SEQUENTIAL MONTE CARLO APPROACH TO INFERENCE IN BAYESIAN CHOICE MODELS

by

İlker Gündoğdu

B.S., Physics, Işık University, 2006

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Computational Science and Engineering

Boğaziçi University

2019

ACKNOWLEDGEMENTS

I wish to thank my supervisor Prof. Ali Taylan Cemgil for all his guidance and time. I also thank Gökhan Çapan for coming up with the model that this thesis is built upon and Ali Caner Türkmen for all his contributions and discussions we've had. Finally, this study couldn't have been possible without my dear wife, Özlem's constant support.

ABSTRACT

SEQUENTIAL MONTE CARLO APPROACH TO INFERENCE IN BAYESIAN CHOICE MODELS

Monte Carlo methods, such as Markov Chain Monte Carlo (MCMC) and Sequential Monte Carlo (SMC), have extensive use cases in probabilistic modeling and inference. They usually appear as a way of drawing samples from a distribution of interest, because even for a relatively small model, the target distribution may easily go out of the domain of standard probability distributions, rendering the analytical tools to be almost useless. Sampling methods have been used effectively in such cases.

In this thesis, we will be dealing with a probabilistic model that captures the interaction between a recommender system and its users and define a posterior distribution over the user's preferences. The model itself is actually very similar to a Dirichlet-Multinomial model, but it has completely different analytical properties. Although it is not the main purpose of this thesis, this fact also serves as a demonstration of how a slight change in a model may result in a problem which requires drastic changes in the methods of approach. We propose a Sequential Monte Carlo scheme, based on the resample-move algorithm with a Metropolis-within-Gibbs style move kernel, that targets the posterior distribution over the user's preferences. We also provide Stan implementations that target the same posterior distribution and use it for validation purposes. Then we investigate a recommender-user interaction mechanism based on the idea of Thompson sampling by simulating interactions.

ÖZET

BAYESCİ SEÇME MODELLERİNDE KESTİRİM PROBLEMLERİNE ARDIŞIK MONTE KARLO YAKLAŞIMI

Markov Zinciri Monte Karlo ve Ardışık Monte Karlo yöntemleri, istatistiksel modelleme ve kestirim problemlerinde yaygın olarak kullanılmaktadır. Standart olasılık dağılımlarından örnek üretmek için kullanılan analitik yöntemler, karmaşık dağılımlar söz konusu olduğunda işlevlerini yitirirler. Görece basit modeller için bile, model parametreleri üzerindeki sonsal dağılım, standart dağılımlar cinsinden ifade edilemeyebilir. Bu sebeple Monte Karlo yöntemlerine sıklıkla başvurulmaktadır.

Bu tez çalışmasında, ilk olarak, bir öneri sistemi ve kullanıcıları arasındaki etkileşim için bir istatistiksel model sunup, kullanıcıların tercihleri üzerinde bir sonsal dağılıma ulaşıyoruz. Önerilen model oldukça basit ve sezgisel olmakla birlikte, literatürde yaygın olarak bilinen Dirichlet-Multinomial modelinin bir varyantı gibi düşünülebilir. Bu tez çalışmasının birincil amacı olmasa da, bu durum, analizi kolaylıkla yapılan modeller için bile, varsayımlardaki küçük değişikliklerin, model analizinde kullanılan yöntemleri ne kadar değiştirebileceğine bir örnek oluşturmaktadır. Daha sonra, kullanıcı tercihleri üzerindeki sonsal dağılımdan örnek üretmek amacıyla, resample-move algoritmasını temel alan ve Gibbs-içinde-Metropolis stilinde bir hareket çekirdeği kullanan bir Ardışık Monte Karlo algoritması öneriyoruz. Önerilen algoritmanın geçerliliğini, aynı sonsal dağılımı hedef alan Stan uygulamasıyla test ediyoruz. Son olarak da Thompson Örneklemesi presibini temel alan bir öneri sistemi-kullanıcı etkileşimi senaryosunu simülasyonlarla inceliyoruz.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

K	Number of items that is available to the recommendation system
L	Presentation size
T	Number of interactions between the recommendation system and the user
x	Choice
Y	Presentation
α	Parameters of the Dirichlet distribution
γ_k	Number of times the item k was chosen
$\mu(Y)$	Number of times the subset Y was presented
θ	Model parameters
$\pi_t(\theta)$	Posterior distribution of θ conditioned on the presentations $Y_{1:T}$ and choices $x_{1:T}$

LIST OF ACRONYMS/ABBREVIATIONS

AD	Automatic Differentiation
CLT	Central Limit Theorem
ESS	Effective Sample Size
HMC	Hamiltonian Monte Carlo
LLN	Law of Large Numbers
MCMC	Markov Chain Monte Carlo
NUTS	No-U-Turn Sampler
SIMD	Single Instruction Multiple Data
SIS	Sequential Importance Sampling
SMC	Sequential Monte Carlo
SSM	State Space Models
i.i.d	independent, identically distributed

1. INTRODUCTION

Uncertainty is one of the most fundamental concepts in nature and probability has served as a natural way of quantifying it for centuries. *Probabilistic modeling* is the practice of using probability theory to conveniently take uncertainty into account. Although probability theory and the so called *inverse probability* has its roots in the works of many great names, such as Laplace and Gauss, the practice of probabilistic modeling is usually associated with another giant, Thomas Bayes. Thus, the ones that prefer using probabilistic models to analyze natural phenomena are called *Bayesians* and the methods themselves are classified as being *Bayesian*. This work aims to demonstrate the idea of modeling a problem in a probabilistic framework, along with some inference techniques in as much self-contained way as possible.

The main problem that this thesis focuses on stems from the domain of recommendation systems. As the name suggests, a recommendation system aims to present relevant options to its users based on previous interactions with that particular user, or any other user whose behavior have correlations with the user of interest. Çapan *et al.* (2019) proposed a probabilistic model to capture the interaction between such a system and its users, based on Luce's choice axiom. The model resembles a well-known one, called *Dirichlet-Multinomial*, which will be stated and demonstrated in the first chapter. Through this example we will also be able to demonstrate that how a slight modification to a well-known problem, which can be considered trivial in analytical sense, changes the approaches that one should take in order to come up with a useful solution. Basically, we will see that the solution is no longer admits a standard form. Therefore we will need to employ methods to deal with such distributions based on the so called *Monte Carlo* methods.

The term *Monte Carlo* has been coined by Stanislaw Ulam during his work on the Manhattan Project, due to heavy reliance on random numbers. In principle, the methods rely on repeatedly generating outcomes from a particular experiment. Then, one may compute an estimate of any real valued function defined on the space on

all possible outcomes of the experiment, by simply averaging the previously obtained results. Therefore, in order to apply a Monte Carlo method, one needs two simple ingredients. First, the possible outcomes of the experiment needs to be specified in terms of probabilities. Then of course one needs to be able to repeat the experiment and obtain realizations of all possible outcomes, also known as samples, from the distribution defined by the experiment specification. First ingredient corresponds to the practice of probabilistic modeling, and the second one is the ever growing collection of Monte Carlo methods. These two practices have complemented each other so well that they became almost inseparable in the last decades, thanks to the rapid increase in the computational capabilities of the personal computers.

In our particular case, we define a vector of positive real numbers to model user preferences. Without loss of generality we also assume that the components of the vector add up to one. Vectors satisfying these two conditions actually form a well-known set, the probability simplex. Since user preferences are unknown to the recommendation system, we assume a probability distribution over all such vectors, in other words we assign a likelihood to each particular element of the simplex. To this end, we will make use of Luce's choice axiom. Then probability theory provides a consistent way to evolve the distribution of interest via the so-called Bayes' rule.

But as already been suggested, this is only a half of the story. A probability distribution doesn't mean much all by itself. One usually needs to compute the expectation of some function with respect to the distribution of interest. This is actually where Monte Carlo methods come into play. There are two broad classes of Monte Carlo methods, called Markov Chain Monte Carlo (MCMC) and Sequential Monte Carlo (SMC). Our problem suggests a sequential approach by its construction. Therefore we will investigate an SMC scheme based on the so-called *Resample-Move Algorithm*. We will also make use of a particular MCMC algorithm via the probabilistic programming language Stan and use the results obtained from Stan to validate our sampling scheme.

1.1. Organization

Chapter 2 aims to provide the required background material and build up the related terminology and notation to be used throughout this thesis. First section introduces the idea of Bayesian modeling and inference while second section of the chapter presents the relevant concepts from the Monte Carlo methods literature. We will introduce the model that this thesis is based on in Chapter 3 and give some examples to demonstrate its certain properties. In Chapter 4 we will first introduce the probabilistic programming language Stan and describe how one might perform inference for the Dirichlet-Luce model using Stan. Then we will propose an alternative method for the same purpose based on Sequential Importance Sampling and resample-move algorithm. Chapter 5 aims to give some example through simulations, using Stan to validate our proposed sampler. We will also compare both samplers in terms of their computational cost. Finally we will give concluding remarks and possible future research directions in Chapter 6.

1.2. Contribution of This Thesis

Major contribution of this thesis is to develop a sequential sampling scheme based on the resample-move algorithm proposed by Gilks and Berzuini (2001) and applied to the so called *static models* by Chopin (2002). This sequential setting allows us to incorporate a presentations mechanism, therefore providing a framework for experimenting with various presentations strategies. We will make use of a presentations mechanism proposed by Çapan *et al.* (2019), which is based on Thompson sampling (Thompson, 1933; Russo *et al.*, 2018).

2. PRELIMINARIES

In this chapter we will introduce fundamental concepts that will be used in the rest of this work. In Section 2.1 we will mention Bayes' rule and provide two introductory examples. Although these examples are used mostly for demonstration purposes, they have a natural connection to our main problem. Therefore we will present them in a somewhat detailed manner. In Section 2.2, we will introduce the main idea behind Monte Carlo methods, and briefly mention Markov chains in order to make the connection to Markov Chain Monte Carlo.

2.1. Bayesian Statistics

2.1.1. Bayes' Rule

As the name suggests, this is the fundamental concept that the whole Bayesian inference literature revolves around.

Let X and Y be random variables; x, y be their realizations and $p(x)$, $p(y)$ be their distribution functions, respectively. By the definition of conditional probability we can write the joint distribution of two random variables X and Y as

$$p(x, y) = p(x \mid y)\ p(y) \tag{2.1}$$

Note the usual abuse of notation. Although all distribution functions in Equation 2.1 are denoted by p, they are clearly different functions. This notation is so common in Bayesian literature, it is best to adopt it here as well. Also note that, as it is usually clear from the context, no distinction is made between discrete and continuous random variables.

Similarly, the joint distribution in Equation 2.1 can also be written in terms of the reverse conditional distribution, $p(y \mid x)$.

$$p(x, y) = p(y \mid x)\ p(x) \tag{2.2}$$

Equating right hand sides of Equations 2.1 and 2.2 we obtain the so called *Bayes' Rule*.

$$p(x \mid y) = \frac{p(y \mid x)\ p(x)}{p(y)} \tag{2.3}$$

In probabilistic inference literature the more familiar form of Equation 2.3 is given by

$$p(\theta \mid x) = \frac{1}{Z}\ p(x \mid \theta)\ p(\theta) \tag{2.4}$$

where θ denotes the parameters of the model, x denotes the observations and Z denotes the normalizing constant, which in most cases is very difficult to compute.

We should also note the usual terminology at this point. The term on the left-hand side (LHS) of Equation 2.4, $p(\theta \mid x)$, is called the *posterior*, while the first term on the right-hand side (RHS), $p(x \mid \theta)$, is referred to as the *likelihood* and the last term, $p(\theta)$, is the *prior*.

The whole practice of Bayesian statistics consists of defining a likelihood for the observations, x, parametrized by θ and infer the posterior distribution of the parameters by means of using Equation 2.4. Obviously this simple process can get extremely complicated. While it is possible for some problems to obtain a closed form solutions for the posterior, usually it is not possible due to the desire of building more complicated models to capture the complexities of the phenomenon under investigation.

2.1.2. Beta-Binomial Model

The canonical example of Bayesian modeling is the so called Beta-Binomial model which is stated as follows:

Let θ be the probability of obtaining heads from a possibly biased coin toss. We don't know what θ is, therefore we aim to obtain a distribution over θ after observing some number of outcomes from tossing the particular coin.

In other words we are interested in the posterior distribution of θ conditioned on the observations, say x.

In order to make use of Equation 2.4 we need to define a *likelihood*, $p(x \mid \theta)$, and a *prior*, $p(\theta)$. Since introducing the prior is the responsibility of the modeler, Bayesian modeling is sometimes referred to as being *subjective*. Which obviously means that considering different prior distributions yields different results for the quantity of interest, θ. At first this may seem like a drawback, but it is in fact a beautiful mechanism which allows the modeler to state all the assumptions explicitly he or she makes in a consistent way.

2.1.2.1. Prior Distribution. We first state the prior distribution. There are many probability distributions studied in various contexts throughout the history. Since they are very well studied, they often serve as good starting points. *Beta distribution* is one such example.

Beta distribution is a family of probability distributions parametrized by two positive parameters, a and b, on the closed interval $[0, 1]$, and denoted by $\text{Beta}(a, b)$. Example distribution functions for several parameter pairs are given in Figure 2.1.

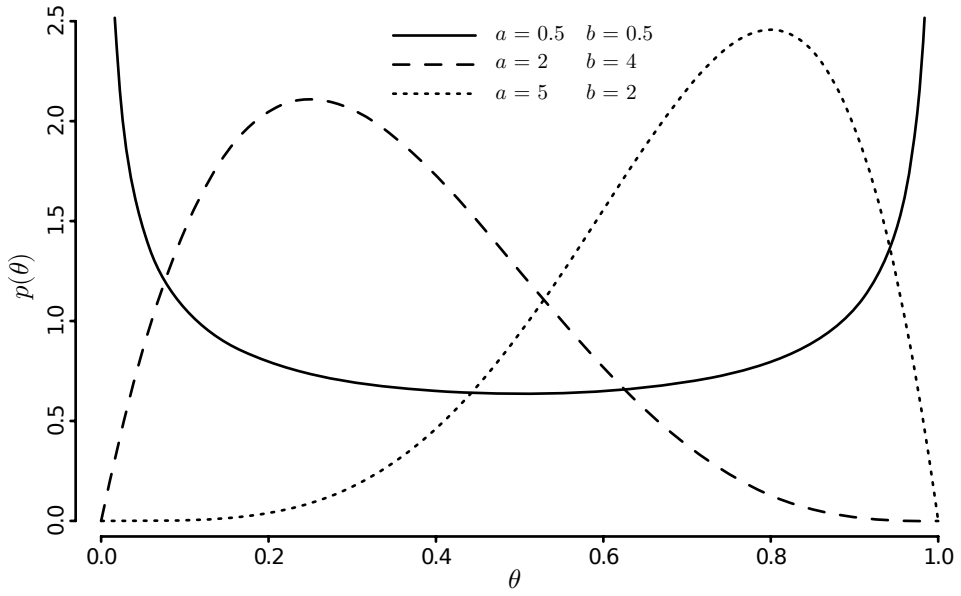

Figure 2.1. Beta PDF.

The density function of the Beta distribution is given by

$$p(\theta) = \frac{\theta^{a-1}\ (1-\theta)^{b-1}}{B(a,b)} \tag{2.5}$$

where $B(a,b)$ is the *Beta function*, which depends only on the parameters a and b. At this stage, it is modeler's duty to state a and b to reflect his or her prior belief in θ. This simply represents the uncertainty on the quantity of interest before conducting any experiments. Once we express our prior belief about θ by using appropriate a and b parameters, $B(a,b)$ is effectively a constant with respect to θ.

2.1.2.2. Likelihood. As the name suggests we use *Binomial likelihood*, which is defined to be the probability of getting x successes (heads in our example) out of N trials (coin tosses). In other words

$$p(x \mid \theta) = \binom{N}{x}\ \theta^{x}\ (1-\theta)^{N-x} \tag{2.6}$$

Substituting Equations 2.5 and 2.6 into Equation 2.4 we obtain

$$\begin{aligned} p(\theta \mid x) &= \frac{1}{Z}\binom{N}{x}\theta^x(1-\theta)^{N-x}\frac{1}{B(a,b)}\theta^{a-1}(1-\theta)^{b-1} \\ &= \frac{1}{\hat{Z}}\theta^{a+x-1}(1-\theta)^{N-x+b-1} \end{aligned} \tag{2.7}$$

where $\hat{Z} = \binom{N}{x}/(Z \times B(a,b))$, is the normalizing constant for the posterior. Note that x is the observation, therefore known. Thus, $\binom{N}{x}$ is constant with respect to θ. $\hat{Z}$ can be computed, but recognizing the unnormalized part as being another *Beta distribution*, namely $\text{Beta}(a+x, b+N-x)$, allows us to write the normalizing constant as $\hat{Z} = B(a+x, b+N-x)$.

