Advances in Markov chain Monte Carlo methods

Iain Murray

M.A., M.Sci., Natural Sciences (Physics), University of Cambridge, UK (2002)

Gatsby Computational Neuroscience Unit
University College London
17 Queen Square
London WC1N 3AR, United Kingdom

THESIS

Submitted for the degree of
Doctor of Philosophy, University of London

2007

I, Iain Murray, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the thesis.

Abstract

Probability distributions over many variables occur frequently in Bayesian inference, statistical physics and simulation studies. Samples from distributions give insight into their typical behavior and can allow approximation of any quantity of interest, such as expectations or normalizing constants. Markov chain Monte Carlo (MCMC), introduced by Metropolis *et al.* (1953), allows sampling from distributions with intractable normalization, and remains one of most important tools for approximate computation with probability distributions.

While not needed by MCMC, normalizers are key quantities: in Bayesian statistics marginal likelihoods are needed for model comparison; in statistical physics many physical quantities relate to the partition function. In this thesis we propose and investigate several new Monte Carlo algorithms, both for evaluating normalizing constants and for improved sampling of distributions.

Many MCMC correctness proofs rely on using reversible transition operators; often these operators lead to slow diffusive motion resembling a random walk. After reviewing existing MCMC algorithms, we develop a new framework for constructing non-reversible transition operators that allow more persistent motion.

Next we explore and extend MCMC-based algorithms for computing normalizing constants. We compare annealing, multicanonical and nested sampling, giving recommendations for their use. We also develop a new MCMC operator and Nested Sampling approach for the Potts model. This demonstrates that nested sampling is sometimes better than annealing methods at computing normalizing constants and drawing posterior samples.

Finally we consider "doubly-intractable" distributions with extra unknown normalizer terms that do not cancel in standard MCMC algorithms. We propose using several deterministic approximations for the unknown terms, and investigate their interaction with sampling algorithms. We then develop novel exact-sampling-based MCMC methods, the *Exchange Algorithm* and *Latent Histories*. For the first time these algorithms do not require separate approximation before sampling begins. Moreover, the Exchange Algorithm outperforms the only alternative sampling algorithm for doubly intractable distributions.

Acknowledgments

I feel very fortunate to have been supervised by Zoubin Ghahramani. My training has benefited greatly from his expertise, enthusiasm and encouragement. I have been equally fortunate to receive regular advice and ideas from David MacKay. Much of this research would never have happened without these mentors and friends.

This work was carried out at the Gatsby computational neuroscience unit at University College London. This is a first-class environment in which to conduct research, both intellectually and socially. Peter Dayan, the director, is owed a lot of credit for this, as are all of Gatsby's members and visitors, past and present. Several individuals have offered me their constant support, advice and good humor, in particular Angela Yu, Ed Snelson, Katherine Heller and members of the Inference group in Cambridge. I'd also like to extend a special thanks to Alex Boss whose administrative help often extended beyond the call of duty.

Some of chapter 4 is a review of John Skilling's nested sampling algorithm. John has been generous with his advice and encouragement, which enabled me to pursue a study based on his work. Thanks also to Radford Neal for comments on aspects of chapters 3 and 5, Hyun-Chul Kim for mean-field code used in chapter 5 and Matthew Stephens for encouraging me to work out the non-infinite explanation of the exchange algorithm. Thanks to my examiners Peter Green and Lorenz Wernisch for reviewing this work and catching some mistakes. Of course I am solely responsible for any errors that remain.

I am enormously grateful to the Gatsby charitable foundation for funding my research and providing travel money. Further travel monies were received from the PASCAL network of excellence, AUAI, the NIPS foundation and the Valencia Bayesian meeting.

Various items of free software have played a vital role in conducting this research: gcc, GNU/Linux, gnuplot, LaTeX, METAPOST (Hobby, 1992), Octave, Valgrind (Seward and Nethercote, 2005), Vim and many more. Projects in the public interest such as these deserve considerable support.

Finally I would like to thank my family and friends for all their support and patience.

Contents

List of algorithms

List of figures

List of tables

Notes on notation

Probability distributions

We have chosen to follow a fairly loose, but commonly-used notation for probabilities.

Occasionally we use $P(X=x)$ to denote the probability that a random variable X takes on the value x. But as long as the meaning can be inferred from context we simply write $P(x)$.

Often there is more than one probability distribution over the same variable. We simply write $q(x)$ for the probability under distribution q and $p(x)$ for the probability under p.

We never mention the space in which X lives, nor any measure theory unless it is actually used. We rarely need to distinguish between probability densities and discrete probabilities. This loose notation is imprecise, but hopefully its simplicity will be appreciated by some readers.

Probability of a "given" b

We use several notations for distributions over variables that depend on other quantities:

- $P(a|b)$ — This is the conditional probability of a given b. Bayes rule can be applied to infer b from a: $P(b|a) = P(a|b)P(b)/P(a)$.
- $P(a;b)$ — The probability of a is a function depending on some parameter b. One should not necessarily assume that Bayes rule holds for a and b.
- $T(a \leftarrow b)$ — T is a transition operator which gives a probability distribution over a new position a given a starting position b. One could also write $T(a;b)$, the arrow is to provide a more obvious distinction from authors that use $T(a,b)$ for the probability of the transition $a \to b$. Transition operators do not necessarily satisfy Bayes rule, known as "detailed balance", so the notation $T(a|b)$ is avoided.
- $T(a \leftarrow b;c)$ — This specifies parameters c in addition to starting location b.

Expectations

We use $\mathbb{E}_{p(x)}[f(x)] \equiv \sum_x p(x)f(x)$ for the expectation or average of $f(x)$ under the distribution $p(x)$. A sum with no specified range should be taken to mean "over all values". The variance of a quantity is given by $\mathrm{var}_p[f] \equiv \mathbb{E}_{p(x)}\left[f(x)^2\right] - \mathbb{E}_{p(x)}[f(x)]^2$.

Chapter 1

Introduction

Probability distributions over many variables occur frequently in Bayesian inference, statistical physics and simulation studies. Computational methods for dealing with these large and complex distributions remains an active area of research.

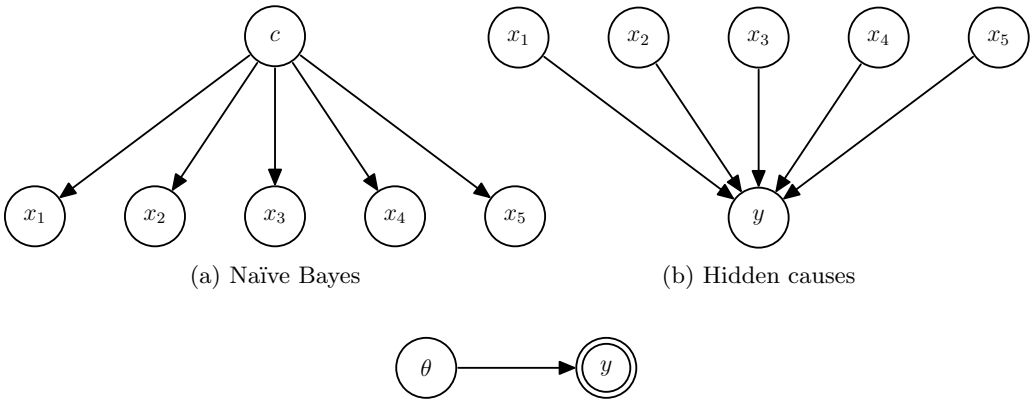
Graphical models (section 1.1) provide a powerful framework for representing these distributions. We use these to explain challenging probability distributions, and sometimes the algorithms to deal with them. A surprising number of statistical problems result in the computation of averages, which we explain in section 1.2. Monte Carlo techniques (section 1.3) approximate these summations using random samples. Many of these methods rely on the use of Markov chains (section 1.4). Extending Markov chain Monte Carlo (MCMC) techniques is the subject of this thesis.

1.1 Graphical models

Compact representations of high-dimensional probability distributions are essential for their interpretation, feasibility of learning and computational reasons. As a concrete example consider a distribution over a vector of D binary variables $\mathbf{x}$. For small D, e.g. two or three, an explicit table of counts for all possible joint settings could be maintained and used for frequency-based estimates of the settings' probabilities. Explicitly storing such a table becomes exponentially more costly as D grows. Even if the table is stored in a sparse data structure, the representation is not useful for learning probabilities: most cells will contain zero observations. Enforcing some structure is essential when learning from large multivariate distributions.

The simplest multivariate distributions assume that all of their component variables x_d are independent:

$$p(\mathbf{x}) = \prod_{d=1}^{D} p(x_d). \tag{1.1}$$



(a) Naïve Bayes (b) Hidden causes

(c) Simple parametric model

Figure 1.1: A selection of directed graphical models.

This assumption is generally inappropriate as it means that no relationships between any of the variables can ever be learned. The *graphical model* for equation (1.1) has a node for each x_d, with no edges between any of the nodes. Graphs with more structure provide a convenient representation for different factorizations of $p(\mathbf{x})$.

Two classes of graphical model are used in this thesis. Directed graphs offer a natural representation of causal relationships or the result of a sequence of operations. While useful in modeling we mainly use directed graphs for explanations of algorithms. Undirected graphical models are a more natural representation for constraints and mutual interactions. These are an important motivation for chapter 5. This section offers a brief review of the required concepts.

1.1.1 Directed graphical models

A directed graphical model is represented by a directed acyclic graph (DAG). Generally the joint distribution factorizes into a product of terms for each node conditioned on all of its parents. Figure 1.1a is a directed graphical model for the joint distribution found in the "naïve Bayes" model with class variable c and feature vector $\mathbf{x}=\{x_d\}$,

$$p(c, \mathbf{x}) \;=\; p(c)p(\mathbf{x}|c) \;=\; p(c)\prod_{d=1}^{D} p(x_d|c). \tag{1.2}$$

Figure 1.1a and equation (1.2) contain exactly the same information. From either form it is readily apparent, to those familiar with the representations, that the features $\mathbf{x}$ are dependent, but independent conditioned on the class variable c. As more variables and more complicated structure are introduced, the graphical model is often easier to interpret at a glance than an equation giving the factorization of the joint distribution. Several graphical model figures have been included throughout this thesis for those that might find them useful.

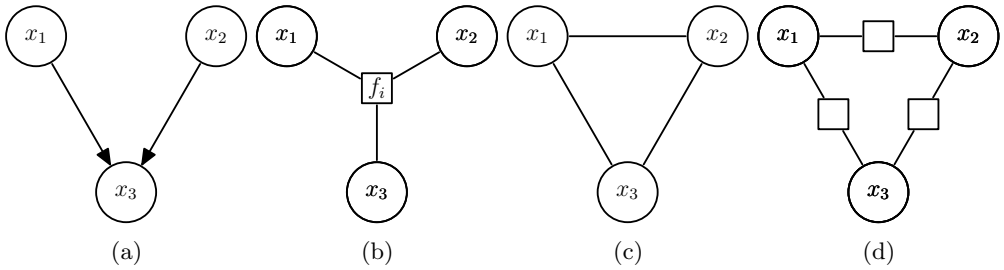


Figure 1.2: Some graphical models over three variables.

Figure 1.1c demonstrates another piece of directed graphical model notation. Sometimes shading or a double-outline is used to indicate that a node represents an observed variable. Here y are the data being modeled, which are assumed to come from a distribution parameterized by θ. Strictly this graph says nothing else about the distribution. Any joint distribution over two variables can be written as $p(y, \theta) = p(y|\theta)p(\theta) = p(\theta|y)p(y)$, so the arrow could equally be drawn the other way. The implication is that generating y from θ is a natural operation.

The arrow directions make a big difference in figure 1.1b. The x variables are independent in the generative process. After observing y our knowledge about $\mathbf{x}$ forms a potentially complex joint distribution: we do not expect direct sampling from $p(\mathbf{x}|y)$ to be easy. Methods based on Markov chains will offer a solution to this problem.

1.1.2 Undirected graphical models

A probability distribution must be normalized, so one representation, used frequently in chapter 5, is

$$p(x) = f(x;\theta)/\mathcal{Z}(\theta), \qquad \mathcal{Z}(\theta) = \sum_x f(x;\theta), \tag{1.3}$$

where $f(x;\theta)$ is any positive function for which the normalization sum (or integral) exists. However, without detailed analysis of f, this representation says nothing about the structure of the distribution. Undirected graphical models represent f as a product of terms, but do not identify some nodes as 'children' and others as 'parents'. This is appropriate for representing mutual interactions.

Modern undirected models use bipartite 'factor graphs'. Figure 1.2b shows three variables which all interact through a central factor node. Traditionally this distribution would be represented as in figure 1.2c, where the three nodes all take part in a common interaction as they form a fully connected clique. The factor representation is more powerful: the traditional representation cannot represent the distribution in figure 1.2d, which contains three factors: $p(x_1, x_2, x_3) = f(x_1, x_2)f(x_1, x_3)f(x_2, x_2)/\mathcal{Z}$.

1.1.3 The Potts model

The Potts model is a widespread example of an undirected graphical model. Its probability distribution is defined over discrete variables $\mathbf{s}$, each taking on one of q distinct settings or "colors":

$$p(\mathbf{s}|J,q) = \frac{1}{\mathcal{Z}_{\mathrm{P}}(J,q)} \exp\Big(\sum_{(ij)\in\mathcal{E}} J_{ij}(\delta_{s_i s_j} - 1) \Big). \tag{1.4}$$

The variables exist as nodes on a graph where $(ij) \in \mathcal{E}$ means that nodes i and j are linked by an edge. The Kronecker delta, $\delta_{s_i s_j}$ is one when s_i and s_j are the same color and zero otherwise. Neighboring nodes pay an "energy penalty" of J_{ij} when they are different colors. Often a single common coupling $J_{ij} = J$ is used. A common extension allows biases to be put on the colors: an additional term $\sum_i h_i(s_i)$ is put inside the exponential. The Potts model with bias terms and $q = 2$ has several different names: the Boltzmann machine, the Ising model, binary Markov random fields or autologistic models.

1.1.4 Computations with graphs

Imagine a problem that involves summing over the settings of a binary Potts model with 100 variables. In general this seems impossible: there are 2^{100} possible states. For comparison the age of the universe is usually estimated to be about 2^{98} picoseconds, while most current processors take about 2^{10} picoseconds to perform a single instruction.

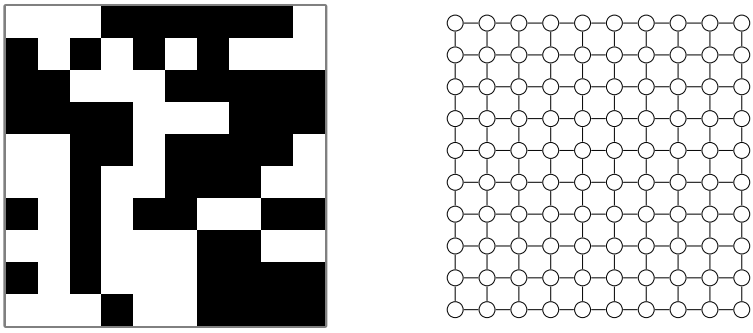
If the model's graph forms a chain, s_1—s_2—$s_3 \cdots s_N$, the distribution can be factored into a product of functions involving each adjacent pair:

$$p(\mathbf{s}|J) = \frac{1}{\mathcal{Z}(J)} \prod_{n=2}^{N} f_n(s_n, s_{n-1}; J). \tag{1.5}$$

Now certain sums, such as the normalizer $\mathcal{Z}(J) = \sum_{\mathbf{s}} \prod f_n(s_n, s_{n-1}; J)$ and expectations of functions of variables $\mathbb{E}_{p(\mathbf{s})}[g(s_i)]$ are tractable. An example showing how the sums can be decomposed is as follows:

$$\begin{aligned} \sum_{\mathbf{s}} g(s_i) \prod_{n=2}^{N} f_n(s_n, s_{n-1}) = & \textstyle\sum_{s_1} \sum_{s_2} f_2(s_2, s_1) \sum_{s_3} f_3(s_3, s_2) \cdots \\ & \textstyle\sum_{s_i} f_i(s_i, s_{i-1}) g(s_i) \sum_{s_{i+1}} f_{i+1}(s_{i+1}, s_i) \cdots \\ & \textstyle\sum_{s_{N-1}} f_{N-1}(s_{N-1}, s_{N-2}) \sum_{s_N} f_N(s_N, s_{N-1}). \end{aligned} \tag{1.6}$$

Performing the sums right to left makes the computation $\mathcal{O}(qN)$ rather than $\mathcal{O}(q^N)$, where $q = 2$ for binary variables.



(a) (b)

Figure 1.3: A "small" state space of 100 binary variables represented as (a) pixels and (b) an undirected graphical model.

The above technique easily generalizes to tree-structured graphs, but other topologies are common in applications. Figure 1.3 shows a grid of 100 binary variables. Arrays like this are common in computer vision, spatial statistics and statistical physics. Treating each row of 10 pixels as a single variable makes a chain-structured graph with $N = 10$ variables each taking on $2^{10} = 1024$ states. Summing has changed from an operation that takes orders of magnitude longer than the age of the universe to being almost instantaneous. Advanced versions of this procedure exist for general graphical models, notably the junction tree algorithm (Lauritzen and Spiegelhalter, 1988). The cost of the algorithm is determined by the tree-width of a graph, which indicates the largest number of variables that need to be joined in order to form a tree structure.

Figure 1.4 shows a genuinely intractable graphical model. Or does it? Even if the variables are only binary then summing over all 50×50 states with a graph-based exact inference algorithm is infeasible. Forming a tractable tree will require making at least one node with 2^{50} joint settings of 50 variables[1]. The topology of the graph isn't everything however. If the variables were continuous and Gaussian distributed then the model is quite tractable. Almost everything one might want to know about a Gaussian distribution can be found easily from a Cholesky decomposition of the covariance matrix. This matrix factorization is flexible, numerically stable and costs $\mathcal{O}(N^3)$ for a $N \times N$ covariance matrix (Seeger, 2005). When $N = 50 \times 50 = 2500$ a current mid-range workstation can perform the required computation in a couple of seconds.

1.2 The role of summation

Summing over all the configurations of a multivariate distribution turns out to be the dominant computational task in many fields. One of the goals of statistical physics is to capture the collective behavior of systems that, like the Potts model, involve enormous numbers of interacting parts. Many physical quantities relate to simple statistics of

[1] Summing in diagonal stripes means this need only happen once rather than 50 times.

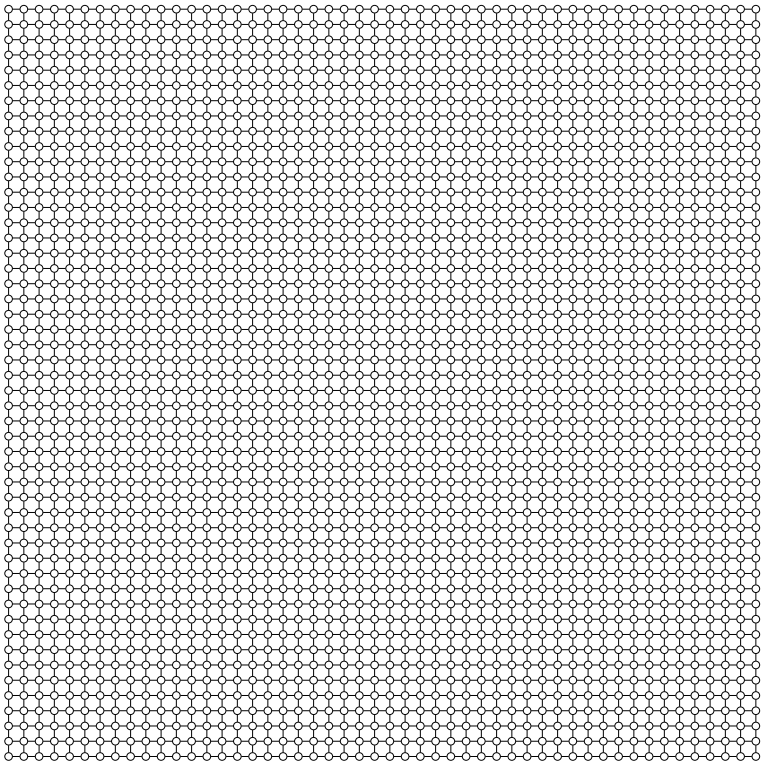


Figure 1.4: An intractable undirected graphical model? A grid of 50×50 variables with pairwise interactions.

these parts averaged over the entire system. Other key quantities can be derived from $\mathcal{Z}$, the *Zustandssumme* or sum over states.

Bayesian inference, the use of probability theory to deal with uncertainty, is also dominated by the computation of averages. As a canonical example we consider a statistical model $\theta \to x$ for data x generated using parameters θ. The predictive distribution over new data $x^{(N+1)}$ given observations of N previous settings $\{x^{(n)}\}_{n=1}^N$ is an average under the "posterior distribution" $p(\theta|\{x^{(n)}\}_{n=1}^N)$,

$$\begin{aligned} p(x^{(N+1)}|\{x^{(n)}\}_{n=1}^N) &= \int p(x^{(N+1)}|\theta)\, p(\theta|\{x^{(n)}\}_{n=1}^N)\, \mathrm{d}\theta \\ &= \mathbb{E}_{p(\theta|\{x^{(n)}\}_{n=1}^N)}\Big[p(x^{(N+1)}|\theta)\Big]\,. \end{aligned} \tag{1.7}$$

The posterior distribution is given by Bayes' rule

$$p(\theta|\{x^{(n)}\}_{n=1}^N) = \frac{p(\{x^{(n)}\}_{n=1}^N|\theta)p(\theta)}{\int p(\{x^{(n)}\}_{n=1}^N|\theta')p(\theta')\,\mathrm{d}\theta'}, \tag{1.8}$$

which involves another sum over θ. In general any time a quantity is unknown we must consider all of its possible settings, which tends to involve an average under a probability distribution.

This thesis concentrates on computational methods, rather than their application.

However, modeling and learning from data are a key motivation for this work, so we briefly mention some more general references. "Bayesian inference" is named after Bayes (1763), although many of the ideas that currently fall under this banner came much later. For more on the philosophy a good start is a beautifully written paper by Cox (1946)[2]. Recent textbooks offer a broader view of probabilistic modeling and are available from the viewpoints of various communities, including statistics (Gelman *et al.*, 2003), machine learning (MacKay, 2003; Bishop, 2006) and the physical sciences (Sivia and Skilling, 2006).

1.3 Simple Monte Carlo

Many of the alleged difficulties with finding averages are unduly pessimistic. Imagine we were interested in the average salary x of people p in the set of statisticians $\mathcal{S}$. Formally this is a large, "intractable" sum over all of the $|\mathcal{S}|$ statisticians in the world:

$$\mathbb{E}_{p\in\mathcal{S}}[x(p)] \equiv \frac{1}{|\mathcal{S}|}\sum_{p\in\mathcal{S}} x(p). \tag{1.9}$$

But to claim that the question is unanswerable is absurd. We can clearly get a reasonable estimate of the answer by measuring just some statisticians,

$$\mathbb{E}_{p\in\mathcal{S}}[x(p)] \approx \frac{1}{S}\sum_{s=1}^{S} x\big(p^{(s)}\big), \quad \text{for random survey of } S \text{ people } \{p^{(s)}\} \in \mathcal{S}. \tag{1.10}$$

No reasonable application needs the exact answer, which in any case is constantly fluctuating as individual statisticians are hired, promoted and retire. So for all practical purposes the problem is solved. Nearly. Conducting surveys that obtain a fair sample from a target population is difficult. But the technique is so useful we are prepared to invest effort into good sampling methods.

This statistical sampling technique is directly relevant to solving difficult integrals in statistical inference. For example, what is the distribution over an unknown quantity x after observing data $\mathcal{D}$ from a distribution with unknown parameters θ?

$$\begin{aligned} p(x|\mathcal{D}) = \int p(x|\theta,\mathcal{D})p(\theta|\mathcal{D})\,\mathrm{d}\theta &= \mathbb{E}_{p(\theta|\mathcal{D})}[p(x|\theta,\mathcal{D})] \\ &\approx \frac{1}{S}\sum_{s=1}^{S} p(x|\theta^{(s)},\mathcal{D}), \quad \theta^{(s)} \sim p(\theta|\mathcal{D}). \end{aligned} \tag{1.11}$$

We draw samples from the target distribution, which rather than the set of statisticians is now $p(\theta|\mathcal{D})$. Approximating general averages by statistical sampling is known as "simple Monte Carlo". The estimates are unbiased, which if not clear now is shown

[2]Cox's work has been subject to some technical objections which are countered by Horn (2003).

more generally in section 2.2. As long as variances are bounded appropriately the sum of independent terms will obey a central limit theorem; the estimator's variance will scale like $1/S$.

Having an estimator's variance shrink only linearly with computational effort is often considered poor. Sokal (1996) begins a lecture course on Monte Carlo with the warning

> *"Monte Carlo is an extremely* bad *method; it should be used only when all alternative methods are worse."*

However, as Sokal goes on to acknowledge, Monte Carlo is also an extremely important method. On some problems it might be the only way to obtain accurate answers. Metropolis (1987) describes how the physicist Enrico Fermi (1901–1954) used Monte Carlo before the development of electronic computers. When insomnia struck he would spend his nights predicting the results of complex neutron scattering experiments by performing statistical sampling calculations with a mechanical adding machine. As analytically deriving the behavior of many neutrons seemed intractable, Fermi's ability to make accurate predictions astonished his colleagues.

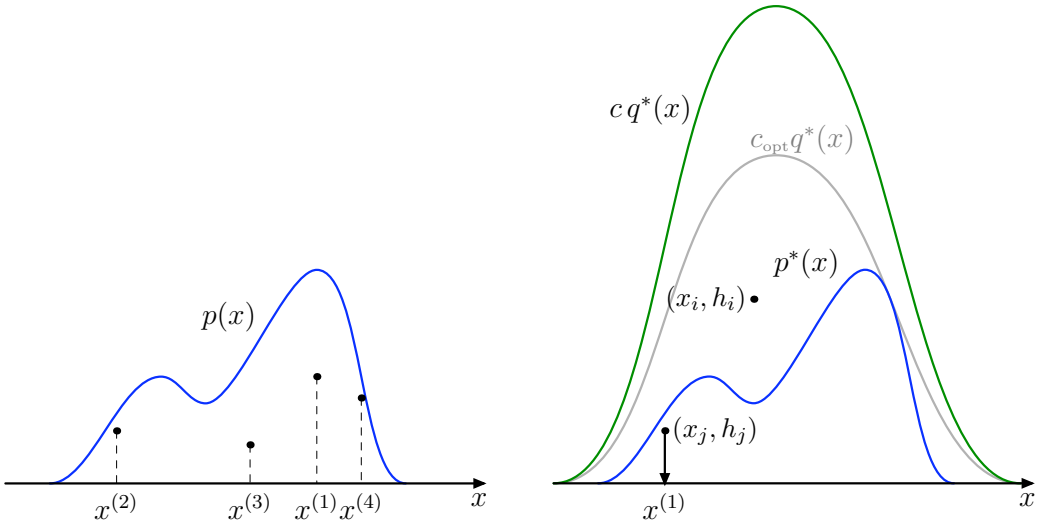
Many problems in physics and statistics are complex and involve many variables. Numerical methods that scale exponentially with the dimensionality of the problem will not work at all. In contrast, Monte Carlo is usually simple and its $1/\sqrt{S}$ scaling of error bars "independent of dimensionality" may be good enough. Even when more advanced deterministic methods are available, Monte Carlo can be appropriate if today's computers can return useful answers with less implementation effort than more complex methods.

1.3.1 Sampling from distributions

Just as finding a fair sample from a population is difficult in surveys, sampling correctly from arbitrary probability distributions is also hard. 'Simple' Monte Carlo is only as easy to implement as the random variate generator for the entire joint distribution involved.

A graphical description of sampling from a probability distribution is given by figure 1.5a. Points are drawn uniformly from the unit area underneath the probability density function and their corresponding value recorded. This correctly assigns each element of the input space $\mathrm{d}x$ with probability $p(x)\mathrm{d}x$.

The probability mass to the left of each point, $u(x)$, is distributed as Uniform$[0, 1]$. To implement a sampler we can first draw u and then compute $x(u)$ from the inverse cumulative distribution. That is $x(u) = \Phi^{-1}(u)$, where $\Phi(x) = \int_{-\infty}^{x} p(x')\,\mathrm{d}x'$. Now the difficulty of sampling from distributions with this general method becomes apparent. It is often infeasible to even normalize p, a single integral over the entire state space, whereas $\Phi(x)$ is a whole continuum of such integrals.



(a) Direct sampling (b) Rejection sampling

Figure 1.5: Drawing samples from low-dimensional distributions. (a) Sample locations can be taken from points drawn uniformly under the density curve. (b) Rejection sampling provides a way to implement this. Samples are drawn uniformly underneath a simpler curve $cq^*(x)$ as in algorithm 1.1. Points above $p^*(x) \propto p(x)$ such as (x_i, h_i) are rejected. The remaining points come uniformly from underneath p^* and are valid samples from p. The constant c is chosen such that cq^* is always above p^*. The best setting is shown as c_{opt}, but finding this value is not always possible.

While Φ^{-1} is tractable for some standard distributions, random number generators must generally use less direct techniques. Rejection sampling is a method for drawing samples from a distribution $p(x)$ by using another distribution $q(x)$ from which we can already sample. The method requires an ability to evaluate the probability density of points under both distributions up to a constant: $p^*(x) \propto p(x)$ and $q^*(x) \propto q(x)$. Further we must be able to find a constant c such that $cq^*(x) \geq p^*(x)$.

Sampling underneath a $p^*(x) \propto p(x)$ curve gives the correct probability distribution. We can sample under $p^*(x)$ by drawing samples from the tractable $q(x)$ distribution and applying algorithm 1.1, figure 1.5b.

The more closely q matches p the lower the rejection rate can be made. Providing that c is adjusted to maintain a valid bound, we can improve q after each step 5 of the algorithm. Ideally q would be updated automatically as points from p are evaluated. In general this could be difficult, but for log-concave distributions there are at least two techniques (Gilks and Wild, 1992; Gilks, 1992).

For applications of rejection sampling to standard distributions and several more techniques for non-uniform random variate generation see Devroye (1986).

Algorithm 1.1 Rejection sampling

Inputs: target distribution $p(x) \propto p^*(x)$,
simple distribution $q(x) \propto q^*(x)$,
and number of samples S.

Outputs: $\{x^{(s)}\}_{s=1}^{S}$, samples from $p(x)$

1. Find a constant c such that $cq^*(x) \geq p^*(x)$ for all x
   (ideally minimize c within the constraint)
2. **for** $s = 1 \ldots S$
3. Draw candidate $x \sim q$
4. Draw height $h \sim \text{Uniform}[0, cq^*(x)]$
5. **if** (x, h) is below p^*, i.e., $h < p^*(x)$, **then** $x^{(s)} \leftarrow x$ **else** go to 3.
6. **end for**

1.3.2 Importance sampling

Computing $p^*(x)$ and $q^*(x)$, then throwing x away along with all the associated computations seems wasteful. Yet discarding samples is a key part of rejection sampling's correct operation. Importance sampling avoids such rejections by rewriting the integral as an expectation under any distribution q that has the same support as p:

$$\begin{aligned} \int f(x)p(x)\,\mathrm{d}x &= \int f(x)\frac{p(x)}{q(x)}q(x)\,\mathrm{d}x, \quad \text{if } q(x) > 0 \text{ wherever } p(x) > 0 \\ &= \int f(x)w(x)\,q(x)\,\mathrm{d}x, \quad w(x) = p(x)/q(x) \\ &\approx \frac{1}{S}\sum_{s=1}^{S} f(x^{(s)})\,w(x^{(s)}), \quad x^{(s)} \sim q(x). \end{aligned} \tag{1.12}$$

One interpretation is that the weights $w(x^{(s)})$ make some samples from q more important than others, rather than assigning harsh weights of zero and one as in rejection sampling. The simplest view is that equation (1.12) is is just simple Monte Carlo integration of an expectation under $q(x)$. We immediately know that the estimator is unbiased and might obey a central limit theorem as before.

If $q(x)$ is much smaller than $p(x)$ in some regions of the state space then the effective function $f(x)p(x)/q(x)$ will have very high or even infinite variance under $q(x)$. States with extreme importance weights are rare events under q and might not be observed within a moderate number of samples. This means that error bars based on the empirical variance of the importance weights can be very misleading. For a practical example of this problem, along with the associated recommendation to use broad distributions with heavy tails see MacKay (2003, section 29.2).

Interestingly the ideal q distribution is not equal to p. If $f(x)$ is a positive function then $q(x) \propto f(x)p(x)$ would give a zero variance estimator. This optimal distribution is

unobtainable as evaluating $q(x)$ requires the normalization $\mathcal{Z}_q = \int f(x)p(x)\,\mathrm{d}x$, the target integral. But sometimes deliberate matches between q and p are useful in practice. For example when interested in gathering statistics about the tails of p.

Nothing about the importance sampling trick in equation (1.12) actually requires the original integral to be an expectation. We can divide and multiply any integrand by a convenient distribution to make it an expectation. Thus importance sampling allows any integral to be approximated by Monte Carlo. As most of the integrals in dominant applications are expectations, we still maintain $p(x)$ in the equations throughout.

Evaluating the importance weights $w(x)$ requires evaluating $p(x) = p^*(x)/\mathcal{Z}_p$, but often $\mathcal{Z}_p$ is unknown. In these cases the normalizing constant must be approximated separately as follows:

$$\begin{aligned}\frac{\mathcal{Z}_p}{\mathcal{Z}_q} &= \frac{1}{\mathcal{Z}_q}\int p^*(x)\,\mathrm{d}x = \int \frac{p^*(x)}{q^*(x)}q(x)\,\mathrm{d}x \\ &= \int w^*(x)\,q(x)\,\mathrm{d}x, \quad w^*(x) = p^*(x)/q^*(x) \\ &\approx \frac{1}{S}\sum_{s=1}^{S} w^*(x^{(s)}), \quad x^{(s)} \sim q(x)\end{aligned} \tag{1.13}$$

This gives an approximation for the importance weights

$$\begin{aligned}w(x) = \frac{p(x)}{q(x)} &= \frac{\mathcal{Z}_q}{\mathcal{Z}_p}\frac{p^*(x)}{q^*(x)} \\ &\approx \frac{w^*(x)}{\frac{1}{S}\sum_{s=1}^{S} w^*(x^{(s)})},\end{aligned} \tag{1.14}$$

which can be used within equation (1.12). The resulting estimator is biased but consistent when both estimators (1.12) and (1.13) have bounded variance.

1.4 Markov chain Monte Carlo (MCMC)

Both rejection sampling and importance sampling require a tractable surrogate distribution $q(x)$. Neither method will perform well if $\max_x p(x)/q(x)$ is large: rejection sampling will rarely return samples and importance sampling will have large variance. Markov chain Monte Carlo methods can be used to sample from $p(x)$ distributions that are complex and have unknown normalization. This is achieved by relaxing the requirement that the samples should be independent.

A Markov chain generates a correlated sequence of states. Each step in the sequence is drawn from a transition operator $T(x' \leftarrow x)$, which gives the probability of moving from state x to state x'. According to the *Markov property*, the transition probabilities depend only on the current state, x. In particular, any free parameters σ, e.g. step

sizes, in a family of transition operators, $T(x' \leftarrow x; \sigma)$, cannot be chosen based on the history of the chain.

A basic requirement for T is that given a sample from $p(x)$, the marginal distribution over the next state in the chain is also the target distribution of interest p:

$$p(x') = \sum_x T(x' \leftarrow x)\, p(x) \quad \text{for all } x'. \tag{1.15}$$

By induction all subsequent steps of the chain will have the same marginal distribution. The transition operator is said to leave the target distribution p stationary. MCMC algorithms often require operators that ensure the marginal distribution over a state of the chain tends to $p(x)$ regardless of starting state. This requires *irreducibility*: the ability to reach any x where $p(x) > 0$ in a finite number of steps, and *aperiodicity*: no states are only accessible at certain regularly spaced times. For more details see Tierney (1994). For now we note that as long as a T satisfies equation (1.15) it can be useful as the other conditions can be met through combinations with other operators.

Given that $p(x)$ is a complicated distribution, it might seem unreasonable to expect that we could find a transition operator T leaving it stationary. However, it is often easy to construct a transition operator satisfying *detailed balance*:

$$T(x' \leftarrow x)\, p(x) = T(x \leftarrow x')\, p(x') \quad \text{for all } x, x'. \tag{1.16}$$

This states that a step starting at equilibrium and transitioning under T has the same probability "forwards" $x \to x'$ or "backwards" $x' \to x$. Proving detailed balance only requires considering each pair of states in isolation, there is no sum over all states as in equation (1.15). Having shown equation (1.16), summing over x on both sides immediately recovers the stationary distribution requirement (1.15). Thus detailed balance is a useful property for deriving many MCMC methods; however it is not always required or even desirable. Chapter 3 introduces some MCMC transition operators that do *not* satisfy equation (1.16).

Given any transition operator T satisfying the stationary condition, equation (1.15), we can construct a *reverse operator* $\widetilde{T}$ defined by

$$\widetilde{T}(x \leftarrow x') \propto T(x' \leftarrow x)\, p(x) = \frac{T(x' \leftarrow x)\, p(x)}{\sum_x T(x' \leftarrow x)\, p(x)} = \frac{T(x' \leftarrow x)\, p(x)}{p(x')}. \tag{1.17}$$

A symmetric form of this relationship shows that an operator satisfying detailed balance is its own reverse transition operator:

$$T(x' \leftarrow x)\, p(x) = \widetilde{T}(x \leftarrow x')\, p(x') \quad \text{for all } x, x'. \tag{1.18}$$

Summing over x or x' reveals that this mutual reversibility condition implies that the stationary condition, equation (1.15), holds for both T and $\widetilde{T}$. Therefore constructing

a pair of mutually reversible transition distributions is an alternative strategy for constructing MCMC operators. Detailed balance is a restricted case where a transition operator must be its own reverse operator.

1.5 Choice of method

In this introduction we described the importance of high-dimensional probability distributions. Samples from these distributions capture their typical properties, which is the basis of the Monte Carlo estimation of expectations such as $\int f(x)p(x)\ \mathrm{d}x$. We assume or hope that extreme values under $p(x)$ do not form a significant contribution to the integral. For expectations in many physical and statistical applications this is a reasonable assumption.

In high-dimensional problems sampling from an interesting target distribution $p(x)$ is often intractable. Methods such as rejection sampling fail because finding a close-matching simple distribution $q(x)$ is not possible. For the same reason importance sampling estimators tend to have very high variance.

We described rejection sampling as it remains an important method when i.i.d. samples from low-dimensional distributions are required. This occurs in some simulation work, and as part of some MCMC methods. Similarly while simple importance sampling has problems in high dimensions, it provides the basis of more advanced methods that are useful on some high-dimensional problems. We neglect much of the importance sampling literature, but will study some methods relating to Markov chains.

The focus of this thesis are methods that use Markov chains. These allow us to draw (correlated) samples from complex distributions and to perform Monte Carlo integration in high-dimensional problems. The next chapter reviews several important algorithms based on Markov chains together with some new contributions. After this we will be in a position to outline the remainder of the thesis.

Chapter 2

Markov chain Monte Carlo

This chapter reviews some important Markov chain Monte Carlo (MCMC) algorithms. Much of this material is standard and could be skipped by those already familiar with the literature. However, some of the material in this chapter is, to the best of our knowledge, novel. These contributions include generalizations of tempered transitions in subsection 2.5.4 and the introduction of a slice-sampling version of the two stage acceptance rule (see sections 2.1.3 and 2.4.2). We also present some results in an unconventional way, which is designed to help with reading later chapters.

2.1 Metropolis methods

Algorithm 2.1 gives a procedure for simulating a Markov chain with stationary distribution $p(x)$ due to Hastings (1970).

Algorithm 2.1 Metropolis–Hastings

Input: initial setting x, number of iterations S

1. **for** $s = 1 \ldots S$
2. Propose $x' \sim q(x' \leftarrow x)$
3. Compute $a = \dfrac{p(x')\, q(x \leftarrow x')}{p(x)\, q(x' \leftarrow x)}$
4. Set $x = x'$ with probability $\min(1, a)$, e.g.
   (a) Draw $r \sim \text{Uniform}[0, 1]$
   (b) **if** $(r < a)$ **then** set $x \leftarrow x'$.
5. **end for**

The setting of x at the end of each iteration is considered as a sample from the target probability distribution $p(x)$. Adjacent samples are identical when the state is not updated in step 4b), but every iteration must be recorded as part of the Markov chain.

It is straightforward to show that the Metropolis–Hastings algorithm satisfies detailed balance:

$$\begin{aligned} T(x' \leftarrow x)\, p(x) &= \min\left(1,\ \frac{p(x')\, q(x \leftarrow x')}{p(x)\, q(x' \leftarrow x)}\right) q(x' \leftarrow x)\, p(x) \\ &= \min\Big(p(x)\, q(x' \leftarrow x),\ p(x')\, q(x \leftarrow x')\Big) \\ &= \min\left(1,\ \frac{p(x)\, q(x' \leftarrow x)}{p(x')\, q(x \leftarrow x')}\right) q(x \leftarrow x')\, p(x') \\ &= T(x \leftarrow x')\, p(x'), \quad \text{as required.} \end{aligned} \tag{2.1}$$

Here the probability for a forwards transition $x \to x'$ was manipulated into the probability of the reverse transition $x' \to x$. However it would have been sufficient to stop after the second line and note that the expression is symmetric in x and x'.

The algorithm is valid for any proposal distribution q. Ideal proposals would be constructed for rapid exploration of the distribution of interest p. We could write $q(x' \leftarrow x; \mathcal{D})$ to emphasize choices of proposal that are based on observed data $\mathcal{D}$. Using any fixed $\mathcal{D}$ is valid, but q cannot be based on the past history of the sampler as such a chain would not be "Markov".

Often proposals are not complicated data-based distributions. Simple perturbations such as a Gaussian with mean x are commonly chosen. The accept/reject step 4. in the algorithm, corrects for the mismatch between the proposal and target distributions. When the proposal distribution is symmetric, i.e. $q(x \leftarrow x') = q(x' \leftarrow x)$ for all x, x', the acceptance ratio in step 3. simplifies to a ratio under the distribution of interest, $a = p(x')/p(x)$. This is the original Metropolis algorithm (Metropolis *et al.*, 1953). Some authors, e.g. MacKay (2003), prefer to drop such distinctions and simply refer to all Metropolis–Hastings algorithms as Metropolis methods. The next section suggests another justification of this view.

2.1.1 Generality of Metropolis–Hastings

Consider a MCMC algorithm that proposes a state from a distribution $q(x' \leftarrow x)$ and accepts with probability $p_a(x' \leftarrow x)$. Restricting attention to transition operators satisfying detailed balance,

$$p_a(x' \leftarrow x) q(x' \leftarrow x)\, p(x) = p_a(x \leftarrow x') q(x \leftarrow x')\, p(x'), \qquad \text{for all } x,\, x', \tag{2.2}$$

gives an equality constraint. We also have inequalities that must hold for probabilities:

$$0 \le p_a(x' \leftarrow x) \le 1 \quad \text{and} \quad 0 \le p_a(x \leftarrow x') \le 1. \tag{2.3}$$

Optimizing the average acceptance probability $p(x)p_a(x' \leftarrow x) + p(x')p_a(x \leftarrow x')$ with respect to $a(x' \leftarrow x)$ and $a(x \leftarrow x')$ must saturate one or more of the inequalities (2.3), a

well known property of *linear programming* problems. Therefore, either $p_a(x' \leftarrow x) = 1$ or $p_a(x \leftarrow x') = 1$ and $p_a(x' \leftarrow x)$ is given by equation (2.2). This gives the Metropolis–Hastings acceptance rule,

$$p_a(x' \leftarrow x) = \min\left(1, \frac{p(x')q(x \leftarrow x')}{p(x)q(x' \leftarrow x)}\right). \tag{2.4}$$

According to the constraint of equation (2.2), a pair of valid acceptance probabilities must have the same ratio $p_a(x' \leftarrow x)/p_a(x \leftarrow x')$ as any other valid pair. Therefore they can be obtained by multiplying the Metropolis–Hastings probabilities by a constant less than one. This corresponds to mixing in some fraction of the "do nothing" transition operator, which leaves the chain in the current state, at those two sites. It also corresponds to adjusting the proposal distribution q to suggest staying still more often. Staying still more often harms the asymptotic variance of the chain: in this sense (although not by all measures) using equation (2.4) is optimal (Peskun, 1973).

Given this result it is unsurprising that Metropolis–Hastings has become almost synonymous with MCMC. It is tempting to conclude that the only way to improve Markov chains for Monte Carlo is by researching domain-specific proposal distributions such as in the vision community's "Data-driven MCMC" (Tu and Zhu, 2002). In fact a rich variety of more generic MCMC-based algorithms exist and continue to be developed. Many of these do satisfy detailed balance, but the corresponding M–H $q(x' \leftarrow x)$ distribution is often defined implicitly and would not be a natural description of the method.

To illustrate the limitations of claiming "all (reversible) MCMC is just Metropolis–Hastings" we show that "all (reversible) MCMC is just Metropolis". This also demonstrates a methodology used throughout the thesis. We construct a new target distribution $\ddot{p}(x, x') = q(x' \leftarrow x)\, p(x)$, i.e. the joint distribution over a point $x \sim p$ and a point x' proposed from that location. The marginal distribution over x is the original target p. Now consider the symmetric Metropolis proposal that swaps the values of x and x' such that putatively x' comes from p and x was proposed from it. The Metropolis acceptance probability for this swap proposal is

$$\min\left(1, \frac{\ddot{p}(x', x)}{\ddot{p}(x, x')}\right) = \min\left(1, \frac{p(x')\, q(x \leftarrow x')}{p(x)\, q(x' \leftarrow x)}\right), \tag{2.5}$$

i.e., the Metropolis–Hastings acceptance probability. In this sense only an algorithm with symmetric proposals is needed. But this is not a very natural description of the Metropolis–Hastings algorithm. Similarly, while it is possible to describe all algorithms satisfying detailed balance as Metropolis or Metropolis–Hastings algorithms, other descriptions may be more natural. However, we *will* find constructing joint distributions like $\ddot{p}(x, x')$ a useful theoretical tool and one that suggests new Markov chain operators.

2.1.2 Gibbs sampling

Gibbs sampling (Geman and Geman, 1984) resamples each dimension x_i of a multivariate quantity $\mathbf{x}$ from their conditional distributions $p(x_i|\mathbf{x}_{j\neq i})$. Any individual update maintains detailed balance, which is easily checked directly. Alternatively we can write the Gibbs sampling update as a Metropolis–Hastings proposal: $q(\mathbf{x}' \leftarrow \mathbf{x}) = p(x'_i|\mathbf{x}_{j\neq i})\mathbb{I}(\mathbf{x}'_{\{j\neq i\}} = \mathbf{x}_{\{j\neq i\}})$, where $\mathbb{I}$ is an indicator function ensuring all components other than x_i stay fixed. The acceptance probability for this proposal is identical to one, so need not be checked.

Gibbs sampling is often easy to implement. If the target distribution is discrete and each variable takes on a small number of settings then the conditional distributions can be explicitly computed,

$$p(x_i|\mathbf{x}_{j\neq i}) \propto p(x_i, \mathbf{x}_{j\neq i}) = \frac{p(x_i, \mathbf{x}_{j\neq i})}{\sum_{x'_i} p(x'_i, \mathbf{x}_{j\neq i})}. \tag{2.6}$$

On continuous problems the one-dimensional conditional distributions are usually amenable to standard sampling methods as mentioned in subsection 1.3.1.

A desirable feature of Gibbs sampling is that it has no free parameters and so can be applied fairly automatically. Indeed the *BUGS* packages can create Gibbs samplers for a large variety of models specified using a simple description language (Spiegelhalter *et al.*, 1996). There are actually some free choices regarding how the variables are updated. "Block-Gibbs" sampling chooses groups of variables to update at once. Also, continuous distributions can be re-parameterized before Gibbs sampling — a basis in which the dimensions are independent would obviously be particularly effective.

Clifford *et al.* (1993) contains several interesting discussions regarding the nature and history of the Gibbs sampler. The method has often been regarded with a rather undeserved special status amongst MCMC methods. It is just one of many ways to construct a Markov chain with a target stationary distribution. In particular there is no need to approximate Gibbs sampling when sampling from conditionals is not tractable as in (Ritter and Tanner, 1992). Metropolis–Hastings updates of each dimension can be used instead, and there is no need to call this method Metropolis-within-Gibbs.

2.1.3 A two stage acceptance rule

If the target distribution is factored into two terms, for example a prior and likelihood $p(x) \propto \pi(x)L(x)$, a proposal can be accepted with probability

$$p_a = \min\left(1, \frac{q(x \leftarrow x')\pi(x')}{q(x' \leftarrow x)\pi(x)}\right) \min\left(1, \frac{L(x')}{L(x)}\right). \tag{2.7}$$

Straightforward checking shows that this satisfies detailed balance. We know from previous discussions that it will accept less often than standard Metropolis–Hastings and is inferior according to Peskun (1973), but computationally it can be more efficient. Algorithm 2.2 is an acceptance rule that will accept proposals with probability p_a. If $\pi(x')$ is able to veto an *a priori* unreasonable proposal then $L(x')$ need not be computed. Factoring out a cheap "sanity-check" distribution $\pi(x)$ may be worth a fall in acceptance rate in problems where likelihood evaluations are expensive.

Algorithm 2.2 A two stage acceptance rule

Input: initial setting x, proposed setting x'

1. Draw $r_1 \sim \text{Uniform}[0,1]$
2. **if** $r_1 < \dfrac{q(x \leftarrow x')\pi(x')}{q(x' \leftarrow x)\pi(x)}$
3. Draw $r_2 \sim \text{Uniform}[0,1]$
4. **if** $r_2 < \dfrac{L(x')}{L(x)}$ Accept **else** Reject
5. **else**
6. Reject

As with standard Metropolis–Hastings none of the probabilities need to be computed exactly. Bounding them sufficiently to make the accept/reject decision in step 2. or 4. is all that is required. In practice few Monte Carlo codes implement the required interval arithmetic to make these savings. In contrast the two stage procedure is generally applicable and easy to implement.

Algorithm 2.2 is a special case of the "surrogate transitions method" (Liu, 2001, pp. 194–195) and also the (different) algorithm of Christen and Fox (2005). The two-stage method here is also equivalent to Dostert *et al.* (2006, Algorithm II), which presented it in the context of a particular choice for $q(x' \leftarrow x)$. This literature has a rich variety of possible approximate distributions that can be used for $\pi(x)$ in particular applications. In a statistical application where $p(x)$ depends on a data set a general choice would be to use a subset of the data to define $\pi(x)$.

