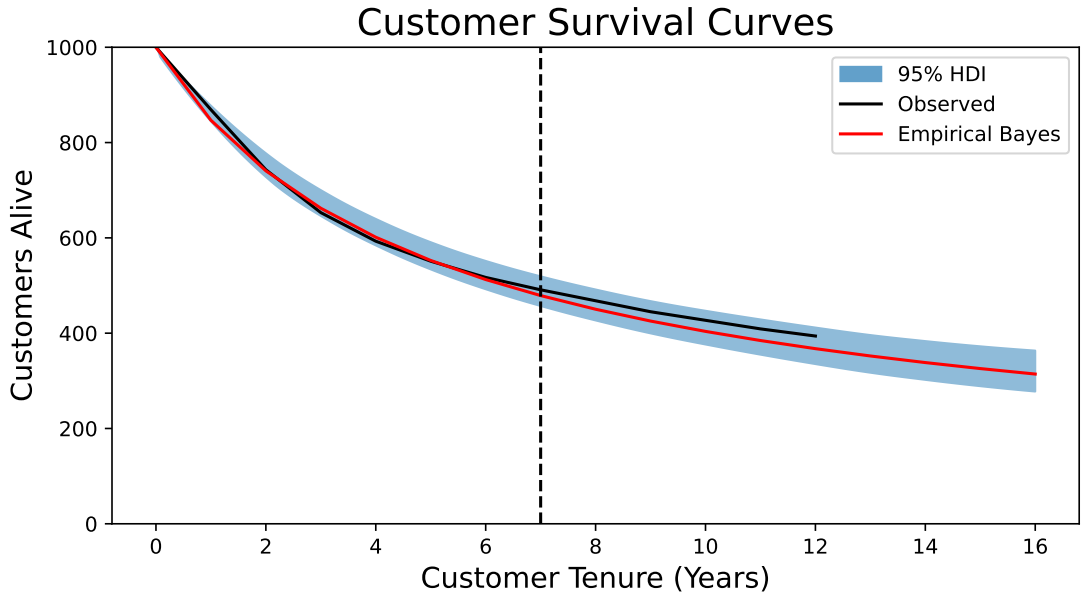



Alpha Samples
Chain 1
Chain 2
Chain 3
Chain 4
Alpha Value
Sample Index
3.0
2.5
2.0
1.5
1.0
0.5
0.0
0
200
400
600
800
1000


Beta Samples
Chain 1
Chain 2
Chain 3
Chain 4
Beta Value
Sample Index
14
12
10
8
6
4
2
0
0
200
400
600
800
1000


Posterior Belief in Alpha (Action Parameter)
mean=0.73
34.2% <0.658< 65.8%
Density
94% HDI
0.52
0.98
1
2
3
4
5
6
7
8
Value of Alpha


Posterior Belief in Beta (Inaction Parameter)
mean=4.3
34.0% <3.802< 66.0%
Density
94% HDI
2.6
6.2
2
3
4
5
6
7
8
9
10
Value of Beta



4 Conclusion and Considerations

I want to eventually implement a generalized version of the stopping criterion as well as a riemannian-gaussian version of the model.

Empirical Bayes

$$p(\theta|y) = \frac{p(y|\theta) \cdot p(\theta)}{p(y)} = \frac{p(y|\theta) \cdot p(\theta)}{\int (p(y|\theta)p(\theta))d\theta}$$

Fully Bayesian (Our Approach)

$$p(\theta|y) = \frac{p(y|\theta) \cdot \int\int p(\theta|\alpha,\beta)p(\alpha,\beta)\,d\alpha\,d\beta}{p(y)} = \frac{p(y|\theta) \cdot \int\int p(\theta|\alpha,\beta)p(\alpha,\beta)\,d\alpha\,d\beta}{\int\int\int p(y|\theta)p(\theta|\alpha,\beta)p(\alpha,\beta)\,d\alpha\,d\beta\,d\theta}$$

A Appendix

There are many techniques and variants of MCMC methods. As we have seen, HMC can be especially useful when dealing with problematic posteriors; however, there are still times when additional techniques must be utilized. This appendix holds the few that I have implemented for this specific use case. It is not an exhaustive list, or even necessary, but those on similar paths to mine will find them well-suited for their needs.

