The Dissertation Committee for Yanxin Li
certifies that this is the approved version of the following dissertation:

Latent Slice Sampling

Committee:

Stephen G. Walker, Supervisor

Antonio R. Linero, Co-Supervisor

Mingyuan Zhou

Sinead Williamson

Bindu Viswanathan

Layla Guyot

Latent Slice Sampling

by

Yanxin Li

DISSERTATION

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT AUSTIN

May 2022

Dedicated to my loved ones.

Acknowledgments

Before I started this adventurous and enriching journey of doctoral studies, I was not aware of how challenging it is or how much I would learn along the journey. I could not have survived the challenges or enjoyed my time as much as I did without countless help or support from my advisors, colleagues, friends, and families.

First, I would like to thank my advisor, Dr. Stephen G. Walker, who has been an incredible mentor and academic role model for me. Dr. Walker is always passionate about solving research problems, strives for creative perspectives and keeps an open mind on new things, which teaches me what it is like to be a true researcher. Dr. Walker was accessible for any kind of guidance I needed, patient for solving any confusions I had and willing to help me improve my writing skills. I am also very grateful that Dr. Walker introduced me to his collaborator, Dr. Antonio Linero, who became my co-advisor since October 2021 and really helpd me to build my academic network and also facilitated my research. Moreover, Dr. Walker gave me a great amount of freedom to choose research problems that interest me. He is supportive whenever I was having a hard time in work or life, and I am beyond grateful for having him as my advisor.

Second, I would like to thank my co-advisor Dr. Linero, who guided me to realize the incomparable satisfaction of solving puzzles, and his honest and rigorous attitude toward science will affect me everlasting. Dr. Linero's patience, enthusiasm and wisdom make me reach further than I thought I could go.

I would also like to thank Dr. Zhou, Dr. Williamson, Dr. Viswanathan, and Dr. Guyot for serving on my doctoral committee with Dr. Walker and Dr. Linero. I owe an enormous debt of gratitude to Mingyuan for his mentorship during our project on deep probabilistic models and inference. I was fortunate to take courses with Dr. Williamson. Her innovative way of thinking always motivated me to think out of the box. I was honored to be the teaching assistant for Dr. Viswanathan and Dr. Guyot. I enjoyed the time we spent together guiding undergraduate students.

I am grateful for the support and companion from my friends: Xinjie Fan, Yuguang Yue, Mingzhang Yin, Junran Chen, and Mauricio Tec. I must apologize for not mentioning all your names here. You made my life more colorful. I also want to thank the staffs and the Department of Statistics and Data Science for offering the wonderful Ph.D. program and financial support.

This journey could not have begun without the recommendations from my mentors during my undergraduate and master studies, my boss and partners in my first job: Bei Hu, Fang Liu, Wanyi Chen, Jian Chen, Ling He, Yunquan Zhang. They encouraged and supported me to pursue a Ph.D. in statistics.

I will always be grateful for the unconditional love from my family. My parents had encouraged me to pursue a graduate degree when I found I was lacking of statistical fundamentals when I first worked as data analyst in China. They fostered my enthusiasm in learning and curiosity in science when I was a child. My uncle not only provided financial support during my master degree, but also instructional guidance when I met with difficulties. He is always my life mentor.

Finally, a special thank to my husband Dr. Bo Hong and my son Andrew W. Hong for bringing endless happiness into my life and being by my side through all the ups and downs.

Latent Slice Sampling

Publication No. ____________________

Yanxin Li, Ph.D.
The University of Texas at Austin, 2022

Supervisor: Stephen G. Walker

The thesis develops a new and generic Markov chain Monte Carlo sampling methodology, naming latent slice sampling, that originates from slice sampling and is capable of efficient sampling. More specifically, three angles are studied to cover different types of random variables: (i). We develop a latent slice sampler for discrete variables by designing a transition probability function that can perform direct sampling without knowing the exact form of target distributions. (ii). We manage to derive a latent slice sampler for continuous variables which has the potential to be a more efficient alternative to the Metropolis-Hasting algorithm, obviates the need for a proposal distribution, and has no accept/reject component. (iii). We further propose a novel algorithm based on latent slice sampling methodology which copes well with multi-modal problem, which can approach well-studied problems from a different angle and provide new perspectives. All the methods bring clear gains, which demonstrate the benefits of applying latent slice sampling to improve Markov chain simulation.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Markov chain Monte Carlo (MCMC) algorithms have been around for nearly 70 years and have become an important method for analyzing complex Bayesian models. Important impacts has been made in the early 1990s [51]. The strength of MCMC algorithms is that they guarantee convergence to the quantity (or quantities) of interest with minimal requirements on the target distribution. MCMC algorithms are robust or universal compared to Monte Carlo methods (see e.g., [139, 133]) which require proximity to the target distribution. The disadvantage to the robustness is that slow convergence to the target could be observed behavior, in that the exploration of the relevant part of the space supporting the distribution may take a long time. This can occur as the simulation usually proceeds by local jumps in the vicinity of the current position. In other words, MCMC like Gibbs sampling and Metropolis-Hastings algorithms, is very often myopic in that it provides a good illumination of a local region, while being unaware of the global support of the distribution. As with most other simulation methods, there always exist ways of creating highly convergent algorithms by taking further advantage of the structure of the target distribution.

Since Metropolis–Hastings based algorithms and Gibbs sampling have different origins, we will distinguish between the background of them, though their mathematical justification via Markov chain theory is the same. The Gibbs sampling is actually a special case of the Metropolis-Hastings algorithm.

Starting in the mid-to-late 1990s, the so-called "second-generation MCMC revolution" includes the development of particle filters, reversible jump and perfect sampling, and concludes with more current work on population or sequential Monte Carlo. The realization that Markov chains could be used in a vast range of situations only came to mainstream statistics with Gelfand and Smith [51], despite earlier publications in Hastings [78], Geman and Geman [59] and Tanner and Wong [146]. Through a series of applications, the method was demonstrated to be easy to understand, easy to implement and practical [53, 54, 141, 150]. The emergence of the Bayesian inference Using Gibbs Sampling (BUGS) software was another compelling argument for adopting MCMC algorithms.

1.1 Markov Chain Monte Carlo - A History

Monte Carlo methods originated during World War II, leading to the introduction of the Metropolis algorithm in the early 1950s, while MCMC was closely brought to statistical practice following the work of Hastings in the 1970s [78]. The Metropolis algorithm, published by Metropolis et al. [112], can be reasonably seen as the first MCMC algorithm. As early as 1949, Metropolis and Ulam [111] published the very first paper about the Monte Carlo method.

1.1.1 Metropolis et al. (1953)

The primary focus of Metropolis et al. [112] is the computation of integrals of the form

$$\bar{F} = \frac{\int F(\theta)\exp\{-E(\theta)/kT\}d\theta}{\int \exp\{-E(\theta)/kT\}d\theta}$$

where F is the value of system property of interest and θ denotes a set of N particles on $\mathbb{R}^2$, with the energy E being defined as

$$E(\theta) = \frac{1}{2}\sum_{i=1}^{N}\sum_{\substack{j=1\\ j\neq i}}^{N} V(d_{ij}(\theta))$$

where V is a potential function and d_{ij} the Euclidean distance between particles i and j in θ. The *Boltzmann distribution* $\exp\{-E(\theta)/kT\}$ is parameterized by the temperature T, with k being the Boltzmann constant, with a normalization factor

$$Z(T) = \int \exp\{-E(\theta)/kT\}d\theta.$$

Since θ is a $2N$-dimensional vector, numerical integration is impossible. For large-dimensional problem, even standard Monte Carlo methods fail to correctly approximate $\bar{F}$, since $\exp\{-E(\theta)/kT\}$ is very small for most realizations of the random configurations of the particle system. To improve the efficiency of the Monte Carlo method, Metropolis et al. [112] propose a random walk modification of the N particles. That is, for each particle i ($1 \leq i \leq N$), values x_i and y_i are moved in succession according to

$$x_i' = x_i + \sigma\xi_{1i} \quad \text{and} \quad y_i' = y_i + \sigma\xi_{2i}$$

where σ is the maximum allowed displacement, and both ξ_{1i} and ξ_{2i} are random numbers between -1 and 1. The energy difference ΔE between the new configuration and the previous one is then computed and the new configuration is accepted with probability

$$\min\{1, \exp(-\Delta E/kT)\} \tag{1.1}$$

and otherwise the previous configuration is replicated, in the sense that its counter is increased by one in the final average of the $F(\theta_t)$'s over the τ moves of the random walk, $1 \leq t \leq \tau$. Metropolis et al. [112] demonstrated the validity of the algorithm by establishing irreducibility and proving ergodicity, that is, convergence to the stationary distribution.

Simulated Annealing algorithm developed by Kirkpatrick, Gelatt and Vecchi [92] is an interesting variation, which connected optimization with annealing, the cooling of a metal. It is one of the most preferred heuristic methods for solving the optimization problems. Annealing procedure defines the optimal molecular arrangements of metal particles where the potential energy of the mass is minimized and refers cooling the metals gradually after subjected to high heat. Generally, Simulated Annealing allows the temperature T to change as the algorithm runs, according to a "cooling schedule", and the algorithm is shown to be able to find the global maximum with probability 1, although it is no longer a time-homogeneous Markov chain with varying T.

1.1.2 Hastings (1970)

The Metropolis algorithm was later generalized by Hastings [78] and Peskun [124, 125] as a statistical simulation tool that could overcome the curse of dimensionality. In Hastings' [78] *Biometrica* paper, the generic probability of acceptance for a move from state i to state j is

$$\alpha_{ij} = \frac{s_{ij}}{1 + (\pi_i/\pi_j)(q_{ij}/q_{ji})}$$

where s_{ij} is a symmetric function of i and j chosen so that $0 \leq \alpha_{ij} \leq 1$ for all i and j, $s_{ij} = s_{ji}$, and π_i denotes the target and q_{ij} the proposal. The above probability encompasses the forms of both Metropolis et al. [112] and Barker [9]. At this stage, Hastings warns against high rejection rates as indicative of a poor choice of transition matrix, but does not mention the opposite pitfall of low rejection rates, which is associated with a slow exploration of the target distribution.

In the paper, Hastings introduces a Gibbs sampling strategy, updating one component at a time and defining the composed transition matrix as satisfying the stationary condition because each component leaves the target invariant, that is

$$\pi_i \, p_{ij} = \pi_j \, p_{ji} \tag{1.2}$$

where $P = \{p_{ij}\}$ is the transition matrix of the Markov chain. The property of (1.2) ensures that $\sum \pi_i p_{ij} = \pi_j$, for all j, and hence that π is a stationary distribution of P.

Hastings [78] actually refers to [42] as a preliminary instance of this sampler, which precisely is Metropolis-within-Gibbs sampler. This is the first introduction of the Gibbs sampler with a completely general proof of convergence based on a composition argument as in Tierney [149]. The remainder of Hastings' paper deals with (i) an importance sampling version of Markov chain Monte Carlo, (ii) general remarks about assessment of the error, and (c) an application to random orthogonal matrices, with another example of Gibbs sampling. Peskun [124] published a comparison of Metropolis' and Barker's forms of acceptance probabilities three years later, which shows the optimal choice the asymptotic variance of any empirical average in a discrete setup. Peskun [124] also establishes that this asymptotic variance can improve upon the independent and identically distributed (i.i.d.) case if and only if the eigenvalues of $P - A$ are all negative, when P is the transition matrix corresponding to the Metropolis algorithm and A the transition matrix corresponding to i.i.d. simulation.

1.1.3 The EM Algorithm

Besides Hastings [78] and Geman and Geman [59], other papers that contained the seeds of Gibbs sampling are Besag and Clifford [13, 11, 12], Tanner and Wong [146], Qian and Titterington [128]. In the early 1970's, Hammersley, Clifford and Besag were working on the specification of joint distributions from conditional distributions and on necessary and sufficient conditions for the conditional distributions to be compatible with a joint distribution. The *Hammersley-Clifford* theorem states that a joint distribution for a vector associated with a dependence graph (edge meaning dependence and absence of edge conditional independence) must be represented as a product of functions depending only on the components indexed by the labels in the clique.

Hammersley [74] explains the reason why the *Hammersley-Clifford* theorem was published only through Besag[11] is that Clifford and Hammersley were dissatisfied with the positivity constraint: the joint density could be recovered from the full conditionals only when the support of the joint was made of the product of the supports of the full conditionals. Hammersley and Handscomb [72] also expressed a more optimistic sentiment earlier in their textbook on Monte Carlo method, where they cover such topics as "Crude Monte Carlo", importance sampling, control variates and "Conditional Monte Carlo", which looks like a missing-data completion approach.

Because of its use for missing data problems, the EM (Expectation-Maximization) algorithm [36] has early connections with Gibbs sampling. For instance, Celeux and Diebolt [26] had tried to overcome the dependence of EM methods on the starting value by replacing the E step with a simulation step, the missing data z being generated conditionally on the observation x and on the current value of the parameter θ_m. The maximization in the M step is then done on the simulated complete-data log-likelihood, a predecessor to the Gibbs step of Diebolt and Robert [37] for mixture estimation. Celeux and Diebolt [27] have also solved the convergence problem by devising a hybrid version called Simulated Annealing EM, where the amount of randomness in the simulations decreases with the iterations, ending up with an EM algorithm.

1.1.4 Gibbs Sampling

The landmark paper which brought Gibbs sampling into the mainstream arena of statistical application is by Geman and Geman [59], which is also responsible for the name Gibbs sampling, with the original implementation on a discrete image processing problem. Described by brothers Stuart Geman and Donald Geman [59], Gibbs sampling is a special case of the Metropolis–Hastings algorithm. However, in its extended versions, it can be considered a general framework for sampling from a large set of variables by sampling each variable in turn, and can incorporate the Metropolis–Hastings algorithm (or methods such as slice sampling [119]) to implement one or more of the sampling steps.

Gibbs sampling is applicable when the joint distribution is not known explicitly or is difficult to sample from directly, but the conditional distribution of each variable is known and is easy (or at least, easier) to sample from. Suppose $p(x, y)$ is a probability density function or probability mass function that is difficult to sample from directly, but the conditional distributions $p(x \mid y)$ and $p(y \mid x)$ is easily to sample from, then the Gibbs sampler proceeds as follows:

i. Set (x, y) to some initial starting values (x_0, y_0)

ii. Sample $x_1 \sim p(x \mid y_0)$, then sample $y1 \sim p(y \mid x_1)$, and then sample

$x_2 \sim p(x \mid y_1)$, and so on.

The Gibbs sampling algorithm generates an instance from the distribution of each variable in turn, conditional on the current values of the other variables. It can be shown that the sequence of samples constitutes a Markov chain, and the stationary distribution of that Markov chain is just the sought-after joint distribution [58]. As illustrated in [131], Gibbs sampling and Metropolis algorithms were extensively in use within the image analysis and point process communities. Besag, York and Mollié [15] is another example of the activity in the spatial statistics community at the end of the 1980s.

1.2 The MCMC Revolution

After Peskun [124, 125], MCMC in the statistical world was dormant for about 10 years, and then several papers appeared that highlighted its usefulness in pattern recognition, image analysis or spatial statistics. In particular, the genuine starting point for an intensive use of MCMC methods by the mainstream statistical community is from the paper written by Gelfand and Smith [52]. It sparked new interest in Bayesian methods, statistical computing, algorithms and stochastic processes through the use of computing algorithms such as the Metropolis–Hastings algorithm and the Gibbs sampler [25].

In the paper of Tanner and Wong [146], the idea that simulating from the conditional distributions is sufficient to asymptotically simulate from the joint is essentially the same as Gelfand and Smith [52]. Compared with Gelfand

and Smith [52], this paper's impact was somehow limited, due to the reasons that the method seemed to only apply to missing data problems, this impression being reinforced by data augmentation, and the authors were more focused on approximating the posterior distribution. The basic algorithm is motivated by a simple representation of the desired posterior density:

$$p(\theta \mid x) = \int p(\theta \mid z, x) p(z \mid x) dz$$

where $p(\theta \mid x)$ denotes the posterior density of the parameter θ given the data x, $p(z \mid x)$ denotes the predictive density of the latent data z given x, and $p(\theta \mid x)$ denotes the conditional density of θ given the augmented data (z, x). Tanner and Wong [146] suggested a Markov chain Monte Carlo approximation to the target $p(\theta \mid x)$ at each iteration of the sampler, based on

i. Generate a sample $(z^{(1)}, \ldots, z^{(m)})$ from the current approximation to the predicative density $p(z \mid x)$

ii. Update the current approximation to $p(\theta \mid x)$ to be the mixture of conditional density of θ given the augmented data generated in (i), i.e.,

$$\theta \sim \frac{1}{m} \sum_{j=1}^{m} p(\theta \mid x, z^{(j)})$$

that is, by replicating m times the simulations from the current approximation to $p(z \mid x)$ of the marginal posterior distribution of the missing data. This focus on estimation of the posterior distribution connected the original Data Augmentation algorithm to EM.

In 1991, many talks were to become influential papers, including Albert and Chib [1], Gelman and Rubin [55], Geyer [63], Gilks and Wild [66], Liu, Wong and Kong [87, 88] and Tierney [149], in an important Markov chain Monte Carlo conference at Ohio State University. Approximately one year later, four papers were presented followed by a discussion in a meeting of the *Royal Statistical Society* on "The Gibbs sampler and other Markov chain Monte Carlo methods".

Perhaps the most influential MCMC theory paper of the 1990s is Tierney [149], who carefully laid out all of the assumptions needed to analyze the Markov chains and then developed their properties, in particular, convergence of ergodic averages and central limit theorems. Liu, Wong and Kong [87, 88] carefully analyzed the covariance structure of Gibbs sampling and formally established the validity of Rao–Blackwellization in Gibbs sampling. Rosenthal [138] obtained one of the earliest results on exact rates of convergence, which is another significant entry. Mengersen and Tweedie [110] set the tone for the study of the speed of convergence of MCMC algorithms to the target distribution. Subsequent works in this area are numerous, with the paper by Roberts, Gelman and Gilks [49] being important for setting explicit targets on the acceptance rate of the random walk Metropolis–Hastings algorithm, as well as Roberts and Rosenthal [135] for getting an upper bound on the number of iterations needed to approximate the target up to 1% by a slice sampler. One pitfall arising from the widespread use of Gibbs sampling was the tendency to specify models only through their conditional distributions, almost always without referring to the positivity conditions. Unfortunately, it is possible to specify a perfectly legitimate-looking set of conditionals that do not correspond to any joint distribution, and the resulting Gibbs chain cannot converge. Hobert and Casella [80] were able to document the conditions needed for a convergent Gibbs chain, and alerted the Gibbs community to this problem, which only arises when improper priors are used.

Much other work followed, and continues to grow. Followed by Neal's [118] introduction of tempering, Geyer and Thompson [65] describe how to put a "ladder" of chains together to have both "hot" and "cold" exploration; Athreya, Doss and Sethuraman [7] gave more easily verifiable conditions for convergence; Meng and van Dyk [109] and Liu and Wu [104] developed the theory of parameter expansion in the Data Augmentation algorithm, leading to construction of chains with faster convergence, and to the work of Hobert and Marchev [81], who give precise constructions and theorems to show how parameter expansion can uniformly improve over the original chain. The reason of the explosion of MCMC methods lies in the fact that an numerous number of problems that were deemed to be computationally intractable could now be solved.

During the early 1990s, researchers found that Gibbs sampling or Metropolis–Hastings algorithms would be able to give solutions to almost any problem that they looked at, and there was a veritable flood of papers applying MCMC to previously intractable models, and getting good answers. For example, Gibbs sampling was quickly realized as an easy route to getting estimates in the linear mixed models [153, 154], and even generalized linear mixed models [156]. Building on the experience gained with the EM algorithm, similar arguments made it possible to analyze probit models using a latent variable approach in a linear mixed model [1], and in mixture models with Gibbs sampling [37]. It progressively dawned on the community that latent variables could be artificially introduced to run the Gibbs sampler in about every situation; see [35] and [119]. An incomplete list of some other applications include changepoint analysis [18, 143], genomics [31, 98, 142], capture–recapture [41, 61], variable selection in regression [60], spatial statistics [129], and longitudinal studies [96].

1.3 Recent MCMC Practice

Problems are now being solved in perhaps deeper and more sophisticated ways. Methodology continues to expand the set of problems for which statisticians can provide meaningful solutions, and thus continues to further the impact of Statistics.

The realization of the possibilities of iterating importance sampling is about as old as Monte Carlo methods themselves, which can be found in Hammersley and Morton [73], Rosenbluth and Rosenbluth [137]. Hammersley and colleagues proposed such a method to simulate a self-avoiding random walk (see [107]) on a grid, due to huge inefficiencies in regular importance sampling and rejection techniques. This early implementation occurred in particle physics and the term “particle” was coined as “particle filter” Carpenter, Clifford and

Fernhead [23]. In signal processing, early occurrences of a particle filter can be traced back to Handschin and Mayne [76]. The paper [69] introduced the bootstrap filter which involves past simulations and possible Markov chain Monte Carlo steps [67]. As described by Doucet, de Freitas and Gordon [40], particle filters are simulation methods adapted to sequential settings where data are collected progressively in time. The methods produce Monte Carlo approximations to the posterior distributions by propagating simulated samples whose weights are actualized against the incoming observations. Modern connections with Markov chain Monte Carlo in the construction of the proposal kernel are to be found, for instance, in [39, 115]. At the same time, sequential imputation was developed in Kong, Liu and Wong [93], while Liu and Chen [103] first formally pointed out the importance of resampling in sequential Monte Carlo, a term coined by them. The recent literature on the topic more closely bridges the gap between sequential Monte Carlo and MCMC methods by making adaptive MCMC a possibility (see, e.g., [4, 136]).

Perfect sampling was introduced by Propp and Wilson [127], of which the ability is to use MCMC methods to produce an exact simulation from the target. The discovery of perfect sampling led to an outburst of papers, including the book by Møller and Waagepetersen [86], and many reviews and introductory materials, such as [24, 38, 47, 48]. However, the construction of perfect samplers is most often close to impossible or impractical.

In the area of point processes and stochastic geometry, Kendall and Møller [91] developed an alternative to the Coupling From The Past (CFPT) algorithm of Propp and Wilson [127], called horizontal CFTP for the point processes, based on continuous time birth-and-death processes. See also [46] for another horizontal CFTP algorithm. Berthelsen and Møller [10] exhibited a use of these algorithms for nonparametric Bayesian inference on point processes.

1.3.1 Reversible Jump and Variable Dimension

From many viewpoints, the emergence of reversible jump algorithm in [70] can be seen as the start of the second MCMC revolution: the formalization of a Markov chain that moves across models and parameter spaces allowed for the Bayesian processing of a wide variety of new models and contributed to the success of Bayesian model choice and subsequently to its adoption in other fields. There exist earlier alternative Monte Carlo solutions like Gelfand and Dey [50] and Carlin and Chib [22], the later being very close to reversible jump MCMC (as shown in [19]), but the definition of a proper balance condition on cross model Markov kernels in [70] gives a generic setup for exploring variable dimension spaces, even when the number of models under comparison is infinite. This new idea leads to a large majority of the talks aimed at direct implementations of RJMCMC to various inference problems at the First European Conference on Highly Structured Stochastic Systems, which took place in the next year. The application of RJMCMC to mixture order estimation in the discussion paper of Richardson and Green [130] ensured further dissemination of the technique. Continuing to develop RJMCMC, Stephens [144] proposed a continuous time version of RJMCMC, based on earlier ideas of Geyer and Møller [64], but with similar properties [21], while Brooks, Giudici and Roberts [19] made proposals for increasing the efficiency of the moves. In retrospect, while reversible jump is somehow unavoidable in the processing of very large numbers of models under comparison, as, for instance, in variable selection [108], the implementation of a complex algorithm like RJMCMC for the comparison of a few models is somewhat of an overkill since there may exist alternative solutions based on model specific MCMC chains, for example, [6].

Godsill [68] defined a composite model space for standard model selection problems in which no parameters are considered as "shared" between any two models. It is later modified to introduce more flexibility in shared parameters problems such as nested models and model selection. The reversible jump sampler for the composite model was derived by considering the proposal which forms a joint distribution over all elements of the model index j and model parameters θ. It is split into three component parts: the model index component $q_1(j' \mid j)$, which proposes a move to a new model index, j'; a proposal for the parameters used by model j', $q_2(\theta'_{j'} \mid \theta_j)$; and a proposal for the remaining unused parameters which is chosen to equal to the pseudo-prior $p(\theta'_{-j'} \mid \theta'_{j'}, j')$. The applications include mixtures with an unknown number of components [130], variable selection, and so on.

1.3.2 Slice Sampling

Slice sampling [119] is an alternative to Gibbs sampling that avoids the need to sample from nonstandard distributions. The main idea of slice sampling is formalized by introducing an auxiliary real variable w, and defining a joint distribution over x and w that is uniform over the region $U = \{(x, w) : 0 < w < f(x)\}$. For single-variable slice sampling, the variation of slice sampling proposed by Neal operates analogously to Gibbs sampling in the sense that to obtain the next point x_1, w is generated from the conditional distribution $[w \mid x_0]$ given the current point x_0 and then x_1 is drawn from $[x \mid w]$. Both $[w \mid x_0]$ and $[x \mid w]$ are uniform distributions. Since the closed form of the support of $[x \mid w]$ is not available, sampling directly from $[x \mid w]$ is not possible. A clever development is Neal's sophisticated (but relatively expensive) sampling procedure to generate x_1 from the "slice" $S = \{x : w < f(x)\}$.

The ith realization of x is constructed according to Algorithm 1 in Appendix C. A sample or "height" under the distribution, w^i, is drawn uniformly from the interval $(0, f(w^{i-1}))$. This height, w^i , defines a horizontal slice across the target density, $S = \{x : f(x) > w^i\}$, which is then sampled from uniformly to generate x^i . In the univariate case, the set $S = \{x : f(x) > w^i\}$ is simply an interval or, perhaps more generally, the union of several intervals (such as in the presence of multiple modes). In contrast, in the multivariate case, the set $\{x : f(x) > w^i\}$ may have a much more complicated form.

Quite often, one lacks an analytic solution for the bounds of the slice S and so, in practice, an approximation to the slice A is constructed. For the single dimension case [119] suggested two methods, stepping out and doubling, to approximate the set S, though we only consider the step-out approach here. This is done by randomly orienting an interval around the starting location x^{i-1}. The lower bound L^i and upper bound U^i are examined, and if either $f(L^i)$ or $f(U^i)$ is above the sampled height w^i , then the interval is extended. Once the interval is constructed, a new location x^i is selected from (L^i, U^i) provided $x^i \in \{x : f(x) > w^i\}$. The sample w^i is then discarded, a new sample w^{i+1} is drawn, and the process repeats. The resulting Markov Chain has the desired stationary distribution. While the preceding description applies to both the step-out and doubling approaches, we now explain the step-out method in detail. In the step-out method, the lower bound is examined first and extended in steps equal to the initial interval width (d) if $f(L^i)$ is above the sampled height w^i. The upper bound is then examined and extended if needed. Once the interval is constructed, a proposed parameter value, $\tilde{x}$, is drawn uniformly from (L^i, U^i). If it falls outside the target slice, $(f(\tilde{x}) < w^i)$, a shrinkage procedure is recommended to maximize sampling efficiency. If the failed proposal $\tilde{x}$ is less than x^{i-1}, then set $L^i = \tilde{x}$. Likewise, if $\tilde{x}$ is greater than x^{i-1}, set $U^i = \tilde{x}$. In this way, the interval collapses on failed proposals and given that the current location must be within the slice, the probability of drawing a point from the slice then increases after each rejected proposal.

In contrast to the univariate slice sampler, which samples from the distribution of a random variable $x \in \mathbb{R}^1$, the multivariate slice sampler, which samples from the distribution of $x \in \mathbb{R}^n$, constructs an approximate slice S as a n-dimensional hypercube which bounds the target slice. As before, we update the variable x by drawing a sample w^i uniformly from the interval $(0, f(x^{i-1}))$ where the current location, x^{i-1}, is now a n-dimensional vector. Next, an interval is randomly oriented around the starting location x_j^{i-1} for each vector component. Then the value of the target density is examined at the vertices of the hypercube, which we will refer to as the lower bound vector L^i and the upper bound vector U^i . If the value of the density at any vertex falls below the sampled height w^i, then the hypercube is expanded. Once the hypercube is constructed, a new location x^i is sampled uniformly from S subject to the constraint that $x^i \in \{x : f(x) > w^i\}$. The sample w^i is then discarded, a new sample w^{i+1} is drawn, and the process repeats.

Multivariate slice sampling is challenging due in large part to the number of likelihood evaluations required at each iteration. First, the number of vertices, 2^n, for the n-dimensional hypercube used to approximate the target slice S grows exponentially as the dimension of the multivariate slice sampler increases. From a computational standpoint, the work doubles for each additional dimension considered. Second, as the dimensionality of the target distribution increases, the n-dimensional hypercube is more likely to waste space, and consequently, the performance of rejection sampling for the proposal step will deteriorate. One final issue relates to the tuning and selection of initial interval widths for the hypercube approximation. In shrinking the hypercube in the obvious way (when proposals fall outside the slice), shrinking all dimensions simultaneously performs poorly when the density does not vary rapidly in some dimensions.

Tuning and selection of interval widths may also be challenging. To address this issue, we performed a grid search to find optimal interval widths which maximized Effective Sample Size per second.

When implementing the univariate and multivariate slice samplers, the step-out method is chosen for constructing the approximate slice. Mira and Roberts [113] note that the step-out method is unable to move between two disjoint modes that are separated by a region of zero probability where these regions are larger than the step size. This implies that the sampler may not be irreducible for some multi-modal distributions when the initial step size is too small.

To address all those issues, different sampling methods were proposed base upon the slice sampling. Parallel multivariate slice sampling [147] is constructed that naturally lends itself to a parallel implementation, which has good mixing properties and is efficient in terms of computing time. The elliptical slice sampling [116] is able to perform inference in models with multivariate Gaussian priors, which works well for a variety of Gaussian process based models. Kalli, Griffin, and Walker [89] proposed a slice-efficient sampler for Dirichlet process mixture models described by Walker [151]. Automated factor slice sampler [148] generalized the the univariate slice sampler by treating treat the selection of a coordinate basis (factors) as an additional tuning parameter automatically selecting tuning parameters to construct an efficient factor slice sampler.

1.3.3 Adaptive Monte Carlo on Multivariate Binary Spaces

In this section, we review some of the Markov transition kernels typically used for MCMC on binary spaces. Many popular Metropolis-Hastings kernels on binary spaces perform a random walk by proposing moves to neighboring states. The random scan Gibbs sampler draws an index i and samples the ith component from the full conditional distribution, while the Metropolized Gibbs sampler uses deterministic flips, which is a Metropolis-Hasting type proposal. On average, a Markov chain with deterministic flips moves faster than the classical random scan Gibbs chain since the Metropolis-Hastings step performs uniform block updating to alter a block of entries.

Swendsen and Wang [145] propose a sampling procedure that introduces a vector of auxiliary variables u such that $\pi(u \mid x)$ is a distribution of mutually independent uniforms and $(x \mid u)$ a distribution with components which are either fixed by constraints or conditionally independent. Higdon [79] suggests to parameterize and control the size of the conditionally independent blocks to further improve the mixing properties. Nott and Green [122] attempt to adapt the rationale behind the algorithm to sampling from a broader class of binary distributions. However, the efficiency gain of the Swendsen-Wang algorithm does not easily carry over to general binary sampling due to the fact that it is based on the exponential multi-linear structure of the distribution of interest.

The Metropolis-Hastings algorithm allows to incorporate any proposal, but obviously not all choices yield good MCMC estimators. In most practical cases, the problem that the parameter θ needs to be calibrated against the target distribution π still exists, even though a suitable family of auxiliary kernels is identified. The obvious idea is to adapt the algorithm to improve the choice of θ. There has been a major interest in adaptive Markov chain Monte Carlo (AMCMC) and convergence results have been established which hold on finite spaces under very mild conditions [136]. For further details on AMCMC we refer to [3] and citations therein. We will review some AMCMC algorithms for sampling on binary spaces in the following.

An adaptive extension of the Gibbs sampler has been proposed by Nott and Kohn [121]. A direct proof of convergence for their AMCMC algorithms is also provided, which needs less preparation than the technical proofs for the general state spaces [136]. The full conditional distribution is the optimal choice in terms of acceptance rates, but oftentimes the chain does not move because the current state has been sampled again. Lamnisos et al. [95] propose to calibrate the distribution of the number of bits to be fipped on average, where they take $\omega = \text{Binomial}(p; n)$ to be a binomial distribution with success probability p. Their work is motivated by the adaptive random walk algorithm developed by Atchadé and Rosenthal [2] for continuous state spaces where the variance of the multivariate normal random walk proposal is adjusted to meet the (asymptotically) optimal acceptance probability. However, in the context of binary spaces the major problem faced is multi-modality. Atchadé and Rosenthal [2] proposed a method for high-dimensional unimodal sampling problems, but the rationale behind the design of the algorithm does not necessarily carry over to multi-modal discrete problems.

Adaptive MCMC algorithms provide an astonishing speed-up over their non-adaptive versions for unimodal distributions and for high-dimensional sampling problems on continuous spaces. Still, it is notoriously difficult to adapt an MCMC sampler to a multi-modal sampling problem. More advanced MCMC algorithms which use parallel tempering ideas combined with more elaborate local moves [17] or self-avoiding dynamics [75] are proposed to overcome the multi-modality problem. However, it seems difficult for these algorithms to tune automatically.