For the sake of introduction, we took our time to explain the likelihood and prior choices in detail. But in general, a probabilistic model is expressed in a more compact form. For example, the Beta-Binomial model is expressed as

$$\begin{aligned} \theta &\sim \text{Beta}(a,b) \\ x &\sim \text{Bin}(N,\theta) \end{aligned}$$

where the symbol $\sim$ means "distributed as", N and x constitutes our *dataset*, hence are known beforehand, a and b are the parameters for the *Beta distribution* which represents our prior belief on the uncertainty of the quantity of interest, θ. Figure 2.2 shows an example inference where the prior belief is represented by $\text{Beta}(2,4)$. After making an observation, which is a single trial here, resulting tails, i.e. $N = 1$ and $x = 0$, the posterior distribution becomes $\text{Beta}(2,5)$. It can be visually validated that after observing another tails, our belief about the probability of getting a heads for this particular coin, represented by θ, is shifted towards zero slightly.

This example illustrates the basic procedure of Bayesian modeling and inference, which happens to have a very nice closed-form solution for the posterior. Also note that the prior and the posterior have the same form, namely they are both *Beta distributions* in this example. This is an instance of a more general concept called *conjugacy*. We

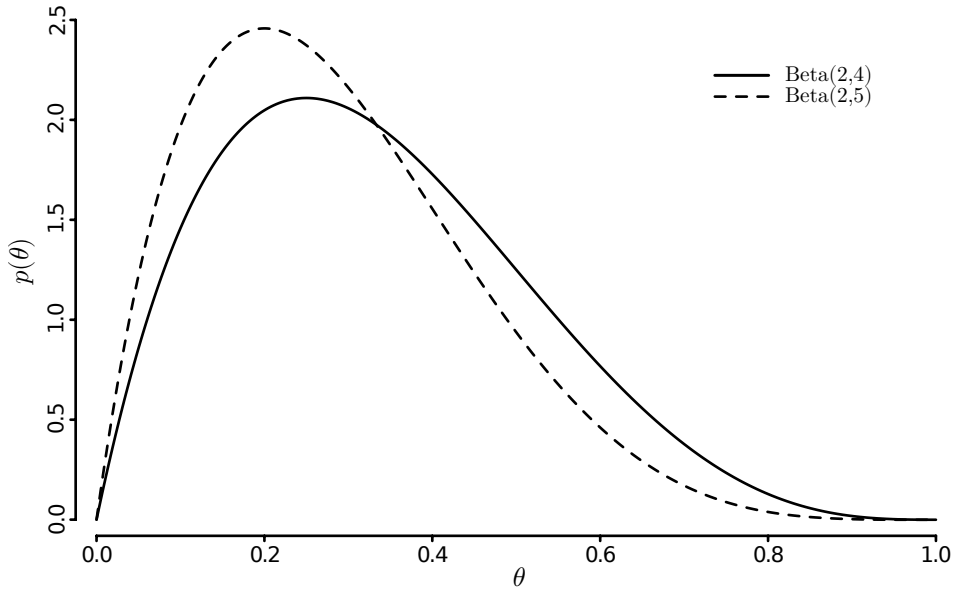

Figure 2.2. Prior and posterior densities with $a = 2$, $b = 4$, $N = 1$ and $x = 0$.

say that *Beta* and *Binomial* distributions are *conjugate* because of this fact. Because of their analytical convenience, conjugate pairs are often used, but of course they must be used with care as analytical convenience doesn't necessarily imply an appropriate model for the phenomenon of interest.

This section also served as an introduction to the terminology and notation commonly used in Bayesian modeling. Therefore we will be able to express and analyze the following model in a more compact fashion.

2.1.3. Dirichlet-Multinomial Model

This is a generalization of the Beta-Binomial model studied in the previous section and will serve as a motivation and a stepping stone to the main problem of this thesis.

Dirichlet distribution is the multivariate generalization of the beta distribution. While beta distribution is parametrized by two positive numbers, Dirichlet distribution is parametrized by a vector of positive real numbers, $\boldsymbol{\alpha}$, and usually denoted as $\mathrm{Dir}(\boldsymbol{\alpha})$.

The density function for $\text{Dir}(\boldsymbol{\alpha})$ is given by

$$p(\boldsymbol{\theta}) = \frac{1}{B(\boldsymbol{\alpha})} \prod_{i=1}^{K} \theta_i^{\alpha_i - 1} \tag{2.8}$$

where $\{\theta_i\}_{i=1}^{K}$ satisfying the conditions $\sum_{i=1}^{K} \theta_i = 1$ and $\theta_i > 0$ for all $i \in \{1, 2, \ldots, K\}$. The region defined by these conditions is called the standard $(K-1)$-*simplex* and will be denoted by Δ^{K-1}. The normalizing constant $B(\boldsymbol{\alpha})$ is called the *multivariate beta function* and is given by

$$B(\boldsymbol{\alpha}) = \frac{\prod_{i=1}^{K} \Gamma(\alpha_i)}{\Gamma(\sum_{i=1}^{K} \alpha_i)}$$

where $\Gamma(.)$ is the gamma function.

Dirichlet distribution is also the *conjugate prior* for the *Multinomial distribution* which we define next:

Multinomial distribution is the generalization of the *binomial distribution.* While binomial distribution models the number of successes in a given number of trials, multinomial distribution is used for trials with more than two outcomes, like a die with K sides. It therefore models the probability of the counts of K outcomes out of a given number of independent trials.

The probability mass function for the multinomial distribution parametrized by the number of trials N and the vector of probabilities $\boldsymbol{\theta}$ is given by

$$p(\mathbf{x}; N, \boldsymbol{\theta}) = \frac{N!}{\prod_{k=1}^{K} x_k!} \prod_{k=1}^{K} \theta_k^{x_k} \tag{2.9}$$

We can state the Dirichlet-Multinomial model as follows:

$$\begin{aligned}\boldsymbol{\theta} &\sim \text{Dir}(\boldsymbol{\alpha}) \\ \mathbf{x} &\sim \text{Mult}(N, \boldsymbol{\theta})\end{aligned}$$

Here, $\boldsymbol{\theta} = \{\theta_i\}_{i=1}^K$ is a vector of probabilities, where θ_i is the probability of obtaining outcome i for each trial, K is the number of possible outcomes and N is the number of trials. $\mathbf{x} = \{x_i\}_{i=1}^K$ is the vector of counts such that x_i is the number of times the outcome i is observed out of N trials.

The usual interest is again in the posterior, $p(\boldsymbol{\theta} \mid \mathbf{x})$ after observing some data $\mathbf{x}$. We can substitute Equations 2.8 and 2.9 into Equation 2.4 with combining all the constant terms to obtain

$$\begin{aligned}p(\boldsymbol{\theta} \mid \mathbf{x}) &\propto \prod_{k=1}^K \theta_k^{\alpha_k - 1} \prod_{k=1}^K \theta_k^{x_k} \\ &= \prod_{k=1}^K \theta_k^{\alpha_k + x_k - 1}\end{aligned} \tag{2.10}$$

Recognizing this as a density function for the Dirichlet distribution with parameter vector $\boldsymbol{\alpha}' = \{\alpha'_k\}_{k=1}^K$ with $\alpha'_k = \alpha_k + x_k$ eliminates the need to worry about the combined constant terms, and the *posterior* turns out to be another Dirichlet distribution. This is expected, since we have already stated that Dirichlet and multinomial distributions form a *conjugate pair*.

This brings us to the main question of this thesis: Dirichlet-Multinomial model assumes that the vector of observed counts, $\mathbf{x}$, follows a multinomial distribution parametrized by the probability vector $\boldsymbol{\theta}$. What happens if we restrict the possible outcomes of the multinomial distribution to a subset of all the outcomes?

More precisely, let Y be any non-empty subset of the set $\{1, 2, \ldots, K\}$ of fixed size, say L, where $L < K$. Define a *restricted multinomial distribution* parametrized

by the probability vector $\boldsymbol{\theta}' = \{\theta'_i\}_{i=1}^{L}$, with $\theta'_i = \dfrac{\theta_{Y_i}}{\sum_{j \in Y} \theta_j}$. In other words, a restricted multinomial distribution is derived from a base multinomial distribution by restricting the trials to only a subset of outcomes by keeping their relative probabilities unchanged, and normalizing appropriately.

This might sound like a superfluous question at first, but it is actually very common situation occurring naturally in *recommendation systems*. The sole reason for the existence of recommendation systems is the massive number of *alternatives*, so that it is impossible for a *user* to scan all of them and make a *choice*. Therefore recommendation systems has to come up with a *presentation*, which is a subset of all the alternatives, forcing users to make a choice among them. A very familiar example would be movie recommendation systems. Although it should be pretty clear, we will precisely define terminology introduced in this paragraph later, when we start analyzing the actual problem.

2.1.4. Summary

Before moving on, it would be helpful to summarize the general framework of a Bayesian modeling and inference problem.

(i) Identify the quantity of interest, e.g. probability of observing a heads from a coin toss. These are usually called the *parameters* of the problem.

(ii) Decide an appropriate prior distribution for the parameters before making any observations. For example, if the modeler is pretty confident that the coin to be fair, then the prior distribution should be concentrated around $1/2$.

(iii) Express the probability of observations in terms of the parameters. For example, how likely would it be to observe 3 heads in 10 trials when the probability of getting heads is θ.

(iv) Using the Bayes' Rule, write down the posterior in the form

$$p(\theta \mid x) = \frac{1}{Z}\phi(\theta) \tag{2.11}$$

where θ represents usually a vector of parameters, x represents the dataset, or observations. $\phi(\theta)$ is usually a multivariate function that can be evaluated at any given point in the parameter space and Z is the appropriate normalizing constant which makes the posterior a proper probability distribution.

Although this framework is very simple and intuitive, in practice there are huge challenges one may face at almost every step. Even the first step, which can be interpreted as "asking the right question" is a challenging one. But this thesis will be mostly concerned with the issues related to the last item, *the posterior distribution.*

At this point we should be very suspicious about how often it is the case where we have nice, closed-form solutions to actual problems, the ones that live outside the domain of the so called "toy problems". Unfortunately, other than a few purposely designed models, the answer is almost always "*never*". But this doesn't mean that it is impossible to find "good enough" solutions to these problems. Among many other approximations, the so called Monte Carlo methods provide a collection of tools that can be used to tackle almost any problem. Of course at the cost of possibly expensive computations. Next section aims to give the idea behind Monte Carlo methods and introduce the relevant concepts that will be used to approach the main question of this work.

2.2. Monte Carlo Methods

Monte Carlo methods are used to compute expectations in the form

$$J = \left\langle f(\theta) \right\rangle_{p(\theta)} = \int f(\theta) p(\theta) d\theta \tag{2.12}$$

by using finite number of samples from the distribution $p(\theta)$. The expectation in Equation 2.12 can be approximated by

$$\hat{J} = \frac{1}{N} \sum_{i=1}^{N} f(\theta^{(i)}), \quad \text{where } \theta^{(i)} \sim p(\theta) \tag{2.13}$$

Law of large numbers (LLN) and *central limit theorem* (CLT) guarantees that $\hat{J}$ is an unbiased estimator for the expectation in Equation 2.12 and the standard deviation of $\hat{J}$ is $\sigma/\sqrt{N}$ regardless of the dimension of θ, where σ is the standard deviation of the distribution $p(\theta)$.

Although it is not the most efficient method to approximate π but personally I can't resist using this example because of its simplicity. Consider a square enclosing a quarter-circle, with its center is on one of the corners of the square, as shown in Figure 2.3. Now consider throwing a dart into the unit square in such a way that it is equally likely to hit any point in the square. Observe that the probability of the dart hitting a point inside the quarter-circle is the ratio of the area of the quarter-circle to the area of the unit square, which is simply $\pi/4$. Now we can estimate this quantity by using Equation 2.12 with θ denoting a point in the unit square; $p(\theta)$, uniform distribution over the unit square and $f(\theta)$, an indicator function $\mathbb{I}(\theta) = 1$ if θ is in the quarter-circle, 0 otherwise. Therefore, by Equation 2.13 the estimator for $\pi/4$ can be computed by drawing random samples from $p(\theta)$ and computing the sample average. With $N = 1000$ samples shown in Figure 2.3, we obtain an estimate $\hat{\pi} = 3.124$. Increasing the number of samples to $N = 10000$, we obtain another estimate of $\hat{\pi} = 3.1304$. Of course these are just single realizations of the estimator, meaning that different runs with the same number of samples yield different results. The discrepancy is governed by the CLT. The more samples we use, the more likely that our estimator to be closer to the exact value of π. Repeating the same experiment with $N = 1.000.000$ samples, we obtain $\hat{\pi} = 3.1436$. Of course there are much more efficient ways to compute π to arbitrary precision, but this method demonstrates that one should always keep in mind that even the simplest methods can be useful. One may not need a super-sophisticated, so called state-of-the-art method, if the problem doesn't require that sophistication.

2.2.1. Importance Sampling

Despite its name, importance sampling is not a sampling method by itself. It is a method for computing expectations by using Monte Carlo estimation when $p(\theta)$ in

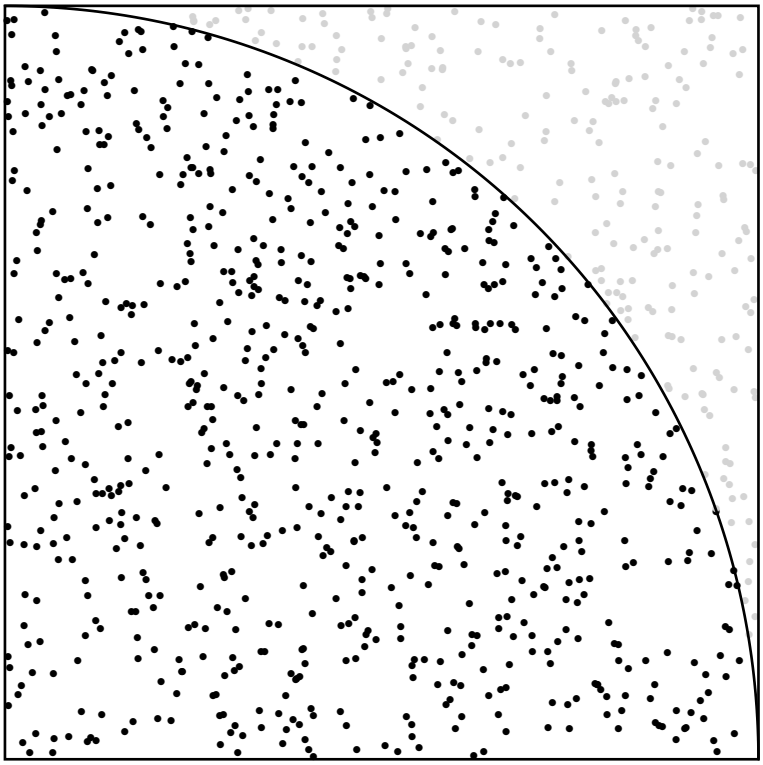

Figure 2.3. Monte Carlo estimation of π with 1000 samples.