An obvious generalization is splitting the acceptance rule into more than two terms. Factoring the distribution into many terms could make the acceptance rate fall dramatically, so there would need to be a specific computational benefit.

2.2 Construction of estimators

The output of an MCMC algorithm is a set of correlated samples drawn from a joint distribution $P(\{x^{(s)}\})$. At equilibrium every marginal $p(x^{(s)})$ is the correct distribution of interest. Using these equilibrium samples in the straightforward Monte Carlo

estimator would be unbiased,

$$\begin{aligned}\mathbb{E}_{P(\{x^{(s)}\})}\left[\frac{1}{S}\sum_{s=1}^{S} f(x^{(s)})\right] &= \sum_{\{x^{(s)}\}} P(\{x^{(s)}\})\frac{1}{S}\sum_{s=1}^{S} f(x^{(s)}) \\ &= \frac{1}{S}\sum_{s=1}^{S}\sum_{x^{(s)}} f(x^{(s)}) \sum_{\{x^{(s')}\}_{s'\neq s}} P(\{x^{(s)}\}) \\ &= \frac{1}{S}\sum_{s=1}^{S}\sum_{x^{(s)}} f(x^{(s)})p(x^{(s)}) = \mathbb{E}_p[f]\,.\end{aligned} \tag{2.8}$$

The states drawn from a Markov chain started at an arbitrary position will not have the correct marginal distribution. Asymptotically equation (2.8) is still an unbiased estimator under weak conditions, but in practice it is advisable to discard some initial states allowing the chain to "burn-in". After this, there is no need to remove intermediate samples in an attempt to obtain a set of approximately independent samples. Such "thinning" will only make the variance of the estimator worse (Geyer, 1992). However, thinning can be justified when adjacent steps are highly correlated or when computing $f(x)$ is costly. Then discarding some samples can save computer time that is better spent running the Markov chain for longer.

A related issue is whether it is better to run one long Markov chain, or perhaps to obtain independent points more quickly by trying many initializations. Geyer (1992) also shows that theoretically a longer chain is to be preferred. Intuitively an arbitrary initialization is bad as it introduces bias, which takes time to remove. It turns out this time is better spent evolving an existing chain to new locations.

Despite the theory favoring running a single chain we tend to prefer running a few rather than one. When more than one computer processor is available this is the easiest way to "parallelize" our code. Although agreement amongst multiple chains does not provide a guarantee that the Markov chains are mixing, differing results would reveal a failure to mix that might go unnoticed from a single chain. Finally, multiple chains are useful for adapting step-sizes in Metropolis methods and turn up naturally in population methods (chapter 3).

2.2.1 Conditional estimators ("Rao-Blackwellization")

Using a sample average of function values is not the only way to construct MCMC estimators. It is an attractive default procedure as the estimator only needs to know $f(x)$ evaluated at the samples. However, it is sometimes possible to perform some computations involving $f(x)$ analytically. A simple identity lets us use this analytical knowledge:

$$\sum_x f(x)p(x) = \sum_h \left[\sum_x f(x)p(x|h)\right] p(h), \tag{2.9}$$

where h is an arbitrary statistic. If we can sum over the distribution conditioned on h, then the bracketed term evaluated under samples of h can be used as an estimator. A special case provides more concrete motivation:

$$\begin{aligned}\sum_{\mathbf{x}} f(x_i)p(\mathbf{x}) &= \sum_{\mathbf{x}_{\{j\neq i\}}} \left[\sum_{x_i} f(x_i)p(x_i|\mathbf{x}_{\{j\neq i\}})\right] p(\mathbf{x}_{\{j\neq i\}}) \\ &= \mathbb{E}_{p(\mathbf{x}_{\{j\neq i\}})}\left[\mathbb{E}_{p(x_i|\mathbf{x}_{\{j\neq i\}})}[f(x_i)]\right].\end{aligned} \tag{2.10}$$

Here we are interested in a function of a particular variable x_i. Sums over a single variable are often tractable. The identity allows averaging $\mathbb{E}_{p(x_i|\mathbf{x}_{\{j\neq i\}})}[f(x_i)]$ under Monte Carlo samples, rather than $f(x)$ itself. As an example consider finding the marginal of a binary variable: $x_i \in \{0,1\}$, $f(x_i)=x_i$. The standard estimator throws away $\mathbf{x}_{\{j\neq i\}}$ and averages the x_i samples, a sequence of zeros and ones. The new estimator throws away the observed x_i values(!), but does use $\mathbf{x}_{\{j\neq i\}}$. The resulting average of real numbers between zero and one can be much better behaved. If x_i is (nearly) independent of the other variables the new estimator gives (nearly) the correct answer from one sample, whereas the standard estimator is always noisy.

Does the conditioning trick improve variances in general? Without loss of generality we assume that the function of interest has zero mean. Under this assumption the variance of the estimator that conditions on a statistic h becomes

$$\begin{aligned}\mathrm{var}_{p(h)}\left[\mathbb{E}_{p(x|h)}[f(x)]\right] &= \sum_h p(h)\left[\sum_x p(x|h)f(x)\right]^2 \\ &\leq \sum_h p(h)\sum_x p(x|h)f(x)^2 = \sum_x p(x)f(x)^2.\end{aligned} \tag{2.11}$$

where the bound is an application of Jensen's inequality using the convexity of the square function. The final equality from summing out h means that

$$\mathrm{var}_{p(h)}\left[\mathbb{E}_{p(x|h)}[f(x)]\right] \leq \mathrm{var}_{p(x)}[f(x)], \tag{2.12}$$

the variance of the conditional estimator is never worse than the standard Monte Carlo estimator. This result applies under independent sampling from $p(x)$, unfortunately the result does not hold in general for correlated MCMC samples (Liu *et al.*, 1994).

Even if the variance is improved, we don't generally know when a conditional estimator will be computationally more efficient than the straightforward Monte Carlo estimator. As an extreme example $h(x)=x$ gives an "estimator" with zero variance, because the target expectation is computed exactly. Clearly the cost of computing the conditional expectation must be considered.

The use of equation (2.10) was suggested by Gelfand and Smith (1990), an influential paper that popularized Gibbs sampling in the statistics community. Their motivation

was essentially equation (2.11), cited as a version of the Rao-Blackwell theorem. The bound's requirement of independent samples was satisfied because the paper proposed running many independent Gibbs sampling runs. As this practice has fallen out of favor, it seems misleading to continue to call the method "Rao-Blackwellization", although some of the literature continues to do so. Despite the lack of Rao-Blackwell guarantees the estimator can often be justified empirically, as was done earlier by at least Pearl (1987).

2.2.2 Waste recycling

The Metropolis–Hastings algorithm has the undesirable property that for many proposal distributions a large fraction of proposed states are rejected from the final set of samples. As computations involving these rejected points were performed it seems a shame not to use this information in Monte Carlo estimators. The same observation followed our description of rejection sampling, for which the solution was to use the same proposal distribution in importance sampling. M–H proposal distributions are generally local in nature and unsuitable as importance sampling proposals. Waste recycling is a framework for constructing better estimators using the importance ratios computed while running MCMC.

A simple way to understand how to perform waste-recycling is to augment the stationary distribution over the current state x and the next proposal x' with an auxiliary variable $\hat{x}$. We declare that $\hat{x}$ was generated by copying one of x and x':

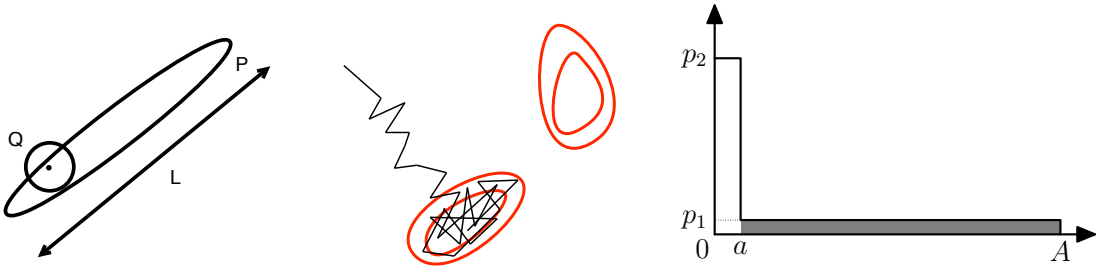
$$p(\hat{x}|x,x') = \begin{cases} \min\left(1, \frac{p(x')\,q(x \leftarrow x')}{p(x)\,q(x' \leftarrow x)}\right) & \hat{x} = x' \\ 1 - \min\left(1, \frac{p(x')\,q(x \leftarrow x')}{p(x)\,q(x' \leftarrow x)}\right) & \hat{x} = x\,. \end{cases} \tag{2.13}$$

Alternatively any other rule that gives $\hat{x}$ the same stationary distribution as x can be used. Now we could use draws of $\hat{x}$ for estimators instead of x. We can also average estimators under the conditional distribution of $\hat{x}$ given x and x':

$$\begin{aligned} \mathbb{E}_{p(x)}[f(x)] &= \mathbb{E}_{q(x' \leftarrow x)p(x)}\left[\sum_{\hat{x} \in \{x, x'\}} f(\hat{x})\, p(\hat{x}|x,x')\right] \\ &= f(x) + \big(f(x') - f(x)\big) \min\left(1, \frac{p(x')\,q(x \leftarrow x')}{p(x)\,q(x' \leftarrow x)}\right). \end{aligned} \tag{2.14}$$

This is just the conditional estimator, equation (2.10) from the previous section, applied to the auxiliary distribution $p(x, x', \hat{x}) = p(\hat{x}|x,x')q(x' \leftarrow x)p(x)$.

We hesitate to assign credit for this estimator to any particular author. It, or closely related estimators can be found in various independent sources, including at least Kalos and Whitlock (1986), Tjelmeland (2004) and Frenkel (2004). Waste-recycling is not necessarily better than the straightforward estimator, let alone worth the computa-



(a) Slow diffusion (b) Burn-in and mode finding (c) Balancing density and volume

Figure 2.1: Challenges for Markov chain exploration.

tional expense. However, there is empirical evidence that it can be useful, dramatically so in the case of Frenkel (2004). Casella and Robert (1996) also derived a "Rao-Blackwellized" estimator for using all the points drawn by the Metropolis algorithm. However their algorithm has an $\mathcal{O}(S^2)$ cost in the number of steps of the Markov chain and, unsurprisingly, has not been widely adopted.

2.3 Convergence

No review of Markov chains would be complete without discussing their convergence properties. If the simulation is initialized at an arbitrary starting location, how long must we wait before we can treat the chain's states as samples from the stationary distribution?

Often Metropolis proposals have a local nature; the typical step-size σ is limited by a need to maintain a reasonable acceptance rate. The amount of time it takes to explore a distance of length L by a diffusive random walk scales like $(L/\sigma)^2$ (figure 2.1a). This offers a rule of thumb to understand sampling within a mode.

An arbitrarily chosen initial state is usually very improbable under the target distribution. Reaching a mode in high dimensions can take a long time with sophisticated optimizers, let alone a Markov chain simulation. Analyzing the chain can identify when the statistics of the sampler settle down, allowing the initial exploratory phase to be discarded. Such diagnostics could be severely misleading if there are multiple modes as in figure 2.1b. In general applications there is no way of knowing if a Markov chain has yet to find the most important regions of a probability distribution.

There are also more subtle problems with convergence that do not relate to finding modes or step sizes. Figure 2.1c illustrates a distribution with a pillar at $0 < x < a$. Points within this mode have higher probability density than any state within the shaded plateau $a < x < A$. But given the large size of the plateau, few proposals will suggest moving to the pillar. Initializing the chain within the pillar using an optimizer would not help. Here $p_2 = 1/(2a)$ and $p_1 = 1/(2(A-a))$, the pillar and plateau have

equal probability mass, yet only a small fraction of proposals from the pillar to the plateau can be accepted — the acceptance rate is $a/(A-a)$ for symmetric proposals. A proposal distribution that knew the probability masses involved (e.g. i.i.d. sampling $q(x' \leftarrow x) = p(x')$) would move rapidly between the modes; but this would involve already having the knowledge that MCMC is being used to discover.

There is a theoretical literature on Markov chain convergence. But for general statistical problems it is difficult to prove much about the validity of any particular MCMC result. In later chapters we make occasional use of a standard diagnostic tool to compare the convergence properties of chains. But in general we would also run the sampling code on a cut-down version of the problem where exact results are available or simple importance sampling will work. In inference problems it is a good idea to check that a sampler is consistent with ground truth when learning from synthetic data generated from the prior model. Geweke (2004)'s method for "getting it right" is a more advanced way to check consistency between samplers for the prior and posterior. Such diagnostics are also a guard against errors in the software implementation, a possibility considered seriously even by experts (Kass *et al.*, 1998).

2.4 Auxiliary variable methods

As emphasized in the introduction, Monte Carlo is usually best avoided when possible. In tractable problems computations based on analytical solutions will usually be far more efficient. One would think this means we should analytically marginalize out variables wherever possible.

Auxiliary variable methods turn this thinking on its head. Given a target distribution $p(x)$, we introduce auxiliary variables z such that $p(x) = \int p(x, z)\ \mathrm{d}z$. We could just sample from $p(x)$, corresponding to analytically integrating out z from $p(x, z)$. Instead, auxiliary variable methods instantiate the auxiliary variables in a Markov chain that explores the joint distribution. Surprisingly, this can result in a better Monte Carlo method.

An alternative way of introducing extra variables is to make the target distribution a conditional of a joint distribution $p(x) = p(x|\beta = 1)$. The whole $p(x, \beta)$ distribution is explored, but only samples where $\beta = 1$ are retained. This seemingly wasteful procedure can also yield better Monte Carlo estimates. An example is simulated tempering, discussed later in subsection 2.5.1.

2.4.1 Swendsen–Wang

The Swendsen–Wang algorithm (Swendsen and Wang, 1987) is a highly effective algorithm for sampling Potts models (subsection 1.1.3) with a small number of colors q. The

algorithm is important in its own right (we use it in chapters 4 and 5) and as a significant development in auxiliary variable methods. Edwards and Sokal (1988) provided a scheme for constructing similar auxiliary variable methods for a wider class of models. They also identified the "Fortuin-Kasteleyn-Swendsen-Wang" (FKSW) auxiliary variable joint distribution that underlies the algorithm.

The FKSW joint distribution is over the original Potts color variables $\mathbf{s}=\{s_i\}$ on the nodes of a graph and binary bond variables $\mathbf{d}=\{d_{ij}\in\{0,1\}\}$ present on each edge:

$$p(\mathbf{s},\mathbf{d}) = \frac{1}{\mathcal{Z}_{\mathrm{P}}(J,q)} \prod_{(ij)\in\mathcal{E}} \left[(1-p_{ij})\delta_{d_{ij},0} + p_{ij}\delta_{d_{ij},1}\delta_{s_i,s_j}\right], \qquad p_{ij}\equiv(1-e^{-J_{ij}}). \quad (2.15)$$

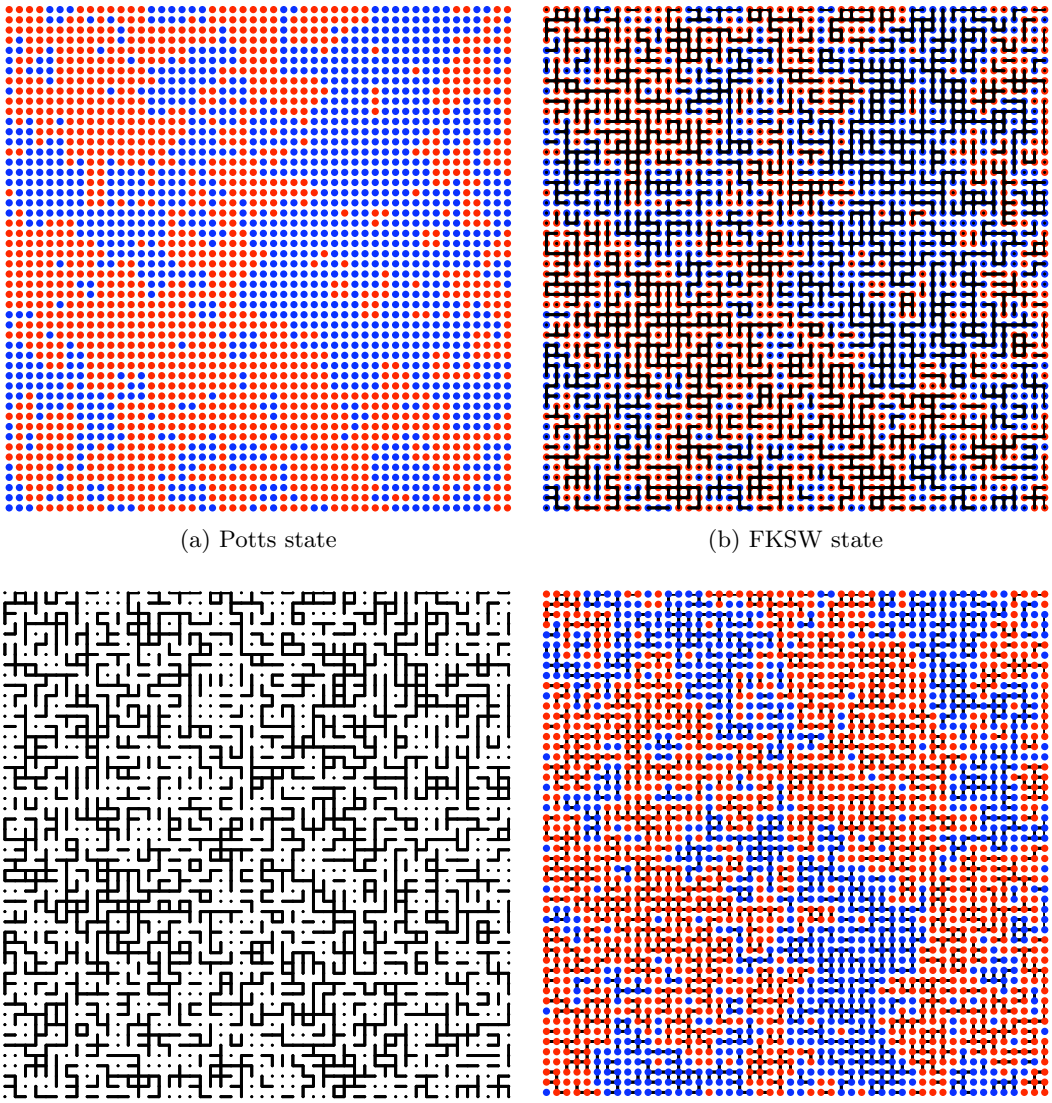
As long as all couplings J_{ij} are positive the marginal distribution over $\mathbf{s}$ is the Potts distribution, equation (1.4). The marginal distribution over the bonds is the random cluster model of Fortuin and Kasteleyn (1972):

$$p(\mathbf{d}) = \frac{1}{\mathcal{Z}_{\mathrm{P}}(J,q)} \prod_{(ij)\in\mathcal{E}} p_{ij}^{d_{ij}}(1-p_{ij})^{1-d_{ij}} q^{C(\mathbf{d})}, \quad (2.16)$$

where $C(\mathbf{d})$ is the number of connected components in a graph with edges wherever $d_{ij}=1$.

The algorithm of Swendsen and Wang (1987) performs block Gibbs sampling on the joint model by alternately sampling from $P(\mathbf{d}|\mathbf{s})$ and $P(\mathbf{s}|\mathbf{d})$. This also allows a sample from any of the three distributions — Potts $p(\mathbf{s})$, random cluster $p(\mathbf{d})$, or FKSW $p(\mathbf{s},\mathbf{d})$ — to be converted into a sample from one of the others. The algorithm is illustrated in figure 2.2. While implementing Swendsen–Wang, we were grateful for the efficient percolation code available in Newman and Ziff (2001).

It is commonly stated that Swendsen–Wang only allows positive J_{ij} couplings, as presented here. In fact Swendsen and Wang described the extension to negative bonds, which also follows easily from Edwards and Sokal's generalization. The resulting algorithm only forms bonds between edges where $J_{ij}<0$ if the adjacent sites have different colors. Colors joined by these negative bonds must remain different when choosing a new coloring. For binary systems two colorings of a cluster are possible: the previous coloring and its inverse. When there are many strong negative interactions the entire system gets locked into one of two configurations, whereas many configurations are probable. Swendsen–Wang can still be very useful when only some connections have negative weights: Stern *et al.* (2005) provide a nice example in the context of modeling the game of Go. For a wider view of related methods in the context of spatial statistics see Besag and Green (1993).



(a) Potts state (b) FKSW state

(c) Random cluster state (d) New FKSW state

Figure 2.2: The Swendsen–Wang algorithm. (a) As an example we run the algorithm on a binary Potts model on a square lattice. (b) Under the FKSW conditional $p(\mathbf{d}|\mathbf{s})$ bonds are placed down with probability $p_{ij} = 1 - e^{1-J_{ij}}$ wherever adjacent sites have the same color. (c) Discarding the colors gives a sample from the random cluster model. (d) Sampling from $p(\mathbf{s}|\mathbf{d})$ involves assigning each connected component or "cluster" a new color uniformly at random, giving a new FKSW state. Discarding the bonds gives a new setting of the Potts model. This coloring is dramatically different from the previous one. In contrast a sweep of single-site Gibbs sampling can only diffuse the boundaries of the red and blue regions by roughly the width of a single site.

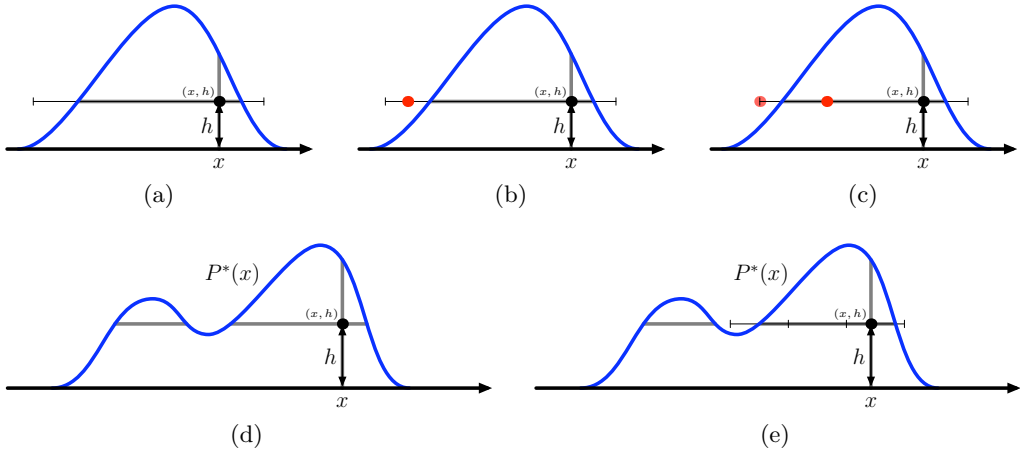


Figure 2.3: Slice sampling. (a)–(c) show a procedure for unimodal distributions: after sampling h from its conditional distribution an interval is found enclosing the region where $H(x) > h$. Samples are drawn uniformly from this interval until one underneath the curve is found. Points outside the curve can be used to shrink the interval, this is an adaptive rejection sampling method. (d) Sampling uniformly from the slice is difficult for multi-modal distributions. (e) One valid procedure uses an initial bracket of width σ that encloses x but is centered uniformly at random. This bracket is extended in increments of σ until it sticks outside the curve at both ends. This can cut off regions of the slice, but applying the previous adaptive rejection sampling procedure still leaves the uniform distribution on the slice invariant.

2.4.2 Slice Sampling

All Metropolis methods make use of a proposal distribution $q(x' \leftarrow x)$. In continuous spaces such distributions have step-size parameters that have to be set well. Almost all proposals are rejected if step-sizes are too large; overly small step-sizes lead to slow diffusive exploration of the target distribution. In contrast, a self-tuning method, which has less-important or even no step-size parameters would be preferable.

Slice sampling (Neal, 1997, 2003) is a method with one auxiliary variable that fits into the framework of Edwards and Sokal (1988). The auxiliary variable does not directly help mixing but allows the use of relatively easy-to-implement algorithms that allow self-tuning while generating a valid Markov chain. Like Gibbs sampling a slice sampling chain has no rejections: it always moves when sampling a continuous distribution. Slice sampling works by stepping back to the view of figure 1.5a: sampling involves drawing points uniformly underneath a curve $H(x) = p^*(x)$. Rather than drawing points i.i.d. as in rejection sampling, a Markov chain explores this uniform distribution over the original variables x and a height variable $0 < h < H(x)$.

Slice sampling algorithms alternate between updating h and one or more of the original variables. The height has a simple conditional distribution from which it can be resampled, $h \sim \text{Uniform}[0, H(x)]$. Conditioned on h the stationary distribution over x is uniform over the 'slice' that lies under the curve: $H(x) \geq h$. When the distribution is

unimodal this conditional can be sampled by rejection sampling, see figure 2.3(a)–(c).

More generally we use transition operators that leave a uniform distribution on the slice stationary. One of the operators introduced by Neal is briefly explained in figure 2.3e. Like most Metropolis methods it uses a step-size parameter σ, which would ideally match the length-scale of the problem, L. When $\sigma \gg L$ the initial search region shrinks exponentially. When $\sigma \ll L$ the search region requires $\mathcal{O}(L/\sigma)$ computations to grow out. This is much better than the $\mathcal{O}(L/\sigma^2)$ steps required by Metropolis to explore the region by a random walk.

Neal also provides a slice sampling operator that can expand its search region exponentially quickly. Skilling and MacKay (2003) provide a version with no step-size: it starts with a large search region, which shrinks with a fast exponent. Both of these operators are slightly harder to implement than the simple version sketched in figure 2.3.

We now propose a simple, we believe novel, method that can shrink slice sampling's search region more efficiently. The method is a generalization of subsection 2.1.3, where we factored the target density into $p(x) \propto \pi(x)L(x)$. Here we also introduce two auxiliary variables: $p(h_1|x) = \text{Uniform}[0, \pi(x)]$ and $p(h_2|x) = \text{Uniform}[0, L(x)]$. The new method follows any of the standard slice sampling algorithms, but defines the slice by requiring both $\pi(x) \geq h_1$ and $L(x) \geq h_2$.

This version of slice sampling still satisfies detailed balance with respect to the stationary distribution $p(x)$. We first identify all of the quantities involved in the slice sampling procedure. The algorithm starts with a point $x \sim p(x)$. Then two auxiliary variables h_1, h_2 are generated. The search procedure, e.g. figure 2.3e, will create an interval around x and possibly some unacceptable points and adaptations of the interval. We collectively refer to all of the intermediate intervals and rejected points as S. Finally an acceptable point x' is found and adopted. The joint probability of all of these quantities is:

$$\begin{aligned} T(x', S, h_1, h_2 \leftarrow x) \cdot p(x) &= \underbrace{1/\pi(x)}_{p(h_1|x)} \underbrace{1/L(x)}_{p(h_2|x)} p(S|h_1, h_2, x) p(x'|S, h_1, h_2, x) \cdot \tfrac{\pi(x)L(x)}{\mathcal{Z}} \\ &= p(S|h_1, h_2, x) p(x'|S, h_1, h_2, x)/\mathcal{Z}. \end{aligned} \tag{2.17}$$

All of the standard slice-sampling procedures are designed to ensure that x' is only accepted as the final point if this expression is symmetric in x and x'. Therefore, $T(x', S, h_1, h_2 \leftarrow x) \cdot p(x) = T(x, S, h_1, h_2 \leftarrow x') \cdot p(x')$. Summing this expression over S, h_1 and h_2 shows that the operator satisfies detailed balance.

Multiple auxiliary variables sometimes allow very fast mixing, as in Swendsen–Wang. However, in this context the slice defined by the two auxiliary variables h_1 and h_2 will typically be smaller than under standard slice sampling. This is likely to increase the mixing time of the chain. Neal warns about this problem with reference to Damien *et al.* (1999). Our motivation is computational: $\pi > h_1$ can be checked first, and only

if it satisfies the slice constraint need $L > h_2$ be checked. Thus schemes that start with a large range and rapidly shrink a bracket around the slice can do so with less computation if $\pi(x)$ can reject some unreasonable points while being cheap to compute.

2.4.3 Hamiltonian Monte Carlo

Physical simulations often work by discretizing time and approximately computing the result of following the *Hamiltonian dynamics* of the system being modeled. These dynamics exploit gradients of a target stationary probability density, not just pointwise evaluations as in the Metropolis algorithm. Theoretically, gradients only ever cost a constant multiple of the computer time needed to evaluate a function (Bischof and Bücker, 2000), and are a much richer source of information. The difficulty with simulation work is that inaccuracies accumulate rapidly unless time is discretized very finely, which has a large computational cost.

The Metropolis algorithm targets only an equilibrium distribution over states. The method doesn't need this distribution to be the result of an underlying dynamical system. Instead the algorithm evolves according to a proposal distribution. The progress of the Metropolis algorithm often resembles a diffusive random walk. In contrast a particle evolving under Hamiltonian dynamics will naturally move in persistent trajectories across its state-space.

These two areas of physical simulation research were combined in *Hybrid Monte Carlo* (Duane *et al.*, 1987). Hamiltonian dynamics are simulated approximately as a Metropolis proposal, which is accepted or rejected. This allows persistent, rapid motion across a state space without damaging the equilibrium distribution of the chain through discretization errors. If the target probability distribution does not correspond to a Hamiltonian system then auxiliary variables are introduced to create a fictitious system that can be simulated.

While Hamiltonian dynamics is time reversible, some care is required to ensure the discretized version is also reversible. Neal (1993) reviews a variety of Hamiltonian based Monte Carlo methods. See MacKay (2003) for another review with simple example code.

Hybrid Monte Carlo crossed into a statistical setting through work on Bayesian neural networks (Neal, 1992, 1996b), where it is extremely successful. Hamiltonian Monte Carlo methods continue to appear in recent work, especially in Gaussian process modeling, although they could be adopted more widely.

2.5 Annealing methods

Simulated annealing (Kirkpatrick *et al.*, 1983) is a heuristic for minimizing a cost function $E(x)$ with isolated local optima. The Metropolis algorithm is run on a probability distribution derived from the "energy" E,

$$p_k(x) = \frac{p_k^*(x)}{\mathcal{Z}(\beta_k)} = \frac{1}{\mathcal{Z}(\beta_k)}\pi(x)e^{-\beta_k E(x)} \quad . \tag{2.18}$$

A base measure $\pi(x)$ is often omitted (set to uniform), but including this term ensures p_k is defined when $\beta = 0$. Initially the inverse temperature β_k is set very low so that the distribution is diffuse. The distribution is then slowly 'cooled' by increasing β_k towards infinity. Eventually all of the probability mass will be concentrated on the global optimum of $E(x)$. There can be no guarantee that a particular sampling procedure will actually find the global optimum; in general it is not possible to do better than an exhaustive search. However, the heuristic has proved useful in a variety of applications.

Monte Carlo is not interested in optima. But if $p(x) \propto L(x)\pi(x)$ and $E(x) = -\log L(x)$ then $\beta_k = 1$ returns the target distribution of interest and distributions with $\beta_k < 1$ give more diffuse distributions. Figure 2.4 shows an example of a sequence of distributions. It might be that isolated modes at $\beta = 1$ are difficult to explore by available Markov chain operators. These modes join and disappear at lower β. These more diffuse distributions will typically be easier to explore. This section reviews algorithms designed to exploit this observation to provide better samples from multi-modal target distributions. It turns out that all of these methods can also provide estimates of the target distribution's normalizing constant. A full discussion of normalizing constants is deferred to chapter 4.

Notation: All of the algorithms in this section and several later in the thesis use a sequence of distributions. We have adopted the following conventions throughout:

- $p_0 = \pi$ is a base distribution, usually easy to explore or amenable to direct sampling. If p_k are defined by inverse temperatures then $\beta_0 = 0$.
- There are K intermediate distributions $\{p_k\}_{k=1}^K$ between the base and target distributions.
- $p_{K+1} = p$ is the target distribution. If p_k are defined by inverse temperatures then $\beta_{K+1} = 1$.

The sequences are usually defined using temperatures as in equation (2.18), but other methods for creating intermediate distributions between the base and target distributions may be used.

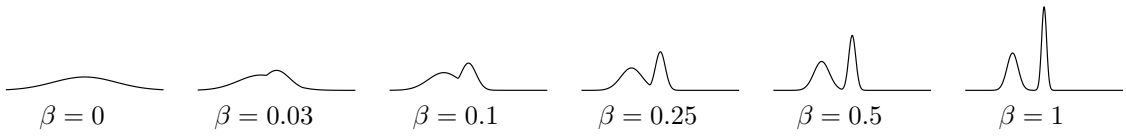


Figure 2.4: The effect of annealing. At $\beta = 0$ the distribution is equal to a convenient base distribution. As the inverse temperature increases the distribution morphs into the target distribution at $\beta = 1$.

2.5.1 Simulated tempering / Expanded Ensembles

A coherent way of dealing with annealing in MCMC simulations was independently discovered under the names *expanded ensembles* (Lyubartsev *et al.*, 1992) and *simulated tempering* (Marinari and Parisi, 1992). The idea is to bring the choice of intermediate distribution p_k into the joint distribution under exploration:

$$p(x, k) \propto w_{\mathrm{ST}}(k)\, p_k^*(x), \tag{2.19}$$

where $w_{\mathrm{ST}}(k)$ are user-chosen weights attached to each temperature level. The conditional distribution at temperature $\beta_k = 1$ returns the original distribution of interest: $p(x|\beta_k = 1) = p(x)$. A Markov chain at equilibrium returns correlated samples from the correct distribution by only recording states when $\beta = 1$. The hope is that the extra computation is justified by the decrease in auto-correlation of the Markov chain.

The extent to which simulated tempering improves a chain's mixing time depends heavily on the target distribution and the Markov chains used. One general characteristic of the algorithm is the amount of time it takes to move the temperature index k between its extreme values. Usually Metropolis proposals $k' \leftarrow k \pm 1$ are used because large changes in temperature are unlikely to be accepted. The adjacent levels must be similar enough for these changes in k to be accepted frequently, this sets a minimum number of temperature levels K. Even if all moves were accepted k would be undergoing a slow random walk. It will take at least $\mathcal{O}(K^2)$ steps to bring information from the rapidly mixing $k = 0$ distribution to the target $k = K + 1$ distribution.

The extra cost of the algorithm depends on the fraction of time spent at the other temperature levels. At equilibrium each level has marginal probability:

$$p(k) \propto \sum_x w_{\mathrm{ST}}(k)\, p_k^*(x) = w_{\mathrm{ST}}(k)\, \mathcal{Z}(\beta_k). \tag{2.20}$$

The partition function $\mathcal{Z}(\beta_k)$ can easily vary by orders of magnitude across temperature levels. In order for simulated tempering to spend a reasonable amount of time at each temperature level we could set $w_{\mathrm{ST}}(k) \approx 1/\mathcal{Z}(\beta_k)$. Except of course that this ideal weighting is unknown: e.g., $\mathcal{Z}(1)$ is the unknown normalization of the target distribution.

A practical implementation of simulated tempering must be an iterative process where

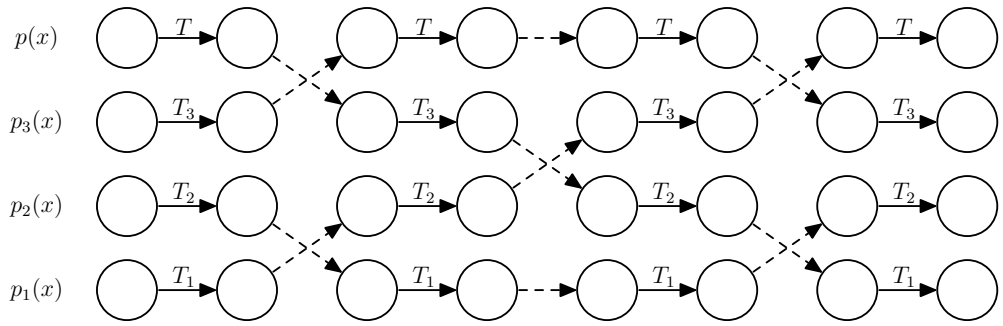


Figure 2.5: An illustration of the idea behind a family of "parallel tempering" algorithms. Each column represents the states of a Markov chain. Each states in the first row have marginal distribution $p(x)$, the other rows have stationary distributions bridging between this distribution and a vague distribution for which exploration is easy. MCMC proceeds by traditional simulation of each of the independent states and (dotted arrows) Metropolis proposals to exchange the states between adjacent levels.

the $w_{\mathrm{ST}}(k)$ are improved over a series of preliminary runs.

2.5.2 Parallel tempering

Parallel tempering[1] is a popular alternative to simulated tempering based on a different joint distribution. While simulated tempering uses a union space consisting of the temperature and the original state space, parallel tempering simulates on a product space. That is, replicas of the original state space exist concurrently for each of the stationary distributions $\{p_k\}$:

$$p(\{x^{(k)}\}) = \prod_{k=1}^{K+1} p_k(x^{(k)}). \tag{2.21}$$

As the states at each level are independent under the target distribution, independent transition operators T_k with corresponding stationary distributions p_k can be applied. For the higher temperature chains to communicate the results of their freer motion to the lower temperature chains interactions are also required.

Figure 2.5 illustrates one possible way of evolving a product space ensemble. The state of the Markov chain consists of an entire column of variables. These evolve using standard MCMC operators or by swaps between states that are accepted or rejected according to the Metropolis rule. The swap proposals can be scheduled in a variety of ways but, as with simulated tempering, information will propagate amongst levels by a random walk or slower.

By construction the amount of computer time spent on each level is completely under the user's control. In particular no weights $w_{\mathrm{ST}}(k)$ are required, which is one source of

[1] This method has several names and independent authors, its history is documented by Iba (2001b).

the method's relative popularity. Another feature of coexisting chains is the possibility of using proposals based on sophisticated interactions between states, e.g. Wang and Swendsen (1988). These advantages are at the expense of an obvious increase in memory requirements and a harder-to-gauge cost for bringing a larger system to equilibrium.

2.5.3 Annealed importance sampling (AIS)

Annealed importance sampling (Neal, 1998a, 2001) looks more like the original simulated annealing for optimization than the above ensemble methods. An initial point x_0 is drawn from p_0 a vague 'high temperature' base distribution. Further points $x_1, x_2, \ldots, x_K$ result from 'cooling' by evolving through a sequence of K Markov chain transition operators $T_1, \ldots, T_K$ whose stationary distributions $p_1, \ldots, p_K$ become sharper at each iteration[2].

The ensemble of $K+1$ points generated in the algorithm is denoted by $X = \{x^{(k)}\}_{k=0}^{K}$. We treat the procedure that generated them as a proposal which has the following joint distribution:

$$Q(X) = p_0(x_0) \prod_{k=1}^{K} T_k(x_k \leftarrow x_{k-1}). \tag{2.22}$$

But what is this a proposal for? The marginal distribution over the final point $q(x_K) = \sum_{x_{k<K}} Q(X)$ is not the target distribution and is probably intractable. We cannot directly compare $q(x_K)$ with $p(x_K)$, instead we must create an ensemble target distribution $P(X)$ that can be compared to $Q(X)$. Under this target model, x_K is drawn from the distribution of interest $p = p_{K+1}$. The remaining, auxiliary quantities are drawn using a sequence of reverse Markov chain transitions such that

$$P(X) = p_{K+1}(x_K) \prod_{k=1}^{K} \widetilde{T}_k(x_{k-1} \leftarrow x_k), \tag{2.23}$$

where $\widetilde{T}_k$ are reverse operators as described in section 1.4. Figure 2.6 illustrates both the ensembles $P(X)$ and $Q(X)$.

Samples of X from Q can be assigned importance weights with respect to P in the usual way (subsection 1.3.2),

$$\begin{aligned} w^*(X) &= \frac{P^*(X)}{Q^*(X)} \\ &= \frac{p^*_{K+1}(x_K)}{p^*_0(x_0)} \prod_{k=1}^{K} \frac{\widetilde{T}_k(x_{k-1} \leftarrow x_k)}{T_k(x_k \leftarrow x_{k-1})} = \frac{p^*_{K+1}(x_K)}{p^*_0(x_0)} \prod_{k=1}^{K} \frac{p_k(x_{k-1})}{p_k(x_k)} \\ &= \prod_{k=0}^{K} \frac{p^*_{k+1}(x_k)}{p^*_k(x_k)}. \end{aligned} \tag{2.24}$$

[2] In Neal's papers the indexes of the points started at $n-1$ and ran down to 0. The presentation here is designed to be consistent with similar algorithms found in later chapters.

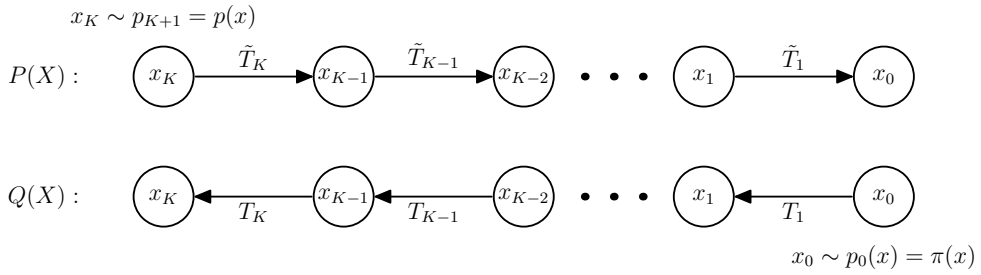


Figure 2.6: Annealed importance sampling (AIS) is standard importance sampling applied to an ensemble $P(X)$. This generative process starts with a draw from the target distribution $p(x) = p_{K+1}(x)$, which is intractable. Importance sampling uses proposals drawn from $Q(X)$, which starts at a simple distribution $\pi(x) = p_0(x)$.

Samples weighted in proportional to $w^*(X)$ can be used as though they came from P. Typically only x_K is of interest, which under P comes from the target $p_{K+1} = p$. The weights themselves are an unbiased estimate of $\mathcal{Z}_P/\mathcal{Z}_Q = \mathcal{Z}(\beta_{K+1})/\mathcal{Z}(\beta_0)$ as in standard importance sampling, equation (1.13).

A valid choice for all of the T_k is the "do nothing" operator, which attaches all of its probability mass to staying still: $T_\varnothing(x' \leftarrow x) = \delta(x' - x)$. Then $Q(X)$ just corresponds to drawing from p_0 (and copying the results), similarly $P(X)$ only draws from p_{K+1}. As one might expect, the importance weights collapse down to those of standard importance sampling of a target distribution $p = p_{K+1}$ with proposals from the base distribution $q = p_0 = \pi$. This highlights that the Markov chains do not need to mix well for AIS to be valid, but poor mixing operators will not improve over standard importance sampling.

A feature of AIS is that the importance weights do not require details of the transition operators T_k. This means that a given ensemble $X^{(s)}$ will always be given the same importance, regardless of whether the T_k mix immediately by drawing from their stationary distributions p_k, or in fact do nothing like $T_\varnothing$. The down-side to this behavior is that even with ideal transition operators there is always a mismatch between $P(X)$ and $Q(X)$. Under perfect mixing the final point is proposed from p_K, not the target distribution p_{K+1}.

The annealed importance sampling technique was first described in the physics literature by Jarzynski (1997). However, earlier still was the method of tempered transitions, which contains the same core idea.

2.5.4 Tempered transitions

Tempered transitions (Neal, 1994, 1996a) defines a reversible Markov chain transition operator leaving a distribution of interest, p_{K+1}, stationary. Thus its use is in MCMC

rather than importance sampling. It uses pairs of mutually reversible base transitions $\hat{T}_k$ and $\check{T}_k$ with corresponding stationary distributions $\{p_k\}_{k=1}^K$.

Given a current state $\hat{x}_K$ a candidate state $\check{x}_K$ is proposed through a sequence of states drawn as follows:

Generate $\hat{x}_{K-1}$ from $\hat{x}_K$ using $\hat{T}_K$.
Generate $\hat{x}_{K-2}$ from $\hat{x}_{K-1}$ using $\hat{T}_{K-1}$.
$\cdots$
Generate $\bar{x}_0$ from $\hat{x}_1$ using $\hat{T}_1$.
Generate $\check{x}_1$ from $\bar{x}_0$ using $\check{T}_1$.
$\cdots$
Generate $\check{x}_{K-1}$ from $\check{x}_{K-2}$ using $\check{T}_{K-1}$.
Generate $\check{x}_K$ from $\check{x}_{K-1}$ using $\check{T}_K$.

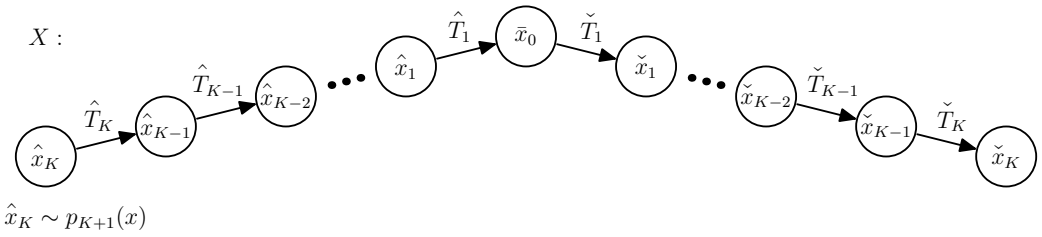
This generating sequence is illustrated in figure 2.7a. The candidate state is accepted with probability

$$a\left(\hat{x}_K, \cdots, \check{x}_K\right) = \min\left[1, \prod_{k=1}^{K} \frac{p_k(\hat{x}_k)}{p_{k+1}(\hat{x}_k)} \frac{p_{k+1}(\check{x}_k)}{p_k(\check{x}_k)}\right] , \tag{2.25}$$

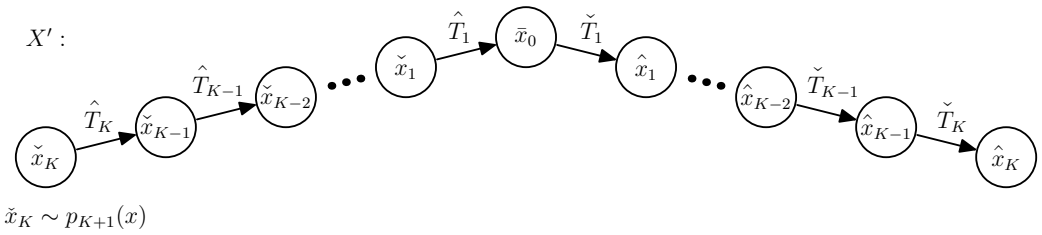
note that the normalizations of p_k cancel so are not needed. One way to derive this acceptance rule it is to identify the whole joint distribution of figure 2.7a as the target of the sampler. If the order of the states were reversed, figure 2.7b, $\check{x}_K$ would be the new state generated at equilibrium. A few lines of manipulation results in equation (2.25) for the Metropolis acceptance probability for this proposal. The intermediate states can be discarded because they are resampled at the next iteration.

Typically p_1 is much more vague than p_{K+1} which allows easier movement around the middle of the proposal sequence. If there are many intermediate distributions then every p_k can be made close to its neighbors $p_{k\pm1}$. This suggests each state visited during the proposal should be close to equilibrium under each of the transition operators and that the final state should nearly be drawn from p_K, which is close to p_{K+1}. Thus the acceptance rate can be high for large K.

One might expect that deterministically raising and lowering the temperature gives better performance than the Markov chain employed by simulated tempering. In fact the advantage is roughly cancelled by the need for a larger number of intermediate distributions, see Neal (1996a) for details. The main advantages of tempered transitions are: 1) it doesn't need a set of weights like simulated tempering; 2) it doesn't have parallel tempering's large memory requirements because equation (2.25) can be accumulated as the states are generated.



(a) Current state



(b) Proposed state

Figure 2.7: Tempered transitions. (a) The algorithm starts with a state $\hat{x}_K$ and generates the ensemble X. (b) The acceptance probability is for the proposal X', which by reversing all the states would make $\check{x}_K$ the new state generated under the equilibrium distribution p_{K+1}.

2.5.4.1 Generalization to only forward transitions

In standard tempered transitions $\hat{T}_k$ and $\check{T}_k$ are mutually reversible, here we consider two modified algorithms. A "forwards" tempered transition operator, $\mathcal{T}$, uses $\hat{T}_k = \check{T}_k = T_k$ to form the proposal distribution Q, where each T_k is any transition operator leaving p_k stationary. A "reverse" tempered transition operator, $\widetilde{\mathcal{T}}$, uses $\hat{T}_k = \check{T}_k = \widetilde{T}_k$ to form the proposal distribution $\widetilde{Q}$, where each $\widetilde{T}_k$ is the reverse transition operator corresponding to T_k. Both algorithms use the same acceptance rule as before. Neal (1996a) shows that the operator defined by standard tempered transitions satisfies detailed balance. Here we present a similar proof to show that $\mathcal{T}$ and $\widetilde{\mathcal{T}}$ are mutually reversible with respect to p_{K+1}.