As an alternative to MCMC sampling, Clyde et al. [33] develop the Bayesian adaptive sampling procedure which draws binary vectors without replacement. The idea is to update the conditional probabilities to ensure that each binary vector is only sampled once. The algorithm starts sampling with some initial mean which is then updated using current estimate of the mean of interest. The updating step of the conditional probabilities scannot be performed after every single sampling step. From a computational perspective this seems reasonable. Schäfer and Chopin [30] introduced a fully adaptive resample-move algorithm for sampling from binary distribution using sequential Monte Carlo [115] methodology. This general class of algorithms alternates importance sampling steps, resampling steps and Markov chain transitions, to recursively approximate a sequence of distributions, using a set of weighted "particles" which represent the current distribution.

1.3.4 Latent Slice Sampling

The latent slice sampling algorithm we developed, as described from Chapter 2 to Chapter 4 is a generic sampling algorithm which has the ability to address the issues raised in reversible jump sampler, slice sampling, Metropolis-Hastings, and adaptive Monte Carlo sampling algorithm. It is able to sample efficiently from very high dimensional distributions and implicit distributions. The key is the latent model combined with the shrinkage procedure based on uniform distributions and an automatic reversible condition, as detailed in Chapter 3.

We modified the algorithm to be applied in many cases, including applications such as MDP model, mixture finite mixtures, model selection, and multiple change-point problem in the field of discrete variables, as shown in Chapter 2. The proposed transition kernel for discrete variables is exempt from the necessity of proposal distribution and the normalizing constant of the target distribution. The parameter, which determines the number of steps transitioned from current state to a new state, is the only parameter that is required to be tune to account for both the computational efficiency and the auto-correlations between successive samples.

Chapter 3 extended the latent slice sampler of Chapter 2 to be applied for the continuous variables. The stochastic search introduced together with the shrinkage procedure of the sampling process leads the implementation to a vast range of applications and to be a universal replacement of Metropolis algorithm. The illustrations of Chapter 3 cover state space model, spike and slab regression analysis, and uniform sampling of high dimensional data with respect to continuous variables. Chapter 4 exploited a slice sampling algorithm in continuous space in order to sample a joint distribution on binary values. Such distributions arise in classic contexts and are known to be problematic to sample when the dimension is large and/or the distribution is multi-modal, like Ising model, variable selection, and Bayesian CART model. The newly modified sampling algorithm works by being able to propose a move to any location from any current location with almost uniform probability. With numerous number of illustrations, we show that the latent slice sampling method can be a substitute of commonly used MCMC sampling methods in many applications with highly improved computational efficiency and no accept/reject component.

Chapter 2

Latent Slice Sampler of Discrete Variables

Several Markov chain methods are available for sampling from a posterior distribution. Two important examples are the Gibbs sampler and the Metropolis algorithm [78, 112]. In addition, many strategies are available for constructing hybrid algorithms. The Metropolis algorithm has been placed among the algorithms that have greatest influence on the development and practice of science and engineering. It is extremely versatile and gives rise to Gibbs sampling as a special case; as pointed out by Gelman [55].

Gibbs sampling is applicable when the joint distribution is not known explicitly or is difficult to sample from directly, but the conditional distribution of each variable is known and is easy (or at least, easier) to sample from. The Gibbs sampling algorithm generates an instance from the distribution of each variable in turn, conditional on the current values of the other variables. It can be shown that the sequence of samples constitutes a Markov chain, and the stationary distribution of that Markov chain is just the sought-after joint distribution [57]. Gibbs sampling is particularly well-adapted to sampling the posterior distribution of a Bayesian network, since Bayesian networks are typically specified as a collection of conditional distributions.

However, there are several limitations to it. First, even if we have the full posterior joint density function, it may not be possible or practical to derive the conditional distributions for each of the random variables in the model. Second, even if we have the posterior conditionals for each variable, it might be that they are not of a known form, and therefore there is not a straightforward way to draw samples from them. Finally, there are cases in which Gibbs sampling will be very inefficient. That is, the "mixing" of the Gibbs sampling chain might be very slow, meaning that the algorithm may spend a long time exploring a local region with high density, and thus take very long to explore all regions with significant probability mass [106]. For example, when the cross-correlation of the posterior conditional distributions between variables is high, successive samples become very highly correlated and sample values change very slowly from one iteration to the next, resulting in chains that basically do not mix.

The Metropolis-Hastings algorithm simulates samples from a probability distribution by making use of the full joint density function and (independent) proposal distributions for each of the variables of interest. It involves sampling a candidate value given the current value according to the proposal. The Markov chain then moves toward candidate with certain acceptance probability, otherwise it remains at the current state. The Metropolis-Hastings algorithm is very simple, but it requires careful design of the proposal distribution. Many MCMC algorithms arise by considering specific choices of this distribution.

As the extension to the scope of Metropolis-Hastings methods, the reversible Markov chain samplers [70] that jump between parameter subspaces of differing dimensionality has then been proposed, which can be applied to Bayesian model determination problems. Previous work on Markov chain Monte Carlo computation with application to aspects of Bayesian model determination includes [126], based on the jump-diffusion samplers [71, 22]. However, there is a conflict between minimizing the distortion caused by using a positive time increment, and improving Monte Carlo efficiency. The reversible jump MCMC requires relative normalizing constants between different subspaces and proceeds with a Metropolis step when sampling varying dimensional problems.

The Metropolis-Hastings algorithm has become the most popular MCMC method. However, the success or failure of the algorithm often hinges on the choice of the proposal distribution. Different choices of the proposal lead to very different results. If the proposal is too narrow, only one mode of the target distribution might be visited. On the other hand, if it is too wide, the rejection rate can be very high, resulting in high correlations. If all the modes are visited while the acceptance probability is high, the chain is said to "mix" well.

The aim of this chapter is to introduce a new MCMC sampling method for discrete variables, such as latent variables in mixture models, and model indicator in the context of model determination. Section 2.1 introduces an latent slice sampler for discrete variables and its numerical properties. Section 2.2 describes an application on the mixture of Dirichlet process and comparisons with the "independent" slice-efficient sampler [89]). In Section 2.3, the reversible jump MCMC method on Bayesian model determination is discussed and compared with both the Metropolis step and the latent slice sampler. Section 2.4 is the experimental study on the applications in Section 2.2 and Section 2.3, as well as a multiple change-point problem discussed by Green [70]. Section 2.5 contains conclusions and a discussion.

2.1 Latent Slice Sampler for Discrete Variables

A transition density is proposed such that it satisfies the detailed balance equation and allows direct sampling with the conditional distributions. The transition kernel requires no auxiliary variables and proposal distribution (i.e., the candidate-generating density) for the sampling procedure of the infinite dimensional and varying-dimension problems. The proposed transition kernel is given by

$$p_k(x' \mid x) = \frac{\pi(x')}{k} \sum_{j=\max(x',x)}^{\min(x'+k-1,x+k-1)} \frac{1}{\sum_{z=\max(1,j-k+1)}^{j} \pi(z)} \tag{2.1}$$

where $x \in \{1, 2, \dots\}$ and the normalizing constant of $\pi(x)$ is unknown, $k > 1$ and $|x' - x| \leq k - 1$. It is easy to show that $p_k(x' \mid x)$ satisfies reversibility condition. See Appendix A for the complete proof of the validity of the transition kernel as a probability mass function.

The choice of k determines the convergence rate and the correlations of the successive samples. Fig. 2.1 shows the plots of the autocorrelation functions (ACF), which illustrates how correlated points are with each other, based on how many time steps they are separated by. The samples are simulated from a Poisson distribution with mean equal to 3, i.e., $\pi(x) = \frac{3^x e^{-3}}{x!}$. Typically, ACF will fall towards 0 as points become more separated. As k increases, the ACF approaches towards 0 with less time periods, i.e., the lags. The number of lags is decreasing from 4 to 2 when $k > 12$. This is very important when it comes to good mixing of the sampling chain. However, the computational workload grows substantially as k increases, see Fig.2.2. The elapsed time which represents the total duration of the task is increasing exponentially as k increases. Therefore, it is necessary to strike a balance between good mixing and efficient computation. Without loss of generality, we choose $k = 6$ for comparison purpose and fast convergence.

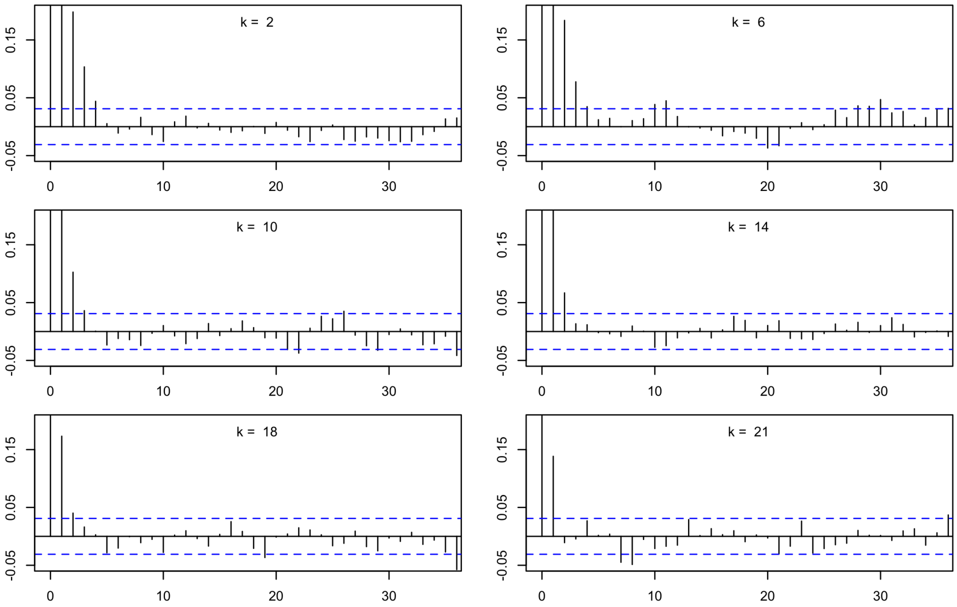


Figure 2.1: Plots of ACF with $k = 2, 6, 10, 14, 18, 22$.

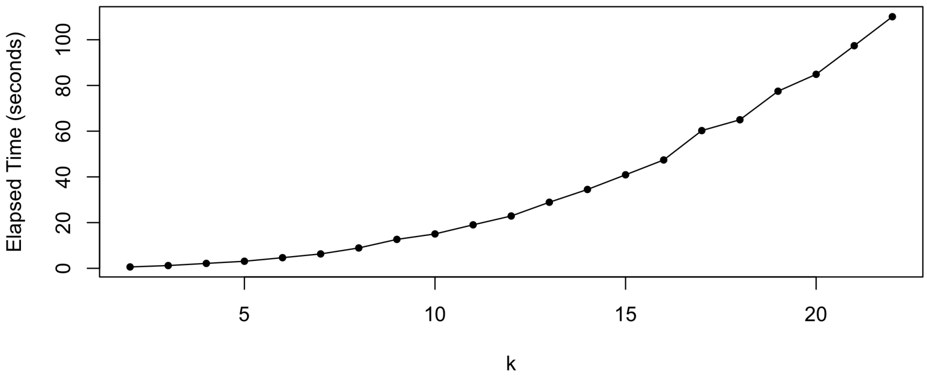


Figure 2.2: Plot of elapsed time in seconds with k varying between 2 and 22.

2.2 Mixture of Dirichlet Process

The well-known and widely used mixture of Dirichlet process (MDP) model [105] is a good example where the indicator variable is discrete and trivial to sample. The MDP model with Gaussian kernel is given by

$$f(x) = \int \mathcal{N}(x; \mu, \sigma^2) dP(\theta)$$

where $\theta = (\mu, \sigma^2)$ with μ to represent the mean and σ^2 the variance of the normal component. Let $\mathrm{DP}(\alpha, P_0)$ denote a Dirichlet process prior [45] with scale parameter $\alpha > 0$ and a prior probability P_0 on the component parameters. The model has been one of the most popular in Bayesian nonparametrics since it is possible to integrate P from the posterior defined by this model. Many so-called "conditional" methods have left the infinite dimensional distribution in the model and found ways of sampling a sufficient but finite number of variables at each iteration of a Markov chain with correct stationary distribution. Ishwaran and James [83] proposed an approximate method and Walker [151] used slice sampling ideas. The following will use the latent slice sampler for MDP model, which avoids dealing with the infinite dimensional problem, and compare with the "independent" slice-efficient sampler.

The MDP model can be written as

$$f(x_i \mid w, \theta) = \sum_{j=1}^{\infty} w_j f(x_i \mid \theta_j) = \sum_{j=1}^{\infty} w_j \mathcal{N}(x_i \mid \theta_j) \tag{2.2}$$

One can then introduce latent variables d_i's, which identify the component of the mixture from which x_i is to be taken, the model then becomes

$$f(x_i, d_i \mid w, \theta) = w_{d_i} \mathcal{N}(x_i \mid \theta_{d_i}) \tag{2.3}$$

Let $x = (x_1, x_2, \dots, x_n)$ and $d = (d_1, d_2, \dots, d_n)$, the complete data likelihood based on a sample of size n is easily seen to be

$$l(x, d \mid w, \theta) = \prod_{i=1}^{n} w_{d_i} \mathcal{N}(x_i \mid \theta_{d_i}) \tag{2.4}$$

For Bayesian nonparametric inference, an elegant constructive characterization of the Dirichlet process is given by the stick-breaking representation [140], which is used as a prior process for generating the mixing proportions of the infinite mixture distribution in (2.2). The stick-breaking representation metaphorically views $\{w_1, w_2, w_3, \dots\}$ as pieces of a unit-length stick that is sequentially broken in an infinite process, with stick-breaking proportions $V = \{v_1, v_2, v_3, \dots\}$, according to independent realizations of a Beta distribution. The stick-breaking process is summarized as follows

$$
\begin{aligned}
v_j &\sim \text{Beta}(1, \alpha) \\
w_j &= v_j \prod_{l=1}^{j-1} (1 - v_l), w_1 = v_1 \\
P &= \sum_{j=1}^{\infty} w_j \delta_{\theta_j} \sim \text{DP}(\alpha, P_0)
\end{aligned}
$$

where δ_θ is the Dirac delta centered at θ, such that draws are composed of a sum of infinitely many point masses.

The prior for the parameters μ_j's will be independent $\mathcal{N}(0, 1/s)$ and the prior for λ_j's will be independent Gamma(τ, τ). Generally, a set of full conditional density functions are required to implement a Gibbs sampler. However, the latent slice sampling algorithm is presented here to sample d_i's, compared with the "independent" slice-efficient sampler. The inferential procedure and algorithms are presented in Section 2.2.1 and Section 2.2.2 for both the latent slice sampler and slice-efficient sampler.

2.2.1 Latent Slice Samplers for the MDP

In fact, we only need to sample a finite set of variables θ and v instead of the entire set at each stage in order to progress to the next iteration. Only the parameters of the "active" clusters for which $\{j : j \leq J = \max(d) + k\}$ are sampled at each iteration. The Gibbs steps and latent slice sampling step for indicator variables are listed below:

i. Starting with $\theta_j = (\mu_j, \lambda_j)$, which are easily derived as

$$p(\mu_j \mid \dots) = \mathcal{N}\left(\frac{\lambda_j \sum_{i:d_i=j} x_i}{\lambda_j n_j + s}, \frac{1}{\lambda_j n_j + s}\right)$$
$$p(\lambda_j \mid \dots) = \text{Gamma}\left(\tau + \frac{n_j}{2}, \tau + \frac{\sum_{i:d_i=j}(x_i - \mu_j)^2}{2}\right)$$

where $\lambda_j = 1/\sigma_j^2$, and $n_j = \sum_{i=1}^{n} \mathbf{1}(d_i = j)$ denotes the number of observations within a given cluster s.t. $n = \sum_{j=1}^{J} n_j$. If there are no d_i equal to j, then $p(\mu_j \mid \dots) = P_0(\mu_j) = \mathcal{N}(0, 1/s)$ and $p(\lambda_j \mid \dots) = P_0(\lambda_j) = \text{Gamma}(\tau, \tau)$.

ii. About the sampling of the v_j's, we have

$$p(v_j \mid \dots) = \text{Beta}\left(1 + n_j, \alpha + \sum_{i:d_i=j} \mathbf{1}(d_i > j)\right)$$

iii. Lastly, we will use the pre-defined transition density instead of posterior to sample the indicator variables

$$p_k(d_i = j \mid d_c, \dots) = \frac{\pi(j)}{k} \sum_{a=\max(j,d_c)}^{\min(j+k-1, d_c+k-1)} \frac{1}{\sum_{b=\max(1,a-k+1)}^{a} \pi(b)}$$

with $\pi(j) = w_j \mathcal{N}(x_i \mid \mu_j, \sigma_j^2)$. Here, d_c is the currect value of d_i.

The infinite dimensional problem automatically converts to a finite one with the latent slice sampling. Also, the conditional distribution of $p(d_i = j \mid \dots) = \pi(j)$ is directly plugged into the transition kernel without introducing auxiliary variables. For the purpose of comparison, Section 2.2.2 describes the basic idea of slice sampling for the MDP.

2.2.2 Slice-efficient Sampler for the MDP

A slice sampler [151] is employed to make finite the number of objects to be sampled within each iteration of a Gibbs sampler, in order to handle countably infinite numbers of values in a Dirichlet process mixture model. An auxiliary variable $u_i > 0$ is introduced, for each observation i, which preserves the marginal distribution of the data x_i and facilitates writing the conditional density of $x_i \mid u_i$ as a finite mixture model. u_i has the effect of truncating the number of components required to be sampled adaptively. Denoting by $\xi = \{\xi_1, \xi_2, \xi_3, \dots\}$ a decreasing sequence of infinite quantities which sum to 1, the joint distribution of (x_i, u_i) is given by

$$f(x_i, u_i \mid \theta, \xi) = \sum_{j=1}^{\infty} w_j \mathrm{Unif}(u_i \mid 0, \xi_j) f(x_i \mid \theta_j) \tag{2.5}$$

with $f(x_i \mid \theta) = \sum_{j=1}^{\infty} w_j f(x_i \mid \theta_j)$ and $f(u_i \mid \xi) = \sum_{j=1}^{\infty} w_j \mathbf{1}(u_i < \xi_j)/\xi_j$. Clearly, integrating out u_i in (2.5) with respect to the Lebesgue measure returns the desired density $f(x_i \mid \theta)$. With probability ξ_j, x_i and u_i are independent, and are, respectively normal and uniform distributed. Since only a finite number of ξ_j are greater than u_i, by denoting $\mathcal{A}_\xi(u_i) = \{j : u_i < \xi_j\}$, the conditional density of $x_i \mid u_i$ can be written as a finite mixture model

$$f(x_i \mid u_i, \theta) = \frac{f(x_i, u_i \mid \theta, \xi)}{f(u_i \mid \xi)} = \sum_{j \in \mathcal{A}_\xi(u_i)} \frac{w_j}{\xi_j f(u_i \mid \xi)} f(x_i \mid \theta_j) \tag{2.6}$$

Typical implementations of the slice sampler arise when $\xi_j = w_j$ [151] but "independent" slice-efficient sampling [89] allows for a deterministic decreasing sequence. The mixing depends on the rate at which the ratio $r_j = \mathbb{E}(w_j)/\xi_j$

increases with j. Faster rates of increase are associated with better mixing but longer running times since the average size of $\mathcal{A}_\xi(u_i)$ increases.

The Bayesian inference here proceeds via a sampler with geometric decay given by $\xi_j = (1-\rho)\rho^{j-1}$, where $\rho \in (0,1]$ is a fixed value determining chain mixing and running time. In general, the higher the value, the better the mixing but with longer running times, as the cardinality of $\mathcal{A}_\xi(u_i)$ increases. Setting $\rho = 0.75$ appears to strike an appropriate balance in the MDP applications here. With the stick-breaking prior and independent slice-efficient sampler, mixture components and their corresponding parameters are recorded at each iteration such that the mixing proportions from a decreasing sequence, as the stick-breaking prior is not invariant to the order of cluster lables [77]).

After introducing the indicator latent variable d_i and denoting $u = (u_1, u_2, \dots, u_n)$, the complete data likelihood is

$$l(x, u, d \mid w, \mu, \sigma^2) = \prod_{i=1}^{n} \frac{w_{d_i}}{\xi_{d_i}} \mathbf{1}(u_i < \xi_{d_i}) \mathcal{N}(x_i \mid \mu_{d_i}, \sigma^2_{d_i}), \tag{2.7}$$

If ξ and v are conditionally independent, the slice Gibbs sampler is then

i. $\theta_j = (\mu_j, \lambda_j)$ and v_j have identical posteriors form as in latent slice sampling algorithm.

ii. The sampling of the indicator variables is given by

$$P(d_i = k \mid \dots) \propto \mathbf{1}(k : \xi_k > u_i) w_k / \xi_k \mathcal{N}(x_i; \mu_k, \sigma^2_k)$$

iii. u_i's are easy to find and are uniformly distributed

$$f(u_i \mid \dots) = \text{Uniform}(u_i \mid 0, \xi_{d_i})$$

This naturally defines a blocking scheme for u and v which are conditionally independent. We simply need to sample up to the integer M for which we have found all the appropriate k in order to do the sampling of d's exactly. In fact it is easy to find the set of (k) required since it will be of the kind $\{1, \ldots, M\}$ where $M = \max_i\{M_i\}$ and M_i is the largest integer of l for which $\xi_l > u_i$. This can often be found analytically for suitable choice of ξ_l:

$$\xi_l = (1-\rho)\rho^{l-1} > u_i \Longrightarrow l \leq 1 + \lfloor \frac{\log u_i/(1-\rho)}{\log \rho} \rfloor = M_i$$

If we take $\xi_j = w_j$, as in [123], it is sufficient to find an M_i such that $\sum_{k=1}^{M_i} w_k > 1 - u_i$, then it is not possible for any w_k, for $k > M_i$, to be greater than u_i. This search is more cumbersome since it can be only checked by simulation. The slice sampling approach uses a slice variable to make the choice of d_i finite at each iteration of a Gibbs, whereas the latent slice sampler not only automatically involves finite set variables but also requires no auxiliary variables.

Besides the application on the infinite dimensional problems like mixture of Dirichlet process, the latent slice sampler can also be flexibly applied in model determination problems with the reversible jump MCMC sampler. The traditional reversible Markov chain sampler is constructed with a Metropolis step that can jump between parameter subspaces of differing dimensionality, while the new framework of reversible jump MCMC is constructed with the latent slice sampler.

2.3 Reversible Jump MCMC

Statistical problems where “the number of things you don’t know is one of the things you don’t know” are ubiquitous in statistical modelling. They arise both in traditional modelling situations such as variable selection in regression, and in more novel methodologies such as object recognition, signal processing, and Bayesian nonparametrics. All such “trans-dimensional” problems can be formulated generically, sometimes with a little ingenuity, as a matter of joint inference about a model indicator j and a parameter vector θ_j, where the model indicator determines the dimension n_j of the parameter, but this dimension varies from model to model.

Inference about these two kinds of unknown is based on different logical principles, but the Bayes paradigm offers the opportunity of a single logical framework – it is the joint posterior $\pi(j, \theta_j \mid x)$ of model indicator and parameter given data x that is the basis for inference. Reversible jump Markov chain Monte Carlo [70] is a method for computing this posterior distribution by simulating from a Markov chain whose state is a vector with unfixed dimension.

The joint inference problem can be set naturally in the form of a simple Bayesian hierarchical model. We suppose that a prior $p(j)$ is specified over models j in a countable set $\mathcal{J}$, and for each j we are given a prior distribution $p(\theta_j \mid j)$, along with a likelihood $\mathcal{L}(x \mid j, \theta_j)$ for the observed data x. We have a countable collection of candidate models $\mathcal{M} = \{\mathcal{M}_1, \mathcal{M}_2, \dots\}$ indexed by a parameter $j \in \mathcal{J}$.

In some settings, $p(j)$ and $p(\theta_j \mid j)$ are not separately available, even up to multiplicative constants; this applies for example in many point process models. However it will be clear that what follows requires specification only of the product $p(j, \theta_j) = p(j) \times p(\theta_j \mid j)$ of these factors, up to a multiplicative constant. In many models there are discrete unknowns as well as continuously distributed ones. Such unknowns, whether fixed or variable in number, cause no additional difficulties; only discrete-state Markov chain notions are needed to handle them, and formally speaking, the variable j can be augmented to include these variables; such problems then fit into the above framework.

The joint posterior distribution of (j, θ_j) given observed data x is obtained as usual via complete likelihood, $\mathcal{L}(x \mid j, \theta_j)$, and the joint prior, $p(j, \theta_j) = p(\theta_j \mid j)p(j)$, constructed from the prior distribution of θ_j under model $\mathcal{M}_j$, and the prior for the model indicator j (i.e. the prior for model $\mathcal{M}_j$). Hence, the joint posterior is

$$p(j, \theta_j \mid x) = \frac{\mathcal{L}(x \mid j, \theta_j)p(\theta_j \mid j)p(j)}{\sum_{j' \in \mathcal{J}} \int_{\mathcal{R}^{n_{j'}}} \mathcal{L}(x \mid j', \theta_{j'})p(\theta_{j'} \mid j')p(j')d\theta_{j'}} \tag{2.8}$$

can always be factorized as

$$\pi(j, \theta_j \mid x) = \pi(j \mid x)\pi(\theta_j \mid j, x)$$

that is as the product of posterior model probabilities and model-specific parameter posteriors. This identity is very often the basis for reporting the inference, and in some of the methods mentioned below is also the basis for computation.

The reversible jump algorithm uses the joint posterior distribution in Equation (2.8) as the target of a Markov chain Monte Carlo sampler over the state space $\Theta = \bigcup_{j \in \mathcal{J}} (\{j\} \times \mathcal{R}^{n_j})$, where the states of the Markov chain are of the form (j, θ_j), the dimension of which can vary over the state space.

The basic formulation embraces not only genuine model-choice situations, where the variable j indexes the collection of discrete models under consideration, but also settings where there is really a single model, but one with a variable dimension parameter, for example a functional representation such as a series whose number of terms is not fixed. In the latter case, j is unlikely to be of direct inferential interest, arising sometimes in Bayesian nonparametrics.

2.3.1 From Metropolis-Hastings to Reversible Jump

The standard formulation of the Metropolis-Hastings algorithm [29] relies on the construction of a time-reversible Markov chain via the detailed balance condition. This condition means that moves from state θ to θ' are made with the same probability as moves from θ' to θ with respect to the target density. This is a simple way to ensure that the equilibrium distribution of the chain is the desired target distribution. The extension of the Metropolis-Hasting algorithm to the setting where the dimension of the parameter vector varies is more challenging; however, the resulting algorithm is surprisingly simple to follow.

For the construction of a Markov chain on a general state space Θ with invariant or stationary distribution π, the detailed balance condition can be written as

$$\int_{(\theta,\theta')\in\mathcal{A}\times\mathcal{B}} \pi(d\theta)P(\theta,d\theta') = \int_{(\theta,\theta')\in\mathcal{A}\times\mathcal{B}} \pi(d\theta')P(\theta',d\theta) \tag{2.9}$$

for all Borel sets $\mathcal{A}\times\mathcal{B}\subset\Theta$, where P is a general Markov transition kernel. More simply, when writing with density functions,

$$\pi(\theta)\,p(\theta' \mid \theta) = \pi(\theta')\,p(\theta \mid \theta').$$

As with the standard Metropolis-Hastings algorithm, Markov chain transitions from a current state $\theta = (j,\theta_j)\in\mathcal{A}$ in model $\mathcal{M}_j$ are realized by first proposing a new state $\theta' = (j',\theta_{j'})\in\mathcal{B}$ in model $\mathcal{M}_{j'}$ from a proposal distribution $q(\theta,\theta')$. The detailed balance condition (2.9) is enforced through the acceptance probability, where the move to the candidate state θ' is accepted with probability $\alpha(\theta,\theta')$. If rejected, the chain remains at the current state θ in model $\mathcal{M}_j$. Under this mechanism, Equation (2.9) becomes

$$\int_{(\theta,\theta')\in\mathcal{A}\times\mathcal{B}} \pi(\theta \mid x)q(\theta,\theta')\alpha(\theta,\theta')d\theta d\theta' = \int_{(\theta,\theta')\in\mathcal{A}\times\mathcal{B}} \pi(\theta' \mid x)q(\theta',\theta)\alpha(\theta',\theta)d\theta d\theta' \tag{2.10}$$

where the distribution $\pi(\theta \mid x)$ and $\pi(\theta' \mid x)$ are posterior distributions with respect to model $\mathcal{M}_j$ and $\mathcal{M}_{j'}$, respectively. One way to enforce Equation (2.10) is by setting the acceptance probability as

$$\alpha(\theta,\theta') = \min\left\{1, \frac{\pi(\theta \mid x)q(\theta,\theta')}{\pi(\theta' \mid x)q(\theta',\theta)}\right\}, \tag{2.11}$$

where $\alpha(\theta, \theta')$ is similarly defined in [29]. It is straightforward to observe that this formulation includes the standard Metropolis-Hastings algorithm as a special case.

Accordingly, a reversible jump sampler with N iterations is commonly constructed as:

i. Initialize j and θ_j at iteration $t = 1$.

ii. For iteration $t \geq 1$ perform

   - Within-model move: with a fixed model j, update the parameters θ_j according to any MCMC updating scheme.
   - Between-models move: simultaneously update model indicator j and the parameters θ_j according to the general reversible proposal/acceptance mechanism in Equation (2.11).

iii. Increment iteration $t = t + 1$. If $t < N$, go to Step ii.

In practice, the construction of proposal moves between different models is achieved via the concept of "dimension matching", as shown by Green [70]. The final form of the acceptance probability combine with the joint posterior expression of Equation (2.8) is

$$\alpha[(j, \theta_j), (j', \theta'_{j'})] = \min\left\{1, \frac{\pi(j', \theta'_{j'} \mid x) q(j' \to j) q_{d_{j' \to j}}(u')}{\pi(j, \theta_j \mid x) q(j \to j') q_{d_{j \to j'}}(u)} \left| \frac{\partial g_{j \to j'}(\theta_j, u)}{\partial(\theta_j, u)} \right| \right\} \tag{2.12}$$

where u is a random vector of length $d_{j \to j'} = n_{j'} - n_j$ generate from a known density $q_{j \to j'}(u)$. The current state θ_j and the random vector u are then mapped to the new state $\theta'_{j'} = g_{j \to j'}(\theta_j, u)$ through a one-to-one mapping function $g_{j \to j'} : \mathbb{R}^{n_j} \times \mathbb{R}^{d_j} \to \mathbb{R}^{j'}$.

2.3.2 The Composite Representation

A composite model space [68] is defined for standard model selection problems in which no parameters are considered as "shared" between any two models. It is later modified to introduce more flexibility in shared parameters problems such as nested models and model selection. The composite model is a straightforward modification of that used by Carlin and Chib [22]. The full posterior distribution for the composite model space is

$$p(j, \theta \mid x) = \frac{p(x \mid j, \theta_j) p(\theta_j \mid j) p(\theta_{-j} \mid \theta_j, j) p(j)}{p(x)}$$

where θ_{-j} denotes the parameters *not* used by the model j. All of the terms in the above expression are defined explicitly by the chosen likelihood and prior structures except for $p(\theta_{-j} \mid \theta_j, j)$, the "prior" for the parameters in the composite model which are not used by model j. It is easily seen that any proper distribution can be assigned arbitrarily to these parameters without affecting the required marginals for the remaining parameters. In many cases it will be convenient to assume that the unused parameters are a priori independent of one another and also of θ_j. In this case, we have that $p(\theta_{-j} \mid \theta_j, j) = p(\theta_{-j} \mid$

$j) = \prod_{\kappa \neq j} p(\theta_\kappa \mid j)$ and the composite model posterior can be rewritten as

$$p(j, \theta \mid x) = \frac{p(x \mid j, \theta_j) p(\theta_j \mid j) \left(\prod_{\kappa \neq j} p(\theta_\kappa \mid j) \right) p(j)}{p(x)} \tag{2.13}$$

This is the form of composite space used by Carlin and Chib [22]. The priors on the unused parameters θ_{-j} are referred as "pseudo-priors" or linking densities in the Carlin and Chib model, appropriate choice of which is crucial to the effective operation of their algorithm.

The key feature of the composite model space is that the dimension remain fixed even when the model number j changes. This means that standard MCMC procedures, under the usual convergence conditions, can be applied to the problem of model uncertainty. For example, a straightforward Gibbs sampler applied to the composite model leads to Carlin and Chib's method, while a more sophisticated Metropolis-Hastings approach leads to reversible jump.