Equation 2.12 is not easy to generate samples from. In that case Equation 2.12 can be re-written as

$$\begin{aligned} J = \big\langle f(\theta) \big\rangle_{p(\theta)} &= \int f(\theta) \frac{p(\theta)}{q(\theta)} q(\theta) d\theta \\ &= \int f(\theta) w(\theta) q(\theta) d\theta \\ &= \big\langle f(\theta) w(\theta) \big\rangle_{q(\theta)}, \quad \text{where } w(\theta) \triangleq \frac{p(\theta)}{q(\theta)} \end{aligned} \tag{2.14}$$

Here $w(\theta)$ is referred to as *importance weights*. *The proposal distribution* $q(\theta)$ can be any distribution from which drawing samples is relatively easy, compared to $p(\theta)$. Only requirement is that $q(\theta) > 0$ whenever $p(\theta) > 0$. Then the expectation in Equation 2.14 can be computed by

$$\hat{J} = \frac{1}{N} \sum_{i=1}^{N} f(\theta^{(i)}) w^{(i)}, \quad \text{where } w^{(i)} \triangleq w(\theta^{(i)}) \text{ and } \theta^{(i)} \sim q(\theta) \tag{2.15}$$

Moreover, in practice one can compute $p(\theta)$ only up to a multiplicative constant, i.e. $p(\theta) = \frac{1}{Z}\varphi(\theta)$, where Z is unknown. Then Equation 2.14 becomes

$$\hat{J} = \frac{1}{N}\sum_{i=1}^{N} f(\theta^{(i)})\ \frac{1}{Z}\tilde{w}^{(i)}, \quad \text{where } \tilde{w}^{(i)} = \frac{\varphi(\theta^{(i)})}{q(\theta^{(i)})} \tag{2.16}$$

But without knowing the value of Z, we can't evaluate $\hat{J}$ in Equation 2.16. Though, we can compute an estimate for Z, again using the Monte Carlo integration since Z is the normalizing constant of $p(\theta)$, i.e. $Z = \int_\Theta \varphi(\theta)\ d(\theta)$. Therefore we can use the same weighted samples obtained from $q(\theta)$ in Equation 2.16 and write

$$\bar{J} = \frac{\frac{1}{N}\sum_{i=1}^{N} f(\theta^{(i)})\tilde{w}^{(i)}}{\frac{1}{N}\sum_{i=1}^{N} \tilde{w}^{(i)}} = \frac{\sum_{i=1}^{N} f(\theta^{(i)})\tilde{w}^{(i)}}{\sum_{i=1}^{N} \tilde{w}^{(i)}} = \sum_{i=1}^{N} f(\theta^{(i)})\ W^{(i)} \tag{2.17}$$

where $W^{(i)} \triangleq \frac{\tilde{w}^{(i)}}{\sum_{i=1}^{N} \tilde{w}^{(i)}}$ are the *normalized importance weights*. Note that $\bar{J}$ is no longer an unbiased estimator for J since it is a ratio of two estimators. But it is still a consistent estimator for J.

2.2.1.1. Idea of Effective Sample Size. Although Equation 2.17 provides a consistent estimator for the quantity of interest, one is often interested in the variance of this estimator, which depends heavily on the *similarity* of the proposal distribution, $q(\theta)$, and the target distribution, $p(\theta)$. Kong (1992) introduced the idea of *effective sample size* (ESS) as

$$\text{ESS} = \frac{N}{1 + \text{Var}_q(W)}$$

where $\text{Var}_q(W)$ is the variance of the normalized importance weights. This quantity can be approximated by the *coefficient of variation* (Liu, 2004, Chapter 2.5.3) from the normalized weights and eventually provide an approximation to ESS, known as Kish's

Effective Sample Size (Kish, 1965) which is given by

$$\text{ESS} = \frac{1}{\sum_{i=1}^{N}(W^{(i)})^2} = \frac{1}{\sum_{i=1}^{N}\left(\frac{\tilde{w}^{(i)}}{\sum_{j=1}^{N}\tilde{w}^{(i)}}\right)^2} = \frac{\left(\sum_{i=1}^{N}\tilde{w}^{(i)}\right)^2}{\sum_{i=1}^{N}\left(\tilde{w}^{(i)}\right)^2} \tag{2.18}$$

where we have substituted unnormalized weights $\tilde{w}^{(i)}$.

2.2.2. Markov Chains

We will only mention key points of Markov chains in this section, but Norris (1998) provides an excellent introduction to the topic.

A *stochastic process* is simply a collection of random variables, e.g. $(X_0, X_1, \dots)$. A stochastic process $\{X_t\}_{t=0}^{T}$ is called a *Markov chain* if it satisfies the *Markov property* given by

$$\mathbb{P}(X_{t+1} = x_{t+1} \mid X_t = x_t, \dots, X_0 = x_0) = \mathbb{P}(X_{t+1} = x_{t+1} \mid X_t = x_t), \text{ for } t > 0$$

In other words, if the future of a stochastic process is completely determined by its current state, then it is said that the process satisfies the Markov property and it is referred to as *Markovian*. The quantities $\mathbb{P}(X_{t+1} = x_{t+1} \mid X_t = x_t)$ are called the *transition probabilities* or the *kernel* of the Markov chain and will be denoted by $\mathcal{K}(x_{t+1}; x_t)$. It is simply a distribution function for the next state parametrized by the current state. A transition kernel by itself isn't sufficient to define a Markov chain though. One also needs to specify an initial distribution, i.e. $p_0(x) \triangleq \mathbb{P}(X_0 = x)$. Then one can write $p_{t+1}(x') = \int \mathcal{K}(x'; x)\, p_t(x)\, dx$ for all $t > 0$ where $p_t(x)$ is the distribution function of the random variable X_t.

A Markov chain is said to be *irreducible* if for any given p_0, $p_t(x) > 0$ as $t \to \infty$ for all x. In other words, any state must be reachable from any initial state. A state x is called *aperiodic* if $p_t(x) > 0$ when $X_0 = x$ for all sufficiently large t, meaning that

starting from the state x, the chain must return to x for all $t > \tau$, for some finite τ. If an irreducible chain has an aperiodic state x, then all states are aperiodic (Norris, 1998). So we say that the chain itself is aperiodic. An irreducible and aperiodic Markov chain is called *ergodic*.

A distribution $\pi(x)$ called a *stationary (invariant) distribution* of the Markov chain with transition kernel $\mathcal{K}$ if $\pi(x') = \int \mathcal{K}(x'; x)\ \pi(x)\ dx$ for all x'.

An important property of ergodic Markov chains is that they have a unique stationary distribution. That is,

$$p_t(x) \to \pi(x) \text{ as } t \to \infty, \text{ for any } p_0(x) \tag{2.19}$$

This statement immediately suggests that, if one has an ergodic Markov chain with a stationary distribution $\pi(x)$, then one can obtain an independent sample from the distribution $\pi(x)$ by simply simulating a Markov chain for sufficiently large number of iterations. Then, one can compute expectations of any function of x with respect to $\pi(x)$ by using Equation 2.13.

But *Ergodic theorem* states that, for an ergodic chain and any bounded function of state, $f(x)$

$$\mathbb{P}(\frac{1}{N}\sum_{i=0}^{N-1} f(X_i) \to \hat{f} \text{ as } N \to \infty) = 1 \tag{2.20}$$

where $\hat{f} = \int f(x)\pi(x)dx$ and $\pi(x)$ is the unique stationary distribution of the chain. In other words, the average value of a function $f(x)$ over N transitions, converges to the expectation of f with respect to the stationary distribution of the chain, $\pi(x)$ as $N \to \infty$, with probability one. This allows one to compute an estimator for the expectation of f with respect to $\pi(x)$, by simply simulating the Markov chain for sufficiently large number of iterations and computing the average of f.

2.2.3. Markov Chain Monte Carlo

Previous section introduced the idea of the stationary distribution of a Markov chain satisfying certain properties. Such chains allow one to compute expectations with respect the unique stationary distribution. But in practice, the inverse of this process is also desired. In particular, one usually has a possibly unnormalized distribution of interest and wishes to draw samples from this distribution, for example a posterior distribution resulting from a statistical model. In such cases, one needs to construct a Markov chain that admits the desired distribution as its stationary distribution.

First example of constructing such a Markov Chain is demonstrated in the seminal paper by Metropolis *et al.* (1953) in a statistical mechanics context. The algorithm is usually referred to as *Metropolis Algorithm*. Then Hastings (1970) generalized the method to allow non-symmetric proposals. This generalized form is called *Metropolis-Hastings Algorithm*.

Let $p(\theta) = \frac{1}{Z}\phi(\theta)$ be the distribution of interest, which is usually referred to as *the target distribution*. Normalizing constant Z is generally unknown and can be very hard to compute. In such cases, one can form a Markov Chain which admits p as its stationary distribution by using Figure 2.4. This means that, if we run the chain *long enough*, we can obtain a sample from the target distribution, p. Of course this immediately raises the question about the precise meaning of being *long enough*. It is not easy to tell whether the chain has reached the stationarity (or *mixed*), but it is usually fairly easy to tell it has not. But once we believe that the chain has mixed, we can collect subsequent samples from then on, as they are also distributed according to p. The important point is that the samples won't be *independent* anymore. *Ergodic theorem* assures that, although the samples are not independent, they can still be used to form an unbiased estimator of any function of the state. Though the variance of the estimator is not trivial to compute, as in the case if the samples had been independent.

Input N: number of iterations

Sample an initial state, $\theta^{(1)}$, from the prior $p_0(\theta)$

for $i = 2$ to N **do**

  Propose a new state, θ', by using $q(\theta'; \theta^{(i-1)})$

  Compute acceptance probability $\alpha = \dfrac{\phi(\theta')\ q(\theta^{(i-1)}; \theta')}{\phi(\theta^{(i-1)})\ q(\theta'; \theta^{(i-1)})}$

  Sample u from $\mathcal{U}(0, 1)$

  if $u < \alpha$ **then**

    $\theta^{(i)} \leftarrow \theta'$ (accept)

  else

    $\theta^{(i)} \leftarrow \theta^{(i-1)}$ (reject)

  end if

end for

Figure 2.4. Metropolis-Hastings Algorithm.

2.2.3.1. Issues with Metropolis-Hastings Algorithm. Metropolis-Hastings algorithm is usually a great starting point in many statistical inference problems. But it is not without issues of course. First of all we need a suitable proposal distribution for the problem. This is not a trivial task for large dimensional problems. Also there are some optional parameters to the algorithm listed below:

- Burn-in period: As we have mentioned in the previous section, telling whether the chain has mixed is not a trivial task. For this purpose a so-called *burn-in period* is usually employed, by discarding some fixed number of initial samples.
- Thinning: Another issue is the dependence between subsequent samples. In order to reduce the dependence, one can discard fixed number of samples before keeping one. This method is called *thinning*. Although this procedure reduces the dependence between samples, it doesn't make the estimator any better, as the ergodic theorem holds no matter what degree of dependence the samples have. But if we are required to limit the number of samples for some reason, e.g. storage considerations, we would like them to be as independent as possible from each other. Then we need to employ an appropriate thinning parameter.

- Number of chains: Metropolis-Hastings is a sequential algorithm. Meaning that we need to have a current sample in order to obtain the next one. It is not parallelizable in this sense. But we can always run multiple chains with different initial samples. As today's computers have multiple cores and clusters of computers readily available for almost anyone, we would like to parallelize everything as much as possible. Although running parallel chains with independent starting states seems like a possible parallelization opportunity, it has its own issues. We will discuss these issues later, but it boils down to using *Pseudo-Random Number Generators* (PRNG) for our source of randomness. It is generally not advised to run more than a handful of chains in parallel because of the possibility of them becoming correlated with each other after some number of iterations.

These parameters may require some tuning up, which is only possible by performing preliminary runs of the chain and evaluating its performance for some values of the parameters. Also, acceptance rate, that is the ratio between the accepted states and the total number of proposal, needs to be monitored.

Metropolis-Hastings algorithm is very simple and easy to implement, but it may be very hard to find an appropriate proposal distribution for the problem at hand. Fortunately there are other ways to construct a Markov Chain with desired stationary distribution. A particularly useful one is called Hamiltonian Monte Carlo (HMC) and it is the topic of the next section. For further details on MCMC see Tierney (1994) and MacKay (2003, Chapters 29, 30).

2.2.4. Hamiltonian Monte Carlo

In essence, HMC involves constructing a physical system with a certain geometry obtained from the desired target distribution and simulating the motion of a particle according to Hamilton's equations of motion for certain amount of time. This statement already introduces two tunable parameters. As for any numerical simulation, we have to take discrete steps in time. And we have to decide how many steps to take before stopping. This is in fact a special case of the Metropolis-Hastings algorithm, where

proposed state is obtained by simulating a physical system. If we could perform the simulation with infinite precision, we would end-up at a new state with exactly the same total mechanical energy, since we have constructed the problem as a mechanical system. So we would end-up with accepting every proposed state. But due to the discretization of time, the proposed state may correspond to slightly lower or higher energy levels. Therefore we perform a Metropolis accept/reject step for each proposal. Neal (2011) provides an in-depth review of HMC.

2.2.5. Gibbs Sampling

Gibbs sampling method is introduced by S. Geman and D. Geman (1984) in a Bayesian image construction context and extended by Gelfand and Smith (1990) to continuous distributions. It is a special case of the Metropolis Method where the transition kernel is defined by composing the so-called *full conditional distributions* of the target distribution $p(\boldsymbol{\theta})$, where $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_M) \in \mathbb{R}^M$. A full conditional distribution of $p(\boldsymbol{\theta})$ corresponding to θ_i is defined to be $p(\theta_i \mid \boldsymbol{\theta}_{-i})$, where $\boldsymbol{\theta}_{-i} = \{\theta_j \mid j \neq i, \forall j \in [M]\}$. The Gibbs sampler updates each θ_i with a new value drawn from the corresponding full conditional $p(\theta_i \mid \boldsymbol{\theta}_{-i})$. The update sequence doesn't need to be deterministic as long as all coordinates are updated at each iteration. Suppose at iteration t, we have $\boldsymbol{\theta}^{(t)} = (\theta_1^{(t)}, \ldots, \theta_M^{(t)})$. Then we update the coordinates at iteration $t+1$ according to the algorithm in Figure 2.5.

for $i = 1$ to M **do**
  Draw $\theta_i^{(t+1)}$ from $p(\theta_i \mid \theta_1^{(t+1)}, \theta_{i-1}^{(t+1)}, \theta_{i+1}^{(t)}, \theta_M^{(t)})$.
end for

Figure 2.5. Gibbs Sampler Algorithm.

3. DIRICHLET-LUCE CHOICE MODEL

Recommendation systems emerged from the need to present their users a small subset of all possible alternatives. Usually, cardinality of the set of alternatives is very large, and it's getting larger everyday as new content is being created continuously. Movie streaming services are probably one of the most popular examples at the time of this writing. But more importantly, number of subsets of all the alternatives is much larger, even with fixed subset size. For example, a set with N items has $\frac{N(N-1)}{2}$ subsets of size 2. For large N, this quantity is dominated by the N^2 term, therefore it is in the order of N^2, i.e. $\mathcal{O}(N^2)$. For this reason, recommendation systems need to employ a selection strategy to come up with a plausible recommendation to its users.

In this chapter, we will introduce the terminology, define relevant concepts and develop a probabilistic model in terms of these concepts from the first principles with as little assumptions as possible.

3.1. Model Statement

Let $\mathcal{A}$ be the set of items and K be the cardinality of $\mathcal{A}$. Without loss of generality, we can uniquely label each item in $\mathcal{A}$ by a natural number from the set $\{1, 2, \ldots, K\}$, which will be denoted by $[K]$.

A *presentation* is any non-empty subset of $[K]$. Here we will be considering only the subsets of fixed size, which will be denoted by L, where in practice $L \ll K$. We will denote presentations by Y and the set of all possible presentations by $\mathcal{Y}$.

We assume that each user is associated a K-dimensional *preference vector* θ, which represents the user's preferences over items in $\mathcal{A}$, in such a way that the probability of choosing the item k when presented with all K items, is θ_k. Therefore, $\theta_k > 0$ for all $k \in [K]$ and $\sum_{k=1}^{K} \theta_k = 1$. In other words, $\theta \in \Delta^{K-1}$.

We further assume that the user chooses one and only one of the items from any given presentation Y. We call this a *choice* and denote by x, i.e. $x \in Y$.

An *interaction* is a tuple of a presentation and the corresponding choice. We will index interactions with t, and denote the total number of interactions by T. $Y_{1:T}$ and $x_{1:T}$ will denote the series of presentations and corresponding choices respectively, i.e.

$$Y_{1:T} = \{Y_1, Y_2, \ldots, Y_T\} \qquad x_{1:T} = \{x_1, x_2, \ldots, x_T\}$$

An example sequence of interactions with $K = 3$, $L = 2$ and $T = 4$ is shown in Table 3.1.

Table 3.1. A sequence of interactions with $K = 3$, $L = 2$ and $T = 4$.

t	Y_t	x_t
1	$\{1, 2\}$	1
2	$\{1, 3\}$	1
3	$\{1, 2\}$	2
4	$\{2, 3\}$	2

Note that, although it is not an analytical requirement, we will not consider the edge cases $L = 1$ and $L = K$. $L = 1$ corresponds to presenting only one item. Since we assume that the user makes a choice from any presentation, a presentation with single item forces the user to choose the presented item. Likewise, $L = K$ corresponds to presenting all items at once, which denies the need for having a recommendation system in the first place.

First step of building any probabilistic model is to describe the relationship between observations and the model parameters in terms of probabilities. In other words, we need to associate a likelihood to the observed data, conditioned on the parameters. In this particular case, the parameters of interest are the components of the preference

vector, θ, and observations are the choices $x_{1:T}$ made by the user to presentations $Y_{1:T}$.