A.1 Metropolis-Hastings Ratio

The method suggested in section 2.3 (random-walk Metropolis) is inadequate when using non-symmetric proposal distributions. Instead of an unmotivated correction, I will loosely introduce the mathematical underpinnings of MCMC methods. Specifically, I will derive the updated acceptance ratio required of such proposals. For those interested only in the result, it can be located at the end of Appendix A.1.1.

At its core, MCMC methods combine the repeated random sampling of Monte Carlo simulation with the structure of a Markov chain, providing an acceptable structure to run the repeated *experiments*.

In general, Monte Carlo methods take advantage of some underlying structure whose characteristics are initially unknown, but, through repeated experiments can be calculated. For example, rolling two dice. Assuming we knew how they were distributed, we could easily calculate the probability of seeing each possible sum; however, what if the dice were non-standard, and no statements had been made about their propensities?

We could roll the dice (a single experiment) and record the sum. Repeating this experiment many times, the relative frequencies would approximate the actual probabilities for the sums given the simulation was ran for a sufficient length of time. Importantly, we have created a good understanding (the probabilities) of an inherently uncertain process (rolling a pair of random, possibly non-standard, dice). Such processes can be used to approximate π as well as many values that are artifacts of some underlying architecture.

In order to use Monte Carlo methods, a process to conduct the experiments is required; however, the motivation for finding a way to numerically approxiamate our posterior distribution is its lack of such process due to its complexity/dimensionality. Such impediments prevent the use of standard sampling techniques. Instead we create our own process which exhibits the same characteristics of the posterior. Explicitly, we create a Markov process that builds a chain of states whose empirical frequencies asymptotically converge to the target distribution.

A.1.1 Asymptotic Convergence

Suppose we have a Markov chain where X_n is the chain's state after n transitions. We can create a stopping time T_y where,

$$T_y = \min\{n \geq 1 : X_n = y\}$$

is the first return of the marvov chain to state y. In plain English, T_y is the minimum of the set of n where n is greater than or equal to 1 such that the state of the Markov chain at time n is equal to y. Also, let

$$\rho_{yy} = P_y(T_y < \infty) = P(T_y < \infty | X_0 = y)$$

be the probability of the markov chain returning to state y after having started at state y. If $\rho_{yy} = 1$ then state y is said to be recurrent. If it holds for all states then the Markov process is said to be reccurent. Assume $N_n(y)$ is the number of times the Markov chain visits state y through time n, and the Markov transition is both irreducible and reccurent then,

$$\frac{N_n(y)}{n} \to \frac{1}{E_y T_y}$$

Put concretely, if the assumptions of irreducibility and recurrence hold, then the empirical fraction of time spent in state y approaches the inverse of the expected value of T_y given the chain started in state y. Further, if a Markov process p is again irreducible and has a stationary distribution π such that $\pi p = \pi$ then,

$$\pi(y) = \frac{1}{E_y T_y}$$

and π is unique.

For finite-state Markov chains, it follows that ensuring p is irredcible, recurrent, and has a stationary distribution, π, is sufficient to show the asymptotic convergence of $\frac{N_n(y)}{n}$ to π. When generalizing to infinite-state Markov processes—the case for Metropolis and Metropolis-Hastings—we must ensure an additional condition, positive recurrence.

In infinite-state markov chains one can have a state x that is recurrent (i.e. $P(T_x < \infty | X_0 = x) = 1$) but have $E_x T_x = \infty$. When this happens it is called null recurrence. As one can see if $E_x T_x = \infty$ then $\pi(y) = \frac{1}{E_y T_y}$ would be 0 at that state. Fortunately, to prove positive recurrence we need only show that p is irreducible and that its stationary distribution π exists.

Proving detailed balance ensures the existence of a stationary distribution. From there, if we prove that p is irreducible, then we prove that all states of the Markov chain are positive recurrent. It follows that the empirical limiting fraction of time spent in each state asymptotically converges to π.