The sampling algorithm of Carlin and Chib [22] is easily obtained from the composite model by applying a Gibbs sampler to the individual parameters θ_j and to the model index j. The sampling steps, which may be performed a random or deterministic scan, are as follows:

$$\begin{gathered} \theta_\kappa \sim p(\theta_\kappa \mid \theta_{-\kappa}, j, x) \propto \begin{cases} p(x \mid j, \theta_j) p(\theta_j) & \kappa = j \\ p(\theta_\kappa \mid \theta_{-\kappa}, j) & \kappa \neq j \end{cases} \\ j \sim p(j \mid \theta, x) \propto p(x \mid j, \theta_j) p(\theta_j \mid j) p(\theta_{-j} \mid \theta_j, j) p(j) \end{gathered} \tag{2.14}$$

The reversible jump sampler achieved model space moves by Metropolis-Hastings proposals with an acceptance probability that is designed to preserve detailed balance within each move type. Suppose that we propose a move to model j' with parameters $\theta_{j'}$ from model j with parameters θ_j using a proposal distribution $q(j', \theta_{j'} \mid j, \theta_j)$. The acceptance probability in order to preserve detailed balance is given by

$$\alpha = \min\left\{1, \frac{p(j', \theta_{j'} \mid x)q(j, \theta_j \mid j', \theta_{j'})}{p(j, \theta_j \mid x)q(j', \theta_{j'} \mid j, \theta_j)}\right\} \tag{2.15}$$

In implementation it will often be convenient to take advantage of any nested structure in the models or interrelationships between the parameters of different models in constructing effective proposal distributions, rather than proposing the entire new parameter vector as in Equation (2.15). Generally, relationships between parameters of different models can be used to good effect by drawing “dimension matching” variables u and u' from proposal distributions $q_2(u)$ and $q_2(u')$, and then forming θ_j' and θ_j as deterministic functions of the form $\theta_j = g(\theta_{j'}, u)$ and $\theta_{j'} = g(\theta_j, u')$. In this way it is straightforward to incorporate useful information from the current parameter vector θ_j into the proposal for the new parameter vector $\theta_{j'}$. Provided that $\dim(\theta_{j'}, u) = \dim(\theta_j, u')$ (dimension matching), the acceptance probability is given by Green [70]:

$$\alpha = \min\left\{1, \frac{p(j', \theta_{j'} \mid x)q_1(j \mid j')q_2(u)}{p(j, \theta_j \mid x)q_1(j' \mid j)q_2(u')}\left|\frac{\partial(\theta_{j'}, u)}{\partial(\theta_j, u')}\right|\right\} \tag{2.16}$$

which now includes a Jacobian term to account for the change of measure between (θ_j, u') and $(\theta_{j'}, u)$. Note the basic form of reversible jump given above in Equation (2.15) is obtained from this formula when we set $\theta_j = g(\theta_{j'}, u) = u$ and $\theta_{j'} = g(\theta_j, u') = u'$, so that the Jacobian term is unity.

We now show that Green's reversible jump sampler can be obtained by applying a special form of Metropolis–Hastings proposal to the composite model space. Consider a proposal from the current state of the composite model (j, θ) to a new state $(j', \theta_{j'})$ that takes the form

$$q(j', \theta_{\prime} \mid j, \theta) = q_1(j' \mid j) q_2(\theta'_{j'} \mid \theta_j) p(\theta'_{-j'} \mid \theta'_{j'}, j')$$

This proposal, which forms a joint distribution over all elements of j and θ, is split into three component parts: the model index component $q_1(j' \mid j)$, which proposes a move to a new model index, j'; a proposal for the parameters used by model j', $q_2(\theta'_{j'} \mid \theta_j)$; and a proposal for the remaining unused parameters which is chosen to equal to the pseudo-prior $p(\theta'_{-j'} \mid \theta'_{j'}, j')$. We thus have a joint proposal across the whole state space of parameters and model index that satisfies the Markov requirement of the Metropolis-Hastings method as it depends only upon the current state (j, θ) to make the joint proposal (j', θ'). There are now no concerns about a parameter space with variable dimension since the composite model retains constant dimensionality whatever the value of j and any issues of convergence can be addressed by reference to standard Metropolis–Hastings results in the composite space.

The acceptance probability for this special form of proposal is given, using the standard Metropolis-Hastings procedure, by

$$\begin{aligned}\alpha &= \min\left\{1, \frac{q(j,\theta \mid j',\theta')p(j',\theta' \mid x)}{q(j',\theta' \mid j,\theta)p(j,\theta \mid x)}\right\}\\ &= \min\left\{1, \frac{q_1(j \mid j')q_2(\theta_j \mid \theta'_{j'})p(\theta_{-j} \mid \theta_j, j)p(j',\theta'_{j'} \mid x)p(\theta'_{-j'} \mid \theta'_{j'}, j)}{q_1(j' \mid j)q_2(\theta'_{j'} \mid \theta_j)p(\theta'_{-j'} \mid \theta'_{j'}, j')p(j,\theta_j \mid x)p(\theta_{-j} \mid \theta_j, j)}\right\}\\ &= \min\left\{1, \frac{q_1(j \mid j')q_2(\theta_j \mid \theta'_{j'})p(j',\theta'_{j'} \mid x)}{q_1(j' \mid j)q_2(\theta'_{j'} \mid \theta_j)p(j,\theta_j \mid x)}\right\}\end{aligned}$$

This last line is exactly the acceptance probability for the basic reversible jump sampler with the proposal distribution factored into two components $q_1(\cdot)$ and $q_2(\cdot)$. We see that the acceptance probability is independent of the value of any parameters which are unused by both models j and j' ; nor are their values required for generating a proposal at the next iteration. Hence the sampling of these is a "conceptual" step only which need not be performed in practice.This feature is a strong point of the reversible jump method compared with the Gibbs sampling version of the Carlin and Chib method, which requires samples for all parameters at each iteration. Conversely, it is a very challenging problem to construct effective proposal distributions for reversible jump methods in complex modeling scenarios, especially in cases where there is no obvious nested structure to the models or other interrelationships between the parameters of the different models; in these cases the Carlin and Chib method, which allows blocking of the parameters within a single model in a way that is not possible for reversible jump, may have the advantage. It is interesting, however, to see that both schemes can be derived as special cases of the composite space sampler.

Convergence properties of the reversible jump scheme derived in the special way given here can now be inherited directly from the Metropolis–Hastings algorithm operating on the fixed dimension composite space. Specifically, irreducibility and aperiodicity of the composite space sampler will ensure the convergence of the chain to the target distribution and the validity of ergodic averages [134].

Statistical problems in which the number of models is itself unknown are extensive, and as such the reversible jump sampler has been implemented in analyses throughout a wide range of scientific disciplines over the last number of years. Within the statistical literature, these predominantly concern Bayesian model determination problems, including change-point models, mixtures with an unknown number of components [130], variable selection, Bayesian nonparametrics, time series model, and so on.

Below describes the example of finite mixture of mixture of exponential densities, as well as the inference procedure by using the reversible jump MCMC sampler and latent slice sampler.

2.3.3 Mixture of Exponentials and Reversible Jump MCMC

The mixture of exponential only has one parameter, which is the weight of each component. Define the mixture of exponential density as

$$f(x_i \mid w_M, M) = \sum_{j=1}^{M} w_{jM} j e^{-jx_i}$$

with M unknowns (taking integers from 1 to ∞)and $w_M = (w_{1M}, \ldots, w_{MM})$, then the joint distribution of (x, d, w_M, M) is

$$f(x, d, w_M, M) = f(w_M \mid M) f(M) \prod_{i=1}^{n} w_{d_i} d_i e^{-d_i x_i},$$

and the priors for w_M and M are

$$f(M) = \frac{\lambda^{M-1} e^{-\lambda}}{(M-1)!}, \quad M = 1, 2, \ldots$$

$$w_M \mid M \sim \text{Dirichlet}(\alpha, \alpha, \ldots, \alpha)$$

Given current M' and the initialized indicator variable d' is a sample from 1 to M' with replacement. The sampling procedure is

i. Initialized w: sample $w \mid x, d', M' \sim \text{Dirichlet}(n_1 + 1, n_2 + 1, \ldots, n_{M'} + 1)$.

ii. Sample $d_i = j \mid x, w, M' \sim \text{Categorical}(p_1, p_2, \ldots, p_{M'})$, where

$$p_j = \frac{w_j j e^{-j x_i}}{\sum_{j=1}^{M'} w_j j e^{-j x_i}}, \quad w \text{ is from Step i.}$$

iii Sample $w \mid x, d, M' \sim \text{Dirichlet}(n_1 + 1, n_2 + 1, \ldots, n_{M'} + 1)$ again with the new simulated d_i's.

iv. Sample $M \mid M'$, where $M \in \{m_1, m_1 + 1, \ldots, m_2 - 1, m_2\}$ with $m_1 = \max(1, M' - k + 1)$ and $m_2 = M' + k - 1$ by using latent slice sampling method.

$$p_k(M \mid M') = \frac{\pi(M)}{k} \sum_{l=\max(M,M')}^{\min(M+k-1, M'+k-1)} \frac{1}{\sum_{h=\max(1,l-k+1)}^{l} \pi(h)} \tag{2.17}$$

$$\pi(M) = f(M)f(w_M \mid M)\prod_{i=1}^{n}\sum_{j=1}^{M} w_j j e^{-jx_i} \cdot p(w_{m_1} \mid w_{m_1+1})$$
$$\dots p(w_{M-1} \mid w_M) \cdot p(w_{M+1} \mid w_M) \dots p(w_{m_2} \mid w_{m_2-1}) \quad (2.18)$$

The product of all the $p(\cdot \mid \cdot)$ in (2.18) is equal to a constant $\frac{(m_1-1)!}{(m_2-1)!}$, as derived in Appendix A, irrespective of the value M. Therefore, the updated $\pi(M)$ is

$$\pi(M) = f(M)f(w_M \mid M)\prod_{i=1}^{n}\sum_{j=1}^{M} w_j j e^{-jx_i} \cdot \frac{(m_1-1)!}{(m_2-1)!} \quad (2.19)$$

where the constant $\frac{(m_1-1)!}{(m_2-1)!}$ will be canceled out when plug in $p_k(M \mid M')$.

Or, we can use the Metropolis step by using random walk as the proposal, with the first three steps identical to latent slice sampling method:

i. When $M > 1$, $q(M-1 \mid M) = 0.5$ and $q(M+1 \mid M) = 0.5$; when $M = 1$, $q(M+1 \mid M) = q(2 \mid 1) = 1$

ii. With M' taking values of $M+1$ and $M-1$, the acceptance probability is

$$\alpha(M, M') = \min\left\{1, \frac{f(M')f(x_1, \dots, x_n \mid w_{M'}, M')p(w_M \mid w_{M'})q(M \mid M')}{f(M)f(x_1, \dots, x_n \mid w_M, M)p(w_{M'} \mid w_M)q(M' \mid M)}\right\}$$

Note that $p(\cdot \mid \cdot)$ can be anything, but we try to give a good proposal as we did in the latent slice method: (a). if $M' = M+1$, $p(w_{M+1} \mid w_M) = 1/M$; (b). if $M' = M-1$, $p(w_{M-1} \mid w_M) = 1/(M-1)$, as described in Section A.2 of Appendix A.

2.4 Experiments

For density estimation we would like to sample from the predictive distribution of

$$f(x_{n+1} \mid x_1, \ldots, x_n)$$

At each iteration, we have (w_j, μ_j, σ_j^2) and we sample a $\theta_j = (\mu_j, \sigma_j^2)$ using the weights. The idea is to sample a uniform random variable r from the unit interval and to take that θ_j for which $w_{j-1} < r < w_j$, with $w_0 = 0$. If more weights are required than currently exist then it is straightforward to sample more as we know the additional v_j's are independent and identically distributed from $\text{Beta}(1, \alpha)$ and the additional θ_j's are independent and identically distributed from P_0. Having taken θ_j , we draw x_{n+1} from $\mathcal{N}(\cdot \mid \theta_j)$.

2.4.1 Experiments on the MDP

Here we present a normal example with non-informative specifications. 400 random variables was sampled independently from $f(x) = \frac{1}{3}\mathcal{N}(x \mid -4, 1) + \frac{1}{3}\mathcal{N}(x \mid 0, 1) + \frac{1}{3}\mathcal{N}(x \mid 8, 1)$. We took $\tau = 0.5$, $s = 1$, $\alpha = 2$, $\rho = 0.75$ and the Gibbs sampler was run for 20,000 iterations and at each iteration from 15,000 onwards a predictive sample y_{n+1} was taken.

Fig. 2.3 shows the histogram of the 400 data points with the density estimators (blue: latent slice, red: slice sampling) based on the 5000 samples of y_{n+1} vs. the true density (black). The density estimators was obtained using the R density routine. It is obvious that the estimators and the true density are close to each other. Both predictive densities are almost overlapped.

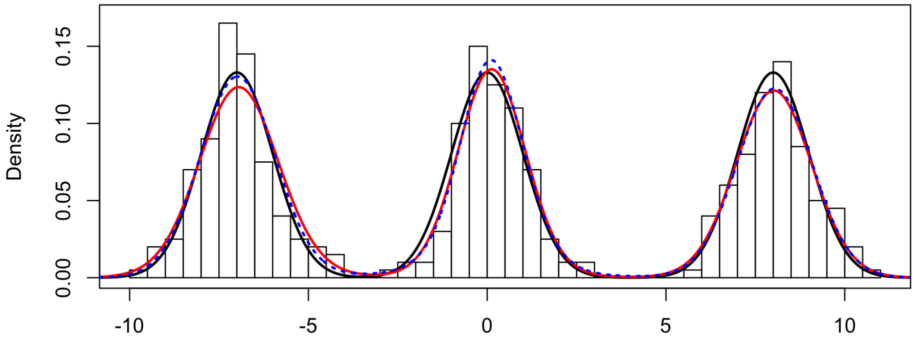


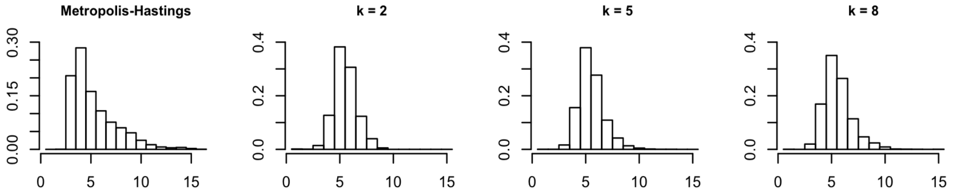
Figure 2.3: Histogram of data and the estimated densities of latent slice sampler (blue) and slice sampler (red) vs. the true density (black).

The advantage of using latent slice step is remarkable with respect to good mixing and predictive density. It also requires no auxiliary variables to make the problem a finite one, which is easy to implement.

2.4.2 Experiments on the MFM

Fig. 2.4 shows the histogram of the sampled M for both the Metropolis method and the latent slice sampling method with setting $\lambda = 3$, $\alpha = 1$, and varying $k = 2, 5, 8$. The number of components for the exponential mixtures is concentrated around 4 while for latent slice it is concentrated around 5 with all different of k's. There is no huge difference between different k's, and we choose $k = 5$ to strike a balance and compare the estimated densities with the

Metropolis approach. In Fig. 2.5, the estimated densities of both methods are close to the true density. To determine which estimator is better, certain measure like Kullback-Leibler divergence need to be inlvolved.



Metropolis-Hastings

k = 2

k = 5

k = 8

Figure 2.4: Sampled M from Metropolis step and latent slice step.

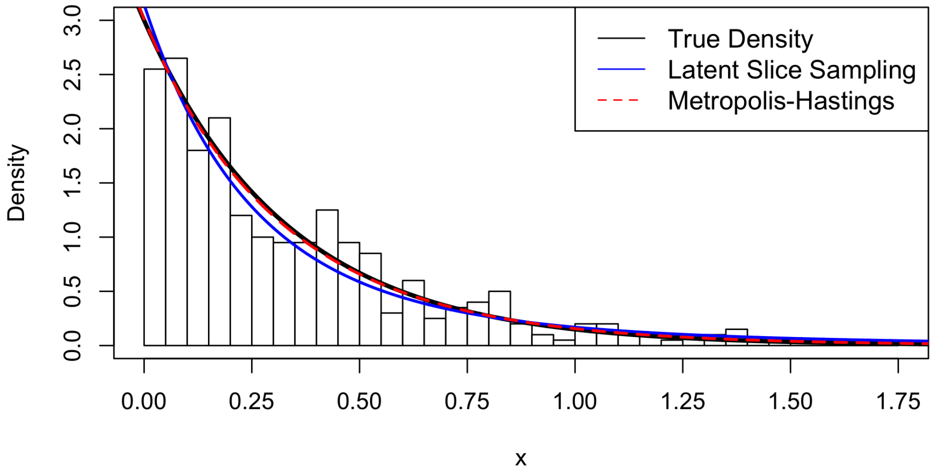


Figure 2.5: Histogram of the data and density estimations from both Metropolis step and latent slice sampler compared with the true density.

2.4.3 Experiments on Change-point Problem

Here we present Bayesian models for multiple change-point analysis in Green's paper [70], and develop a reversible jump Markov chain Monte Carlo sampler latent slice sampling to compute the posterior distribution.

Definition. *A point process N is called a* ***Poisson Process*** *with intensity function λ, if*

i. $N(t)$ has independent increments;

ii. $N(b) - N(a)$ is Poisson$(\int_a^b \lambda(t)dt)$ -distributed.

The data set is the point process of dates of coal mining disasters, which is given in "coal" command in one R package called "boot". The "coal" data frame gives the dates of 191 explosions in coal mines from March 15, 1851 until March 22, 1962. The integer part of the date gives the year, while the day is represented as the fraction of the year that has elapsed on that day. Fig. 2.6 displays the dates of the 191 disasters in these 112 years as a dot plot, together with the cumulative counting process, shown as a dashed line.

We assume that the rate function to be a step function $\lambda : \mathbb{R} \to \mathbb{R}^+$ on $[s_1, s_{k+1}]$, where $s_1, s_2, \ldots, s_{k+1}$ is ordered by year. We fixed s_1 to the beginning of 1851 and s_{k+1} to the end of 1962. Moreover, h_j, the jth piece of the step function $\lambda(\cdot)$, is a constant function on $[s_j, s_{j+1})$. In our model, λ is

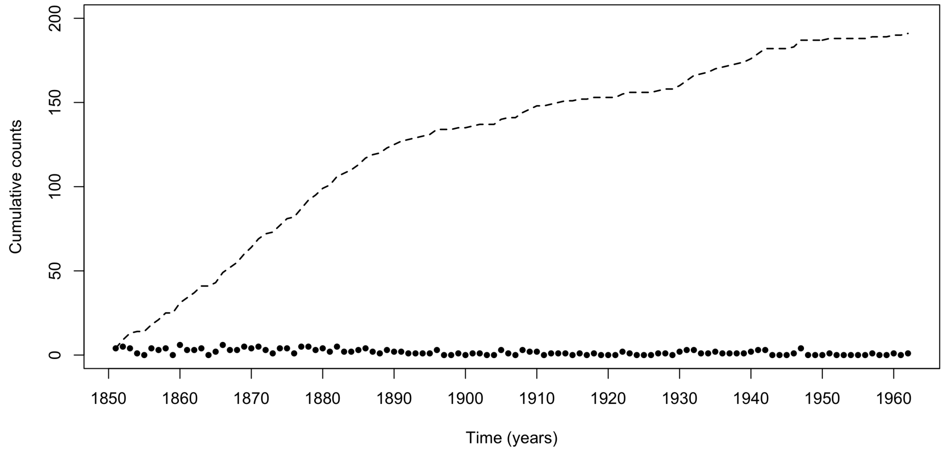


Figure 2.6: Coal mining disaster data: dates of disasters, cumulative counting process (dashed).

the intensity function of a Poisson process $\{y_i, i = 1, 2, \ldots, n\}$. The probability of m disasters occurring in any interval $[a, b] \subseteq [s_1, s_{k+1}]$ is

$$P(\text{the number of } y_i \in [a, b] = m) = e^{-\Lambda_{a,b}} \frac{\Lambda_{a,b}^m}{m!}, \text{ where } \Lambda_{a,b} = \int_a^b \lambda(s) ds$$

Conditioned on m points being present in the interval $[a, b]$, the individual points are independently distributed with density $\lambda/\Lambda_{a,b}$. Therefore, we can compute the likelihood of the data given λ, as

$$p(y_i \in [a, b] \mid \lambda) = m! \cdot e^{-\Lambda_{a,b}} \frac{\Lambda_{a,b}^m}{m!} \cdot \prod_{i=1}^{m} \frac{\lambda(y_i)}{\Lambda_{a,b}}$$

where the factor $m!$ is the number of arrangements when m points are ordered.

In the interval $[s_1, s_{k+1}]$, there are n disasters in total. Thus we have

$$\begin{aligned}
p(y_1, y_2, \ldots, y_n \mid \lambda) &= n! \cdot e^{-\Lambda} \frac{\Lambda^n}{n!} \cdot \prod_{i=1}^{n} \frac{\lambda(y_i)}{\Lambda}, \qquad \text{where } \Lambda = \int_{s_1}^{s_{k+1}} \lambda(s) ds \\
&= e^{-\Lambda} \prod_{i=1}^{n} \lambda(y_i) \\
&= \exp\left(-\Lambda + \sum_{i=1}^{n} \log(\lambda(y_i)) \right) \\
&= \exp\left(-\Lambda + \sum_{j=1}^{k} (\text{number of } y_i \in [s_j, s_{j+1})) \cdot \log(\lambda(y_i)) \right)
\end{aligned}$$

In our model, $\lambda(y_i) = h_j$ when $y_i \in [s_j, s_{j+1})$, thus $\Lambda = \int_{s_1}^{s_{k+1}} \lambda(s) ds = \sum_{j=1}^{k} h_j (s_{j+1} - s_j)$. We denote m_j as the number of disasters occurring between $[s_j, s_{j+1})$, then the explicit form of the likelihood is

$$p(y_1, y_2, \ldots, y_n \mid \lambda) = \exp\left(-\sum_{j=1}^{k} h_j (s_{j+1} - s_j) + \sum_{j=1}^{k} m_j \log h_j \right) \tag{2.20}$$

There are three kinds of random variables in the above model (2.20):

i. The number of steps k:

The corresponding number of change points is $k - 1$ given the number of steps K. k is assumed to be drawn from Poisson distribution.

ii. The heights $\{h_j : j = 1, 2, \ldots, k\}$:

The heights $h_1, h_2, \ldots, h_k$ are independently distributed as $\Gamma(\alpha, \beta)$, where $\alpha \neq 0$ and $\beta \neq 0$.

iii. Step positions $\{s_1, s_2, \ldots, s_{k+1}\}$ with s_1 and s_{k+1} fixed points:

The step positions $s_2, s_3, \ldots, s_k$ are distributed as the even-numbered order statistics from $2k-1$ points $\{t_i, i = 1, 2, \ldots, 2k-1\}$ uniformly distributed on $[s_1, s_{k+1}]$. That is, $s_2 = t_2, s_3 = t_4, \ldots, s_j = t_{2j-2}, \ldots, s_k = t_{2k-2}$.

A reversible jump Monte Carlo sampler is a good way to approach this change-point problem. The four types of transitions from the prior distribution to the posterior distribution are detailed in [70], including a change to the height, a change to the position, "birth" of a new step, and "death" of a randomly chosen step. In section A.3 of Appendix A, each type of move is introduced and individual acceptance probability of the Metropolis updates is calculated, respectively.

We will mainly focus on the latent slice sampler for the coal mining change-point problem. Since there is no reject/accept procedure for the latent slice sampling, we only need to sample k and update corresponding heights $(h_1, \ldots, h_k)$ and positions $(s_1, s_2, \ldots, s_k, s_{k+1})$, where s_1 and s_{k+1} are always fixed no matter what value k takes. From the reversible jump sampler, the posterior distribution for k is

$$\begin{aligned}\pi(k) &\propto p(y_1, y_2, \ldots, y_n \mid k, \theta^{(k)}) p(k, \theta^{(k)}) \\ &= p(y_1, y_2, \ldots, y_n \mid k, \theta^{(k)}) \cdot p(k \mid \lambda) \cdot \prod_{j=1}^{k} p(h_j \mid k, a, b) \cdot p(s_2, \ldots, s_j, \ldots, s_k \mid k) \\ &\propto \exp\left(\sum_{j=1}^{k} [m_j \log h_j - h_j s_{dj}]\right) \cdot \frac{\lambda^{k-1}}{(k-1)!} \cdot \prod_{j=1}^{k} h_j^{\alpha-1} e^{-\beta h_j} \\ &\quad \cdot \frac{(2k-1)!}{(s_{k+1} - s_1)^{2k-1}} \prod_{j=1}^{k} s_{\mathrm{dj}} \cdot \prod_{j=1}^{k-1} p(\theta^{(j)} \mid \theta^{(j+1)}) \cdot \prod_{j=k+1}^{k_{\max}} p(\theta^{(j)} \mid \theta^{(j-1)})\end{aligned}$$

with the difference $s_{dj} = s_{j+1} - s_j$. For the "death" steps, we randomly remove one position except for the first one and last one. The probability is $p(\theta^{(j)} \mid \theta^{(j+1)}) = \frac{1}{j}$. For the "birth" steps, we randomly choose a position i and split the range $[s_i, s_{i+1}]$ into $[s_i, s']$ and $[s', s_{i+1}]$ with $s' = u \cdot (s_{i+1} - s_i) + s_i$, where $u \sim \text{Uniform}(0, 1)$. The probability is then derived as $p(\theta^{(j)} \mid \theta^{(j-1)}) = \frac{1}{j-1}$. The heights of both cases are derived in terms of the new positions and adding appropriate perturbation. Therefore,

$$\prod_{j=1}^{k-1} p(\theta^{(j)} \mid \theta^{(j+1)}) \cdot \prod_{j=k+1}^{k_{\max}} p(\theta^{(j)} \mid \theta^{(j-1)}) = \frac{1}{(k_{\max} - 1)!}$$

which is a constant. The finalized form of the posterior of k is then given by

$$\pi(k) \propto \mathrm{e}^{\sum_{j=1}^{k} (m_j \log h_j - h_j s_{dj})} \cdot \frac{\lambda^{k-1}}{(k-1)!} \cdot \prod_{j=1}^{k} h_j^{\alpha-1} e^{-\beta h_j} \cdot \frac{(2k-1)!}{(s_{k+1} - s_1)^{2k-1}} \prod_{j=1}^{k} s_{dj} \tag{2.21}$$

By using the following discrete latent slice sampler, we can sample from (2.21),

$$p_\tau(k' \mid k) = \frac{\pi(k')}{\tau} \sum_{j=\max(k',k)}^{\min(k'+\tau-1, k+\tau-1)} \frac{1}{\sum_{z=\max(1, j-\tau+1)}^{j} \pi(z)}$$

where $k \in \{1, 2, \dots\}$ and the normalizing constant of $\pi(k)$ is unknown. τ is the parameter of the transition matrix $p_\tau(k' \mid k)$, with $\tau > 1$ and $|k' - k| \leq \tau - 1$. Taking the computation cost and autocorrelations into account, we choose to set $\tau = 6$.

Now we can determine a model with flexible number of steps k. The parameter λ of the prior distribution of k is set to be 3 and the prior distribution of heights h is assumed to be Gamma$(1, 0.5476)$.

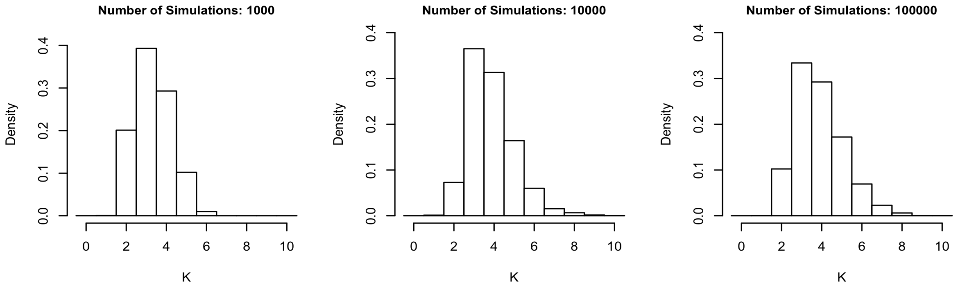


Figure 2.7: Coal mining disaster data: posterior distributions of the number of steps k for 1000, 10000, and 100000 simulations of Markov chain.

The Monte Carlo simulation is run for 1000, 10000, and 100000 updates respectively and Fig. 2.7 shows the corresponding histograms of k. It can be seen that as the estimating steps get larger, the range of k values tends to be wider with maximum equal to 10. When the simulation time is not large enough, like the left histogram of Fig. 2.7, $k = 2$ appears to keep high proportion. However, as the estimating time gets longer, the basic distribution of k almost does not change. $k = 3$ and $k = 4$ always keep their dominance. $k = 5$ also has a proportion greater than 15%.

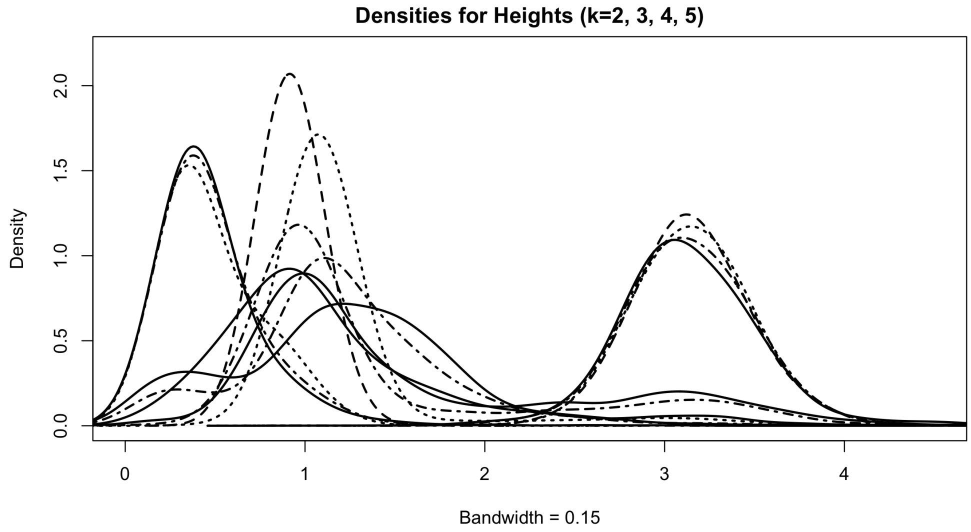


Figure 2.8: Coal mining disaster data: posterior density estimates of heights conditional on number of change-point $k = 2$ (dashed curves), $k = 3$ (dotted curves) and $k = 4$ (solid curves).

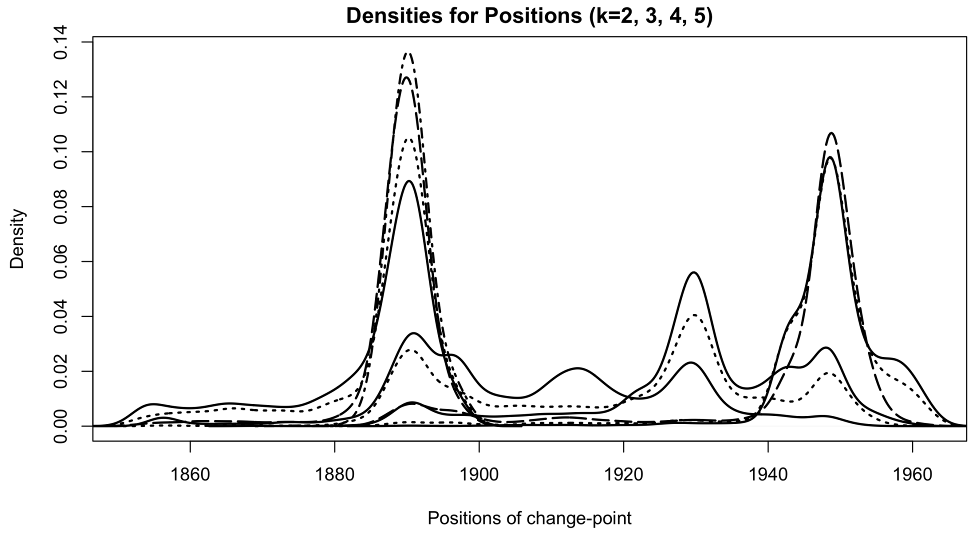


Figure 2.9: Coal mining disaster data: posterior density estimates of positions of change-point, conditional on number of change-point $k = 2$ (dashed curve), $k = 3$ (dotted curves) and $k = 4$ (solid curves).

Figure 2.8 shows the posterior density estimates of the step heights, conditional on values $k = 2, 3, 4$ and 5. The density estimates are obtained using a Gaussian kernel with bandwidth 0.15. Similarly, Figure 2.9 shows the corresponding conditional posterior density estimates of the step positions, using kernel standard deviation 2 years.

Reversible jump has been successfully applied in many contexts, but it is perceived to be difficult to use, and applying it to new situations requires one to design good reversible jump moves, which can be nontrivial, particularly in high-dimensional parameter spaces. With this, discrete latent slice sampler is a better choice for tremendously simplifying the Metropolis steps of the reversible jump sampler and can be used for direct sampling.

2.5 Discussion

In this chapter, we present a latent slice sampler for discrete random variables by designing a reversible transition matrix. The discrete latent slice sampler can be used for direct sampling of implicit form of target distributions of discrete variables. Applications range from Mixture of Dirichlet process, mixture of infinite mixture models, clustering problems, and multiple change-point problems. The advantage of using the discrete latent slice sampler is demonstrated by comparing with the slice-efficient sampler and the nontrivial reversible jump sampler. The latent slice sampler is capable of state-of-the-art performances on a number of problems on discrete random variables.