Here we will assume Luce's choice axiom (Luce, 1959; Luce, 1977), which is stated as follows: When a choice is made from a subset of all alternatives, the probability of choosing an item over one another remains unchanged, for all items. In particular, if a person chooses item i over j with probability $\theta_i/(\theta_i + \theta_j)$, then this probability does not depend on the other items in the set.

This allows us to write the probability of choosing x from the presentation Y by a user with the preference vector θ as

$$p_Y(x \mid \theta) = \frac{\theta_x}{\sum_{j \in Y} \theta_j} \tag{3.1}$$

In other words, for any given presentation $Y = \{y_1, \ldots, y_L\}$,

$$x \mid \theta \sim \text{Cat}(\eta), \qquad \text{where } \eta = \{\eta_i\}_{i=1}^{L} = \frac{\theta_{y_i}}{\sum_{j \in Y} \theta_j}$$

Note that in Equation 3.1, x and Y are known, as they constitute the observed data. Therefore $p_Y(x \mid \theta)$ is a function θ only, parametrized by x and Y. We denote this function by $\lambda(\theta; x, Y)$, i.e.

$$\lambda(\theta; x, Y) \triangleq \frac{\theta_x}{\sum_{j \in Y} \theta_j} \tag{3.2}$$

Recommender's interest is of course in the posterior distribution $p_Y(\theta \mid x)$, so that it can be employed to generate more relevant recommendations for the next interaction with the same user. Here we need to make it clear that we assume each user has a particular θ, which doesn't change in time. Of course user preferences are not time invariant in general, but in a particularly short interval, we can assume that they do not change drastically. But this point should be kept in mind as a further research direction.

We also assume that the interactions are independent from each other. In other words, the choice made for the presentation at time t does not depend on the previous interactions. Therefore the probability of observing choices up to time T is

$$p_{Y_{1:T}}(x_{1:T} \mid \theta) = \prod_{t=1}^{T} \lambda(\theta; x_t, Y_t) \tag{3.3}$$

Denoting the prior distribution of the preference vector θ as $p(\theta)$, we can write the posterior distribution conditioned on the choices up to time T as

$$\begin{aligned} p_{Y_{1:T}}(\theta \mid x_{1:T}) &= \frac{1}{Z_T}\ p_{Y_{1:T}}(x_{1:T} \mid \theta)\ p(\theta) \\ &= \frac{1}{Z_T}\ \prod_{t=1}^{T} \lambda(\theta; x_t, Y_t)\ p(\theta) \end{aligned} \tag{3.4}$$

where Z_T is the normalization constant.

Since we have set the context, we will introduce the following notation to shorten the expressions for the rest of this work. $\pi_t(\theta)$ will denote the posterior distribution at time t, i.e. $\pi_t(\theta) \triangleq p_{Y_{1:t}}(\theta \mid x_{1:t})$. Similarly, we will use $\lambda_t(\theta) \triangleq \lambda(\theta; x_t, Y_t)$. We also denote the prior distribution by $\pi_0(\theta)$. Therefore we can write Equation 3.4 as

$$\pi_T(\theta) = \frac{1}{Z_T} \prod_{t=1}^{T} \lambda_t(\theta)\ \pi_0(\theta) \tag{3.5}$$

Although we have suppressed the observations $x_{1:T}$ and $Y_{1:T}$ in Equation 3.5, the notation implicitly implies the dependence. In this form, Equation 3.5 immediately reflects the relationship between consecutive posterior distributions, π_{t+1} and π_t as

$$\begin{aligned} \pi_{t+1}(\theta) &= \frac{1}{Z_{t+1}} \prod_{t=1}^{t+1} \lambda_{t+1}(\theta)\ \pi_0(\theta) \\ &= \frac{1}{Z_{t+1}} \lambda_{t+1}(\theta) \prod_{t=1}^{t} \lambda_t(\theta)\ \pi_0(\theta) \\ &= \frac{Z_t}{Z_{t+1}} \lambda_{t+1}(\theta)\ \pi_t(\theta) \end{aligned}$$

In other words

$$\pi_{t+1}(\theta) \propto \lambda_{t+1}(\theta)\ \pi_t(\theta), \quad \text{for} \quad t \in \{0, 1, \ldots, T-1\} \tag{3.6}$$

Finally we complete the model statement by assuming a Dirichlet prior, π_0, with concentration parameters α. Of course it can be any distribution with appropriate support, but as it will be made clear later, we need a prior distribution that we can draw samples from relatively easily. Dirichlet distribution serves this purpose very well (Devroye, 1986; Marsaglia and Tsang, 2000).

Substituting density function for $\mathrm{Dir}(\alpha)$ and explicit expression of $\lambda_t(\theta)$ into Equation 3.5 we obtain

$$\pi_T(\theta) = \frac{1}{Z_T} \prod_{t=1}^{T} \frac{\theta_{x_t}}{\sum_{j \in Y_t} \theta_j} \prod_{k=1}^{K} \theta_k^{\alpha_k - 1} \tag{3.7}$$

Consider only the numerator of the first product term in Equation 3.7.

$$\prod_{t=1}^{T} \theta_{x_t} = \prod_{t=1}^{T} \prod_{k=1}^{K} \theta_k^{[x_t = k]} = \prod_{k=1}^{K} \prod_{t=1}^{T} \theta_k^{[x_t = k]} = \prod_{k=1}^{K} \theta_k^{\sum_{t=1}^{T} [x_t = k]} = \prod_{k=1}^{K} \theta_k^{\gamma_k} \tag{3.8}$$

where $\gamma_k = \sum_{t=1}^{T} [x_t = k]$ is the total number of times the item k was chosen in T interactions and [.] is the Iverson bracket, which is equal to 1 if the condition inside the bracket evaluates to true, 0 otherwise. Similarly for the denominator we can write

$$\begin{aligned}
\prod_{t=1}^{T} \sum_{j \in Y_t} \theta_j &= \prod_{t=1}^{T} \prod_{Y \in \mathcal{Y}} \left(\sum_{j \in Y} \theta_j \right)^{[Y_t = Y]} \\
&= \prod_{Y \in \mathcal{Y}} \prod_{t=1}^{T} \left(\sum_{j \in Y} \theta_j \right)^{[Y_t = Y]} \\
&= \prod_{Y \in \mathcal{Y}} \left(\sum_{j \in Y} \theta_j \right)^{\sum_{t=1}^{T} [Y_t = Y]} \\
&= \prod_{Y \in \mathcal{Y}} \left(\sum_{j \in Y} \theta_j \right)^{\mu(Y)}
\end{aligned} \tag{3.9}$$

where $\mu(Y) = \sum_{t=1}^{T}[Y_t = Y]$ is the number of times a particular Y was presented. Substituting Equations 3.8 and 3.9 into Equation 3.7 we obtain

$$\pi_T(\theta) = \frac{1}{Z_T}\prod_{k=1}^{K}\theta_k^{\gamma_k+\alpha_k-1}\prod_{Y\in\mathcal{Y}}\left(\sum_{j\in Y}\theta_j\right)^{-\mu(Y)} \tag{3.10}$$

Equations 3.7 and 3.10 are equivalent and we will make use of both forms, depending on the situation.

Note that Equation 3.10 depends on the presentations $Y_{1:T}$ and corresponding choices $x_{1:T}$ only through the statistics γ_k and $\mu(Y)$. In other words, the posterior density $\pi_t(\theta)$ evaluates to the same value independent of the order of presentations and choices, as long as the total numbers of presentations and choices up to time t remain invariant. This property is called *exchangeability* and provides a computational advantage as we will discuss later.

3.2. Examples

In this section we will try to visualize the distribution in Equation 3.5 for various scenarios in order to gain some insights about its properties. Since the posterior distribution's support is Δ^{K-1}, only practical case for visualization purposes is the one with $K = 3$. $K = 4$ is also possible but 3-simplex is a 3D object, therefore it is more suitable for interactive visualizations, which can be performed on a computer.

For $K = 3$, the only non-trivial choice for L is 2. Therefore we will use $K = 3$ and $L = 2$ for all the examples in this section.

Also note that the condition $\theta_1 + \theta_2 + \theta_3 = 1$ defines one of the components implicitly in terms of the others. The choice of the implicit component is arbitrary, so we will pick θ_3. Therefore we will substitute $\theta_3 = 1 - \theta_1 - \theta_2$ whenever necessary in all examples.

3.2.1. Single interaction with non-informative prior

This examples illustrates the scenario where a recommender has 3 items from which it needs to form a presentation of size 2 for a particular user with unknown preference vector θ. Recommender has no information on θ. Therefore it assumes all $\theta \in \Delta^2$ is equally likely. That is, the prior $\pi_0(\theta)$ is given by

$$\pi_0(\theta) = \begin{cases} \text{constant}, & \text{for } \theta \in \Delta^2 \\ 0, & \text{everywhere else} \end{cases}$$

Recommender picks a particular θ via a strategy, possibly making use of its belief about the current distribution for the θ. Say, recommender forms a presentation as $\{1, 2\}$ and gets a response from the user with $x = 1$. We are interested in the posterior $\pi_1(\theta)$ for $Y = \{1, 2\}$, $x = 1$. We can use Equation 3.5 with $\lambda_1(\theta) = \dfrac{\theta_1}{\theta_1 + \theta_2}$ and write

$$\pi_1(\theta) = \frac{1}{Z_1} \frac{\theta_1}{\theta_1 + \theta_2}, \quad \text{where } \theta \in \Delta^2$$

Here, Z_1 denotes the normalizing constant which makes π_1 a proper probability distribution, that is

$$Z_1 = \int_{\theta \in \Delta^2} \frac{\theta_1}{\theta_1 + \theta_2} d\theta_1 d\theta_2$$

In this particular case we can calculate Z_1, as the integration involves a simple function of θ_1 and θ_2 and the integration region is only 2-dimensional. But in general we won't be able to, since in the general case it turns out to be a special hypergeometric function, called Carlson's $\mathcal{R}$ function (Carlson, 1971). Therefore we visualize only the unnormalized part in Figure 3.1. Lighter regions correspond to higher likelihood. Observe that, although we had no information regarding user's preferences before the interaction, we were able to update our belief according to the fact that user has chosen 1 from the presentation $\{1, 2\}$. Posterior distribution $\pi_1(\theta)$ reflects this belief by placing more probability mass over the region $\theta_1 > \theta_2$.

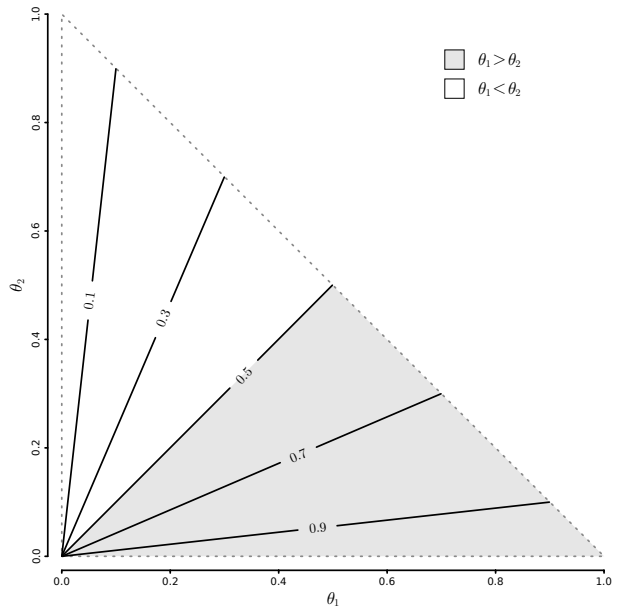

Figure 3.1. Unnormalized posterior for $K = 3$, $L = 2$, $Y = \{1, 2\}$, $x = 1$.

3.2.2. Single interaction with informative prior

Consider the same interaction as in Example 3.2.1, i.e. $Y = \{1, 2\}$, $x = 1$. But this time recommender has some assumption about the user preferences, for example via domain knowledge. In other words, instead of having a constant $\pi_0(\theta)$ over Δ^2, we assume that $\pi_0(\theta)$ is a Dirichlet distribution with concentration parameters, say $\alpha = (1, 3, 1)$. This particular choice of α corresponds to a prior belief that θ_2 is likely to be bigger that both θ_1 and θ_3.

Let's use Equation 3.10 this time to obtain the expression for posterior $\pi_1(\theta)$. Since we have single presentation, there is only one $Y \in \mathcal{Y}$ with non-zero $\mu(Y)$, i.e $\mu(\{1, 2\}) = 1$ and $\mu(Y) = 0$ for all $Y \in \mathcal{Y} \setminus \{\{1, 2\}\}$. Therefore, second product in Equation 3.10 is effectively $(\theta_1 + \theta_2)^{-1}$. Again, since we have single interaction with choice 1, $(\gamma_1, \gamma_2, \gamma_3) = (1, 0, 0)$ as γ_k corresponds to the total number of times the item k was chosen. We have already stated that $\alpha = (\alpha_1, \alpha_2, \alpha_3) = (1, 3, 1)$. Combining all three parameters we obtain the corresponding posterior distribution as

$$\pi_1(\theta) = \frac{1}{Z_1}\theta_1\theta_2^2(\theta_1 + \theta_2)^{-1} \propto \frac{\theta_1\theta_2^2}{\theta_1 + \theta_2}$$

Figure 3.2 shows the contour plot of this posterior distribution. Observe that, our prior belief for $\theta_2 > \theta_1$ is so strong that, even after observing the user's choice of 1 over 2, recommender is not yet *convinced*, as the posterior distribution for the user's preference vector still places higher probability mass over the region $\theta_2 > \theta_1$.

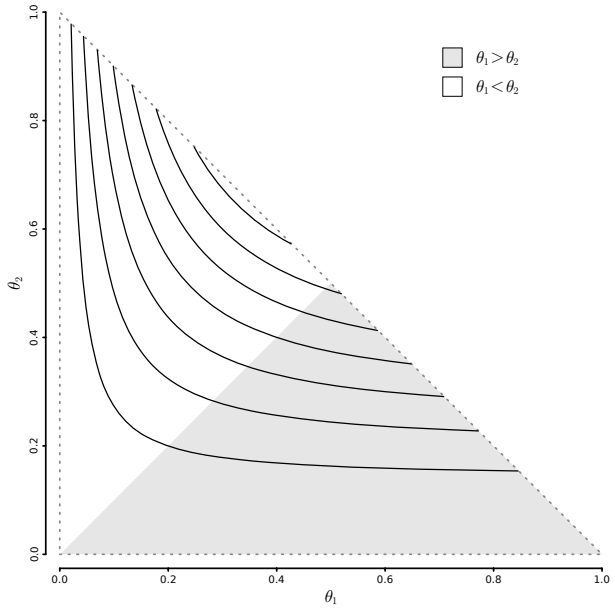

Figure 3.2. Unnormalized posterior for $K = 3$, $L = 2$, $Y = \{1, 2\}$, $x = 1$ with Dir(1,3,1) prior.

3.2.3. Transitive Interactions

This example illustrates a property of the posterior distribution which is called *transitivity*. In this scenario, recommender does not assume any prior information about the preferences of the user, i.e. all $\theta \in \Delta^2$ are equally likely. Then, it first asks the user to choose from $Y_1 = \{1, 2\}$ and gets a response $x_1 = 1$. Intuitively, this interaction increases the recommender's belief about $\theta_1 > \theta_2$. At $t = 2$, it asks the user to choose from $Y_2 = \{2, 3\}$ and gets a response $x_2 = 2$. Again, this interaction increases the recommender's belief about $\theta_2 > \theta_3$. The posterior distribution for these

two interactions with non-informative prior is given by

$$\pi_2(\theta) = \frac{1}{Z_2}\frac{\theta_1}{\theta_1+\theta_2}\frac{\theta_2}{\theta_2+\theta_3}, \quad \text{for } \theta \in \Delta^2$$

and unnormalized part is visualized in Figure 3.3. Observe that, although the recommender hasn't yet asked the user to choose from $\{1, 3\}$, the two interactions result a posterior which places higher probability mass over the region where $\theta_1 > \theta_3$. Note that the region $\theta_1 > \theta_3$ is equivalent to

$$\theta_1 > \theta_3 \implies \theta_1 > 1 - \theta_1 - \theta_2 \implies 2\theta_1 > 1 - \theta_2 \implies \theta_1 > \frac{1-\theta_2}{2}$$

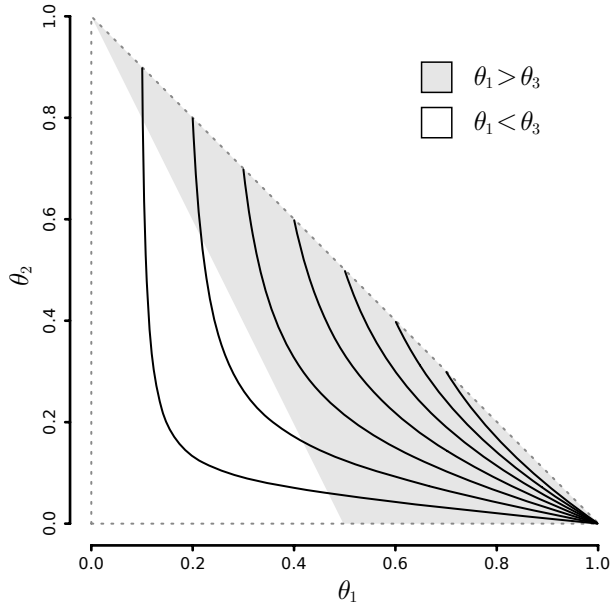

Figure 3.3. Unnormalized posterior that demonstrates transitivity property.