As in standard tempered transitions we start at $\hat{x}_K$ and generate a sequence of states $\hat{x}_{K-1}, \hat{x}_{K-2}, \ldots, \bar{x}_0, \ldots, \check{x}_K$. To simplify notation both $\hat{x}_0$ and $\check{x}_0$ are taken to mean $\bar{x}_0$ throughout. The probability of generating a particular sequence of states given the initial state using only forward transitions T_k is:

$$Q\left(\check{x}_K, \cdots, \hat{x}_{K-1} \leftarrow \hat{x}_K\right) = \left[\prod_{k=1}^{K} T_k(\hat{x}_{k-1} \leftarrow \hat{x}_k)\right] \left[\prod_{k=1}^{K} T_k(\check{x}_k \leftarrow \check{x}_{k-1})\right], \tag{2.26}$$

$$= \left[\prod_{k=1}^{K} \frac{\widetilde{T}_k(\hat{x}_k \leftarrow \hat{x}_{k-1}) p_k(\hat{x}_{k-1})}{p_k(\hat{x}_k)}\right] \left[\prod_{k=1}^{K} \frac{\widetilde{T}_k(\check{x}_{k-1} \leftarrow \check{x}_k) p_k(\check{x}_k)}{p_k(\check{x}_{k-1})}\right]$$

$$= \left[\prod_{k=1}^{K} \widetilde{T}_k(\check{x}_k \leftarrow \check{x}_{k-1})\right] \left[\prod_{k=1}^{K} \widetilde{T}_k(\hat{x}_{k-1} \leftarrow \hat{x}_k)\right] \frac{p_{K+1}(\check{x}_K)}{p_{K+1}(\hat{x}_K)} \left[\prod_{k=1}^{K} \frac{p_{k+1}(\hat{x}_k)}{p_k(\hat{x}_k)} \frac{p_k(\check{x}_k)}{p_{k+1}(\check{x}_k)}\right].$$

Using equation (2.25) and equation (2.26) we can compute the equilibrium probability of starting at $\hat{x}_K$, generating the sequence $\hat{x}_{K-1}, \cdots, \check{x}_K$ and accepting the move with operator $\mathcal{T}$:

$$
\begin{aligned}
&\mathcal{T}(\check{x}_K, \cdots, \hat{x}_{K-1} \leftarrow \hat{x}_K) p_{K+1}(\hat{x}_K) \\
&= Q(\check{x}_K, \cdots, \hat{x}_{K-1} \leftarrow \hat{x}_K) p_{K+1}(\hat{x}_K) a(\hat{x}_K, \cdots, \check{x}_K) \\
&= \left[\prod_{k=1}^{K} \widetilde{T}_k(\check{x}_k \leftarrow \check{x}_{k-1})\right] \left[\prod_{k=1}^{K} \widetilde{T}_k(\hat{x}_{k-1} \leftarrow \hat{x}_k)\right] p_{K+1}(\check{x}_K) \min\left[1, \prod_{k=1}^{K} \frac{p_{k+1}(\hat{x}_k)}{p_k(\hat{x}_k)} \frac{p_k(\check{x}_k)}{p_{k+1}(\check{x}_k)}\right] \\
&= Q(\hat{x}_K, \cdots, \check{x}_{K-1} \leftarrow \check{x}_K) p_{K+1}(\check{x}_K) a(\check{x}_K, \cdots, \hat{x}_K) \\
&= \widetilde{\mathcal{T}}(\hat{x}_K, \cdots, \check{x}_{K-1} \leftarrow \check{x}_K) p_{K+1}(\check{x}_K). \qquad (2.27)
\end{aligned}
$$

This is the equilibrium probability of observing the corresponding reverse move under $\widetilde{\mathcal{T}}$. Summing both sides over all intermediate quantities $\{\hat{x}_k, \check{x}_k\}_{k \neq K}$ gives

$$
\mathcal{T}\left(\check{x}_K \leftarrow \hat{x}_K\right) p_{K+1}(\hat{x}_K) = \widetilde{\mathcal{T}}\left(\hat{x}_K \leftarrow \check{x}_K\right) p_{K+1}(\check{x}_K), \qquad (2.28)
$$

which shows that both $\mathcal{T}$ and $\widetilde{\mathcal{T}}$ leave p_{K+1} invariant. Therefore, we only need one of them, $\mathcal{T}$ say. This means that the standard tempered transitions algorithm still leaves the target distribution invariant if forward operators T_k are used for both $\hat{T}_k$ and $\check{T}_k$; implementing reverse operators is not required.

2.5.4.2 Generalization to a single pass

The second half of tempered transitions is identical to annealed importance sampling. Neal (2001) states that the major difference between the two methods is the requirement for an upward pass in tempered transitions, making the updates twice as expensive. In fact, this upward pass is not necessary.

Looking back at figure 2.6 there is no reason why the AIS ensemble $X' \sim Q$ cannot be used as a Metropolis–Hastings proposal for updating a state X drawn, at equilibrium, from P. This is a *Metropolis independence sampler*, where the proposal does not depend on the current state. This method of turning an importance sampler into a Markov chain operator dates back at least to Hastings (1970).

Importance sampling seems a more natural way to construct estimators than the Metropolis independence sampler. Notice that an independence sampler sometimes rejects states when the current state is more important. The states kept by the algorithm will depend on the order in which the proposals are made, which is arbitrary as they are independent. It seems likely that treating the proposals identically through importance weights will be better, and theoretically this is usually the case (Liu, 1996).

Tempered transitions and annealed importance sampling are really two applications of the same joint distribution. The main difference is whether an MCMC operator or an importance sampler is preferable. In isolation the importance sampler may yield better estimators, but the MCMC operator can be combined with other MCMC operators. Hastings argued that unweighted MCMC samples are more interpretable, although one could also obtain approximate unweighted samples by resampling from a distribution over points proportional to their importance-weights.

2.6 Multicanonical ensemble

The multicanonical ensemble (Berg and Neuhaus, 1992) is another method for allowing easier movement around a state space. Rather than introducing extra distributions, the original probability distribution is reweighted:

$$p_{\mathrm{MC}}(x) \propto w_{\mathrm{MC}}(E(x)) \cdot e^{-E(x)}, \quad E = -\log p^*(x). \tag{2.29}$$

The weight function $w_{\mathrm{MC}}(x)$ only depends on a state's *energy* E, which is the log of the state's original unnormalized probability. The weights are chosen such that the marginal distribution over energies $p(E)$ is uniform, as far as possible. If the energy is unbounded a cut-off is introduced.

Why set the distribution over energies uniform? As with the temperature-based algorithms, more time is spent in regions of low probability, which helps join isolated modes. But the multicanonical ensemble *is* just a heuristic, there is nothing fundamental about this distribution. In fact, visualizing such a distribution can be somewhat surprising. Figure 2.8 shows the result of reweighting a simple one-dimensional distribution. For simplicity we created a set of energy bands and used a procedure that only ensured that the distribution over bands was uniform.

The most probable states in figure 2.8c are in the extreme tail of the original distribution: few states occupy the lowest energy bands, so any particular one must be visited frequently. In many high-dimensional problems the lowest probability states will be plentiful and the tails will be given much less prominence. However, when there are only a relatively small number states at a particular energy the very high probabilities assigned by the multicanonical ensemble can be very useful.

Berg and Neuhaus invented the multicanonical ensemble to deal with distributions containing 'bottlenecks', regions with small numbers of states that separate regions with significant probability mass. Strangely, bottle-neck states can be problematic even when they are individually more probable than many of the states typical under the distribution. A Markov chain is not able to spend much time exploring the states between important regions if there are many more states elsewhere with much higher total probability. This phenomenon is known as an *entropic barrier*, and can be overcome by

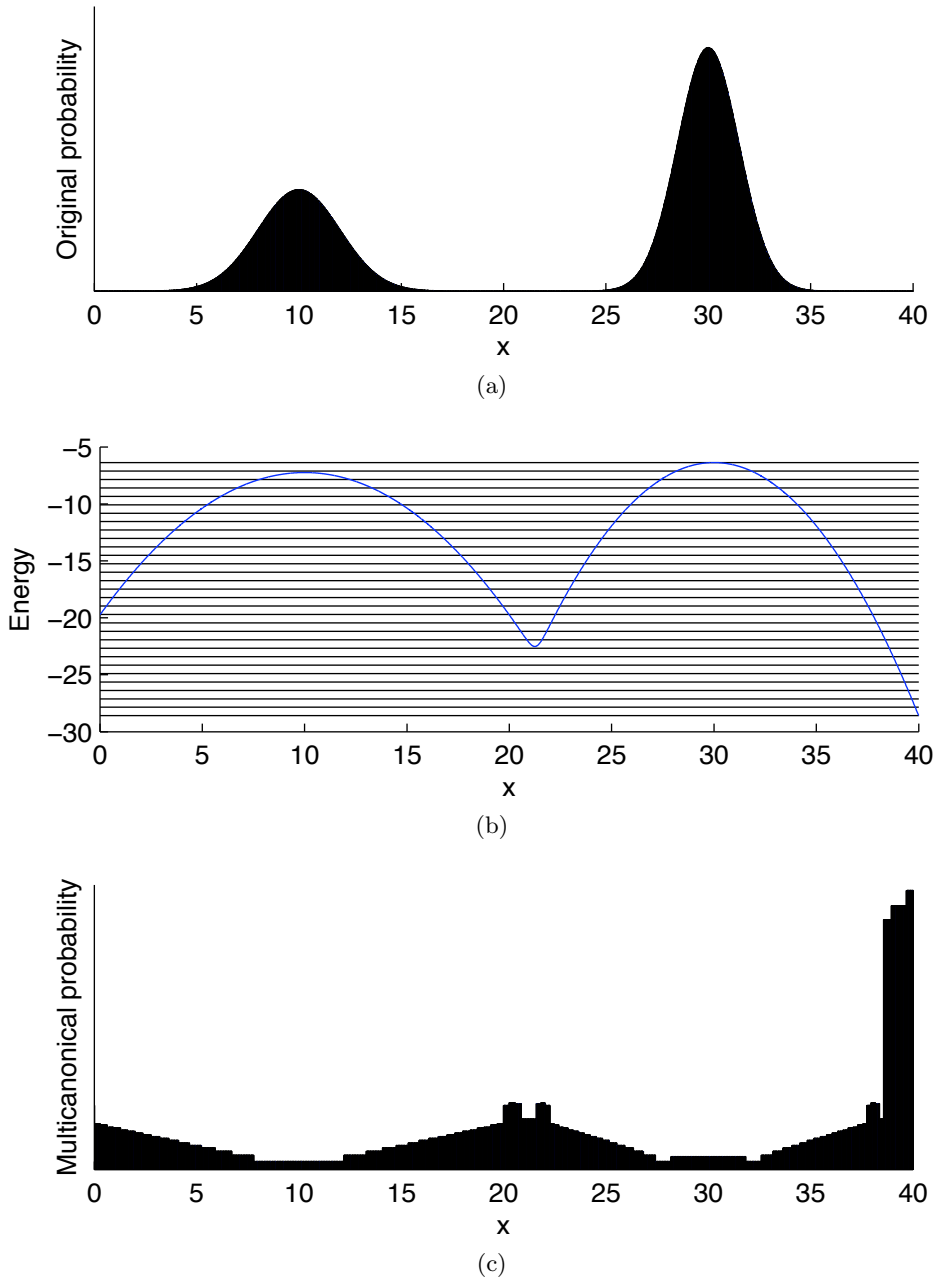


Figure 2.8: An example of a "multicanonical ensemble". (a) The target distribution of interest. (b) The energy function that defines this distribution. The horizontal lines mark bands of energy used to construct the new ensemble. (c) The final figure results from setting the probability of each state proportional to the reciprocal of the number of states in its energy range. Drawing a state from this "multicanonical" distribution and reading off its energy band gives a uniform distribution over the bands.

Some artifacts in the plot are due to the precise way bands were chosen. But note that the sharp peaks in the middle of the multicanonical plot are due in part to the truncation of the original distribution at $x = 0$. Setting the distribution over energies is a global operation and can have unexpected consequences.

a multicanonical ensemble, which makes states with abnormal energies — low or high — much more probable.

As in simulated tempering, finding the weights or weighting function for the multicanonical method is difficult. The original application of the multicanonical ensemble used prior knowledge to construct a parametric approximation that would give a near-uniform distribution over energy. General applications often use histogram methods that bin ranges of energy together and iteratively fit weights for each bin on the basis of preliminary runs. For more advanced methodologies there are whole theses largely dedicated to these methods (e.g. Smith, 1995; Ferkinghoff-Borg, 2002).

Unlike annealing methods, the multicanonical method never simulates the target distribution. However correlated samples from p_{MC} can be used to construct importance sampling estimators of any quantity of interest. This is explored further in chapter 4.

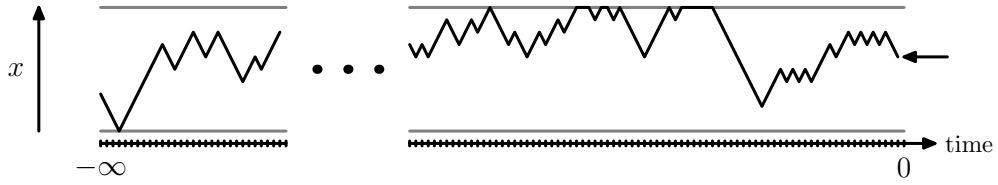
2.7 Exact sampling

Exact or perfect sampling simply means drawing a sample from a target distribution. Markov chain Monte Carlo uses approximate samples: the distribution over the position at the sth step of a Markov chain depends, perhaps weakly, on its starting position at $s=0$.

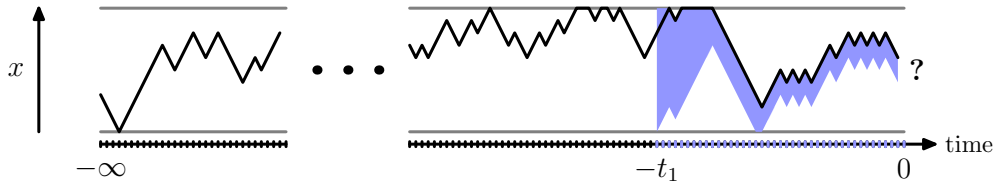
Surprisingly it is sometimes possible to obtain exact samples from Markov chains. A sketch of the technique known as *coupling from the past* (CFTP) (Propp and Wilson, 1996) is given in figure 2.9. The core idea is to find the location of a Markov chain that has run for an infinitely long time without having to simulate its entire length. Typically exact sampling algorithms require the ability to track sets of states through a random, possibly large, number of Markov chain steps. This is technically challenging, but provides a means to draw samples from some distributions such as large Potts models and some spatial point processes where traditional means fail. See Wilson (1998) for reviews, examples and more algorithms for exact sampling with Markov chains.

Is exact sampling actually useful? Some users like to see samples, which can give insight into a problem. But in many statistical applications there is no interest in the samples themselves, only estimates of expectations. Given some exact samples, the variance of an estimator could be improved by running Markov chains from the samples to produce more (correlated) samples. If it is possible to start with exact samples some comfort is derived from removing any bias associated with the 'burn in' of the chain. But it is still better to run a small number of long Markov chains than investing in equilibrating many independent ones (section 2.2).

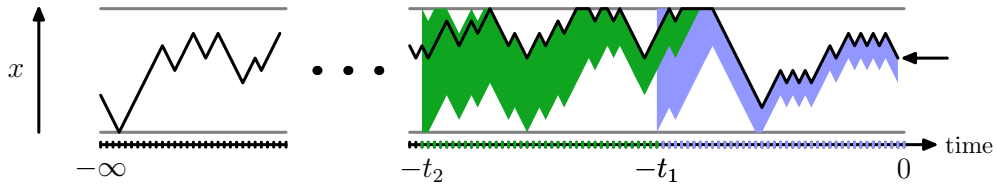
Several algorithms in chapter 5 rely heavily on exact sampling. These require an exact sample from a different distribution at each iteration. Using approximations is dangerous as biases can accumulate at every iteration, so Markov chains potentially



(a) A sample is at the end of an infinite chain.



(b) We try to find the end with a finite number of random numbers.



(c) Eventually, after a random amount of time, we find the sample

Figure 2.9: Coupling from the past (CFTP) overview. Each dash on the time axis represents a random number used by the chain. Everything must be consistent with the infinite chain in (a). We try to locate the final state by looking at (drawing) just some of the random numbers. (b) Before time $-t_1$ we know nothing, so the chain could be anywhere, we evolve a bound on the state space forward to time zero. At first this might only localize the sample. (c) We look at more random numbers further into the past until a bound collapses to a single state, we then follow it to the exact sample at time zero. It takes a random amount of computation before the bound can be made tight.

have to be run for a very long time. Exact sampling gives a stopping rule, and one that guarantees zero bias.

2.7.1 Exact sampling example: the Ising model

Coupling from the past relies on being able to prove where the final state of a Markov chain lies without knowing the starting location. To demonstrate that this is sometimes possible we describe the *summary state* technique, following Childs *et al.* (2001). The method is originally due to Huber (1998), who uses the name *bounding chains*.

Our task is to find the setting of an Ising model, a Potts model (subsection 1.1.3) where each variable takes on $q = 2$ settings $s_i \in \{+1, -1\}$. We assume the system evolved under Gibbs sampling (subsection 2.1.2) from time $t = -\infty$ to $t = 0$. Each variable was updated in order at each time step according to the following algorithm:

$$\begin{aligned} u_{i,t} &\sim \text{Uniform}[0,1] \\ p_+ &= p\big(s_i^{(t)} = +1 \,\big|\, s_{j<i} = s_{j<i}^{(t)},\ s_{j>i} = s_{j>i}^{(t-1)}\big) \\ &= \frac{\exp\big(\sum_{j\in\mathcal{E}_i} J_{ij}(\delta_{+1,s_j} - 1)\big)}{\sum_{s_i'} \exp\big(\sum_{j\in\mathcal{E}_i} J_{ij}(\delta_{s_i' s_j} - 1)\big)} \\ \text{if } u_{i,t} < p_+ &\text{ then set } s_i^{(t)} \leftarrow +1 \text{ else } s_i^{(t)} \leftarrow -1. \end{aligned} \tag{2.30}$$

In general, the setting of variable i at time t depends on the random number $u_{i,t}$ and the settings of its neighbors $j \in \mathcal{E}_i$.

The summary states algorithm starts by drawing all the random numbers $u_{i,t}$ for some finite range of time $t = -T \ldots 0$. We don't know what happened before time $-T$, so for each variable we set $s_i^{(-T)} = ?$. We then follow the Gibbs sampling updates from $t = -T$ to $t = 0$, setting the states to ?'s whenever the result of an update depends on the settings of states we do not know:

$$\begin{aligned} &u_{i,t} \sim \text{Uniform}[0,1] \\ &p_{+,\max} = \max_{s_j : s_j = ?} p\big(s_i^{(t)} = +1 \,\big|\, s_{j<i} = s_{j<i}^{(t)},\ s_{j>i} = s_{j>i}^{(t-1)}\big) \\ &p_{+,\min} = \min_{s_j : s_j = ?} p\big(s_i^{(t)} = +1 \,\big|\, s_{j<i} = s_{j<i}^{(t)},\ s_{j>i} = s_{j>i}^{(t-1)}\big) \\ &\text{if } u_{i,t} < p_{+,\min} \text{ then set } s_i^{(t)} \leftarrow +1 \\ &\text{if } u_{i,t} > p_{+,\max} \text{ then set } s_i^{(t)} \leftarrow -1 \\ &\text{otherwise set } s_i^{(t)} \leftarrow ?. \end{aligned} \tag{2.31}$$

If some of the $s_i^{(0)}$ states are set to ? then we do not know the setting of our exact sample. We draw more of the random numbers $u_{i,t}$ from time $-T-1$ back to time $-2T$ and start again from $\mathbf{s}^{(-2T)} = ?$. The hope is that eventually, after doubling the number of Markov chain steps enough times, no ?'s will remain and we will know $\mathbf{s}^{(0)}$.

More sophisticated versions of the algorithm have been developed for models with $q > 2$ and for tracking dependencies among unknown states. When the connection strengths J are large it won't be possible to get rid of all the ?'s in practice. Better Markov chains than Gibbs sampling are required. While the details are considerably more involved, it is possible to bound the behavior of random-cluster samplers (Propp and Wilson, 1996) and carefully implemented Swendsen–Wang samplers (Huber, 2002).

2.8 Discussion and Outlook

In this chapter we have reviewed a subset of the established literature on Markov chain Monte Carlo methods. Most MCMC algorithms are underpinned by proving detailed balance, which is almost synonymous with the seminal algorithms of Metropolis *et al.* (1953) and Hastings (1970). Some choices still remain in the choice of acceptance rule and the construction of estimators. We presented an interpretation of "waste-recycling" estimators, which we will apply in a new setting in chapter 3. We also reviewed a "two-stage" acceptance rule in subsection 2.1.3, which trades off statistical optimality to give computational savings. We extended this idea to slice sampling in subsection 2.4.2.

Algorithms and proposal distributions tuned for particular applications are clearly important, but were beyond the scope of this review. Moreover, we did not discuss techniques that sample distributions with variable dimensionality (e.g. Green, 1995), or infinite dimensionality (e.g. Neal, 1998c). While important in some applications, or as a replacement for model selection techniques, they are not used by the remainder of the thesis. We do have a brief excursion into sampling from an infinite-dimensional distribution at the end of chapter 5, although this is more closely related to the work on exact sampling reviewed in the previous section.

Auxiliary variable methods have the potential to dramatically accelerate Markov chain exploration. Figure 2.2 demonstrated the large changes possible in a single step of Swendsen–Wang. Slice sampling doesn't make such dramatic moves, but it is an easy-to-apply method that we have found personally useful both in this thesis and in quickly testing hypotheses in discussions with colleagues. Similarly Hamiltonian Monte Carlo has proved personally useful in work outside this thesis (Murray and Snelson, 2006), and we feel it should be used more widely.

The use of annealing methods appears somewhat mysterious — especially in a statistical setting. It seems strange to sample at a range of "temperatures" in applications outside of physics. However, the annealing heuristic, figure 2.4, can be very effective in problems with multiple modes. It is less clear whether these distributions need to be simulated together as in parallel tempering or separately as in the other algorithms. We explore the issue of populations of samples in the next chapter. This work leads us to construct

new algorithms for constructing non-reversible Markov chains, operators that do not satisfy detailed balance.

Returning to annealing, chapter 4 explores the computation of normalizing constants. There we explore why standard MCMC is insufficient for solving this problem and the advantages of using multiple distributions as in annealing. We then provide a critical and comparative review of annealed importance sampling, the multicanonical ensemble and *nested sampling*, a new method by John Skilling. We will propose new procedures for running both nested sampling and annealed importance sampling.

Finally, all of the algorithms discussed so far rely on being able to evaluate a function proportional to the target distribution, $p^*(x) \propto p(x)$. It turns out that this assumption is not met by a class of statistical inference problems, which we label *doubly-intractable*. Chapter 5 is dedicated to constructing new algorithms for this problem. Their operation draws on ideas from all areas of this review, including annealing and exact sampling.

Chapter 3

Multiple proposals and non-reversible Markov chains

Markov chain Monte Carlo is ambitious: a huge amount is asked of a solitary point diffusing through the void of a complex high-dimensional space. The target stationary distribution may have multiple modes, a representative sample of which must be found. Also, somehow, the mass of each mode is (implicitly) estimated so that the correct amount of time is spent in each one. Even navigating a single mode can be difficult. A long standing idea is that *multiple* points evolving simultaneously should help. Communication amongst points might help find and explore the regions of high probability mass.

The simplest way to introduce multiple points is to target a product model $P(X) = \prod_{k=1}^{K} p(x_k)$. This is just like parallel tempering (subsection 2.5.2), except that each of the independent distributions are the same. Although each of the x_k variables are independent, a Markov chain operator used to update one of them can depend on the setting of the others at the same time step. This provides a limited but adapting source of information about length-scales in the problem and the positions of modes.

Adaptive Direction Sampling (ADS) is a name given to at least two algorithms. One of these updates a point by resampling from its conditional distribution (i.e. Gibbs sampling) along a direction suggested by another pair of points (Gilks *et al.*, 1994). This allows coupled movement of highly correlated variables, which would be difficult under standard axis-aligned Gibbs sampling. Initial experiments with ADS highlighted the need for many more particles than dimensions in order to equilibrate in all necessary directions. Then the burden of bringing K points all to equilibrium simultaneously removes the advantage. On the other hand some advantage is obtained with a small number of points when combined with standard axis-aligned updates to ensure mixing in the full space. I have been able to produce very similar results using a slice sampling based version of ADS as suggested in MacKay (2003). Related ideas are

still being actively pursued in the literature (Ter Braak, 2006; Christen and Fox, 2006). While general theoretical results seem elusive, a small number of parallel chains is often empirically useful and seems advisable in general.

This chapter is dedicated to exploring other Monte Carlo algorithms that attempt to leverage an advantage from multiple points. The first of these, "Population Monte Carlo" as in Cappé *et al.* (2004), is an existing importance sampling method described as a competitor to MCMC methods. We highlight some important differences with MCMC-based methods like ADS, not made apparent enough in the current literature. We then return to MCMC methods, looking at those not based on the ADS product model. These methods have a single Markov chain 'backbone', which branches to a set of points for consideration at each step. Ensembles of dependent points might allow more thorough local exploration. Section 3.2 reviews and extends Multiple-Try Metropolis (MTM), an algorithm for making good use of expensive proposal mechanisms. Section 3.3 covers ordered over-relaxation an existing extension to Gibbs sampling, which draws multiple points to improve navigation of highly correlated variables. We extend ordered over-relaxation from just Gibbs sampling to arbitrary transition operators and multiple Metropolis proposals.

3.1 "Population Monte Carlo"

Many Monte Carlo algorithms could be described as population methods; Iba (2001a) provides a review of some of them. This section refers to a specific method with the name "Population Monte Carlo" (PMC) as described by Cappé *et al.* (2004) and popularized in Robert and Casella (2004). It is described as an algorithm which can be iterated like MCMC, but unlike Markov methods can easily be adapted over time. Examples were presented in which replacing MCMC with PMC gave better results. In fact it is also easy to demonstrate the reverse. This section provides a critical review of PMC.

The template of a PMC method is given in algorithm 3.1. K particles are evolved in parallel. An arbitrary distribution q_{ks} is used to move each particle at each step. These distributions can depend on all the particles' entire histories. Unbiased estimates can be constructed from the proposed points, $\{\tilde{x}_k^{(s)}\}$, weighted by their importance weights. As usual (subsection 1.3.2) the weights are usually only available up to a constant. Consistent estimators are still available by using normalized weights.

The resampled points drawn in step 7 are a biased sample from the target distribution for finite K. For example, $K = 1$ would just give a sample from the proposal distribution. When K is large the points are approximately unbiased samples from the target distribution. It is hoped these will form a useful basis for constructing the

Algorithm 3.1 "Population Monte Carlo" as in Cappé *et al.* (2004)

1. **for** $s = 1$ to S
2. **for** $k = 1$ to K
3. Select proposal distribution q_{ks}
4. Generate $\tilde{x}_k^{(s)} \sim q_{ks}(x)$
5. Compute weight ${w_k}^{(s)} = p(\tilde{x}_k^{(s)})/q_{ks}(\tilde{x}_k^{(s)})$
6. **end for**
7. Resample K positions with replacement from $\{\tilde{x}_k\}$ with probabilities proportional to the weights giving this step's sample $(x_1^{(s)}, x_2^{(s)}, \ldots x_K^{(s)})$.
8. **end for**

proposal distributions at the next iteration. Estimators should be formed from the importance-weighted ensemble before the resampling step.

As PMC is a framework for importance sampling the q_{ks} distributions must have appreciable probability mass across the entire support of the target distribution. While Cappé *et al.* do mention the need for heavy tails, they simultaneously suggest that PMC is a replacement for MCMC:

> *"Population Monte Carlo borrows from MCMC algorithms for the construction of the proposal, from importance sampling for the construction of appropriate estimators,..."*

These two goals seem mutually exclusive. Metropolis–Hastings proposals are typically local in nature, we do not expect them to capture the entire target distribution. Sometimes we might propose dramatic moves, perhaps from a kernel density estimate (Tierney and Mira, 1999; Warnes, 2000). But such proposals do *not* need to be a good global approximation to the target distribution for MCMC to work. Such transition operators can be combined with local diffusion for robustness. Importance sampling has no such choice — using M–H proposals which tend to cut off some or most of the target density can have disastrous effects. In PMC some particles would occasionally have very large weights. This would give high variance estimators and the occasional collapse of all or most particles to a single point in the resampling step.

Both Cappé *et al.* (2004) and Robert and Casella (2004) describe a "Mixture PMC" algorithm, which seems very likely to suffer from these failure modes of importance sampling. The algorithm suggests choosing q_{ks} from a set of spherical Gaussian distributions with a variety of widths. The probability of adopting each width is adapted according to their past ability to make proposals that survive the resampling step.

To demonstrate the problem with mixture PMC we use a version of the "funnel" distribution described by Neal (2003). The distribution is over a zero-mean Gaussian distributed variable v with standard deviation 3 and several independent, zero-mean Gaussian variates $\{x_i\}$ each with variance $\exp(v)$. This type of structure can easily exist in hierarchical Bayesian models. The original funnel distribution used nine x_i

variates and was exceedingly challenging for simple Metropolis sampling. Slice sampling largely alleviated this problem. Here we make the distribution easier to handle, using only four x_i variates so that Metropolis sampling will equilibrate more quickly.

We used spherical Gaussian proposals with standard deviations selected uniformly from an *ad hoc* set of 12 proposal widths: $\{0.05, 0.1, 0.2, 0.5, 1, 2, 4, 8, 10, 12, 14, 16\}$. Metropolis was run 1000 times with 100,000 steps in each run, which produced a good histogram for the marginal distribution of v, figure 3.1a. Only in the tails of the distribution are some frequencies not quite right. It would have been better to sample fewer, longer runs (section 2.2), but we wished to provide a direct comparison to 100,000 PMC steps with 1000 particles. Figure 3.1b shows a histogram of samples obtained from step 7 of PMC algorithm 3.1. These samples do not match the true marginal distribution, although we can expect this distribution to be biased. Figures 3.1 (c)–(f) show estimates of v's marginal distribution based on the importance weights of the particles before resampling. The PMC estimates are terrible for all four variants of the algorithm described in the figure. None of the Gaussian distributions are good importance samplers for this distribution. A combination of the estimators does not work well either.

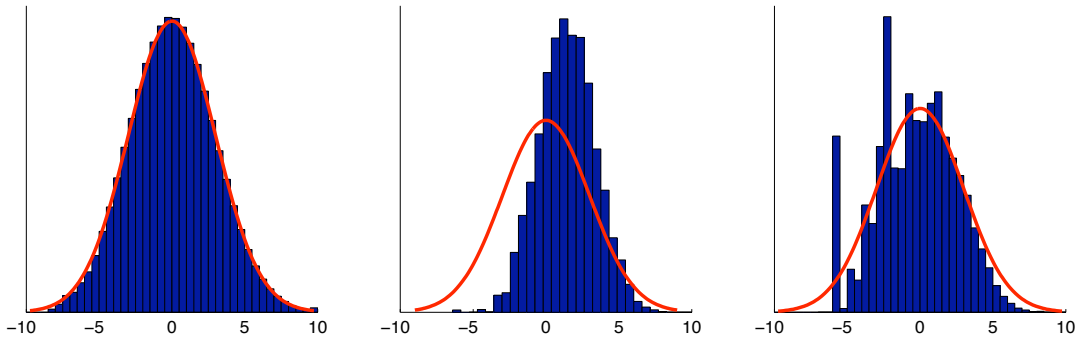
Large importance weights during the PMC runs often caused many particles in the population to move together into a new region of space. Unlike Cappé *et al.* we do not see this as a sign that the space is being explored. Instead it indicates the underlying high variance of the importance sampling estimator. When the distributions are inappropriate for importance sampling there is no guarantee that the time spent by the population within a region reflects the target distribution. Instead the times between large importance weight events depend on the details of the tails of the proposal distributions. In this case adapting the distributions did not solve the problem. In any case, favoring proposals that can lead to extremely large weights seems unwise.

Iterated importance sampling algorithms are attractive in some contexts. However, using proposals from MCMC algorithms within importance sampling is not generally advisable. We now return to algorithms that stay within the MCMC framework.

3.2 Multiple-Try Metropolis

This section concerns the Multiple-Try Metropolis method (MTM) of Liu *et al.* (2000). We review the method and provide a simple interpretation as an evolving joint distribution including auxiliary variables. This almost trivial rewriting of the algorithm makes a possible improvement obvious and relates to the pivot-based Metropolis method introduced later in the chapter.

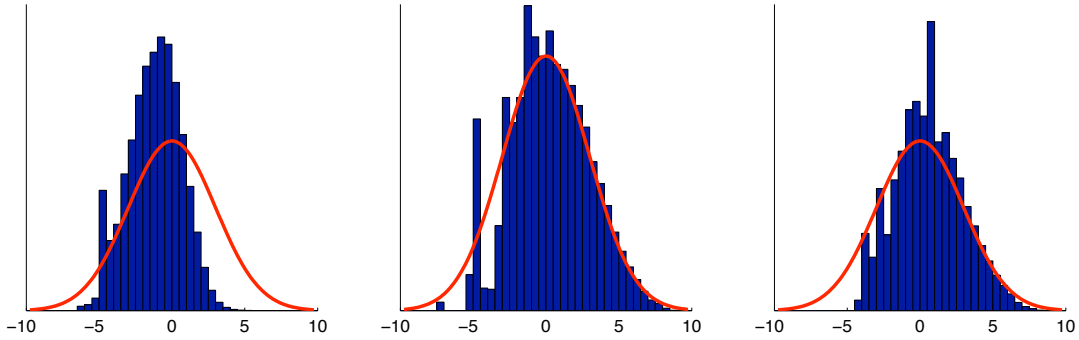
The MTM algorithm was motivated as follows: local proposal distributions $q(x' \leftarrow x)$ may lead to slow convergence, but longer range moves are rarely accepted. If multi-



(a) Metropolis

(b) PMC samples. q separate, adapted.

(c) PMC weights. q separate, adapted.



(d) PMC weights. q combined, adapted.

(e) PMC weights. q combined, fixed.

(f) PMC weights. q separate, fixed.

Figure 3.1: Histograms of samples of the v parameter from the funnel distribution. The curves show the correct marginal posterior. All samplers were initialized at $v = 0$, $\mathbf{x} = \mathbf{1}$. Both Metropolis and PMC used spherical Gaussian proposal distributions with the same set of proposal widths. For PMC these were 'adapted' as in Cappé *et al.* (2004) or chosen from a 'fixed' uniform distribution. The PMC importance weights were either computed using the proposal width that was used, 'q separate', as in Cappé *et al.* (2004) or using 'q combined', the mixture of Gaussians obtained by summing over the choice of width (Robert and Casella, 2004).

ple points are proposed from a long-range distribution then it is more likely that an acceptable one will be found. MTM, algorithm 3.2, is one way of allowing K draws from a proposal distribution. Figure 3.2 illustrates and motivates the procedure with a graphical model.

Algorithm 3.2 A single step of the Multiple-Try Metropolis Markov chain

1. Draw K i.i.d. proposals from $x_k \sim q(x_k \leftarrow x)$
2. Compute weights $w(x, x_k) = p(x)\, q(x_k \leftarrow x)\, \lambda(x, x_k)$
   λ is any symmetric function $\lambda(x, x_k) = \lambda(x_k, x)$
3. Choose one proposal $x' \in x_1, \ldots x_K$, using probabilities proportional to the weights
4. Generate a new set of $K-1$ points $x'_k \sim q(x'_k \leftarrow x')$ for $k = 1 \ldots K-1$ and define $x'_K = x$.
5. Compute $a(x', x) = \dfrac{w(x_1, x) + \cdots + w(x_K, x)}{w(x'_1, x') + \cdots + w(x'_K, x')}$
6. Assign $x \leftarrow x'$ with probability $\min(a, 1)$

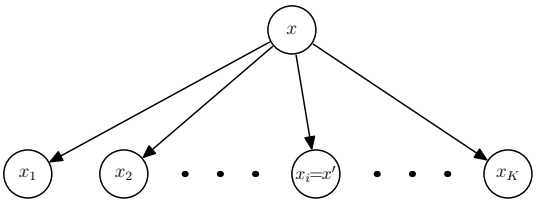
MTM is actually a family of algorithms defined by the λ function used at step 2 to weight the points. MTM(II) is the MTM algorithm with symmetric q and $\lambda = 1/q$ giving weights $w(x_k) \propto p(x_k)$. As various of choices of λ were reported to behave very similarly, we use MTM(II) in our experiments.

Figure 3.2's graphical model description immediately suggests an alternative algorithm. Step 1 of the MTM algorithm resamples $x_1, \ldots x_K$ from the joint distribution. This discards the setting already made by a previous iteration. In some circumstances keeping the existing proposals could save computer time. This would be achieved by omitting step 1 after the first iteration and updating step 6 to include $\{x_k\} \leftarrow \{x'_k\}$ in the assignment.

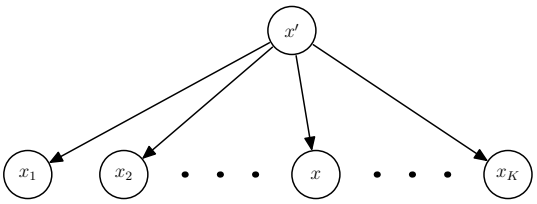
3.2.1 Efficiency of MTM

For a given proposal distribution $q(x_k \leftarrow x)$, each step of MTM is more expensive than a single Metropolis–Hastings proposal. Counting the number of target distribution evaluations suggests that the computational cost is roughly $2K$ times more for MTM. Various practical issues make the exact cost trade-off more complex. In many circumstances the time taken to evaluate $2K$ probabilities will grow sub-linearly with K. This may be because some intermediate results can be shared, or because related computations can be *vectorized* and computed efficiently on some types of computer. The sequential nature of MCMC often makes it difficult to vectorize operations, yet such code transformations can have surprisingly large benefits, especially in popular interpreted environments such as R, Octave or Matlab.

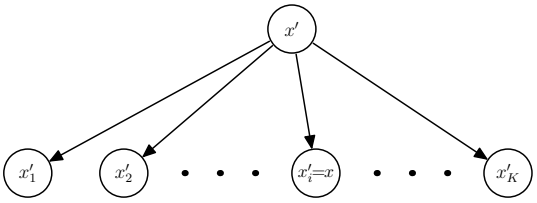
The results reported by Liu *et al.* (2000) suggest that MTM outperforms Metropolis without needing implementation-based code-tuning. One example involved simple



(a) K proposals are generated and one selected.



(b) A suggested swap makes x' the sample from $p(x)$



(c) Resampling all but x and x' makes the joint setting more plausible.

Figure 3.2: Multiple-Try Metropolis (MTM): (a) x is assumed to come from the target distribution $p(x)$. K proposals are drawn $x_k \sim q(x_k \leftarrow x)$. This defines a joint target distribution on which the Markov chain operates. Firstly one of the proposed values, x_i say, is chosen and labeled x'. (b) One could consider proposing a swap $x \leftrightarrow x'$, giving a joint state in which x' generated $\{x, x_{k\neq i}\}$. The acceptance probability for this proposal could typically be low: for large K it will often be obvious that x rather than x' generated the remaining variables. (c) MTM swaps x and x' and also resamples the remaining variables from their new stationary distribution $x'_{k\neq i} \sim q(x'_k \leftarrow x')$. The acceptance rule is simply Metropolis–Hastings applied to the proposal (a) $\rightarrow$ (c).

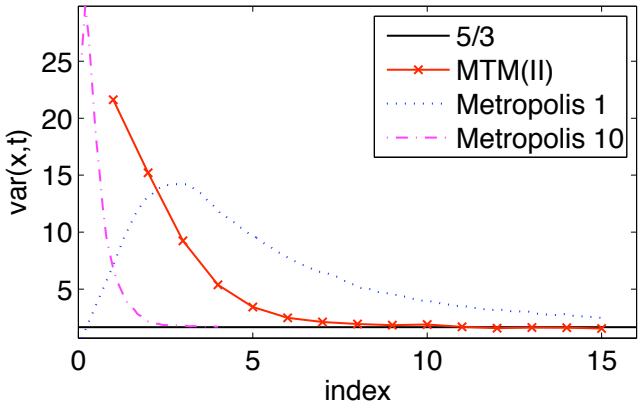


Figure 3.3: Performance of Multiple-Try Metropolis based on figure 2 from Liu *et al.* (2000). Samplers were run on a t-distribution as described in the main text. Plotted is $\mathrm{var}(x,t)$, the variance of a sampler's state after t steps from $x_0 = 9$, estimated from 5000 runs. The time taken for a chain's states to match the true variance of 5/3 is a measure of its *burn-in* time. Computer time was measured as follows: for MTM(II) with $K = 5$ `index` is the number of steps taken; for the Metropolis algorithms `index` is (optimistically) the number of steps divided by $2K$. The performance of Metropolis with $\sigma = 1$ and MTM(II) with $\sigma = 10$ closely reproduce the results from the original paper. Rerunning Metropolis with a Gaussian step-size of $\sigma = 10$ shows a much shorter burn-in time than MTM.

Gaussian-based proposals on a one-dimensional t-distribution with five degrees of freedom. The amount of computer time to reach equilibrium from an extreme fixed starting point ($x^{(0)} = 9$) appeared to be considerably less for MTM(II) with $\mathcal{N}(x, 10^2)$ proposals than for Metropolis with $\mathcal{N}(x, 1)$ proposals. As shown in figure 3.3 this claim is reproducible, but why were different proposal widths $\sigma = 1$ and $\sigma = 10$ used while making the comparison? Presumably a longer step-size was deemed appropriate for MTM as it has more attempts to find a good proposal. But our results show that Metropolis with $\sigma = 10$ approaches equilibrium (according to the measure used) faster than Metropolis with $\sigma = 1$ or MTM with $\sigma = 10$.

The amount of computer time needed to forget a particular atypical initialization is only one measure of performance. We also used R-CODA (Cowles *et al.*, 2006) to estimate the effective number of samples based on a spectral analysis of the time series. These are compared in table 3.1.

Table 3.1: Equilibrium efficiency of MTM and Metropolis

Method	Effective samples / proposals
i.i.d. sampling (control)	0.998 ± 0.002
M–H, $\sigma = 1$	0.065 ± 0.002
M–H, $\sigma = 10$	0.118 ± 0.001
MTM, $\sigma = 10$	0.036 ± 0.001

The acceptance rates for Metropolis–Hastings with $\sigma = 1$ and $\sigma = 10$ were 0.72 and 0.15 respectively. This suggests that the optimal step size is somewhere in between the two

values tried. MTM's acceptance rate was 0.45, close to ideal for both standard M–H and — according to Liu *et al.*'s experience — MTM. Despite this, MTM's sampling efficiency on this very simple example distribution is worse than M–H at both equilibrium sampling and initial burning in.

This is a negative result, but one worth noting. Using multiple parallel proposals is not, in our experience, worthwhile in itself. This makes sense: when proposals are made sequentially a good proposal can be accepted immediately and future proposals move from the new location. In contrast, only one point from a set of parallel proposals can be accepted, if there is more than one good point the rest are wasted.

The motivation for using multiple proposals cannot be to explore larger search regions. Instead we should consider MTM if making multiple proposals together is computationally cheaper than making them separately. The orientational bias procedure in Frenkel and Smit (2001, algorithm 22, section 13.1.2) is equivalent to MTM(II). In this context the proposals involve moving molecules. A molecule's probability depends on independent terms depending on its position and orientation. The orientational bias procedure proposes a single new position with several possible orientations. While the issue isn't discussed in detail, presumably it is sharing the positional energy computation across proposals that makes the algorithm worthwhile in molecular dynamics simulations.

A statistical example claiming benefit from multiple proposals is given by Qi and Minka (2002). They drew proposals from local Gaussian approximations based on the Hessian of the log posterior at the current sample location. In high-dimensional problems these proposals would be expensive to construct[1], $\mathcal{O}(D^3)$ to prepare a matrix factorization for a multivariate-Gaussian sampler. Drawing multiple samples is then cheaper, $\mathcal{O}(D^2)$. Even sequential sampling can take advantage of this, because the proposal will be reused until a move is accepted. One wonders if, given the effort expended to construct the distribution, drawing more samples than needed for acceptance could be useful. The reported accuracy for a given CPU time does appear to be slightly better by combining Hessian-based sampling with an MTM variant, multiple-importance try.

3.2.2 Multiple-Importance Try

MTM makes it more likely that the chain moves, but is wasteful of information: multiple evaluations of the probability density are made at each iteration but at most one of these points is used. Qi and Minka (2002) suggested a way to use all of the $\{x_k\}$ points and called their combined Hessian-based MTM method *Adaptive Multiple-Importance Try* (AMIT). We will refer to the part of the algorithm that concerns including all the points in MTM as *Multiple-Importance Try* (MIT).

[1] Qi and Minka also discuss cheap approximations to Hessians, but are still keen to get the best use from them however constructed.

MIT is the standard multiple-try Metropolis algorithm with a particular choice for post-processing the samples. If p can be evaluated then the samples are weighted by

$$w_k = \frac{p(x_k)}{q(x_k \leftarrow x)}. \tag{3.1}$$

When $p(x)$ is only known up to a constant the w_k are computed using the unnormalized version $p^*(x)$ and then made to sum to one. This is standard importance sampling using $q(x_k \leftarrow x)$ as a proposal distribution. The locations x of these importance samplers are chosen using MCMC.

Using local Gaussian approximations can lead to importance sampling estimators with large or even infinite variances. This suggests that MIT is likely to give erroneous results on high-dimensional problems. MIT will work when good global proposal distributions can be obtained, but we are unsure why MCMC would be an appropriate way to adapt these distributions. These are similar concerns to those discussed in section 3.1 regarding "population Monte Carlo".

3.2.3 Waste-recycled MTM

Waste recycling (subsection 2.2.2) is a method for using proposals that are not accepted as part of the Markov chain. Here we describe how to recycle the points from the MTM method that are normally unused. Waste recycling does not carry the risks of huge variances associated with using importance sampling estimators with Metropolis proposals.

We first take the MTM joint distribution identified in figure 3.2a,

$$p\left(x, \{x_k\}_{k=1}^K\right) = p(x) \prod_{k=1}^{K} q(x_k \leftarrow x), \tag{3.2}$$

and augment it with a new variable $\hat{x}$. The marginal distribution of the new variable should be the target distribution of interest. This could be achieved by copying x or picking one of the proposals at random and copying it according to the Metropolis–Hastings acceptance rule:

$$p\left(\hat{x}|x, \{x_k\}\right) = \begin{cases} a_k = \frac{1}{K} \min\left(1, \frac{p(x_k)\, q(x \leftarrow x_k)}{p(x)\, q(x_k \leftarrow x)}\right) & \hat{x} = x_k \\ 1 - \sum_k a_k & \hat{x} = x\,. \end{cases} \tag{3.3}$$

As in subsection 2.2.2, any quantity can be estimated by averaging over each of the $\{x, \{x_k\}\}$ settings weighted by their probability under this conditional distribution.

So far we are still discarding the $\{x'_k\}$ variables drawn by the algorithm. If the $(x', \{x'_k\})$ joint state were accepted we could draw a new auxiliary variable $\hat{x}' \sim p\left(\hat{x}'|x', \{x'_k\}\right)$ using the conditional distribution analogous to equation (3.3). To include both possible joint

settings in an estimator we use a further auxiliary variable $\hat{\hat{x}}$, which is equal to one of $\hat{x}$ or $\hat{x}'$. We could define the probability of $\hat{\hat{x}}=\hat{x}'$ according to the $\min(1,a)$ acceptance probability of the MTM algorithm. However, this would sometimes give zero weight to $\hat{x}$. In normal waste-recycling we do not mind ignoring the initial state as it was already used at the previous iteration, but the $\{x_k\}$ were created at this iteration[2]. Instead we use

$$p(\hat{\hat{x}}=\hat{x}'|x,\{x_k\},x',\{x'_k\}) = \frac{\sum_k w(x_k,x)}{\sum_k w(x_k,x)+\sum_k w(x'_k,x')}, \tag{3.4}$$

which also gives $\hat{\hat{x}}$ the correct stationary distribution. The final estimator averages over all states visited at each iteration based on the following identity for $\mathbb{E}_{p(x)}[f(x)]$:

$$\mathbb{E}_{p(x,\{x_k\},x',\{x'_k\})}\left[\sum_{\substack{\hat{x}\in\{x,\{x_k\}\}\\ \hat{x}'\in\{x',\{x'_k\}\}}} p\left(\hat{x}|x,\{x_k\}\right) p\left(\hat{x}'|x',\{x'_k\}\right) \sum_{\hat{\hat{x}}\in\{\hat{x},\hat{x}'\}} f(\hat{\hat{x}})\, p(\hat{\hat{x}}|x,\{x_k\},x',\{x'_k\})\right]. \tag{3.5}$$

The expression looks cumbersome, but when expanded out only costs $\mathcal{O}(K)$ to compute. The estimator can be implemented once and used as a "black-box".

The conditional probabilities of $\hat{x}$ and $\hat{x}'$, equation (3.3), require some backwards proposal probabilities not normally needed by MTM or MIT. This would remove the potential advantage when proposals are expensive, as in Qi and Minka (2002), but would not be a problem when, as in Frenkel and Smit (2001), the purpose of multiple proposals is shared computation of $p(x)$.

We compared the MTM's mean square error (MSE) in estimating $|x|$ to that obtained by the waste-recycled version (WR-MTM) on the same t-distribution example as before.

Table 3.2: Accuracy on t-distribution after 1000 proposals

Method	Mean Square Error (MSE)
M–H, $\sigma=1$	0.0119 ± 0.0010
M–H $\sigma=10$	0.0105 ± 0.0005
MTM, $\sigma=10$	0.0267 ± 0.0014
WR-MTM, $\sigma=10$	0.0167 ± 0.0008
MIT, $\sigma=10$	0.0047 ± 0.0002

Post-processing the MTM run with waste-recycling reduces the MSE considerably, but still not to the level of Metropolis–Hastings. Speedups from efficient implementation of parallel proposals would still be required to justify waste-recycled MTM. Treating the x_k proposals as draws from importance samplers, the MIT estimator, obtains the

[2]Unless we are applying the alternative algorithm suggested just before subsection 3.2.1.

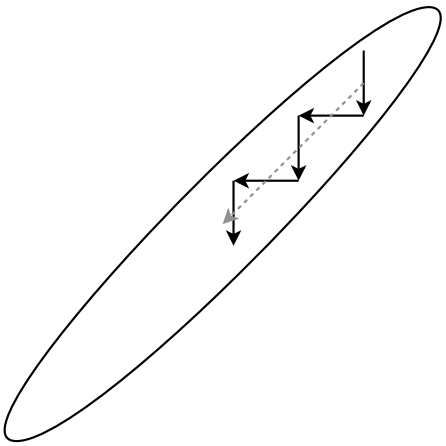


Figure 3.4: The idea behind successive overrelaxation for minimization: when performing component updates moving past the minimum along the current search direction can give faster progress along the direction that points to the optimum. Monte Carlo overrelaxation methods attempt to successively move to the other side of a component's conditional distribution, which can cause persistent motion along a more interesting direction.

best MSE. This reflects the fact that importance sampling is usually more appropriate than MCMC for estimating the statistics of simple 1D densities.

In higher dimensional problems, where MCMC is favored over simple importance sampling, the proposals will generally be more local and take longer to explore the target distribution. This situation was simulated by setting the proposal width to 0.1 and taking 10000 MTM steps with $K = 5$. Averaging over 1000 such runs the mean square error (MSE) of MTM in estimating $|x|$ was 0.051 ± 0.011 this wasn't hurt by waste-recycling, which had an MSE of 0.046 ± 0.005. There wasn't much to gain from using the extra samples as the MTM sequence is highly correlated. As such, recycling would not be worth the computational expense, but wouldn't be too harmful. The MIT importance sampling estimate had a significantly worse MSE of 0.131 ± 0.002. MIT does not necessarily reduce variance. In high-dimensional problems it could be much worse.