Chapter 3

Latent Slice Sampler for Continous Variables

The original motivation is to be found in a discrete sampler presented in Walker [152]. Since this provides motivation for the latent part of our slice sampler we briefly describe it here. One of the key ideas behind the Metropolis-Hastings algorithm [112, 78], is the transition density $p(y \mid x)$, defined for all $x, y \in \Omega$, satisfying

$$p(y \mid x)\, \pi(x) = p(x \mid y)\, \pi(y) \tag{3.1}$$

where π is the target density. The Metropolis-Hastings algorithm has transition density $p(y \mid x) = \alpha(x,y)\, q(y \mid x) + (1 - r(x))\, \mathbf{1}(y = x)$, where $q(y \mid x)$ is a proposal density, to be chosen,

$$\alpha(x,y) = \min\left\{1, \frac{\pi(y)\, q(x \mid y)}{\pi(x)\, q(y \mid x)}\right\},$$

and $r(x) = \int \alpha(x,y)\, q(y \mid x)\, dy$. It is easily seen that this $p(\cdot \mid \cdot)$ satisfies (3.1).

The content in this chapter is under review, Yanxin Li and Stephen G. Walker, "A Latent Slice Sampling Algorithm", in *Computational Statistics and Data Analysis* 2021. Dr. Walker proposed the problem and helped brainstorm the evaluation settings. I implemented the main algorithms and wrote the draft paper. Dr. Walker helped with the draft rewriting and revising.

An alternative when the sample space is discrete, say $\Omega = \{0, 1, 2, \ldots\}$, with $p(\cdot \mid \cdot)$ satisfying equation (3.1), is given by

$$p(y \mid x) = \frac{\pi(y)}{k} \sum_{l=\max(y,\,x)}^{\min(y+k-1,\,x+k-1)} \frac{1}{\sum_{z=\,l-k+1}^{l} \pi(z)}, \tag{3.2}$$

where $|y - x| < k$, and k is to be chosen. The choice of k is easy to set; as large as possible while computations required to sample $p(y \mid x)$ remain time feasible. So note that with this transition density there is no possibility for the sampler to get stuck and neither is there an accept/reject component. Note also that π only needs to be known up to a normalizing constant, a strong requirement in any sampler, as often, in many applications, the target density is only specified up to an unknown normalizing constant. Finally, note that (3.2) is easy to sample. A multivariate version of (3.2) is easy to establish and has been applied to a certain class of optimization problem in Ekin et al. [43].

The aim in the present paper was to find a continuous counterpart to (3.2). In fact a suitable transition density is not difficult to write down as a direct analog of (3.2);

$$p(y \mid x) = \frac{\pi(y)}{k} \int_{l=\max(y,\,x)}^{\min(y+k,\,x+k)} \frac{dl}{\int_{z=l-k}^{l} \pi(z)dz}, \tag{3.3}$$

where here we have $\Omega = (-\infty, \infty)$ and $|y - x| \leq k$. Just as (3.2) can be seen as a Gibbs sampler, so can (3.3). To see this, consider the joint density function

$$p(y, l) = \pi(y)\, \frac{\mathbf{1}(y < l < y + k)}{k}, \tag{3.4}$$

so clearly $\pi(y)$ is the required marginal density. Then (3.3) is given by $p(y \mid x) = \int p(y \mid l)\, p(l \mid x)\, dl$, where $p(l \mid x)$ is uniform on the interval $(x, x + k)$.

Further, (3.3) also satisfies the detailed balance equation (3.1). The only outstanding question is how to sample (3.3), which is the main focus of the paper. For this we use the slice sampler, but the existence of the framework already established means we can avoid the doubling or stepping out procedures. This is important as it is these which make the slice sampler prohibitively slow in high dimensional problems.

In chapter 3.2 the aim is to show how to sample from $p(y \mid x)$ given by (3.3) but with necessary extensions involving making k random. This also requires some further latent variable; specifically a “slice” variable, similar in spirit to Besag and Green [14], Damien et al. [35] and Neal [119]. Slice sampling, as it has become known, is a popular approach to sampling complex densities usually within a Gibbs sampling framework. In fact slice samplers have good convergence properties; Roberts and Rosenthal [135] show that slice samplers are nearly always geometrically ergodic while Mira and Tierney [114] provide sufficient conditions for a slice sampler to be uniformly ergodic. On the other hand, [97] show that hybrid samplers, which both our latent slice sampler and Neal's slice sampling algorithm are, do not share the properties of a slice sampler for which the level sets can be sampled directly. However, in most cases it is nigh impossible to find and sample the level sets without resorting to some hybrid idea.

Recent uses of Neal's approach include the elliptical slice sampler, see Murray et al. [116], and the generalized elliptical slice sampler, see Nishihara et al. [120], and factor slice sampling, see Tibbits et al. [148]. Once the slice variable has been incorporated within (3.3), it is then possible to compare the new sampler with Neal's slice sampler. Indeed, as it stands with k fixed, it is precisely a version of Neal's algorithm. Both us and Neal extend from this fixed k, but in different directions. Neal adopts the reversible framework while we adopt a random k approach and use the framework established by the joint density (3.3). This allows us to maintain a Gibbs sampling framework while avoiding a slow (i.e. doubling/stepping out) detailed balance constraint. We make a direct comparison with Neal's slice sampler in chapter 3.3. Numerous illustrations are presented in chapter 3.4 and chapter 3.5 concludes with a brief description and a full layout of the algorithm for an arbitrary multivariate distribution.

We first describe the algorithm in one dimension and later detail the extension to multi-dimensions. The change in notation is that we now write k as s; so the fixed discrete k is now written as a random continuous s. To develop the joint density (3.4), we make it more flexible by allowing k to be a random variable, which we will now refer to as s, and assign s to have density $p(s)$, to be chosen, and allow for l to be in the interval $(y-s/2, y+s/2)$. Hence, the joint density of interest becomes

$$p(y,s,l) = \pi(y)\, p(s)\, \frac{\mathbf{1}\big(y - s/2 < l < y + s/2\big)}{s}. \tag{3.5}$$

The $p(s)$ will be tuned, but just as with the discrete k the general idea is to ensure large s can be sampled from it. The large s allow for the possibility of big jumps; whereas if $p(s)$ only generates small s, the chain is still theoretically correct but the chain will only make small moves.

A key aspect of the innovation in the sampler is on display in equation (3.5); we have introduced a y term outside of the $\pi(y)$ term without altering the correct marginal. So the marginal density of y remains $\pi(y)$ and the marginal density of s is $p(s)$. A Gibbs sampler based directly on (3.5) would be difficult to implement as it is not possible to sample from $\pi(y)$; or rather it is assumed not to be able to do so. In such cases, a slice sampler can be utilized. By introducing a slice variable w, the joint density then becomes

$$p(y, w, s, l) = \mathbf{1}\big(\pi(y) > w\big)\, p(s)\, \frac{\mathbf{1}\big(y - s/2 < l < y + s/2\big)}{s}. \qquad (3.6)$$

While this is more than used by Neal [119], the extra component, i.e. $\mathbf{1}\big(y - s/2 < l < y + s/2\big)\, p(s)/s$ is effectively providing the stochastic search engine for the set of y for which $\pi(y) > w$. Such a procedure was also required by Neal [119] who used a search strategy while needing also to maintain a detailed balance criterion. On the other hand, we are free from some such constraints. For us, this is greatly simplified, yet just as effective, by incorporating the search component into the joint density. This means we do not have to implement a stepping out or a doubling procedure which is a part of Neal's algorithm. This is an important point. This is the part of Neal's algorithm which makes it slow in high dimensional problems. This will be demonstrated numerically later in the paper.

We implement a Gibbs sampler based on (3.6). So $p(w, l \mid y, s)$ is easy to sample; being two conditionally independent uniform random variables. Further

$$p(s \mid y, w, l) \propto \frac{p(s)}{s}\, \mathbf{1}\big(s > 2|l - y|\big). \qquad (3.7)$$

This conditional density is also straightforward to sample; and throughout we take $p(s) \propto s\,e^{-\lambda s}$ for some λ, typically in order to provide a large variance. Finally,

$$p(y \mid w, s, l) \propto \mathbf{1}\big(\pi(y) > w\big)\,\mathbf{1}(l - s/2 < y < l + s/2).$$

We sample this using an adaptive rejection sampler; it is also a shrinkage procedure as described in Neal [119]. Before describing the adaptive rejection sampler we present a simple illustration of the key aspects of the one step algorithm, starting with the current value y_0.

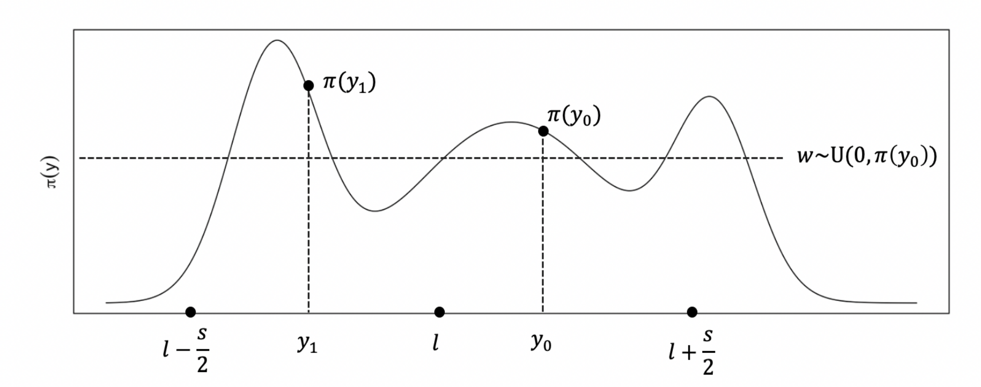


Figure 3.1: Illustration of latent slice sampler

An illustration is provided in Fig. 3.1. The current values of y_0, w and l are indicated. The illustration for this case gives a value of s for which the relevant values of $l - s/2$ and $l + s/2$ are indicated. The proposed value of y_1 is sampled uniformly from $(l - s/2, l + s/2)$ and is accepted if $\pi(y_1) > w$, as shown in the graph. Rejected y gives information about the location of the interval $\pi(y) > w$ and this can be used to improve the proposal with the shrinkage procedure. To generalize the setting we consider adaptive rejection sampling of

$$p(y) \propto \mathbf{1}(y \in C)\,\mathbf{1}(a < y < b),$$

where $C \cap (a, b) \neq \emptyset$ and $y_0 \in C \cap (a, b)$. Let $a_1 = a$ and $b_1 = b$; at iteration m, starting at $m = 1$,

1. Sample y^* uniformly from (a_m, b_m).
2. While $y^* \notin C$: if $y^* < y_0$ then $a_{m+1} \leftarrow \max\{a_m, y^*\}$ else $b_{m+1} \leftarrow \min\{b_m, y^*\}$ and $m \to m + 1$.
3. Repeat steps 1. and 2. until $y^* \in C$; then $y = y^*$.

This works for reasons outlined in Neal [119], and see also the discussion by Walker in Neal's paper. The basic idea is that the sampling strategy resulting in $y = y^*$ conditional on y_0, and write this density as $p(y \mid y_0)$, satisfies detailed balance with respect to $p(y)$; i.e.

$$p(y \mid y_0)\, p(y_0) = p(y_0 \mid y)\, p(y).$$

The obvious points here are that as $p(y)$ is uniform, one only need establish that $p(y \mid y_0) = p(y_0 \mid y)$ which is straightforward to understand. The key being that y and y_0 are both in $C \cap (a_m, b_m)$ for all m and all generated random sets are done so uniformly.

EXAMPLE 1. To see how efficient this sampling strategy is, we take the target for y as a mixture of two normal densities with variances 1 and means -10 and $+10$, and with equal weights. That is,

$$\pi(y) = \frac{1}{2}\,\mathrm{N}(y \mid -10, 1) + \frac{1}{2}\,\mathrm{N}(y \mid 10, 1).$$

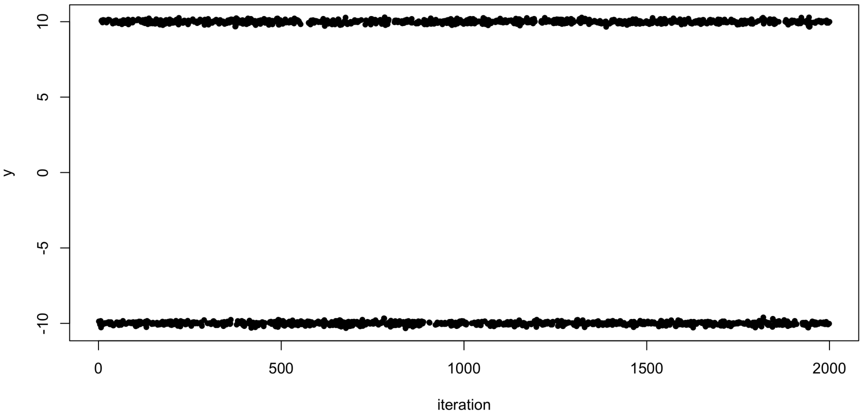


Figure 3.2: Samples from latent slice algorithm from mixture of two normals

We take $p(s)$ to be a gamma distribution with parameters shape equal to 2 and scale equal to 100, i.e., $p(s) \propto s\exp(-0.01s)$, and generate 2,000 samples from the algorithm. The subsequent plot of the sampled y is given in Fig. 3.2. As can be seen, the mixing and accuracy of the samples is excellent. It should be noted that there are very few, if any, alternative algorithms using Markov chains, which could achieve this.

3.0.1 Multivariate Case

From the univariate case there is an easy way to set up a multivariate latent slice sampler when y is a d-dimensional variable. We have the relevant joint density now as

$$p(y, w, s, l) = \mathbf{1}\big(\pi(y) > w\big)\, p(s) \prod_{j=1}^{d} \frac{\mathbf{1}(l_j - s_j/2 < y_j < l_j + s_j/2)}{s_j}.$$

So w remains a one dimensional variable, but the other two; i.e. s and l, are both d-dimensional.

The sampling strategy using a Gibbs sampler is an obvious extension to the one dimensional case. The conditional for y is given by

$$p(y \mid w, s, l) \propto \mathbf{1}\big(\pi(y) > w\big) \prod_{j=1}^{d} \mathbf{1}(l_j - s_j/2 < y_j < l_j + s_j/2).$$

This can also be sampled using the shrinkage procedure; writing $a_j = l_j - s_j/2$, $b_j = l_j + s_j/2$, $y_0 = (y_{01}, \ldots, y_{0d})$ as the current y, and $\{y : \pi(y) > w\} = C$, we sample proposal $y^* = (y_1^*, \ldots, y_d^*)$ from $\prod_{j=1}^{d} \mathbf{1}(a_j < y_j < b_j)$ and accept $y = y^*$ if $y^* \in C$. Otherwise, do for all $j = 1, \ldots, d$:

$$\text{if} \quad y_j^* < y_{0j} \quad \text{then} \quad a_j \leftarrow \max\{a_j, y_j^*\} \quad \text{else} \quad b_j \leftarrow \min\{b_j, y_j^*\}.$$

Example 2. As an illustration we take $\pi(y)$ to be a bivariate normal density with a very high correlation; i.e. we take a mean of $(0, 0)$ and a covariance matrix with unit variances and correlation $\rho = 0.95$. It is known that slice sampling algorithms can perform poorly when the variables are highly correlated; indeed, as stated in Tibbits et al. [148], "It is particularly difficult to create an efficient sampler when there is strong dependence among the variables". We take $p(s)$ to be independent gamma distributions with shape equal to 2 and scale equal to 10. The bivariate plot and contour of the (y_1, y_2) from the output of the

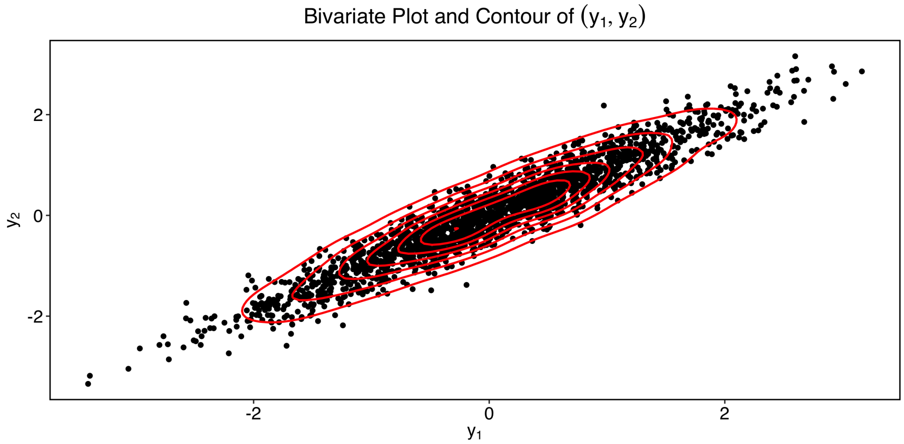


Figure 3.3: Samples from latent slice algorithm from bivariate normal

sampling algorithm is presented in Fig. 3.3. As can be seen this has worked extremely well.

EXAMPLE 3. Here we do a $d = 50$ dimensional example with the target density

$$\pi(y) \propto \exp\left(-\frac{1}{2}\sum_{j=1}^{d} y_j^2\right).$$

The code was written in R and 5000 samples of y were collected. The time for execution was two seconds. We take the same $p(s)$ as that of Example 2. The samples of y_1 are presented as a histogram in Fig. 3.4 along with the standard normal density function for comparison.

3.1 Comparison with Slice Sampling

The algorithm of Neal [119] is concerned with the sampling of $p(y \mid w) \propto \mathbf{1}(\pi(y) > w)$ which is uniform, and let $S = \{y : \pi(y) > w\}$. The aim is to find an interval $I = (L, R)$ which contains the whole, or a part, of S, and to

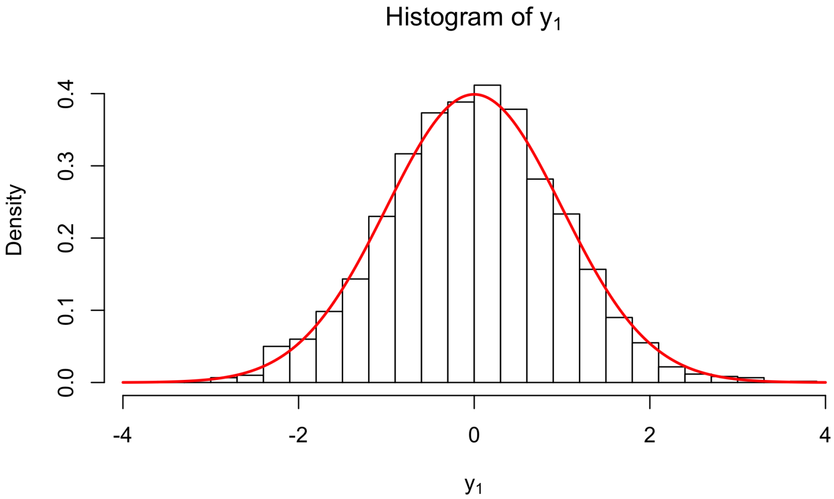


Figure 3.4: Samples of y_1 from latent slice algorithm with 50 dimensional multivariate normal target density

sample a proposal y^* uniformly from I and accept it as y if $y^* \in S$. Now the interval I will be constructed stochastically from $x = y_c$ and hence, as we are dealing with uniform densities; it is required that

$$p(y \mid x, w) = p(x \mid y, w).$$

Effectively, this boils down to the probability of getting I from x being the same as the probability of getting I from y. Neal [119] has two key ideas for constructing I and we will focus on the "stepping out" procedure.

The idea here is to select a positive value k and an integer $m \geq 1$ and start with

$$L = x - k\,(1 - U) \quad \text{and} \quad R = x + k\,U,$$

where U is a uniform random variable from $(0, 1)$. It is already interesting to note that with $m = 1$ this approach would coincide exactly with our own by choosing $s^{-1}\,p(s)$ to be a point mass of 1 at $s = k$. This can be seen by noting that our algorithm selects l uniformly from the interval $(x - k/2, x + k/2)$; i.e. $l = x - k/2 + k\,U$ and then takes y^* uniformly from $(l - k/2, l + k/2)$ which can be written as $(x - k\,(1 - U), x + k\,U)$.

To move on from this rather inflexible strategy, whereas with our algorithm we take $k = s$ as a random variable, Neal accounts for the rigidity of k by allowing the interval to broaden out by extending $L \to L - k$ and $R \to R + k$ until $\pi(L) < w$ and $\pi(R) < w$, respectively, or $J = 0$ and $K = 0$, respectively, where J is a random number in $[0, \ldots, m - 1]$ and $K = m - 1 - J$ and J and K go down by 1 every time an extension is made, respectively. The exact details are presented in Fig. 3 of Neal's paper where a proof is provided that this stochastic construction of I does indeed satisfy detailed balance.

An alternative idea described in Neal [119] is the "doubling" procedure and is described in Fig. 4 of his paper. The starting point is as with the stepping out procedure but now the intervals double in size when the interval is allowed to grow. In short, the additional latent variables l and s we introduce at the outset obviate the need for a doubling or stepping out procedure. So while we are able to treat $k = s$ as random within our framework, and hence deal with any issue arising as a consequence of it being fixed, it has recently been pointed out that some problems are sensitive to the choice of k within Neal's slice sampler; see Karamanis and Beutler [90].

3.1.1 Numerical Comparison

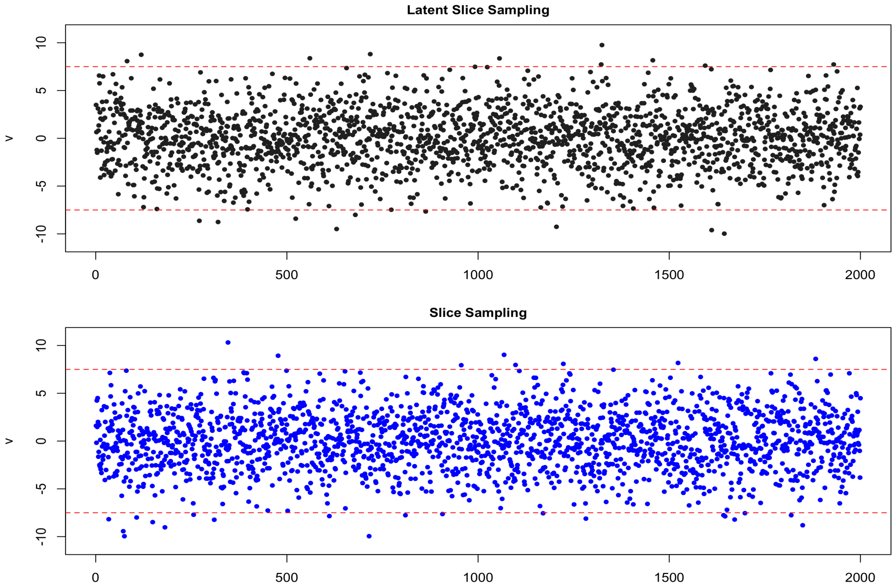
We compared the latent slice sampler with the slice sampling algorithm by using the illustrations in section 8 of Neal's paper. It is a ten-dimensional funnel-like distribution of ten real-valued variables v and x_1 to x_9. The marginal distribution of v is Gaussian with mean zero and standard deviation 3. Conditional on a given value of v, the variables x_1 to x_9 are independent, with the conditional distribution for each being Gaussian with mean zero and variance e^v, which can be formulated as $v \sim \mathcal{N}(v \mid 0, 3^2)$ with $[x_i \mid v] \sim \mathcal{N}(x_i \mid 0, e^v)$ for $i = 1, \ldots, 9$. The joint distribution is obviously given by

$$p(v, x_1, \ldots, x_9) = \mathcal{N}(v \mid 0, 3^2) \prod_{i=1}^{9} \mathcal{N}(x_i \mid 0, e^v). \tag{3.8}$$

Such a distribution is typical of priors for components of Bayesian hierarchical models; x_1 to x_9 might, for example, be random effects for nine subjects, with v being the log of the variance of these random effects. If the data is largely informative, the problem of sampling from the posterior will be similar to that of sampling from the prior. From the above framework, we know the correct marginal distribution for v, which is the focus of the illustration, and we can sample for each of x_1 to x_9 given the value for v.

In Neal's paper, the single variable slice sampling method is used to sample from a multivariate distribution by sampling repeated for each variable in turn. Each update uses the step-out and shrinkage procedure. Fig. 3.5 compared the result of trying to sample from the funnel distribution using latent slice sampling and single-variable slice sampling. The upper plot shows

2,000 iterations of a run, which is the subsampling of 4,000,000 samples with a spacing of $m = 200$ to reduces the autocorrelation of successive samples. If every 200^{th} iteration is used and the rest thrown away, this produces another reversible Markov chain with asymptotic variance. The selection of spacing $m = 200$ can yield better estimates of the true posterior and yet smooth out autocorrelation. We use a gamma distribution with shape 2 and scale 5 to randomize the "slice", i.e $p(s) \propto se^{-s/5}$ so that the sampler is able to explore the distribution efficiently. The lower plot of Fig. 3.5 shows the results of trying to sample from the funnel distribution using single-variable slice sampling. To avoid the high autocorrelation, the same spacing of $m = 200$ is also used to "thin" the simulations.



Latent Slice Sampling

Slice Sampling

Figure 3.5: Sampling the funnel distribution using latent slice sampling (dark dots) and single-variable slice sampling (blue dots)

The resulting 2,000 updates are shown in the scatterplot. Both the latent slice sampler and the single-variable slice sampling perform fairly well with small and large values of v sampled quite good, compared with single-variable Metropolis updates and multivariate Metropolis updates, as discussed in Neal's paper. However, slice sampling method takes much greater cost in wasted computation. The average time for 10,000 iterations are at least 14 times of that for latent slice sampling algorithm. The simplicity of the latent slice sampling makes it favorable for sampling distribution without selecting proposal distribution. By using stochastic search we accelerate the convergence to the stationary distribution.

3.1.2 Effective Sample Size Comparison

To compare the performance of the latent slice sampling with Neal's slice sampling we use *Effective Sample Size* (ESS); see [93]. The ESS is defined as the equivalent number of independent simulation draws from the target distribution which yields the same efficiency in the estimation obtained via the sampling algorithm. It measures the amount by which the autocorrelation in the samples increases the sample standard deviation relative to an independent sample.

For a parameter of interest ν (we are interested in v of the ten-dimensional funnel-like distribution in Chapter 3.1), the ESS, see [56], is given by

$$\text{ESS} = \frac{mn}{1 + 2\sum_{t=1}^{\infty} \rho_t}, \tag{3.9}$$

where m is the number of chains run and n is the length of each chain, and we write $N = nm$ as the sample size, and ρ_t is the autocorrelation function (ACF) at lag t of the chain. The ρ_t is estimated by computing the variogram V_t and using the estimated marginal posterior variance τ^2 of the parameter; i.e. $\widehat{\rho}_t = 1 - V_t/(2\widehat{\tau^2})$, where

$$V_t = \frac{1}{m(n-t)} \sum_{j=1}^{m} \sum_{i=t+1}^{n} (\nu_{i,j} - \nu_{i-t,j})^2$$

and

$$\widehat{\tau^2} = \frac{1}{mn} \sum_{j=1}^{m} \sum_{i=1}^{n} (\nu_{ij} - \bar{\nu}_{\cdot j})^2 + \frac{1}{m-1} \sum_{j=1}^{m} (\bar{\nu}_{\cdot j} - \bar{\nu}_{\cdot\cdot})^2$$

with ν_{ij} $(i = 1, \ldots, n; j = 1, \ldots, m)$ being the output from chain j at iteration i, $\bar{\nu}_{\cdot j} = \frac{1}{n} \sum_{i=1}^{n} \nu_{ij}$ being the within-sequence means, and $\bar{\nu}_{\cdot\cdot} = \frac{1}{m} \sum_{j=1}^{m} \bar{\nu}_{\cdot j}$ being the between-sequence mean. Given the above definition of effective sample size, we take computational efficiency into account and define the measure *MCMC Efficiency*, representing the number of effective independent samples generated per second; i.e. to combine both chain mixing and computational speed, given by

$$\textit{MCMC Efficiency} = \frac{\textit{ESS}}{\textit{Computation Time (in seconds)}}.$$

Table 3.1 and Table 3.2 show the comparisons of computation time (obtained by using `proc.time()` in R), effective sample size and MCMC efficiency between the latent slice sampler and slice sampling under different scenarios. All the numbers are averaged over 5 runs under different initializations. As can be seen from Table 3.1, latent slice sampling is much more computationally

Sampler Sample size	Latent slice sampling	Slice sampling
$N = 10,000$	$1.4s$	$19.0s$
$N = 50,000$	$6.0s$	$88.0s$
$N = 250,000$	$29.9s$	$470.4s$

Table 3.1: The comparison of average system computation time (in seconds) over 5 runs with different initializations under different sample size.

Sampler Sample/Thinning size	Latent slice sampling	Slice sampling
$N = 10,000$ $M = 1$	737 (526.4)	600 (31.6)
$N = 10,000$ $M = 10$	552 (394.3)	464 (24.4)
$N = 50,000$ $M = 1$	3508 (584.7)	2727 (31.0)
$N = 50,000$ $M = 10$	2649 (441.5)	2152 (24.5)
$N = 250,000$ $M = 25$	8528 (285.2)	7561 (16.1)

Table 3.2: The comparison of effective sample size and MCMC efficiency (red colored) under different combination of sample size N and thinning size M (thinning is used to reduce autocorrelation of chains).

efficient than slice sampling. This is because Neal's slice sampling algorithm uses the stepping out procedures for each direction at each step and a posterior probability calculation is required to check the suitability of the new sample, which means the process is time consuming. On the other hand, the latent slice sampling is able to obtain for each direction a valid sample in parallel which does not need to be checked. This is because the latent variables, all trivial to sample and in parallel, result in a Gibbs sampling structure and so the samples can be accepted wherever they fall. The effective sample size shown in Table 3.2 of slice sampling in each scenario is slightly higher than slice sampling, but the MCMC efficiency of latent slice sampling is much higher than Neal's slice sampling due to faster computation. These differences are more significant in very high-dimensional problems.

3.2 Illustrations

In this chapter we present a number of illustrations. We consider a state space model and a variable selection model where the vectors of unknowns are typically sampled component-wise using a Gibbs sampler. In these latter two examples we use the multivariate latent slice sampler to sample the entire vector as a single block.

3.2.1 State Space Model

3.2.1.1 Simulation Example

In this subsection we sample a 500 dimensional space which is the unknown states of a state space, also known as a hidden Markov model. We consider

$$[y_i \mid x_i] \sim \text{Poisson}\big(\theta\, \exp(x_i)\big) \quad \text{and} \quad x_i = \rho\, x_{i-1} + \sigma\, z_i$$

for $i = 1, \ldots, n$ with $n = 500$ and $x_0 = 0$ and the (z_i) independent standard normal. To generate the data set we take $\rho = 0.8$, $\sigma = 1$ and $\theta = 1$.

The joint density of the $x = (x_{1:n})$ given θ is

$$\pi(x \mid \theta) \propto \exp\left\{\sum_{i=1}^{n}\left[x_i\, y_i - \theta\, e^{x_i} - \frac{1}{2}(x_i - \rho\, x_{i-1})^2\right]\right\},$$

for simplicity we assume ρ and σ to be known, without any loss to the illustration about to be presented. Typically, the $\pi(x \mid \theta)$ is sampled component by component, i.e. by sampling $p(x_i \mid x_{-i}, \theta)$ for $i = 1, \ldots, n$ within a Gibbs sampling framework. In some special cases, conditionally normal dynamic linear models, it can be sampled as a block by backward sampling. The most common approaches nowadays are based on particle filters; see Andrieu et al. [5].

Using the multivariate latent slice sampling algorithm we sample the entire vector of state spaces in one block. We only assume θ is unknown and the conditional density of θ with a gamma prior with shape and rate parameters

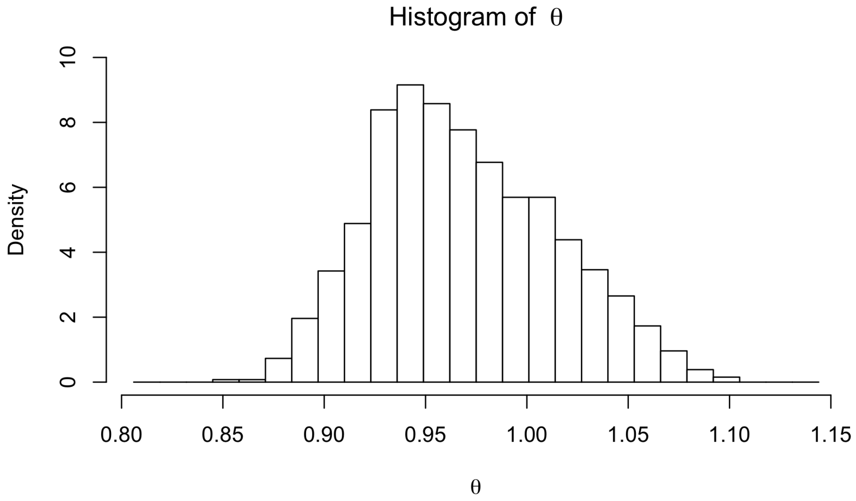


Figure 3.6: Posterior density of θ for state space model

both equal to 0.5 is given by a gamma distribution with shape parameter $0.5 + \sum_{i=1:n} y_i$ and rate parameter $0.5 + \sum_{i=1:n} e^{x_i}$.

The chain was run for 2000 iterations and the time taken was 20 secs. A plot of the posterior θ samples is presented in Fig. 3.6. The mean value is 0.97.