4. INFERENCE

We have introduced Dirichlet-Luce choice model in the previous chapter and gave some examples to gain some insight about its features. In this chapter we will propose a method to generate samples from the posterior, $\pi_T(\theta)$, conditioned on some interactions $Y_{1:T}$ and $x_{1:T}$.

The recursive form of the posterior given by Equation 3.6 clearly suggests a sequential importance sampling scheme, where we can use samples from the posterior at time t, π_t, to obtain samples from the posterior at time $t+1$, π_{t+1}, provided that we can generate samples from the prior distribution π_0. Although we are free to use any proposal distribution for each importance sampling step, we can exploit the multiplicative relationship between consecutive posterior distributions, again given by Equation 3.6 and use π_t as the proposal distribution to generate weighted samples from π_{t+1}.

Since Sequential Monte Carlo (SMC) methods are mostly used for *State Space Models* (SSM), the terminology is set by the problems associated with that literature. These sort of methods are usually referred to as *particle filters*, therefore weighted samples are called *particles*. We will adopt the terminology here to be consistent with the nomenclature and call samples as *particles*.

But before moving on to studying the proposed sampling method, we will establish a baseline by making use of the probabilistic programming language Stan, which allows declaring a statistical model in a convenient way and then produces samples from the posterior distribution over model parameters by using an HMC variant. Next section will introduce Stan and provide an example listing for the Dirichlet-Luce model. The samples generated by Stan will then be used to validate our proposed sampling method.

4.1. Stan and Hamiltonian Monte Carlo

As mentioned in Section 2.2.4 and detailed in Neal (2011), an HMC application requires careful parameter tunings and performance monitoring in order to make it work effectively for the particular problem. An adaptive HMC algorithm, called No-U-Turn Sampler (NUTS) is proposed by Hoffman and Gelman (2014) and implemented in the probabilistic programming language Stan (Carpenter *et al.*, 2017). Stan is a concise language to describe a statistical model, and performs sampling by using NUTS. Internally it maps Δ^{K-1} to $\mathbb{R}^{K-1}$ to get rid of the constraints imposed by the simplex and uses *automatic differentiation* (AD) to compute the required derivatives to simulate Hamiltonian dynamics (Carpenter *et al.*, 2015). *Stan Modeling Language Users Guide and Reference Manual* is the ultimate reference for the language. But we need to point out that it provides a convenient environment for statistical modeling, as it implements most of the standard distributions and has convenient data types. For example, `simplex[K] theta` simply declares `theta` as an element of Δ^{K-1}.

A complete model description with a Dirichlet prior, reflecting the sequential form of the posterior in Equation 3.7, is given in Appendix A.1. Although this form is more intuitive, it is not the most efficient in computational sense for large number of interactions. Because it needs to pass over all interactions to compute the log posterior for any given $\theta \in \Delta^{K-1}$. A more efficient implementation is provided in Appendix A.2, reflecting the form of the same posterior distribution in terms of sufficient statistics given by Equation 3.10. This form exploits the multiplicity of the choices and presentations. For large T, we would expect a popular item k to be chosen multiple times, which is denoted by γ_k. If we use the sequential form, we would need to compute $\log(\theta_k)$ multiple times, specifically γ_k times, and then add them up. Second implementation computes $\log(\theta_k)$ only once and multiplies it by γ_k. Of course algebraically these two operations are identical, but latter is much faster. Similar argument applies to the presentations as well. Although it depends on the presentation mechanism used by the recommender system, we would certainly expect that some presentations to be used multiple times, i.e. $\mu(Y) > 1$ for some $Y \in \mathcal{Y}$.

4.1.1. Example

Consider a series of interactions with $Y_{1:3} = \{\{1,2\},\{1,3\},\{1,3\}\}$ and $x_{1:3} = \{1,3,3\}$, with prior $\pi_0(\theta) = \mathrm{Dir}(\theta;(2,3,5))$. Posterior $\pi_3(\theta)$ can be written by using Equation 3.10 as

$$\pi_3(\theta) \propto \frac{\theta_1^2\ \theta_2^2\ \theta_3^6}{(\theta_1+\theta_2)(\theta_1+\theta_3)^2}$$

Substituting $\theta_3 = 1-\theta_1-\theta_2$ into above equation, we obtain

$$\pi_3(\theta) \propto \frac{\theta_1^2\ \theta_2^2\ (1-\theta_1-\theta_2)^6}{(\theta_1+\theta_2)(1-\theta_2)^2}$$

1000 samples generated from this posterior distribution using Stan, together with the contour plot of the unnormalized part is shown in Figure 4.1.

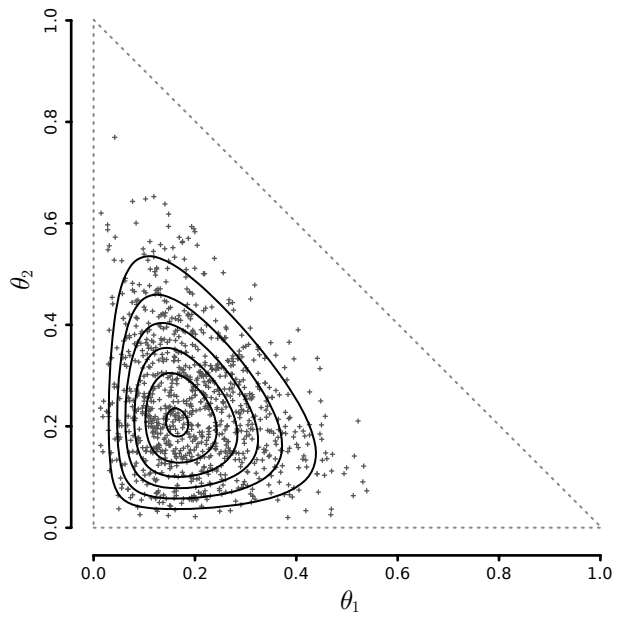

Figure 4.1. Contour plot of the posterior distribution of Example 4.1.1 and its sample approximation with 1000 samples generated by Stan.

4.2. Sequential Importance Sampling

We have already stated that we can draw independent samples from the prior distribution $\pi_0(\theta)$. For this reason we have further assumed that it is a Dirichlet distribution with some α. Then, in order to generate weighted samples from the posterior $\pi_1(\theta)$, we can use importance sampling with target distribution $\pi_1(\theta)$ and the proposal distribution $\pi_0(\theta)$. In this particular case, importance weights can be computed by

$$w_1(\theta^{(i)}) = \frac{\pi_1(\theta^{(i)})}{\pi_0(\theta^{(i)})}, \quad \theta^{(i)} \sim \pi_0 \text{ i.i.d} \quad \text{for } i = 1, 2, \ldots, N \tag{4.1}$$

where N is the number of particles. Superscript (i) stands for the ith particle, for example $\theta^{(i)}$ refers to the ith sample generated from π_0. Since we can evaluate $\pi_1(\theta)$ and $\pi_0(\theta)$ only up to a multiplicative constant we can compute weights only up to proportionality. More specifically, by Equation 3.6

$$w_1(\theta^{(i)}) \propto \lambda_1(\theta^{(i)}) = \frac{\theta_{x_1}^{(i)}}{\sum_{j \in Y_1} \theta_j^{(i)}} \tag{4.2}$$

In general, we will denote the importance weights at time t by w_t and we use $w_t^{(i)}$ for $w_t(\theta^{(i)})$. Similarly any function f, evaluated at a particular $\theta^{(i)}$, i.e. $f(\theta^{(i)})$, will be denoted by $f^{(i)}$ for short. Therefore we can re-write Equation 4.2 as

$$w_1^{(i)} \propto \lambda_1^{(i)}, \text{ for } i = 1, 2, \ldots, N$$

Similarly we could use samples from π_1 to obtain samples from π_2 by another application of importance sampling if we had i.i.d samples form π_1. But since this is not the case, we had to use π_0 again as the proposal distribution, which results importance weights

$$w_2^{(i)} = \frac{\pi_2^{(i)}}{\pi_0^{(i)}} \propto \lambda_2^{(i)} \lambda_1^{(i)} = \lambda_2^{(i)} w_1^{(i)}$$

This procedure can be applied sequentially, hence the name Sequential Importance Sampling (SIS), to obtain weighted samples at time t as

$$w_t^{(i)} \propto \lambda_t^{(i)} w_{t-1}^{(i)}, \text{ for } i = 1, 2, \ldots, N \tag{4.3}$$

The SIS algorithm is given in Figure 4.3 and an instance of the application for the scenario in Example 4.1.1 is shown in Figure 4.2 after each interaction, starting with the prior π_0. Particles are sized according to their weights. Note that at $t = 0$ all particles have unit weight, because they are i.i.d. samples from the prior distribution π_0. Observe that the posterior at $t = 3$ agrees with the samples obtained by Stan in Figure 4.1.

But it is a well known fact that, as t grows, w_t becomes practically zero for almost all particles except only a single particle which happens to *represent* the observations better than any other particle. This phenomenon is called *particle degeneracy*, and it drastically affects the variance of any expectation one would like to compute using the weighted samples. Even with only 3 interactions, we can observe this phenomenon in Figure 4.2. Some particles get larger as they happen to fit observations better, while others get smaller and smaller. Therefore, in order to be able to use such a sequential mechanism, we need to deal with this issue.

4.3. Resample and Move Algorithm

In particle filter applications, one method to deal with particle degeneracy issue is to generate a new set of particles with unit weights from the original set of weighted particles in such a way that the probability of a particle copied into the new set is proportional to its weight. In other words, particles with large weights get replicated in the new set possibly many times, while the ones with negligible weights get discarded. This method is called *resampling* and probably the most widely used one is referred to as *multinomial resampling*.

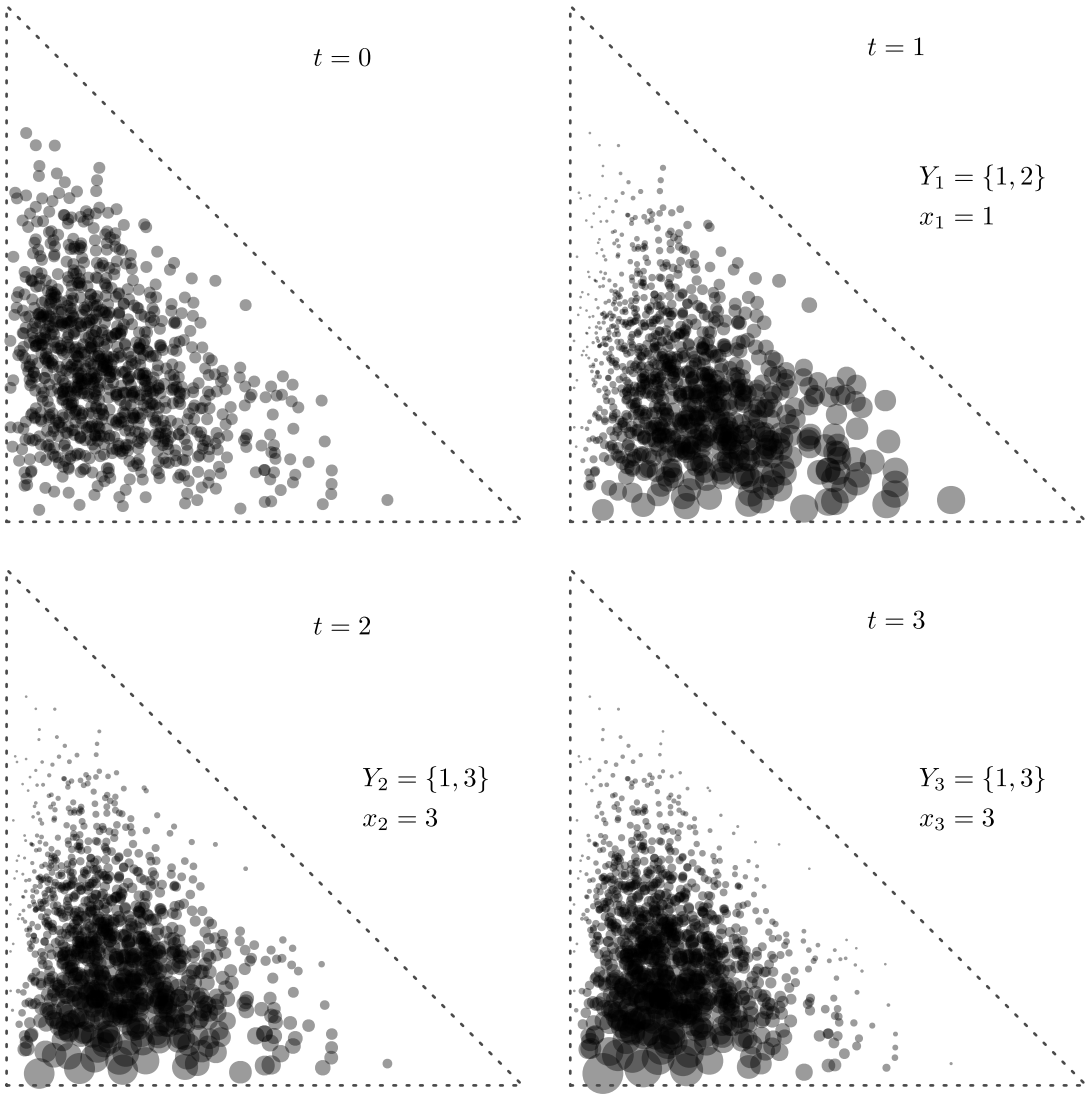

Figure 4.2. Sample approximations to the posterior distributions of Example 4.1.1 after each interaction, generated by the SIS algorithm using 1000 particles.

Input N: number of particles

Input T: number of interactions

Input $Y_{1:T}$: presentations

Input $x_{1:T}$: choices

for $i = 1$ to N **do**

  Sample $\theta^{(i)}$, from the prior $\pi_0(\theta)$

  Set $w^{(i)} = 1$

end for

for $t = 1$ to T **do**

  for $i = 1$ to N **do**

    Compute $\lambda_t^{(i)}$ from Equation 3.2

    Update $w^{(i)} \leftarrow \lambda_t^{(i)} w^{(i)}$

  end for

end for

Return weighted particles $(\theta^{(i)}, w^{(i)})$

Figure 4.3. Sequential Importance Sampling Algorithm.

But as we have mentioned before, particle filters usually applied to state space models, where particles *move* in state space according to a model-specific dynamic. Therefore, although some of the particles get replicated and end up being at the same point in the state space, the transition dynamics of such models allow particles to move to different positions before updating associated weights by using the observation.

In our case, we simply apply importance sampling sequentially, and update importance weights at each time t by using the observation at t. But there is no dynamic that moves our particles in the simplex. Chopin (2002) refers these models as being *static*. Therefore, even if we perform a resampling step to replace the current set of weighted particles with another set, we won't be able resolve the degeneracy issue, as particles with large weights get replicated many times at the same point in the simplex (Chopin, 2002). This is demonstrated in Figure 4.4 by performing a multinomial resampling step for the particles of Example 4.4.1.1 at $t = 3$, shown in Figure 4.2. Since these particles are plotted with transparency, darker points correspond to replicated particles at the same position. Although most of the particles with tiny weights got discarded by the resampling operation, replicated particles are still a major issue that we have to deal with.

The state space models analogy suggests a *move* step naturally. Gilks and Berzuini (2001) introduced the resample-move algorithm and demonstrated its use. Chopin (2002) uses the same idea for so-called static models, which happens to apply our problem extremely well because subsequent posterior distributions, i.e. π_t and π_{t-1} for $t \in \{1, 2, \ldots, T\}$ does not differ drastically, especially after some number of initial interactions. The algorithm is given in Figure 4.5.