3.3 Ordered overrelaxation

Ordered overrelaxation (Neal, 1995, 1998b) is a technique to improve the convergence of Gibbs sampling for some distributions with strong positive correlations amongst its variables. A variety of MCMC methods exist that are inspired by successive overrelaxation in optimization, see figure 3.4. Ordered overrelaxation is one of these methods, considered here because it is based on a population of proposals.

Ordered overrelaxation is based on Gibbs sampling. A normal iteration of Gibbs sampling involves resampling x_d from its conditional distribution $p(x_d|\mathbf{x}_{\{d' \neq d\}})$. A (wasteful) way to implement this rule is to draw K samples from the conditional distribution and then pick one of them uniformly at random. We could leave the conditional dis-

tribution stationary by also including the current setting and choosing uniformly from $K+1$ settings. Moreover we need only apply a transition operator that leaves this uniform distribution over $K+1$ settings stationary. Ordered overrelaxation provides such an operator.

The ordered overrelaxation transition operator requires that the K+1 candidates for the d^{th} component can be ordered in some way. If they are real scalars we simply sort by numerical value. If only a partial ordering is available then ties are broken arbitrarily. After sorting the points they are relabeled in order:

$$s_0 \leq s_1 \leq \cdots \leq s_i = x_d \leq \cdots \leq s_K. \tag{3.6}$$

The operator chooses $x'_d = s_{K-i}$ as the next step in the Markov chain. As long as K is odd this rule will always pick one of the K new values. Because the rule is deterministic and reversible it clearly leaves a uniform distribution over the points stationary. Choosing $x'_d = s_{K-i}$ tends to move to the opposite side of the conditional distribution from x_d. This is precisely the goal of overrelaxation. The effect becomes stronger when increasing K. For very large K the point is moved almost deterministically from its current quantile q to $1-q$.

In naïve implementations any benefits from persistent motion is cancelled by the cost of drawing additional samples. Whether ordered overrelaxation is useful will depend on whether repeated draws, $s_k \sim T(s_k \leftarrow z)$, can be performed cheaply. Neal (1998b) discusses some circumstances in which this can be done. Another potential cost saving is to make the number of points, K, small when ordered overrelaxation is less useful. We now introduce a new procedure with this goal.

3.3.1 Adapting K automatically

Our new algorithm attempts to use fewer samples in ordered overrelaxation when the current position is close to the center of its conditional distribution. This is a situation where over-relaxation is typically unhelpful. It requires the user to set $K_{\min}$ and $K_{\max}$ parameters, giving the smallest and largest acceptable number of samples to draw per iteration.

The procedure is detailed in algorithm 3.3. The numbers of new points to the 'right' and 'left' of the current position are tracked as r and l. In an extreme situation, where the current position is at the very edge of the conditional distribution one of r and l will remain at zero and $K_{\max}$ points will be drawn. Conversely when x_d is near the median of the conditional distribution, the algorithm will typically return sooner.

After obtaining an unordered set of points from algorithm 3.3 the list is sorted together with the current point $x_d = s_i$ giving an ordered set $\{s_k\}_{k=0}^{K}$ as before. We now show that ordered overrelaxation still satisfies detailed balance using a variable K. To do this

Algorithm 3.3 Self size-adjusting population for ordered overrelaxation.
Generates number of points K and unordered set $\{s_k\}_{k=1}^{K}$.
Inputs: $K_{\min}$, $K_{\max}$, current location $\mathbf{x}$
Initialize: $l = 0$; $r = 0$

1. **for** $k = 1 \dots K_{\max}$
2. $\quad s_k \sim p(x_d|\mathbf{x}_{\{d' \neq d\}})$
3. $\quad$ **if** $s_k > x_d$ **then**
4. $\quad\quad r = r + 1$
5. $\quad$ **else if** $s_k < x_d$ **then**
6. $\quad\quad l = l + 1$
7. $\quad$ **end if**
8. $\quad$ **if** $(r \geq K_{\min}/2)$ and $(l \geq K_{\min}/2)$ **then**
9. $\quad\quad K = k$
10. $\quad\quad$ **return**
11. $\quad$ **end if**
12. **end for**

we need the equilibrium probability of starting at point $x_d = s_i$, generating a set $\{s_k\}$ and then deterministically transitioning to $x'_d = s_{K-i}$. The order in which the set of points was generated does not matter, although some orderings are not possible because algorithm 3.3 would terminate before generating the entire set. Let $C(l, r)$ be the number of orderings of $r+l$ points with r greater than x_d that can be generated by the algorithm. The equilibrium probability of all the points is $C(i, K–i) \prod_{k=0}^{K} p(s_k|\mathbf{x}_{\{d' \neq d\}})$. Given the symmetry $C(l, r) = C(r, l)$, this is also the probability of starting at $x'_d = s_{K-i}$, producing the same set of points and then deterministically moving to $x = s_i$. Summing over all intermediate points shows that the transition $x_d \leftrightarrow x'_d$ satisfies detailed balance.

3.4 Pivot-based transitions

Inspired by the ordered overrelaxation generalization of Gibbs sampling, we now introduce a new method which uses general Markov chain transition operators. We start from any pair of mutually reversible transition operators, $T(x' \leftarrow x)p(x) = \widetilde{T}(x \leftarrow x')p(x')$ which leave a desired target distribution, $p(x)$, stationary. A single reversible operator $T = \widetilde{T}$ will also suffice. Algorithm 3.4 takes the existing operator (pair) and constructs a new transition operator, $\mathcal{T}(x' \leftarrow x)$. This procedure is illustrated in figure 3.5. The pivot-based transition operator will favor moves to the opposite side of the region accessible by T, as defined by a user supplied ordering. The stationary target stationary distribution $p(x)$ is maintained without introducing any rejections into the chain (although $\mathcal{T}(x' = x \leftarrow x)$ will carry finite probability in some discrete settings, or if $T(x' = x \leftarrow x)$ is finite).

Each of the $\{s_k\}_{k=0}^{K}$ points in algorithm 3.4 are marginally distributed according to

Algorithm 3.4 The pivot-based transition, $\mathcal{T}(x' \leftarrow x)$:

1. Take one step of $\widetilde{T}$ from point x to a "pivot state" z, i.e.: $z \sim \widetilde{T}(z \leftarrow x)$
2. Use T to draw K points one step away from z i.e.: $s_k \sim T(s_k \leftarrow z), \quad k = 1 \ldots K$
3. Use the s_k and x points to create an ordered list $\{s_k\}_{k=0}^{K}$. Break ties arbitrarily and relabel the points such that $s_k \leq s_{k+1}$. Identify index i giving $x = s_i$.
4. Set x' equal to s_{K-i}.

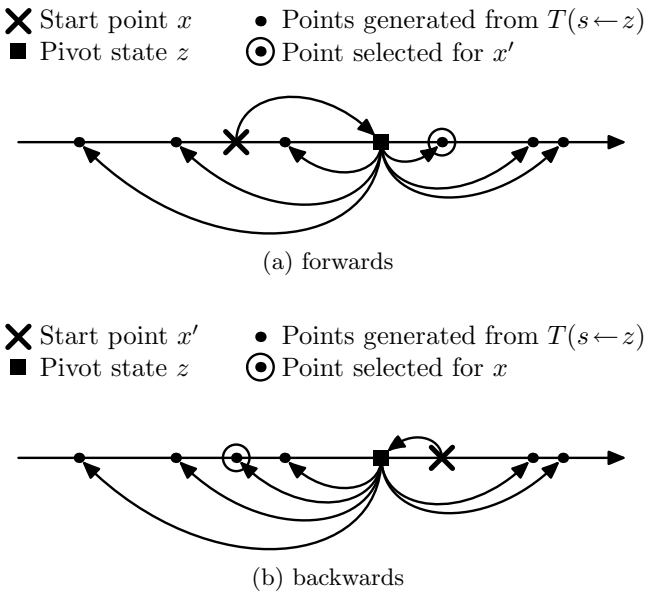


Figure 3.5: Illustration of the pivot-based transitions operator for $K = 6$ between points x and x' via points z and set S consisting of the set of round dots excluding the ringed one. (a) moving forwards generating a draw from $\mathcal{T}(x', S, z \leftarrow x)$. (b) the reverse process involving the same S and z occurs with probability $\mathcal{T}(x, S, z \leftarrow x')$.

the target distribution. They also have the same relation to the pivot state: all points including x and x' can be seen as draws from $T(s_k \leftarrow z)$. By symmetry the point labeled as the starting position can be resampled from a uniform distribution over the points, or updated as in ordered overrelaxation, which leaves the uniform distribution stationary. While perhaps unconvincing as a proof, this symmetry is important. If all the proposals started at x, then this location would be made special, making it harder to construct plausible reverse moves starting at any other point.

We now show explicitly that the new operator satisfies detailed balance. Given a starting point x, the probability of generating a pivot state z, a new point x' and $K-1$ other points (set S), as in figure 3.5a, is:

$$\mathcal{T}(x', S, z \leftarrow x) = \widetilde{T}(z \leftarrow x)\ K!\ T(x' \leftarrow z)\ \prod_{s \in S} T(s \leftarrow z). \tag{3.7}$$

The first step must produce the pivot state z from x. The other points can then be produced in any of $K!$ possible orderings. The probability of producing the same configuration of points by starting at x', as in figure 3.5b, is:

$$\mathcal{T}(x, S, z \leftarrow x') = \widetilde{T}(z \leftarrow x')\ K!\ T(x \leftarrow z)\ \prod_{s \in S} T(s \leftarrow z). \tag{3.8}$$

A small amount of manipulation gives:

$$\begin{aligned} \frac{\mathcal{T}(x, S, z \leftarrow x')}{\mathcal{T}(x', S, z \leftarrow x)} &= \frac{\widetilde{T}(z \leftarrow x')}{T(x' \leftarrow z)} \frac{T(x \leftarrow z)}{\widetilde{T}(z \leftarrow x)} = \frac{p(z)}{p(x')} \frac{p(x)}{p(z)} = \frac{p(x)}{p(x')} \\ \Rightarrow \mathcal{T}(x, S, z \leftarrow x') p(x') &= \mathcal{T}(x', S, z \leftarrow x) p(x) \\ \Rightarrow \mathcal{T}(x \leftarrow x') p(x') &= \mathcal{T}(x' \leftarrow x) p(x)\ , \end{aligned} \tag{3.9}$$

i.e. the new chain satisfies detailed balance. The second line was obtained from the T and $\widetilde{T}$ operators' mutual reversibility condition. The final line is obtained simply by summing over z and S.

The number of points K used by the algorithm could be automatically adjusted. We would use algorithm 3.3 to draw the points, replacing the conditional distribution with $T(s_k \leftarrow z)$. The proof above still holds if we replace $K!$ with $C(l, r)$.

3.4.1 Ordered overrelaxation with pivot-based transitions

Gibbs sampling updates each component of a distribution x_d in turn by applying a transition operator T_d that resamples from the conditional $p(x_d | x_{\{d' \neq d\}})$. The original ordered overrelaxation replaces Gibbs sampling's T_d operators with modified versions that tend to move to the opposite side of the conditional distribution from the current setting of x_d.

Gibbs sampling is not the only Monte Carlo algorithm that updates components of a distribution in turn. Now we can perform ordered overrelaxation by replacing any algorithm's component-based update operators T_d with their pivot-based versions. Each update will tend to move the current setting to the opposite side of the distribution defined by T_d. When the original Markov chain is Gibbs sampling the z states become irrelevant and the resulting transition operator is equivalent to the standard ordered overrelaxation algorithm.

Figure 3.6 gives an illustrative example of ordered overrelaxation applied to a bivariate Gaussian distribution. Using pivot-based transitions we can obtain the same benefits as the original ordered overrelaxation with slice sampling, which doesn't require the ability to sample from conditional distributions. Again, as with standard ordered overrelaxation, the benefits are only worth the computational cost if some saving can be made when drawing multiple points.

Although pivot-based transitions do not introduce any rejections into the chain, if the underlying transition operator is a Metropolis method then the chain will sometimes stay still. The rejections will cause reversals in the direction of motion, reducing the ability of the chain to maintain a persistent direction. This will tend to occur at the edges of a distribution where rejections are more frequent and where ordered overrelaxation is normally of most use. Figure 3.6c demonstrates that pivot-based transitions based on Metropolis works much less well than the same method based on slice or Gibbs sampling. In section 3.5 we will develop a better Metropolis method based on pivot states. First we explore another use of pivot-based transitions.

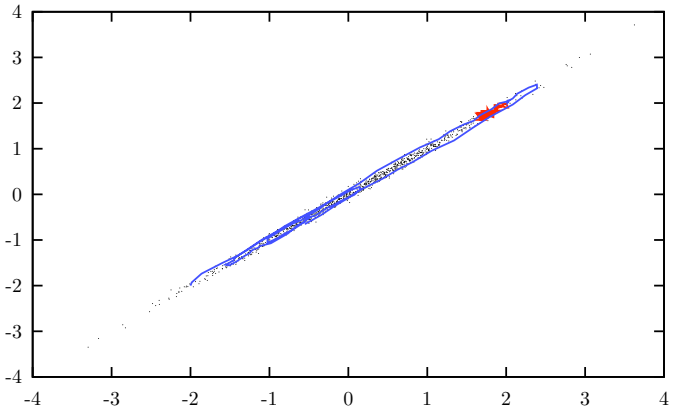
3.4.2 Persistence with pivot states

As pivot-based transitions apply to general transition operators we can consider algorithms that are not tied to component-based updates. In this section we present an alternative way to achieve persistent motion using pivot-based transitions. This algorithm is based on the *expanded* distribution,

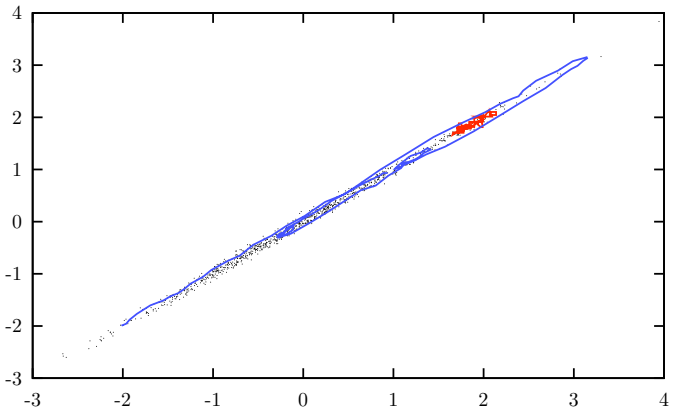
$$\ddot{p}(x, z) = \widetilde{T}(z \leftarrow x)p(x) = T(x \leftarrow z)p(z). \tag{3.10}$$

For other ways of constructing non-reversible chains that use this distribution see Neal (2004b). Here we do not require the base operator T to be reversible.

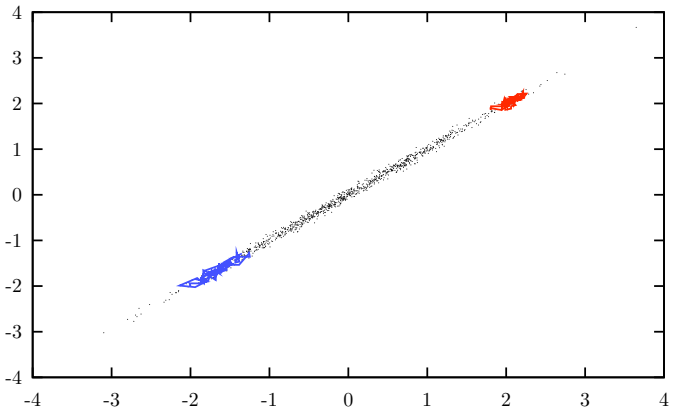
A sample (x, z) from the expanded distribution provides an equilibrium sample x and a pivot state z as produced by step 1 of algorithm 3.4. Following the rest of the algorithm gives a new setting (x', z), the same pivot state with a new starting point.



(a) Standard ordered overrelaxation (Gibbs sampling)



(b) Pivot-based transitions (univariate slice sampling)



(c) Pivot-based transitions (Metropolis)

Figure 3.6: Ordered overrelaxation schemes applied to a highly correlated bivariate Gaussian. Some exact samples from the distribution are shown in black. Standard Metropolis shown in red proceeds by slow diffusion. Ordered overrelaxation schemes shown in blue are able to make persistent progress along the distribution, here exaggerated with $K = 32$. Rejections in the underlying transition operators upset this persistent motion however. The Metropolis sampler in (c) gains little benefit from pivot-based transitions.

This procedure leaves the expanded distribution invariant, the proof is similar to before:

$$\mathcal{T}(x', S;\, x, z) = K!\; T(x' \leftarrow z) \prod_{s \in S} T(s \leftarrow z)$$
$$\mathcal{T}(x, S;\, x', z) = K!\; T(x \leftarrow z) \prod_{s \in S} T(s \leftarrow z)$$

$$\begin{aligned}\frac{\mathcal{T}(x, S;\, x', z)}{\mathcal{T}(x', S;\, x, z)} &= \frac{T(x \leftarrow z)}{T(x' \leftarrow z)} = \frac{\widetilde{T}(z \leftarrow x)p(x)}{p(z)} \frac{p(z)}{\widetilde{T}(z \leftarrow x')p(x')} \\ &= \frac{\widetilde{T}(z \leftarrow x)p(x)}{\widetilde{T}(z \leftarrow x')p(x')}\end{aligned}$$

$$\begin{aligned}\Rightarrow \mathcal{T}(x, S;\, x', z)\, \widetilde{T}(z \leftarrow x')p(x') &= \mathcal{T}(x', S;\, x, z)\, \widetilde{T}(z \leftarrow x)p(x) \\ \Rightarrow \mathcal{T}\big((x, z) \leftarrow (x', z)\big)\, \ddot{p}(x', z) &= \mathcal{T}\big((x', z) \leftarrow (x, z)\big)\, \ddot{p}(x, z)\,.\end{aligned} \tag{3.11}$$

The final line was obtained by using equation (3.10) and summing over the intermediate points S. We have shown that the appropriate detailed balance relationship holds with respect to the expanded distribution.

Equation (3.10) suggests that the sample from the expanded distribution can also be interpreted as an equilibrium sample z and a pivot state $x \sim T(x \leftarrow z)$. This pivot state could be updated by an alternative pivot-based transition operator $\widetilde{\mathcal{T}}$ based on algorithm 3.4. As T was used in step 1 we switch to using $\widetilde{T}$ to produce the K points in step 2. A version of the above proof shows that this operator also leaves the expanded distribution invariant.

Given a pair sampled from the expanded distribution $(x_1, x_2) \sim \ddot{p}(x_1, x_2) = \widetilde{T}(x_2 \leftarrow x_1)p(x_1)$ we alternate between two Markov chain updates. First we update x_1 using x_2 as a pivot state with algorithm 3.4 (operator $\mathcal{T}$). Next we update x_2 using x_1 as a pivot state with the alternate operator $\widetilde{\mathcal{T}}$. Figure 3.7 illustrates the effect of alternately applying these two operators. Both operators have a tendency to reverse the ordering of x_1 and x_2. By continually jumping over each other, the pair move persistently across the space that induces the ordering. By chance, for finite K, the points will sometimes stay in the same order. In subsequent iterations motion will be in the opposite direction allowing the chain to mix across the whole space.

Figure 3.8 shows pivot-based persistent motion when the base transition operator has a small step size, which would normally lead to slow diffusion. The effect is dramatic: the entire range of the distribution is explored in many fewer iterations. However, the improved dynamics alone are not actually sufficient to justify the increased cost of drawing multiple proposals. As with MTM, some further savings, such as parallel computation, are required.

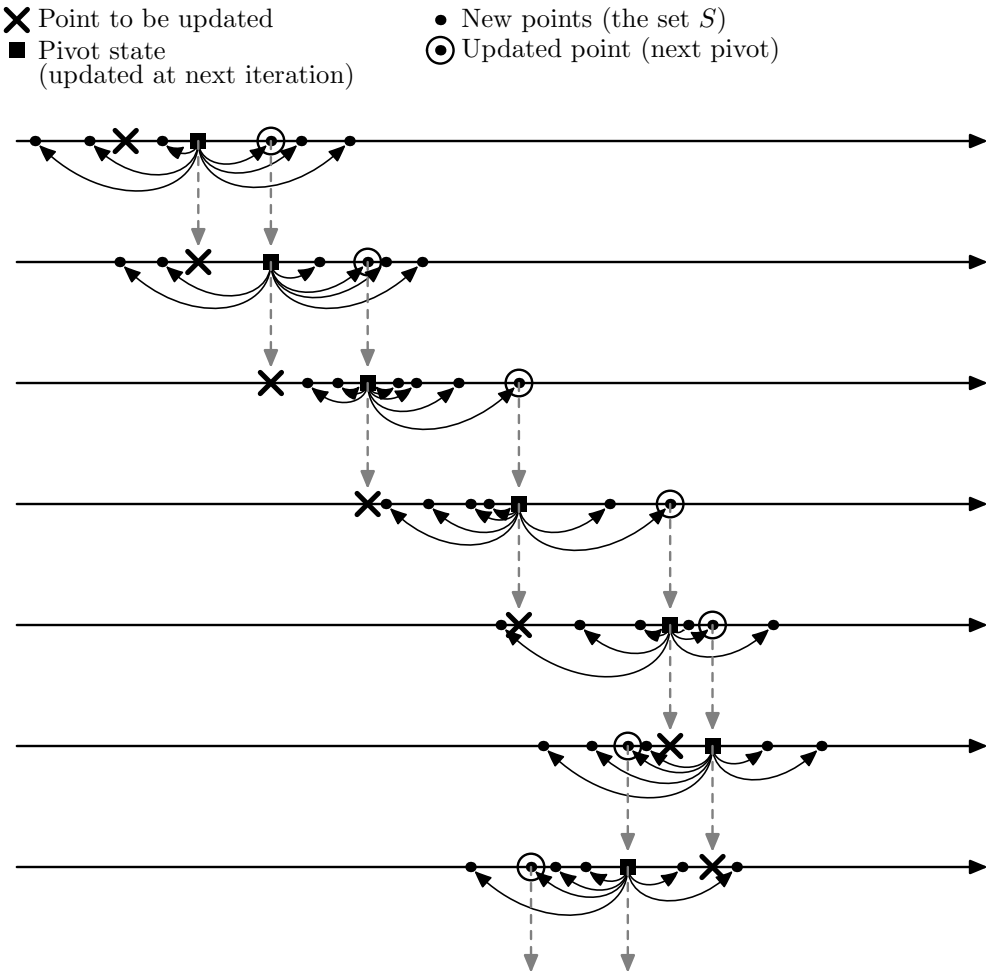


Figure 3.7: Using pivot states for persistent motion. Each row shows a new iteration of the algorithm described in the main text. The pivot state tends to move in the same direction for multiple iterations. In the final two iterations shown the direction of motion has reversed. Drift in one direction would persist for longer if a larger number of states were drawn from the pivot state.

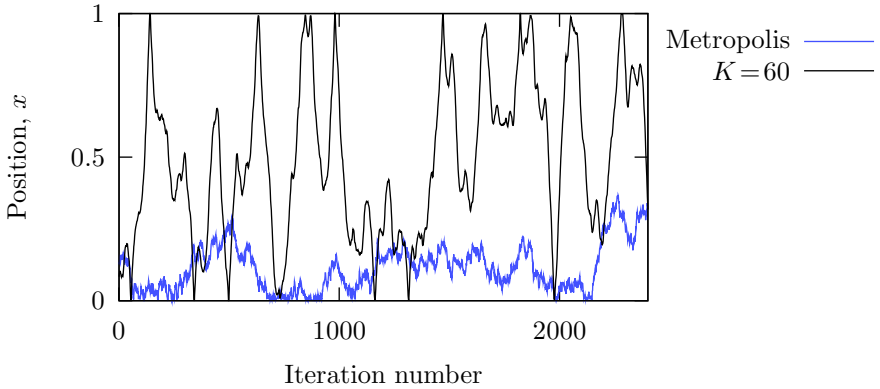


Figure 3.8: Example of persistent motion: exploring Uniform$[0, 1]$ with $\sigma = 0.05$ Gaussian proposals. Metropolis's random walk behavior is greatly reduced by pivot-based transitions, although at a greater computational cost.

3.5 Pivot-based Metropolis

Pivot-based Metropolis is a new alternative to MTM-based methods for using multiple Metropolis proposals. As in MTM we use an arbitrary proposal distribution $q(x_k \leftarrow x)$, but by making different trade-offs ordered-overrelaxation updates become possible.

The new procedure will follow algorithm 3.4 closely but uses arbitrary proposal distributions rather than transition operators. Arbitrary proposals could be turned into Metropolis–Hastings transition operators for use in algorithm 3.4, but many of the candidate points would be in the same place due to rejections. Algorithm 3.5 provides a way to use all of the points proposed from a proposal distribution q. The pivot state can optionally be created using a different proposal distribution $\widetilde{q}$. Also, the pivot state need not live in the same space as the distribution being sampled (unless applying the persistence procedure of subsection 3.4.2). This procedure can be seen as an instance of "Random Proposal Distributions" (Besag *et al.*, 1995).

Algorithm 3.5 The pivot-based Metropolis operator, $\mathcal{T}(x' \leftarrow x)$:

1. Take one step of $\widetilde{q}$ from point x to a "pivot state" z, i.e.: $z \sim \widetilde{q}(z \leftarrow x)$
2. Use q to draw K points one step away from z i.e.: $s_k \sim q(s_k \leftarrow z), \quad k = 1 \ldots K$
3. Use the s_k and x points to create an ordered list $\{s_k\}_{k=0}^{K}$. Break ties arbitrarily and relabel the points such that $s_k \leq s_{k+1}$. Identify index i giving $x = s_i$.
4. Compute weights $w_k = \dfrac{\widetilde{q}(z \leftarrow s_k)}{q(s_k \leftarrow z)}\, p(s_k)$
5. Choose $x' = s_j$ from $\{s_k\}_{k=0}^{K}$ using a distribution proportional to the weights, or any reversible move $T_s\big(j \leftarrow i; \{s_k\}\big)$ that leaves this distribution invariant.

The pivot-based Metropolis operator of algorithm 3.5 is a valid transition operator for $p(x)$. Proof: the equilibrium probability of starting at s_i, generating pivot state z the remainder of the set of points $S = \{s_k\}_{k=0}^{K}$ and choosing final point s_j is

$$\mathcal{T}\big((z, S, j) \leftarrow s_i\big) p(s_i) = T_s(j \leftarrow i; S)\, p(s_i)\, \widetilde{q}(z \leftarrow s_i)\, K! \prod_{k \neq i} q(s_k \leftarrow z), \tag{3.12}$$

which is invariant to swapping $s_i \leftrightarrow s_j$. Summing over all intermediate points $\{s_{k \neq i,j}, z\}$ shows that $\mathcal{T}(s_j \leftarrow s_i)$ satisfies detailed balance.

Ordered over-relaxation uses a transition rule that leaves a uniform distribution over a set of points stationary. Pivot-based Metropolis and perhaps other algorithms require a generalized move that leaves a non-uniform discrete distribution stationary.

A probability distribution over points $\{s_k\}$ can be represented on a unit interval. Each point occupies a segment with width equal to its probability p_k. The segments can be ordered by any method that does not use the current setting s_i or the history of the Markov chain.

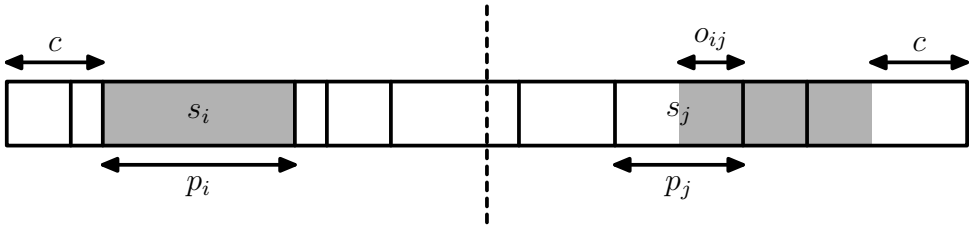


Figure 3.9: Reflect move for discrete distributions. A unit interval is constructed from segments $\{s_k\}$. Each of these bars has width equal to the point's probability. Given current point s_i, the probability of transitioning to another point is proportional to its overlap with s_i's reflection. For example s_j will be selected with probability o_{ij}/s_i. Self-transitions are only possible if s_i overlaps with the middle of the unit interval. This is when ordered overrelaxation is less helpful.

Assume we have a current sample s_i and wish to a apply a transition operator that tends to take us to the opposite end of the distribution. Such a transition operator is illustrated in figure 3.9. The probability interval corresponding to the current point, $[c, c{+}p_i]$, is reflected to $[1{-}c{-}p_i, 1{-}c]$. The reflected interval will have some overlap with one or more points' probability segments. We sample from these points with probability proportional to their overlap o_{ij} with the reflected interval. The probability of observing a transition from $s_i \to s_j$ is

$$T_s(j \leftarrow i) \cdot p_i = \frac{o_{ij}}{p_i} \cdot p_i = o_{ij}, \tag{3.13}$$

which is independent of the direction of the transition. Therefore detailed balance holds; the generalized reflection rule leaves the discrete target distribution invariant. Using this rule within algorithm 3.5 gives an operator based on arbitrary proposals that is suitable for use in ordered overrelaxation or the persistent motion of subsection 3.4.2.

3.6 Summary

This chapter explored ways to use populations of points drawn from proposal distributions. We started with a review of two existing methods, "Population Monte Carlo" (PMC) and Multiple-Try Metropolis (MTM), which highlighted two important points:

1. Proposal distributions used by MCMC are usually unsuitable for simple importance sampling. Attempts to combine MCMC with estimators based on importance sampling should not forget this.

2. Drawing multiple points in parallel from a proposal distribution usually gives slower mixing than drawing the same number of points sequentially. The parallel methods require savings from shared computations to be competitive.

In response to item 1. we derived a "waste recycling" scheme for MTM as an alternative to an existing importance sampling-based estimator.

We then focussed on methods that use a population of points to move the state of a Markov chain to the opposite side of a user-defined ordering. Chaining these updates together can cause persistent motion through Neal's ordered overrelaxation or the method we introduced in subsection 3.4.2. A random walk will explore a distribution with length-scale L in $\mathcal{O}(L/\sigma^2)$ steps of size σ. More persistent motion can bring this down to $\mathcal{O}(L/\sigma)$. Our contributions to this area were:

1. Algorithm 3.3, which automatically reduces the size of a population when ordered overrelaxation is less likely to be useful, saving computation.
2. Generalizing ordered overrelaxation to transition operators other than Gibbs sampling, algorithm 3.4.
3. Using ordered transition operators for persistent motion without using component-based updates, subsection 3.4.2.
4. Generalizing ordered updates to operate directly on a set of weighted points, section 3.5. These moves are less likely to reject than using Metropolis–Hastings within algorithm 3.4, which is important because rejections destroy persistent motion.

The usefulness of these innovations relies on the assumption that drawing multiple proposals is computationally efficient. It remains to be seen if implementations leveraging caching of intermediate results or parallel computations will make our work effective in real applications. Similar assumptions are made by related work in the literature, which we now briefly consider along with directions for future research.

3.7 Related and future work

Tjelmeland (2004) and Stormark (2006) investigate an idea similar to pivot-based Metropolis. They suggest drawing a set of antithetic variables, possibly using quasi Monte Carlo. Recent work also applies these ideas to MTM (Craiu and Lemieux, 2007). Pivot-based Metropolis could also be extended in this way. Tjelmeland also recommends waste-recycling, which would also apply to pivot-based algorithms. The main difference in his algorithm is that all the states are drawn as proposals from the current state x, rather than an intermediate state z. Making reversible moves based on these proposals requires working out the probability of each state reproducing the entire ensemble, an $\mathcal{O}(K^2)$ computation. Our pivot state approach reduces this computation to $\mathcal{O}(K)$ and makes an exchange of the current point with one of the proposals seem more natural.

As an example we showed pivot-based ordered overrelaxation applied to slice sampling. In fact, slice sampling already has its own method for overrelaxation (Neal, 2003), although this is susceptible to rejections unless the edges of the slice are found accurately.

Ordered overrelaxation may be preferable as it never has rejections while needing only enough information to draw K samples. There is a way to eliminate the pivot state when slice sampling from unimodal distributions: it is possible to Gibbs sample the slice sampling auxiliary distribution, so standard ordered overrelaxation applies.

Subsection 3.4.2 described a new way to make persistent motion along a direction that defines an ordering on the state space. Choosing suitable methods for ordering points will be problem-dependent in general. One generic application could be simulated tempering (subsection 2.5.1), where energy could be used to define an ordering. Persistent motion along this ordering should encourage the inverse-temperature to move more rapidly between zero and one.

Radford Neal has suggested (personal communication) considering an alternative to pivot-based transitions. This method starts with a random integer f drawn uniformly between 0 and K inclusive. The current point x is evolved through a sequence of f transitions using T. Then starting at x again, a sequence of $b = K - f$ 'backwards' transitions is simulated using $\widetilde{T}$. Choosing uniformly from the resulting $K+1$ states maintains the stationary distribution. Also reordering the points and choosing the complement to x as in ordered overrelaxation is valid. Producing a chain of states rather than a star of steps starting at a single state is likely to produce larger moves.

A version of Neal's idea based on the weighted reflect operator of section 3.5 is also possible. The chain could use transition operators that leave a cheaper-to-evaluate surrogate distribution invariant. Proposal weights could be evaluated for a thinned subset of the chain and then the reflect operator applied. Yet another possibility is the *multipoint Metropolis method* (Qin and Liu, 2001), an extension to MTM that uses a chain of states.

A disadvantage of using a chain of transitions is that each move must be made sequentially. The pivot-based algorithms introduced in this chapter could evaluate their proposals in parallel on suitable computer hardware.

Chapter 4

Normalizing constants and nested sampling

The computation of normalizing constants plays an important role in inference and statistical physics. For example, Bayesian model comparison needs the evidence, or marginal likelihood of a model $\mathcal{M}$ given observed data $\mathcal{D}$

$$\mathcal{Z} = p(\mathcal{D}|\mathcal{M}) = \int p(\mathcal{D}|\theta, \mathcal{M}) p(\theta|\mathcal{M}) \, \mathrm{d}\theta \equiv \int L(\theta)\pi(\theta) \, \mathrm{d}\theta, \tag{4.1}$$

where the model has prior π and likelihood L over parameters θ. In statistical physics $\mathcal{Z}(\beta) = \int \exp(-\beta E(\theta)) \mathrm{d}\theta$ is the normalizing constant for the distribution over states with energy E at inverse temperature β. In this context $\mathcal{Z}$ is an important quantity known as the *partition function*, a function of β from which several fundamental properties of a system can be derived. The marginal likelihood, equation (4.1), is also a function, in this case of the model $\mathcal{M}$, which will become important in chapter 5. In this chapter we focus on estimating the constant for a given model. We also consider estimating the whole $\mathcal{Z}(\beta)$ function from physics which, perhaps surprisingly, is useful for solving the marginal likelihood problem.

Statistical physics normally uses the normalization in its log form; $-\log \mathcal{Z}$ is known as the free energy of a system. When used in statistical inference $\log \mathcal{Z}$ has also been found to be a useful and natural scale for comparing models. Given two models $\mathcal{M}_0$ and $\mathcal{M}_1$ the log-likelihood ratio, $\log {}^{\mathcal{Z}_1}\!/_{\mathcal{Z}_0} = \log p(\mathcal{D}|\mathcal{M}_1) - \log p(\mathcal{D}|\mathcal{M}_0)$, can be used in a classical hypothesis test which rejects $\mathcal{M}_0$ in favor of $\mathcal{M}_1$ when a threshold is exceeded. This means that absolute differences in $\log \mathcal{Z}_0$ correspond to shifts in model quality that are meaningful without reference to an alternative model. Researchers decrypting codes at Bletchley park judged that a difference of roughly a *deciban*, $\log_{10} {}^{\mathcal{Z}_1}\!/_{\mathcal{Z}_0} = {}^1\!/_{10}$, is about the smallest difference in two models that people can express. Table 4.1 contains qualitative descriptions of the importance of marginal likelihood ratios.

Table 4.1: A selection of qualitative descriptions of the importance of evidence values taken from Kass and Raftery (1995) and Jeffreys (1961, Appendix B, p432). The third column could be turned into a "deviance" by multiplying by two. For more on units of information content, particularly the ban, see MacKay (2003, p264).

$\mathcal{Z}_1/\mathcal{Z}_0$	$\log_2 \mathcal{Z}_1/\mathcal{Z}_0$ (bits)	$\log_e \mathcal{Z}_1/\mathcal{Z}_0$ (nats)	$\log_{10} \mathcal{Z}_1/\mathcal{Z}_0$ (bans)	Evidence against $\mathcal{M}_0$
1 to 3.2	0 to 1.7	0 to 1.2	0 to 0.5	Weak
3.2 to 10	1.7 to 3.3	1.2 to 2.3	1/2 to 1	Substantial
10 to 100	3.3 to 6.6	2.3 to 4.6	1 to 2	Strong
> 100	> 6.6	> 4.6	> 2	Decisive
> 1000	> 10	> 7	> 3	Beyond reasonable doubt(?)

The precise interpretation of the numerical value of a model's evidence will depend on context. In some applications it is quite possible to keep using $\mathcal{M}_0$ even with a marginal-likelihood ratio that in isolation seems in favor of $\mathcal{M}_1$ "beyond reasonable doubt". In a legal setting it *may* be that a guilty hypothesis being a thousand times more likely than innocent is enough to convict. But other applications associate even more extreme losses with adopting a particular model. In data compression of large files a difference in size of 10 bits is insignificant. If compressing with $\mathcal{M}_0$ is computationally cheap an improvement in marginal likelihood very much greater than 10 bits is required to adopt a new model. Even in the legal setting one must remember that the most probable model is not necessarily the one with the highest marginal likelihood. Extreme prior ratios can cancel large likelihood ratios, although this shouldn't happen very often. Under the rules of rational inference one should mechanically apply Bayes' rule, but if the data are strongly out of line with prior expectations it would seem sensible to carefully reexamine the models' assumptions and check for computational errors.

This chapter focusses on the computation of $\mathcal{Z}$. The message to take from the above discussion is that computing the evidence of a model to a very high level of precision is usually pointless. It is difficult to appreciate a difference in $\log \mathcal{Z}$ of less than one, and some errors larger than this may not affect decisions based on the computation. This means that even noisy Monte Carlo estimates can be useful. Indeed Monte Carlo techniques are sometimes identified as providing "gold standard" estimates of $\mathcal{Z}$ suitable for comparison with other approximate techniques (e.g. Beal and Ghahramani, 2003; Kuss and Rasmussen, 2005). However, just as with any Monte Carlo integration method, it is quite possible to get wrong answers for some classes of difficult problems. This includes algorithms guaranteed to be correct asymptotically. Developing a variety of methods and checks and hoping for the best is all we can do in general.

This chapter first examines the necessary ingredients of a Monte Carlo method for computing $\mathcal{Z}$. Next we review *nested sampling*, a new method due to John Skilling. We provide a comparative analysis of this new algorithm and more established techniques. Then, having analyzed nested sampling in its own right, we consider using it as a method for guiding and checking other algorithms.

4.1 Starting at the prior

By definition the prior $\pi(\theta)$ should spread its probability mass over settings of the parameters deemed reasonable before observing the likelihood $L(\theta)$. Therefore, samples from this distribution can sometimes provide a useful representation of the whole parameter space. In particular, as $\mathcal{Z}$ is just an average under the prior, a simple Monte Carlo approach could be attempted,

$$\mathcal{Z} = \int L(\theta)\pi(\theta)\,\mathrm{d}\theta \approx \frac{1}{S}\sum_{s=1}^{S} L(\theta^{(s)}), \quad \theta^{(s)} \sim \pi(\theta). \tag{4.2}$$

The variance of this estimator may be huge. Most of the mass of the integral is associated with parameter settings that are typical under the posterior. The posterior often occupies a small effective fraction of the prior's volume, so it can take a long time to obtain a representative sample. However simple Monte Carlo, equation (4.2), does work on small problems and provides a simple-to-implement check of more complex code. Starting at the prior will also form the basis of more advanced methods.

Why not start at the posterior instead? Posterior samples are concentrated around the mass of the integral. Also practitioners already sample from the posterior for making predictions. It would be convenient if these samples could also estimate $\mathcal{Z}$. The obvious importance sampling estimator for $\mathcal{Z}$ based on posterior samples is nonsensical as it involves knowing $\mathcal{Z}$. A moment of inspiration yields the following harmonic mean estimator noted by Newton and Raftery (1994):

$$\begin{aligned}\frac{1}{\mathcal{Z}} = \frac{1}{\mathcal{Z}}\frac{L(\theta)}{L(\theta)}\int \pi(\theta)\,\mathrm{d}\theta &= \int \frac{1}{L(\theta)}\,p(\theta|\mathcal{D},\mathcal{M})\,\mathrm{d}\theta \\ &\approx \frac{1}{S}\sum_{s}\frac{1}{L(\theta^{(s)})}, \quad \theta^{(s)} \sim p(\theta|\mathcal{D},\mathcal{M}).\end{aligned} \tag{4.3}$$

As acknowledged by its creators, this estimator has the unfortunate property that the least probable points have small L and so carry the largest weights. Thus the effective sample size tends to be very small; indeed the estimator can easily have infinite variance.

Various attempts have been made to fix the harmonic mean estimator, both within the original paper and in more recent research (Raftery *et al.*, 2007). The authors are keen to avoid sampling from the prior as well as the posterior, however this goal seems misplaced. Firstly, implementing samplers for most priors is relatively simple and allows several diagnostics to be performed, e.g.: checking prior assumptions look reasonable, checking against simple importance sampling (equation (4.2)) and Geweke (2004)'s tests for "getting it right". Secondly, while some specific instances seem to work, there is a generic problem with posterior-only methods as pointed out by Neal in the discussion of Newton and Raftery (1994). The choice of prior has a strong influence on the value of $\mathcal{Z}$; taking a broad prior to an improper limit will send $\mathcal{Z}$ towards zero.

However in many inference problems the data are so influential that the statistics of the posterior are fairly insensitive to such changes in the prior. This demonstrates that posterior statistics alone bear little relation to $\mathcal{Z}$ in many statistical problems.

Reliable estimates of $\mathcal{Z}$ require finding the prior mass associated with the large likelihood values found under the posterior. In low-dimensional problems this is a feasible density estimation problem, although deterministic approximations to $\mathcal{Z}$ based on a direct fit to the posterior may be preferable in those cases. Chib (1995) notes that it is sufficient to know the posterior probability at just a single point. For any point θ^*

$$\mathcal{Z} = \frac{L(\theta^*)\,\pi(\theta^*)}{P(\theta^*|\mathcal{D},\mathcal{M})}. \tag{4.4}$$

Chib provides an estimator for $P(\theta^*|\mathcal{D},\mathcal{M})$ based on the probability of a Gibbs sampler's transitioning to θ^* from each of the points visited by the sampler. Chib's method is relatively easy to apply and should work well on unimodal problems.

Posteriors with more complex shapes will lead to subtle difficulties. In general it is unreasonable to expect the sequence of states visited by a sampler to densely cover the posterior in models with many parameters. If this were possible it would be feasible to build a kernel density estimate on the basis of a preliminary run and perform importance sampling. Instead practitioners simply hope to get a representative sample of points that are useful for prediction. It is accepted that the sampler may not go near some typical points. As an analogy consider conducting a survey of the world population. A sample of people might not include anyone from some cities, yet will be adequate for many statistical purposes. In contrast Chib's method is likely to fail without fairly dense sampling. If none of the posterior samples can easily reach θ^* within one Gibbs-sampling iteration ("are in its city") the estimator will be badly behaved. This won't occur if θ^* was chosen from the posterior samples, but this is only hiding the problem at the expense of biasing the estimator.

Jumping directly to the posterior is fraught with problems. The normalizer $\mathcal{Z}$ is a global quantity that depends on the entire posterior's relationship to the prior. If standard numerical methods apply, that is good news. But if the posterior is sufficiently complicated to justify MCMC the place to start is probably a much simpler distribution like the prior. On small problems simple Monte Carlo can be more reliable than Chib's method Neal (1999). When solving larger problems prior sampling, equation (4.2), doesn't work either. We need advanced methods that start with a feasible sampling problem but overcome the variance problems of simple importance sampling.

4.2 Bridging to the posterior

When samples from the prior rarely find regions of high posterior mass a distribution closer to the posterior is needed for importance sampling to work. Designing or adaptively finding a tractable approximation to the posterior is one approach, although hard in general. Another solution is a divide-and-conquer approach; rather than finding $\mathcal{Z}$ directly a series of easier sub-problems are solved.

First consider a family of distributions, specified by an *inverse temperature* β,

$$p(\theta;\beta) = \frac{1}{\mathcal{Z}(\beta)}\pi(\theta)L(\theta)^{\beta} = \frac{1}{\mathcal{Z}(\beta)}\pi(\theta)\, e^{-\beta E(\theta)}, \quad E(\theta) \equiv -\log L(\theta). \tag{4.5}$$

The prior has $\beta=0$, while $p(\theta;\beta=1)$ is the posterior. All of the distributions for $\beta>0$ involve unknown normalizations given by the *partition function* $\mathcal{Z}(\beta)$. We will try to compute this function for a range of inverse temperatures $\beta_0=0 < \beta_1 < \beta_2 \ldots < \beta_K < \beta_{K+1}=1$.

When adjacent distributions $p(\theta;\beta_k)$ and $p(\theta;\beta_{k+1})$ are close they can be compared with standard importance sampling. As reviewed in subsection 1.3.2, weights give the relative importance of states evaluated under the two distributions,

$$w_k^*(\theta) = \frac{\pi(\theta)L(\theta)^{\beta_{k+1}}}{\pi(\theta)L(\theta)^{\beta_k}} = L(\theta)^{\beta_{k+1}-\beta_k}. \tag{4.6}$$

The expected value of these weights is the ratio of the two distributions' normalizers:

$$\frac{\mathcal{Z}(\beta_{k+1})}{\mathcal{Z}(\beta_k)} \approx \frac{1}{S}\sum_{s=1}^{S} w_k^*(\theta^{(s)}), \quad \theta^{(s)} \sim p(\theta;\beta_k). \tag{4.7}$$

These estimates can then be combined using

$$\mathcal{Z} = \frac{\mathcal{Z}(\beta_{K+1})}{\mathcal{Z}(\beta_0)} = \frac{\mathcal{Z}(\beta_1)}{\mathcal{Z}(\beta_0)}\frac{\mathcal{Z}(\beta_2)}{\mathcal{Z}(\beta_1)}\frac{\mathcal{Z}(\beta_3)}{\mathcal{Z}(\beta_2)}\cdots\frac{\mathcal{Z}(\beta_{K+1})}{\mathcal{Z}(\beta_K)}. \tag{4.8}$$

Numerical concerns suggest taking logs of a large positive product,

$$\log \mathcal{Z} = \log \frac{\mathcal{Z}(\beta_{K+1})}{\mathcal{Z}(\beta_0)} = \sum_{k=0}^{K} \log \frac{\mathcal{Z}(\beta_{k+1})}{\mathcal{Z}(\beta_k)}, \tag{4.9}$$

where each term in the sum can be estimated from samples under $p(\theta;\beta_k)$. This form is also convenient for error bars: the variance of an estimator for $\log \mathcal{Z}$ is the sum of the variances of the estimators for the log of each ratio.

Given this basic annealing approach there are several implementation alternatives. Simulated tempering and parallel tempering reviewed back in section 2.5 both offer correlated samples from $p(\theta;\beta)$ over a user-set range of β. Annealed importance sampling,

subsection 2.5.3, samples from distributions which are only close to $p(\theta;\beta)$, but a single run gives the same estimator as above for $S=1$:

$$\widehat{\log \mathcal{Z}} = \sum_{k=0}^{K} \log w_k^*(\theta^{(k)}) = \sum_{k=0}^{K} (\beta_{k+1} - \beta_k) \log L(\theta^{(k)}), \tag{4.10}$$

where $\theta^{(k)}$ is the sample drawn at the k^{th} iteration of AIS, or a single sample from $p(\theta;\beta_k)$.

When $S>1$ or multiple AIS runs are performed the samples are combined differently. Independent importance samplers provide local gradients or ratio information at each temperature. The samples are first combined at each level as in equation (4.7). AIS gives up the requirement for valid samples from each $p(\theta;\beta_k)$ by running importance sampling on a global distribution (see subsection 2.5.3). Thus AIS dictates forming independent $\widehat{\mathcal{Z}}$ estimates from each run and then averaging them. Generally the AIS estimator should be used because it does not assume exact samples are available from each temperature. In some cases applying equation (4.7) to pseudo-samples from rapidly mixing Markov chains can work better than AIS, but is difficult to justify.

4.2.1 Aside on the 'prior' factorization

Some readers may be more familiar with *canonical* distributions defined by $p(\theta;\beta) = \exp(-\beta E(\theta))/\mathcal{Z}(\beta)$. Here the energy E is defined in terms of the negative-log of the whole unnormalized probability rather than just the likelihood. For finite state spaces this corresponds to equation (4.5) with a uniform $\pi(\theta)$. In an unbounded state space $p(\theta;\beta=0)$ is an improper distribution. Introducing a tractable, normalized base measure $\pi(\theta)$ ensures it is always possible to sample from $p(\theta;\beta=0)$.

The base measure $\pi(\theta)$ need not be a prior distribution. Another choice may be computationally convenient, or the problem may not be Bayesian and so have no prior. In subsection 4.7.3 a distribution is factorized in an unconventional way for algorithmic reasons. The term *prior* is used throughout this chapter to emphasize a common usage of these algorithms, and that $\pi(\theta)$ should usually be a simple, tractable starting point.