3.2.1.2 Real Data Example

The following illustration is on a real dataset from the NYSE and available from the R package *fBasics*. We take (y_i) to be the log return of the stock price (p_i) with $i = 1, \ldots, n$; i.e., $y_i = \log(p_i/p_{i-1})$, and $n = 9310$. We adopt a stochastic volatility model Hull and White [82],

$$[y_i \mid x_i] \sim \mathrm{N}(0, e^{x_i/2}) \quad \text{and} \quad x_i = \rho x_{i-1} + \sigma z_i \text{ with } 0 < \rho < 1 \text{ and } x_0 = 0.$$

Here the (z_i) are independent standard normal and an initial set of (x_i) are obtained by setting $x_i = 2\log(y_i + \epsilon)$ for some small $\epsilon > 0$. The joint density of $(x_{1:n} \mid \rho, \sigma)$ is given by

$$\pi(x_1, \ldots, x_n \mid \rho, \sigma) \propto \exp\left\{ -\sum_{i=1}^{n} \left(\frac{x_i}{4} + \frac{y_i^2}{2e^{x_i/2}} + \frac{(x_i - \rho x_{i-1})^2}{2\sigma^2} \right) \right\}.$$

The prior for $\lambda = 1/\sigma^2$ is taken to be gamma with both shape a and rate b parameters set to 0.5, while the prior for ρ is uniform on $(0, 1)$. The conditional posterior of λ and ρ are given by

$$\begin{aligned} [\lambda \mid x_1, \ldots, x_n] &\sim \text{Gamma}\left(a + \frac{n}{2}, b + \frac{\sum_{i=1}^{n} (x_i - \rho x_{i-1})^2}{2} \right) \\ [\rho \mid x_1, \ldots, x_n] &\sim \text{N}\left(\frac{\sum_{i=1}^{n} x_i x_{i-1}}{\sum_{i=1}^{n} x_{i-1}^2}, \frac{1}{\lambda \sum_{i=1}^{n} x_{i-1}^2} \right). \end{aligned}$$

We remove 12 outliers that have log returns greater than 5. The final dataset has 9298 observations. On implementation of the latent slice sampler, sampling all the (x_i) together, we constructed posterior distributions from the samples from the output and also show how well the data fits the model. So Fig. 3.7 shows the posterior density of both ρ and λ. Fig. 3.8 shows that estimated mean of the $(e^{x_i/2})$ excellently recovers the original data (y_i^2).

3.2.2 Spike and Slab Model

In this subsection we consider a popular approach to variable selection within the Bayesian framework; namely the spike and slab prior [60]. The model is given by

$$Y = X\beta + \epsilon, \quad \epsilon \sim \text{N}(0, \sigma^2 \mathbf{I}_n)$$

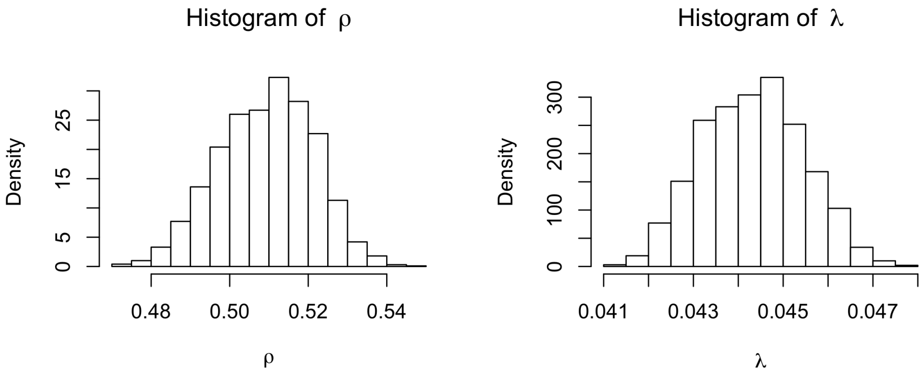


Figure 3.7: Posterior density of ρ and λ

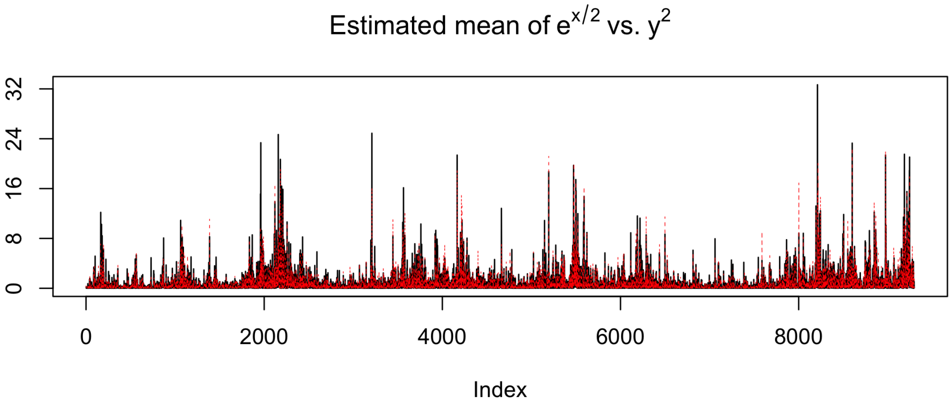


Figure 3.8: The traceplot of estimated mean of $e^{x/2}$ vs. y^2

where $Y \in \mathbb{R}^n$ is a vector of responses, $X = [X_1, \ldots, X_p] \in \mathbb{R}^{n \times p}$ is a regression matrix of p predictors, $\beta = (\beta_1, \ldots, \beta_p)^T \in \mathbb{R}^p$ is a vector of unknown regression coefficients, and $\epsilon \in \mathbb{R}^n$ is the noise vector of independent normal random variables with σ^2 as their unknown common variance. The spike and slab prior for β is given by

$$\pi(\beta) \propto \prod_{j=1}^{p} \left[\sigma_1^{-1} \exp(-\frac{1}{2}\beta_j^2/\sigma_1^2) + \sigma_2^{-1} \exp(-\frac{1}{2}\beta_j^2/\sigma_2^2) \right],$$

where $\sigma_1 \approx 0$ yields the spike and $\sigma_2 \approx \infty$ yields the slab. Markov chain Monte Carlo methods for this model require the Gibbs sampling of β_j conditional on the β_{-j}, i.e. the vector of β without the β_j. See, for example, Narisetty and He [117]. Here we use the latent slice sampler to sample β as one block.

3.2.2.1 Simulated Data

We assume $\sigma = 1$ is known and generate data for $n = 100$ with $p = 90$. We take $\beta_1 = 1$, $\beta_{2:5} = 5$ and $\beta_{6:90} = 0$. All the elements in the design matrix X are generated as independent standard normal random variables. We take $\sigma_1 = 0.1$ and $\sigma_2 = 10$; writing down the posterior for β is quite straightforward and is in particular easy to compute for any given value of β. We ran the latent slice sampler for 10,000 iterations; taking a few seconds to complete the task.

For illustration we present the posterior samples for β_1 and β_2 and β_6; the true values being 1, 5 and 0, respectively. As is visible from Fig. 3.9 the samples are accumulating at the correct locations and the mixing of the chain is good.

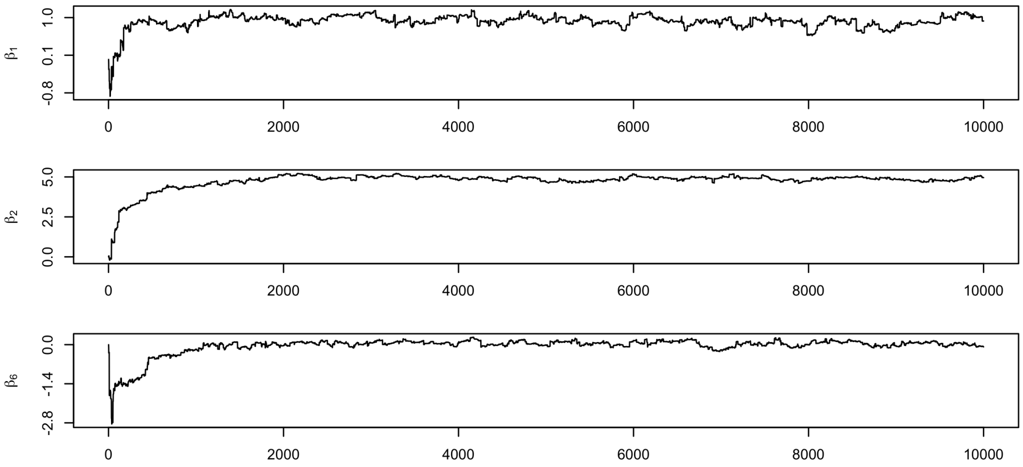


Figure 3.9: Posterior samples of β_1, β_2 and β_6 from spike and slab model

3.2.2.2 General Spike and Slab Model

The spike and slab model with n and K denoting number of observations and number of coefficients is specified in Ishwaran and Rao [84]:

$$
\begin{aligned}
[Y \mid \beta, \sigma] &\sim \mathrm{N}(X\,\beta, \sigma^2\,\mathbf{I}) \\
\beta &\sim \mathrm{N}(0, \Gamma), \text{ where } \Gamma = \mathrm{diag}(d_k \tau_k^2) \\
[d_k \mid v_0, w] &\sim (1-w)\,\delta_{v_0}(\cdot) + w\,\delta_1(\cdot) \\
\lambda_k = \tau_k^{-2} &\sim \mathrm{Gamma}(a_1, a_2) \\
w &\sim \mathrm{Uniform}(0,1) \\
s = \sigma^{-2} &\sim \mathrm{Gamma}(b_1, b_2),
\end{aligned}
$$

where v_0 is a small near-zero value, $\delta_v(\cdot)$ is used to denote a discrete measure concentrated at the value v, and a_1, a_2, b_1, b_2 are known hyperparameters. A variable selection procedure uses a Gibbs sampler and latent slice sampler to simulate posterior values

$$[\beta, \mathcal{D}, \tau, w, \sigma^2 \mid Y]$$

where $\mathcal{D} = (d_1, \ldots, d_K)^T$ and $\tau = (\tau_1, \ldots, \tau_K)^T$. Also, $\gamma_k = d_k \tau_k^2$, so simulating $\mathcal{D}$ and τ provides a value for $\gamma = (\gamma_1, \ldots, \gamma_K)^T$. Denote $\Gamma = \mathrm{diag}(\gamma_1, \ldots, \gamma_K)$ as the $K \times K$ diagonal matrix, then the sampling procedure works as follows: note the only difference between our algorithm and the standard Gibbs sampler is given in item 1. below,

i. For the *latent slice sampler*, simulate β with the posterior distribution given by

$$p(\beta \mid \gamma, \sigma^2, X, Y) \propto \exp\left(-\frac{1}{2\sigma^2}\,||Y - X\,\beta||^2 - \frac{1}{2}\,\beta^T\Gamma^{-1}\beta\right)$$

For the *Gibbs sampler*, sample $[\beta \mid \gamma, \sigma^2, Y] \sim \mathrm{N}(\mu, \sigma^2\,\Sigma)$, where

$$\mu = \Sigma\,X^TY \qquad \text{and} \qquad \Sigma = \left(X^TX + \sigma^2\,\Gamma^{-1}\right)^{-1}$$

ii. Simulate d_k from its conditional distribution

$$[d_k \mid \beta, \tau, w] \sim \frac{w_{1,k}}{w_{1,k} + w_{2,k}}\delta_{v_0}(\cdot) + \frac{w_{2,k}}{w_{1,k} + w_{2,k}}\delta_1(\cdot)$$

where

$$w_{1,k} = (1 - w)v_0^{-1/2}\exp\left(-\frac{\beta_k^2}{2v_0\tau_k^2}\right) \quad \text{and} \quad w_{2,k} = w\exp\left(-\frac{\beta_k^2}{2\tau_k^2}\right)$$

iii. Simulate $\lambda_k = \tau_k^{-2}$ from its conditional distribution

$$[\tau_k^{-2} \mid \beta, \mathcal{D}] \sim \mathrm{Gamma}\left(a_1 + \frac{1}{2}, a_2 + \frac{\beta_k^2}{2d_k}\right)$$

iv. Simulate w, the complexity parameter, from its conditional distribution

$$[w \mid \mathcal{D}] \sim \text{Beta}\big(1 + \#\{k : d_k = 1\}, 1 + \#\{k : d_k = v_0\}\big)$$

v. Simulate $s = \sigma^{-2}$ from its conditional distribution

$$[\sigma^{-2} \mid \beta, Y] \sim \text{Gamma}\left(b_1 + \frac{n}{2}, b_2 + \frac{1}{2}||Y - X\,\beta||^2\right)$$

vi. Update γ by setting $\gamma_k = d_k \tau_k^2 = d_k/\lambda_k$, for $k = 1, \ldots, K$, and this completes one iteration.

One of the applications of the spike and slab regression is on “big$-p$, small$-n$” (denoted as $p \gg n$) problems. Gene expression arrays are examples of high dimensions with large magnitude of variables. They typically have 50 to 100 samples and 5,000 to 20,000 variables (genes). There have been many attempts to adapt statistical models for regression and classification to these data, and in many cases these attempts have challenged the computational resources. Standard statistical models, such as linear regression model, logistic regression, and the Cox model cannot be used “out of box", since the standard fitting algorithms all require $p < n$. Many existing methods used a standard fitting method with quadratic regularization to overcome the dilemma. Ishwaran et al. [85] introduced a generalization of the elastic net (*gnet*) to obtain much sparser variable selection.

To demonstrate the efficiency of latent slice sampling on the spike and slab regression model of the “big$-p$, small$-n$” problem, we take $n = 100$

observations with $\beta_1 = 1$, $\beta_{2:5} = 5$ and $\beta_{6:p} = 0$, where $p = 2000$. All the elements in the design matrix X are generated as independent standard normal random variables. Fig. 3.10 shows the traceplots of three randomly selected coefficients (β_1, β_5, and β_{1800}) simulated by latent slice sampling. In this case, Gibbs sampler fails completely with dimension 100×2000 due to the inefficient calculation of the inverse matrix $X^T X + \sigma^2 \Gamma^{-1}$ in the covariance, while latent slice sampling is able to work on high-dimensional data and maintain decent convergence with appropriate size of simulations. As the number of variables decreases to less than 500, both methods are capable of gaining good mixing. The latent slice sampling tends to be more computationally efficient than Gibbs sampling with large amount of variables, while Gibbs sampling performs better in terms of stationarity and efficient computation with lower dimensional data.

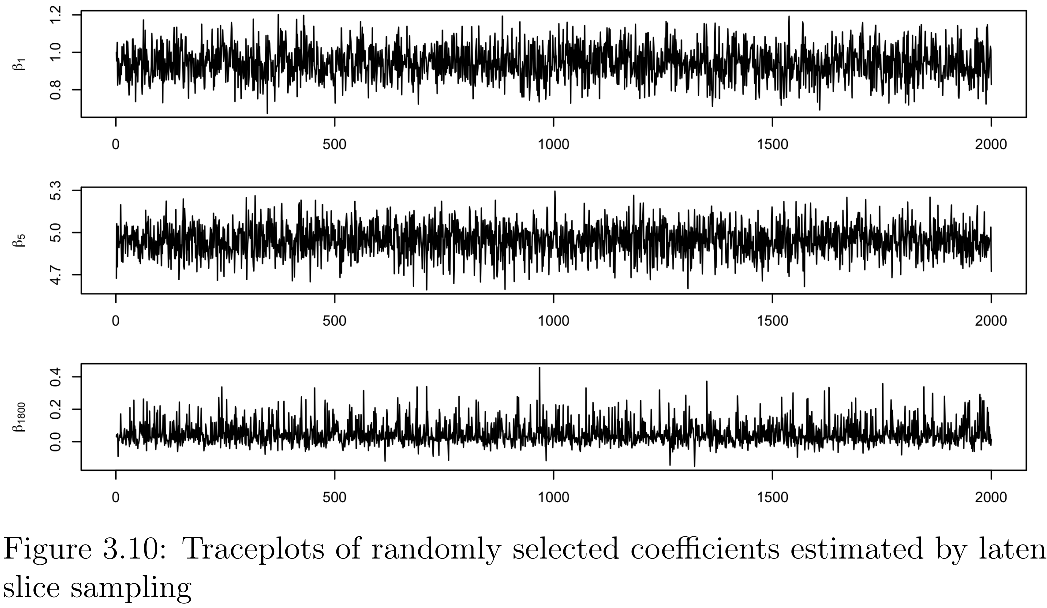


Figure 3.10: Traceplots of randomly selected coefficients estimated by latent slice sampling

3.2.2.3 Real Data Analysis

We apply the latent slice sampling method to the Engel Curve data [16], which consists of a random sample taken from the British Family Expenditure Survey for 1995 and can be loaded from the R package *np* with the dataset labelled as Engel95. There are 1655 household-level observations and 10 variables, including expenditure share of food, catering, alcohol, fuel, motor, fares, leisure, logarithm of total earnings, number of children, and the dependent variable – logarithm of total expenditure. In addition to the 9 baseline explanatory variables, we added 20 dummy variables, each sampled independently from a binomial distribution with probability generated uniformly between 0 and 1, which gives us a dataset with dimension 1655 × 29.

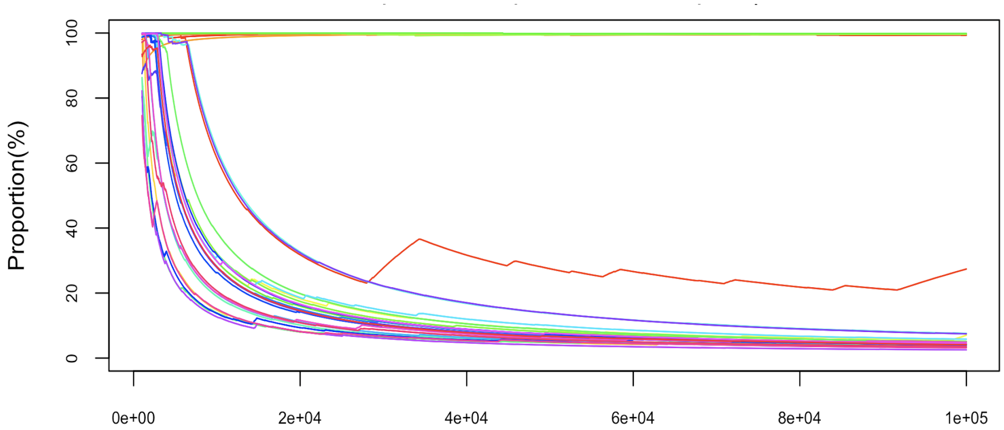


Figure 3.11: Proportions of the spikes to slabs for all the β

Fig. 3.11 shows traceplots of the proportions of spikes to slabs samples on all coefficients. As can be seen, the proportions converge to fixed values after adequate amount of sampling. The proportion of spikes converge to 1 for variables food, alcohol, fuel, leisure, logarithm of total earnings and one of the simulated dummy variables.

3.2.3 Uniform Sampling in High Dimension

A common problem in many contexts is the ability to sample uniformly from a region $S \subset \mathbb{R}^d$ for some large d. A recent paper, Chen et al. [28] considers various existing and new Markov chain Monte Carlo sampling algorithms for uniform sampling from a polytope; a set of the form $S = \{y \in \mathbb{R}^d : A\,y \leq b\}$. However, the largest d considered in this paper appears to be $d = 50$. On the other hand, while sampling as a single block and with more general forms for S, we achieve good mixing and output with spaces of size $d = 5000$.

We can achieve this using our latent slice sampling algorithm as follows. So consider the joint density for (y, s, l) where each component is a d dimensional vector,

$$p(y, s, l) \propto \mathbf{1}(y \in S)\, p(s) \prod_{j=1}^{d} \frac{\mathbf{1}(y_j - s_j/2 < l_j < y_j + s_j/2)}{s_j}.$$

By implementing a Gibbs sampler, the conditionals for s and l are easy to sample. As usual we can take $p(s) \propto s \exp(-\lambda s)$ for some $\lambda > 0$. Then the conditional for y as a single vector is

$$\pi(y \mid s, l) \propto \mathbf{1}(y \in S) \prod_{j=1}^{d} \mathbf{1}(a_j < y_j < b_j)$$

and $a_j = l_j - s_j/2$ and $b_j = l_j + s_j/2$.

This is sampled using the shrinkage procedure; as this progresses, sample

y^* uniformly from (a_j, b_j) and while $y^* \notin S$, update a_j to $\max\{a_j, y_j^*\}$ if $y_j^* < y_j$ else update b_j to $\min\{b_j, y_j^*\}$. Here $y = (y_j)$ is the current vector of the chain.

We apply this scheme to the set

$$S = \left\{ y : \sum_{j=1}^{d} y_j \leq 1, \quad y_j \geq 0, \quad D(y, \underline{1}) \leq c_1 \ \& \ D(y, \underline{1}) > c_2 \right\},$$

where $D(y, \underline{1})$ denotes the Kullback-Leibler divergence between y and the uniform vector $\underline{1} = (1/(1+d))$; i.e.

$$D(y, \underline{1}) = \sum_{j=1}^{d} y_j \log y_j + (1-w)\log(1-w) + \log(1+d),$$

where $w = \sum_{j=1:d} y_j$. This is a disconnected region; indeed for $d = 2$ and $c_1 = 0.3$ and $c_2 = 0.7$ there are 4 separated regions within $y_1 + y_2 \leq 1$.

We take d as a super large value; $d = 5000$, and take $c_1 = 0.3$ and $c_2 = 0.5$ and sample y as a single block. We take $\lambda = 1$ and generate $n = 10,000$ samples. In Fig. 3.12 the trace output of the y_1 samples are plotted. As can be seen the chain is mixing well and is able to move with both small and large steps. The time taken for the output was approximately one minute.

3.3 Discussion

In this chapter we have presented a generic sampling algorithm which has the ability to sample efficiently very high dimensional distribution functions at great speed. The key is the latent model combined with the shrinkage procedure based on uniform distributions and an automatic reversible condition.

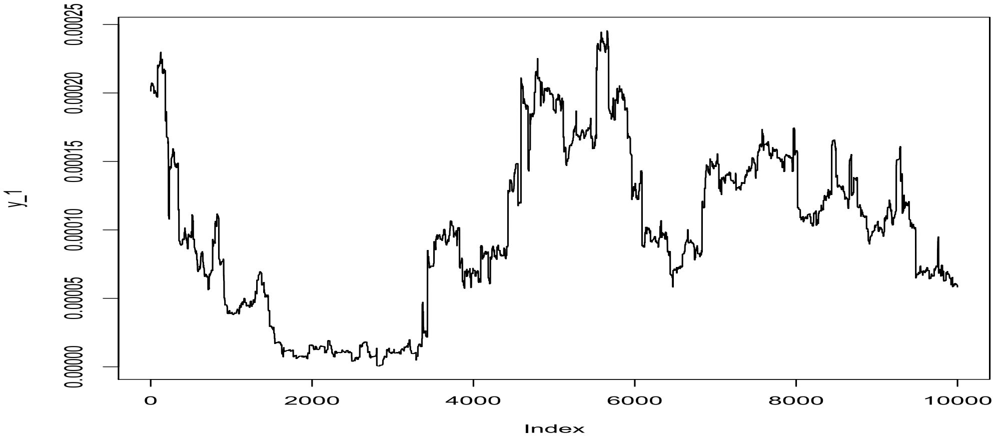


Figure 3.12: Traceplot of y_1 samples from uniform sampling in high dimension

Given the simplicity of the algorithm we present it here, in the general d-dimensional case, with target density $\pi(y)$ and $y = (y_1, \ldots, y_d)$. Let $\lambda = 0.1$, for example; we describe a single loop with current values $y_0 = (y_{01}, \ldots, y_{0d})$ and $s_0 = (s_{01}, \ldots, s_{0d})$.

i. Sample $w \sim \mathrm{U}(0, \pi(y_0))$ and, for $j = 1, \ldots, d$, sample

$$l_j \sim \mathrm{U}(y_{0j} - s_{0j}/2,\, y_{0j} + s_{0j}/2)$$

and sample s_j from the density proportional to

$$\exp(-\lambda\, s_j)\, \mathbf{1}(s_j > 2\,|l_j - y_{0j}|).$$

ii. Set $a_j = l_j - s_j/2$ and $b_j = l_j + s_j/2$.

iii. For $j = 1, \ldots, d$, sample

$$y_j^* \sim \mathrm{U}(a_j, b_j);$$

```
    if $\pi(y^*) > w$, accept $y = y^*$; else, for $j = 1, \ldots, d$,
        if   $y^*_j < y_{0j}$   then   $a_j \leftarrow \max\{a_j, y^*_j\}$   else   $b_j \leftarrow \min\{b_j, y^*_j\}$.
iv. Repeat step 3. until $\pi(y^*) > w$ and set $y = y^*$.
```

As we have demonstrated, such an algorithm can work with a nonlinear state space model with dimension 500 and return output in short time. Future work will consider sampling of constrained spaces, such as uniform sampling on polytopes and truncated distributions, such as the multivariate normal [132, 34].

Chapter 4

A Latent Slice Sampler on Multivariate Binary Spaces

In this chapter, we review standard Monte Carlo methods for sampling high-dimensional binary vectors and motivate the work on an alternative sampling scheme based on latent slice sampling methodology. Most of this discussion was published in Li and Walker [102]. Standard approaches are typically based on random walk type Markov chain Monte Carlo, where the equilibrium distribution of the chain is the distribution of interest and its ergodic mean converges to the expected value of interest. While MCMC methods are asymptotically valid, convergence of Markov chains may be very slow if the distribution of interest is highly multi-modal.

This chapter proposes a novel algorithm based on latent slice sampling methodology which copes well with multi-modal problems. This work approaches a well-studied problem from a different angle and provides new perspectives. Firstly, there is numerical evidence that particle methods, which

The content in this chapter has been revised for the Journal of Computational and Graphical Statistics.

track a population of particles, initially well spread over the sampling space, are often more robust than local methods based on MCMC, since the latter are prone to get trapped in the neighborhood of local modes. Secondly, latent slice sampling type algorithms are easily parallelizable, and parallel computing for Monte Carlo algorithms has gained a tremendous interest in the very recent years [100, 120], due to the increasing availability of multi-core processing units in standard computers. Thirdly, we argue that the latent slice sampler is fully adaptive and requires practically no tuning to perform well. A Monte Carlo algorithm is said to be adaptive if it adjusts, sequentially and automatically, its sampling distribution to the problem at hand. Important classes of adaptive Monte Carlo are sequential Monte Carlo [115], adaptive importance sampling [20] and adaptive Markov chain Monte Carlo [3], among others. The choice of the parametric family which defines the range of possible sampling distributions is critical for good performance.

Slice sampling is a powerful technique for generating random variables from complicated density functions; see Besag and Green [14], Damien et al. [35], and Neal [119]. The motivation is simple enough; for a target density $\pi(x)$, $x \in \mathbb{R}^M$, the idea is to introduce the latent variable w and consider the joint density $f(x, w) = \mathbf{1}(w < \pi(x))$. A Gibbs sampler can be implemented in which the sampling of w is straightforward and the sampling of x, conditional on w, involves sampling uniformly from the interval $A_w = \{x : \pi(x) > w\}$. It is the sampling of this latter uniform distribution which poses the problem for slice samplers. Indeed, uniform sampling from high dimensional spaces is a problem in its own right; see, for example, Chen et al. [28]. However, the aim would be to achieve this without recourse to complicated MCMC algorithms since this would defeat the object of the slice sampler, and one could well be better off performing a direct MCMC on $\pi(x)$.

Neal [119] introduced a clever procedure for sampling the uniform distribution on A_w using a transition density $f(x' \mid x)$ satisfying $f(x' \mid x, w) = f(x \mid x', w)$, and hence is stationary with respect to the target uniform density. A strategy for proposing a x' given x satisfying the reversible condition is given in Neal [119]. This involves a stepping out or doubling procedure combined with a shrinkage procedure. The former procedures require choosing a width parameter which becomes fixed over the run of the chain and is, particularly in high dimensions when one is required for each dimension, potentially a tricky tuning parameter to set. Further, the stepping out and doubling procedures need to be performed sequentially with a computation of $\pi(x)$ after each step. This makes it difficult to implement in high dimensions.

On the other hand, Li and Walker [102] introduce further latent variables which facilitates the uniform sampling via a Gibbs framework. The algorithm avoids the stepping out or doubling procedures and hence avoids the need for the tuning width parameters and a potentially slow sequential search for a valid substitute for A_w when the dimensions are large. They start with, writing in the one dimensional case, the joint density

$$f(x, w, s, l) = \mathbf{1}(w < \pi(x))\, s^{-1} p(s)\, \mathbf{1}(x - s/2 < l < x + s/2) \qquad (4.1)$$

for some density function $p(s)$ on $(0, \infty)$. As before, all variables are easy to sample and now the required density for x we need to sample is

$$f(x \mid w, s, l) \propto \mathbf{1}(w < \pi(x))\, \mathbf{1}(l - s/2 < x < l + s/2).$$

This structure allows for an easy search for a valid substitute for A_w, just from the sampling of s. It is then also easy to set up a sequence of proposals x' satisfying $f(x' \mid x, w, s, l) = f(x \mid x', w, s, l)$. The algorithm is detailed in Li and Walker [102]; briefly here, take an initial proposal x^* uniformly from the interval $(L_-, L_+) = (l - s/2, l + s/2)$ and keep repeating this until $w < \pi(x^*)$. After each rejection, the uniform interval, currently (L_-, L_+), can be narrowed to

$$(x^*, L_+) \quad \text{if} \quad x^* < x, \qquad \text{or} \qquad (L_-, x^*) \quad \text{if} \quad x^* > x,$$

where x is the current value. As the rejections mount, the interval will start to concentrate on x. This is effectively equivalent to the shrinkage procedure described in Neal [119]. Hence, at the very least, the algorithm can become a local sampler, but with the possibility of having large jumps.

The accepted sample x' and the current value are reversible according to this procedure; i.e. $f(x' \mid x, \ldots) = f(x \mid x', \ldots)$. This is one of the ideas behind Neal [119], though we avoid the stepping out and doubling parts of his algorithm. Extending the dimensions is straightforward and numerous illustrations are presented in Li and Walker [102].

Walker [152] and Ekin et al. (2021) cover the case when $x \in \mathbb{N}^M$.

However, a class of problematic density functions is given by

$$\pi(z_1, \ldots, z_M) \tag{4.2}$$

where each $z_j \in \{0, 1\}$, or an equivalent binary set. The two aforementioned papers do not cover this case; they would both collapse to a Gibbs sampler, which is not in general suitable for the distributions we are going to look at.

When M is large these densities can be difficult to sample, particularly if they are multimodal. Typically, Metropolis algorithms or Gibbs samplers are applied where one of the variables is updated at a time. Such algorithms may not produce sufficiently well-mixing chains, or even a chain that moves at all. We demonstrate through a number of illustrations the mixing abilities of the latent slice sampler for multivariate binary distributions. We compare with other generic algorithms, such as the single flip proposal Metropolis algorithm. Our algorithm is closely related, but an extension of the random walk kernel sampler, described in Schäfer and Chopin [30]. The random walk sampler proposes moves z' from z for which $||z' - z|| \leq k$; i.e. the number of switches is bounded by a fixed number k. On the other hand, our algorithm can be seen as a version of this in which the right latent variables are introduced so that we can make k random and maintain a valid chain. There are models, such as the Ising model, where due to the nature of the distributions, specialized algorithms work, such as the Swendsen-Wang [145] and Wolff [155] algorithms.

In this chapter, we apply the latent slice sampling algorithm to the joint density (4.2). The trick is to introduce a latent variable, say y_j for each z_j, and set $z_j = \mathbf{1}(y_j > 0)$. This gives us a joint density in (y_j) which can be sampled as in Li and Walker [102]. If it is possible to allow the sampler on the (y_j) to move sufficiently around some bounded space in M-dimensions, we should be able to construct a sampler which also jumps around in the $\{0, 1\}^M$ space.

The layout of this chapter is as follows: in Chapter 4.1 we describe the details of the algorithm. We also provide some theory about the algorithm and prove the reversibility of the sampling of $f(x' \mid w, s, l)$. In Chapter 4.2 we first present a couple of introductory examples with some further substantial illustrations presented subsequently; in Chapter 4.3 and Chapter 4.4, the Ising model and a Bayesian variable selection model are illustrated. Chapter 4.5 presents a comparison of the latent slice sampler and Metropolis algorithms by looking at eigenvalues of transition probability matrices and also consider effective sample sizes. Chapter 4.6 presents a brief discussion.

4.1 Latent Slice Sampling Algorithm

Sampling from $\pi(z)$ is equivalent to sampling from the joint density

$$f(y, w, s, l) \propto \mathbf{1}\Big(w < \pi(\mathbf{z})\Big) \prod_{j=1}^{M} \frac{p(s_j)}{s_j} \mathbf{1}\Big(y_j - \frac{s_j}{2} < l_j < y_j + \frac{s_j}{2},\ |y_j| < a\Big),$$

where $z_j = \mathbf{1}(y_j > 0)$, for some $a > 0$. The introduction of the finite a here is to ensure the joint density is proper. As with the continuous case, the variables are all easy to sample, and the $y = (y_1, \ldots, y_M)$ can be sampled jointly, as in the continuous case, and with the shrinking procedure, until the proposal y^* satisfies $w < \pi(z^* = \mathbf{1}(y^* > 0))$. Write $y^* = y_0^*$ as the initial proposal and, if all are rejected, let (y_r^*) be the sequence of proposals.