A.1.2 Detailed Balance

Commonly, in introductory classes on stochastic processes, one is given a transition probability that describes a Markov chain—commonly, a matrix. One can then be expected to know how to find a stationary distribution π that satisfies $\pi p = \pi$. For processes defined by a transition matrix, this is the eigen vector v associated with the eigen value 1 scaled such that $\sum_{i=1}^{n} v_i = 1$.

Conversely, what we will do is assign the stationary distribution π to be equal to our posterior distribution. We will then use the detailed balance equation to construct a Markov process that is irreducible and by extension, positive recurrent. If we are able to construct such a process, then we can use the asymptotic convergence of relative frequencies to approxiamate the posterior. Specifically, the detailed balance equation is given by this equation:

$$\pi(x) P(X_{n+1} = y | X_n = x) = \pi(y) P(X_{n+1} = x | X_n = y)$$

One can loosely think of this as representing that the flow from state x to state y is equal to the flow from state y to state x given one begins at the stationary distribution π. Also it is important to remember that the keeping detailed balance is the way we ensure that the Markov transition process has a stationary distribution. Since we are constructing such transition process ourselves, we must be careful to not violate detailed balance in doing so. To simplify any following expressions, we will take $P(X_{n+1} = y | X_n = x)$ to be $p(y|x)$. Additionally, as we are enforcing $\pi(x)$ to be the posterior distribution, I will represent it as $p(x)$. Accordingly, detailed balance can be shown by the equation:

$$p(x) p(y|x) = p(y) p(x|y)$$

Rearranging the equation we can see that,

$$\frac{p(x)}{p(y)} = \frac{p(x|y)}{p(y|x)}$$

Adhering to the structure of the Metropolis algorithm we will propose a value y from a proposal distribution $g(y|x)$ and then accept it with probability $A(y|x)$. So the probability of transition to state y given we are in state x is:

$$p(y|x) = g(y|x) \cdot A(y|x)$$

Inserting this into the detailed balance equation we have:

$$\frac{p(x)}{p(y)} = \frac{g(x|y) \cdot A(x|y)}{g(y|x) \cdot A(y|x)}$$

rearranging we can have:

$$A(y|x) = \frac{g(x|y) \cdot p(y)}{g(y|x) \cdot p(x)} \cdot A(x|y)$$

At this point, the requirement of the Metropolis algorithm for $g(y|x)$ to be symmetric allows it to be cancelled out, but for Metropolis-Hastings, it remains. In any case, the proposal distribution is a free parameter that the user chooses. For any choice, detailed balance holds. The last step is now to choose an acceptance rate that maintains detailed balance. A common choice is:

$$A(x|y) = \min\left(1, \frac{g(y|x) \cdot p(x)}{g(x|y) \cdot p(y)}\right)$$

Plugging this into the detailed balance formula we can see that:

$$A(y|x) = \frac{g(x|y) \cdot p(y)}{g(y|x) \cdot p(x)} \cdot \min\left(1, \frac{g(y|x) \cdot p(x)}{g(x|y) \cdot p(y)}\right) = \min\left(1, \frac{g(x|y) \cdot p(y)}{g(y|x) \cdot p(x)}\right)$$

We can see that for any proposal either $A(y|x)$ or $A(x|y)$ is 1 and the other is equal to its associated ratio. For example, if the proposal y has much more likelihood than the current state x, then $\min\left(1, \frac{g(y|x) \cdot p(x)}{g(x|y) \cdot p(y)}\right)$ will evaluate to its second term, which is the recipricoal of the term on the outside of the minimum function. Then, $A(y|x)$ will evaluate to 1. The inverse is also true. Detailed balance, by our construction of transition probabilities, holds.

A.2 Re-parameterization

The HMC, who simulates trajectory paths on for a particle gliding on the surface of our posterior, uses a symplectic integrator called the leapfrog integrator. Being a simulated, imperfect path, care must be taken when one has constrained parameters. It is easy to constrain a parameter when coding an implementation of the algorithm; however, doing so without introducing bias can be difficult. It would be easier to allow the particle to travel unconstrained to and then transforming its coordinates to be within the constraints required by the posterior.