4.2.2 Thermodynamic integration

The bridging procedures above have been independently developed by various communities (Gelman and Meng, 1998). These different views can bring a better understanding to the problem. The statistical physics community views $\mathcal{Z}$ as a useful function. Applying operators to the partition function often yields interesting, physically meaningful

quantities. In particular for the canonical distribution

$$p(\theta;\beta) = \frac{1}{\mathcal{Z}(\beta)}\pi(\theta)e^{-\beta E(\theta)}, \qquad \mathcal{Z}(\beta) = \int \pi(\theta)e^{-\beta E(\theta)}\,\mathrm{d}\theta, \tag{4.11}$$

the log-normalizer is a moment generating function:

$$\begin{aligned}
\frac{\mathrm{d}\log\mathcal{Z}}{\mathrm{d}\beta} &= -\frac{1}{\mathcal{Z}}\int E(\theta)\;\pi(\theta)e^{-\beta E(\theta)}\,\mathrm{d}\theta \\
&= -\mathbb{E}_{p(\theta;\beta)}[E(\theta)] \\
\frac{\mathrm{d}^2\log\mathcal{Z}}{\mathrm{d}\beta^2} &= \frac{1}{\mathcal{Z}}\int E(\theta)^2\;\pi(\theta)\,e^{-\beta E(\theta)}\,\mathrm{d}\theta - \left(\frac{1}{\mathcal{Z}}\int E(\theta)\;\pi(\theta)\,e^{-\beta E(\theta)}\,\mathrm{d}\theta\right)^2 \\
&= \mathbb{E}_{p(\theta;\beta)}\left[E(\theta)^2\right] - \mathbb{E}_{p(\theta;\beta)}[E(\theta)]^2 \\
&= \mathrm{var}_{p(\theta;\beta)}[E(\theta)] \\
&\ldots
\end{aligned} \tag{4.12}$$

Unlike $\mathcal{Z}$ itself these expectations can reasonably be approximated for a given temperature using samples recorded under simulations at that temperature. Of course $\log\mathcal{Z}$ is related to its gradients by a trivial identity:

$$\begin{aligned}
\log\mathcal{Z}(1) &= \log\mathcal{Z}(0) + \int_0^1 \left.\frac{\mathrm{d}\log\mathcal{Z}}{\mathrm{d}\beta}\right|_{\tilde{\beta}}\,\mathrm{d}\tilde{\beta} \\
\log\mathcal{Z} = \log\frac{\mathcal{Z}(1)}{\mathcal{Z}(0)} &= -\int_0^1 \mathbb{E}_{p(\theta;\beta)}[E(\theta)]\;\mathrm{d}\beta.
\end{aligned} \tag{4.13}$$

The simplest discrete approximation to equation (4.13) uses measurements at a sequence of temperatures $\beta_0{=}0, \beta_1, \ldots \beta_K$ in order to estimate the normalizer at inverse temperature $\beta_{K+1}{=}1$,

$$\widehat{\log\mathcal{Z}} = -\sum_{k=0}^{K}(\beta_{k+1}-\beta_k)E(\theta_k), \quad \theta_k \sim p_{\beta_k}. \tag{4.14}$$

Comparing to equation (4.10) and remembering that $E(\theta) = -\log L(\theta)$ we see that simple thermodynamic integration and a bridging sequence of importance samplers are the same approximation.

Thermodynamic integration seems deceptively general. The difficulty is in choosing intermediate distributions. Appropriate temperature-based distributions may be difficult and sometimes impossible to find. This chapter explores how to choose a sequence of temperatures and alternative methods that don't use annealing schedules.

4.3 Multicanonical sampling

The multicanonical ensemble was mentioned in section 2.6 as a distribution that might allow better Markov chain exploration of the state space than the original target distribution. The multicanonical ensemble is available in terms of the target distribution $p_t(\theta) = p_t^*(\theta)/\mathcal{Z}_t$ and a multicanonical weighting function $w_{\mathrm{MC}}(\theta)$:

$$p_{\mathrm{MC}}(\theta) = \frac{p_{\mathrm{MC}}^*(\theta)}{\mathcal{Z}_{\mathrm{MC}}}, \qquad p_{\mathrm{MC}}^*(\theta) = p_t^*(\theta)\, w_{\mathrm{MC}}(\theta). \tag{4.15}$$

The multicanonical heuristic suggests finding a weighting such that a Markov chain exploring $p_{\mathrm{MC}}(\theta)$ spends equal times at all energies, including those typical of the posterior and the prior. A weighting function giving approximately this behavior must be found from preliminary runs.

Samples from p_{MC} can be used as an importance sampling proposal distribution. As in subsection 1.3.2 weights $w^*(\theta) = p^*(\theta)/p_{\mathrm{MC}}^*(\theta)$ give the normalizing constant ratio between any distribution $p(\theta) = p^*(\theta)/\mathcal{Z}_p$ and the multicanonical distribution p_{MC}:

$$\Delta(p) = \frac{1}{S}\sum_{s=1}^{S} w^*(\theta^{(s)}) \approx \frac{\mathcal{Z}_p}{\mathcal{Z}_{\mathrm{MC}}}. \tag{4.16}$$

The multicanonical normalization $\mathcal{Z}_{\mathrm{MC}}$ is generally unavailable but can be eliminated by comparing to the base distribution:

$$\widehat{\log \mathcal{Z}_t} = \log \Delta(p_t) - \log \Delta(\pi). \tag{4.17}$$

Instead of bridging carefully between the prior and posterior, the multicanonical ensemble has the ambitious goal of capturing both at once. One hopes that the broad coverage of p_{MC} gives estimators with reasonable variance for quantities relating to both p_t and π. This is explored theoretically and experimentally in later sections.

4.4 Nested sampling

Nested sampling is a new Monte Carlo method by Skilling (2004, 2006, 2007) intended for "general Bayesian computation", which bears some relation to earlier work found in McDonald and Singer (1967). It is designed to be a general and robust alternative to annealing-based methods. Like annealing it starts at the prior and samples from a sequence of distributions that become more constrained at each iteration. However, nested sampling makes no use of temperature and does not require tuning of intermediate distributions or other large sets of parameters. It also provides a natural means to compute error bars on all of its results without needing multiple runs of the algorithm.

The key feature and technical difficulty of nested sampling is sampling from the prior subject to a series of lower bounds on the likelihood. The reward for attempting this novel challenge is an estimate of the prior probability mass associated with each observed likelihood value. This representation of the posterior provides an estimate of the normalization constant and any other property of the posterior.

In addition to reviewing necessary material the remainder of this chapter provides:

1. An improved implementation of nested sampling for problems with degenerate likelihoods or discrete distributions.
2. A brief study of a deterministic approximation of nested sampling's behavior. This has theoretical implications for its performance and practical benefits for some applications.
3. A comparative analysis of the generic properties of annealed importance sampling, nested sampling and multicanonical sampling.
4. Illustrative examples: some simple continuous distributions and the Potts model, an undirected graphical model. Incidentally a new variant of Swendsen–Wang is derived.

Some of this material was previously presented in Murray *et al.* (2006b).

4.4.1 A change of variables

The normalization of a posterior over variables θ is a weighted sum of likelihoods $L(\theta)$ over elements of prior mass $\pi(\theta)\mathrm{d}\theta$:

$$\mathcal{Z} = p(\mathcal{D}|\mathcal{M}) = \int p(\mathcal{D}|\theta, \mathcal{M})\; p(\theta|\mathcal{M})\mathrm{d}\theta \;= \int L(\theta)\; \pi(\theta)\mathrm{d}\theta. \tag{4.18}$$

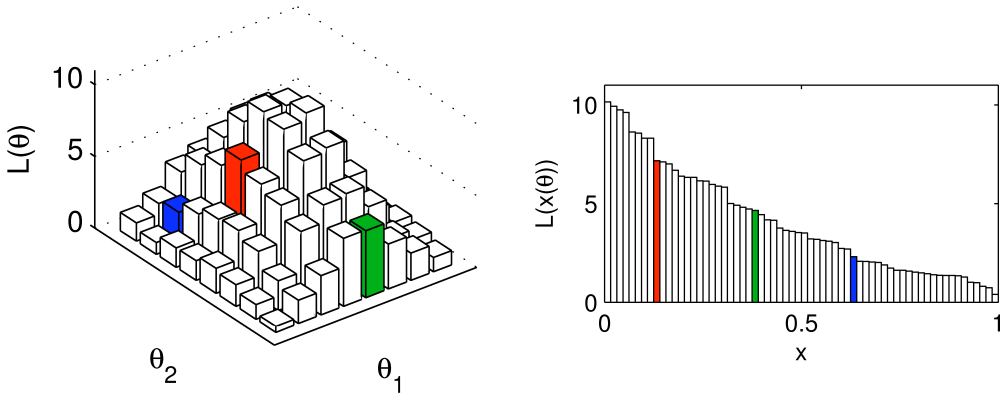
We label each element of prior mass $\mathrm{d}x(\theta) = \pi(\theta)\mathrm{d}\theta$. This is only a change in notation:

$$\mathcal{Z} = \int L(\theta)\; \mathrm{d}x(\theta). \tag{4.19}$$

The integral could, in principle, by accumulating elements in a standard raster order in the θ parameter space, figure 4.1a. Alternatively we can add up the contributions from each scalar element of prior mass $\mathrm{d}x$ in any order we like. As illustrated in figure 4.1b we choose to order the elements by their corresponding likelihood. As π is a distribution, its elements sum to one and can be arranged along a unit interval:

$$\mathcal{Z} = \int_0^1 L(\theta(x))\; \mathrm{d}x, \qquad x(\theta) = \int\limits_{\theta' : L(\theta') > L(\theta)} \pi(\theta')\; \mathrm{d}\theta'. \tag{4.20}$$

For now assume θ is continuous and $L(\theta)$ provides a total ordering of elements so that $x(\theta)$ is an invertible change of variables. In subsection 4.4.6 these assumptions will be relaxed so that the mapping will always be invertible.



(a) (b)

Figure 4.1: Views of the integral $\mathcal{Z} = \int L(\theta)\,\pi(\theta)\mathrm{d}\theta$ for the posterior normalization: (a) Elements of the parameter space $\mathrm{d}\theta$ are associated with likelihood values (heights) $L(\theta)$. These elements' likelihood values are summed weighted by their prior masses $\pi(\theta)\mathrm{d}\theta$. (b) Bars with exactly the same set of heights are arranged in order along a scalar unit interval. Three bars are colored to help illustrate the correspondence. In (a) the bars have a (hyper-)area base corresponding to an element in parameter space. In (b) they have a scalar width corresponding to an element of prior mass $\mathrm{d}x = \pi(\theta(x))\mathrm{d}\theta(x)$. The area of each bar is its weighted likelihood, so $\mathcal{Z}$ is the sum of these, the area under the curve in (b).

The mapping $x(\theta)$ may seem strange, but is closely related to the familiar cumulative distribution function (CDF) for a one-dimensional density p:

$$c(\theta) = \int_{\theta' : \theta' < \theta} p(\theta')\,\mathrm{d}\theta'. \tag{4.21}$$

This quantity is often considered very natural; some authors prefer to define distributions in terms of the CDF. Unlike a density the CDF is invariant to monotonic changes of variable and it directly returns the probability mass between two settings of the parameters. Also its inverse allows the straightforward generation of samples from uniform random variates. Each element $\mathrm{d}c$ corresponds to an element of probability mass, $p(\theta)\mathrm{d}\theta$. Naturally these elements sum to one: $\int_0^1 \mathrm{d}c = 1$.

Cumulative distribution functions for multivariate distributions are less frequently used in statistics. A sensible ordering of the elements of probability mass is less obvious, and the multivariate integrals involved may be difficult. Comparing equations (4.20) and (4.21) we see that x is just a cumulative distribution function of the prior corresponding to a particular choice of ordering. For scalar parameters the standard inverse CDF $\theta(c)$ gives the parameter such that fraction c of the prior mass is associated with parameters less than θ. For general parameters the inverse of the likelihood-sorted CDF $\theta(x)$ gives the parameter value such that fraction x of the prior mass is associated with likelihoods greater than $L(\theta)$. This is illustrated in figure 4.2a. For multimodal likelihoods the prior mass satisfying a likelihood constraint will be scattered over disconnected regions.

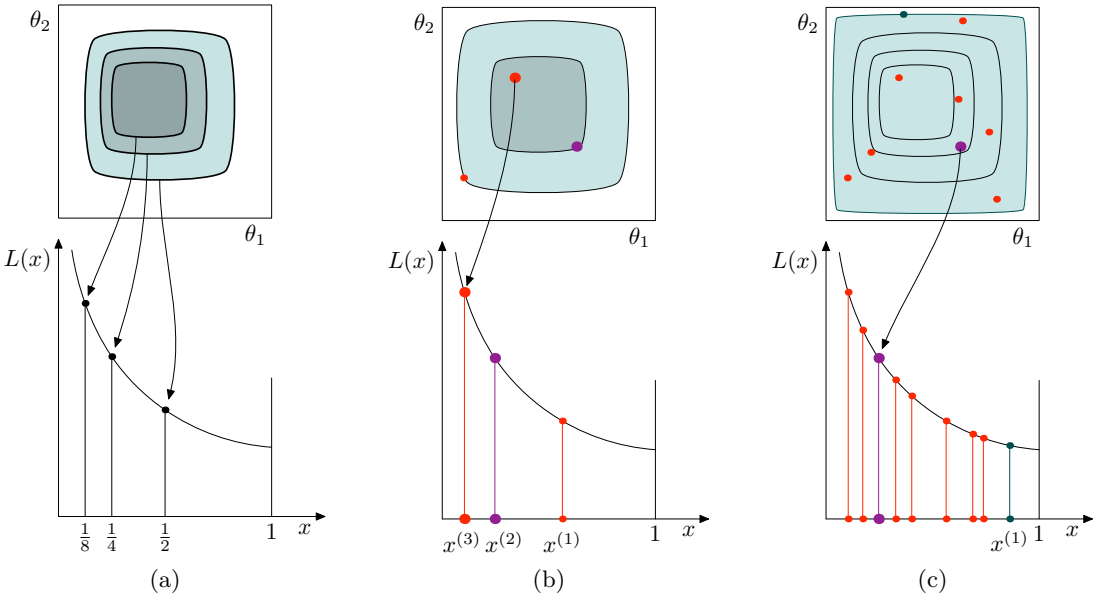


Figure 4.2: Nested sampling illustrations (adapted from MacKay (2004)). (a) Elements of parameter space (top) are sorted by likelihood and arranged on the x-axis. An eighth of the prior mass is inside the innermost likelihood contour in this figure. (b) Point $x^{(s)}$ is drawn from the prior inside the likelihood contour defined by $x^{(s-1)}$. $L^{(s)}$ is identified, the ordering on the x-axis and $p(\{x^{(s)}\})$ are known, but exact values of $x^{(s)}$ are not known. (c) With N particles, the one with smallest likelihood defines the likelihood contour and is replaced by a new point inside the contour ($\{L^{(s)}\}$ and $p(\{x^{(s)}\})$ are still known).

4.4.2 Computations in the new representation

Given the change of variables described in the previous section the normalizer is just the area under a monotonic one-dimensional curve "L vs. x". One-dimensional integrals are usually easy to approximate numerically. Assuming an oracle provided some points $\{(x^{(s)}, L^{(s)})\}_{s=1}^{S}$ ordered such that $x^{(s)} > x^{(s+1)}$, we can obtain an estimate $\widehat{\mathcal{Z}}$ based on quadrature:

$$\widehat{\mathcal{Z}} = \sum_{s=1}^{S} L^{(s)} w_q^{(s)} \qquad w_q^{(s)} = \frac{1}{2}\left(x^{(s-1)} - x^{(s+1)}\right), \tag{4.22}$$

where the quadrature weights given correspond to the trapezoidal rule. Rectangle rules can upper and lower bound the error $\widehat{\mathcal{Z}} - \mathcal{Z}$, as long as an upper bound on L is known. The boundary conditions of the sum require an arbitrary choice for the edge x-values such as $x^{(0)} = 2 - x^{(1)}$ and $x^{(S+1)} = -x^{(S)}$. Sensitivity to such choices could always be checked; in our experience they do not matter.

Quadrature is effectively approximating the posterior with a distribution over S particles each with probability proportional to $L^{(s)} w_q^{(s)}$. Samples from this distribution can approximate samples from the true posterior. The approximate distribution can also be used to directly approximate posterior expectations.

The change of variables has removed details of the high-dimensional θ space and made

the integration problem apparently easy. Of course the high-dimensional space is still in the original problem and will make the change of variables intractable. This is now the target for approximation.

4.4.3 Nested sampling algorithms

Nested sampling aims to provide a set of points $\{\theta^{(s)}, L^{(s)}\}_{s=1}^{S}$ and a probability distribution over their corresponding $\mathbf{x}=\{x^{(s)}\}$ values. A simple algorithm to draw S such points is algorithm 4.1, see also figure 4.2b.

Algorithm 4.1 Single particle nested sampling

1. Initial point: draw $\theta^{(1)} \sim \pi(\theta)$
2. **for** $s = 2$ to S
3. draw $\theta^{(s)} \sim \breve{\pi}(\theta; L(\theta^{(s-1)}))$, where
$$\breve{\pi}(\theta; L(\theta^{(s-1)})) \propto \begin{cases} \pi(\theta) & L(\theta) > L(\theta^{(s-1)}) \\ 0 & \text{otherwise.} \end{cases} \tag{4.23}$$
4. **end for**

The first parameter set by this algorithm is drawn from the prior, which implies that the corresponding cumulative prior quantity must have distribution $p(x^{(1)}) = \text{Uniform}[0, 1]$. Similarly $p(x^{(s)}|x^{(s-1)}) = \text{Uniform}[0, x^{(s-1)}]$, as each point is drawn from the prior subject to $L(\theta^{(s)}) > L(\theta^{(s-1)}) \Rightarrow x^{(s)} < x^{(s-1)}$. This recursive relationship defines $p(\mathbf{x})$.

A simple generalization, algorithm 4.2, uses multiple θ particles; at each step the one with smallest likelihood is replaced with a draw from a constrained prior (figure 4.2c).

Algorithm 4.2 Multiple particle nested sampling

1. Initialize: draw N points $\{\theta_{(n)} \sim \pi(\theta)\}_{n=1}^{N}$
2. $m = \text{argmin}_n L(\theta_{(n)})$
3. $\theta^{(1)} = \theta_{(m)}$
4. **for** $s = 2$ to S
5. redraw $\theta_{(m)} \sim \breve{\pi}(\theta; L(\theta^{(s-1)}))$, given by equation (4.23), algorithm 4.1
6. $m = \text{argmin}_n L(\theta_{(n)})$
7. $\theta^{(s)} = \theta_{(m)}$
8. **end for**

The first parameter $\theta^{(1)}$ is the setting with the smallest L from the initial N draws, this is the particle with the largest x. For this parameter to be at $x^{(1)}$ the other $(N-1)$ points must have $x < x^{(1)}$ so $p(x^{(1)}|N) = N(x^{(1)})^{N-1}$. The extra factor of N comes from the invariance of the points to reordering and is needed for correct normalization. Alternatively it is immediately identified as a standard result[1] from order statistics

[1] The n-th ordered point drawn from a distribution has its cumulative quantity distributed according to Beta(n, $N+1-n$) where our case corresponds to $n=N$.

(e.g. Balakrishnan and Clifford Cohen, 1991). After replacing a particle in step 5. there will be N samples uniformly between 0 and $x^{(s-1)}$. The point with smallest L and largest x will be a fraction $r = x^{(s)}/x^{(s-1)}$ through this range, distributed as $p(r|x^{(s-1)}, N) = Nr^{N-1}$. Changing variables gives:

$$p(x^{(s)}|x^{(s-1)}, N) = N\frac{(x^{(s)})^{N-1}}{(x^{(s-1)})^{N}}, \qquad 0 < x^{(s)} < x^{(s-1)}. \tag{4.24}$$

This defines the joint distribution over the entire sequence:

$$p(\mathbf{x}) = p(x^{(1)}|N)\prod_{s=2}^{S} p(x^{(s)}|x^{(s-1)}, N). \tag{4.25}$$

Note that $p(\mathbf{x})$ depends only on N, it is the same distribution regardless of the problem-dependent likelihood function.

If we knew the $\mathbf{x}$ locations, we could combine these with the likelihood observations $\mathbf{L} = \{L^{(s)}\}_{s=1}^{S}$ and compute the estimate of the normalization $\widehat{\mathcal{Z}}$ given in equation (4.22). Instead we have a distribution over $\mathbf{x}$, which gives a distribution over what this estimator would be:

$$p(\widehat{\mathcal{Z}}|\{L^{(s)}\}, N) = \int \delta(\widehat{\mathcal{Z}}(\mathbf{x}) - \widehat{\mathcal{Z}})\, p(\mathbf{x}|N)\ \mathrm{d}\mathbf{x}. \tag{4.26}$$

We defer philosophizing over the precise meaning of this distribution to subsection 4.9.3. For now we assume that this distribution gives reasonable beliefs over the estimate that would be obtained by quadrature. As can be checked in a given application, the uncertainty of this distribution tends to be much larger than the differences between choices of quadrature scheme. We can therefore, somewhat loosely, take equation (4.26) to be a distribution over $\mathcal{Z}$ itself. Similarly distributions over any other quantity such as $\log \mathcal{Z}$ or posterior expectations can be obtained by averaging quadrature estimates over $p(\mathbf{x}|N)$.

To recap, the key ideas required to understand nested sampling are:

- It would be convenient if we could perform an intractable mapping from the original state space θ to a cumulative quantity x. Numerical computation of $\mathcal{Z}$ or posterior expectations would only involve a one-dimensional function.
- Samples from the prior, subject to a nested sequence of constraints, equation (4.23), give a probabilistic realization of the mapping.
- These samples give a distribution over the results of any numerical computation that could be performed given the change of variables.

No algorithm can solve all problems: some pathological integration problems will always be impossible. For nested sampling the difficulty is in obtaining samples from the nested sequence of constrained priors.

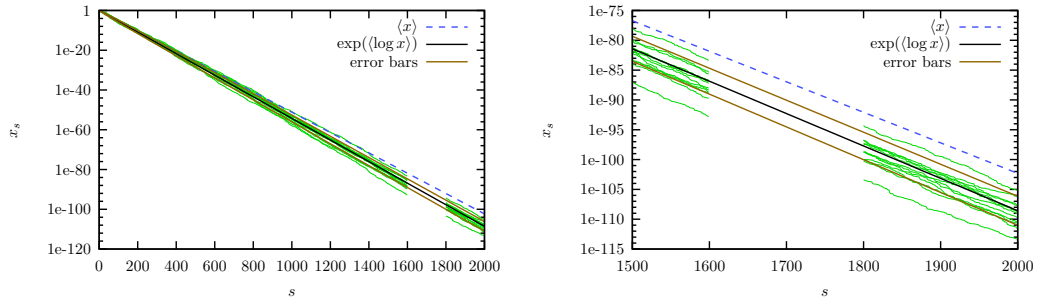


Figure 4.3: The arithmetic and geometric means of $x^{(s)}$ against iteration number, s, for algorithm 4.2 with $N = 8$. Error bars on the geometric mean show $\exp(-s/N \pm \sqrt{s}/N)$. Samples of $p(\mathbf{x}|N)$ are superimposed ($s = 1600 \ldots 1800$ omitted for clarity).

4.4.4 MCMC approximations

The nested sampling algorithm assumes obtaining samples from $\breve{\pi}(\theta; L(\theta^{(s-1)}))$, equation (4.23), is possible. This is somewhat analogous to thermodynamic integration's requirement for samples drawn from a sequence of temperature-based distributions in section 4.2. Annealed importance sampling offers a way to approximately sample from temperature-based distributions and still obtain unbiased estimates of $\mathcal{Z}$. In contrast, nested sampling really needs exact samples from its intermediate distributions, $\breve{\pi}$, for its theoretical justification to be valid.

Rejection sampling of the constrained distributions using π would slow down exponentially with iteration number s. In general we do not know how to sample efficiently from the constrained distributions so, despite the theoretical difficulties, we will replace step 5 of algorithm 4.2 with an approximate sampler using a few steps of a Markov chain. In practice this often works well, but one must remember to be careful when interpreting error bars that ignore the approximation.

We must initialize the Markov chain for each new sample somewhere. One possibility is to start at the position of the deleted point, $\theta^{(s-1)}$, on the contour constraint, which is independent of the other points and not far from the bulk of the required uniform distribution. However, if the Markov chain mixes slowly amongst modes, the new point starting at $\theta^{(s-1)}$ may be trapped in an insignificant mode. Experience suggests it is generally better to start at one of the other $N-1$ existing points inside the contour constraint. These are all (ideally) draws from the correct distribution, $\breve{\pi}(\theta; L(\theta^{(s-1)}))$, so represent modes fairly. Making this new point effectively independent of the point it cloned may take many Markov chain steps. How many to use is an unfortunate free parameter of MCMC-based nested sampling.

4.4.5 Integrating out x

To estimate quantities of interest, we must average over $p(\mathbf{x}|N)$, as in equation (4.26). The mean of a distribution over $\log(\widehat{\mathcal{Z}})$ can be found by simple Monte Carlo estimation:

$$\log(\mathcal{Z}) \approx \int \log(\widehat{\mathcal{Z}}(\mathbf{x}))p(\mathbf{x}|N)\ \mathrm{d}\mathbf{x} \approx \frac{1}{T}\sum_{t=1}^{T}\log(\widehat{\mathcal{Z}}(\mathbf{x}^{(t)})), \qquad \mathbf{x}^{(t)} \sim p(\mathbf{x}|N). \tag{4.27}$$

This scheme is easily implemented for any expectation under $p(\mathbf{x}|N)$, including error bars from the variance of $\log(\widehat{\mathcal{Z}})$. To reduce noise in comparisons between runs it is advisable to reuse the same samples from $p(\mathbf{x}|N)$ (e.g. clamp the seed used to generate them).

A simple deterministic approximation of $p(\mathbf{x}|N)$ is useful for understanding, and also provides fast-to-compute, low-variance estimators. After S iterations $x^{(S)} = \prod_{s=1}^{S} t^{(s)}$, where the $t^{(s)}$ are drawn i.i.d. from $p(t) = Nt^{N-1}$. Dealing with this distribution directly requires substantial work, but notice that $\log(x^{(S)}) = \sum_{s=1}^{S}\log(t^{(s)})$ is a sum of independent terms. The central limit theorem suggests that this should be well characterized by its mean and standard deviation:

$$\log(x^{(S)}) = -S/N \pm \sqrt{S}/N. \tag{4.28}$$

Figure 4.3 shows how well this description summarizes sequences sampled from $p(\mathbf{x}|N)$.

Using the geometric path as a proxy for the unknown $\mathbf{x}$ is a cheap alternative to Monte Carlo averaging over settings (equation (4.27)); see figure 4.4. We might further assume that trapezoidal estimates of integrals are dominated by a number of trapezoids found around a particular iteration s^*. The corresponding uncertainty in $\log\widehat{\mathcal{Z}}$ will be dominated by uncertainty in $\log x^{(s^*)} = -s^*/N \pm \sqrt{s^*}/N$. It usually takes extensive sampling to distinguish $\sqrt{s^*}/N$ from the true standard deviation of the posterior over $\log\widehat{\mathcal{Z}}$.

4.4.6 Degenerate likelihoods

The progress of nested sampling is supposed to be independent of the likelihood function. The procedures that estimate quantities based on nested sampling results rely on this behavior. As an example, the geometric mean path suggests that with $N = 100$ particles nested sampling should typically get to $x = 0.1$ after about $-100\log 0.1 \approx 230$ iterations. Now consider running nested sampling algorithm 4.2 on a problem with the following likelihood function:

$$L(x) = \begin{cases} 0.5 & 0 < x < 0.1 \\ 0.01 & 0.1 < x < 1. \end{cases} \tag{4.29}$$

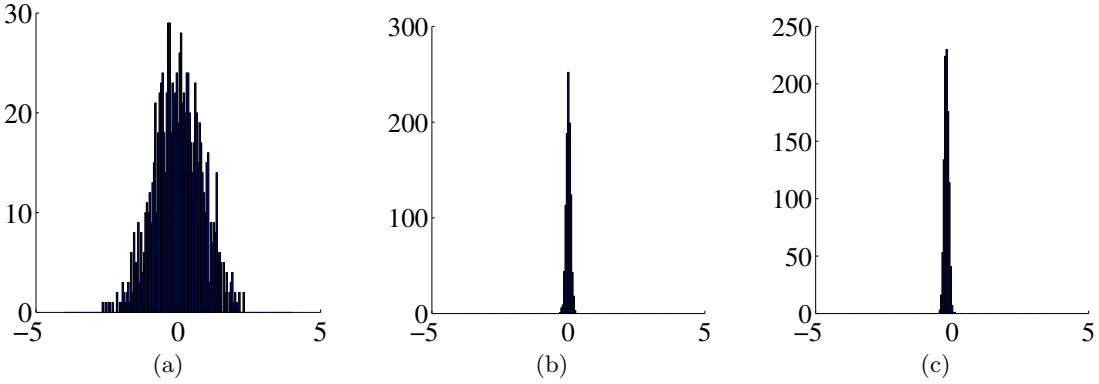


Figure 4.4: Histograms of errors in computing $\mathbb{E}_{p(\mathbf{x})}\left[\log \widehat{\mathcal{Z}}\right]$ under three approximations for integrating over $p(\mathbf{x})$ (1000 random experiments shown). The test system was a 40-dimensional hypercube of length 100 with uniform prior centered on the origin. The log-likelihood was $L = -\theta^\top\theta/2$. Nested sampling used $N = 10$, $S = 2000$. (a) Monte Carlo estimation (equation (4.27)) using $T = 12$ sampled trajectories. (b) $T = 1200$ sampled trajectories. (c) Deterministic approximation using the geometric mean trajectory. In this example the distribution $p(\log \widehat{\mathcal{Z}})$ (from equation (4.26)) has a width ≈ 3 so the errors in finding its mean in (b) and (c) are tolerable.

After drawing $N = 100$ particles from the prior about 90 of them will have $x > 0.1$ and $L = 0.01$. If the inequality $L(\theta^{(s)}) > L(\theta^{(s-1)})$ is strictly enforced all new particles must have $L > 0.01$ and therefore $x < 0.1$. Therefore $x = 0.1$ will be reached in about $s = 90$ iterations, leading to very wrong results. If instead $L(\theta^{(s)}) \geq L(\theta^{(s-1)})$ is enforced the results will still be wrong: $x = 0.1$ will not be reached until about iteration $s = 900$.

The problem is that the mapping set up in subsection 4.4.1 required a total ordering of all the $\mathrm{d}x$ elements of prior mass. It was assumed the likelihood function $L(\theta(x))$ provided a unique key for this sort operation. In many applications this will be sufficient; as long as no two likelihood evaluations in algorithm 4.2 are numerically identical there is no problem. But degenerate likelihoods like equation (4.29) require special treatment. Completely flat plateaus in likelihood functions of continuous variables are rare. But when θ is discrete, finite elements of prior mass carry the same likelihood. This will always introduce plateaus in $L(x)$.

Solving the degeneracy problem requires imposing a total ordering on the $\mathrm{d}x$ elements. Earlier presentations of nested sampling (Skilling, 2006) suggest introducing an auxiliary labeling of the states, $\ell(\theta)$ that resolves likelihood ties. It was suggested these labels could be chosen arbitrarily:

> *"Random samples from Uniform*$[0, 1]$ *suffice, as would a cryptographic identification key derived from* θ*, or almost anything else."*

However, a fixed key does not solve the problem with discrete parameters. If the $0.1 < x < 1$ plateau in the example above originated from a single discrete θ setting, then all the $\mathrm{d}x$ elements in this range would receive the same cryptographic key. Likelihood

ties would still be unresolved. If random labels are employed and always regenerated on repeated observations of the same θ then nested sampling will give the correct results. This is how the code Skilling has made available is implemented[2].

Skilling (2007) mentions a better way of thinking about the random 'labels'. Instead of an extra piece of information attached to each parameter setting, ℓ is an extra input variable for an extended problem with a likelihood that depends on θ and ℓ. The distinction may seem fine: on large problems the same θ location will rarely be revisited. But getting this right is important for obtaining correct answers on small test problems. The joint distribution view also allows a rejectionless algorithm for the Potts example in subsection 4.8.3, which was (tersely) introduced in Murray *et al.* (2006b).

The auxiliary joint distribution is over the variables of interest θ and an independent variable $\ell \in [0, 1]$:

$$\begin{aligned} p(\theta, \ell) = p(\theta) \cdot p(\ell) &= \frac{1}{\mathcal{Z}} L(\theta)\pi(\theta) \cdot \frac{1}{\mathcal{Z}_\ell} L_0(\ell)\pi_0(\ell) \\ &= \frac{1}{\mathcal{Z}} L_J(\theta, \ell)\pi(\theta), \end{aligned} \tag{4.30}$$

where $L_J(\theta, \ell) = L(\theta)L_0(\ell)$, $L_0(\ell) = 1 + \epsilon(\ell - 0.5)$, $\pi_0(\ell) = 1$ and $\mathcal{Z}_\ell = 1$. Standard nested sampling is now possible. At each iteration $L_J(\theta, \ell)$ must increase. We can choose ϵ such that $\log(\epsilon)$ is smaller than the smallest difference in $\log(L(\theta))$ allowed by machine precision. This ensures the auxiliary variable ℓ will only matter when $L(\theta) = L(\theta^{(s-1)})$. The constrained-prior distribution becomes

$$\breve{\pi}_J(\theta, \ell; \theta^{(s-1)}, \ell^{(s-1)}) \propto \begin{cases} \pi(\theta) & L(\theta) > L(\theta^{(s-1)}) \\ \pi(\theta) & L(\theta) = L(\theta^{(s-1)}) \text{ and } \ell > \ell^{(s-1)} \\ 0 & \text{otherwise.} \end{cases} \tag{4.31}$$

MCMC-based implementations of nested sampling require transition operators that leave $\breve{\pi}_J$ stationary. The simplest method is to propose θ with an operator that leaves the original $\breve{\pi}$ constrained prior stationary and then generate $\ell \sim \text{Uniform}[0, 1]$, rejecting the proposal if $\ell < \ell^{(s-1)}$.

A rejectionless method would be preferable and is sometimes possible by marginalizing out ℓ

$$\breve{\pi}_m(\theta; \{\theta, \ell\}^{(s-1)}) \propto \begin{cases} \pi(\theta) & L(\theta) > L(\theta^{(s-1)}) \\ \pi(\theta)(1 - \ell^{(s-1)}) & L(\theta) = L(\theta^{(s-1)}) \\ 0 & \text{otherwise.} \end{cases} \tag{4.32}$$

If the operator used for $\breve{\pi}$ was slice sampling (subsection 2.4.2), then $\breve{\pi}_m$ can also be slice sampled directly. Similarly a Gibbs sampler could easily be adapted to use the conditionals of $\breve{\pi}_m$ instead of $\breve{\pi}$. If Metropolis–Hastings proposals are used it will

[2] Available from `http://www.inference.phy.cam.ac.uk/bayesys/`

sometimes be possible to re-weight $q(\theta' \leftarrow \theta)$ to reduce or eliminate rejections. An example of such a transition operator is developed in subsection 4.8.3.

After generating $\theta^{(s)}$ the auxiliary variable $\ell^{(s)}$ can be drawn from its conditional distribution

$$\breve{\pi}(\ell^{(s)}|\theta^{(s)};\{\theta,\ell\}^{(s-1)}) = \begin{cases} \text{Uniform}[0,1] & L(\theta^{(s)}) > L(\theta^{(s-1)}) \\ \text{Uniform}[\ell^{(s-1)},1] & L(\theta^{(s)}) = L(\theta^{(s-1)}). \end{cases} \tag{4.33}$$

As a minor refinement this could be done lazily if and when it is required by the next iteration.

4.5 Efficiency of the algorithms

This section attempts to compare the computational efficiency of three significant algorithms for estimating normalizing constants. General comparisons of Monte Carlo algorithms are hard to make as they are designed for use with problems where ground truth is unavailable. Also some algorithms are targeted at a particular class of problem, making performance comparisons on other classes unfair.

All three normalizing constant methods considered below are designed to deal with problems in which the base measure $\pi(\theta)$ is much more diffuse than the compact target distribution $p_t(\theta)$. We look at the scaling of computational cost with the nature of this disparity. The behavior of a multivariate Gaussian toy problem is worked out for each algorithm. The base distribution π is vaguely distributed with precision τ_π,

$$\pi(\theta) = \frac{1}{\mathcal{Z}_\pi} \exp\left(-\frac{\tau_\pi}{2}\sum_{d=1}^{D}\theta_d^2\right). \tag{4.34}$$

The target distribution p_t has higher precision τ_t,

$$\begin{aligned} p_t(\theta) &= \frac{1}{\mathcal{Z}_t} \exp\left(-\frac{\tau_t}{2}\sum_{d=1}^{D}\theta_d^2\right) \\ &= \frac{\mathcal{Z}_\pi}{\mathcal{Z}_t}\pi(\theta)L(\theta), \qquad L(\theta) = \exp\left(-\frac{\tau_t-\tau_\pi}{2}\sum_{d=1}^{D}\theta_d^2\right). \end{aligned} \tag{4.35}$$

We pretend that the normalization constants of the Gaussians are unknown and require all of the algorithms under study to return an approximation to the (log of the) ratio $\mathcal{Z}_t/\mathcal{Z}_\pi = (\tau_t/\tau_\pi)^{(D/2)}$.

Many continuous distributions have locally-Gaussian peaks, so poor performance on the toy problem would be a general cause for concern. We also make some general remarks relevant to non-Gaussian modes. Despite this the theoretical comparison presented here is necessarily limited: unreasonable assumptions are made regarding the

ability to draw samples from intermediate distributions. Obviously this doesn't test the ability of the algorithms to find and then fairly represent isolated modes. This will be probed in the empirical comparisons in the sections that follow. For now the analysis is confined to behavior that does not require details of particular Markov chains used in implementations.

4.5.1 Nested sampling

The scaling behavior of nested sampling can be found without detailed calculation. The algorithm is characterized by an exponential shrinkage of the prior mass under consideration. If the bulk of the posterior mass is found around x^*, then an N-particle nested sampling run reaches dominant terms in the integral after $s^* = -N \log x^*$ iterations. Subsection 4.4.5 explained that the uncertainty in $\log \mathcal{Z}$ is typically $\pm\sqrt{s^*}/N$.

Obtaining a unit uncertainty in $\log \mathcal{Z}$ would require setting the number of particles to $N = \sqrt{s^*}$. For this choice the number of iterations required to reach the target distribution's mass from the prior is $s^* = (\log x^*)^2$.

Most of a high-dimensional Gaussian's mass is within a hyper-spherical shell out at a radius of $\sqrt{D/\tau}$. This means that the bulk of the target distribution is in a shell enclosing a fraction $\sim (\tau_\pi/\tau_t)^D$ of the prior volume. If the prior were uniform then we could immediately say that the bulk of the target distribution is found around $x^* \approx (\tau_\pi/\tau_t)^D$. It turns out[3] that this is a good enough description for Gaussians too: $\log x^* \approx -D \log \tau_t/\tau_\pi$. Therefore, the $s^* = (\log x^*)^2$ computational cost of nested sampling on the toy problem is:

$$\text{Number of samples for unit error, } s^* \approx D^2 \left(\log \tau_t/\tau_\pi\right)^2 . \tag{4.36}$$

The $\mathcal{O}(D^2)$ scaling revealed by this analysis is somewhat disappointing. If each sample costs $\mathcal{O}(D)$ to produce then the total cost for a unit error is $\mathcal{O}(D^3)$. If one knew that each dimension were independent one could estimate the log-normalizer of each Gaussian separately. Making the D estimates have variance $1/D$ would require $\approx D \left(\log \tau_t/\tau_\pi\right)^2$ samples each. But samples of just one variable only cost $\mathcal{O}(1)$ computation, which gives a total cost of $\mathcal{O}(D^2)$ rather than $\mathcal{O}(D^3)$. The reason to resort to Monte Carlo is that real computations do not decompose this way. But we shall see that without detailed knowledge of the distribution annealing does only need $\mathcal{O}(D)$ iterations. The advantages of nested sampling do come at a cost.

[3] According to the χ^2 cumulative distribution function the log-cumulative prior mass inside a radius of $\sqrt{D/\tau_t}$ is $\log \frac{\gamma(D/2,(D\tau_\pi)/(2\tau_t))}{\Gamma(D/2)}$, which numerically matches the scaling of $-D \log \tau_t/\tau_\pi$ to within a small factor over a large range of τ_π/τ_t and D.

4.5.2 Multicanonical sampling

As there are many possible algorithms for constructing an approximate multicanonical ensemble a general comparison to 'the' method is hard. Instead we generously assume that we have obtained (by chance) the ideal function that reweights the target distribution such that the distribution over energy is uniform. This avoids having to consider the details of histogram-building methods or other energy density approximations. The danger is that this section's results may be severely misleading if the dominant computational cost is actually obtaining the reweighting. The analysis here should be seen as a lower bound of the costs involved and could be useful for indicating which method is best for refining initial results, however they were obtained.

By definition the multicanonical ensemble has a uniform marginal distribution over energy. To be defined this distribution must be confined to some finite range, width H say, so $p_{\text{MC}}(E(\theta)) = 1/H$. This ensemble could be used to compute the normalizing constant of a simple target distribution, p_s, which also has a uniform energy distribution but over an energy range with smaller width h. The importance weights, equation (4.16), are a constant with probability h/H and zero otherwise. This distribution gives

$$\text{var}[\log \Delta(p_s)] \approx \frac{H}{Sh} \tag{4.37}$$

after S (effective) samples. Empirically the estimator still exhibits quite heavy tails after the $S = H/h$ samples predicted for unit error, but the expression gives good error bars for larger S. Finding effective widths corresponding to H and h will provide a rule-of-thumb estimate of multicanonical's variance on more complex distributions.

We now derive properties of the multicanonical ensemble for the Gaussian problem. Under the target multivariate Gaussian the energy is

$$E(\theta) = -\log L(\theta) = \frac{\tau_t - \tau_\pi}{2} \sum_{d=1}^{D} \theta_d^2 = \frac{\tau_t - \tau_\pi}{2} R, \tag{4.38}$$

which is proportional to $R = \sum_{d=1}^{D} \theta_d^2$. The square-radius R follows a scaled χ^2-distribution with D degrees of freedom: $p(R) \propto R^{D/2-1} \exp(-\tau_t R/2)$. Weighting by the reciprocal of $p(R)$ makes the distribution over R and the energy uniform, giving the multicanonical ensemble:

$$\begin{aligned} p^*_{\text{MC}}(\theta) &\propto p^*(\theta) \cdot w_{\text{MC}}(\theta) \\ &\propto e^{-\tau_t R(\theta)/2} \cdot R(\theta)^{1-D/2} e^{\tau_t R(\theta)/2} \\ &\propto R(\theta)^{1-D/2} \qquad 0 < R(\theta) < R(H). \end{aligned} \tag{4.39}$$

A maximum energy, H must be imposed for the distribution to be defined. A reasonable choice for the square radius defining this limit is 4σ larger than the prior mean value: $R(H) = (D + 4\sqrt{2D})/\tau_\pi$. This cut-off excludes little of the prior mass from the ensemble.

The target distribution is softly confined to a narrow range of the energy spectrum with an effective width likely to be related to the standard deviation of a χ^2-distribution: $R(h_{\text{eff}}) \sim \sqrt{2D}/\tau_t$.

While the exact variance of the estimator does not have a simple analytical form, a more detailed analysis[4] shows that the guess $H/(Sh_{\text{eff}}) = R(H)/(SR(h_{\text{eff}})$ is basically correct:

$$\operatorname{var}\left[\widehat{\log \mathcal{Z}_t}\right] \approx \frac{1}{S}\frac{\tau_t}{\tau_\pi}\sqrt{\frac{D}{8\pi}}\,. \tag{4.45}$$

Astoundingly the number of effective samples required to obtain a unit error scales only as $\mathcal{O}(\sqrt{D})$. This is better than possible by a method that exploited independence and estimated the normalizer of each of the D Gaussians separately. Perhaps this is a sign that through the weighting function the method has unreasonable *a priori* knowledge of the answer.

Despite the ideal weighting function the scaling with precision ratio τ_t/τ_π is not impressive. As the target distribution's precision grows the multicanonical sampler becomes exponentially worse than nested sampling, equation (4.36). This is a clear indication that the energy spectrum ratio H/h_{eff} can be quite different from the volume ratio that determines nested sampling's performance. The details of a particular target distribution can make either method dramatically better than the other.

Another concern is the cost of each of the S effective samples. The ensemble p_{MC} is a distribution capturing a large energy range which will usually take many steps of MCMC exploration to reach equilibrium. In contrast the intermediate distributions of nested sampling or annealing were designed to be close, so a small number of MCMC steps may be sufficient to equilibrate states at each iteration. The log-probability of

[4] The variance of the two parts (equation (4.16)) of the multicanonical estimator (equation (4.17)) can be approximated as follows:

$$\operatorname{var}[\log \Delta(\tau)] \approx \frac{\operatorname{var}[w_\tau^*]}{S\,\mathbb{E}[w_\tau^*]^2} \approx \frac{\mathbb{E}[{w_\tau^*}^2]}{S\,\mathbb{E}[w_\tau^*]^2} \tag{4.40}$$

The ratio of the target distribution, equation (4.35), and multicanonical distribution, equation (4.39), give importance weights:

$$w_\tau^*(\theta) = R(\theta)^{D/2-1}\exp(-\tau R(\theta)/2). \tag{4.41}$$

Noting the integrals

$$\int_0^\infty w_\tau^*(R)\,\mathrm{d}R = \frac{2^{D/2}\Gamma(D/2)}{\tau^{D/2}} \quad \text{and} \quad \int_0^\infty w_\tau^*(R)^2\,\mathrm{d}R = \tau^{1-D}\Gamma(D-1), \tag{4.42}$$

and that, for suitably large H, these give approximately $R(H)$ times the required expectations we have

$$\operatorname{var}[\log \Delta(\tau)] \approx \frac{R(H)\tau\Gamma(D-1)}{S2^D(\Gamma(D/2))^2} \approx \frac{R(H)\tau}{S\sqrt{8\pi(D-2)}} \quad \text{(Stirling's approx. to both } \Gamma \text{ functions)}. \tag{4.43}$$

Substituting in $R(H)$, assuming large D and using the approximate independence of the separate $\log\Delta$ estimators when $\tau_t \gg \tau_\pi$ gives the variance of the final estimator as

$$\operatorname{var}\left[\widehat{\log \mathcal{Z}_t}\right] \approx \frac{\tau_t/\tau_\pi + 1}{S}\left(\sqrt{\frac{D}{8\pi}} + \frac{2}{\sqrt{\pi}}\right). \tag{4.44}$$

The version in the main text drops further terms for clarity. This footnote is terse because what matters is the simpler H/h_{eff} description in the main text and the following assertion: empirically the expression above reasonably matches the observed estimator variance for $\tau_t \gg \tau_\pi$ in a few or more dimensions.

p_{MC} has a range of

$$\Delta E_{\text{MC}} = (1 - D/2) \log \tau_t/\tau_\pi \tag{4.46}$$

between the typical sets of the target and prior distributions. If $\mathcal{O}(1)$ iterations of MCMC are required to change this energy by 1, then it will take at least $\mathcal{O}(\Delta E_{\text{MC}}^2)$ iterations to equilibrate the multicanonical ensemble by a random walk. This would make the cost of a obtaining a single effective sample from the multicanonical ensemble the same order as the total cost of an entire nested sampling run! The potentially good performance of the multicanonical method can be wiped out by the use of some standard Markov chains.

4.5.3 Importance sampling

We now analyze the performance of combining a bridge of $K+1$ importance sampling estimates as in section 4.2. For the toy Gaussian problem all the intermediate distributions are also Gaussian,

$$\begin{aligned} p_k(\theta) &= \frac{1}{\mathcal{Z}_k} \pi(\theta) L(\theta)^{\beta_k} \\ &= \frac{1}{\mathcal{Z}_k} \exp\left(-\frac{\tau_k}{2} \sum_{d=1}^{D} \theta_d^2\right), \quad \tau_k = (1-\beta)\tau_\pi + \beta\tau_t. \end{aligned} \tag{4.47}$$

This allows computation of the variance of the estimator in equation (4.10) under the ideal case of direct sampling from each p_k.

The log of the importance weights at each level from equation (4.6) become

$$\log w_k^*(\theta) = \frac{\tau_k - \tau_{k+1}}{2} \sum_{d=1}^{D} \theta_d^2. \tag{4.48}$$

The variance of the log weights under p_k is

$$\begin{aligned} \text{var}_{p_k}[\log w_k^*] &= \frac{(\tau_k - \tau_{k+1})^2}{4} \text{var}_{p_k}\left[\sum_{d=1}^{D} \theta_d^2\right] = \frac{(\tau_k - \tau_{k+1})^2}{4} \frac{2D}{\tau_k^2} \\ &= \frac{D}{2}\left(1 - \frac{\tau_{k+1}}{\tau_k}\right)^2. \end{aligned} \tag{4.49}$$

The variance of $\log w_k^*$ only depends on k through the ratio $\alpha_k = \tau_{k+1}/\tau_k$. The variance of $\widehat{\log \mathcal{Z}} = \sum_{k=0}^{K} \log w_k^*(\theta_k)$, is minimized by equalizing all the contributing variances, $\alpha_k = \alpha = (\tau_t/\tau_\pi)^{1/(K+1)}$. The resulting estimator has variance

$$\begin{aligned} \text{var}\left[\log \widehat{\mathcal{Z}}\right] &= \frac{D(K+1)}{2}\left(1 - \left(\frac{\tau_t}{\tau_\pi}\right)^{1/(K+1)}\right)^2 \\ &\approx \frac{D}{2(K+1)}\left(\log \frac{\tau_t}{\tau_\pi}\right)^2, \quad \text{for large } K. \end{aligned} \tag{4.50}$$

The approximation becomes accurate very quickly. The amount of computation ($\propto K+1$) required for unit variance scales with dimensionality of the problem as $\mathcal{O}(D)$:

$$\text{Number of samples for unit error, } (K+1) \approx \frac{D}{2} \left(\log \tau_t/\tau_\pi\right)^2 . \tag{4.51}$$

Comparing to equation (4.36) we see that the τ-dependence of the computational cost is the same as nested sampling and that the scaling with dimensionality is better. This result is not strongly tied to the tail behavior of a Gaussian. Redoing the calculations for distributions $p^* \propto \exp(-\tau/2 \sum_d |\theta_d|^p)$ only changes the number of samples required slightly to $(D/p)(\log \tau_t/\tau_\pi)^2$.

Before hastily discarding nested sampling, remember the caveats at the beginning of this section. Practical realities of Markov chain based samplers will make performance worse than predicted here. The Potts model (subsection 1.1.3) will be an example of a distribution, which while easily sampled by multicanonical and nested sampling, is totally unapproachable with temperature-based annealing methods.

As with multicanonical sampling we were very generous to annealing by assuming the free parameters defining the annealing schedule were chosen optimally. We were unlikely to guess

$$\beta_{k+1} = \frac{\tau_\pi(\alpha-1)}{\tau_t - \tau_\pi} + \alpha\beta_k, \quad \alpha = \left(\frac{\tau_t}{\tau_\pi}\right)^{1/(K+1)} . \tag{4.52}$$

Indeed the optimal schedule balances variances, which through equation (4.12) are intimately related to $\mathcal{Z}$ itself. We cannot know the optimal schedule without already having solved the problem.