At each iteration, the initial proposal z^* is being sampled approximately uniformly on $\{0,1\}^M$. This is equivalent to restarting the chain. However, rather than the chain move aimlessly along points with low probability looking for a point with high probability, it drifts back to the current value, with each interim point being tested for a possible move. If nothing is accepted along the way, the chain stays at its current value and the next iteration proceeds with another uniform sample being generated. Viewed in this way, the algorithm provides a jump mechanism with multiple proposals and if these are all rejected it behaves as a local sampler.

Here we write the algorithm (detailed as a single loop) for the sampler for a given π and with $a = 2$ (the choice of a is without loss of generality), and $p(s) \propto \exp(-s\lambda)$:

i. Initializing step: given $s = s_{1:M}$ and $z = z_{1:M}$ and for each $i = 1, \ldots, n$, set $y_i > -a$ negative if $z_i = 0$ and $y_i < a$ positive if $z_i = 1$

ii. Set $a_i = b_i = 0$ for $i = 1, \ldots, n$. Sample $w = u_1 \pi(z)$ and for $i = 1, \ldots, n$, take $l_i = y_i - s_i/2 + u_{i2} s_i$, where u_1 is a uniform r.v. and the u_{i2} are i.i.d. uniform r.v.s.

iii. Sample s_i as an exponential r.v. with mean $1/\lambda$ and constrained to be greater than $2|l_i - y_i|$. Set $a_i = \max\{-a, l_i - s_i/2\}$ and $b_i = \min\{a, l_i + s_i/2\}$

iv. Propose the new y'_i as $a_i + v_i(b_i - a_i)$, where the v_i's are i.i.d. uniform r.v.s, and $z'_i = 1(y'_i > 0)$

(1). Compute $w' = \pi(z')$

(2). If $w' > w$ then $z = z'$ and GoTo step (ii)

(3). Else, set $q = 1(y'_i < y_i)$ then reset $a_i = q \max\{a_i, y'_i\} + (1-q)a_i$ and $b_i = (1-q)\min\{b_i, y'_i\} + qb_i$. GoTo step (iv) and repeat until $w' > w$.

To demonstrate the properties of the sampler, we now show that it can propose any point from any current location with high probability. Due to the independence nature of the proposals, we only need to consider one dimension. Let y_0 and s_0 be the current values and assume without loss of generality that $y_0 > 0$. We take $p(s)$ to be proportional to $se^{-s/\lambda}$ so the conditional for s constrained to be larger than η can be sampled as $\psi + \eta$, where ψ is an exponential random variable with mean λ.

Lemma 4.1.1. *The probability that the initial proposal for y, i.e. y^*, has probability of being negative, given the current value y_0 is positive, is given by*

$$\frac{1}{2} \max\left\{0, 1 - \frac{y_0 + s_0(2v-1)}{\psi + s_0|2v-1|}\right\},$$

where v is an independent standard uniform random variable.

The proof is straightforward. Noting that $0 < y_0 < a$ we see that provided λ is sufficiently large, the probability of y^* being negative when y_0 is positive (and vice versa) can be close to $\frac{1}{2}$.

Before proceeding we focus on the sampling of the $f(y \mid w, s, l)$ via a reversible Markov sequence. To do this we set up a more generic setting for the problem. So let

$$f(y) \propto \mathbf{1}(y \in C)\,\mathbf{1}(y \in B)$$

where C is an unknown interval, but a specific value of y can be tested to see whether it lies in C or not, and B is a known single connected interval. Define the initial $B = B_0 = (a_0, b_0)$ and let y_0 be the current point which lies in $C \cap B$. The sequence of proposals $(y_r)_{r\geq 1}$ is given by

$$y_r = a_{r-1} + u_r(b_{r-1} - a_{r-1}), \tag{4.3}$$

where the (u_r) are an independent sequence of standard uniform random variables, and if $y_r \notin C$, the B_{r-1} is updated to B_r via

$$a_r = a_{r-1}\,\mathbf{1}(y_r > y_0) + y_r\,\mathbf{1}(y_r < y_0) \quad \text{and} \quad b_r = b_{r-1}\,\mathbf{1}(y_r < y_0) + y_r\,\mathbf{1}(y_r > y_0). \tag{4.4}$$

This sequence continues until $y_r \in C$ for some r.

Lemma 4.1.2. *If I_r is the current length of the interval from which y_r is taken uniformly, then the size of the next interval is random and $I_{r+1} = u\,I_r$, where u is a uniform random variable from $(0,1)$ and independent of I_r.*

Proof. If the interval $B_r = (a_r, b_r)$, then $y_{r+1} = a_r + u(b_r - a_r)$, and

$$a_{r+1} = a_r\,\mathbf{1}(y_{r+1} > y_0) + (a_r + u(b_r - a_r))\,\mathbf{1}(y_{r+1} < y_0)$$

and

$$b_{r+1} = b_r\,\mathbf{1}(y_{r+1} < y_0) + (a_r + u(b_r - a_r))\,\mathbf{1}(y_{r+1} > y_0).$$

Hence,

$$I_{r+1} = b_{r+1} - a_{r+1} = (b_r - a_r)\left((1-u)\mathbf{1}(y_{r+1} > y_0) + u\mathbf{1}(y_{r+1} < y_0)\right).$$

This completes the proof. □

Corollary 4.1.2.1. *If $m = \min_r\{I_r/I_0 < \epsilon\}$, then m is a $1 + \mathit{Pois}(-\log\epsilon)$ random variable.*

These two lemmas indicate how the sampler acts as both a jump, almost uniform, and local sampler. And recall that at each iteration as the sampler moves from its initial proposal y^* back to y_0, a new proposal is being made. In short, a sequence of proposals is being generated ranging from a jump proposal to a local proposal, the latter applying if all the jump proposals are rejected.

The next two results establish that our shrinking procedure leaves the posterior distribution invariant. First, we note that Corollary 4.1.2.1 implies that the shrinking procedure will terminate almost-surely for almost-all starting values y. In particular, it will terminate when y is a continuity point of $\pi(y)$, and the set of discontinuity points of $\pi(y)$ has Lebesgue measure 0.

Lemma 4.1.3. *Let J denote the number of rejected points in the shrinking procedure and suppose that y is a continuity point of $f(y)$. Then J is finite almost-surely.*

Proof. With probability 1 we will have $w < f(y)$, and by continuity we will have $w < f(y \pm \epsilon)$ for sufficiently small ϵ. Hence, if the shrinking procedure is eventually contained in an ϵ-neighborhood of y the procedure will terminate. Corollary 4.1.2.1 implies that the time for this to occur is Poisson distributed, and hence finite almost surely. □

Theorem 4.1.4. *The shrinking procedure defined by (4.3) and (4.4) defines a Markov transition function $Q(y \mid y', w, s, l)$ which is* reversible *in the sense that $f(y \mid w, s, l)\, Q(y' \mid y, w, s, l) = f(y' \mid w, s, l)\, Q(y \mid y', w, s, l)$.*

Complete proof of Theorem 4 by Dr. Antonio Linero can be found in Appendix B.

4.2 Introductory Illustrations

In the following, we present a number of illustrations, starting with two simple expository examples. We then move to more substantive cases involving high dimensional models, including the Ising model in Chapter 4.3 and variable selection models in Chapter 4.4.

4.2.1 Example 1

To demonstrate the accuracy of the algorithm we present a simple example where M is small enough so we know exactly the 2^M probabilities. We take $M = 3$ and

$$\pi(z_1, z_2, z_3) = \frac{e^{z'Az}}{\sum_{z \in C} e^{z'Az}}$$

where C is the set of 8 possible values of z. The matrix A is randomly generated with independent standard normal random variables. The matrix A is

$$A = \begin{pmatrix} -0.322 & -0.314 & -1.541 \\ 0.332 & 1.109 & -0.909 \\ -0.391 & 0.213 & 0.118 \end{pmatrix}$$

and the correct probabilities are

$$p_{0,0,0} = 0.099, \quad p_{0,0,1} = 0.111, \quad p_{0,1,1} = 0.168, \quad p_{1,1,1} = 0.018$$

$$p_{1,0,1} = 0.012, \quad p_{0,1,0} = 0.300, \quad p_{1,0,0} = 0.072, \quad p_{1,1,0} = 0.221.$$

The algorithm was run for 100,000 iterations and the estimated probabilities are

$$\widehat{p}_{0,0,0} = 0.097, \quad \widehat{p}_{0,0,1} = 0.113, \quad \widehat{p}_{0,1,1} = 0.171, \quad \widehat{p}_{1,1,1} = 0.018$$

$$\widehat{p}_{1,0,1} = 0.012, \quad \widehat{p}_{0,1,0} = 0.295, \quad \widehat{p}_{1,0,0} = 0.072, \quad \widehat{p}_{1,1,0} = 0.221.$$

The mixing is excellent, as an illustration we present a plot of the first 100 samples of the (z_1) variable in Fig. 4.1.

The choices for the algorithm include $p(s)$ and a. The idea is to enable the intervals $(l - s/2 < y < l + s/2 \cap |y| < a)$ to be large; therefore a is not such an important choice, and we fix it at 2, while to ensure the largest intervals we take $p(s) \propto se^{-\lambda s}$ with $\lambda = 0.05$. Note then that the sampling of the s within an iteration is an exponential random variable with parameter λ added to $2|y - l|$.

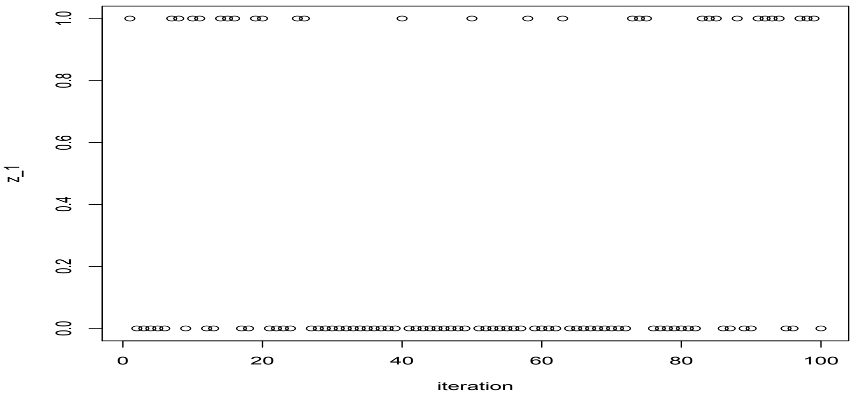


Figure 4.1: Plot of first 100 samples of z_1

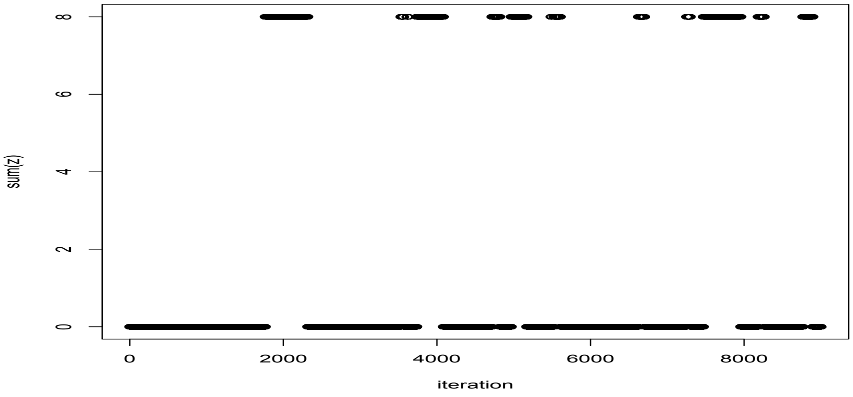


Figure 4.2: Plot of sum of components of z vector

4.2.2 Example 2

Another example, but a demanding one, is taking $M = 8$ and $\log\pi(z_j \equiv 1) = \log\pi(z_j \equiv 0) \propto 100$ with all the other vectors for z have $\log\pi(z) \propto 1$, so the probabilities differ by 100 on log scale. This distribution is bimodal with no route via local sampling from one to the other. Indeed, any local sampler would fail to move from one of the modes once there.

This illustration is as difficult for local samplers as it possibly can be for distributions on $\{0, 1\}^M$. There are two separated modes with single points and with all other probabilities effectively 0. The only way to be able to jump between modes in this case is to have uniform proposals. Our algorithm has this as a key component. Needless to say, a Metropolis sampler or Gibbs sampler would not be able to switch mode once one has been reached.

The algorithm mixes over the two modes well; see Fig. 4.2. The vertical axis represents the sum of the components of the z vector which has modes at 0 and m. A pure local sampler would of course not leave a mode once reached. To test this illustration to an extreme, we set $M = 20$. On a number of runs of size 10^6 we get at least one switch between the two modes. Note that 2^{20} is just over 1,000,000. Hence, the nature of the sampler is as if we restart the chain randomly at a location for each iteration. However, instead of the chain then moving locally about this location, it moves, with proposals at each step, which can be accepted, towards the previous location and hence can then at least mimic a local sampler.

4.3 Ising Model

In 1920s, the physicists Ernst Ising and Wilhelm Lenz proposed a very simple model, called Ising model, to study magnetism. Since that time, the ising model has been intensively investigated for the betterment of statistical physics.

The Ising model [32] is considered as a lattice of sites containing N spins, a structure that gives the model computational advantages over other statistical systems. A spin of the Ising model has two states: an up state and a down state. If we denote a spin at a site i by the symbol z_i, then the up state takes the value of $z_i = 1$ and the down state $z_i = -1$.

Let i and j refer to two nearest neighbor sites on the lattice and let z_i and z_j be the spins on these sites, which are also considered as dipole moments. Having the spins of the Ising model as dipole moments, we consider this model as a real magnetic material, in which constant dipole interactions are taking place. With that in mind, the energy associated with a pair of nearest neighbor spins is then given by $-Jz_iz_j$, meaning that, if the two spins are aligned (up or down), then the energy associated with them is $-J$; otherwise, this energy is $+J$. Physically, the value of J measures the strength of a spin-spin interaction such that $J > 0$ corresponds to a *ferromagnetic* material; $J < 0$, the *anti-ferromagnetic* material.

Now we take $\mathbb{C}_M^2 = \{(\alpha, \beta) : \alpha, \beta \in [1, M]\}$ and define a neighborhood

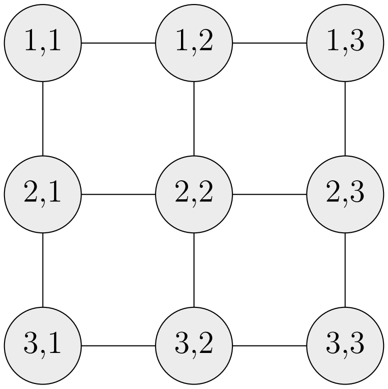


Figure 4.3: Graph showing the neighborhood system of the Ising model on $\mathbb{C}_3^2$.

system via

$$\mathcal{N}_c = \left\{ (\alpha, \beta) : |\alpha - c_1| + |\beta - c_2| = 1 \right\} \text{ for all } (c_1, c_2) \in \mathbb{C}_M^2.$$

The graph of the neighborhood system is a square lattice and is shown in Fig. 4.3 for $M = 3$.

The nodes $\{z_i\}_{i \in \mathbb{C}_M^2}$ each take values in $\{-1, +1\}$ and the energy function of the Ising model is given by

$$\mathcal{E}(z) = -J \sum_{(i,j) \in E} z_i z_j - H \sum_{i \in \mathbb{C}_M^2} z_i,$$

where $H \in \mathbb{R}$ corresponds to the presence of an “external magnetic field”, $J \in \mathbb{R}$ controls the strength of interaction between neighbors, and E is the edges in the graph induced by the neighborhood system. The first sum is over all pairs of nearest neighbors in the lattice and the second sum is over all lattice sites.

In general, we consider simple cases where $H = 0$ and $J = 1$ (or, sometimes $J = -1$), and denote T as the temperature; then we have a distribution of the form

$$\pi_T(z) = \frac{1}{Z_T} \exp \left\{ \frac{1}{T} \sum_{(i,j) \in E} z_i z_j \right\},$$

where Z_T is the normalizing constant. If $z_i = z_j = 1$ or $z_i = z_j = -1$, then $z_i z_j = 1$. Otherwise, $z_i z_j = -1$, which indicates that the configuration with the lower energy (or high probability) are those where adjacent variables have the same value.

The energy function will take its lowest values at the global minimum (there could be more than one) of $\mathcal{E}$. Thus, the global minimums are the most likely configurations if we draw from $\pi_T(z)$ (so long as $T < \infty$). A large value of T (high temperature) tends to "flatten out" the distribution, making all configurations more or less equally likely, while a small value of T (low temperature) tends to accentuate the probabilities of the lowest energy states [32]. This suggests that if we could sample from the distribution $\pi_T(z)$ with a sufficiently small value of T, then we would obtain a global minimum of $\mathcal{E}$ with high probability.

The Ising model exhibits certain interesting properties. For instance, when the temperature T is lower than the critical temperature T_c, most of the spins are aligned, giving a large total magnetization. On the other hand, as the temperature rises from T_c, spins become more randomly oriented to give a zero total magnetization for small external magnetic field H.

During a Monte Carlo simulation of the Ising model, the magnetization stands as good order parameter to study the phase transitions in the ferromagnetic case. As previously mentioned, this order parameter is zero in a high temperature regime and non-zero in a low temperature regime. As stated before, Ising analytically solved this model in one dimension and showed that there is no phase transition. In one dimension, the magnetization decreases slowly and continuously as the temperature increases. L. Onsager (1944) exactly solved the two dimensional Ising model and revealed that the model exhibits a phase transition at a temperature $T_c = \frac{2J}{\log(1+\sqrt{2})}$

The Metropolis-Hastings algorithm and Gibbs sampling are two common methods to sample from the Ising model. However, as the temperature decreases, the Markov chains generated by either of the these two sampling methods would locally be trapped. On the other hand, the discrete latent slice sampler can simulate all the configurations of the nodes in parallel, rather than having to randomly pick a node and making a proposal, as is in the Metropolis-Hastings algorithm. Fig. 4.4 shows how the temperature parameter changes the output of the Ising model with simulations from the discrete latent slice sampler. Since all the nodes configurations of the square lattice are sampled, we can see the energy function is still exploring its global minimum even at $T = 0.5$. This is not possible for the Metropolis algorithm.

To demonstrate how the latent slice sampler jumps between modes, we compare with Gibbs sampling and the standard Metropolis-Hastings algorithm. For the latent slice sampler, we only need to know the exact form of the numerator of $\pi(z)$, while for the Gibbs sampler we need to derive the full set

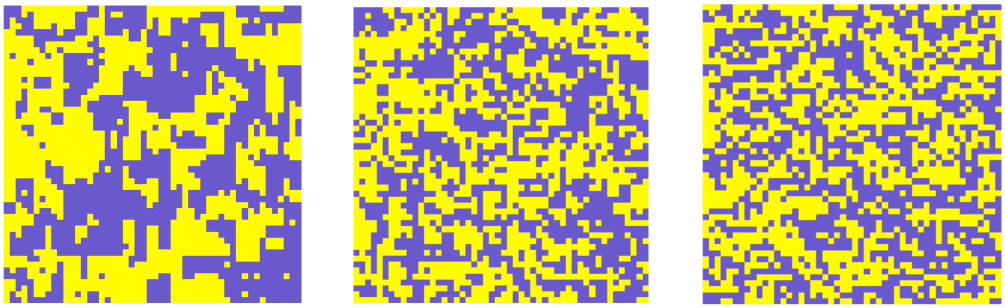


Figure 4.4: The Ising model on the 50×50 square lattice simulated from latent slice sampler. Left to right: $T_1 = 0.5, T_2 = 5, T_3 = 20$.

of posterior conditional distributions for the (z_i); i.e.

$$\begin{aligned}\pi(z_i = +1|z_{-i}) &= \frac{\exp\left\{\frac{1}{T}\sum_{\{j:i\sim j\}} z_j\right\}}{\exp\left\{-\frac{1}{T}\sum_{\{j:i\sim j\}} z_j\right\} + \exp\left\{\frac{1}{T}\{j : i \sim j\} z_j\right\}} \\ &= \frac{1}{2} + \frac{1}{2}\tanh\left(\frac{1}{T}\sum_{\{j:i\sim j\}} z_j\right).\end{aligned}$$

One possible Metropolis algorithm is to flip one randomly chosen vertex of the current configuration of z (i.e. replace 1 with -1 or replace -1 with 1), and use a distribution that is uniform over all such configurations at each step. This gives a proposal z' and the acceptance probability for it is given by

$$\alpha(z, z') = \frac{\exp\left(-\mathcal{E}(z')/T\right)}{\exp\left(-\mathcal{E}(z)/T\right)} = \exp\left\{\frac{\mathcal{E}(z) - \mathcal{E}(z')}{T}\right\}.$$

To illustrate, we take an Ising model on a 25×25 square lattice and a sequence of decreasing temperatures, that is, a cooling sequence. For the latent slice sampler, we take $p(s) \propto se^{-0.02s}$, which ensures almost uniform jumps. The proportion of a randomly selected $z_{(6,21)} = 1$ out of all 10,000 samples

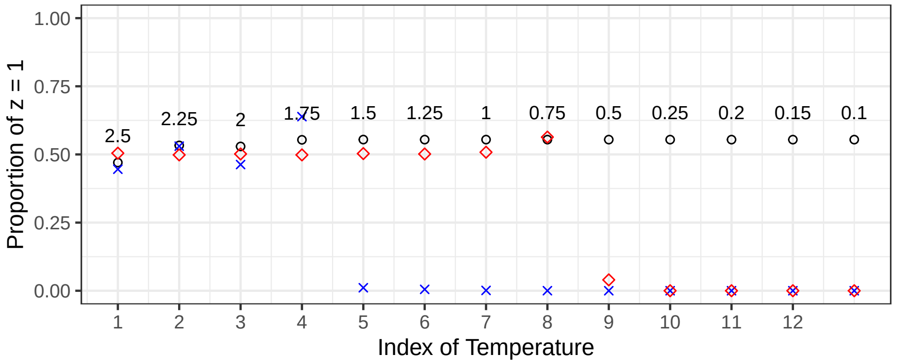


Figure 4.5: The proportion of $z_{(6,21)} = 1$ out of all samples from latent slice sampling (black circle), Gibbs sampling (blue cross), and Metropolis-Hastings algorithm (red diamond) under different temperatures.

from the above three sampling methods under different temperatures is shown in Figure 4.5. The samples are able to jump between $+1$ and -1 when the temperature is higher than 1.5 . The samples from the latent slice sampler still move under the lower temperatures 1, 0.2, and 0.05; while samples from Gibbs sampling do not move when the temperature is lower than 1.75.

In general, Markov Chain will mix faster if they explore the state space quickly. However, most of the algorithms we considered so far, including the latent slice sampler, only make *local* changes at each step (that is, the samples do not change too much from step to step). For example, the algorithm for the Ising model only change the value at one site in each step. It would clearly be better if we could make *global* changes at each step (that is, totally change the value of the Markov chain from one step to the next). The problem, however, is that global changes are very difficult to make. We certainly cannot just change everything at random and try to accept it using Metropolis-Hastings (this would be almost as bad as the acceptance rejection method, which is very bad). Much more sophisticated techniques must be used, like Parallel Tempering [62, 44] and the cluster-update-based Wolff Algorithm [155, 8], together with our either latent slice sampling or Metropolis-Hastings algorithm. More details about the parallel tempering and Wolff algorithm can be found in Appendix B.

4.4 Bayesian Variable Selection

In this subsection we obtain a joint distribution for the variable selection indicators of a linear model. The model is given by

$$y_i = \sum_{j=1}^{p} x_{ij} z_j \beta_j + \sigma \epsilon_i, \tag{4.5}$$

where the $z = (z_j)$ are the indicators, taking the values 0 or 1, and the (ϵ_i) are assumed to be independent standard normal. This model was first proposed in Kuo and Mallick [94] as an alternative framework to the hierarchical model of George and McCulloch [60]. We adopt a slightly different prior set up compared to that of Kuo and Mallick [94]. We write the likelihood, using $\lambda = 1/\sigma^2$, as

$$\lambda^{n/2} \exp\left\{ -\frac{\lambda}{2}(y - X\beta)'(y - X\beta) \right\}$$

where $X = X_0\, Z$ with X_0 the design matrix based on the (x_{ij}) and $Z = \mathrm{diag}(z_j)$. We take a g-prior [157] for β; so for some $g > 0$, $\beta \sim \mathcal{N}\left(0, g\sigma^2 (X'X)^{-1}\right)$. If $Z \equiv 0$ then $\beta = 0$ which is compatible with the idea that no predictors are active. The prior for λ is taken to be gamma with parameters (a, a).

The aim now is to find the marginal posterior distribution for z given the data. This involves some straightforward integration. First

$$p(y \mid \lambda, x, z) \propto \lambda^{n/2+a-1} \mathrm{e}^{-\lambda(a+\frac{1}{2}y'y)} \left(\frac{g}{1+g}\right)^{|z|/2} \mathrm{e}^{-\frac{\lambda}{2(1+g)}y'\,H_X\,y}$$

where H_X is the hat matrix corresponding to the X_0 and Z; effectively removing the columns for which the $z = 0$, and $|z|$ is the number of $\{z = 1\}$. Hence, assuming a uniform prior for z, we get

$$p(z \mid y, x) \propto \left(\frac{g}{1+g}\right)^{|z|/2} \left\{a + \frac{1}{2}y'y + \frac{1}{2}y'\,H_X\,y/(1+g)\right\}^{-a-n/2}$$

On the other hand, Kuo and Mallick [94] employed a MCMC algorithm which worked as a Gibbs sampler and sampled the conditional distributions of β, z and λ.

When the distribution $p(z \mid y, x)$ is unimodal both the latent slice sampler and Metropolis algorithms work well. The latter, using single move proposals, mixing slightly better, though all the marginal probabilities of the (z_j) are estimated exactly the same. To illustrate this we take a sample of size $n = 100$ and $p = 3$, the $\sigma = 1$ and the design matrix elements are taken as independent standard normal. The true value of $\beta = (0.3, -0.3, 0)$. We ran the slice sampling algorithm for 10000 iterations and the means of the sampled indicator variables were $\bar{z}_1 = 0.359, \quad \bar{z}_2 = 0.988, \quad \bar{z}_3 = 0.090$. With the same dataset we ran a Metropolis algorithm also over 10000 iterations. One iteration involves proposing a flip of each indicator variable and the Metropolis accept/reject criterion is used to determine whether the flip occurs or not. The corresponding sampled means are $\bar{z}_1 = 0.359, \quad \bar{z}_2 = 0.984, \quad \bar{z}_3 = 0.097$, which are essentially the same as those from the slice sampler.

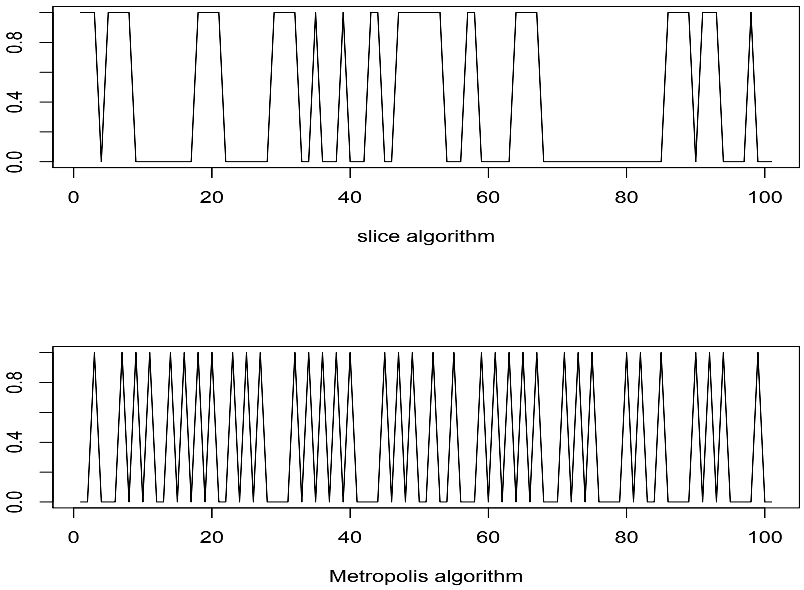


Figure 4.6: Comparison of mixing of slice algorithm and Metropolis algorithm

In Fig. 4.6 we illustrate the sampled indicator variable z_1 for both the slice sampling algorithm (top) and the Metropolis algorithm (bottom) over a period of 100 iterations. It can be seen the Metropolis algorithm mixes better than the slice sampling algorithm. However, in this simple case the local sampler is effective as the distribution of z is well behaved and nicely unimodal. The slice sampling algorithm acts as both a local and global sampler, explaining the differences.

However, when $p(z \mid x, y)$ is bi-modal, the mixing of the slice sampler is superior due to its ability to make large jumps in the z-space. A bimodal distribution can be arranged and can also occur naturally when there is high co-linearity between predictor variables. To describe the experiment, we take

$n = 100$ and $p = 10$, and only one predictor is active, say $x_1 = (x_{11}, \ldots, x_{1p})$, where the (x_{1j}) are taken as independent standard normal. The true $\beta_1 = 5$ and we generate the data with $\sigma = 1$. To create co-linearity we take $x_{2j} = 0.99\, x_{1j} + 0.01\, \xi_j$, with the (ξ_j) as independent standard normal. Hence, the $p(z \mid y, x)$ is bi-modal at $z = (1, 0, 0, \ldots)$ and $(0, 1, 0, \ldots)$ with approximately equal weight for each. Indeed, for both the slice sampling algorithm and the Metropolis algorithm, the mean values of the z_1 and z_2 are 0.59 and 0.47.

However, for the single move Metropolis algorithm, the only way from one mode to the other is to go via $(1, 1, \ldots)$. The probability of this combination is 0.05 and it is this probability which determines the mixing ability of the Metropolis algorithm. For example, over 100 iterations, we would expect 5 switches. This is demonstrated in Fig. 4.7. The bold lines are the z_1 values and the lines in red are the z_2 values. As is seen the number of switches for the Metropolis algorithm is 6, while for the slice sampler it is 13, since for this algorithm the number of switches does not depend on the probability of $p(1, 1, \ldots)$.

If the probability of $p(1, 1, \ldots)$ becomes too small then the ability of the Metropolis algorithm to move between the two modes becomes increasingly improbable. To make this point we take $p = 2$ and the value of g as 10^{-6} with all other settings remaining the same. This makes the $p(1, 1)$ probability very small. The slice sampler chain mixes well and the mean values for z_1 and z_2 are 0.504 and 0.406, respectively. On the other hand, the corresponding values for the Metropolis sampler are 1 and 0, respectively, indicating the chain is fixed at one of the modes. See also the Example 2 in Chapter 4.2.2. The slice sampler can generate effectively uniform proposals in z space which if rejected set up a sequence of proposals contracting back to the current point, which can then accept small local moves. So if the Metropolis chain of only local moves based on flips of a single z are switched to a uniform proposal to solve the bottleneck problem, the inferiority to the slice sampler becomes very apparent in that now the probability of a small move is becoming negligible.

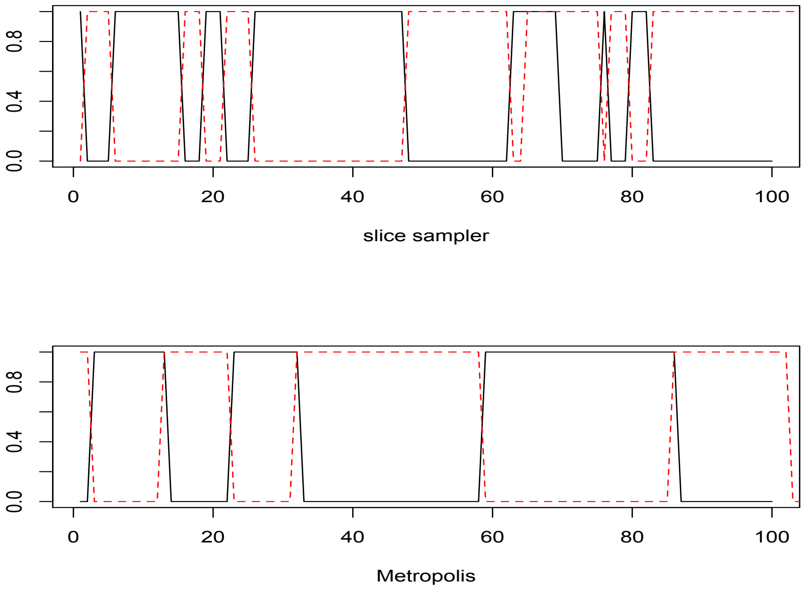


Figure 4.7: Comparison of switching between modes for slice sampler (top) and Metropolis (bottom)

In Chapter 4.5 we discuss the mixing of the two types of chain via transition matrices in z space. By only considering 4 states we can easily compute the second largest eigenvalues of each transition matrix. Generally speaking, the second largest eigenvalue quantifies the mixing of the chain, with smaller eigenvalues corresponding to faster mixing chains. The Metropolis chain has the bottleneck which creates a large second largest eigenvalue, whereas the jumping potential of our latent slice sampler allows the second largest eigenvalue to be small. Though the setting is zooming in on a few states, the problem is going to be the same whatever the overall dimension of the z space is.