But one major ingredient of the resample-move algorithm is the so called *move kernel*, which is essentially a Markov transition kernel $\mathcal{K}_t(\theta; \theta')$ that admits the posterior at t, π_t as its stationary distribution. But this kernel doesn't need to be ergodic (Gilks and Berzuini, 2001, Section 3.3). Therefore before moving on to applying the resample-move algorithm, we need to find an appropriate move kernel.

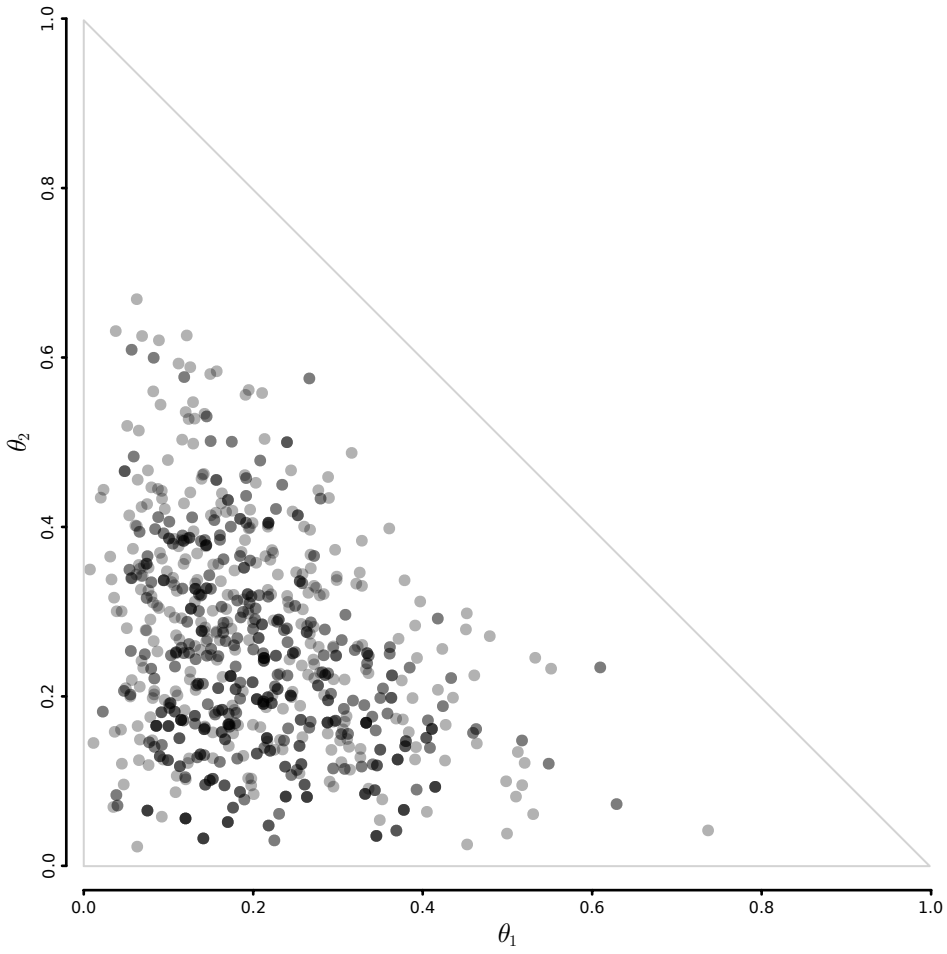

Figure 4.4. Particle degeneracy caused by performing a resampling step at $t = 3$ of the SIS application in Figure 4.2.

Input N: number of particles

Input T: number of interactions

Input $Y_{1:T}$: presentations

Input $x_{1:T}$: choices

Input ϵ: ESS threshold

for $i = 1$ to N **do**

Sample $\theta^{(i)}$, from the prior $\pi_0(\theta)$

Set $w^{(i)} = 1$

end for

for $t = 1$ to T **do**

for $i = 1$ to N **do**

Compute $\lambda_t^{(i)}$ from Equation 3.2

Update $w^{(i)} \leftarrow \lambda_t^{(i)} w^{(i)}$

end for

Compute ESS from Equation 2.18

if ESS $< \epsilon$ **then**

Perform resampling

for $i = 1$ to N **do**

$\theta^{(i)} \leftarrow \mathcal{K}_t(\theta; \theta^{(i)})$ (Move according to the kernel $\mathcal{K}_t$ which targets π_t)

$w^{(i)} \leftarrow 1$ (Reset particle weights to one)

end for

end if

end for

Return weighted particles $(\theta^{(i)}, w^{(i)})$

Figure 4.5. Resample-Move Algorithm.

4.4. Move Kernel $\mathcal{K}_t(\theta; \theta^{(i)})$

We have introduced the idea of the resample-move method to overcome the particle degeneracy issue in the previous section. We also noted that, in order to apply the method we need a move kernel which targets the posterior at time t, π_t. In this section we will discuss some options for the move kernel $\mathcal{K}_t(\theta; \theta^{(i)})$ together with their advantages and disadvantages.

Recall that, after the interaction at time t, we approximate the posterior π_t with a weighted set of particles, which we denote by $P_t = \{\theta^{(i)}, w^{(i)}\}$ for $i = 1, 2, \ldots, N$. Suppose that we perform a resampling step at t, which simply generates another set of particles of unit weights, i.e. $P'_t = \{\theta'^{(i)}, 1\}$. Here, some particles in P'_t are exact replicates of each other. How many times a particular $\theta^{(i)}$ in P_t gets replicated in P'_t depends on the weight of that particular particle, $w^{(i)}$. The purpose of the move kernel $\mathcal{K}_t$ is to distribute these replicated particles to the state space according to the posterior, π_t.

An important point to keep in mind is that, since the particles are already approximately distributed according to π_t, we don't have to worry about ergodicity or mixing. Even a single application of the kernel may suffice, depending of the statistical properties of the kernel.

4.4.1. Metropolis-Hastings Kernel

We have discussed Metropolis-Hastings algorithm in Section 2.2.3. Due to its simplicity, it is usually the first candidate for any MCMC application. But as already stated, it provides only a general framework and doesn't specify how to choose a good proposal $q(\theta'; \theta)$, which is a crucial ingredient of a successful MCMC application, especially in our particular case where we want to get rid of particle degeneracy by moving the replicated particles.

More specifically, even a bad proposal asymptotically converges to the target distribution in theory, albeit very inefficiently, if one waits *long enough*. Therefore with a sufficiently long burn-in and total number of iterations of the chain, we may be able to get away with a not-so-good proposal distribution q, if the aim is to generate samples from a specific target distribution. But here we already have samples from an approximation of the target distribution, where some samples are exact replicates, causing particle degeneracy. Obviously it is very unlikely to get rid of this degeneracy with a proposal distribution with very high rejection rate. But this doesn't mean that we should aim for a proposal distribution which has very low rejection rate. As stated by Chopin (2002), it is not an easy task to assign a metric to the degree of degeneracy. We can construct a proposal distribution with very low rejection rate by applying a sufficiently small perturbation to $\theta^{(i)}$, for example a multivariate normal with mean $\theta^{(i)}$ and very small variance. Of course this proposal would have a very low rejection rate, but the particles replicated at $\theta^{(i)}$ end up being very close to each other, which doesn't make much of a difference than the case where they sit on top of each other.

Unfortunately there is no universal recipe one can apply to construct a good proposal distribution for Metropolis-Hastings method. In our particular case, we would like to move particles by using as small iterations as possible keeping in mind that our main priority is to reduce particle degeneracy. Therefore applying small perturbations just to be able to move particles, is not an acceptable strategy.

We have introduced HMC and Gibbs sampling as special cases of the Metropolis-Hastings method in Sections 2.2.4 and 2.2.5, respectively. HMC exploits the geometry of the state space determined by the target distribution, π_t, in order to generate a proposed state, while Gibbs sampling makes us of conditional distributions of π_t for the same purpose. Theoretical acceptance probability of the new state generated by HMC is one. But due to discretization errors, it is not exactly one, but in general the simulation parameters can be tuned to obtain acceptance probabilities. And since it is incorporating the geometry determined by the current posterior, the original and proposed states can be very far apart from each other. A variant of HMC, called NUTS (Hoffman and Gelman, 2014) is especially designed for this purpose. Gibbs sampling

on the other hand produces a proposed state which gets accepted with probability one. Both of these methods seem like good candidates for our needs, with their high acceptance rates and ability to move particles *away* from the original state. But we need to consider their drawbacks as well.

HMC simulates Hamiltonian dynamics constructed from the target distribution. Specifically, we construct a potential energy function $V(\theta) = -\log(\pi_t(\theta))$. Hamiltonian dynamics involves partial derivatives of $V(\theta)$ with respect to each component of θ. In cases where these derivatives are available in closed form, implementing an HMC algorithm can be considered even trivial. But our posterior π_t has a recursive form, which doesn't yield nice, closed form partial derivatives unfortunately. In short, computing partial derivatives of $V(\theta)$ can be very expensive in high dimensional problems with large number of interactions.

Gibbs sampling on the other hand, doesn't require any derivatives, but it relies on conditional distributions of $\pi_t(\theta)$. Unfortunately, conditional distributions of π_t are not in standard form in general, introducing another challenge. Specifically, we can compute $\pi_t(\theta_i \mid \theta_{-i})$, for $i = 1, 2, \ldots, K-1$ only up to a multiplicative constant, i.e. $\pi_t(\theta_i \mid \theta_{-i}) \propto \phi(\theta_i)$. This means, if we wish to use a Gibbs kernel for the move step, we need to employ a method to draw samples from unnormalized, univariate distributions.

4.4.1.1. Example. We will consider the same setting as in Example 4.1.1. That is, $Y_{1:3} = \{\{1,2\},\{1,3\},\{1,3\}\}$ and $x_{1:3} = \{1,3,3\}$, with prior $\pi_0(\theta) = \mathrm{Dir}(\theta;(2,3,5))$. The posterior distribution $\pi_3(\theta)$ was

$$\pi_3(\theta) \propto \frac{\theta_1^2\ \theta_2^2\ (1-\theta_1-\theta_2)^6}{(\theta_1+\theta_2)(1-\theta_2)^2}$$

Conditional distributions of $\pi_3(\theta)$ are therefore given by

$$\pi_3(\theta_1 \mid \theta_2) \propto \frac{\theta_1^2\ (1-\theta_2-\theta_1)^6}{\theta_1+\theta_2}, \qquad \text{for } 0<\theta_1<1-\theta_2 \tag{4.4}$$

$$\pi_3(\theta_2 \mid \theta_1) \propto \frac{\theta_2^2\ (1-\theta_1-\theta_2)^6}{(\theta_1+\theta_2)(1-\theta_2)^2}, \qquad \text{for } 0<\theta_2<1-\theta_1 \tag{4.5}$$

The contour plot of π_3 is shown in Figure 4.6, unnormalized conditional distributions $\pi_3(\theta_1 \mid \theta_2)$ and $\pi_3(\theta_2 \mid \theta_1)$ for the point $(\theta_1, \theta_2) = (0.2, 0.6)$ are shown in Figures 4.7 and 4.8, respectively.

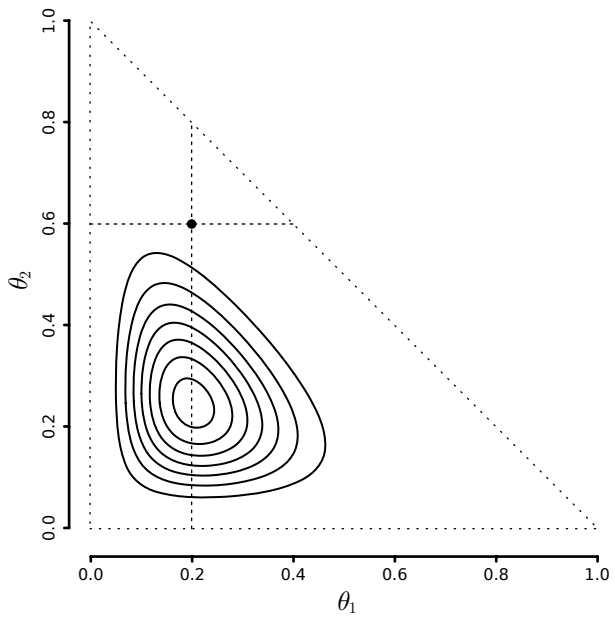

Figure 4.6. Unnormalized posterior, π_3 of Example 4.4.1.1.

4.4.2. Gibbs Based Move Kernel

In general we wish to apply the move step after performing resampling, where we end up with a set of particles, $\theta^{(i)}$ for $i = 1, 2, \ldots, N$, possibly some of them are exact replicates of one another. We propose a Gibbs based move kernel where each component is updated in turn. Since the conditional distributions are not in standard form, we can't generate samples directly. Therefore we need to incorporate a method to sample from unnormalized univariate distributions. Although there are methods which may be more efficient statistically, like slice sampling (Neal, 2003), they usually

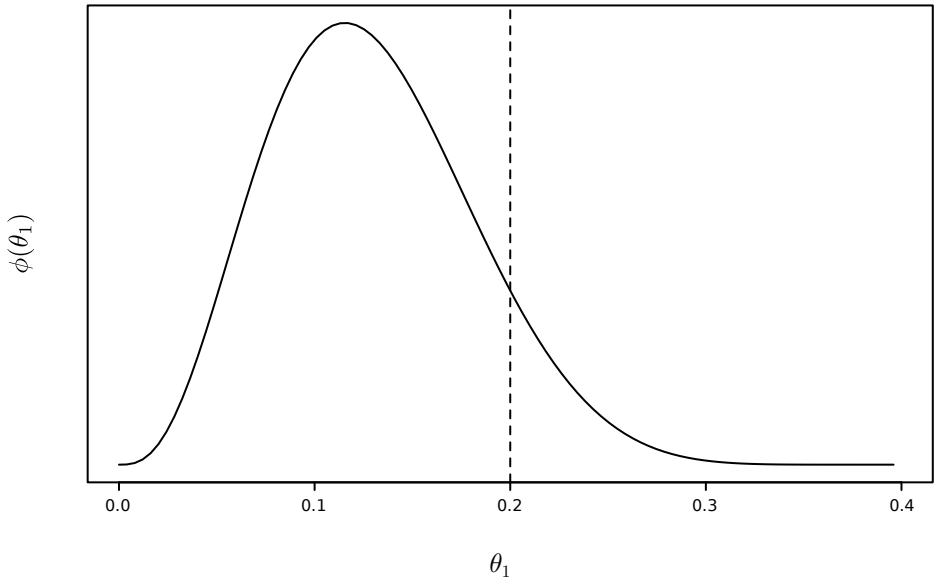

Figure 4.7. Unnormalized conditional distribution of $\pi_3(\theta_1 \mid \theta_2)$ for $\theta_2 = 0.6$.

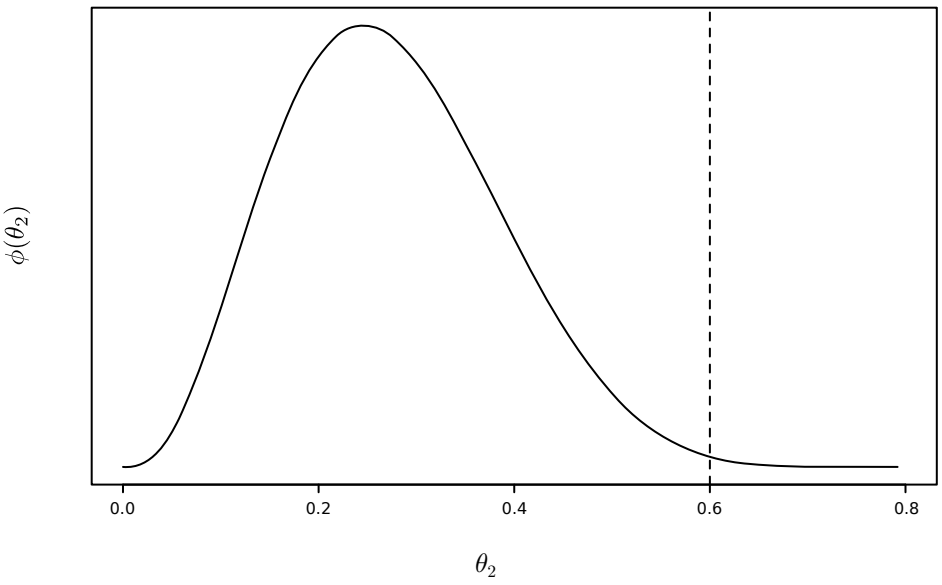

Figure 4.8. Unnormalized conditional distribution $\pi_3(\theta_2 \mid \theta_1)$ for $\theta_1 = 0.2$.

require a considerable number of evaluations of the conditional distributions at various points, which makes them computationally demanding. We use a simple independence proposal, which requires only two evaluations of the conditional distribution for each component. Algorithm in Figure 4.9 describes the proposed move mechanism. This is an application of the so called *Metropolized independence sampler*, first suggested by Hastings (1970). Gelman and Rubin (1992) and Tierney (1994) also suggests this approach when it is difficult to sample from corresponding full conditionals directly. Figure 4.10 shows the application of the proposed move method to Example 4.1.1 using the conditional distributions given in Equations 4.4 and 4.5. Comparing it with Figure 4.4 we can at least visually confirm that the particles that were on top of each other are moved around the state space according to the posterior π_3.