Some work is required to find a good schedule; default choices may not be sufficient. For example, the linear annealing schedule $\beta_k = k/(K+1)$ gives[5]

$$\text{var}_{\beta_k = k/(K+1)} \left[\widehat{\log \mathcal{Z}}\right] \approx \frac{D}{2K} \frac{(\tau_t - \tau_\pi)^2}{\tau_t \tau_\pi}. \tag{4.53}$$

For large τ_t this is exponentially worse than the correct schedule and nested sampling.

4.6 Constructing annealing schedules

Various families of annealing schedule have been used in the literature. While demonstrating AIS on some example distributions, Neal (2001) spliced together a linear schedule at high temperatures with a geometric schedule at lower temperatures. This is not too far from the behavior of equation (4.52), the optimal schedule for some well-behaved

[5]Found by a large-K integral approximation to the discrete sum over equation (4.49) and also checked numerically.

posteriors. Beal (2003) suggested a family of non-linear annealing schedules

$$\beta_k = \frac{e_\tau k/K}{e_\tau + 1 - k/K}, \tag{4.54}$$

where the "linearity parameter" e_τ makes the schedule approximately linear for large values of e_τ but dwells for longer at high temperatures for small positive values. Another *ad hoc* non-linear scheme is given by Kuss and Rasmussen (2005), where a linear schedule is raised to the power of four: $\beta_k = (k/K)^4$, $k = 0 \dots K$.

Each of the above schemes were presumably the result of some experimentation. Preliminary runs are required to check that a proposed schedule, e.g. "fourth power of a linear schedule", is adequate and to eliminate others. Also any free parameters must be found, such as the number of levels, linearity setting e_τ or the change-point between a linear and geometric schedule. Algorithms for these fitting procedures could be, although rarely are, described in detail. Part of the difficulty is deciding how to include results from earlier, higher variance runs. Often these are only run informally and simply discarded. Regardless of how an annealing schedule is chosen, the selection should ideally be performed automatically. Otherwise computer time may become irrelevant compared to the time required by manual adjustments.

An annealing schedule should control the variance of the log weights,

$$\mathrm{var}_{\beta_k}[\log w_k^*] = (\beta_{k+1} - \beta_k)^2 \, \mathrm{var}_{\beta_k}[\log L(\theta)] \, . \tag{4.55}$$

Rearranging gives a recurrence relationship for the inverse temperatures,

$$\beta_{k+1} = \beta_k + \sqrt{\frac{v}{\mathrm{var}_{\beta_k}[\log L(\theta)]}} \, , \tag{4.56}$$

which achieves a given target variance $v = \mathrm{var}[\log w^*]$ on each weight. This update rule is applied from $\beta_0 = 0$ until an inverse temperature of 1 is reached. The variance of the $\widehat{\log \mathcal{Z}}$ estimator, equation (4.10), is $(K+1)v$. A binary search on the control parameter v can find a balanced annealing schedule with a user chosen total variance or computational cost.

Implementing this algorithm for constructing annealing schedules requires the variance of the log-likelihood or energy at each β_k considered. These quantities are not available exactly, otherwise $\log \mathcal{Z}$ would be known, so the variances must be approximated from some preliminary experiments. Running MCMC at each β_k considered would be too costly. One could try to interpolate results measured at a preliminary range of temperatures. Alternatively statistics at any temperature are available from a single run of either multicanonical or nested sampling.

We propose using nested sampling to set a schedule for AIS. The details are given in algorithm 4.3. Sampling from the multicanonical ensemble would need a large set of

control parameters to be set. Nested sampling can be run before annealing as it only has two control parameters, N and the amount of effort to put into sampling at each iteration. Annealed importance sampling can then be run with a schedule estimated from nested sampling. Comparing the annealing results to those predicted by nested sampling could uncover problems with Markov chains that would go unnoticed using a single method.

Algorithm 4.3 Construction of an annealing schedule from nested sampling
Inputs: Total target variance V_{target}, num. nest particles N, numerical tolerance tol.

1. Run nested sampling algorithm 4.2, obtaining $\{\theta^{(s)}\}$.
2. Assuming $x^{(s)} = \exp(-s/N)$, compute weights $w_q^{(s)}$ as in equation (4.22).
3. $v_{\text{max}} \leftarrow V_{\text{target}}$
4. $v_{\text{min}} \leftarrow 0$
5. $K \leftarrow 0$, $\beta_0 \leftarrow 0$
6. **while** $(v_{\text{max}} - v_{\text{min}})(K+1)/V_{\text{target}} > \text{tol}$
7. $\quad v_{\text{trial}} \leftarrow (v_{\text{max}} + v_{\text{min}})/2$
8. $\quad$ Create discrete proxy for $p(\theta; \beta_k)$ distribution: $\tilde{p}_{\beta_K} \propto w_q L(\theta^{(s)})^{\beta_K}$.
9. $\quad \beta_{K+1} \leftarrow \beta_K + \sqrt{\frac{v}{\text{var}_{\tilde{p}_{\beta_K}}[\log L(\theta)]}}$.
10. $\quad$ **if** $\beta_{K+1} > 1$ **then**
$\quad\quad$ (passed end of schedule, start again with a higher variance per level)
11. $\quad\quad v_{\text{min}} \leftarrow v_{\text{trial}}$, $K \leftarrow 0$
12. $\quad$ **else if** $(K+1)v_{\text{trial}} > V_{\text{target}}$ **then**
$\quad\quad$ (exceeded target variance, start again with lower variance per level)
13. $\quad\quad v_{\text{max}} \leftarrow v_{\text{trial}}$, $K \leftarrow 0$
14. $\quad$ **else**
15. $\quad\quad K \leftarrow K + 1$
16. $\quad$ **end if**
17. **end while**
18. $\beta_{K+1} \leftarrow 1$
19. **return** annealing schedule $\beta_0 \ldots \beta_{K+1}$

4.7 Markov chains for normalizing constants

Each of the Monte Carlo algorithms in this chapter require sampling from complex distributions: $p(\theta; \beta)$, $\breve{\pi}(\theta)$ or $p_{\text{MC}}(\theta)$. Standard sampling techniques — Metropolis–Hastings, slice sampling, etc. — should apply, but there are some issues special to these algorithms that are worth considering.

4.7.1 Randomize operator orderings

Many MCMC operators concatenate several operators with different behaviors together. Gibbs sampling for example updates each dimension of a parameter vector separately.

Some users prefer to randomize the order of these updates so that the resulting mixture operator maintains detailed balance. But many use a deterministic ordering, if only for convenience of implementation. This is not a good idea with algorithms that start out of equilibrium at each iteration.

The problem is most easily demonstrated by nested sampling with $N = 1$. At each iteration the only particle is by definition on the boundary of the constrained prior $\breve{\pi}$. The first update must increase the likelihood of the particle. Subsequent updates have some freedom to decrease the likelihood again. As only a limited number of Markov chain steps can be performed at each iteration the particle will climb unnaturally fast up the likelihood surface in the direction of the first transition operator.

In subsection 4.8.1 nested sampling is run using a univariate slice sampler applied to each variable in a random order. Initially these experiments used a fixed ordering. The first variable to be updated would systematically become much more constrained than the last, even if by symmetry they were equivalent. Fortunately this pathology is so severe that it quickly made itself known by causing numerical problems and crashing the slice sampling code.

Experiments with spherical Gaussians confirm that annealed importance sampling suffers from a similar problem. Histograms of the unweighted final states obtained show that the statistics of each dimension depend on the order in which they were updated. The AIS weights correct this bias asymptotically, but samples without these artifacts will tend to need lower variance weights. The easiest fix is to randomly permute the Markov chain operators at each iteration.

4.7.2 Changes in length-scale and energy

There is usually a dramatic difference in scale between a prior and posterior. It is unlikely that the same Markov chain operators are appropriate for both, yet annealing has to sample them and all the interpolating distributions in between. Similarly nested sampling has to sample from a sequence of distributions that shrink exponentially in volume from the prior to the posterior and beyond.

Step size parameters in Metropolis–Hastings algorithms must be changed as the algorithm proceeds. It is also profitable to adapt the initial step size of slice sampling. Some authors set a schedule of step sizes by hand, but automatic schemes are clearly desirable. One option is to adapt based on the acceptance rate of the previous iteration or to use parallel chains. For AIS this would require giving up some theoretical correctness, while nested sampling is already in an approximate setting by using a Markov chain at all. Another option for AIS is to adapt the schedule of proposals after each run. Each run is unbiased in $\mathcal{Z}$ using any step-size, but when adapting it is still advisable to discard early runs, which will have higher variance.

There is usually a large change in the scale of probabilities involved between a diffuse prior and a posterior which concentrates mass on a much smaller number of states. Many standard Markov chain operators, such as simple Metropolis and slice sampling, are only able to make small changes in log-probability at each iteration. Depending on the algorithm this may or not be desirable.

The majority of the volume in a high-dimensional solid body is in a thin shell near its surface (MacKay, 2003, p37). For nested sampling this means that much of $\breve{\pi}$'s mass is likely to be close to the likelihood contour surface and large changes in likelihood are not required. Instead we need efficient chains that sample well at close to constant likelihood[6]. Temperature-based distributions have soft constraints that lead to broader distributions over energy, although in many problems they are still constrained to a relatively narrow range.

The multicanonical method samples from a single distribution which has significant overlap with both the prior and posterior. Making the distribution over energies uniform requires making some states much more probable than others under p_{MC}. Simple Metropolis methods are unable to move rapidly between regions of many states with low probability and more compact regions with high probability. The exploration of p_{MC}'s energy spectrum will be characterized by a random walk or slower process. This suggests that equilibrating p_{MC} will need at least $\sim (\Delta E_{\text{MC}})^2$ steps, where ΔE_{MC} is the range of log probabilities under p_{MC} not the original distribution.

Some Monte Carlo algorithms, such as Hybrid Monte Carlo, are able to make larger changes in energy. Hamiltonian dynamics based on p_{MC} could be simulated, as long as the reweighting function is smooth. Nested sampling could also benefit from Hamiltonian dynamics for its large movements in state-space, although $\breve{\pi}$ is not compatible with large changes in energy. Fortunately versions of slice sampling that can use Hamiltonian dynamics on the prior and reflections from constraint boundaries have already been developed (Neal, 2003).

Another important Markov chain operator for dramatic moves in state-space and energy is Swendsen–Wang (subsection 2.4.1). While this algorithm can work at any temperature, it does not allow reweightings of the energy and is not easily modified to sample at near constant energy. By recasting the problem we can develop a version of Swendsen–Wang that will work with multicanonical and nested sampling.

4.7.3 A new version of Swendsen–Wang

The partition functions of the Potts model, equation (1.4), the random cluster model, equation (2.16), and the FKSW joint distribution, equation (2.15), are identical. Also a sample from any of these distributions is easily converted into a sample from one of

[6]For literature searches it is helpful to know that in physics a constant energy distribution is known as a *microcanonical ensemble*.

the others. This allows using any of the distributions to simulate the Potts model and find its normalization, $\mathcal{Z}_{\mathrm{P}}(J,q)$. We focus on the random cluster model.

Assuming identical positive couplings $J>0$ on each edge, we rewrite the random cluster distribution in an unconventional way:

$$p(\mathbf{d}) = \frac{1}{\mathcal{Z}_{\mathrm{N}}} L(\mathbf{d})\pi(\mathbf{d}) \quad \text{where} \tag{4.57}$$

$$\mathcal{Z}_{\mathrm{N}} = \frac{\mathcal{Z}_{\mathrm{P}}(J,q)}{\mathcal{Z}_\pi}\exp(J|\mathcal{E}|), \quad L(\mathbf{d}) = \exp(D\log(e^J-1)), \quad \pi(\mathbf{d}) = \frac{1}{\mathcal{Z}_\pi} q^{C(\mathbf{d})}.$$

Under this factorization the "energy" is minus the total number of bonds $D=\sum d_{ij}$ and the inverse-temperature, $\log(e^J-1)$, is set by the coupling parameter J. As in subsection 2.4.1, $C(\mathbf{d})$ is the number of connected components or clusters formed by the bonds $\mathbf{d}$.

Algorithm 4.4 gives an MCMC operator to update the bond configuration, $\mathbf{d}\to\mathbf{d}'$. The stationary distribution is a weighted prior $\propto w_{\mathrm{MC}}(D)\pi(\mathbf{d})$, where w_{MC} could be multicanonical weights, or set to $w_{\mathrm{MC}}=1$ for prior sampling, or set to zero and one to sample from the prior subject to constraints on D.

Algorithm 4.4 Swendsen–Wang for weighted bond configurations

1. Create a random coloring, $\mathbf{s}$, uniformly from the $q^{C(\mathbf{d})}$ colorings satisfying the bond constraints $\mathbf{d}$, as in the Swendsen–Wang algorithm.
2. Count sites that allow bonds, $E=\sum_{(ij)\in\mathcal{E}}\delta_{s_i,s_j}$.
3. Draw D' from $T(D';E(\mathbf{s})) = \frac{1}{\mathcal{Z}_T(\mathbf{s})} w_{\mathrm{MC}}(D')\binom{E(\mathbf{s})}{D'}$.
4. Throw away the old bonds, $\mathbf{d}$, and pick uniformly from one of the $\binom{E(\mathbf{s})}{D'}$ ways of setting D' bonds in the E available sites.

The probability of proposing a particular coloring and new setting of the bonds is

$$\begin{aligned} T(\mathbf{s},\mathbf{d}'\leftarrow\mathbf{d}) &= T(\mathbf{d}';\mathbf{s},D')\,T(D';E(\mathbf{s}))\,T(\mathbf{s};\mathbf{d}) \\ &= \frac{1}{\binom{E(\mathbf{s})}{D'}} T(D';E(\mathbf{s}))\frac{1}{q^{C(\mathbf{d})}} = \frac{w_{\mathrm{MC}}(D')}{\mathcal{Z}_T(\mathbf{s})\,q^{C(\mathbf{d})}}. \end{aligned} \tag{4.58}$$

Summing over all possible intermediate colorings, the probability of starting with D bonds $\mathbf{d}$ and ending with D' bonds $\mathbf{d}'$ is proportional to

$$\begin{aligned} T(\mathbf{d}'\leftarrow\mathbf{d})\; w_{\mathrm{MC}}(D)\,\pi(\mathbf{d}) &= \pi(\mathbf{d})\,w_{\mathrm{MC}}(D)\sum_{\mathbf{s}} T(\mathbf{s},\mathbf{d}'\leftarrow\mathbf{d}) \\ &= \frac{w_{\mathrm{MC}}(D)\,w_{\mathrm{MC}}(D')}{\mathcal{Z}_\pi\,q^{C(\mathbf{d})}\,q^{C(\mathbf{d}')}}\sum_{\mathbf{s}}\frac{1}{\mathcal{Z}_T(\mathbf{s})}. \end{aligned} \tag{4.59}$$

This expression is symmetric under the exchange of $(D,\mathbf{d})$ and $(D',\mathbf{d}')$. Therefore the transition operator satisfies detailed balance with respect to the weighted prior. It is

also ergodic. Proof: with finite probability all s_i are given the same color, then any D' with non-zero weight is possible, in turn all allowable $\mathbf{d}'$ have finite probability.

When performing nested sampling using the weighted prior representation the likelihood constraints in $\breve{\pi}$ are thresholds on the total number of bonds D. This can be realized by setting $w_{\mathrm{MC}} = 0$ for states with fewer bonds. Many states have identical D, which requires careful treatment, as discussed in subsection 4.4.6. The simple implementation that draws a random key for each state will lead to some rejections when proposing moves to a state with the same D on the constraint surface. Sampling without rejections can be achieved by setting the weights such that algorithm 4.4 leaves the auxiliary constrained distribution $\breve{\pi}_m$, equation (4.32), invariant.

The number of bonds has previously been identified as a useful energy-like target for reweighting (Janke and Kappler, 1995). In that work the bonds were updated by single site Gibbs sampling rather than the block-Gibbs sampling move of step 4. This does not allow simulation of the fixed D ensemble, or rapid exploration near fixed D. Single site updates are easier to implement however, and would become more attractive on systems that allow a different J_{ij} on each edge. In this case the implementation of global updates is much more involved.

4.8 Experiments

Detailed comparisons of nested sampling and more established Monte Carlo techniques are not currently available in the literature. Anecdotally, nested sampling has already been useful in astronomy. Mukherjee *et al.* (2006) and Shaw *et al.* (2007) claim nested sampling gives speed-ups of 1–2 orders of magnitude over annealing based methods. The annealing approach that was cited in both papers was very carefully implemented (Beltrán *et al.*, 2005). These results are somewhat surprising given that both nested sampling and annealing follow a sequence of increasingly constrained ensembles and theoretically seem quite similar. If anything nested sampling seems slightly worse, although this should be a small effect for the $D = 5$ astronomy problems that were tested.

The focus of this section is not applications, but well understood test problems. Hopefully these will give some better insight into the relative merits of the approaches.

4.8.1 Description of slice sampling experiments

A set of experiments were performed on some continuous distributions that are amenable to slice sampling. This allowed the same MCMC code to be used within each algorithm. The distributions tested are described first and then details of the algorithms. The results, which appear in table 4.2, are discussed in the next section.

Gaussian base and target distributions as considered theoretically in section 4.5 were run in ten dimensions. These experiments reveal the actual performance when confounded by interactions with a particular MCMC operator. In version **(a)** we set the standard deviation of the base distribution to be 10 times wider than the target, i.e., $\tau_t/\tau_\pi = 100$. In version **(b)** we made the prior 100 times wider: $\tau_t/\tau_\pi = 10000$.

t-distribution: this was included as another simple standard distribution with different tail behavior. The target distribution was a ten-dimensional multivariate-t with five degrees of freedom. The base distribution was Gaussian with $\tau_\pi = 1/100$.

Two modes: a mixture of Gaussians as tested in Neal (1998a, 2001).

$$p_t(\theta)\mathcal{Z}_t = \exp\left[-\frac{1}{2}\sum_{d=1}^{6}\frac{(\theta_d-1)^2}{0.1^2}\right] + 128\exp\left[-\frac{1}{2}\sum_{d=1}^{6}\frac{(\theta_d+1)^2}{0.05^2}\right] \tag{4.60}$$

The base distribution was a unit six-dimensional Gaussian, $p_\pi = \mathcal{N}(0, \mathbb{I})$.

Deceptive: this two-dimensional problem bridges from a spherical Gaussian distribution with precision $1/25^2$ to a two-dimensional mixture of 4292 Gaussians taken from Neal (1994, 1996a).

$$\begin{aligned} p_t(\theta)\mathcal{Z}_t = &\sum_{i=-5}^{+5}\sum_{j=-5}^{+5}\exp\left(-\frac{|\theta-\mu_{1,i,j}|^2}{2\sigma^2}\right) + \sum_{i=-5}^{+5}\sum_{j=-5}^{+5}\exp\left(-\frac{|\theta-\mu_{2,i,j}|^2}{2\sigma^2}\right) \\ &+ \sum_{i=-22}^{+22}\sum_{j=-22}^{+22}\exp\left(-\frac{|\theta-\mu_{3,i,j}|^2}{2\sigma^2}\right) + \sum_{i=-22}^{+22}\sum_{j=-22}^{+22}\exp\left(-\frac{|\theta-\mu_{4,i,j}|^2}{2\sigma^2}\right) \end{aligned} \tag{4.61}$$

where $\sigma = 0.001$, and the mixture components are in four groups:

$\mu_{1,i,j} = (0.0025i + 15,\ 0.0025j + 15)$ 121 means in the upper-right quadrant,

$\mu_{2,i,j} = (0.1500i - 15,\ 0.1500j + 15)$ 121 means in the upper-left quadrant,

$\mu_{3,i,j} = (0.0025i - 15,\ 0.0025j - 15)$ 2025 means in the lower-left quadrant,

$\mu_{4,i,j} = (0.1500i + 15,\ 0.1500j - 15)$ 2025 means in the lower-right quadrant.

This target distribution is exceedingly challenging and more pathological than experienced in many statistical problems. It is interesting however. The different spacings of the means make it hard for algorithms to know from a distance where the bulk of the probability mass is. This highlights differences between the mass-finding heuristics implicitly performed by the algorithms.

In all cases one Markov chain update consisted of a simple univariate slice sampler applied once to each variable in a new random order at each iteration. A linear stepping out procedure was employed with an initial step-size equal to one or to the range of settings currently occupied by particles being simulated in parallel. Some additional choices were needed by each method.

Nest: nested sampling runs had two free parameters: the number of particles N and the number of slice-sampling steps used to update $\breve{\pi}$ at each iteration. Unless $N = 1$ each

new particle was initialized at one of the $N-1$ particles already satisfying the likelihood constraint. As slice samplers always move and there are no plateaus in these problems' likelihood functions the details in subsection 4.4.6 were not required. Rather than setting a number of iterations, S, we terminated the nested sampler when the estimate of $\log \mathcal{Z}$ appeared to have converged. In particular, we used a geometric approximation for $\mathbf{x}$ to estimate $\log \mathcal{Z}$ with equation (4.22) and terminated when the remaining prior mass appeared to contribute a fraction of only 10^{-6} to the sum. The reported results used 1000 Monte Carlo samples of $\mathbf{x}$ in equation (4.27). These are very similar to those obtained from the geometric mean approximation but also provide quantiles of the predictive posterior over $\log \mathcal{Z}$.

AIS$_{r,c}$: annealed importance sampling has two free parameters in addition to its annealing schedule. An experiment was repeated r times, each run using c parallel chains. When $c > 1$ the parallel chains are used to set the initial step sizes of the slice sampler. For multimodal distributions this is not strictly justified within the AIS framework but it seems unlikely the results will differ greatly from using preliminary runs instead. An AIS initialization of "(e, nest $N = n$. $K = k$)" estimated an annealing schedule from a nested sampling run with the given N and 1 step per iteration. The given value of K intermediate temperatures was needed for target standard error e. The details of this procedure and the linear and fourth power schedules were given in section 4.6.

Multicanonical: ten slice sampling chains initialized from the prior were run in parallel on the multicanonical ensemble. Simple error bars were calculated from the variance of the estimates from each chain. On the Gaussian problem the analytically-derived weighting function that sets a uniform distribution over energies (equation (4.39)) could be used. We also tried setting the multicanonical weights by estimating the distribution over energies from a nested sampling run.

4.8.2 Discussion of slice sampling results

The experiments with a Gaussian highlight differences between the theoretical ideals in section 4.5 and the realities of MCMC. Nested sampling is fastest with $N = 1$, which was tried as a preliminary run. The predictions from one or ten slice sampling sweeps per iteration are imprecise as one might expect, but also overconfident: the posterior over $\log \mathcal{Z}$ has negligible overlap with the true answer. Increasing the amount of MCMC sampling to 100 slice sampling steps per iteration overcomes the bias, but at a large computational cost.

Increasing the number of particles N gives more precise answers, and increases accuracy. There are three reasons: 1) each particle can start at one of the other particles which are supposed to be drawn from the target $\breve{\pi}$ distribution; 2) the more particles there are the easier it is to forget exactly which one was copied; 3) the $\breve{\pi}$ distributions change more slowly with larger N. The answer from $N = 19$ with one step of sampling gives

Table 4.2: Empirical behavior of slice-sampling-based nested sampling, AIS and "multicanonical" sampling. The distributions and methods are described in the main text, subsection 4.8.1. The results are discussed in subsection 4.8.2.

Distribution	**Method (initialization)**	$\log L(\theta)$ **evals.**	$\log \mathcal{Z}$
Gaussian (a)	**Truth**		**9.18**
	Nest $N=1$, 1 step	2098	21.9 ± 2.7
	Nest $N=1$, 10 steps	13563	26.4 ± 1.4
	Nest $N=1$, 100 steps	349570	12.5 ± 3.9
	Nest $N=19$, 1 step	29969	9.16 ± 0.96
	Nest $N=19$, 10 steps	300755	9.37 ± 0.95
	Nest $N=212$, 1 step	320983	9.18 ± 0.30
	Nest $N=212$, 10 steps	3216381	9.12 ± 0.29
	$\mathrm{AIS}_{1,1}$ (0.3, nest $N=212$. $K=1190$)	185405	$8.96 \pm$ —
	$\mathrm{AIS}_{10,1}$ (0.3, nest $N=212$. $K=122$)	189494	8.98 ± 0.30
	$\mathrm{AIS}_{100,1}$ (0.3, nest $N=212$. $K=15$)	241692	6.48 ± 0.48
	$\mathrm{AIS}_{1,10}$ (0.3, nest $N=212$. $K=122$)	60148	9.35 ± 0.43
	$\mathrm{AIS}_{1,100}$ (0.10, nest $N=212$. $K=122$)	593793	9.01 ± 0.16
	$\mathrm{AIS}_{1,100}$ (linear, $K=122$)	594301	8.23 ± 0.23
	$\mathrm{AIS}_{1,100}$ (fourth power, $K=122$)	592592	9.24 ± 0.15
	Multicanonical (nest $N=212$)	647144	9.82 ± 0.19
	Multicanonical (nest $N=212$)	6495834	9.71 ± 0.07
	Multicanonical (analytic weighting)	500963	9.66 ± 0.47
	Multicanonical (analytic weighting)	5001964	9.23 ± 0.08
Gaussian (b)	**Truth**		**9.18**
	Nest $N=200$, 1 step	531556	8.74 ± 0.47
	Multicanonical (analytic weighting)	5014383	9.70 ± 0.68
t-distribution	**Truth**		**9.69**
	Nest $N=10$, 1 step	11426	11.13 ± 0.83
	Nest $N=93$, 1 step	97847	9.66 ± 0.28
	Nest $N=946$, 1 step	989453	9.71 ± 0.10
	$\mathrm{AIS}_{1,100}$ (0.03, nest $N=10$. $K=193$)	978189	9.70 ± 0.05
	$\mathrm{AIS}_{1,100}$ (linear, $K=193$)	973347	9.65 ± 0.05
	$\mathrm{AIS}_{1,100}$ (fourth power, $K=193$)	952803	9.80 ± 0.08
Two modes	**Truth**		**-7.20**
	Nest $N=10$, 1 step	8212	-10.1 ± 1.2
	Nest $N=158$, 1 step	115968	-8.24 ± 0.29
	Nest $N=1455$, 1 step	1197722	-6.47 ± 0.11
	Nest $N=1849$, 1 step	1573194	-7.44 ± 0.09
	Nest $N=7000$, 1 step	5972352	-7.29 ± 0.05
	$\mathrm{AIS}_{1,100}$ (0.1, nest $N=1849$. $K=101$)	319831	-8.16 ± 0.13
	$\mathrm{AIS}_{1,1000}$ (0.1, nest $N=1849$. $K=13$)	419767	-8.69 ± 0.18
	$\mathrm{AIS}_{1,1000}$ (0.02, nest $N=1849$. $K=197$)	6321987	-7.18 ± 0.20
Deceptive	**Truth**		**-3.61**
	Nest $N=16000$, 1 step	10046143	-3.73 ± 0.03
	$\mathrm{AIS}_{1,1000}$ (0.02, nest $N=4523$. $K=579$)	10051190	-3.66 ± 0.23

a much better answer than $N = 1$ with 100 steps and is also an order of magnitude cheaper. Even with only one sampling step per iteration the estimates and error bars with $N = 19$ and $N = 212$ are indistinguishable from those obtained by exact sampling from the constrained priors[7].

Three standard annealed importance sampling runs were performed with the same target error of 0.3: $\mathrm{AIS}_{1,1}$, $\mathrm{AIS}_{10,1}$ and $\mathrm{AIS}_{100,1}$. Obtaining the target error from only one run requires a very long annealing schedule, $K = 1190$. This gave similar results to 10 shorter runs with $K = 122$, but error bars are much more easily obtained from multiple runs. Increasing the number of runs to 100 required shortening the annealing schedule further for a similar target error or computational cost. The short $K = 15$ runs had larger errors than predicted and unreliable error bars. This schedule was designed assuming that the algorithm would keep close to the equilibrium distributions defined in the annealing schedule. In this case there were not enough bridging distributions for this approximation to be accurate.

The result with $\mathrm{AIS}_{1,10}$, $K = 122$, which uses ten parallel chains is similar to the result from separate runs, $\mathrm{AIS}_{10,1}$, but with many fewer likelihood evaluations. In the absence of a population of points the slice sampling code used an initial step size of one, which is inappropriately small for this problem at high temperatures. A user not prepared to adapt step-sizes based on a population should find some way to set them appropriately.

The sequence of three $\mathrm{AIS}_{1,100}$ runs with $K = 122$ are deliberately at higher precision than necessary so that the error bars are somewhat reliable and reproducible. Something that is sadly not true of the lower precision AIS runs. These confirm that a linear annealing schedule is worse than one that dwells for longer at higher temperatures. In this case the fourth power schedule quite closely follows that set by nested sampling and has very similar performance.

The first multicanonical result with weights set by nested sampling seems to give comparable uncertainty to AIS for the amount of computer effort. However, running the chains for ten times longer reveals problems with this "multicanonical" estimator. We had set the weights by estimating the probability mass between each pair of likelihood values $L^{(s)}$ and $L^{(s+1)}$ visited by nested sampling. For simplicity we used the deterministic approximation in equation (4.28). This gives noisy estimates of the ideal multicanonical weights which, it turns out, are not good enough. We could attempt to smooth the approximate multicanonical weighting function. Instead, for the Gaussian case, we went directly to the "correct" multicanonical ensemble, equation (4.39). This confirmed that the problem with the estimator was the choice of weighting function and gives multicanonical a performance somewhere in between nested sampling and AIS on the Gaussian (a) problem. Even with the ideal weighting multicanonical fails to equilibrate and fails to give reasonable error bars on the Gaussian (b) problem. This is to be expected given the method's poor scaling with precision ratio τ_t/τ_π. We did

[7] We were able to confirm this because $\breve{\pi}$ is quite tractable for this toy problem.

not continue to try multicanonical: adapting the weighting function beyond a nested sampling initialization seems important and this would complicate the comparison.

Turning to the t-distribution example we see behavior different from the Gaussian case. The annealing schedule estimated from nested sampling is closer to linear than to a fourth power. Moreover, the linear schedule and that produced from nested sampling are reproducibly better than the fourth power. Notice the reversal from the Gaussian experiment. A good annealing schedule was estimated cheaply from nested sampling with $N = 10$. Obtaining as good answers as AIS with nested sampling alone would be more expensive. Exactly how much is hard to tell from individual runs because the error bars can be unreliable.

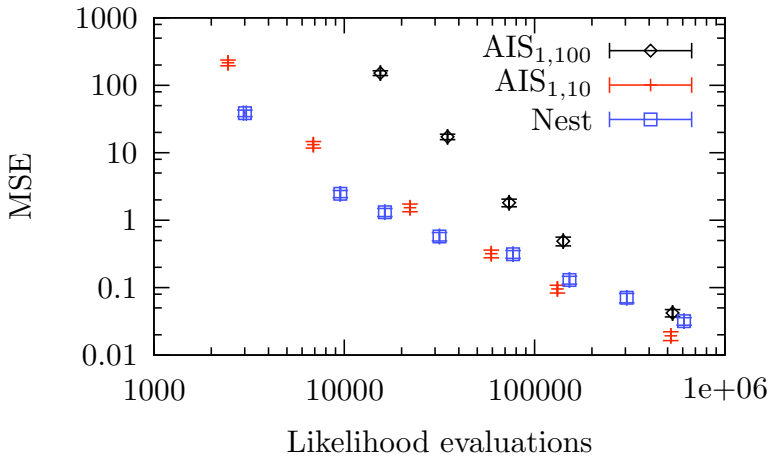
A more reliable comparison of nested sampling and AIS can be made based on many repeated runs. Nested sampling was run 100 times each over a range of N and compared to AIS with 100 runs of schedules of varying lengths set by nested sampling. Figure 4.5a shows that the actual performance, measured as a mean squared error of the $\log \mathcal{Z}$ estimates, was similar for the two methods. Nested sampling is slightly more accurate at low computational costs, being taken over by AIS at higher precisions. Increasing the number of AIS runs at the expense of annealing schedule length performs worse for the same computational cost. This difference goes away at high precisions, but may be a concern when many runs are required on multimodal problems.

The main problem with AIS is that at lower precisions simple estimates and error bars based on the mean and variance of the importance weights are unreliable. Figure 4.5b shows large biases in estimates for $\log \mathcal{Z}$. Figure 4.5c shows that the error bars are too small on average. Jackknife estimates (not shown) are sometimes slightly better calibrated but give very similar results. For a user the error bars are often very important, so in this case the nested sampling estimator is the favorable choice.

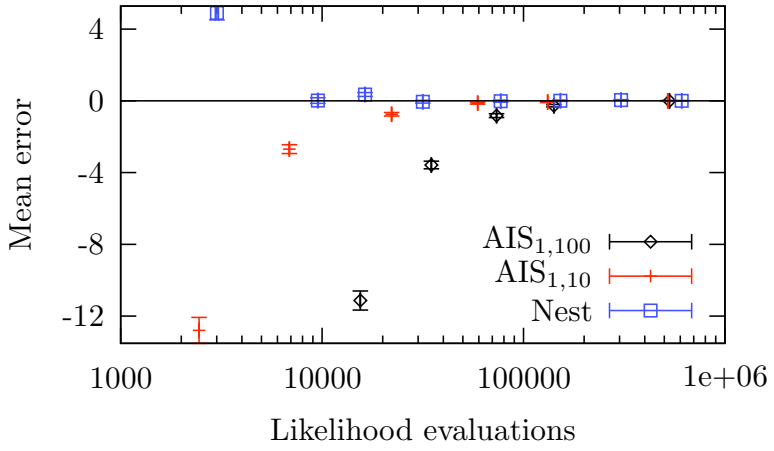
The peaks of the "two modes" target distribution are highly separated at low temperatures. The fraction of AIS chains ending in the mode in the positive orthant is ≈ 0.97, compared to its actual probability mass of a third. Through weighting these runs AIS converges to the correct answer once enough sufficiently-long chains have been run. Further checks show that it also estimated the mass of each mode correctly. The same is true for nested sampling, the results from large N obtain nearly the correct relative masses of the modes and in turn the overall normalization. Detailed analysis still shows some signs of bias; the posterior overlap with the correct answer is smaller than apparent from the scalar error bars in the table.

Both methods find "two-modes" difficult because it is not feasible to sample correctly from $\breve{\pi}$ at late iterations or from low temperature distributions. AIS relies on reweighting over many runs, nested sampling requires many particles to maintain a fair representation of $\breve{\pi}$.

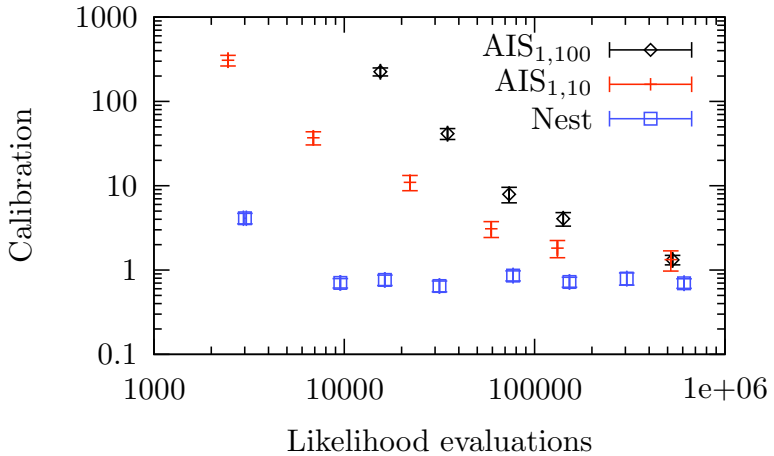
Even after $\approx 10^7$ likelihood evaluations, both nested sampling and AIS have problems



(a)



(b)



(c)

Figure 4.5: Average behavior of AIS and nested sampling over 100 runs for a range of N or target errors. The x-axis gives computational cost in likelihood evaluations. (a) Mean square error of $\log \mathcal{Z}$ estimate. (b) Mean error or bias of point estimate. (c) Mean square normalized error (error divided by error bar width), a measure of calibration.

Table 4.3: Estimates of the deceptive distribution

Quadrant	True probability	Nest estimate	AIS estimate
upper-left	0.0282	0.0285 ± 0.0002	0.0305 ± 0.0080
upper-right	0.0282	0.0214 ± 0.0002	0.0002 ± 0.0002
lower-left	0.4718	0.4039 ± 0.0006	0.47 ± 0.12
lower-right	0.4718	0.5482 ± 0.0005	0.50 ± 0.12

with the deceptive distribution. Nested sampling has an error bar on $\log \mathcal{Z}$ that is far too small. AIS's estimate is consistent with the correct answer, although as with "two modes" with a higher standard error than targeted due to poor mixing at low temperatures. Additional problems are revealed by looking at the probability distribution over the four quadrants containing each cluster of modes, table 4.3. Again the nested sampling run is far too confident. In this case AIS's error bars are generally much better, although it is very certain that the upper-right quadrant has much less probability mass than it actually does. Runs of nested sampling with smaller N show that it too can easily lose the upper-right quadrant.

Nested sampling's answers are wrong, according to its error bars, due to the Markov chain approximation. At late iterations with high likelihood constraints the slice sampler is unable to equilibrate $\breve{\pi}$ and is relying on a large number of particles to provide a good starting point. On this problem the inaccuracy of this approximation does not reduce as fast as the size of the error bars with a large number of points.

Increasing the number of slice sampling steps per iteration does not solve the problem because it would take an unrealistically large number to move between isolated modes. However, proposals that take all of the particles' positions into account could help. Shaw *et al.* (2007) uses an approximate rejection sampler based on uniform samples within ellipsoidal fits to clusters of the existing particles. No problems with biases were reported using this method, although clearly problems could still occur when the ellipsoids fail to correctly capture $\breve{\pi}$. In general ellipsoids could be used as Metropolis proposals, as part of several and various attempts to equilibrate a particle.

4.8.3 The Potts model

The Potts model was introduced in subsection 1.1.3. It describes a class of undirected graphical models over discrete variables $\mathbf{s}$. The variables take on one of q 'colors' and the model has a temperature-like "coupling" parameter J. Gibbs sampling updates of $\breve{\pi}$ are the simplest way to implement (approximate) nested sampling. We also try cluster-based updates, subsection 4.7.3. While physicists tend to be interested in a broader range of quantities, we focus here on the normalization constant $\mathcal{Z}_{\mathrm{P}}(J, q)$, where the discrete variables $\mathbf{s}$ are the θ variables that need to be integrated (i.e. summed) over.

Table 4.4: Partition function results for 16×16 Potts systems (see text for details).

Method	$q = 2$ (Ising), $J = 1$	$q = 10$, $J = 1.477$
Gibbs AIS	7.1 ± 1.1	1.5
Swendsen–Wang AIS	7.4 ± 0.1	1.2
Gibbs nested sampling	7.1 ± 1.0	12.2 ± 2.4
Random-cluster nested sampling	7.1 ± 0.7	14.1 ± 1.8
Acceptance ratio	7.3	11.2

Table 4.4 shows results on two example systems: an Ising model, $q = 2$, and a $q = 10$ Potts model in a difficult parameter regime. We tested nested sampling and AIS with Gibbs sampling and cluster-based updates. Annealed importance sampling (AIS) was run 100 times, with a geometric spacing of 10^4 settings of J as the annealing schedule. Nested sampling used $N = 100$ particles and 100 full-system MCMC updates to approximate each draw from $\breve{\pi}$. We also developed an acceptance ratio method (Bennett, 1976) based on our representation in equation (4.57), which we ran extensively and should give nearly correct results.

The Markov chains used by nested sampling were initialized at one of the $N-1$ particles satisfying the current constraint. Preliminary experiments that initialized a new particle $\theta^{(s)}$ at $\theta^{(s-1)}$ on the constraint surface, were a failure: the Gibbs nested sampler could get stuck permanently in a local maximum of the likelihood, while the cluster method gave erroneous answers for the Ising system. This supports the suggestions of subsection 4.4.4.

AIS performed very well on the Ising system and can work with $q = 10$ at low coupling strengths. We took advantage of its performance in easy parameter regimes to compute $\mathcal{Z}_\pi$, which was needed to interpret the results from the cluster-based nested sampler. However, with a "temperature-based" annealing schedule, AIS was unable to give useful answers for the $q = 10$ system close to the critical J evaluated. Nested sampling appears to be correct within its error bars under these conditions.

It is known that even the efficient Swendsen–Wang algorithm mixes slowly for Potts models with $q > 4$ near critical values of J which correspond to a first order phase transition (Gore and Jerrum, 1997, 1999), see figure 4.6. Typical Potts model states are either entirely disordered or ordered; disordered states contain a jumble of small regions with different colors (e.g. figure 4.6b), in ordered states the system is predominantly one color (e.g. figure 4.6d). Moving between these two phases is difficult; defining a valid MCMC method that moves between distinct phases requires knowledge of the relative probability of the whole collections of states in those phases.

Temperature-based annealing algorithms explore the model for a range of settings of J and fail to capture the correct behavior near the transition. Despite using closely related Markov chains to those used in AIS, nested sampling can work in all parameter

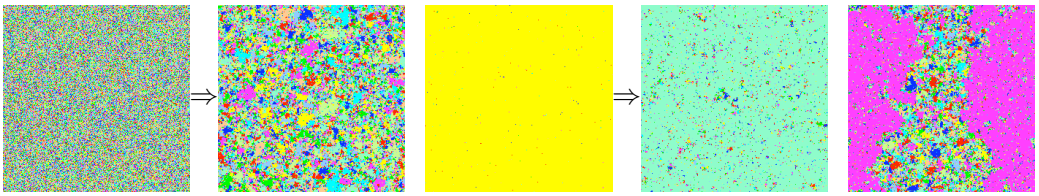


Figure 4.6: Two 256×256, $q = 10$ Potts models with starting states (a) and (c) were simulated with 5×10^6 full-system Swendsen–Wang updates with $J = 1.42577$. The corresponding results, (b) and (d) are typical of all the intermediate samples: Swendsen–Wang is unable to take (a) into an ordered phase, or (c) into a disordered phase, although both phases are typical at this J. (e) in contrast shows an intermediate state of nested sampling, which succeeds in bridging the phases.

regimes. Figure 4.6e shows how nested sampling can explore a mixture of ordered and disordered phases. By moving steadily through these states, nested sampling is able to estimate the prior mass associated with each likelihood value. This behavior is not possible in algorithms that use J as a control parameter, such as AIS with a temperature-based schedule.

4.9 Discussion and conclusions

4.9.1 Summary

The main purpose of this chapter was to study nested sampling and its relationship with more established methods. We find that it fits into a unique position amongst Monte Carlo algorithms. Unlike the majority of annealing-based methods, nested sampling can deal with first order phase transitions. While multicanonical sampling also solves this problem its properties are different in almost every other respect: nested sampling does not need prior setting of weights, its scaling with dimensionality and energy ranges are very different and nested sampling follows a sequence of distributions like annealing. In some statistical settings nested sampling has clear advantages over multicanonical. Undoubtedly it will perform badly on some applications where multicanonical has already been successful.

Nested sampling's theoretical scaling with dimensionality, $\mathcal{O}(D^2)$ iterations, is worse than that of annealing, $\mathcal{O}(D)$ iterations. In practice the difference can be less dramatic as bringing a $\breve{\pi}$ ensemble to equilibrium may require fewer function evaluations than a temperature based distribution. The slice-sampling results do support AIS's superiority, at least for higher accuracy results. But other issues such as ease of implementation and quality of error bars may be more significant. Issues with annealing's error bars were blamed for an order of magnitude extra cost by both Beltrán *et al.* (2005) and Shaw *et al.* (2007).

Despite this the relative under-performance of annealing reported by these astronomers is difficult to account for given the theoretical and practical results of this chapter. The most likely explanation is a difference in the operators used to update the intermediate distributions. In this chapter the same or closely related operators where used with AIS and nested sampling. In Shaw *et al.* (2007) nested sampling was described with its own special update rule using ellipsoids fitted to the current particles. This algorithm could easily require fewer function evaluations than a simple slice sampler or Metropolis method as used within annealing in Beltrán *et al.* (2005). Although the ellipsoid algorithm was suggested by the need to sample from nested sampling's $\breve{\pi}$, there is no reason not to use the same basic code to propose moves for an annealing method. Combined with an annealing schedule specified by nested sampling AIS might perform as well as or better than nested sampling.

4.9.2 Related work

This chapter only considered a subset of the available methods for computing normalizing constants. The focus has been on simple methods that compute the normalizing constant of a generic probability distribution. We avoided assuming detailed knowledge about the target distribution and its relationships with other distributions as much as possible. For users with more time, a richer set of methods are available, some of which are mentioned here.

The intermediate distributions in annealed importance sampling do not have to be temperature-based distributions. That has been the focus here because of the simplicity and generality of raising the likelihood to a power. Any other way of bringing in the likelihood gradually can be used: "one data point at a time" may be natural in a statistical setting. A generalization, linked importance sampling (Neal, 2005) may be helpful for such cases where the intermediate levels are fixed and limited in number.

Annealed importance sampling can be seen as a member of a wider family of "Sequential Monte Carlo" (SMC) methods. Some of these algorithms allow transfer of particles amongst modes, and should definitely be considered by anyone attracted to nested sampling by this property. A recent example of SMC combined with an interesting set of bridging distributions leveraging the power of graphical models is presented in Hamze and de Freitas (2006).

Some situations require the computation of more than one normalizing constant. An option is to construct a distribution containing each of the models and explore it with MCMC. Reversible jump MCMC (Green, 1995) makes this possible even when the models have parameter vectors of different dimensionalities. The relative probability of each model is available from the amount of time the sampler spent exploring each model, and more advanced estimators based on transition probabilities may be available. This

only provides the normalizing constants up to a constant, although if one of the models considered is tractable this constant could be found.

The path sampling approach of Gelman and Meng (1998) suggests computing normalizing constants by integrating along a path of model hyperparameters rather than along a single inverse-temperature parameter. If the normalizer for each setting of the hyperparameters is required this is particularly effective. Contour plots based on independent estimates will usually be very noisy compared to a path sampling approach.

4.9.3 Philosophy

Various authors have noted the irony of using Monte Carlo, a frequentist procedure, for Bayesian computation (e.g. O'Hagan, 1987; Neal, 1993; Rasmussen and Ghahramani, 2003). Rather than giving beliefs about quantities given the computations performed, Monte Carlo algorithms provide "frequentist" statistical estimators. Skilling claims that nested sampling is Bayesian. This section examines the extent to which this holds.

Our target is a posterior distribution over $\mathcal{Z}$. Philosophically this is slightly tricky, because $\mathcal{Z}$ is a constant which we should be able to work out given the prior and likelihood functions. Thus according to any rational calculus of beliefs as in Cox (1946) or Jaynes (2003) the posterior should concentrate all of its mass at the true value. The problem is not lack of available information, but computer time to use it perfectly.

The solution to this conundrum is to be careful about what we claim to know. Assume that some agent is running nested sampling on our behalf and only reports to us $\mathbf{L} = \{L^{(s)}\}$. If it reported more information, such as $\theta^{(s)}$ locations, we would be in an embarrassing situation. To use this information we would need a probabilistic model including $\theta^{(s)}$ in which inference would probably be hard. But throwing away knowledge is hard to deal with rationally, so we pretend it was never available. Thus the inferences for nested sampling, as in Rasmussen and Ghahramani (2003), are rational for a fictitious observer with limited information. This observer's results will be more vague than if $\{\theta^{(s)}\}$ were not ignored but should be sensible, which is not guaranteed by general approximations to Bayesian inference.

We first set up priors based on the knowledge that an agent is running nested sampling, which will provide $L^{(s)}$ quantities with associated $x^{(s)}$ cumulative mass values. Our knowledge of nested sampling defines a problem-independent prior distribution over $\mathbf{x}=\{x^{(s)}\}$. We should also specify a prior distribution over functions $L(x)$.

After observing $\mathbf{L} = \{L^{(s)}\}$, we update our beliefs about the underlying cumulative values according to Bayes' rule: $P(\mathbf{x}|\mathbf{L}) = P(\mathbf{L}|\mathbf{x})\,P(\mathbf{x})/P(\mathbf{L})$. It is this posterior distribution that should be used when computing posteriors over other quantities such

as $\mathcal{Z}$,

$$P(\mathcal{Z}|\mathbf{L}) = \int P(\mathcal{Z}|\mathbf{x}, \mathbf{L})\, P(\mathbf{x}|\mathbf{L})\, \mathrm{d}\mathbf{x}. \tag{4.62}$$

Note that specifying a prior over monotonic functions $P(L(x))$ and computing with it appears difficult in general. Skilling declares that "I can't, so I don't". Instead the prior $P(\mathbf{x})$ is used. This corresponds to a particular assumption: an improper uniform prior over likelihood functions. This cannot be avoided by claiming general ignorance: unlike $\{\theta^{(s)}\}$, we must be told $\{L^{(s)}\}$. Thus to maintain a claim of rationality, we must be happy with this particular choice of prior or type of ignorance about the likelihood function.

In addition, and much more seriously, we must assume the agent is actually running nested sampling as in algorithm 4.2. In fact we really know that some approximation will be involved. As we have seen this can give us unreasonable beliefs—as with most probabilistic modeling where we know *a priori* that a model's joint distribution does not really capture every detail of a real system.

Similar concerns must surround all such attempts to introduce Bayesian methodology into Monte Carlo algorithms for general inference problems. For example Bayesian learning of the one-dimensional weighting function of the multicanonical method has been attempted (e.g. Smith, 1995). This should be a difficult inference problem as the output of a sampler has a complicated dependence structure. Only with incorrect assumptions, i.e., approximations, can Bayesian methods be applied. The ultimate justification for such methods must be their empirical performance, which is sometimes very good.

Chapter 5

Doubly-intractable distributions

Most of this thesis has been dedicated to sampling from probability distributions where the key difficulty has been an intractable normalization. When considering a posterior over parameters θ given data y,

$$p(\theta|y) = \frac{p(y,\theta)}{p(y)} = \frac{p(y|\theta)\,p(\theta)}{p(y)}, \tag{5.1}$$

we assumed that the joint probability in the numerator could be easily evaluated for any particular joint setting (y,θ). Chapter 4 described how the important quantity $p(y)$ can be approximated, but it is often infeasible to compute this quantity exactly.