Another version of the model incorporates a different prior; specifically the Ising prior, see Li and Zhang [101]. Here the application is relevant to a genomic variable selection model where covariates form a structured sequence. In particular, the Ising prior imposes a Markov dependence between indicator variables. The likelihood component of the model is the usual $y = X\beta + \sigma\epsilon$, though we take, with a slight modification to Li and Zhang [101], the prior to be

$$\beta = \mathcal{N}\left(0, \sigma^2\,(X'X)^{-1}\,D\right) \quad \text{and} \quad D = \operatorname{diag}((1-\gamma_j)v_0^2 + \gamma_j\,v_1^2)$$

with v_0 small and v_1 large. The prior for $\gamma = (\gamma_1, \ldots, \gamma_d)$ is

$$P(\gamma) \propto \exp\{a'\gamma + \gamma' B\gamma\}$$

for some vector a and matrix B. The model is essentially a "g"-prior set up with a d-dimensional g and a "spike and slab" framework. The β can be integrated out leaving

$$P(\gamma \mid y, \sigma^2) \propto \left(\prod_{j=1}^{d} \frac{1}{1+g_j}\right) \exp\left\{\frac{1}{2}\sigma^{-2} y' X \widetilde{D} (X'X)^{-1} X' y\right\} P(\gamma),$$

where $g_j = (1 - \gamma_j)v_0^2 + \gamma_j v_1^2$ and $\widetilde{D} = \mathrm{diag}(g_j/(1 + g_j))$.

Following Li and Zhang [101] we conduct a simulation for which $n = 100$ and $d = 1000$. For our setup we take $\gamma_j = 1$ for $j \in [4, 6]$, with all the rest being 0. The relevant nonzero (β_j) are taken to be all 1 and $\sigma = 1$. The matrix X is comprises independent standard normal random variables while $v_0 = 0.01$ and $v_1 = 10$. Finally, we take a to be the constant vector $\log(0.03/1/12^2)$ and B to be the matrix with entry $\log(5 \times 1.12)$ in position (j, k) for which $|j - k| = 1$. These are values used from Li and Zhang (2010).

We ran the chain for 10000 iterations; assuming the σ is known. The chain took 1 minute to complete all iterations. Our primary aim is to show that the algorithm runs quickly and provides good answers. We monitored the output of the (γ_j) for $j \in [3, 7]$ over the run of the algorithm; taking 10,000 iterations which took 2 minutes to complete (running on R code on a MacBook Pro with 2.2 GHz processor and 16 GB Memory).

We obtained $\widehat{\gamma}_{j=3:7} = (0.002, 1.00, 0.99, 0.99, 0.384)$. This type of result was the norm over multiple runs of the same problem; e.g. $\widehat{\gamma}_{j=3:7} = (0.106, 0.895, 0.999, 0.999, 0.036)$. In short, the support is present for the nonzero β_j for $j = 4 : 6$ while lack of support for the case when the β_j are 0.

The message from this section is that the latent slice sampler demonstrates a robust algorithm. It is adequate for cases where the target distribution is well-behaved, while also having the ability to jump between modes when the target distribution is not well-behaved. Whereas the Metropolis algorithm works well in the simple cases, it fails when modes are separated by regions of very low probability.

4.5 Mixing Properties of Algorithms

In this section we explain why the slice sampling algorithm improves on the Metropolis algorithm, the latter using the flip proposal as described in illustrations in Chapter 4.2, when there is a bottleneck; i.e. regions of very small probability between modes of high probability. To make the comparison we only need to demonstrate with a small number of states.

4.5.1 Eigenvalues

Consider the joint probability mass function $\pi(z_1, z_2)$ with $z_1, z_2 \in \{0, 1\}$ and

$$\pi(0,0) = \pi(1,1) = \frac{1}{2} - \epsilon \quad \text{and} \quad \pi(0,1) = \pi(1,0) = \epsilon,$$

for some small ϵ.

The Metropolis transition matrix obtained from proposing a flip of a z_j, $j = 1, 2$, with probability $\frac{1}{2}$ each is given by

$$P_M = \begin{pmatrix} 1 - \frac{2\epsilon}{1-2\epsilon} & \frac{\epsilon}{1-2\epsilon} & \frac{\epsilon}{1-2\epsilon} & 0 \\ \frac{1}{2} & 0 & 0 & \frac{1}{2} \\ \frac{1}{2} & 0 & 0 & \frac{1}{2} \\ 0 & \frac{\epsilon}{1-2\epsilon} & \frac{\epsilon}{1-2\epsilon} & 1 - \frac{2\epsilon}{1-2\epsilon} \end{pmatrix}.$$

Here row 1 corresponds to $(0,0)$, row 2 to $(0,1)$, row 3 to $(1,0)$ and row 4 to $(1,1)$ with the same ordering for the columns. It is straightforward to confirm

$\pi\, P_M = \pi$.

As can be seen, to arrive at $(1,1)$ from $(0,0)$, for example, the chain must pass through either $(0,1)$ or $(1,0)$, yet the probability of such a move is small. Hence there is a bottleneck separating $(1,1)$ from $(0,0)$. This will hinder convergence of the chain and a measure of this is to use the eigenvalues of P. Indeed, the largest eigenvalue is 1, and the second largest eigenvalue contributes to the convergence rate: the closer it is to 1, the slower the rate. The distance between π and π_k, where π_k is the probability mass function of z at iteration k, depends on λ_2^k, where λ_2 is the second largest eigenvalue of the transition matrix. See for example Diaconis and Strook (1991). The second largest eigenvalue is given by 0.89 when e.g. $\epsilon = 0.05$ and is 0.98 when $\epsilon = 0.01$.

When we consider the transition matrix for the slice sampler, we assume that the proposal is uniform, i.e. for any z the proposal for the new z is uniform over the 4 states. This is based on the idea that the y_j^*, for $j = 1, 2$, is with probability $\frac{1}{2}$ each either negative or positive. For simplicity, we only consider a transition matrix with strictly inferior mixing compared to our slice sampler; specifically, we ignore the multiple proposals possible during the proposed states return to the current state. Such neglect of multiple proposals only occurs for moves from either $(0,0)$ or $(1,1)$ to either $(0,1)$ or $(1,0)$.

For a proposal from e.g. $(0,0)$ to $(0,1)$, the acceptance is based on

$$\epsilon > v\,(\frac{1}{2} - \epsilon)$$

where v is a standard uniform random variable. Hence the probability of acceptance is $2\epsilon/(1-2\epsilon)$; whereas from e.g. $(0,1)$ to $(0,0)$ it is 1, given $\epsilon < \frac{1}{4}$. Hence, the inferior mixing transition matrix for the slice sampler in this case is given by

$$P_S = \begin{pmatrix} \frac{1}{4} + \frac{1}{2}(1 - \frac{2\epsilon}{1-2\epsilon}) & \frac{1}{4}\frac{2\epsilon}{1-2\epsilon} & \frac{1}{4}\frac{2\epsilon}{1-2\epsilon} & \frac{1}{4} \\ \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} \\ \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} \\ \frac{1}{4} & \frac{1}{4}\frac{2\epsilon}{1-2\epsilon} & \frac{1}{4}\frac{2\epsilon}{1-2\epsilon} & \frac{1}{4} + \frac{1}{2}(1 - \frac{2\epsilon}{1-2\epsilon}) \end{pmatrix}.$$

The second largest eigenvalues are given by 0.44 when $\epsilon = 0.05$ and by 0.49 when $\epsilon = 0.01$, so approximately half the values from the Metropolis sampler.

To illustrate the theory, we simulate the slice sampler algorithm with $\epsilon = 0.05$. We take $a = 2$ and $p(s)/s$ to be an exponential density with mean 20. Over a run of 10000 iterations, the estimated transition probability from $(0,0)$ to $(0,1)$ is given by 0.283, whereas the value within P_S is evaluated at 0.0278, which is smaller than the estimated value, yet extremely close to it.

Further, to investigate the assumption of uniform proposals, we recorded the number of first proposals to be 0 for z_1. Out of the 10000 from the run of the chain, 5003 were 0. See also Lemma 1 for theoretical support for uniform proposals.

4.5.2 Effective Sample Size

Another measure of the adequacy of a Markov chain is the number of effective samples generated per evaluation of the likelihood. We consider the two simulation settings of Chapter 4.4: recall that the first (which we

	Easy			Hard		
Method	z_1	z_2	z_3	z_1	z_2	z_3
Slice	0.166	0.178	0.192	0.002	0.002	0.015
Metropolis	0.866	0.776	0.417	0.000	0.000	0.095

Table 4.1: Effective sample size per evaluation of the likelihood function for the two variable selection settings described in Chapter 4.4 for the variables z_1, z_2, z_3. "Easy" refers to the setting with $p = 3$ and independent covariates while "Hard" refers to the setting with $p = 10$ and colinear covariates.

label "Easy") set $\beta = (0.3, -0.3, 0)$, $n = 100$, and $\sigma = 1$ with iid normal covariates, while the second (which we label "Hard") set $\beta = (5, 0, \ldots, 0)$ with $p = 10, \sigma = 1, n = 100$, and x_{2j} chosen to be highly correlated with x_{1j}. The "Hard" setting is more difficult than the "Easy" setting for a Gibbs sampler because the posterior is bimodal and the Gibbs sampler must navigate over a region of very low probability to move from one mode to another.

Results for these simulation settings are given in Table 4.1. We see that, for the "Easy" setting, the Metropolis algorithm (which sweeps over the variables, proposing a transition from 0 to 1 and vice-versa for each variable) performs remarkably well, while our discrete latent slice sampler is less efficient. However we again see the ability of the slice sampler to navigate across modes in a discrete space, as it performs much better than the Metropolis algorithm at sampling the highly-correlated variables z_1 and z_2.

4.6 Discussion

In this chapter we have exploited a slice sampling algorithm in continuous space in order to sample a joint distribution on binary values. Such distributions arise in classic contexts and are known to be problematic to sample when the dimension is large and/or the distribution is multimodal. Our new sampling algorithm works by being able to propose a move to any location from any current location with almost uniform probability. If a large move is not accepted, the sampler reverts to at least a local sampler in any given iteration.

When the distribution is in fact simple, in the sense it is unimodal, the single flip proposal Metropolis chain works well and mixes faster than the latent slice algorithm. However, this hides a couple of important issues. One is that in practice it would not necessarily be known that the distribution is unimodal and so other modes would be left undetected. A further point is that algorithms which have the ability to jump between modes are needed and currently such suitable algorithms are lacking. These new algorithms would also be required to exhibit certain flexibility, which is that local moves can occur if the large jumps get rejected, as a lot of them will be. This is precisely a feature of the latent slice sampler; as the shrinkage proceeds from the initial large move proposal, and if these get rejected, so the proposals become more local to the current point.

A succinct way to describe the performance of the latent slice sampler is that it is robust. It performs well if the distribution is simple, such as being unimodal, yet has the ability to find different modes if they exist.

In Chapter 4.2.1 we demonstrated the accuracy of the latent slice sampler. The example in Chapter 4.2.2 is extreme but makes a point very clearly about the ability of the latent slice sampler to jump between modes and maintain a correct stationary distribution. This is certainly a challenging problem and it is not clear there are even any alternative algorithm capable of achieving this outcome. Chapter 4.3 considered the classic Ising model and again highlights the problem of standard algorithms such as the Metropolis sampler moving satisfactorily for low temperatures. The converse is true for the latent slice sampler and it can still move and maintain adequate mixing for much lower temperatures. In Chapter 4.4 we look at a variable selection problem. In this case, when the problem is regular, also referred to as “easy”, the Metropolis sampler has an advantage over the latent slice sampler. Though as we have previously mentioned, this can be deceptive. For it might not be known that other modes exist. On the other hand, when high co-linearity exists the latent slice sampler outperforms the Metropolis, and with sufficiently high co-linearity the Metropolis could be forced to come to a stop.

Chapter 5

Conclusions and Future Work

5.1 Conclusions

In this thesis, I have summarized my work on a generic Markov chain Monte sampling method – latent slice sampling for discrete variable, continuous variable, and multi-binary variables respectively, and demonstrate the great potential to be a replacement of widely-used Markov chain Monte Carlo sample algorithms. Moreover, the latent slice sampling takes the computational performance and convergence rate of Markov chains into account, which can be used efficiently in high-dimensional data dominated in a vast range of applications.

5.2 Future Work

Future work can go in a number of directions. One is to look at problems involving multivariate distributions with mixed types of variable; e.g. the most simple being a joint distribution on $\{0, 1\} \times \mathbb{R}$. Obviously more complicated cases can be considered including mixture models where in a Bayesian setting there will be a joint distribution on the component indicator variables as well as the component parameters.

Another direction would be to consider an adaptive algorithm and this would be naturally arising via a general version of (4.1) letting the "free" density for s to depend on x; i.e.

$$f(x, w, s, l) = \mathbf{1}(w < \pi(x))\, s^{-1} p(s \mid x)\, \mathbf{1}(x - s/2 < l < x + s/2).$$

The marginal density for x remains as $\pi(x)$. The aim would be to adapt $p(s \mid x)$ as the chain proceeds so to better propose regions of higher probability, such as separated modes.

our work shows promising results on Bayesian CART model and imbalanced data clustering problems, which are potential direction of areas of machine learning [4] and deep learning [99]. Whenever the Metropolis-Hastings or slice sampling algorithms exist, we can substitute with latent slice sampling.

Appendices

Appendix A

Appendix for Latent Slice Sampler of Discrete Variables

A.1 Validity of The Transition Matrix

Suppose we will use latent slice sampling algorithm to sample x' given x. The goal is to prove the following transition density is a valid probability mass function. $\{X_n\}$ is a Markov chain with π being stationary,

$$p_k(x'|x) = \frac{\pi(x')}{k} \sum_{l=\max(x',x)}^{\min(x'+k-1,x+k-1)} \frac{1}{\sum_{j=\max(1,l-k+1)}^{l} \pi(j)} \tag{A.1}$$

where $k > 1$ and $|x' - x| \leq k - 1$. Clearly, it satisfies the detailed balance equation:

$$p_k(x'|x)\pi(x) = p_k(x|x')\pi(x')$$

To prove the summation of the above density $p_k(x'|x)$ is equal to 1, we first notice that the denominator summation depends on the relationship between k and x, x'. Hence, without loss of generality, we assume $x \geq k$ such that the ultimate goal is to prove

$$\sum_{x'=x-k+1}^{x+k-1} p_k(x'|x) = 1$$

which is equivalent to prove

$$S = \sum_{x'=x-k+1}^{x+k-1} \pi(x') \sum_{l=\max(x',x)}^{\min(x'+k-1,x+k-1)} \frac{1}{\sum_{j=\max(1,l-k+1)}^{l} \pi(j)} = k$$

Proof.

$$\begin{aligned}
S &= \sum_{x'=x-k+1}^{x+k-1} \pi(x') \sum_{l=\max(x',x)}^{\min(x'+k-1,x+k-1)} \frac{1}{\sum_{j=\max(1,l-k+1)}^{l} \pi(j)} \\
&= \sum_{x'=x-k+1}^{x} \sum_{l=x}^{x'+k-1} \frac{\pi(x')}{\sum_{j=l-k+1}^{l} \pi(j)} + \sum_{x'=x+1}^{x+k-1} \sum_{l=x'}^{x+k-1} \frac{\pi(x')}{\sum_{j=l-k+1}^{l} \pi(j)} \\
&= \sum_{x'=x-k+1}^{x} \sum_{l=x-k+1}^{x'} \frac{\pi(x')}{\sum_{j=l}^{l+k-1} \pi(j)} + \sum_{x'=x+1}^{x+k-1} \sum_{l=x'}^{x+k-1} \frac{\pi(x')}{\sum_{j=l-k+1}^{l} \pi(j)} \\
&= \underbrace{\frac{\pi(x-k+1)}{\sum_{j=x-k+1}^{x} \pi(j)}}_{A_1\text{:1 term}} + \cdots + \underbrace{\frac{\pi(x)}{\sum_{j=x-k+1}^{x} \pi(j)} + \cdots + \frac{\pi(x)}{\sum_{j=x}^{x+k-1} \pi(j)}}_{A_k\text{:k terms}} \\
&\quad + \underbrace{\frac{\pi(x+1)}{\sum_{j=x-k+2}^{x+1} \pi(j)} + \cdots + \frac{\pi(x+1)}{\sum_{j=x}^{x+k-1} \pi(j)}}_{B_1\text{:k-1 terms}} + \cdots + \underbrace{\frac{\pi(x+k-1)}{\sum_{j=x}^{x+k-1} \pi(j)}}_{B_{k-1}\text{:1 term}} \qquad \text{(A.2)}
\end{aligned}$$

From here, we need to perform summations of (A.2) as follows: summing over all the first terms from A_1 to A_k gives 1, and summing over all the second terms from A_2 to A_k and the first term of B_1 also gives 1; the calculation continues. Summing over all the $(k-1)^{\text{th}}$ terms from A_{k-1} to A_k and all $(k-1-m)^{\text{th}}$ terms from B_m's gives 1 (where $m = 1, \ldots, k-2$), and summing over the k^{th} terms of A_k and all $(k-m)^{\text{th}}$ terms from B_m's gives 1 (where $m = 1, \ldots, k-1$). It is obvious that the total number of 1's is k, which means $S = k$. □

A.2 Derivation of $p(\cdot \mid \cdot)$ in Equation 2.18

The proposal of $p(w_{M'} \mid w_M)$ is given as follows: a) $w_M \to w_{M+1}$, randomly choose w_i and split it into ww_i and $(1-w)w_i$ with $w \sim \mathrm{Beta}(1,1)$ and place them on the original place, then $p(w_{M+1} \mid w_M) = \mathrm{Beta}(w \mid 1,1)/M = \frac{1}{M}$; b) $w_M \to w_{M-1}$, randomly choose w_i from w_M except for the last one and merge it with w_{i+1}, then $p(w_{M-1} \mid w_M) = \mathrm{Beta}\left(\frac{w_i}{w_i+w_{i+1}} \mid 1,1\right)/(M-1) = \frac{1}{M-1}$.

If $M = m_1$, we need to first get $p(w_{m_1+1} \mid w_{m_1}) = \frac{1}{m_1}$, then we have $p(w_{m_1+2} \mid w_{m_1+1}) = \frac{1}{m_1+1}, \ldots$, up to $p(w_{m_2}|w_{m_2-1}) = \frac{1}{m_2-1}$. The product of all the $p(\cdot \mid \cdot)$ is

$$\frac{1}{m_1(m_1+1)\ldots(m_2-2)(m_2-1)} = \frac{(m_1-1)!}{(m_2-1)!},$$

If $m_1 < M < m_2$, we need to get $p(w_{M-1} \mid w_M) = \frac{1}{M-1}$, $p(w_{M-2} \mid w_{M-1}) = \frac{1}{M-2}, \ldots$, up to $p(w_{m_1} \mid w_{m_1+1}) = \frac{1}{m_1}$, and then we have $p(w_{M+1} \mid w_M) = \frac{1}{M}$, $p(w_{M+2} \mid w_{M+1}) = \frac{1}{M+1}, \ldots$, up to $p(w_{m_2} \mid w_{m_2-1}) = \frac{1}{m_2-1}$. The product of all the $p(\cdot \mid \cdot)$ is

$$\frac{1}{(M-1)(M-2)\ldots m_1 \cdot M(M+1)\ldots(m_2-2)(m_2-1)} = \frac{(m_1-1)!}{(m_2-1)!},$$

If $M = m_2$, we need to first get $p(w_{m_2-1} \mid w_{m_2}) = \frac{1}{m_2-1}$, then $p(w_{m_2-2} \mid w_{m_2-1}) = \frac{1}{m_2-2}, \ldots$, up to $p(w_{m_1} \mid w_{m_1+1}) = \frac{1}{m_1}$. The product of all the $p(\cdot \mid \cdot)$ is

$$\frac{1}{(m_2-1)(m_2-2)\ldots(m_1+1)m_1} = \frac{(m_1-1)!}{(m_2-1)!}$$

By summarizing all three possible cases, we therefore derive that the product of $p(\cdot \mid \cdot)$ is equal to a constant $(m_1 - 1)!/(m_2 - 1)!$, where $m_1 = \max(1, M_0 - k + 1)$ and $m_2 = M_0 + k - 1$.

A.3 Metropolis-Hastings Algorithm for Change-point Problem

In the following, we introduce each type of move in detail and calculate individual acceptance probability of the Metropolis updates, respectively, of the change-point problem.

i. The first kind of moves is the height h_j:

   A change to a height is attempted by first choosing one of $h_1, h_2, \ldots, h_k$ randomly, obtaining h_j say, then proposing a change to h'_j such that $\log(h'_j/h_j)$ is uniformly distributed on the interval $[-\frac{1}{2}, \frac{1}{2}]$. Note that $h'_j = e^u h_j$ with $u \sim \text{Uniform}(-\frac{1}{2}, \frac{1}{2})$. The derived transition density $q(h'_j|h_j)$ of this proposing is

   $$q(h'_j|h_j) = \frac{1}{h'_j}, \qquad \text{where } h'_j \in [h_j/\sqrt{e}, \sqrt{e}h_j]$$

   The target density is the posterior distribution which has the form

   $$\pi(h_j) \propto p(y_1, y_2, \ldots, y_n|h_j)p(h_j|\alpha, \beta)$$

   Through calculation, the acceptance probability of the Metropolis-Hastings updates is

$$\begin{aligned}\alpha(h_j, h_j') &= \min\left\{1, \frac{p(y_1, y_2, \dots, y_n|h_j') \cdot p(h_j'|\alpha, \beta) \cdot h_j}{p(y_1, y_2, \dots, y_n|h_j) \cdot p(h_j|\alpha, \beta) \cdot h_j'}\right\} \\ &= \min\left\{1, \mathrm{e}^{(h_j - h_j')(s_{j+1} - s_j + \beta) + (m_j + \alpha)(\log h_j' - \log h_j)}\right\} \qquad \text{(A.3)}\end{aligned}$$

ii. The second kind of moves is the position s_j:

For $\forall s_i \in [s_1, s_{k+1}], i = 2, 3, \dots, k$, s_i are random variable and each s_i is uniformly distributed with density $p(s_i) = \frac{1}{s_{k+1} - s_1}$. Note that they are not independent. The joint distribution of $s_2, s_3, \dots, s_k$ is given by

$$p(s_2, \dots, s_j, \dots, s_k) = \frac{(2k-1)!}{(s_{k+1} - s_1)^{2k-1}} \mathbf{1}_{\{s_1 < s_2 < \dots < s_k < s_{k+1}\}} \prod_{j=1}^{k} (s_{j+1} - s_j)$$

If $s_1, \dots, s_{j-1}, s_{j+1}, \dots, s_{k+1}$ are determined and only s_j is the variable, we can obtain

$$p(s_j|s_{-j}) = \frac{1}{s_{j+1} - s_{j-1}} \mathbf{1}_{\{s_{j-1} < s_j < s_{j+1}\}} (s_j - s_{j-1})(s_{j+1} - s_j)$$

where $s_{-j} = (s_2, \dots, s_{j-1}, s_{j+1}, \dots, s_k)$. The target density is given by

$$\pi(s_j|s_{-j}) \propto p(y_1, y_2, \dots, y_n|s_j) p(s_j|s_{-j})$$

with the transition density $q(s_j'|s_j) = \frac{1}{s_{j+1} - s_{j-1}}$.

The acceptance probability of the Metropolis-Hastings updates is then derived with the form

$$
\begin{aligned}
\alpha(s_j, s'_j) &= \min\left\{1, \frac{p(y_1, y_2, \ldots, y_n|s'_j) \cdot p(s'_j|s_{-j})}{p(y_1, y_2, \ldots, y_n|s_j) \cdot p(s_j|s_{-j})}\right\} \\
&= \min\{1, \exp((h_j - h_{j-1})(s'_j - s_j) + (m'_{j-1} - m_{j-1}) \log h_{j-1} \\
&\qquad + (m'_j - m_j) \log h_j + \log(s'_j - s_{j-1}) + \log(s_{j+1} - s'_j) \\
&\qquad - \log(s_j - s_{j-1}) - \log(s_{j+1} - s_j))\} \qquad \text{(A.4)}
\end{aligned}
$$

iii. The third kind of moves is “birth” steps of k:

Assumed the parameter subspaces $\mathscr{M}_k = \{k\} \times \mathscr{R}^{2k+1}$, where $k \in \mathbb{N}^+$ and $1 \leq k \leq k_{\max}$. k is drawn from Poisson distribution conditioned on $k \leq k_{\max}$. The prior probability is

$$p(k) = e^{-\lambda} \frac{\lambda^{k-1}}{(k-1)!}$$

Denote $\theta^{(k)} = (h_1, \ldots, h_j, \ldots, h_k, s_2, \ldots, s_j, s_{j+1}, \ldots, s_k)$, then the likelihood is

$$p(y_1, y_2, \ldots, y_n|k, \theta^{(k)}) = \exp\left(-\sum_{j=1}^{k} h_j(s_{j+1} - s_j) + \sum_{j=1}^{k} m_j \log h_j\right)$$

For the “birth” steps, $k' = k + 1$. Green first chose a position s^* for the proposed new step, uniformly distributed on $[0, L]$. It is assumed to lie within an existing interval (s_j, s_{j+1}) with probability 1. If accepted, s'_{j+1} will be set to s^*, and $s_{j+1}, s_{j+2}, \ldots, s_k$ will be relabelled as $s'_{j+2}, s'_{j+3}, \ldots, s'_{k+1}$, with corresponding changes to the labelling of step heights. The new heights h'_j, h'_{j+1} are proposed for the step function on the subintervals $[s_j, s^*)$ and $[s^*, s'_{j+2})$ and should be perturbed in a way such that h_j is a compromise between them. To preserve positivity and maintain simplicity in the acceptance ratio calculations, we use s weighted geometric mean for this compromise, so that

$$(s^* - s_j) \log h'_j + (s_{j+1} - s^*) \log h'_{j+1} = (s_{j+1} - s_j) \log h_j$$

and define the perturbation such that

$$\frac{h'_{j+1}}{h'_j} = \frac{1-u}{u}, \qquad \text{where } u \sim \text{Uniform}(0, 1)$$

Then we can derive the form of h'_j and h'_{j+1}:

$$h'_j = h_j \left(\frac{u}{1-u} \right)^{\frac{s_{j+1}-s^*}{s_{j+1}-s_j}}, \qquad h'_{j+1} = h_j \left(\frac{1-u}{u} \right)^{\frac{s^*-s_j}{s_{j+1}-s_j}}$$

The target density here is

$$\pi(k) \propto p(y_1, y_2, \ldots, y_n | k, \theta^{(k)}) p(k, \theta^{(k)})$$

with the transition density $q(k', \theta^{(k')}) = q(u^{(k)}) = q(u) = 1$.

Suppose the probability of choosing move $(k, \theta^{(k)})$ is $j(k, \theta^{(k)})$, then the expression of the acceptance probability of the birth proposal is

$$\alpha = \min\{1, \text{likelihood ratio} \times \text{prior ratio} \times \text{proposal ratio} \times \text{Jacobian}\}$$

Below we listed the computation results of the four terms in the above acceptance probability.

(a). Likelihood ratio:

$$\frac{p(y_1, y_2, \ldots, y_n|k', \theta^{(k')})}{p(y_1, y_2, \ldots, y_n|k, \theta^{(k)})} = \exp\{-h'_j(s^* - s_j) - h'_{j+1}(s_{j+1} - s^*) + h_j(s_{j+1} - s_j) + m'_j \log h'_j + m'_{j+1} \log h'_{j+1} - m_j \log h_j\}$$

(b). Prior ratio:

$$\frac{p(k', \theta^{(k')})}{p(k, \theta^{(k)})} = \frac{2\lambda(2k+1)}{(s_{k+1} - s_1)^2} \frac{(s^* - s_j)(s_{j+1} - s^*)}{s_{j+1} - s_j} \times \frac{\beta^\alpha}{\Gamma(\alpha)} \left(\frac{h'_j h'_{j+1}}{h_j}\right)^{\alpha-1} \exp\{-\beta(h'_j + h'_{j+1} - h_j)\}$$

(c). Proposal ratio:

$$\frac{j(k', \theta^{(k')})}{j(k, \theta^{(k)}) q(u^{(k)})} = \frac{d_k \cdot p(s_{j+1})}{b_{k-1} \cdot p(s^*) \cdot q(u)} = \frac{d_k(s_{k+1} - s_1)}{k b_{k-1}}$$

b_{k-1} means the probability of choosing the move from $\mathscr{M}_k$ to $\mathscr{M}_{k+1}$.

d_k means the probability of choosing the move from $\mathscr{M}_{k+1}$ to $\mathscr{M}_k$.

The subscript of the probability means the number of change-points in the subspace.

(d). Jacobian:

$$\left|\frac{d(\theta^{(k')})}{d(\theta^{(k)}, u^{(k)})}\right| = \frac{(h'_j + h'_{j+1})^2}{h_j}$$

Combining these formulas together, we can get an explicit form of the acceptance probability.

iv. The fourth kind of moves is "death" steps of k:

For the "death" steps, $k' = k - 1$. If the random draw s_{j+1} from $s_2, s_3, \ldots, s_k$ is proposed for removal , the new height over the interval $[s'_j, s'_{j+1})$ is h'_j with the weighted geometric mean satisfying

$$(s_{j+1} - s_j) \log h_j + (s_{j+2} - s_{j+1}) \log h_{j+1} = (s'_{j+1} - s'_j) \log h'_j$$

With the same perturbation $u = h_j/(h_j + h_{j+1})$, we derive the new height h'_j as

$$h'_j = h_j^{\frac{s_{j+1}-s_j}{s_{j+2}-s_j}} \cdot h_{j+1}^{\frac{s_{j+2}-s_{j+1}}{s_{j+2}-s_j}}$$

The acceptance probability for the corresponding death step has the same form with appropriate change of labelling of the variables, and the ratio terms inverted. Below we listed the four terms as the "birth" steps of the acceptance probability.

(a). Likelihood ratio:

$$\begin{aligned} \frac{p(y_1, y_2, \ldots, y_n | k', \theta^{(k')})}{p(y_1, y_2, \ldots, y_n | k, \theta^{(k)})} &= \exp\{-h'_j(s_{j+2} - s_j) + h_j(s_{j+1} - s_j) \\ &\quad + h_{j+1}(s_{j+2} - s_{j+1}) + m'_j \log h'_j \\ &\quad - m_j \log h_j - m_{j+1} \log h_{j+1}\} \end{aligned}$$

(b). Prior ratio:

$$\begin{aligned} \frac{p(k', \theta^{(k')})}{p(k, \theta^{(k)})} &= \frac{(s_{k+1} - s_1)^2}{2\lambda(2k+1)} \frac{(s_{j+2} - s_j)}{(s_{j+1} - s_j)(s_{j+2} - s_{j+1})} \\ &\quad \times \frac{\Gamma(\alpha)}{\beta^\alpha} \left(\frac{h'_j}{h_j h_{j+1}} \right)^{\alpha - 1} \exp\{\beta(h_j + h_{j+1} - h'_j)\} \end{aligned}$$

(c). Proposal ratio:

$$\frac{j(k',\theta^{(k')})q(u^{(k')})}{j(k,\theta^{(k)})} = \frac{b_{k-2}\cdot p(s^{*})\cdot q(u)}{d_{k-1}\cdot p(s_{j+1})} = \frac{kb_{k-2}}{d_k(s_{k+1}-s_1)}$$

d_{k-1} means the probability of choosing the move from $\mathscr{M}_k$ to $\mathscr{M}_{k-1}$.

b_{k-2} means the probability of choosing the move from $\mathscr{M}_{k-1}$ to $\mathscr{M}_k$.

(d). Jacobian:

$$\left|\frac{d(\theta^{(k')},u^{(k')})}{d(\theta^{(k)})}\right| = \frac{h_j'}{(h_j+h_{j+1})^2}$$

Appendix B

Appendix for Multivariate Binary Sampling

B.1 Slice Sampling Algorithm

Algorithm 1: Univariate slice sampling algorithm for variable x

```
1. Sample $w \sim \text{Uniform}(0, f(x_0))$
2. Sample $x_1 \sim$ Uniform on $S = \{x : f(x) > w\}$
```

B.2 Proof of Theorem 4

Theorem 4. *The shrinking procedure defined by (4.3) and (4.4) defines a Markov transition function $Q(y \mid y', w, s, l)$ which is* reversible *in the sense that $f(y \mid w, s, l)\, Q(y' \mid y, w, s, l) = f(y' \mid w, s, l)\, Q(y \mid y', w, s, l)$.*

Proof. For simplicity, we will consider only the case of univariate y; the proof for multivariate y is essentially the same. Also, we suppress dependence of Q on w, s, l to lighten notation. First, we note that $f(y \mid w, s, l)$ is uniform on the set $\{y : w \le \pi(y), l \in [y - s/2, y + s/2], |y| \le a\}$. If either y or y' are outside of this set, we will have $f(y \mid w, s, l)\, Q(y' \mid y) = f(y' \mid w, s, l)\, Q(y \mid y') = 0$ trivially, so assume without loss-of-generality that this is not the case.