Input i: index of the particle to be moved
Input T: number of interactions
Input K: number of dimensions
Input $Y_{1:T}$: presentations
Input $x_{1:T}$: choices
for $k = 1$ to $K - 1$ **do**
  $\phi \leftarrow \pi_T(\theta_k^{(i)} \mid \{\theta_l^{(i)}\}_{l \neq k})$
  $\theta_k'^{(i)} \sim \mathcal{U}(0, 1 - \sum_{j \neq k} \theta_j^{(i)})$ (Propose a new value for the kth coordinate)
  $\phi' \leftarrow \pi_T(\theta_k'^{(i)} \mid \{\theta_l^{(i)}\}_{l \neq k})$
  Sample u from $\mathcal{U}(0, 1)$
  if $u < \dfrac{\phi'}{\phi}$ **then**
    $\theta_k^{(i)} \leftarrow \theta_k'$ (accept)
  end if
end for

Figure 4.9. Gibbs sampling based move algorithm.

4.4.2.1. Evaluation of the Full Conditional Distributions. Recall that $\pi_T(\theta)$ is defined on Δ^{K-1}. That is, although θ is a K-dimensional vector, it is constrained by the simplex. Therefore it has $K - 1$ degrees of freedom. In other words, one of the

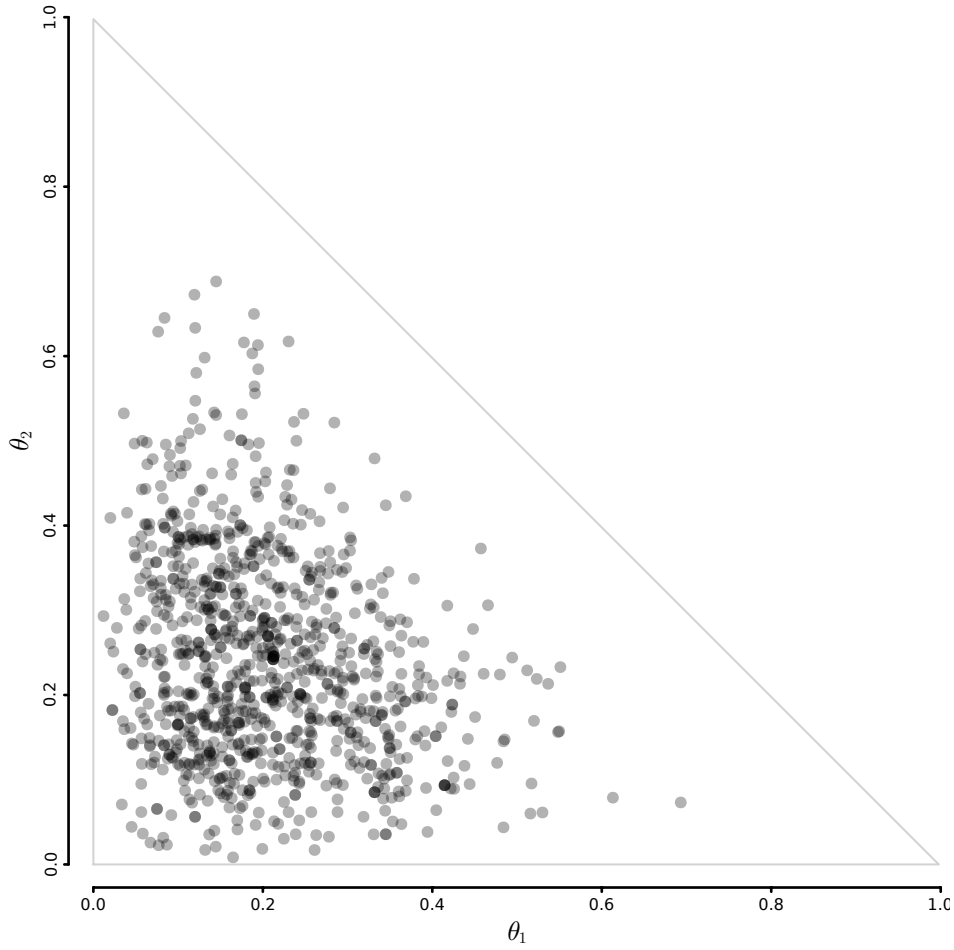

Figure 4.10. Reduction in sample degeneracy after the performing the move step.

coordinates is implicitly defined by the others due to the constraint $\sum_{k=1}^{K} \theta_k = 1$. A full conditional in the form $\pi_T(\theta_i \mid \{\theta_j\}_{j \neq i})$ may seem like a degenerate distribution at first if one is not careful. After all $\theta_i = 1 - \sum_{j \neq i} \theta_j$, for $i = 1, 2, \ldots, K$. If we condition on all the components of θ, except the ith one, then θ_i is a deterministic quantity. But this is not what we mean by the full conditional. Without loss of generality we will use θ_K as the implicit coordinate. Therefore the full conditional distribution of θ_i is conditioned on all θ_j where $j \notin \{i, K\}$, for $i = 1, 2, \ldots, K-1$.

By definition, conditional distribution of θ_i is given by

$$\pi_T(\theta_i \mid \{\theta_j\}_{j \notin \{i,K\}}) = \frac{\pi_T(\theta)}{\pi_T(\{\theta_j\}_{j=1, j \neq i}^{K-1})}$$

The denominator does not depend on θ_i. Therefore we can write

$$p_i(\theta_i) \propto \pi_T(\theta)$$

where $p_i(\theta_i)$ denotes the full conditional distribution $\pi_T(\theta_i \mid \{\theta_j\}_{j \notin \{i,K\}})$. Substituting $\pi_T(\theta)$ from Equation 3.10 we obtain

$$p_i(\theta_i) \propto \prod_{k=1}^{K} \theta_k^{\gamma_k + \alpha_k - 1} \prod_{Y \in \mathcal{Y}} \left(\sum_{j \in Y} \theta_j \right)^{-\mu(Y)} \quad \text{for } i = 1, 2, \ldots, K-1 \tag{4.6}$$

Recall that, given T interactions, γ_k is the number of times the item k was chosen, $\mu(Y)$ is the number of times a particular $Y \in \mathcal{Y}$ was presented and $\alpha = \{\alpha_1, \ldots, \alpha_K\}$ is the concentration parameter of the Dirichlet prior.

Note that, by definition $p_i(\theta_i)$ is a function of θ_i only, implicitly parametrized by the coordinates of θ that the distribution is conditioned on. Therefore some of the terms in Equation 4.6 are simply constants with respect to θ_i. Since we can compute $p_i(\theta_i)$ up to a multiplicative constant, we can further discard these constant terms to save valuable computation resources. Thus, we need to identify which terms depend

on θ_i.

First product in Equation 4.6 involves only the choices. Likewise, the second product involves only the presentations. Expanding the first product as

$$\prod_{k=1}^{K} \theta_k^{\gamma_k+\alpha_k-1} = \theta_1^{\gamma_1+\alpha_1-1}\ \theta_2^{\gamma_2+\alpha_2-1} \dots\ \theta_K^{\gamma_K+\alpha_K-1}$$

and substituting $\theta_K = 1 - \sum_{k=1}^{K-1} \theta_k = 1 - \theta_1 - \theta_2 - \cdots - \theta_{K-1}$ yields

$$\prod_{k=1}^{K} \theta_k^{\gamma_k+\alpha_k-1} = \theta_1^{\gamma_1+\alpha_1-1}\ \theta_2^{\gamma_2+\alpha_2-1} \dots (1 - \theta_1 - \theta_2 - \cdots - \theta_{K-1})^{\gamma_K+\alpha_K-1}$$

It is clear from this expression that, only terms with non-constant contribution to $p_i(\theta_i)$ are the ones that involve θ_i, i.e $\theta_i^{\gamma_i+\alpha_i-1}$ and $\theta_K^{\gamma_K+\alpha_K-1}$.

Second product in Equation 4.6 is a bit more involved as it is over all possible presentations $Y \in \mathcal{Y}$. But note that we have only a finite number of interactions, specifically T, which is usually much smaller than the cardinality of $\mathcal{Y}$. Even if $Y_{1:T}$ are all distinct, we would have at most T terms with non-zero $\mu(Y)$. Therefore the product involves at most T terms. Now we need to determine which presentations depend on θ_i.

The term corresponding to a particular presentation Y with $\mu(Y) > 0$ in Equation 4.6 is $\left(\sum_{j \in Y} \theta_j\right)^{-\mu(Y)}$. Recall that we use θ_K as the implicit coordinate, that is, we substitute $1 - \sum_{k=1}^{K-1} \theta_k$ for θ_K. Therefore, the only presentations that can provide a non-constant contribution to $p_i(\theta_i)$ are the ones that include items i or K. So we can analyze each presentation in the following four cases for all $i = 1, 2, \dots, K-1$:

(i) $i \in Y$, $K \in Y$:

$$\begin{aligned}\sum_{j\in Y}\theta_j &= \theta_i + \theta_K + \sum_{j\in Y\setminus\{i,K\}}\theta_j \\ &= \theta_i + 1 - \sum_{k=1}^{K-1}\theta_k + \text{Const} \\ &= \theta_i + 1 - (\theta_i + \sum_{k=1,k\neq i}^{K-1}\theta_k) + \text{Const} \\ &= \text{Constant}\end{aligned}$$

(ii) $i \in Y$, $K \notin Y$

$$\begin{aligned}\sum_{j\in Y}\theta_j &= \theta_i + \sum_{j\in Y\setminus\{i\}}\theta_j \\ &= \theta_i + \text{Const}\end{aligned}$$

(iii) $i \notin Y$, $K \in Y$

$$\begin{aligned}\sum_{j\in Y}\theta_j &= \theta_K + \sum_{j\in Y\setminus\{K\}}\theta_j \\ &= 1 - \sum_{k=1}^{K-1}\theta_k + \text{Const} \\ &= 1 - (\theta_i + \sum_{k=1,k\neq i}^{K-1}\theta_k) + \text{Const} \\ &= -\theta_i + \text{Constant}\end{aligned}$$

(iv) $i \notin Y$, $K \notin Y$: This case is trivially constant, as the presentation Y doesn't include neither item i nor K, i.e. all θ_j for $j \in Y$ are given.

We can summarize that the only cases where a particular presentation Y provides a non-constant contribution to $p_i(\theta_i)$ are the ones that include either the item i or K, but not both. Define the set

$$A_i \triangleq \{Y \in \mathcal{Y} \mid \mu(Y) > 0, (i \in Y) \oplus (K \in Y)\}$$

where $\oplus$ denotes logical XOR, meaning either $i \in Y$ or $K \in Y$, but not both. Then we can write Equation 4.6 as

$$p_i(\theta_i) \propto \theta_i^{\gamma_i+\alpha_i-1}\ \theta_K^{\gamma_K+\alpha_K-1} \prod_{Y \in A_i} \left(\sum_{j \in Y} \theta_j \right)^{-\mu(Y)} \quad \text{for } i = 1, 2, \ldots, K-1 \tag{4.7}$$

4.5. Summary

In this chapter we have proposed a sequential method based on resample-move algorithm to draw samples from the posterior π_T, conditioned on interactions up to time T. We have also demonstrated that we can target this posterior distribution by using the probabilistic programming language Stan. At this point we can question the need for the proposed sequential method. But we should always keep in mind that, MCMC methods which includes NUTS used by Stan, always target the posterior at π_T. If we add another observation, say at $T+1$, Stan or any other MCMC method doesn't care if we already had samples from π_T. They need to start afresh targeting π_{T+1}. This may be a deal-breaker for some applications, as we could have used samples from π_T by updating their weights according to the interaction at $T+1$ to obtain a sample approximation of the posterior π_{T+1}, almost trivially.

Another scenario where this proposed sequential mechanism have a huge advantage over MCMC samplers, is the fact that it allows experimenting with various presentation mechanisms. Until this point, we haven't discussed the proposal mechanism of the recommender, the one that forms the recommendation for the next interaction. We have always assumed that we are provided with some interactions $Y_{1:T}$ and $x_{1:T}$. Together with a prior distribution, we were able to draw samples from the posterior π_T. But the whole purpose of this effort was to gain insight about the user's preferences to guide the recommender for coming up with more relevant recommendations. Therefore the posterior at t has a direct effect on the presentation at time $t+1$. Experimenting with different presentation mechanisms would only be practically possible within a sequential setting, where we have the opportunity to form the presentation, Y_{t+1},

using the sample approximation of the posterior π_t. Next section will provide some experiments performed by using a specific presentations mechanism based on Thompson Sampling (Thompson, 1933; Russo *et al.*, 2018).

5. EXPERIMENTS

Finding the optimal presentation mechanism for a recommendation system may depend on a lot of factors. First, one needs to define a metric to measure the *goodness* of any given presentation mechanism, before considering searching for an optimum one. The purpose of this thesis is not to propose a method to search for an optimum presentation mechanism, but to develop a sequential sampling scheme that targets $\pi_T(\theta)$. Therefore, we will make use of a presentation mechanism based on Thompson sampling, proposed by Çapan *et al.* (2019), which was demonstrated to have desirable convergence properties. It can be summarized as follows:

Given a sample approximation to π_t of N weighted particles with corresponding weights $w^{(i)}$, $i = 1, 2, \ldots, N$

(i) Choose one of the particles according to weights, i.e. the probability of choosing the jth particle is $w^{(j)} / \sum_{i=1}^{N} w^{(i)}$, for $j = 1, 2, \ldots, N$.
(ii) Form Y_{t+1} with the items corresponding to top-L components of the particle chosen at (i)

We will make use of this presentations mechanism for all the experiments in this chapter.

5.1. Learning a Sparse Preference Vector

This experiment aims to demonstrate that the proposed sampling scheme produces a comparable posterior distribution to that of Stan's, but much more efficiently.

Suppose there are $K = 20$ items and the recommender wishes to present $L = 2$ items to a particular user. Initially the recommender makes no assumption on the preference vector of that particular user, therefore assumes a flat prior. We generate the latent preference vector of the user, denoted by θ^*, from a Dirichlet distribution

with concentration parameters α. All but five randomly chosen components of α are set to unity. Randomly chosen components are set to 10. The particular θ^* is shown in Figure 5.1.

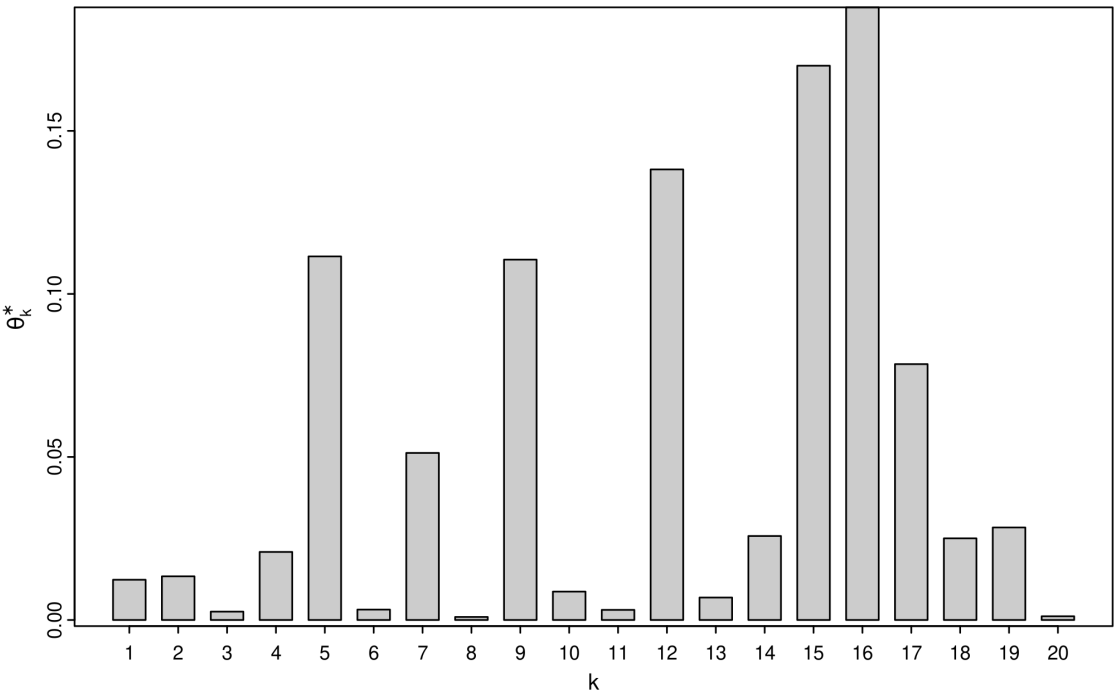

Figure 5.1. Latent user preferences, θ^*, of the experiment in Section 5.1.