Standard MCMC methods are designed for use with these intractable distributions. Markov chain operators can be constructed by restricting consideration to a manageable subset of θ's state space at each step. In Metropolis–Hastings only two settings are considered, the current setting θ and a randomly chosen proposal, θ'. Gibbs sampling changes only one component of θ at a time. Metropolis requires an ability to evaluate $p(y,\theta)$ ratios for various θ pairs, and Gibbs sampling requires the ability to sample from the conditional distributions $p(\theta_i|\theta_{\{j\neq i\}}, y)$. By considering restricted parts of the state space, neither method needs to know the global normalizing constant $p(y)$.

But what if $p(y,\theta)$, like $p(y)$, contains a summation over a large state space so it cannot feasibly be evaluated point-wise? Then the problem is *doubly-intractable* and, as we shall see, even performing Markov chain Monte Carlo is potentially exceedingly difficult.

The next section explains why doubly-intractable distributions arise and the difficulties involved. Section 5.2 explores approximations of standard MCMC algorithms in the context of undirected graphical models. Møller *et al.* (2004, 2006) provided the first feasible algorithm for models where sampling from $p(y|\theta)$ is possible but its normalization is unknown. Their method is reviewed in section 5.4 and generalized by us in section 5.5. Working on these algorithms inspired the new *exchange algorithm*, sec-

tion 5.3. These innovations were first described in (Murray *et al.*, 2006a). This chapter provides a slightly more general version of the exchange algorithm with a new derivation which is somewhat simpler than the original description. Full mathematical derivations of detailed balance are provided, which were previously omitted for space reasons. In section 5.7 we consider further new valid MCMC algorithms for doubly-intractable distributions, which provide a connection to the Approximate Bayesian Computation (ABC) literature. Section 5.8 provides slice sampling algorithms for doubly-intractable distributions. Finally new directions for research in this area are considered in section 5.9.

5.1 Bayesian learning of undirected models

The Potts model discussed in the previous chapter belongs to a very wide class of "energy-based models" where

$$\begin{aligned} p(y|\theta) &= \frac{1}{\mathcal{Z}(\theta)} \exp\left(-\sum_j E_j(y_{c_j}, \theta_j)\right) \\ &= \frac{1}{\mathcal{Z}(\theta)} \prod_j f_j(y_{c_j}, \theta_j), \qquad \mathcal{Z}(\theta) = \sum_y \prod_j f_j(y_{c_j}, \theta_j). \end{aligned} \tag{5.2}$$

The sets c_j each index a subset of variables that take part in a corresponding potential function f_j, parameterized by θ_j. Each potential expresses mutual compatibilities amongst a subset of variables y_{c_j}. Sampling from the y variables is possible with MCMC, but computing the normalization can be very difficult.

Section 1.1 reviewed how special structure in a graphical model sometimes allows efficient computation of the normalization $\mathcal{Z}(\theta)$. Most of this chapter concerns methods that apply to general distributions, so it is convenient for clarity to collapse the model to a simpler form:

$$p(y|\theta) = f(y;\theta)/\mathcal{Z}(\theta). \tag{5.3}$$

We now consider sampling from the posterior over parameters, equation (5.1), when the likelihood is of the unnormalized form given in equation (5.3). This posterior,

$$p(\theta|y) = \left(\frac{f(y;\theta)p(\theta)}{\mathcal{Z}(\theta)}\right) \Big/ p(y), \tag{5.4}$$

offers a new difficulty. As before $p(y)$ is not needed for MCMC, but the normalizing 'constant' $\mathcal{Z}(\theta)$ cannot be ignored as it is a function of the parameters θ, the variables being sampled. Every time new parameter values are considered it appears that an intractable computation involving $\mathcal{Z}$ will be required. As MCMC estimators are approximations unless an infinite number of iterations are performed, and each itera-

tion is generally infeasible, $p(\theta|y)$ in equation (5.4) can be called a *doubly-intractable* distribution.

While sampling from parameter posteriors by MCMC is a well established technique, it is largely associated with distributions that could be represented as directed graphical models. However, sampling parameters in anything but the most trivial[1] *undirected* graphical model is doubly-intractable. While directed models are a more natural tool for modeling causal relationships, the soft constraints provided by undirected models have proven useful in a variety of problem domains; we briefly mention six applications.

(a) In computer vision Markov random fields (MRFs), a form of undirected model, are used to model the soft constraint a pixel or image feature imposes on nearby pixels or features (Geman and Geman, 1984); this use of MRFs grew out of a long tradition in spatial statistics (Besag, 1974). **(b)** In language modeling a common form of sentence model measures a large number of features of a sentence $f_j(s)$, such as the presence of a word, subject-verb agreement, the output of a parser on the sentence, etc, and assigns each such feature a weight λ_j. A random field model of this is then $p(s|\lambda) = (1/\mathcal{Z}(\lambda)) \exp\{\sum_j \lambda_j f_j(s)\}$ where the weights can be learned via maximum likelihood iterative scaling methods (Della Pietra *et al.*, 1997). **(c)** These undirected models can be extended to coreference analysis, which deals with determining, for example, whether two items (e.g., strings, citations) refer to the same underlying object (McCallum and Wellner, 2003). **(d)** Undirected models have been used to model protein folding (Winther and Krogh, 2004) and the soft constraints on the configuration of protein side chains (Yanover and Weiss, 2003). **(e)** Semi-supervised classification is the problem of classifying a large number of unlabeled points using a small number of labeled points and some prior knowledge that nearby points have the same label. This problem can be approached by defining an undirected graphical model over both labeled and unlabeled data (Zhu and Ghahramani, 2002). **(f)** Given a set of directed models $p(y|\theta_j)$, the *products of experts* idea is a simple way of defining a more powerful (undirected) model by multiplying them: $p(y|\theta) = (1/\mathcal{Z}(\theta)) \prod_j p(y|\theta_j)$ (Hinton, 2002). The product assigns high probability when there is consensus among component models.

Despite the long history and wide applicability of undirected models, until recently Bayesian treatments of learning the parameters of large undirected models have been virtually non-existent! Indeed there *is* a related statistical literature on Bayesian inference in undirected models, log linear models, and contingency tables (Albert, 1995; Dellaportas and Forster, 1999; Dobra *et al.*, 2006). However, this literature, with the notable exception of the technique reviewed in section 5.4, assumes that the partition function $\mathcal{Z}(\theta)$ can be computed exactly. But for many of the machine learning applications of undirected models cited above, this assumption is unreasonable. This chapter addresses Bayesian learning for models with intractable $\mathcal{Z}(\theta)$.

[1] i.e., low tree-width graphs, graphical Gaussian models and small contingency tables.

5.1.1 Do we need $\mathcal{Z}(\theta)$ for MCMC?

The modeler's effort was put into specifying $f(y;\theta)$; it is tempting to think that $\mathcal{Z}(\theta)$ should have little relevance and there must be some way to side-step computing it. It was established above that the normalizer $\mathcal{Z}(\theta)$ is not a constant in the context of sampling θ. This section explores the rôle of $\mathcal{Z}(\theta)$ in more detail, addressing the consequences for any scheme that avoids computing it.

Algorithm 5.1 gives the straightforward application of standard Metropolis–Hastings (algorithm 2.1) to the doubly-intractable distribution in equation (5.4).

Algorithm 5.1 Standard (but infeasible) Metropolis–Hastings (M–H)

Input: initial setting θ, number of iterations S

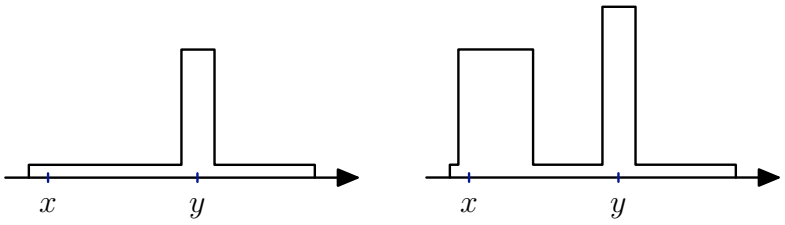
1. **for** $s = 1 \dots S$
2. Propose $\theta' \sim q(\theta' \leftarrow \theta; y)$
3. Compute

$$a = \frac{p(\theta'|y)}{p(\theta|y)} \frac{q(\theta \leftarrow \theta'; y)}{q(\theta' \leftarrow \theta; y)} = \frac{f(y;\theta')\, p(\theta')\, q(\theta \leftarrow \theta'; y)}{f(y;\theta)\, p(\theta)\, q(\theta' \leftarrow \theta; y)} \cdot \frac{\mathcal{Z}(\theta)}{\mathcal{Z}(\theta')}$$

4. Draw $r \sim \text{Uniform}[0, 1]$
5. **if** $(r < a)$ **then** set $\theta \leftarrow \theta'$
6. **end for**

Computing the acceptance ratio requires a ratio of normalizing constants, or at least a bound tight enough for step 5. This is difficult in general. There are usually some free choices while constructing MCMC algorithms. Perhaps some "nuisance" parameters could be judiciously set to remove $\mathcal{Z}(\theta)$ through some fortunate cancellations? We are not free to cancel out $\mathcal{Z}$ through a choice of prior $p(\theta)$: the prior would have to depend on the number of observed data points and would take on extreme values dominating any inferences (Murray and Ghahramani, 2004). In theory the proposal distribution could be defined to remove explicit $\mathcal{Z}$ dependence from the acceptance ratio, but in practice this does not seem to help: e.g. $q(\theta' \leftarrow \theta; y) = \mathcal{Z}(\theta) g(\theta')$ or $q(\theta' \leftarrow \theta; y) \propto 1/\mathcal{Z}(\theta')$ would be difficult to construct without knowing $\mathcal{Z}$, and would be terrible proposals. The only distribution we know of that contains $\mathcal{Z}(\theta)$ gives probabilities in data space, not over parameters.

Section 5.4 reviews and extends a method by Møller *et al.* (2004, 2006) which introduced auxiliary variables taking on values in the data space. This allows proposals that cancel out the unknown terms, but only if it is possible to draw from the (intractable) distribution $p(y|\theta)$. This key insight makes it possible to sample from a limited but significant class of distributions for which MCMC was previously impossible. However, the algorithm and its extensions are not a panacea. It is not always possible to draw exact samples from the data distribution. Another problem (or opportunity) is



(a) $f(\cdot;\theta)$ (b) $f(\cdot;\theta')$

Figure 5.1: (a) The dash marked y shows the position of our observation in data space. The curve shows an unnormalized function $f(\cdot;\theta)$, which gives low probability to the alternative data set x. (b) After changing the parameter $\theta \rightarrow \theta'$ the unnormalized function evaluated at the observation has increased, $f(y;\theta') > f(y;\theta)$. However the likelihood has decreased, $p(y|\theta') < p(y|\theta)$. Noticing this requires considering the new high-probability region of data space containing x, which is not necessarily close to the observation.

that specifying a workable stationary distribution for the auxiliary variables requires approximating the parameter posterior before sampling begins.

An optimistic train of thought sees no problem as $\mathcal{Z}$ is "just a sum" and summing over unknowns is a standard feature of MCMC. For example models with latent variables, z, often have intractable likelihoods. But while it may be difficult to evaluate

$$p(y|\theta) = \sum_z p(y, z|\theta) = \sum_z p(y|z, \theta)p(z|\theta), \tag{5.5}$$

we can jointly sample over $p(\theta, z|y)$ as long as this distribution is known up to a constant. Discarding the latents gives samples from the correct marginal $p(\theta|y)$. In doubly-intractable problems the likelihood contains $1/\mathcal{Z}(\theta)$, the reciprocal of a sum of terms, rather than a sum over latent variables. But could there be some similar way of instantiating latent variables to remove the need to compute $\mathcal{Z}$? Section 5.7 provides such a method with an infinite number of latent variables. But how do we know there is not a better choice; in general must all simple algorithms fail?

Figure 5.1 shows a hypothetical unnormalized probability function for two settings of its parameters. An obvious but important observation is that the change in an unnormalized probability function evaluated at an observation does not necessarily tell us anything about the change in the likelihood of the parameters. In fact we must consider the parameters' effect on the entire observation space. This is in sharp contrast to MCMC sampling of the y variables for fixed parameters where only two settings need be considered at a time. If we are not going to compute $\mathcal{Z}(\theta)$ explicitly then a valid MCMC algorithm must have some other source of global information that is sensitive to changes anywhere in observation space.

One of the simplest ways to get information from a probability distribution is, of course, through a sample. A sample $x \sim f(x;\theta')/\mathcal{Z}(\theta')$ using the same function as figure 5.1b could easily land in the new region of high probability on the left. Noticing that

$f(x;\theta) \ll f(x;\theta')$ gives some indication that any perceived benefit of $f(y;\theta') > f(y;\theta)$ should be penalized. Despite the apparent paucity of such information, surprisingly these single samples at each parameter setting are sufficient to create valid MCMC algorithms for $p(\theta|y)$ that do not need $\mathcal{Z}(\theta)$. This is how the approach in Møller *et al.* (2004) works, which explains why it requires samples from the target distribution using an exact or perfect sampling method (section 2.7). Approximate samples from a few MCMC steps cannot guarantee considering the new bubble in data space in figure 5.1b. A Markov chain started at (e.g.) the observation will be heavily biased towards a mode near the starting point and may never consider any other modes.

Are spurious modes in data space a problem in practice? Contrastive divergence learning (Hinton, 2002) uses very brief MCMC sampling starting at the observed data and can give useful results on complex problems in machine learning. Sometimes we know the parameter posterior is simple, it is log-concave for fully-observed exponential-family distributions. In such cases deterministic approximations to the parameter posterior perform favorably compared to pseudo-MCMC approaches (Welling and Parise, 2006). However, if we wish to construct a valid MCMC method, we must formally show that the entire data space has been properly considered. Exact sampling explicitly tracks bounds that start off considering the entire observation space (section 2.7), a procedure with this flavor seems an essential part of samplers for doubly-intractable distributions. Thus even though a new latent variable approach introduced in section 5.7 does not use exact sampling as such, the requirements are, and must be, very similar.

Exact sampling is a formidable requirement; deterministic approaches and MCMC methods that do not use the correct posterior distribution will always have a place. Some possibilities are studied in the next section. There is also always a place for "gold standard" methods that given enough time should give correct answers. Those are the focus of the remainder of the chapter.

5.2 Approximation Schemes

For concreteness this section studies a simple but widespread type of graphical model. The Boltzmann machine (BM) is a Markov random field which defines a probability distribution over a vector of binary variables $\mathbf{s} = [s_1, \ldots, s_k]$ where $s_i \in \{0, 1\}$:

$$p(\mathbf{s}|W) = \frac{1}{\mathcal{Z}(W)} \exp\left(\sum_{i<j} W_{ij} s_i s_j,\right). \tag{5.6}$$

The symmetric weight matrix W parameterizes this distribution. In a BM there are usually also linear bias terms $\sum_i b_i s_i$ in the exponent, with these the model is equivalent to a Potts or Ising model with "magnetic field" parameters. We omit these biases to simplify notation, although the models in the experiments assume them.

The usual algorithm for learning BMs is a maximum likelihood version of the EM algorithm (assuming some of the variables are hidden $\mathbf{s}_H$ and some observed $\mathbf{s}_O$) (Ackley *et al.*, 1985). The gradient of the log probability is:

$$\frac{\partial \log p(\mathbf{s}_O|W)}{\partial W_{ij}} = \mathbb{E}_c[s_i s_j] - \mathbb{E}_u[s_i s_j]\,, \tag{5.7}$$

where $\mathbb{E}_c[\cdot]$ denotes expectation under the "clamped" data distribution $p(\mathbf{s}_H|\mathbf{s}_O, W)$ and $\mathbb{E}_u[\cdot]$ denotes expectation under the "unclamped" distribution $p(\mathbf{s}|W)$. For a data set $S = [\mathbf{s}^{(1)} \ldots \mathbf{s}^{(n)} \ldots \mathbf{s}^{(N)}]$ of i.i.d. data the gradient of the log likelihood is simply summed over n. For Boltzmann machines with large tree-width (see section 1.1) these expectations would take exponential time to compute, and the usual approach is to approximate them using Gibbs sampling or one of many more recent approximate inference algorithms.

5.2.1 Targets for MCMC approximation

Metropolis–Hastings for the parameters of a Boltzmann machine given fully observed data needs (to bound)

$$\begin{aligned} a &= \frac{p(S|W')p(W')q(W \leftarrow W'; S)}{p(S|W)p(W)q(W' \leftarrow W; S)} \\ &= \frac{p(W')\, q(W \leftarrow W'; S)}{p(W)\, q(W' \leftarrow W; S)} \left(\frac{\mathcal{Z}(W)}{\mathcal{Z}(W')}\right)^N \exp\left(\sum_{n,i<j} (W'_{ij} - W_{ij})\; s_i^{(n)} s_j^{(n)}\right). \end{aligned} \tag{5.8}$$

The first class of approximation we will pursue is to substitute a deterministic approximation $\mathcal{Z}(W) \simeq \tilde{\mathcal{Z}}(W)$ into the above expression. Clearly this results in an *approximate* sampler, which does not converge to the true equilibrium distribution over parameters. Moreover, it seems reckless to take an approximate quantity to the N^{th} power. Despite these caveats we explore empirically whether approaches based on this class of approximation are viable.

Note that above we need only compute the ratio of the partition function at pairs of parameter settings, $\mathcal{Z}(W)/\mathcal{Z}(W')$. This ratio can be approximated directly by importance sampling:

$$\frac{\mathcal{Z}(W)}{\mathcal{Z}(W')} \equiv \mathbb{E}_{p(\mathbf{s}|W')}\left[\exp\left(\sum_{i<j}(W_{ij} - W'_{ij}) s_i s_j\right)\right]. \tag{5.9}$$

Thus any method for estimating expectations under $p(\mathbf{s}|W')$ — sampling-based or deterministic — can be nested into the Metropolis sampler for W.

For small steps $W \to W'$ estimating ratios of normalizers and finding gradients with respect to the parameters are closely related problems. Gradients may be more useful

as they provide a direction in which to move, which is useful in algorithms based on dynamical systems such as Hybrid Monte Carlo (subsection 2.4.3). Hybrid Monte Carlo would also require a $\mathcal{Z}$ ratio for its accept/reject step. This effort may not be justified when the gradients and $\mathcal{Z}(W)$ are only available as approximations. Simpler schemes that use gradient information also exist (Neal, 1993). The simplest of these is the "uncorrected Langevin method". Parameters are updated without any rejections according to the rule:

$$\theta_i' = \theta_i + \frac{\epsilon^2}{2}\frac{\partial}{\partial\theta_i}\log p(y,\theta) + \epsilon n_i, \tag{5.10}$$

where n_i are independent draws from a zero-mean unit variance Gaussian. Intuitively this rule performs gradient descent but explores away from the optimum through the noise term. Strictly this is only an approximation except in the limit of vanishing ϵ.

Using the above or other dynamical methods, a third target for approximation for systems with continuous parameters is the gradient of the joint log probability. In the case of BMs, we have:

$$\frac{\partial \log p(S,W)}{\partial W_{ij}} = \sum_n s_i^{(n)} s_j^{(n)} - N\mathbb{E}_{p(\mathbf{s}|W)}[s_i s_j] + \frac{\partial \log p(W)}{\partial W_{ij}}. \tag{5.11}$$

Assuming an easy to differentiate prior, the main difficulty arises, as in equation (5.7), from computing the middle term: the unclamped expectations over the variables.

Interestingly, although many learning algorithms for undirected models, e.g. the original Boltzmann machine learning rule, are based on computing gradients such as equation (5.11), and it would be simple to plug these into approximate stochastic dynamics MCMC methods to do Bayesian inference, this approach does not appear to have been investigated. We explore this approach in our experiments.

This section has considered two existing sampling schemes (Metropolis and Langevin) and identified three targets for approximation to make these schemes tractable: $\mathcal{Z}(W)$, $\mathcal{Z}(W)/\mathcal{Z}(W')$ and $\mathbb{E}_{p(\mathbf{s}|W)}[s_i s_j]$. While the explicit derivations above focused on Boltzmann machines, these same expressions generalize in a straightforward way to Bayesian parameter inference in a general undirected model as in equation (5.2). In particular, many undirected models of interest can be parameterized to have potentials in the exponential family, $-E_j(y_{c_j}, \theta_j) = \mathbf{u}_j(y_{c_j})^\top \theta_j$. For such models, the key ingredients to an approximation are the expected sufficient statistics, $\mathbb{E}_{p(y|\theta)}\big[\mathbf{u}_j(y_{c_j})\big]$.

5.2.2 Approximation algorithms

In this section approximations for each of the three target quantities in equations (5.8), (5.9) and (5.11) are identified. These are used to propose a variety of approximate sampling methods for doubly-intractable distributions, first outlined in Murray and Ghahramani (2004).

Variational lower bounds were developed in statistical physics, but are also widely used in machine learning. They use Jensen's inequality to lower bound the log partition function in the following way:

$$\begin{aligned}\log \mathcal{Z}(\theta) &= \log \sum_x f(x;\theta) = \log \sum_x q(x) \frac{f(x;\theta)}{q(x)} \\ &\geq \sum_x q(x) \log \frac{f(x;\theta)}{q(x)} \\ \log \mathcal{Z}(\theta) &\geq \sum_x q(x) \log f(x;\theta) + \mathcal{H}(q) \equiv \mathcal{F}(\theta, q).\end{aligned} \tag{5.12}$$

The relationship holds for any distribution $q(x)$, provided it is not zero where $p(x;\theta)$ has support. The second term in the bound is called the entropy of the distribution $\mathcal{H}(q) \equiv -\sum_x q(x) \log q(x)$. The over-all bound $\mathcal{F}$ is often called the free energy. See Winn and Bishop (2005) for more details and a framework for automatic construction and optimization of variational bounds for a large class of graphical models.

The naïve mean field method is a variational method with q constrained to belong to the set of fully factorized distributions $\mathcal{Q}_{\text{mf}} = \{q : q(x) = \prod_i q_i(x_i)\}$. For the Boltzmann machine a local maximum of this lower bound $\log \mathcal{Z}_{\text{mf}}(\theta) = \max_{q \in \mathcal{Q}_{\text{mf}}} \mathcal{F}(\theta, q)$ can be found with an iterative and tractable fixed-point algorithm, see for example MacKay (2003, chapter 33). Let the **mean-field Metropolis** algorithm be defined by using $\mathcal{Z}_{\text{mf}}(\theta)$ in place of $\mathcal{Z}(\theta)$ in the acceptance probability computation, equation (5.8). The expectations from the naïve mean field algorithm could also be used to compute direct approximations to the gradients in equation (5.11) for use in a stochastic dynamics method.

Jensen's inequality can be used to obtain much tighter bounds than those given by the naïve mean-field method. For example, when constraining q to be in the set of all tree-structured distributions $\mathcal{Q}_{\text{tree}}$ optimizing the lower bound on the partition function is still tractable (Wiegerinck, 2000), obtaining $\mathcal{Z}_{\text{tree}}(\theta) \leq \mathcal{Z}(\theta)$. The **tree Metropolis** algorithm is defined through the use of this approximation in equation (5.8). Alternatively, expectations under the tree could be used to form the gradient estimate for a stochastic dynamics method, equation (5.11).

Bethe approximation. A recent justification for applying belief propagation to graphs with cycles is the relationship between this algorithm's messages and the fixed points of the Bethe free energy (Yedidia *et al.*, 2005). This breakthrough gave a new approximation for the partition function. In the **loopy Metropolis** algorithm belief propagation is run on each proposed system, and the Bethe free energy is used to approximate the acceptance probability, equation (5.8). Traditionally belief propagation is used to compute marginals; pairwise marginals can be used to compute the expectations used in gradient methods, e.g. equation (5.11), or in finding partition function ratios, equation (5.9). These approaches lead to different algorithms, although their

approximations are clearly closely related.

Langevin using brief sampling. The pairwise marginals required in equations (5.10) and (5.11) can be approximated by MCMC sampling. The Gibbs sampler used in subsection 5.2.4 is a popular choice, whereas in subsection 5.2.5 a more sophisticated Swendsen-Wang sampler is employed. Unfortunately — as in maximum likelihood learning, equation (5.7) — the parameter-dependent variance of these estimates can hinder convergence and introduce biases. The **brief Langevin** algorithm, inspired by work on *contrastive divergence* (Hinton, 2002), uses very brief sampling starting from the data, X, which gives biased but low variance estimates of the required expectations. As the approximations in this section are run as an inner loop to the main sampler, the cheapness of brief sampling makes it an attractive option.

Langevin using exact sampling. Unbiased expectations can be obtained in some systems using an exact sampling algorithm (section 2.7). Although the gradients from this method are guaranteed to be unbiased, parameter-dependent variance could lead to worse performance than the proposed brief Langevin method. Variance could be reduced by reusing pseudo random numbers. However, we shall see later that there are much more elegant ways to use an exact sampler if one is available.

Pseudo-Likelihood. Replacing the likelihood of the parameters with a tractable product of conditional probabilities is a common approximation in Markov random fields for image modeling. One of the earliest Bayesian approaches to learning in large systems of which we are aware was in this context (Wang *et al.*, 2000; Yu and Cheng, 2003). The models used in the experiments of subsection 5.2.4 were not well approximated by the pseudo-likelihood, so it is not explored further here.

5.2.3 Extension to hidden variables

So far we have only considered models of the form $p(y|\theta)$ where all variables, y, are observed. Often models need to cope with missing data, or have variables that are always hidden. These are often the models that would most benefit from a Bayesian approach to learning the parameters. In fully observed models in the exponential family the parameter posteriors are often relatively simple as they are log concave if the prior used is also log concave (as seen later in figure 5.2). The parameter posterior with hidden variables will be a linear combination of log concave functions, which need not be log concave and can be multi-modal.

In theory the extension to hidden variables is simple. First consider a model $p(y, h|\theta)$, where h are unobserved variables. The parameter posterior is still proportional to

$p(y|\theta)p(\theta)$, and we observe

$$\begin{aligned} p(y|\theta) &= \sum_h p(y,h|\theta) = \frac{1}{\mathcal{Z}(\theta)} \sum_h \prod_j f_j((y,h)_{c_j}, \theta_j) \\ \log p(y|\theta) &= -\log \mathcal{Z}(\theta) + \log \sum_h \prod_j f_j((y,h)_{c_j}, \theta_j) \\ &= -\log \mathcal{Z}(\theta) + \log \mathcal{Z}_y(\theta). \end{aligned} \tag{5.13}$$

That is, the sum in the second term is a partition function, $\mathcal{Z}_y$, for an undirected graph of the variables h. To see this compare to equation (5.2) and consider the fixed observations y as parameters of the potential functions. In a system with multiple i.i.d. observations $\mathcal{Z}_y$ must be computed for each setting of y. Note however that these additional partition function evaluations are for systems smaller than the original. Therefore, any method that approximates $\mathcal{Z}(\theta)$ or related quantities directly from the parameters can still be used for parameter learning in systems with hidden variables.

The brief sampling and pseudo-likelihood approximations rely on settings of every variable provided by the data. For systems with hidden variables these methods could use settings from samples conditioned on the observed data. In some systems this sampling can be performed easily (Hinton, 2002). In subsection 5.2.5 several steps of MCMC sampling over the hidden variables are performed in order to apply the brief Langevin method.

5.2.4 Experiments involving fully observed models

The approximate samplers described in subsection 5.2.2 were tested on three systems. The first, taken from Edwards and Havránek (1985), lists six binary properties detailing risk factors for coronary heart disease in 1841 men. Modeling these variables as outputs of a fully-connected Boltzmann machine, we attempted to draw samples from the distribution over the unknown weights. We can compute $\mathcal{Z}(\theta)$ exactly in this system, which allows us to compare methods against a Metropolis sampler with an exact inner loop. A previous Bayesian treatment of these data also exists (Dellaportas and Forster, 1999).

While predictions wouldn't need as many samples, we performed sampling for 100,000 iterations to obtain reasonable histograms for each of the weights (figure 5.2). The mean-field, tree and loopy Metropolis methods each proposed changes to one parameter at a time using a zero-mean Gaussian with variance 0.01. The brief Langevin method used a step-size $\epsilon = 0.01$. Qualitatively the results are the same as those reported by Dellaportas and Forster (1999), parameters deemed important by them have very little overlap with zero.

The mean-field Metropolis algorithm failed to converge, producing noisy and wide histograms over an ever increasing range of weights (figure 5.2). The sampler with the

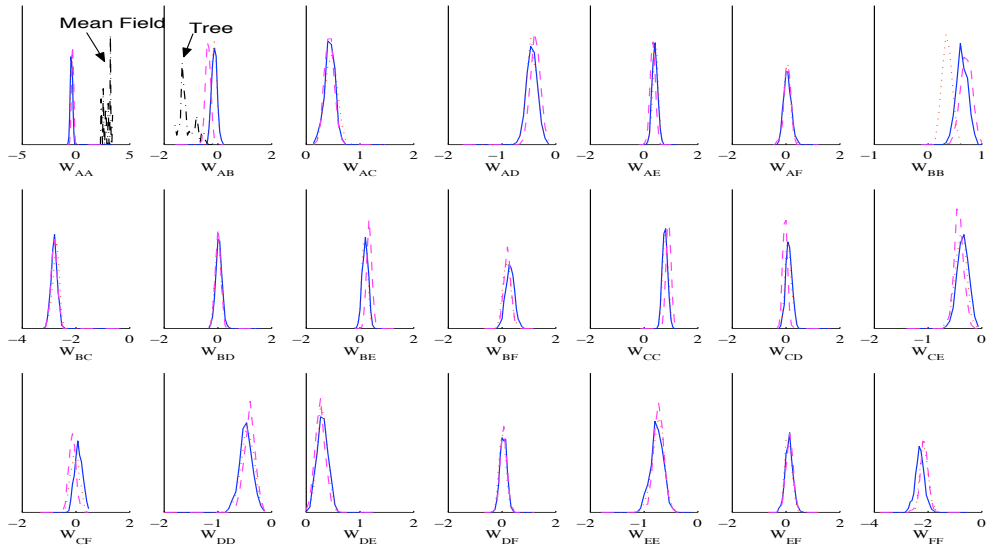


Figure 5.2: Histograms of samples for every parameter in the heart disease risk factor model. Results from exact Metropolis are shown in solid (blue); loopy Metropolis dashed (purple); brief Langevin dotted (red). These curves are often indistinguishable. The mean-field and tree Metropolis algorithms performed very badly; to reduce clutter these are only shown once each in the plots for W_{AA} and W_{AB} respectively, shown in dash-dotted (black).

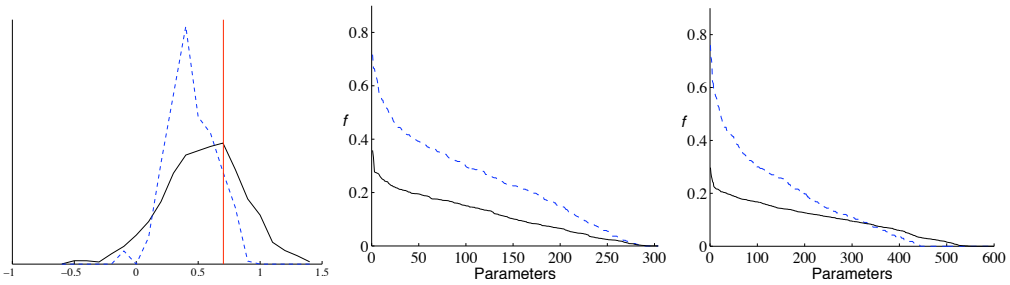


Figure 5.3: Loopy Metropolis is shown dashed (blue), brief Langevin solid (black). Left: an example histogram as in figure 5.2 for the 204 edge BM; the vertical line shows the true weight. Also shown are the fractions of samples, f, within ± 0.1 of the true value for every parameter in the 204 edge system (center) and the 500 edge system (right). The parameters are sorted by f for clarity. Higher curves indicate better performance.

tree-based inner loop did not always converge either and when it did, its samples did not match those of the exact Metropolis algorithm very well. The loopy Metropolis and brief Langevin methods closely match the marginal distributions predicted by the exact Metropolis algorithm for most of the weights. Results are not shown for algorithms using expectations from loopy belief propagation in equation (5.11) or equation (5.9) as these gave almost identical performance to loopy Metropolis based on equation (5.8).

Our other two test systems are 100-node Boltzmann machines and demonstrate learning where exact computation of $\mathcal{Z}(W)$ is intractable[2]. We considered two randomly generated systems, one with 204 edges and another with 500. Each of the parameters not set to zero, including the 100 biases, was drawn from a unit Gaussian. Experiments on an artificial system allow comparisons with the true weight matrix. We ensured our training data were drawn from the correct distribution with an exact sampling method (Childs *et al.* (2001), reviewed in subsection 2.7.1). This level of control would not be available on a natural data set.

The loopy Metropolis algorithm and the brief Langevin method were applied to 100 data points from each system. The model structure was provided, so that only non-zero parameters were learned. Figure 5.3 shows a typical histogram of parameter samples; the predictive ability over all parameters is also shown. Short runs on similar systems with stronger weights show that loopy Metropolis can be made to perform arbitrarily badly more quickly than the brief Langevin method on this class of system.

5.2.5 Experiment involving hidden variables

Finally we consider an undirected model approach taken from work on semi-supervised learning in Zhu and Ghahramani (2002). Here a graph is defined using the 2D positions, $X = \{(x_i, y_i)\}$, of unlabeled and labeled data. The variables on the graph are the class labels, $S = \{s_i\}$, of the points. The joint model for the l labeled points and u unobserved or hidden variables is

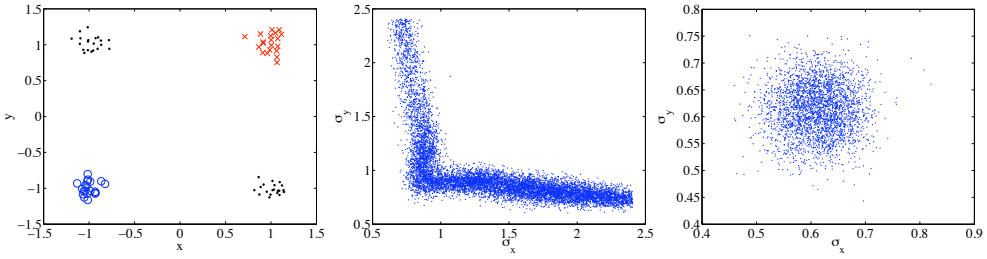
$$p(S|X, \boldsymbol{\sigma}) = \frac{1}{\mathcal{Z}(\boldsymbol{\sigma})} \exp\left(\sum_{i=1}^{l+u} \sum_{i<j} \delta(s_i, s_j) W_{ij}(\boldsymbol{\sigma})\right), \tag{5.14}$$

where

$$W_{ij}(\boldsymbol{\sigma}) = \exp\left(-\frac{1}{2}\left(\frac{(x_i - x_j)^2}{\sigma_x^2} + \frac{(y_i - y_j)^2}{\sigma_y^2}\right)\right). \tag{5.15}$$

The edge weights of the model, W_{ij}, are functions of the Euclidean distance between points i and j measured with respect to scale parameters $\boldsymbol{\sigma} = (\sigma_x, \sigma_y)$. Nearby points wish to be classified in the same way, whereas far away points may be approximately uncorrelated, unless linked by a bridge of points in between.

[2]These test sets are available online: `http://www.gatsby.ucl.ac.uk/~iam23/04blug/`



(a) Data set (b) Brief Langevin (c) Loopy Metropolis

Figure 5.4: (a) a data set for semi-supervised learning with 80 variables: two groups of classified points ($\times$ and $\circ$) and unlabeled data ($\cdot$). (b) 10,000 approximate samples from the posterior of the parameters σ_x and σ_y, equation (5.14). An uncorrected Langevin sampler using gradients with respect to $\log(\boldsymbol{\sigma})$ approximated by a Swendsen-Wang sampler was used. (c) 10,000 approximate samples using Loopy Metropolis.

The likelihoods in this model can be interesting functions of $\boldsymbol{\sigma}$ (Zhu and Ghahramani, 2002), leading to non-Gaussian and possibly multi-modal parameter posteriors with any simple prior. As the likelihood is often a very flat function over some parameter regions, the MAP parameters can change dramatically with small changes in the prior. There is also the possibility that no single settings of the parameters can capture our knowledge.

When performing binary classification equation (5.14), which is a type of Potts model, can be rewritten as a standard Boltzmann Machine. The edge weights W_{ij} are now all coupled through $\boldsymbol{\sigma}$, so our sampler will only explore a two-dimensional parameter space (σ_x, σ_y). However, little of the above theory is changed by this: we can still approximate the partition function and use this in a standard Metropolis scheme, or apply Langevin methods based on equation (5.11) where gradients include sums over edges.

Figure 5.4a shows an example data set for this problem. This toy data set is designed to have an interpretable posterior over $\boldsymbol{\sigma}$ and demonstrates the type of parameter uncertainty observed in real problems. We can see intuitively that we do not want σ_x or σ_y to be close to zero. This would disconnect all points in the graph making the likelihood small ($\approx 1/2^l$). Parameters that correlate nearby points that are the same will be much more probable under a large range of sensible priors. Neither can both σ_x and σ_y be large: this would force the $\times$ and $\circ$ clusters to be close, which is also undesirable. However, one of σ_x and σ_y can be large as long as the other stays below around one. These intuitions are closely matched by the results shown in figure 5.4b. This plot shows draws from the parameter posterior using the brief Langevin method based on a Swendsen-Wang sampling inner loop described in Zhu and Ghahramani (2002). We also reparameterized the posterior to take gradients with respect to $\log(\boldsymbol{\sigma})$ rather than $\boldsymbol{\sigma}$. This is important for any unconstrained gradient method like Langevin. Note that predictions from typical samples of $\boldsymbol{\sigma}$ will vary greatly. For example large σ_x

predicts the unlabeled cluster in the top left as mainly ×'s, whereas large σ_y predicts ○'s. It would not be possible to obtain the same predictive distribution over labels with a single 'optimal' setting of the parameters as was pursued in Zhu and Ghahramani (2002). This demonstrates how Bayesian inference over the parameters of an undirected model can have a significant impact on predictions.

Figure 5.4c shows that loopy Metropolis converges to a very poor posterior distribution, which does not capture the long arms in figure 5.4b. This is due to poor approximate partition functions from the inner loop. The graph induced by W contains many tight cycles, which cause problems for loopy belief propagation. As expected, loopy propagation gave sensible posteriors on other problems where the observed points were less dense and formed linear chains.

5.2.6 Discussion

Although MCMC sampling in general undirected models is intractable, there are a variety of approximate methods that can be brought forth to tackle this problem. We have proposed and explored a range of such approximations including two variational approximations, brief sampling and the Bethe approximation, combined with Metropolis and Langevin methods. Clearly many more approximations could be explored.

The mean field and tree-based Metropolis algorithms performed disastrously even on simple problems. We believe these failures result from the use of a lower bound as an approximation. Where the lower bound is poor, the acceptance probability for leaving that parameter setting will be exceedingly low. Thus the sampler is often attracted towards extreme regions where the bound is loose, and does not return.

The Bethe free energy based Metropolis algorithm performs considerably better and gave the best results on one of our artificial systems. However it also performed terribly on our final application. In general if an approximation performs poorly in the inner loop then we cannot expect good parameter posteriors from the outer loop. In loopy propagation it is well known that poor approximations result for frustrated systems, and systems with large weights or tight cycles.

The typically broader distributions from brief Langevin and its less rapid failure with strong weights means that we expect it to be more robust than loopy Metropolis. Another advantage is the cost per iteration: the mean field and belief propagation algorithms are iterative procedures with potentially large and uncontrolled costs for convergence. Brief Langevin gave reasonable answers on some large systems where the other methods failed, although it too can suffer from very large artifacts. Even on the very simple heart disease data set one of the posterior marginals was very different under this approximation. This actually means the whole joint distribution is in the wrong place. We now turn to valid MCMC algorithms, which are clearly required if correct inferences are important.

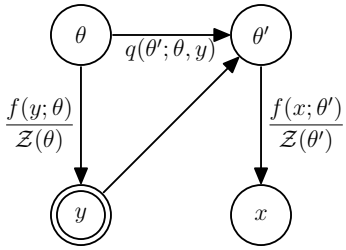


Figure 5.5: An augmented model with the original generative model for observations $p(y, \theta)$ as a marginal. The exchange algorithm is a particular sequence of Metropolis–Hastings steps on this model.

5.3 The Exchange Algorithm

The standard Metropolis–Hastings algorithm proposes taking the data y away from the current parameters θ, which would remove a factor including $1/\mathcal{Z}(\theta)$ from the joint probability $p(\theta, y)$. The proposal also suggests giving the data to the new parameters θ', which introduces a new factor including $1/\mathcal{Z}(\theta')$. If each parameter setting always owned a data set under the model's joint distribution then we would not need to keep adding and removing these $1/\mathcal{Z}$ factors.

We bring the proposed parameter θ' into the model and give it a data set of its own, x. This augmented distribution illustrated in figure 5.5 has joint probability:

$$p(y, \theta, x, \theta') = p(\theta)\frac{f(y;\theta)}{\mathcal{Z}(\theta)}q(\theta' \leftarrow \theta; y)\frac{f(x;\theta')}{\mathcal{Z}(\theta')}. \tag{5.16}$$

Given a setting of (θ, y), a second setting of parameters is generated from an arbitrary distribution, $q(\theta' \leftarrow \theta; y)$. A fantasy dataset is then generated from $p(x|\theta') = f(x;\theta')/\mathcal{Z}(\theta')$, where the function f is the same as in the data-generating likelihood, equation (5.3). The original joint distribution $p(\theta, y)$ is evidently maintained on adding these 'child' variables to the graphical model. As long as we can generate fantasies from this prior model, two Monte Carlo operators are now feasible.

Operator one resamples (θ', x) from its distribution conditioned on (y, θ). This is a block-Gibbs sampling operator, which is naturally implemented by sampling θ' from the proposal followed by generating a fantasy for that parameter setting. Thus we are able to change θ', as long as we can sample from $p(x|\theta')$. Resampling x at the same time removes the need to evaluate the change in $\mathcal{Z}(\theta')$, although this does require an exact sample. This will need a method like *coupling from the past* (section 2.7). Just having a Markov chain with stationary distribution $p(x|\theta')$ is not sufficient.

Operator two is a Metropolis move that proposes swapping the values of the two parameter settings: $\theta \leftrightarrow \theta'$. As the proposal is symmetric the acceptance ratio is simply

the ratio of joint probabilities before and after the swap:

$$\begin{aligned} a &= \frac{q(\theta \leftarrow \theta'; y)\, p(\theta')\, f(y;\theta')\, f(x;\theta)}{\mathcal{Z}(\theta)\,\mathcal{Z}(\theta')} \Big/ \frac{q(\theta' \leftarrow \theta; y)\, p(\theta)\, f(y;\theta)\, f(x;\theta')}{\mathcal{Z}(\theta)\,\mathcal{Z}(\theta')} \\ &= \frac{p(\theta')\, q(\theta \leftarrow \theta'; y)\, f(y;\theta')\, f(x;\theta)}{p(\theta)\, q(\theta' \leftarrow \theta; y)\, f(y;\theta)\, f(x;\theta')} \end{aligned} \tag{5.17}$$

Combining the two operators gives algorithm 5.2, the exchange algorithm.

Algorithm 5.2 Exchange algorithm

Input: initial θ, number of iterations S

1. **for** $s = 1 \dots S$
2. propose $\theta' \sim q(\theta' \leftarrow \theta; y)$
3. generate an auxiliary variable $x \sim f(x;\theta')/\mathcal{Z}(\theta')$
4. compute acceptance ratio:

$$a = \frac{q(\theta \leftarrow \theta'; y)\, p(\theta')\, f(y;\theta')}{q(\theta' \leftarrow \theta; y)\, p(\theta)\, f(y;\theta)} \cdot \frac{f(x;\theta)}{f(x;\theta')}$$

5. draw $r \sim \text{Uniform}[0,1]$
6. **if** $(r < a)$ **then** set $\theta \leftarrow \theta'$
7. **end for**

Each step tries to take the data y from the current parameter setting θ. We speculate that a better parameter setting is θ', which was generated by $q(\theta' \leftarrow \theta; y)$. How can we persuade θ to give up the data to the rival parameter setting θ'? We offer it a replacement data set x from the θ' distribution. If $f(x;\theta)/f(y;\theta) > 1$ then this replacement is preferred by θ to the real data y, which is a good thing. We have to consider both sides of the exchange: the ratio $f(y;\theta')/f(x;\theta')$ measures how much θ' likes the trade in data sets. Only by liking the data and generating good replacements can a parameter enter the set of the posterior samples. Parameters that spread high f settings over many data sets will like the swap, but tend to generate unacceptable fantasies. This penalty replaces the need to compute $\mathcal{Z}$.

Comparing the acceptance ratio to the infeasible M–H algorithm's ratio in algorithm 5.1, we identify that the exchange algorithm replaces the ratio of normalizers with a one-sample unbiased importance sampling estimate (cf equation (1.13)):

$$\frac{\mathcal{Z}(\theta)}{\mathcal{Z}(\theta')} \approx \frac{f(x;\theta)}{f(x;\theta')}, \qquad x \sim f(x;\theta')/\mathcal{Z}(\theta'). \tag{5.18}$$

This gives an interpretation of why the algorithm works. But it is important to remember that arbitrary estimators based on importance sampling or other methods for computing $\mathcal{Z}$ and its ratios are unlikely to give a transition operator with the correct stationary distribution. We emphasize the exchange motivation because the algorithm is simply Metropolis–Hastings, which is already well understood, here applied to an augmented model. For more mathematical detail see subsection 5.3.3 which provides a proof of the stationary distribution for a more general version of the algorithm.

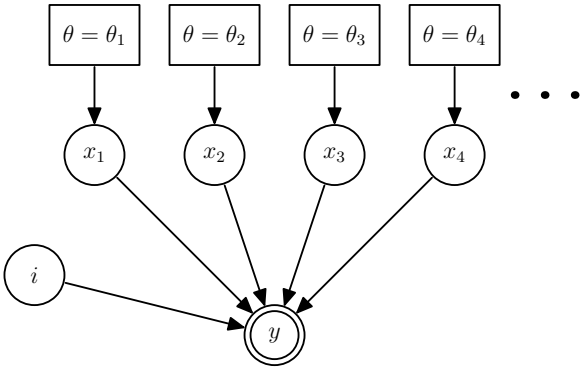


Figure 5.6: An alternative representation of the generative model for observations y. All possible parameter settings, $\{\theta_j\}$, are instantiated, fixed and used to generate a set of data variables $\{x_j\}$. The indicator i is used to set $y = x_i$. The posterior over θ_i, the parameter chosen by the indicator variable i, is identical to $p(\theta|y)$ in the original model.

5.3.1 Product space interpretation

The exchange algorithm was originally inspired by Carlin and Chib (1995), which gives a general method for model comparison by MCMC. In this approach every model is instantiated in the sampler, which explores a product space of all the models' parameter settings. An indicator variable, which is also sampled, specifies which model was used to generate the data. The posterior probability distribution over the indicator variable is the posterior distribution over models. Exchange moves in product spaces are common in the parallel or replica tempering methods in the physics literature, e.g. Swendsen and Wang (1986) and Geyer (1991).

By temporarily assuming that there is only a finite discrete set of possible parameters $\{\theta_j\}$, each setting can be considered as a separate model. This suggests a somewhat strange joint distribution (see also figure 5.6):

$$p(\{x_j\}, y, \theta_i) = p(\theta_i)\mathbb{I}(y = x_j) \prod_j \frac{f(x_j; \theta_j)}{\mathcal{Z}(\theta_j)}, \tag{5.19}$$

where $\mathbb{I}(y = x_i)$ is an indicator function enforcing $y = x_i$. This corresponds to choosing a parameter setting, θ_i, from the prior and generating the observed data as before; but then also generating unobserved data sets using every other setting of the parameters. Although this only appears to be a convoluted restatement of the original generative process, all $\mathcal{Z}(\theta_j)$ terms are now always present; $\mathcal{Z}$ is a constant again.

Each of the unobserved datasets $\{x_{j\neq i}\}$ can be updated by standard MCMC methods such as Gibbs sampling. Conditioned on i, the remaining dataset $x_i = y$ is not updated, because it has a single known, observed value. To update i we notice that an isolated M–H proposal $i \to i'$ will not work; it is exceedingly unlikely that $x_{i'} = y$. Instead we couple $i \to i'$ proposals with an *exchange* of datasets $x_i = y$ and $x_{i'}$. The M–H acceptance

ratio for this proposal is simple; much of the joint distribution in equation (5.19) cancels leaving:

$$a = \frac{q(i \leftarrow i'; y)\, p(\theta_{i'})\, f(y; \theta_{i'})\, f(x_{i'}; \theta_i)}{q(i' \leftarrow i; y)\, p(\theta_i)\, f(y; \theta_i)\, f(x_{i'}; \theta_{i'})}. \tag{5.20}$$

The only remaining problem is that if the number of parameter settings is very large, we require huge amounts of storage and a very long time to reach equilibrium. In particular the method seems impractical if θ is continuous.

The solution to these problems recreates the exchange algorithm of the previous section. We declare that at each time step all of the x variables are at equilibrium conditioned on the indicator variable i. But we do not need to store these values; only when a swap of ownership between θ and θ' is proposed do we need to know the current setting of θ''s data. We can then draw the value from its equilibrium distribution: $x \sim p(x|\theta') = f(x; \theta')/\mathcal{Z}(\theta')$, pretending that this had already been computed. After the exchange has been accepted or rejected the intermediate quantity x can be discarded. We redraw any unobserved data set from its stationary distribution as required. This *retrospective sampling* trick has been used in a variety of other MCMC algorithms for infinite-dimensional models (Papaspiliopoulos and Roberts, 2005; Beskos *et al.*, 2006).

An infinite-dimensional model and retrospective sampling are not required to describe the exchange algorithm, but provide connections to the literature, which might be useful for some readers. The product model works without exact sampling for models with a small number of parameter settings and is also how the algorithm was originally conceived. Other readers may prefer the explanation in the previous section: even on continuous parameters it uses Metropolis–Hastings on a finite auxiliary system, which is more established theoretically.

5.3.2 Bridging Exchange Algorithm

The Metropolis–Hastings acceptance rule has the maximum acceptance rate for any reversible transition operator that may only accept or reject proposals from q (see subsection 2.1.1). As the exchange algorithm is only approximately the same as the direct M–H approach, in some sense it rejects moves that it should not. As we have much better approximations of normalizing constant ratios than equation (5.18), better methods should be possible.