One of the the logarithm's characeteristics is to take some number $x > 0$ to the log-space existing on the entirety of the real line. This property would be greatly useful for the sBG's α and β parameters. From a practical perspective using the logarithm transform our parameters to the log-space to simulate and sample to then transform back to the

original space would be redundant. In fact, the computer/program doesn't know that our parameters are constrained or in a specific space. It only does what we have defined. We can simply define the initial space the parameters are in to be the log-space itself. All we need to do once is to ensure that we transform the parameters to correct space when sampling/calculating the posterior. The corresponding transformation for the logarithm is the exponential. Specifically we have the transformations:

$$\alpha = e^x$$
$$\beta = e^y$$

Where x and y are the corresponding coordinates in the log-space. We must also remember that the exponential function is not a linear transformation, so we must include a Jacobian correction to the posterior definition so the same volume of space accounts for the same proportion of density. The Jacobian matrix is represented by:

$$\mathbf{J} = \begin{pmatrix} \frac{\partial \alpha}{\partial x} & \frac{\partial \alpha}{\partial y} \\ \frac{\partial \beta}{\partial x} & \frac{\partial \beta}{\partial y} \end{pmatrix}$$

Our two transformations are independent so the resulting matrix is

$$\mathbf{J} = \begin{pmatrix} e^x & 0 \\ 0 & e^y \end{pmatrix}$$

To update the posterior distribution we need to multiply it by the determinant of $\mathbf{J}$. As we are taking the log of the posterior, we would need to add the log of the determinant of $\mathbf{J}$.

$$\begin{aligned} |\mathbf{J}| &= \begin{vmatrix} e^x & 0 \\ 0 & e^y \end{vmatrix} \\ &= e^x \cdot e^y - 0 \\ &= e^{x+y} \\ \ln |\mathbf{J}| &= x + y \end{aligned}$$

A.3 Mass Matrix/Metric

In many applications, posterior estimates of our model parameters will be correlated. If too large, trajectory constructions will be inefficient or even wrong. Even if a particular transformation chosen for an alternate reason decorrelates the parameters, its very unlikely to be perfect.

The current metric space is the one we are all familiar with—the standard Euclidean space. We can scale and rotate this space however we please to construct a any number of different (but still Euclidean) metric spaces. The most useful would be one in which the parameters are uncorrelated allowing for ease of trajectory constructions. This favorable

space can be defined by something called a metric. In the most simple terms, a metric space is a set of elements called points with some understanding/definition of distance between its them. This distance calculation is uniquely defined by the *metric* we define.

The ideal metric would be one whose inverse approxiamates the covariance matrix of the parameters.

From a practical standpoint, we are not constructing the desired space initially for the particle to exist in. We are updating the formulas that plays out the particles dynamics with the information contained within the metric. Specifically, we are updating how we sample new momenta and how we calculate kinetic energies. That is, the formulas remain unchanged, but the metric M is no longer the identity matrix.

For especially complicated metrics, one can further simplify the kinetic energy by transforming both the momenta and the position parameters using the square roots of the inverse metric and metric respectively. This is to avoid handling the metric at each leapfrog calculation, only needing to use it for back-transforming parameters.

A.3.1 Welford's Online Algorithm

More complicated HMC implementations can have thousands of parameters. Storing and then computing a covariance matrix could be slow and unwieldy, often resulting in numerical instability. Instead, it would be attractive to compute the variance interactively, on-the-fly. We can take advantage of Welford's online algorithm to do just that. (and online algorithm is simply computer science talk for computing quantities only as they are received and not needing to have all the data at once.)

Welford's algorithm updates the current mean and the current *estimate* of the variance with each additional sample of one's parameters. Specifically it keeps track of the estimate of the sum of squares total (SST). Each additional sample of the parameters updates the corresponding mean and estimated SST. Once we terminate the algorithm, we can produce biased and unbiased sample variance estimates by dividing by n and $n-1$ respectively.

show the M2,n here as well as the formula for updating the mean.

A.3.2 Windowed Adaption

mention how we copied this structure from stan and what the structure is.