Following Neal (2003), we let $r = (r_1, \dots, r_J)$ denote the sequence of rejected points in the shrinking procedure; by Lemma 4.1.3, r is a random vector of finite length. Let $Q(y', r \mid y)$ denote the transition density of moving from y to y' via the intermediate rejected points r; formally, $Q(y', r \mid y)$ is a density with respect to $dy \times \sum_{j=0}^{\infty} \lambda_j(dr)\ I(J = j)$ where λ_j denotes Lebesgue measure on $\mathbb{R}^J$. To show reversibility, it suffices to establish the stronger result that $Q(y', r \mid y) = Q(y, r \mid y')$ for all r. To show $Q(y', r \mid y) = Q(y, r \mid y')$, we first consider the case that some r_j lies in between y and y'. In this case, the shrinking procedure starting from y will eliminate y' as a potential value, and vice-versa. Hence $Q(y', r \mid y) = Q(y, r \mid y') = 0$ in this case. Otherwise, by the uniformity of the sampling, we have $Q(y, r \mid y') = Q(y', r \mid y) = \prod_{j=0}^{J}(b_j - a_j)^{-1}$ where (a_0, b_0) is the starting interval, (a_1, b_1) is the interval after rejecting the joint r_1, and so forth. Hence $Q(y, r \mid y') = Q(y', r \mid y)$.

The logic behind extending this proof to the multivariate setting is essentially the same: we again introduce the set of intermediate moves r, where it will only be possible to transition from y to y' if none of the rejected proposed points $y_j^\star$ for coordinate j lies in between y_j and y'_j, and in this case the probability of transitioning from y to y' via r is the same as transitioning from y to y' via r by uniformity. □

B.3 Parallel Tempering and Wolff Algorithm

B.3.1 Parallel tempering

In the Ising model, the most likely configurations are all 1's and all −1's. Configurations that are mixture of 1's and −1's have much lower probabilities. Generally, it is hard to make samplers that can escape from high probability region, travel quickly through low energy regions, and find other high energy regions. The idea of parallel tempering is to use multiple Markov chains to accomplish this.

In parallel tempering, we suppose we want to sample from the Boltzmann distribution of the form

$$\pi_T(z) = \frac{1}{Z_T} \exp\left\{\frac{1}{T}\mathcal{E}(z)\right\}, \tag{B.1}$$

With a little bit of work, almost any distribution can be written in this form. The basic idea is then to run multiple Markov chains, each at a different temperature, with the Markov chain with the lowest temperature having stationary distribution π_T. The Markov chains running at high temperatures will quickly explore the state space, while the Markov chains at lower temperatures will mainly move in small regions of high probability. Periodically, we will switch the values of two of the Markov chains using a Metropolis move (so that the stationary distributions of the Markov chains are preserved). In this way, the Markov chains with lower temperatures can 'teleport' from one region of high probability to another. More formally, the setup is as follows. We consider a sequence of K temperatures

$$T = T_1 < T_2 < \cdots < T_K$$

and define associated Boltzmann distributions $\pi_{T1}, \ldots, \pi_{T_K}$. We then define a Markov chain, $\{(Z_n^{(1)}, \ldots, Z_n^{(K)})\}_{n\in\mathbb{N}}$ with stationary distribution $\pi(z_1, \ldots, z_K) = \pi_{T_1}(z_1) \ldots \pi_{T_K}(z_K)$. Note that the projection of the first coordinate of this

Markov chain is a Markov chain, $\{Z_n^{(1)}\}_{n\in\mathbb{N}}$, with stationary distribution π_T. We can simulate the Markov chain $\{(Z_n^{(1)}, \ldots, Z_n^{(K)})\}_{n\in\mathbb{N}}$ by running separate MCMC samplers for each of its components.

B.3.2 Wolff Algorithm

We now have a few options for simulating the Ising model, however they are by no means perfect. The issue still remains of falling into local optima instead of a global optima. From our previous mathematical study we know that energy minima for the Ising model are "far away" from each other, that is they have very little overlapping spins. By flipping individual spins 1 by 1, it is very hard to make the chains explore the energy landscape fully. The natural way to solve this is to flip multiple spins simultaneously at each step. From the general definition of the Metropolis-Hastings method there is nothing stopping us in following this line of reasoning.

Unfortunately this makes things much harder, the complications arise in finding a valid scheme for flipping multiple spins at once. We have glossed over the mathematical foundations of MCMC here but the proposal/acceptance probabilities need to be selected in "smart way" in order for the resulting Markov chain to have certain properties. When looking at more than 1 spin at a time in the Ising model this proved fairly difficult. This is evidenced by the original Metropolis-Hastings scheme being proposed in 1953 yet the first multi-spin method not being proposed and justified until the late 1980s.

The main idea of the algorithm is to look for "clusters" of spin sites with the same spin. We then decide to flip the spin of all the sites within this cluster at once. We then pick a new cluster and repeat this process as necessary. The pseudocode for this algorithm as it applies to the Ising model is:

i. A site i with spin z_i is selected at random and added to an empty cluster.

ii. For each neighbour j of i such that $z_i = z_j$, we add j to the cluster stack with probability $p_{ij} = 1 - \exp(-2\beta J)$, else move onto next neighbor.

iii. After all neighbours are exhausted, select next site in the cluster stack and repeat the previous step until the cluster stack is exhausted.

iv. Once the cluster is fully specified, flip the spins of all sites in the cluster and begin again.

We can see that like the Gibbs sampling algorithm, here the Wolff algorithm is “rejection free”, that is, all proposed sites are flipped. We also note that there is nothing in this method that is incompatible with simulated annealing/tempering - these techniques are often used together.

The Wolff algorithm does not “converge” to a low energy state, instead it samples from the entire space in a “smart way” - even if it finds itself in the global energy minima there is still a relatively high probability of escaping. If we were interested in finding a ground state we could keep track of the configuration corresponding to the lowest observed energy state.

Bibliography

[1] James H. Albert and Siddhartha Chib. Bayesian analysis of binary and polychotomous response data. *Journal of the American Statistical Association*, 88(422):669–679, 1993.

[2] Yves F. Atchadé andJeffrey S. Rosenthal. On adaptive markov chain monte carlo algorithms. *Bernoulli*, 11(5):815–828, 2005.

[3] Christophe Andrieu and Johannes Thoms. A tutorial on adaptive mcmc. *Statistics and Computing*, 18:343–373, 2008.

[4] Christophe Andrieu, Nando de Freitas, Arnaud Doucet, and Michael I. Jordan. An introduction to mcmc for machine learning. *Journal of the American Statistical Association*, 50:5–43, 2003.

[5] Christophe Andrieu, Arnaud Doucet, and Roman Holenstein. Particle Markov chain Monte Carlo methods. *Journal of the Royal Statistical Society. Series B*, 72:269–342, 2010.

[6] Ming-Hui Chen ands Qi-Man Shao and Joseph G. Ibrahim. *Monte Carlo Methods in Bayesian Computation.* Springer, New York, 2000.

[7] Krishna B. Athreya, Hani Doss, and Jayaram Sethuraman. On the convergence of the markov chain simulation method. *The Annals of Statistics*, 24(1):69–100, 1996.

[8] S. Bae, S. H. Ko, and P. D. Coddington. Parallel wolff cluster algorithms. *International Journal of Modern Physics C*, 6(2):197–210, 1995.

[9] A. A. Barker. Monte carlo calculations of the radial distribution functions for a proton electron plasma. *Australian Journal of Physics*, 18:119–133, 1965.

[10] Kasper K. Berthelsen and Jesper Møller. Likelihood and non-parametric bayesian mcmc inference for spatial point processes based on perfect simulation and path sampling. *Scandinavian Journal of Statistics*, 30(3): 549–564, 2003.

[11] Julian Besag. Spatial interaction and the statistical analysis of lattice systems. *Journal of the Royal Statistical Society. Series B*, 36(2):192–236, 1974.

[12] Julian Besag. On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society. Series B*, 48(3):259–302, 1986.

[13] Julian Besag and Peter Clifford. Generalized monte carlo significance tests. *Biometrika*, 76(4):633–642, 1989.

[14] Julian Besag and Peter J. Green. Spatial statistics and bayesian computation. *Journal of the Royal Statistical Society. Series B*, 55(1):25–37, 1993.

[15] Julian Besag, Jeremy York, and Annie Mollié. Bayesian image restoration, with two applications in spatial statistics (with discussion). *Annals of the Institute of Statistical Mathematics*, 43:1–20, 1991.

[16] Richard Blundell, Amanda Gosling, Hidehiko Ichimura, and Costas Meghir. Changes in the distribution of male and female wages accounting for employment composition using bounds. *Econometrica*, 75(2):323–363, 2007.

[17] Leonard Bottolo and Sylvia Richardson. Evolutionary stochastic search for bayesian model exploration. *Bayesian Analysis*, 5(3):583–618, 2010.

[18] Alan E. Gelfand Bradley P. Carlin and Adrian F. M. Smith. Hierarchical bayesian analysis of change point problems. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 41(2):389–405, 1992.

[19] Stephen P. Brooks, Paolo Giudici, and Gareth O. Roberts. Efficient construction of reversible jump markov chain monte carlo proposal distributions. *Journal of the Royal Statistical Society. Series B*, 65(1):3–39, 2003.

[20] Olivier Cappé, Randal Douc, Arnaud Guillin, Jean-Michel Marin, and Christian P. Robert. Adaptive importance sampling in general mixture classes. *Statistics and Computing*, 18:448–459, 2008.

[21] Olivier Cappér, Christian P. Robert, and Tobias Rydén. Reversible jump, birth-and-death and more general continuous time markov chain monte

carlo samplers. *Journal of the Royal Statistical Society. Series B*, 65(3): 679–700, 2003.

[22] Bradley P. Carlin and Siddhartha Chib. Bayesian model choice through markov chain monte carlo. *Journal of the Royal Statistical Society. Series B*, 57(3):473–484, 1994.

[23] James Carpenter, Peter Cliffordy, and Paul Fearnhead. Building robust simulation-based filters for evolving datasets. *Journal of Computational and Graphical Statistics*, Technical report, Dept. Statistics, Oxford Univ., 1997.

[24] George Casell, Michael Lavine, and Christian P. Robert. Explaining the perfect sampler. *The American Statistician*, 55:299–305, 2001.

[25] George Casella and Edward I. George. Explaining the gibbs sampler. *The American Statistician*, 46(3):167–174, 1992.

[26] G. Celeux and J. Diebolt. The sem algorithm: A probabilistic teacher algorithm derived from the em algorithm for the mixture problem. *Computational Statistics Quarterly*, 2:73–82, 1985.

[27] G. Celeux and J. Diebolt. Une version de type recuit simulé de l'algorithme em. *Comptes rendus de l'Académie des Sciences. I Math*, 310:119–124, 1990.

[28] Yuansi Chen, Raaz Dwivedi, Martin J. Wainwright, and Bin Yu. Fast MCMC sampling algorithms on polytopes. *Journal of Machine Learning Research*, 19(55):1–86, 2018.

[29] Siddhartha Chib and Edward Greenberg. Understanding the metropolis-hastings algorithm. *The American Statistician*, 49(4):327–335, 1995.

[30] Nicolas Chopin Christian Schäfer. Sequential monte carlo on large binary sampling spaces. *Statistics and Computing*, 23(2), 2011.

[31] Gary A. Churchill. Accurate restoration of dna sequences. *Biometrics Unit Technical Reports*, 2:90–148, 1995.

[32] Barry A. Cipra. An introduction to the ising model. *The American Mathematical Monthly*, 94(10):937–959, 1987.

[33] Merlise A. Clyde, Joyee Ghosh, and Michael L. Littman. Bayesian adaptive sampling for variable selection and model averaging. *Journal of Computational and Graphical Statistics*, 20(1):80–101, 2011.

[34] Paul Damien and Stephen G. Walker. Sampling truncated Normal, Beta, and Gamma densities. *Journal of Computational and Graphical Statistics*, 10(2):206–215, 2001.

[35] Paul Damien, Jon C. Wakefield, and Stephen G. Walker. Gibbs sampling for Bayesian nonconjugate and hierarchical models using auxiliary variables. *Journal of the Royal Statistical Society, Series B*, 61(2):331–344, 1999.

[36] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm (with discussion). *Journal of the Royal Statistical Society. Series B*, 39(1):1–38, 1977.

[37] Jean Diebolt and Christian P. Robert. Estimation of finite mixture distributions through bayesian sampling. *Journal of the Royal Statistical Society. Series B*, 56(2):363–375, 1994.

[38] Xeni K. Dimakos. A guide to exact simulation. *International Statistical Review*, 69(1):27–48, 2001.

[39] Arnaud Doucet, Simon Godsill, and Christophe Andrieu. On sequential monte carlo sampling methods for bayesian filtering. *Statistics and Computing*, 10:197–208, 2000.

[40] Arnaud Doucet, Nando de Freitas, and Neil Gordon. *Sequential Monte Carlo methods in practice.* Springer, New York, 2001.

[41] Jerome A. Dupuis. Bayesian estimation of movement and survival probabilities from capture–recapture data. *Biometrika*, 82(4):761–772, 1995.

[42] J. R. Ehrman, L. D. Fosdick, and D. C. Handscomb. Computation of order parameters in an ising lattice by the monte carlo method. *Journal of Mathematical Physics*, 1(6):547–558, 1960.

[43] Tahir Ekin, Stephen G. Walker, and Paul Damien. Augmented simulation methods for discrete stochastic optimization with recourse. *Annals of Operations Research*, 2020.

[44] Marco Falcioni and Michael W. Deem. A biased monte carlo scheme for zeolite structure solution. *Journal of Chemical Physics*, 110:1754–1766, 1999.

[45] Thomas S. Ferguson. A bayesian analysis of some nonparametric problems. *The Annals of Statistics*, 1(2):209–230, 1973.

[46] Pablo A. Ferraria, Roberto Fernández, and Nancy L. Garcia. Perfect simulation for interacting point processes, loss networks and ising models. *Stochastic Processes and their Applications*, 102(1):63–88, 2002.

[47] James A. Fill. An interruptible algorithm for perfect sampling via markov chains. *Annals of Applied Probability*, 8(1):131–162, 1998.

[48] James A. Fill. The move-to-front rule: A case study for two perfect sampling algorithms. *Probability in the Engineering and Informational Sciences*, 12(3):283–302, 1998.

[49] A. Gelman G. O. Roberts and W. R. Gilks. Weak convergence and optimal scaling of random walk metropolis algorithms. *The Annals of Applied Probability*, 7(1):110–120, 1997.

[50] A. E. Gelfand and D. K. Dey. Bayesian model choice: Asymptotics and exact calculations. *Journal of the Royal Statistical Society. Series B*, 56 (3):501–514, 1994.

[51] Alan E. Gelfand and Adrian F. M. Smith. Sampling-based approaches to calculating marginal densities. *Journal of the American Statistical Association*, 85(410):398–409, 1990.

[52] Alan E. Gelfand and Adrian F. M. Smith. Sampling-based approaches to calculating marginal densities. *Journal of the American Statistical Association*, 85(410):398–409, 1990.

[53] Alan E. Gelfand, Susan E. Hills, Amy Racine-Poon, and Adrian F. M. Smith. Illustration of bayesian inference in normal data models using gibbs sampling. *Journal of the American Statistical Association*, 85(412): 972–985, 1990.

[54] Alan E. Gelfand, Adrian F. M. Smith, and Tai-Ming Lee. Bayesian analysis of constrained parameter and truncated data problems using gibbs sampling. *American Statistical Association*, 87(418):523–532, 1992.

[55] Andrew Gelman and Donald B. Rubin. Inference from iterative simulation using multiple sequences. *Statistical Science*, 7(4):457–472, 1992.

[56] Andrew Gelman, John B. Carlin, Hal S. Stern, David B. Dunson, Aki Vehtari, and Donald B. Rubin. *Bayesian Data Analysis, 3rd Edition*. Chapman & Hall/CRC, New York, NY, 2013.

[57] Andrew Gelman, Jessica Hwang, and Aki Vehtari. Understanding predictive information criteria for bayesian models. *Statistics and Computing*, 24:997–1016, 2014.

[58] Andrew Gelman, Aki Vehtari, John Carlin, Hal S. Stern, David Dunson, and Donald Rubin. *Bayesian Data Analysis.* Chapman & Hall/CRC Texts in Statistical Science, 3rd edition edition, 2014.

[59] Stuart Geman and Donald Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of image. *IEEE Transactions on Pattern Analysis and Mathematical Intelligence*, 6:721–741, 1984.

[60] Edward I. George and Robert E. McCulloch. Variable selection via Gibbs sampling. *Journal of the American Statistical Association*, 88(423): 881–889, 1993.

[61] Edward I. George and Christian P. Robert. Capture–recapture estimation via gibbs sampling. *Biometrika*, 79(4):667–683, 1992.

[62] Charles J. Geyer. Markov chain monte carlo maximum likelihood. *Computing Science and Statistics: Proceedings of the 23rd Symposium on the Interface*, pages 156–163, 1991.

[63] Charles J. Geyer. Practical markov chain monte carlo. *Statistical Science*, 7(4):473–483, 1992.

[64] Charles J. Geyer and Jesper Møller. Simulation procedures and likelihood inference for spatial point processes. *Scandinavian Journal of Statistics*, 21(4):359–373, 1994.

[65] Charles J. Geyer and Elizabeth A. Thompson. Annealing markov chain monte carlo with applications to ancestral inference. *Journal of the American Statistical Association*, 90(431):909–920, 1995.

[66] W. R. Gilks and P. Wild. Adaptive rejection sampling for gibbs sampling. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 41 (2):337–348, 1992.

[67] Walter R. Gilks and Carlo Berzuini. Following a moving target—monte carlo inference for dynamic bayesian models. *Journal of the Royal Statistical Society. Series B*, 63(1):127–146, 2001.

[68] Simon J. Godsill. On the relationship between Markov chain Monte Carlo methods for model uncertainty. *Journal of Computational and Graphical Statistics*, 10(2):230–248, 2001.

[69] N. Gordon, D. Salmond, and Adrian F. M. Smith. A novel approach to non-linear/non-gaussian bayesian state estimation. *IEEE Proceedings on Radar and Signal Processing*, 140:107–113, 1993.

[70] Peter J. Green. Reversible jump markov chain monte carlo computation and bayesian model determination. *Biometrika*, 82(4):711–732, 1995.

[71] Ulf Grenander and Michael I. Mille. Representations of knowledge in complex systems. *Journal of the Royal Statistical Society. Series B*, 56 (4):549–581, 1994.

[72] J. M. Hammersley and D. C. Handscomb. *Monte Carlo methods*. Wiley, New York, 1964.

[73] J. M. Hammersley and K. W. Morton. Poor man's monte carlo. *Journal of the Royal Statistical Society. Series B*, 16(1):23–38, 1954.

[74] John M. Hammersley. Discussion of mr besag's paper. *Journal of the Royal Statistical Society. Series B*, 36:230–231, 1974.

[75] Firas Hamze, Ziyu Wang, and Nando de Freitas. Self-avoiding random dynamics on integer complex systems. *ACM Transactions on Modeling and Computer Simulation*, 23(1):1–25, 2013.

[76] J. E. Handschin and D. Mayne. Monte carlo techniques to estimate the conditional expectation in multi-stage non-linear filtering. *International Journal of Control*, 9:547–559, 1969.

[77] David I. Hastie, Silvia Liverani, and Sylvia Richardson. Sampling from dirichlet process mixture models with unknown concentration parameter: mixing issues in large data implementations. *Statistics and Computing*, 25:1023–1037, 2015.

[78] Wilfred K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.

[79] David M. Higdon. Auxiliary variable methods for markov chain monte carlo with applications. *Journal of the American Statistical Association*, 93(442):585–595, 1998.

[80] James P. Hobert and George Casella. The effect of improper priors on gibbs sampling in hierarchical linear mixed models. *Journal of the American Statistical Association*, 91(436):1461–1473, 1996.

[81] James P. Hobert and Dobrin Marchev. A theoretical comparison of the data augmentation, marginal augmentation and px-da algorithms. *The Annals of Statistics*, 36(2):532–554, 2008.

[82] John Hull and Alan White. The pricing of options on assets with stochastic volatilities. *The Journal of Finance*, 42(2):281–300, 1987.

[83] Hemant Ishwaran and Lancelot F. James. Gibbs sampling methods for stick-breaking priors. *Journal of the American Statistical Association*, 96 (453):161–173, 2001.

[84] Hemant Ishwaran and J. Sunil Rao. Spike and slab variable selection: Frequentist and Bayesian strategies. *Annals of Statistics*, 33(2):730–773, 2005.

[85] Hemant Ishwaran, Udaya B. Kogalur, and J. Sunil Rao. spikeslab: Prediction and variable selection using spike and slab regression. *The R Journal*, 2(2):68–73, 2010.

[86] Rasmus P Waagepetersen Jesper Møller. *Statistical Inference and Simulation for Spatial Point Processes.* Chapman and Hall/CRC, Boca Raton, 2003.

[87] Wing Hung Wong Jun S. Liu and Augustine Kong. Covariance structure of the gibbs sampler with applications to the comparisons of estimators and augmentation schemes. *Biometrika*, 81(1):27–40, 1994.

[88] Wing Hung Wong Jun S. Liu and Augustine Kong. Covariance structure and convergence rate of the gibbs sampler with various scans. *Journal of the Royal Statistical Society. Series B*, 57(1):157–169, 1995.

[89] Maria Kalli, Jim E. Griffin, and Stephen G. Walker. Slice sampling mixture models. *Statistics and Computing*, 21:93–105, 2011.

[90] Minas Karamanis and Florian Beutler. Ensemble slice sampling. *arXiv e-prints*, art. arXiv:2002.06212, 2020.

[91] Wilfrid S. Kendall and Jesper Møller. Perfect simulation using dominating processes on ordered spaces, with application to locally stable point processes. *Advances in Applied Probability*, 32(3):844–865, 2000.

[92] Scott Kirkpatrick, Daniel Gelatt, and Mario P. Vecchi. Optimization by simulated annealing. *American Association for the Advancement of Science, New Series*, 220(4598):671–680, 1983.

[93] Augustine Kong, Jun S. Liu, and Wing Hung Wong. Sequential imputations and bayesian missing data problems. *Journal of the American Statistical Association*, 89(425):278–288, 1994.

[94] Lynn Kuo and Bani Mallick. Variable selection for regression models. *The Indian Journal of Statistics, Series*, 60(1):65–81, 1998.

[95] Demetris Lamnisos, Jim E. Griffin, and Mark F. J. Steel. Adaptive monte carlo for bayesian variable selection in regression models. *Journal of Computational and Graphical Statistics*, 22(3):729–748, 2013.

[96] Nicholas Lange, Bradley P. Carlin, and Alan E. Gelfand. Hierarchical bayes models for the progression of hiv infection using longitudinal cd4 t-cell numbers. *Journal of the American Statistical Association*, 87(419): 615–626, 1992.

[97] Krzysztof Latuszynski and Daniel Rudolf. Convergence of hybrid slice sampling via spectral gap. *arXiv preprint arXiv:1409.2709*, 2014.

[98] Charles E. Lawrence, Stephen F. Altschul, Mark S. Boguski, Jun S. Liu, Andrew F. Neuwald, and John C. Wootton. Detecting subtle sequence signals: A gibbs sampling strategy for multiple alignment. *Science*, 262: 208–214, 1993.

[99] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521:436–444, 2015.

[100] Anthony Lee, Christopher Yau, Michael B. Giles, Arnaud Doucet, and Christopher C. Holmes. On the utility of graphics to perform massively parallel simulation of advanced monte carlo methods. *Journal of Computational and Graphical Statistics*, 19(4):769–789, 2010.

[101] Fan Li and Nancy R. Zhang. Bayesian variable selection in structured

high–dimensional covariate spaces with applications in genomics. *Journal of the American Statistical Association*, 105(491):1202–1214, 2010.

[102] Yanxin Li and Stephen G. Walker. A latent slice sampling algorithm. *Preprint: https://arxiv.org/abs/2010.08509*, 2021.

[103] Jun S. Liu and Rong Chen. Blind deconvolution via sequential imputations. *Journal of the American Statistical Association*, 90(430):567–2576, 1995.

[104] Jun S. Liu and Ying Nian Wu. Parameter expansion for data augmentation. *Journal of the American Statistical Association*, 94(448):1264–1274, 1999.

[105] Albert Y. Lo. On a class of bayesian nonparametric estimates: I. density estimates. *The Annals of Statistics*, 12(1):351–357, 1984.

[106] Scott M. Lynch. *Introduction to Applied Bayesian Statistics and Estimation for Social Scientists*. Springer, New York, 2007.

[107] Neal Madras and Gordon Slade. *The self-avoiding walk*. Birkhäuser, Boston, MA, 1993.

[108] Jean-Michel Marin and Christian P. Robert. *Bayesian Core: A Practical Approach to Computational Bayesian Statistics*. Springer, New York, 2007.

[109] Xiao-Li Meng and David A. Van Dyk. Seeking efficient data augmentation schemes via conditional and marginal augmentation. *Biometrika*, 86(2): 301–320, 1999.

[110] K. L. Mengersen and R. L. Tweedie. Rates of convergence of the hastings and metropolis algorithms. *The Annals of Statistics*, 24(1):101–121, 1996.

[111] Nicholas Metropolis and Stan Ulam. The monte carlo method. *Journal of the American Statistical Association*, 44(247):335–341, 1949.

[112] Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. Equation of state calculations by fast computing machines. *Journal of Chemical Physics*, 21(6):1087–1092, 1953.

[113] A. Mira, J. Moller, and G. O. Roberts. Perfect slice samplers. *Journal of the Royal Statistical Society. Series B*, 63(3):593–606, 2001.

[114] Antonietta M. Mira and Luke Tierney. Efficiency and convergence properties of slice samplers. *Scandinavian Journal of Statistics*, 29:1–12, 2002.

[115] Pierre Del Moral, Arnaud Doucet, and Ajay Jasrau. Sequential monte carlo samplers. *Journal of the Royal Statistical Society. Series B*, 68(3): 411–436, 2006.

[116] Iain Murray, Ryan P. Adams, and David J.C. MacKay. Elliptical slice sampling. *The Proceedings of the 13th International Conference on Artificial Intelligence and Statistics*, 9:541–548, 2010.

[117] Naveen N. Narisetty and Xuming He. Bayesian variable selection with shrinking and diffusing priors. *Annals of Statistics*, 42(2):789–817, 2014.

[118] Radford M. Neal. Sampling from multimodal distributions using tempered transitions. *Statistics and Computing*, 6:353–366, 1995.

[119] Radford M. Neal. Slice sampling. *Annals of Statistics*, 31(3):705–767, 2003.

[120] Robert Nishihara, Iain Murray, and Ryan P. Adams. Parallel MCMC with generalized elliptical slice sampling. *Journal of Machine Learning Research*, 15(61):2087–2112, 2014.

[121] David J. Nott and Robert Kohn. Adaptive sampling for bayesian variable selection. *Biometrika*, 92(4):747–763, 2005.

[122] David J. Nott and Daniela Leonte. Sampling schemes for bayesian variable selection in generalized linear models. *Journal of Computational and Graphical Statistics*, 13(2):362–382, 2004.

[123] Omiros Papaspiliopoulos and Gareth O. Roberts. Sampling from dirichlet process mixture models with unknown concentration parameter: mixing issues in large data implementations. *Biometrika*, 95(1):169–186, 2008.

[124] Peter H. Peskun. Optimum monte carlo sampling using markov chains. *Biometrika*, 60(3):607–612, 1973.

[125] Peter H. Peskun. Guidelines for choosing the transition matrix in monte carlo methods using markov chains. *Journal of Computational Physics*, 40(2):327–344, 1981.

[126] Divid B. Phillips and Andrian F.M. Smith. Bayesian model comparison via jump diffusions. In *Markov Chain Monte Carlo in Practice*, pages 215–239. Chapman & Hall/CRC, 1995.

[127] James G. Propp and David B. Wilson. Exact sampling with coupled markov chains and applications to statistical mechanics. *In Proceedings of the Seventh International Conference on Random Structures and Algorithms (Atlanta, GA*, 9:223–252, 1996.

[128] W. Qian and D.M.Titterington. Parameter estimation for hidden gibbs chains. *Statistics & Probability Letters*, 10(1):49–58, 1990.

[129] Adrian E. Raftery and Jeffrey D. Banfield. Stopping the gibbs sampler, the use of morphology, and other issues in spatial statistics. *Annals of the Institute of Statistical Mathematics*, 43:32–43, 1991.

[130] Sylvia Richardson and Peter J. Green. On bayesian analysis of mixtures with an unknown number of components. *Journal of the Royal Statistical Society. Series B*, 59(4):731–792, 1997.

[131] Brian D. Ripley. *Stochastic Simulation.* Wiley, New York, 1987.

[132] Christian P. Robert. Simulation of truncated normal variables. *Statistics and Computing*, 5:121–125, 1995.

[133] Christian P. Robert and George Casella. *Monte Carlo statistical methods*. Springer, 1999.

[134] Christian P. Robert and George Casella. *Monte Carlo Statistical Methods*. Springer, New York, 1999.

[135] Gareth O. Roberts and Jeffrey S. Rosenthal. Convergence of slice sampler markov chains. *Journal of the Royal Statistical Society. Series B*, 61(3): 643–660, 1999.

[136] Gareth O. Roberts and Jeffrey S. Rosenthal. Coupling and ergodicity of adaptive markov chain monte carlo algorithms. *Journal of Applied Probability*, 44(2):458–475, 2007.

[137] Marshall N. Rosenbluth and Arianna W. Rosenbluth. Monte carlo calculation of the average extension of molecular chains. *Journal of Chemical Physics*, 23:356–359, 1955.

[138] Jeffrey S. Rosenthal. Minorization conditions and convergence rates for markov chain monte carlo. *Journal of the American Statistical Association*, 90(420):558–566, 1995.

[139] Reuven Rubinstein and Dirk Kroese. *Simulation and the Monte Carlo method*. John Wiley & Sons, Inc., 1981.

[140] Jayaram Sethuraman. A constructive definition of dirichlet priors. *Statistica Sinica*, 4(2):639–650, 1994.

[141] Adrian F. M. Smith and Alan E. Gelfand. Bayesian statistics without tears: a sampling-resampling perspective. *The American Statistician*, 46 (2):84–88, 1992.

[142] D.A. Stephens and A.F.M Smith. Bayesian inference in multipoint gene mapping. *Annals of Human Genetics*, 57:65–82, 1993.

[143] David A. Stephens. Bayesian retrospective multiple change-point identification. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 43(1):159–178, 1994.

[144] Matthew Stephens. Bayesian analysis of mixture models with an unknown number of components—an alternative to reversible jump methods. *The Annals of Statistics*, 28(1):40–74, 2000.

[145] Robert H. Swendsen and Jian-Sheng Wang. Nonuniversal critical dynamics in monte carlo simulations. *Physical Review Letters*, 58(2):86–88, 1987.

[146] Martin A. Tanner and Wing Hung Wong. The calculation of posterior distributions by data augmentation. *Journal of the American Statistical Association*, 82(398):528–540, 1987.

[147] Matthew M. Tibbits, Murali Haran, and John C. Liechty. Parallel multivariate slice sampling. *Statistics and Computing*, 21(3):415–430, 2011.

[148] Matthew M. Tibbits, Chris Groendyke, Murali Haran, and John C. Liechty. Automated factor slice sampling. *Journal of Computational and Graphical Statistics*, 23(2):543–563, 2014.

[149] Luke Tierney. Markov chains for exploring posterior distributions. *The Annals of Statistics*, 22(4):1701–1728, 1994.

[150] Jon C. Wakefield, Adrian F. M. Smith, Amy Racine-Poon, and Alan E. Gelfand. Bayesian analysis of linear and non-linear population models by using the gibbs sampler. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 43(1):201–221, 1994.

[151] Stephen G. Walker. Sampling the dirichlet mixture model with slices. *Communications in Statistics - Simulation and Computation*, 36(1):45–54, 2007.

[152] Stephen G. Walker. Sampling unnormalized probabilities: An alternative to the metropolis–hastings algorithm. *SIAM Journal on Scientific Computing*, 36(2):A482–A494, 2014.

[153] C. S. Wang, J.J. Rutledge, and D. Gianola. Marginal inferences about variance components in a mixed linear model using gibbs sampling. *Genetics Selection Evolution*, 25(1):41–62, 1993.

[154] C. S. Wang, J.J. Rutledge, and D. Gianola. Bayesian analysis of mixed linear models via gibbs sampling with an application to litter size in iberian pigs. *Genetics Selection Evolution*, 26(2):91–115, 1994.

[155] Ulli Wolff. Collective monte carlo ppdating for spin systems. *Physical Review Letters*, 62(4):361–364, 1989.

[156] Scott L. Zeger and M. Rezaul Karim. Generalized linear models with random effects; a gibbs sampling approach. *Journal of the American Statistical Association*, 86(413):79–86, 1991.

[157] Arnold Zellner. Applications of bayesian analysis in econometrics. *Journal of the Royal Statistical Society. Series D (The Statistician)*, 32(1/2):23–34, 1983.

Vita

Yanxin received her Bachelor of Science degree in Electronics Information Sciences and Technology and Minor of Economics in Finance from Nankai University in 2012 and Master of Science degree in Applied and Computational Mathematics and Statistics from University of Notre Dame in 2016. She began her Ph.D. study in the Department of Statistics and Data Sciences at the University of Texas at Austin in 2017. Her research interests span Markov chain Monte Carlo sampling methods and Bayesian statistics with applications to model selection, regression analysis, clustetring, etc.

Permanent email address: liyanxin2015@gmail.com.

This dissertation was typeset with LaTeX by the author.