Then we simulate $T = 100$ interactions, using $N = 10000$ particles according to the presentation mechanism described in the beginning of this chapter. We performed resampling when effective sample size (ESS) drops below $N/2$. Figure 5.2 shows the evolution of ESS in time, demonstrating the decreasing frequency of resampling operations as t grows. Figure 5.3 shows the marginal distributions of the posterior approximations of $\pi_T(\theta)$ generated by both samplers, along with the components of the latent preference vector, θ^*. Note that the marginal distributions of both samplers agree considerably well. Although they are far from the actual values of the components of θ^*, they capture the relative relationship between the components. In other words, both samplers are able to place high probability mass where the user's actual preferences are. But they tend to underestimate the items where the user prefers the most, while overestimating the rest. The reason for this behavior is actually the assumed flat prior,

and the presentation mechanism. The recommender assumes all items are equally popular apriori. This assumption results an underestimation for the popular items, which at the same time means an overestimation for the rest. After each observation, recommender's belief gets updated in favor of the popular items. But overcoming the apriori assumption takes some time to get reflected in the posterior. Nevertheless, even with this non-informative prior, the recommender was able to identify the user's favorite items, which was the main purpose. The exact measure of how much a user likes a particular item, represented by the latent preference vector's respective component, is not relevant for most of the cases.

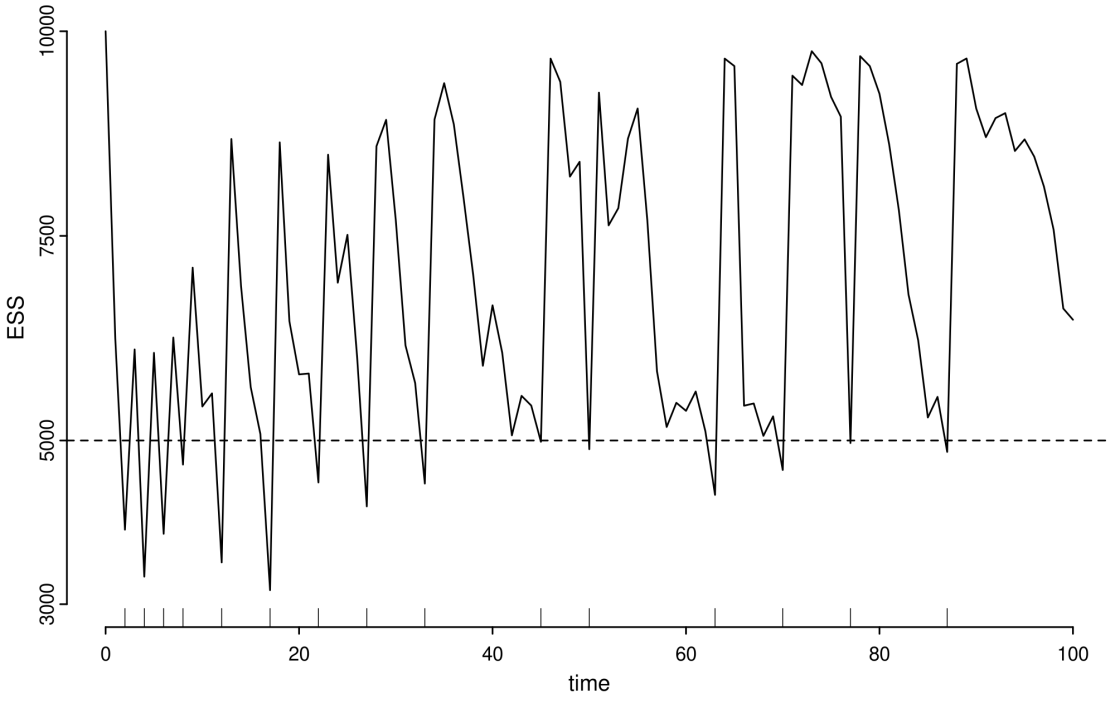

Figure 5.2. Evolution of ESS with time. We perform resampling whenever ESS drops below $N/2 = 5000$. Vertical ticks above the time axis indicates the points where resampling is performed, just before the next interaction.

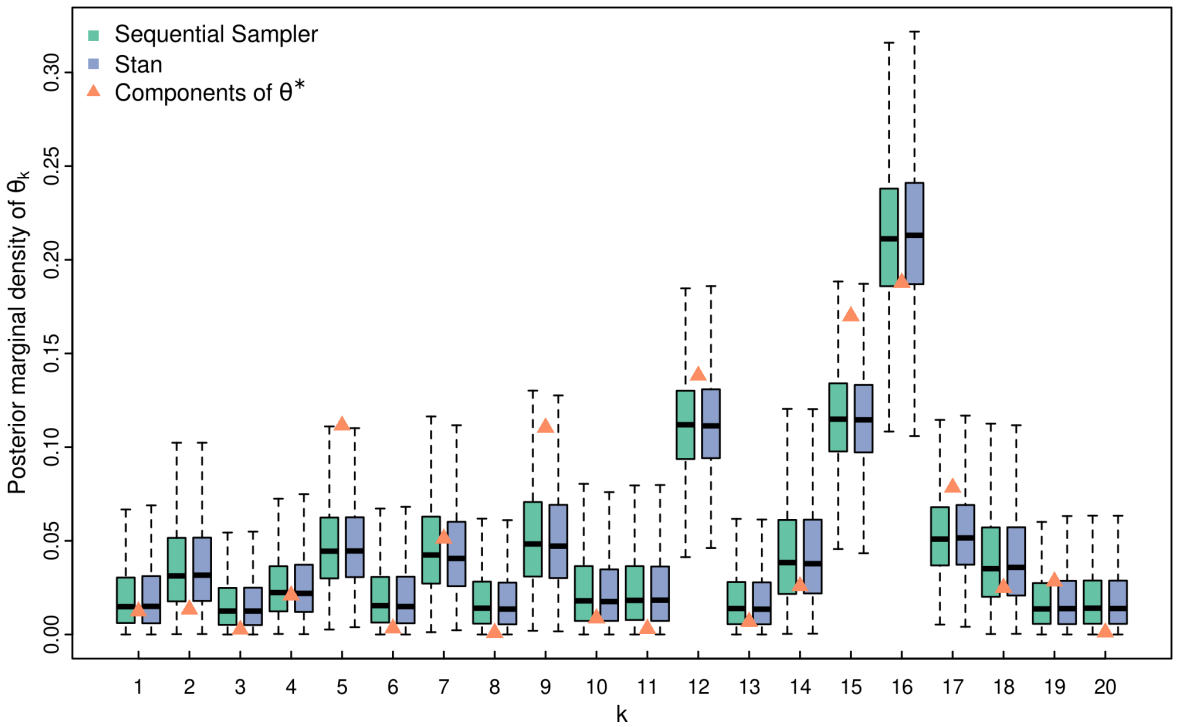

Figure 5.3. Posterior marginals of both samplers, after 100 interactions with flat prior, along with latent user preferences, θ^*.

6. CONCLUSION

We have presented and demonstrated by simulations that the proposed sequential sampling scheme is able to generate posterior samples in an efficient way. Due to its sequential nature, one major advantage is to be able to experiment with different presentation mechanisms. We have made use of a simple and intuitive presentation mechanism based on the idea of Thompson sampling and demonstrated that, it is capable of identifying user's actual preferences without requiring large number of interactions.

In this form, the recommendation system is assumed to be interacting one particular user, with an unknown but fixed preference vector θ^*. Of course this assumption should be relaxed to design a more realistic recommendation system, as they serve millions of users. Therefore one should assume a generative model for θ^* as well. Each user might have a specific preference vector, but there are certainly correlations between users. In such a scenario, the proposed model can be used as a building block in a collaborative setup.

Since the proposed move kernel uses a uniform independence proposal for the component-wise updates, it's rejection rate for some components get very high for large T, because the full conditional distribution of a popular item gets more and more peaked. Therefore the uniformly proposed value for that coordinate gets rejected with high probability. But since the move kernel traverses all coordinates, even when some components fail to attain a new value due to rejection, in our simulations we observed at least one component gets updated with very high probability. This phenomenon is most likely due to the presentation mechanism, which guides the presentations in such a way that the posterior distributions evolves *smoothly* in time. But this observation should be analyzed further from the analytical point of view. Of course one should seek a better proposal for component-wise updates, as uniform independence proposal is not ideal in such cases mainly because of the fact that it doesn't account for the previous interactions.

An alternative move kernel would be HMC, especially NUTS, since it takes care of parameter tuning issues of plain HMC. It is computationally more demanding than the proposed Gibbs-based move kernel, but as mentioned before, due to the fact that the consequent posterior distributions do not differ drastically, the frequency of resampling steps decreases with growing number of interactions. Since the move step always follows a resampling step, the relative computational cost of the resampling decreases in time.

Also note that we kept the number of particles, N, constant in all simulations. But it is not a requirement (Chopin, 2002). We can adjust N as the posterior distributions get concentrated in a region over time. Starting with large number of particles at the initial stages is advantageous, because at these early stages the consecutive posterior distributions possibly vary the most. After some number of iterations, we could decrease the number of particles. But when and by how much that N can be decreased should be analyzed formally.

Another consideration would be the number of items, K. In practice, this number can be in the order of millions for most applications, making the inference procedure very computationally demanding. But this computational burden can be reduced substantially, exploiting the fact that the user preferences are actually very sparse. Combined with the prior information provided by the domain knowledge and some number of initial interactions, the recommender would get fairly confident about some of the irrelevant items for a particular user, or a group of similarly behaving users. Since the recommender aims to identify the most relevant items for a particular user, it can simply discard such irrelevant items from future recommendations, effectively reducing the dimensionality of the problem. Current posterior samples would just need to be renormalized after eliminating a possibly irrelevant item.

REFERENCES

Çapan, G., İ. Gündoğdu, A. C. Türkmen, Ç. Sofuoğlu, A. T. Cemgil (2019), "A Bayesian Choice Model for Eliminating Feedback Loops", arXiv: 1908.05640.

Carlson, B. C. (1971), "Appell Functions and Multiple Averages", *SIAM Journal on Mathematical Analysis* Vol. 2, No. 3, pp. 420–430.

Carpenter, B., A. Gelman, M. Hoffman, D. Lee, B. Goodrich, M. Betancourt, M. Brubaker, J. Guo, P. Li, A. Riddell (2017), "Stan: A Probabilistic Programming Language", *Journal of Statistical Software, Articles* Vol. 76, No. 1, pp. 1–32.

Carpenter, B., M. D. Hoffman, M. Brubaker, D. Lee, P. Li, M. Betancourt (2015), "The Stan Math Library: Reverse-mode Automatic Differentiation in C++", arXiv: 1509.07164.

Chopin, N. (2002), "A sequential particle filter method for static models", *Biometrika* Vol. 89, No. 3, pp. 539–552.

Devroye, L. (1986), *Non-Uniform Random Variate Generation*, Springer, New York.

Gelfand, A. E., A. F. Smith (1990), "Sampling-based approaches to calculating marginal densities", *Journal of the American statistical association* Vol. 85, No. 410, pp. 398–409.

Gelman, A., D. B. Rubin (1992), "Inference from Iterative Simulation Using Multiple Sequences", *Statistical science* Vol. 7, No. 4, pp. 457–472.

Geman, S., D. Geman (1984), "Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images", *IEEE Transactions on Pattern Analysis and Machine Intelligence* Vol. 6, pp. 721–741.

Gilks, W. R., C. Berzuini (2001), "Following a moving target—Monte Carlo inference for dynamic Bayesian models", *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* Vol. 63, No. 1, pp. 127–146.

Hastings, W. K. (1970), "Monte Carlo sampling methods using Markov chains and their applications", *Biometrika* Vol. 57, No. 1, pp. 97–109.

Hoffman, M. D., A. Gelman (2014), "The No-U-Turn sampler: adaptively setting path lengths in Hamiltonian Monte Carlo.", *Journal of Machine Learning Research* Vol. 15, No. 1, pp. 1593–1623.

Kish, L. (1965), *Survey Sampling*, Wiley, New York.

Kong, A. (1992), *A Note on Importance Sampling using Standardized Weights*, tech. rep., University of Chicago, Dept. of Statistics.

Liu, J. S. (2004), *Monte Carlo Strategies in Scientific Computing*, Springer Series in Statistics, Springer-Verlag, New York.

Luce, R. D. (1959), *Individual choice behavior*, Wiley, New York.

Luce, R. D. (1977), "The choice axiom after twenty years", *Journal of Mathematical Psychology* Vol. 15, No. 3, pp. 215–233.

MacKay, D. J. (2003), *Information Theory, Inference and Learning Algorithms*, Cambridge University Press, Cambridge.

Marsaglia, G., W. W. Tsang (2000), "A Simple Method for Generating Gamma Variables", *ACM Trans. Math. Softw.* Vol. 26, No. 3, pp. 363–372.

Metropolis, N., A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, E. Teller (1953), "Equation of State Calculations by Fast Computing Machines", *The Journal of Chemical Physics* Vol. 21, No. 6, pp. 1087–1092.

Neal, R. M. (2003), "Slice Sampling", *The Annals of Statistics* Vol. 31, No. 3, pp. 705–767.

Neal, R. M. (2011), "MCMC Using Hamiltonian Dynamics", *Handbook of Markov Chain Monte Carlo*, Chapman & Hall / CRC Press, pp. 113–162.

Norris, J. R. (1998), *Markov Chains*, Cambridge University Press, Cambridge.

Russo, D. J., B. Van Roy, A. Kazerouni, I. Osband, Z. Wen (2018), "A Tutorial on Thompson Sampling", *Foundations and Trends® in Machine Learning* Vol. 11, No. 1, pp. 1–96.

Team, S. D. (2018), *Stan Modeling Language Users Guide and Reference Manual*, version 2.18.0, `http://mc-stan.org`, accessed on 20 June 2019.

Thompson, W. R. (1933), "On the Likelihood that One Unknown Probability Exceeds Another in View of the Evidence of Two Samples", *Biometrika* Vol. 25, No. 3/4, pp. 285–294.

Tierney, L. (1994), "Markov Chains for Exploring Posterior Distributions", *The Annals of Statistics* Vol. 22, No. 4, pp. 1701–1728.

APPENDIX A: CODE LISTINGS

A.1. Stan Model for the Sequential Form

```
data {
  int<lower=0> T;           // number of interactions
  int<lower=0> K;           // number of items
  int<lower=0> L;           // presentation size
  int<lower=0> Y[T, L];     // presentations
  int<lower=0> x[T];        // choices

  vector<lower=0>[K] alpha; // prior on theta
}
parameters {
  simplex[K] theta;
}
transformed parameters {
  vector[K] ltheta = log(theta);
}
model {
  theta ~ dirichlet(alpha);

  for (t in 1:T) {
    target += ltheta[x[t]] - log_sum_exp(ltheta[Y[t, ]]);
  }
}
```

A.2. Stan Model for the Sufficient Statistics Form

```
data {
    int<lower=0> K;           // number of options
    int<lower=0> L;           // presentation size
    int<lower=0> nY;          // number of unique presentations
    int<lower=0> nx;          // number of unique choices
    int<lower=0> Y[nY, L];    // unique presentations
    int<lower=0> x[nx];       // unique choices
    int<lower=0> mu[nY];      // multiplicity of presentations
    int<lower=0> gamma[nx];   // multiplicity of choices
    vector<lower=0>[K] alpha; // prior on theta
}
parameters {
    simplex[K] theta;
}
transformed parameters {
    vector[K] ltheta;
    ltheta = log(theta);
}
model {
    theta ~ dirichlet(alpha);
    for (i in 1:nx) {
        target += gamma[i] * ltheta[x[i]];
    }
    for (j in 1:nY) {
        target += -mu[j] * log_sum_exp(ltheta[Y[j, ]]);
    }
}
```