How can good parameter proposals get rejected? It may be that θ' is a much better explanation of the data y than the current parameters θ and would be accepted under M–H. However, the exchange algorithm can reject the swap because $x \sim f(x; \theta')/\mathcal{Z}(\theta')$ is improbable under θ. This makes movement unnecessarily slow and suggests searching for a way to make swaps look more favorable.

The exchange algorithm with bridging draws a fantasy as before, $x_0 \sim f(x_0; \theta')/\mathcal{Z}(\theta')$, but then applies a series of modifications $x_0 \to x_1 \to \cdots x_K$ such that x_K is typically

more appealing to θ. To describe the precise form of these modifications we specify a new augmented joint distribution.

$$\begin{aligned} p(y,\theta,\{x_k\}_{k=0}^K,\theta') = \quad & \text{The augmented model combines:} \\ p(\theta)\frac{f(y;\theta)}{\mathcal{Z}(\theta)} \quad & \text{the original model,} \\ \times\ q(\theta' \leftarrow \theta; y) \quad & \text{a parameter proposal,} \\ \times\ \frac{f(x_0;\theta')}{\mathcal{Z}(\theta')} \quad & \text{generating a fantasy} \\ \times \prod_{k=1}^{K} T_k(x_k \leftarrow x_{k-1};\theta,\theta') \quad & \text{and bridging steps.} \end{aligned} \tag{5.21}$$

We choose the T_k to be Markov chain transition operators with corresponding stationary distributions p_k. A convenient choice is

$$\begin{aligned} p_k(x;\theta,\theta') \propto f(x;\theta')^{(1-\beta_k)} f(x;\theta)^{\beta_k} \equiv f_k(x;\theta,\theta'), \\ \text{where } \beta_k = \frac{k}{K+1}, \end{aligned} \tag{5.22}$$

giving K intermediate distributions that bridge between $p_0(x;\theta,\theta') \equiv f(x;\theta')/\mathcal{Z}(\theta')$ and $p_{K+1}(x;\theta,\theta') \equiv f(x;\theta)/\mathcal{Z}(\theta)$. Other bridging schemes could be used. In what follows we only assume the directional symmetry $p_k(x;\theta,\theta') \equiv p_{K+1-k}(x;\theta',\theta)$. Similarly we require

$$T_k(x_k \leftarrow x_{k-1};\theta,\theta') = T_{K+1-k}(x_k \leftarrow x_{k-1};\theta',\theta), \tag{5.23}$$

which is easily achieved by always using the same transition operator for the same underlying stationary distribution. As before there are two possible Markov chain operators.

Operator one: block Gibbs sample from

$$p(\theta',\{x_k\}|y,\theta) = q(\theta' \leftarrow \theta; y)\frac{f(x_0;\theta')}{\mathcal{Z}(\theta')}\prod_{k=1}^{K} T_k(x_k \leftarrow x_{k-1};\theta,\theta'). \tag{5.24}$$

Operator two: propose swapping $\theta' \leftrightarrow \theta$ whilst simultaneously reversing the order of the $\{x_k\}$ sequence, as illustrated in figure 5.7.

If the T_k operators satisfy detailed balance, i.e.

$$T_k(x' \leftarrow x;\theta,\theta')p_k(x;\theta,\theta') = T_k(x \leftarrow x';\theta,\theta')p_k(x';\theta,\theta'), \tag{5.25}$$

then the M–H acceptance ratio for operator two does not depend on the details of T_k and is easily computed. The concatenation of the two operators results in algorithm 5.3, figure 5.7. Note that $K = 0$ reduces to the previous exchange algorithm.

As before the $\prod_{k=0}^{K} f_{k+1}(x_k;\theta,\theta')/f_k(x_k;\theta,\theta')$ term in the acceptance ratio corresponds

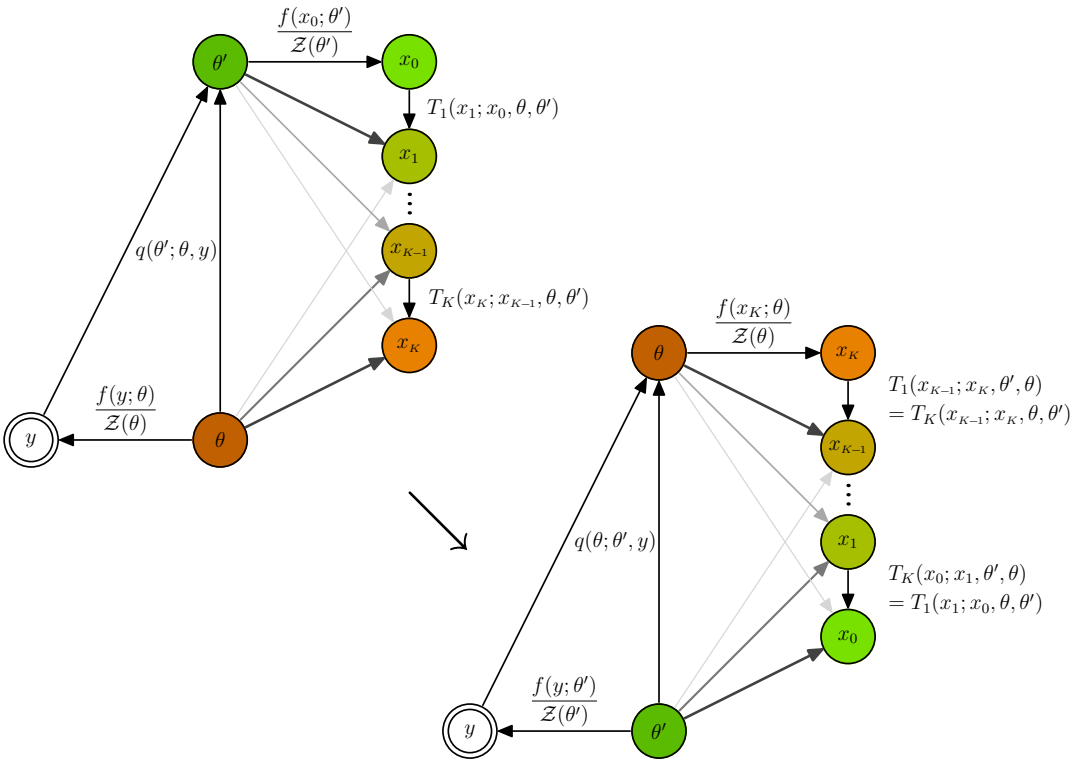


Figure 5.7: The proposed change under a bridged exchange. Given a current parameter θ the algorithm generated θ', an exact sample x_0 from its distribution and a sequence of data sets that come from distributions closer and closer to $p(x|\theta)$. The swap move proposes making θ' the current parameter and θ the owner of a fantasy. As x_0 was typical of θ' not θ the stack of auxiliary variables is reversed so it looks like it might have been generated under the original process.

Algorithm 5.3 Exchange algorithm with bridging

Input: initial θ, #iterations S, #bridging levels K

1. **for** $s = 1 \dots S$
2. propose $\theta' \sim q(\theta' \leftarrow \theta; y)$
3. generate an auxiliary variable by exact sampling:
$$x_0 \sim p_0(x_0; \theta, \theta') \equiv f(x_0; \theta')/\mathcal{Z}(\theta')$$
4. generate K further auxiliary variables with transition operators:
$$x_1 \sim T_1(x_1 \leftarrow x_0; \theta, \theta')$$
$$x_2 \sim T_2(x_2 \leftarrow x_1; \theta, \theta')$$
$$\dots$$
$$x_K \sim T_K(x_K \leftarrow x_{K-1}; \theta, \theta')$$
5. compute acceptance ratio:
$$a = \frac{q(\theta \leftarrow \theta'; y)\, p(\theta')\, f(y; \theta')}{q(\theta' \leftarrow \theta; y)\, p(\theta)\, f(y; \theta)} \cdot \prod_{k=0}^{K} \frac{f_{k+1}(x_k; \theta, \theta')}{f_k(x_k; \theta, \theta')} \tag{5.26}$$
6. draw $r \sim \text{Uniform}[0, 1]$
7. **if** $(r < a)$ **then** set $\theta \leftarrow \theta'$
8. **end for**

to an unbiased estimate of $\mathcal{Z}(\theta)/\mathcal{Z}(\theta')$. Proof: it is an annealed importance sampling (AIS) weight, equation (2.24). This is the natural extension to the simple importance estimate, equation (5.18), in the original exchange algorithm. Linked importance sampling (Neal, 2005) could also be used as a drop-in replacement.

The bridging extension to the exchange algorithm allows us to improve its implicit normalization constant estimation and improve the acceptance rate, for some additional cost. Fortunately no further expensive exact sampling, on top of that needed by the original algorithm, is required per iteration. The performance as a function of K is explored in section 5.6.

5.3.3 Details for proof of correctness

It turns out that the algorithm as stated is also valid when the operators T_k do not satisfy detailed balance. We give the details of a proof for this more general case.

We define a set of reverse Markov chain operators

$$\widetilde{T}_k(x \leftarrow x'; \theta, \theta') \propto T_k(x' \leftarrow x; \theta, \theta')\, p_k(x; \theta, \theta') = \frac{T_k(x' \leftarrow x; \theta, \theta')\, p_k(x; \theta, \theta')}{p_k(x'; \theta, \theta')}. \quad (5.27)$$

Define the transition operator $\mathcal{T}$ by steps 2–7 of algorithm 5.3. Let $\widetilde{\mathcal{T}}$ be defined by the same algorithm but using $\{\widetilde{T}_k\}$ rather than $\{T_k\}$. We now prove that both $\mathcal{T}$ and $\widetilde{\mathcal{T}}$ leave $p(\theta|y)$ stationary:

$$\begin{aligned}
\mathcal{T}(x_0, \ldots, x_K, \theta' \leftarrow \theta) p(\theta|y) = & \quad \text{Probability of transition is} \\
p(\theta|y) = \frac{p(y|\theta)p(\theta)}{p(y)} = \frac{f(y;\theta)p(\theta)}{\mathcal{Z}(\theta)p(y)} & \quad \text{probability of being at } \theta, \\
\times\, q(\theta' \leftarrow \theta; y) & \quad \text{proposing a move to } \theta', \\
\times\, \frac{f(x_0;\theta')}{\mathcal{Z}(\theta')} \equiv \frac{f_0(x_0;\theta,\theta')}{\mathcal{Z}(\theta')} & \quad \text{generating } x_0, \\
\times\, \prod_{k=1}^{K} T_k(x_k \leftarrow x_{k-1}; \theta, \theta') & \quad \text{generating } (x_1, \ldots, x_K) \\
\times\, \min\left[1, \frac{q(\theta \leftarrow \theta'; y)\, p(\theta')\, f(y;\theta')}{q(\theta' \leftarrow \theta; y)\, p(\theta)\, f(y;\theta)} \prod_{k=0}^{K} \frac{f_{k+1}(x_k;\theta,\theta')}{f_k(x_k;\theta,\theta')}\right] & \quad \text{and finally accepting.}
\end{aligned}$$

Rearranging and using $f_{K+1}(x_K; \theta, \theta') \equiv f(x_K; \theta)$:

$$\begin{aligned}
= \min \Bigg[& \frac{p(\theta)\, f(y;\theta)\, q(\theta' \leftarrow \theta; y)\, f(x_0;\theta')}{p(y)\, \mathcal{Z}(\theta)\, \mathcal{Z}(\theta')} \prod_{k=1}^{K} T_k(x_k \leftarrow x_{k-1}; \theta, \theta')\ , \\
& \frac{p(\theta')\, f(y;\theta')\, q(\theta \leftarrow \theta'; y)\, f(x_K;\theta)}{p(y)\, \mathcal{Z}(\theta')\, \mathcal{Z}(\theta)} \prod_{k=1}^{K} \frac{T_k(x_k \leftarrow x_{k-1}; \theta, \theta')\, f_k(x_{k-1};\theta,\theta')}{f_k(x_k;\theta,\theta')} \Bigg].
\end{aligned}$$

We substitute equation (5.27) and equation (5.23) into the second product and reorder its terms:

$$= \min \left[\frac{p(\theta)\, f(y;\theta)\, q(\theta' \leftarrow \theta; y)\, f(x_0;\theta')}{p(y)\, \mathcal{Z}(\theta)\, \mathcal{Z}(\theta')} \prod_{k=1}^{K} T_k(x_k \leftarrow x_{k-1}; \theta, \theta') \;, \right. \\ \left. \frac{p(\theta')\, f(y;\theta')\, q(\theta \leftarrow \theta'; y)\, f(x_K;\theta)}{p(y)\, \mathcal{Z}(\theta')\, \mathcal{Z}(\theta)} \prod_{k=1}^{K} \widetilde{T}_k(x_{K-k} \leftarrow x_{K+1-k}; \theta', \theta) \right] . \tag{5.28}$$

Now swapping θ and θ', reversing the order of the auxiliary variables, i.e. mapping $(x_0, x_1, \ldots, x_K) \to (x_K, \ldots, x_1, x_0)$ and swapping T and $\widetilde{T}$ throughout, only swaps the arguments of the min leaving the value of the expression unchanged. Therefore, the probability of the reverse transition under $\widetilde{\mathcal{T}}$, via the same intermediate values, is the same as in equation (5.28). That is, for any values of $\{x_k\}$

$$\mathcal{T}((x_0, \ldots, x_K), \theta' \leftarrow \theta)\; p(\theta|y) = \widetilde{\mathcal{T}}((x_K, \ldots, x_0), \theta \leftarrow \theta')\; p(\theta'|y)\,. \tag{5.29}$$

Summing over all possible intermediate values $\{x_k\}$ gives

$$\mathcal{T}(\theta' \leftarrow \theta)\, p(\theta|y) = \widetilde{\mathcal{T}}(\theta \leftarrow \theta')\, p(\theta'|y)\,. \tag{5.30}$$

Further summing over θ or θ' shows that both $\mathcal{T}$ and $\widetilde{\mathcal{T}}$ leave $p(\theta|y)$ stationary. If, as was originally proposed, T is reversible then $\mathcal{T} \equiv \widetilde{\mathcal{T}}$ and the bridged exchange algorithm also satisfies detailed balance.

As an aside we mention that the bridging scheme is very similar to that found in the method of *dragging fast variables* (Neal, 2004a). Although dragging was presented using reversible transition operators, a similar proof to the one above shows that it too can use non-reversible transition operators.

5.4 The Single Auxiliary Variable Method

The first valid MCMC algorithm for doubly-intractable distributions was discovered by Møller *et al.* (2004, 2006). For comparison this section reviews their method, which we call the Single Auxiliary Variable Method (SAVM).

SAVM extends the original model to include a single auxiliary variable, x, which shares the same state space as y (figure 5.8):

$$p(x, y, \theta) = p(x|\theta, y)\frac{f(y;\theta)}{\mathcal{Z}(\theta)}p(\theta). \tag{5.31}$$

The joint distribution $p(y, \theta)$ is unaffected. No known method of defining auxiliary variables removes $\mathcal{Z}(\theta)$ from the joint distribution. However, through careful choice

Figure 5.8: (a) the original model, unknown parameters θ generated observed variables y, (b) the SAVM augmented model. The conditional distribution of x must have a tractable θ dependence. In existing approaches this distribution is only a function of one of y or θ, e.g. $f(x;\hat{\theta}(y))/\mathcal{Z}(\hat{\theta})$ or a normalizable function of (x,θ).

of q, explicit $\mathcal{Z}(\theta)$ dependence can be removed from the M–H ratio

$$a = \frac{p(x',\theta'|y)\, q(x,\theta \leftarrow x',\theta';y)}{p(x,\theta|y)\, q(x',\theta' \leftarrow x,\theta;y)} \ . \tag{5.32}$$

A convenient form of proposal distribution is

$$q(x',\theta' \leftarrow x,\theta;y) = q(\theta' \leftarrow \theta;y)\, q(x';\theta'), \tag{5.33}$$

which corresponds to the usual change in parameters $\theta \to \theta'$, followed by a choice for the auxiliary variable. If this choice, which happens to ignore the old x, uses

$$q(x';\theta') = f(x';\theta')/\mathcal{Z}(\theta') \ , \tag{5.34}$$

where f and $\mathcal{Z}$ are the same functions as in $p(y|\theta)$, equation (5.3), then the M–H acceptance ratio becomes

$$\begin{aligned} a &= \frac{p(x'|\theta',y)}{p(x|\theta,y)} \frac{p(\theta'|y)}{p(\theta|y)} \frac{q(x;\theta)}{q(x';\theta')} \frac{q(\theta \leftarrow \theta';y)}{q(\theta' \leftarrow \theta;y)} \\ &= \frac{p(x'|\theta',y)}{p(x|\theta,y)} \frac{\mathcal{Z}(\theta) f(y;\theta') p(\theta')}{\mathcal{Z}(\theta') f(y;\theta) p(\theta)} \frac{f(x;\theta)\mathcal{Z}(\theta')}{f(x';\theta')\mathcal{Z}(\theta)} \frac{q(\theta \leftarrow \theta';y)}{q(\theta' \leftarrow \theta;y)} \\ &= \frac{f(y;\theta')p(\theta')}{f(y;\theta)p(\theta)} \frac{q(\theta \leftarrow \theta';y)}{q(\theta' \leftarrow \theta;y)} \cdot \frac{p(x'|\theta',y)}{p(x|\theta,y)} \frac{f(x;\theta)}{f(x';\theta')} \ . \end{aligned} \tag{5.35}$$

Now every term can be computed. As with the exchange algorithm, the big assumption is that we can draw independent, exact samples from the proposal distribution, equation (5.34).

The missing part of this description was the conditional distribution of the auxiliary variable $p(x|\theta,y)$. This choice is not key to constructing a valid M–H algorithm but our choice will have a strong impact on the efficiency of the Markov chain. Normally we have a choice over the proposal distribution. Here that choice is forced upon us and instead

we choose the target distribution $p(x|y,\theta)$ to match the proposals as closely as possible. We cannot maximize the acceptance rate by choosing $p(x|y,\theta) = f(x;\theta)/\mathcal{Z}(\theta)$, as that would reintroduce explicit $\mathcal{Z}(\theta)$ terms into the M–H ratio. Møller *et al.* suggested two possibilities: 1) use a normalizable approximation to the ideal case, 2) replace θ with a point estimate $\hat{\theta}$, such as the maximum pseudo-likelihood estimate based on the observations y. This gives an auxiliary distribution

$$p(x|\theta,y) = p(x|y) = p(x|\hat{\theta}(y)) = \frac{f(x;\hat{\theta})}{\mathcal{Z}(\hat{\theta})} \ , \tag{5.36}$$

with a fixed normalization constant $\mathcal{Z}(\hat{\theta})$, which will cancel in equation (5.35). The broken lines in figure 5.8b indicate that while x could be a child of θ and y, in practice previous work has only used one of the possible parents. For concreteness we assume $p(x|\theta,y) = f(x|\hat{\theta})/\mathcal{Z}(\hat{\theta})$ for some fixed $\hat{\theta}(y)$ in all that follows, but our results are applicable to either case.

5.4.1 Reinterpreting SAVM

Seen in the light of the product model in figure 5.6, Møller *et al.*'s SAVM method appears slightly strange. SAVM can be reproduced by augmenting our joint model in figure 5.6, containing variables $x_1, x_2, \ldots$, with an additional arbitrary latent, x with no subscript, as used in SAVM. Then we can define the following proposal:

1. Draw $j \sim q(j \leftarrow i)$
2. Perform the deterministic three-way swap $(x, x_i, x_j) \leftarrow (x_j, x, x_i)$.

Before the swap x_i was equal to the observed data y, so after the swap x_j will be equal to the data, now "owned" by θ_j. The acceptance ratio for this proposal is precisely as in SAVM, equation (5.35). If we want to take y from θ_i and give it to rival setting θ_j why involve a third parameter $\hat{\theta}$? In section 5.6 we will see that the third party can make the transaction harder or mediate it. The bridging steps in subsection 5.3.2 were specifically designed to make the swap more palatable. In the next section we propose an extension of SAVM which has similar bridging steps.

5.5 MAVM: a tempered-transitions refinement

As with the exchange algorithm, SAVM's acceptance ratio, equation (5.35), can be seen as an approximation to the exact normalization constant evaluation in algorithm 5.1. SAVM uses the following two unbiased one-sample importance-sampling estimators:

$$\frac{\mathcal{Z}(\hat{\theta})}{\mathcal{Z}(\theta')} \approx \frac{f(x';\hat{\theta})}{f(x';\theta')} \qquad x' \sim f(x;\theta')/\mathcal{Z}(\theta'), \tag{5.37}$$

$$\frac{\mathcal{Z}(\hat{\theta})}{\mathcal{Z}(\theta)} \approx \frac{f(x;\hat{\theta})}{f(x;\theta)} \qquad x \sim f(x;\theta)/\mathcal{Z}(\theta). \tag{5.38}$$

A biased estimate of $\mathcal{Z}(\theta)/\mathcal{Z}(\theta')$ is obtained by dividing equation (5.37) by equation (5.38). The unknown constant $\mathcal{Z}(\hat{\theta})$ fortuitously cancels and amazingly substituting this elementary approximation into the M–H algorithm 5.1 gave a valid method.

As with the exchange algorithm, SAVM's "importance sampling" estimators are very crude. A large mismatch between $p(x|\theta, y)$ and $q(x; \theta, y)$ can cause a high M–H rejection rate. Bridging between these two distributions might help, which suggests replacing the two importance sampling estimators with annealed importance sampling estimates. This gives algorithm 5.4. Our new algorithm has K+1 auxiliary variables and collapses to SAVM for $K = 0$. We call this method with $K \geq 1$ the multiple auxiliary variable method (MAVM).

Algorithm 5.4 Multiple auxiliary variable method (MAVM)

Input: initial (θ, X), #iterations S, #bridging levels K

1. **for** $s = 1 \ldots S$
2. propose $\theta' \sim q(\theta' \leftarrow \theta; y)$
3. propose the first component of X' by exact sampling:
$$x'_0 \sim p_0(x'_0; \hat{\theta}(y), \theta') \equiv f(x'_0; \theta')/\mathcal{Z}(\theta')$$
4. propose remainder of X' using K transition operator steps:
$$\begin{aligned} x'_1 &\sim T_1(x'_1 \leftarrow x'_0; \hat{\theta}(y), \theta') \\ x'_2 &\sim T_2(x'_2 \leftarrow x'_1; \hat{\theta}(y), \theta') \\ &\cdots \\ x'_K &\sim T_K(x'_K \leftarrow x'_{K-1}; \hat{\theta}(y), \theta') \end{aligned}$$
5. compute acceptance ratio:
$$a = \frac{f(y; \theta')\, p(\theta')}{f(y; \theta)\, p(\theta)} \frac{q(\theta \leftarrow \theta'; y)}{q(\theta' \leftarrow \theta; y)} \cdot \prod_{k=0}^{K} \frac{f_k(x_k; \hat{\theta}, \theta)}{f_{k+1}(x_k; \hat{\theta}, \theta)} \frac{f_{k+1}(x'_k; \hat{\theta}, \theta')}{f_k(x'_k; \hat{\theta}, \theta')}$$
6. draw $r \sim \text{Uniform}[0, 1]$
7. **if** $(r < a)$ **then** set $(\theta, X) \leftarrow (\theta', X')$
8. **end for**

As in the bridged exchange algorithm, ratios involving the auxiliary variables can be computed online as they are generated; there is no need to store the whole X ensemble. The T_k are any convenient transition operators that leave corresponding distributions p_k stationary where

$$p_k(x; \hat{\theta}, \theta) \propto f(x; \theta)^{(1-\beta_k)} f(x; \hat{\theta})^{\beta_k} \equiv f_k(x; \hat{\theta}, \theta), \tag{5.39}$$

$$\text{and } \beta_k = \frac{k}{K+1}. \tag{5.40}$$

Other sequences of stationary distributions are possible. They must start at $p_0 = f(x; \theta)/\mathcal{Z}(\theta)$, as we are forced to draw an exact sample from this as part of the proposal. From there they should bridge towards the approximate or estimator-based distribution used by SAVM.

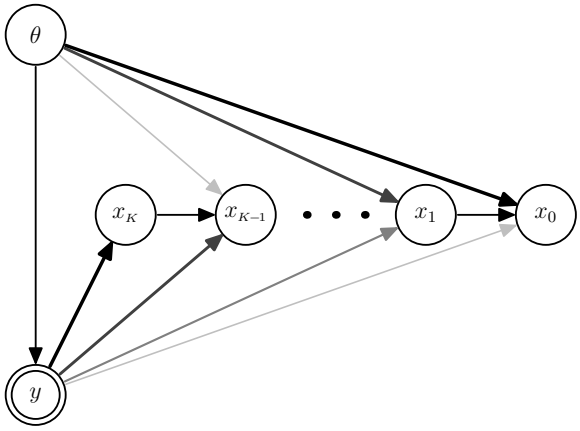


Figure 5.9: The joint distribution for the annealing-based multiple auxiliary variable method (MAVM). Here it is assumed that $p(x_K|\theta, y)$ is based only on a data-driven parameter estimate as in equation (5.36). The auxiliary variables bridge towards the distribution implied by θ. The gray-level and thickness of the arrows from y and θ indicate the strengths of influence on the auxiliary variables. These are controlled by $\{\beta_k\}$ in equation (5.39).

To motivate and validate this algorithm we extend the auxiliary variables x to an ensemble of variables $X = \{x_0, x_1, x_2, ...x_K\}$, figure 5.9. We give x_K the same conditional distribution as the single auxiliary variable x in SAVM, equation (5.36). The distribution over the remaining variables is defined by a sequence of Markov chain transition operators $\widetilde{T}_k(x_{k-1} \leftarrow x_k)$ with $k = K \ldots 1$:

$$\begin{aligned} p(x_{K-1}|x_K, \theta, y) &\sim \widetilde{T}_K(x_{K-1} \leftarrow x_K; \hat{\theta}(y), \theta) \\ &\cdots \\ p(x_1|x_2, \theta, y) &\sim \widetilde{T}_2(x_1 \leftarrow x_2; \hat{\theta}(y), \theta) \\ p(x_0|x_1, \theta, y) &\sim \widetilde{T}_1(x_0 \leftarrow x_1; \hat{\theta}(y), \theta), \end{aligned} \tag{5.41}$$

where as usual

$$T_k(x' \leftarrow x)\, p_k(x) = \widetilde{T}_k(x \leftarrow x')\, p_k(x') \ . \tag{5.42}$$

This defines a stationary distribution over the ensemble $p(X|\theta, y)p(\theta|y)$. Treating the procedure in algorithm 5.4 as a proposal, $q(X', \theta'; X, \theta)$, the acceptance rule is that of standard Metropolis–Hastings. As in other bridging schemes the details of the transition operators cancel after substituting equation (5.42).

While we started by replacing SAVM's importance sampling estimators with AIS, the resulting algorithm is more closely related to "Tempered Transitions" (Neal, 1996a). Our approach has cheaper moves than standard tempered transitions, which would regenerate $x_K \ldots x_0$ from $p(X|\theta, y)$ before every M–H proposal. This is exploiting the generalization of tempered transitions introduced in subsection 2.5.4.2.

As with adding bridging to the exchange algorithm, MAVM makes SAVM a closer match to ideal Metropolis–Hastings sampling. There is an additional cost of K Markov

chain steps per iteration, but no additional exact sampling, which might need many Markov chain steps. We have also provided an answer to an open question in Møller *et al.* (2004) on how to use both θ and y in the auxiliary distribution $p(x|\theta, y)$. We use y in coming up with a point estimate of the parameters to get a distribution in roughly the right place. Then we bridge towards a better fit to $f(x;\theta)/\mathcal{Z}(\theta)$ using ideas from annealing.

5.6 Comparison of the exchange algorithm and MAVM

We first consider a concrete example for which all computations are easy. This allows comparison with exact partition function evaluation (algorithm 5.1) and averaging over chains starting from the true posterior. We consider sampling from the posterior of a single precision parameter θ, which has likelihood corresponding to N i.i.d. zero-mean Gaussian observations $y = \{y_1, y_2, \dots y_N\}$, with a conjugate prior:

$$p(y_n|\theta) = \mathcal{N}(0, 1/\theta), \quad p(\theta|\alpha, \beta) = \text{Gamma}(\alpha, \beta). \tag{5.43}$$

The corresponding posterior is tractable

$$p(\theta|y) = \text{Gamma}\left(N/2 + \alpha,\ \textstyle\sum_n y_n^2/2 + \beta\right), \tag{5.44}$$

but we pretend that the normalizing constant in the likelihood is unknown. We compare the average acceptance rate of the algorithms for two choices of proposal distribution $q(\theta' \leftarrow \theta; y)$.

All of the algorithms require N exact Gaussian samples, for which we used standard generators. For large N one could also generate the sufficient statistic $\sum_n x_n^2$ with a Chi-squared routine. We also draw directly from the Gaussian stationary distributions, p_k, in the bridging algorithms. This simulates an ideal case where the energy levels are close, or the transition operators mix well. More levels would be required for the same performance with less efficient operators. We now report results for $\alpha = 1$, $\beta = 1$, $N = 1$ and $y = 1$.

The first experiment uses proposals drawn directly from the parameter posterior, equation (5.44). The M–H acceptance probability becomes $a \equiv 1$; all proposals are accepted when $\mathcal{Z}(\theta)$ is computed exactly. Therefore any rejections are undesirable by-products of the auxiliary variable scheme, which can only (implicitly) obtain noisy estimates of the normalizing constants. Figure 5.10a shows that both MAVM and the exchange algorithm improve over the SAVM baseline of Møller *et al.* (2006). It appears that a large number, K, of bridging levels are required to bring the acceptance rate close to the attainable $a = 1$. However, significant benefit is obtained from a relatively small number of levels, after which there are diminishing returns. As each algorithm requires an exact

sample, which in applications can require many Markov chain steps, the improvement from a few extra steps ($K>0$) can be worth the cost (see subsection 5.6.1).

In this artificial situation the performance of MAVM was similar to the exchange algorithm. This result favors the exchange algorithm, which has a slightly simpler update rule and does not need to find a maximum (pseudo)-likelihood estimate before sampling begins. In figure 5.10a we had set $\hat{\theta}=1$. Figure 5.10b shows that the performance of MAVM falls off when this estimate is of poor quality. For moderate K, the exchange algorithm automatically obtains an acceptance rate similar to the best possible performance of MAVM; only for $K=0$ was performance considerably worse than SAVM. For this simple posterior $\hat{\theta}$ sometimes manages to be a useful intermediary, but by $K=1$ the exchange algorithm has caught up with MAVM.

More importantly, the exchange algorithm performs significantly better than SAVM and MAVM in a more realistic situation where the parameter proposal $q(\theta' \leftarrow \theta; y)$ is not ideal. Figure 5.10c shows results using a Gaussian proposal centered on the current parameter value. The exchange algorithm exploits the local nature of the proposal, rapidly obtaining the same acceptance rate as exactly evaluating $\mathcal{Z}(\theta)$. MAVM performs much worse, although adding bridging levels does rapidly improve performance over the original SAVM algorithm. SAVM is now hindered by $\hat{\theta}$, which is more rarely between θ and θ'.

The posterior distribution over θ, equation (5.44), becomes sharper for $N>1$. This makes the performance of SAVM and MAVM fall off more rapidly as $\hat{\theta}$ is moved away from its optimum value. These methods require better estimates of θ with larger datasets.

5.6.1 Ising model comparison

We have also considered the Ising model distribution with $y_i \in \{\pm 1\}$ on a graph with nodes i and edges E:

$$p(y|\theta) = \frac{1}{\mathcal{Z}(\theta)} \exp\Big(\sum_{(i,j)\in E} \theta_J y_i y_j + \sum_i \theta_h y_i \Big). \tag{5.45}$$

We used the *summary states* algorithm (subsection 2.7.1) for exact sampling and a single sweep of Gibbs sampling for the transition operators T. Results are reported for y drawn from a model with $\theta_h=0$ and $\theta_J=0.3$.

Møller *et al.* (2004) used uniform priors over $|\theta_h|<1$ and $0<\theta_J<1$. This only works (or seems to) for a $q(\theta' \leftarrow \theta)$ with small step sizes. The algorithms hang if $\theta_J>0.44$ is proposed because CFTP based on Gibbs sampling takes a *very* long time to return a sample in this regime. We used a uniform prior over $0<\theta_J<0.4$ and larger, closer-to-optimal step sizes.

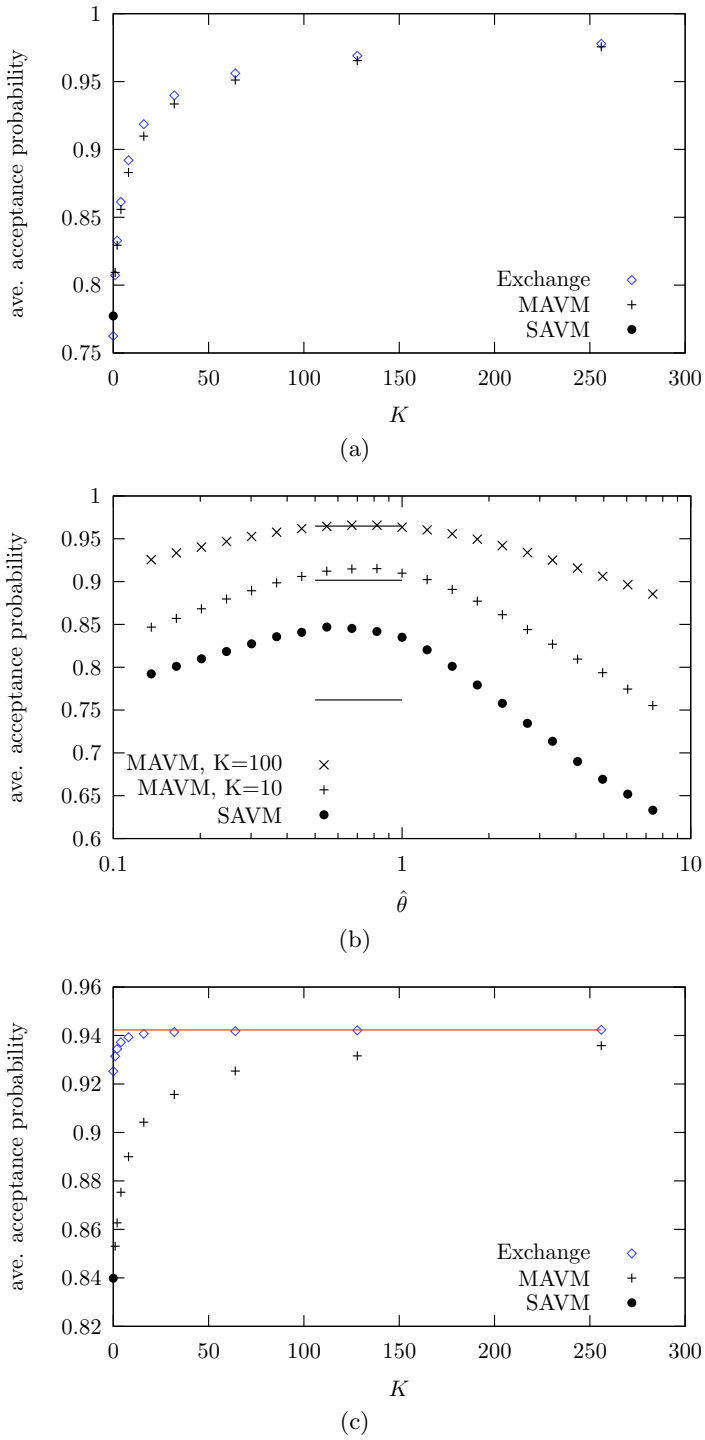
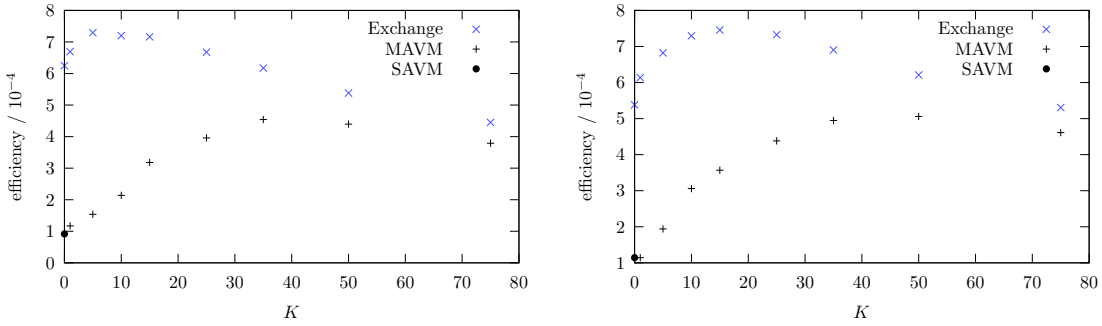


Figure 5.10: Comparison of MAVM and the exchange algorithm learning a Gaussian precision. (a) Average acceptance rate as a function of K. MAVM with $K=0$ corresponds to SAVM, the method of Møller *et al.* (2004). Exact normalizing constant evaluation in would give an acceptance rate of one. (b) Average acceptance rate as a function of the initial parameter estimate required by SAVM ($K=0$) and the extended version, MAVM. Horizontal bars show the results for the exchange algorithm, which has no $\hat{\theta}$, for $K = 0, 10, 100$. (c) As in (a) but with a Gaussian proposal distribution of width 0.1 centered on the current parameter setting. The horizontal line shows the maximum average acceptance rate for a reversible transition operator, this is obtained by exact normalizing constant evaluation.



(a) $\sigma = 0.04$ (b) $\sigma = 0.10$

Figure 5.11: Performance on a 10×30 toroidal square Ising lattice. The data were generated from an exact sample with $\theta_J = 0.3$ and $\theta_h = 0$. Proposals were Gaussian perturbations of width σ. The plot shows efficiency: effective number of samples, estimated by R-CODA (Cowles *et al.*, 2006), divided by the total number of Gibbs-sampling sweeps (computer time). All the algorithms are expensive: SAVM requires 10^4 Gibbs sweeps of the Ising model to obtain one effective sample of the parameters. This could be improved by using more advanced exact sampling methods.

Figure 5.11 gives some example results for two different step sizes. For both methods the number of effective samples obtained per iteration will tend to the maximum possible as $K \to \infty$. In practice the performance of the two methods does converge. However, these bridging steps come at a cost; the best K is a compromise between the computer time per iteration and the mixing time of the chain. In the examples we tried the exchange algorithm provided better mixing times than SAVM/MAVM for all finite K. Bridging provided good improvements over SAVM, but only gave appreciable benefits to the exchange algorithm for larger step sizes.

5.6.2 Discussion

MCMC methods typically navigate complicated probability distributions by local diffusion — longer range proposals will be rejected unless carefully constructed. It is usually the case that as the step-size of proposals are made sufficiently small the acceptance rate of a M–H method tends to one. However, SAVM does not have this property, it introduces rejections even when $\theta' = \theta$. While the exchange algorithm has $a \to 1$ for all K as the step-size tends to zero, MAVM will only recover $a = 1$ as $K \to \infty$. This is because the third party in the proposed swap (see subsection 5.4.1) is not necessarily close to θ. Even in a simple unimodal 1-dimensional posterior distribution, figure 5.10c, this is a significant disadvantage in comparison with the exchange algorithm. We found the exchange algorithm performs better than the only other existing MCMC method for this problem and is simpler to implement.

5.7 Latent History methods

Both MAVM and the exchange algorithm require exact samples, and use them to compute similar Metropolis–Hastings acceptance ratios. In this section we propose new algorithms, which are also valid MCMC methods for doubly-intractable distributions but which do not draw exact samples. They do follow a very similar coupling procedure, but the properties of the resulting algorithms are quite different.

We start with an inspiring but impractical rejection sampling algorithm for the parameters θ, given for example by Marjoram *et al.* (2003, Algorithm B):

Algorithm 5.5 Simple rejection sampling algorithm for $\theta \sim p(\theta|y)$

1. Generate $\theta \sim p(\theta)$
2. Simulate $y' \sim p(y'|\theta)$
3. Accept θ if $y' = y$.

On real problems where the data can take on many values this algorithm will rarely accept. Marjoram *et al.* suggest relaxing the equality in step 3 by accepting data sets that are close in some sense. They also suggest MCMC algorithms, which diffuse the parameters in step 3 rather than drawing from the prior. These algorithms are valid samplers for the wrong distribution: $p(\theta|y$ is near observed value). This section will introduce an alternative approach, which tries to make it more probable that the fantasy data in step 2 is equal to the observed data, while keeping the validity of the original algorithm.

A generative model specifies a likelihood $p(y|\theta)$ but not necessarily the details of an algorithm to sample from this distribution. The latent history representation brings a particular sampling procedure into the description of the model. We declare that a stationary ergodic Markov chain T was simulated for an infinite number of time steps, generating an arbitrarily-initialized sequence $X = \{x_{-\infty}, \ldots, x_{-2}, x_{-1}, x_0\}$ and that we observed the data $y \equiv x_0$. The marginal probability of any x_t, including x_0, is the stationary distribution of the Markov chain, which we set to the target model's $p(y|\theta)$. Unraveling a Markov chain and writing it down as a generative model was a trick employed by Hinton *et al.* (2006).

Our latent history representation is illustrated in figure 5.12. As in *coupling from the past* (section 2.7) we also consider the random variables $U = \{u_{-\infty}, \ldots, u_{-2}, u_{-1}, u_0\}$ used by the Markov chain transition operator at each time step. Given θ and the u_t variables, which provide T with its source of randomness, each state is a deterministic function of the previous: $x_t = \tau(x_{t-1}, u_t, \theta)$.

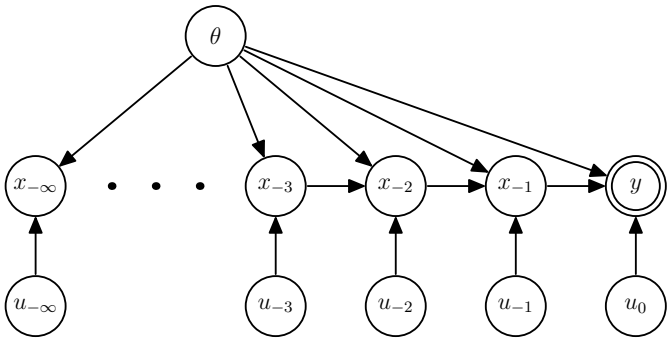


Figure 5.12: The latent history representation augments the observed data y and parameters θ with the history of a Markov chain, $\{x_{-\infty}, \ldots, x_{-2}, x_{-1}\}$, with stationary distribution $p(y|\theta)$. If we can (effectively) sample over all unknowns in this graphical model we have, by discarding $\{x_t, u_t\}$, an algorithm to sample θ.

5.7.1 Metropolis–Hastings algorithm

The latent history representation is a distribution with an infinite number of latent variables. Although we can not explicitly update all of these variables, MCMC sampling is notionally straightforward. Algorithm 5.6 would be a valid MCMC procedure if we had an oracle to perform the infinite number of computations in steps 2 and 5 on our behalf. As an aside we note that this algorithm is similar to Algorithm F in Marjoram *et al.* (2003) for "Approximate Bayesian Computation" (ABC)[3].

Algorithm 5.6 Template for a latent history sampler

Input: initial setting θ, number of iterations S

1. **for** $s = 1 \ldots S$
2. block Gibbs sample from $p(x_{<0}, u_{\leq 0} | x_0 = y, \theta)$
3. propose $\theta' \sim q(\theta' \leftarrow \theta; y)$
4. with probability $\min\left(1, \dfrac{p(\theta')\, q(\theta \leftarrow \theta'; y)}{p(\theta)\, q(\theta' \leftarrow \theta; y)}\right)$
5. follow deterministic map $x'_t = \tau(x'_{t-1}, u_t, \theta')$ to find $y' \equiv x'_0$.
6. **if** $y' = y$ **then** accept move $\theta \leftarrow \theta'$
7. **end for**

The algorithm can feasibly be implemented if T provably coalesces to a single final state from any start state given the sequence of random numbers $\{u_t\}$. This allows step 5 of algorithm 5.6 to be replaced with the procedure used by *coupling from the past* (CFTP, see section 2.7) to identify y' while only examining some of the u_t's. Further,

[3]There are two differences. Firstly the equivalent of the generation in step 5 is always performed before checking the equivalent to step 4. Our version would also be advisable in the original ABC setting, assuming data simulation is more expensive than evaluation of prior probabilities. Secondly, the major difference, we sample the fantasy y' in a way that has a bias towards reproducing y. The algorithms are not directly comparable however: the ABC community assumes that they cannot evaluate a function proportional to $p(y|\theta)$, which would make it difficult to construct the transition operators needed by our algorithm.

by combining steps 5 and 6 the coupling from the past can be terminated early if at any point it becomes clear that $y' \neq y$.

All we require now is an oracle that can return any u_t on request. Creating this service can replace the block Gibbs sampling in step 2, as long as we are consistent with its infinite computation. Markov properties in the model imply

$$p(x_{<0}, u_{\leq 0}|x_0, \theta) = \prod_{t<0} p(x_{t-1}|x_t, \theta) \prod_{t<0} p(u_t|x_t, x_{t-1}). \tag{5.46}$$

This structure allows resampling of $x_{-1}, x_{-2}, \ldots$ in sequence from

$$\begin{aligned} p(x_{t-1}|x_t, \theta) &= \frac{p(x_t|x_{t-1}, \theta)\, p(x_{t-1}|\theta)}{p(x_t|\theta)} \\ &= \frac{T(x_t \leftarrow x_{t-1}; \theta)\, p(x_{t-1}|\theta)}{p(x_t|\theta)} = \widetilde{T}(x_{t-1} \leftarrow x_t; \theta), \end{aligned} \tag{5.47}$$

the reverse transition operator corresponding to T. As long as $p(u_t|x_t, x_{t-1})$ is known for operator T, any u_t value can be sampled after the corresponding x_t and x_{t-1} are available. This means that step 2 in algorithm 5.6 can be implemented lazily: "construct an object that will provide u_t values on request consistent with block Gibbs sampling the entire model". This object will respond to a request for u_t by returning the value from memory if it has previously been sampled, or if necessary by sampling back to x_{t-1} from the furthest back x value that is known and returning a sample from $p(u_t|x_t, x_{t-1})$. Within each iteration all samples are stored for future use to ensure consistency.

The practical implementation of a latent history Metropolis–Hastings sampler is summarized in algorithm 5.7. This includes an optional refinement, 'version b', which analytically integrates out u_0. This requires computing the transition probability for one step of T, which is usually possible and should increase the acceptance rate. The refinement also makes the algorithm feasible for Markov chains that do not completely coalesce, for example on continuous state spaces. A disadvantage of version b) is that step 8 may be difficult to implement without always identifying x'_{-1}, whereas step 5 in version a) will be able to prematurely halt CFTP whenever $y \neq x'_0$, which is often easy to prove.

5.7.2 Performance

A simple special case highlights large differences between latent history samplers and the previous auxiliary variable algorithms.

We first consider version a) of the sampler applied to a large collection of D independent Bernoulli variables sharing a single unknown parameter θ. This is an Ising model in the limit of zero connection strengths. Assume that the transition operator in use, T,

Algorithm 5.7 Metropolis–Hastings latent history sampler

Input: initial setting θ, number of iterations S and an algorithm CFTP that can use a finite number of random numbers to prove where an infinitely long Markov chain ends (see section 2.7).

1. **for** $s = 1 \ldots S$
2. create lazy u_t generator consistent with $p(x_{<0}, u_{\leq 0} | x_0 = y, \theta)$
3. Propose $\theta' \sim q(\theta' \leftarrow \theta; y)$

Either (version a):

4. with probability $\min\left(1, \dfrac{p(\theta')\, q(\theta \leftarrow \theta'; y)}{p(\theta)\, q(\theta' \leftarrow \theta; y)}\right)$
5. identify whether $y' \equiv x'_0$ is y using CFTP and u_t generator
6. **if** $y' = y$ **then** accept move $\theta \leftarrow \theta'$

Or (version b):

7. Draw $r \sim \text{Uniform}[0, 1]$
8. identify (enough about) x'_{-1} for acceptance rule using CFTP and u_t's
9. **if** $r < \dfrac{T(x_0 = y \leftarrow x'_{-1}; \theta')\, p(\theta')\, q(\theta \leftarrow \theta'; y)}{T(x_0 = y \leftarrow x_{-1}; \theta)\, p(\theta)\, q(\theta' \leftarrow \theta; y)}$ **then** accept move $\theta \leftarrow \theta'$
10. **end for**

is Gibbs sampling, which for these independent variables draws from the stationary distribution in one sweep. The sampler is implemented as follows: for each of the observed spins y_d, the corresponding $u_{0,d}$ random variate is uniformly distributed in $[0, \theta]$ if the spin is up, and $[\theta, 1]$ if the spin is down. Now updates to θ are constrained to lie within the two closest surrounding $u_{0,d}$ values. This range has a typical scale of $1/D$. The posterior over parameters has a width that scales as $1/\sqrt{D}$. This suggests that it will take at least $\mathcal{O}(D)$ steps to equilibrate the single scalar parameter θ by random walk sampling. Disappointing given the ideal conditions!

Notice that the performance of the algorithm depends on the details of the transition operators. An unreasonably clever transition operator could generate y if $u_0 < p(y|\theta)$ and some other data set otherwise. Then the probability of finding $y' = y$ is $\min(1, p(y|\theta)/p(y|\theta'))$, giving an overall algorithm similar to the ideal Metropolis–Hastings algorithm (actually the "two stage" rule described in algorithm 2.2). Neither MAVM nor the exchange algorithm have such dependence on the transition operators — how the exact sample was obtained is irrelevant.

We now turn to version b) of the sampler with the same independent Bernoulli model and Gibbs sampling operators. Here the transition probabilities in step 9 are equal to the parameters' likelihoods under the model. The algorithm reduces to standard Metropolis–Hastings for θ, which has the best possible acceptance rate for a given proposal distribution q. Rapidly mixing operators are rewarded by this version of latent histories, operators that move very slowly will perform as in version a).

We have implemented both versions of the sampler and applied them to the same Ising

model problem as in subsection 5.6.1. We used Gibbs sampling for the y variables combined with the summary states algorithm (subsection 2.7.1) to perform inferences required by the algorithm. The summary states code required some modification so that its transitions were driven by the latent history sampler's u-generator rather than a generic random number generator, then the state at time $t=0$ or $t=-1$ was identified as in subsection 2.7.1. The parameter proposals were Gaussian with width $\sigma = 0.04$, which preliminary runs indicated was roughly optimal. In version a) we terminated each iteration as soon as the summary state algorithm identified the state at time zero or a non-? state at time zero was incompatible with the data. In version b) we always waited until x'_{-1} had been identified. More elaborate code could terminate earlier based on bounds of the acceptance ratio, but this is also true for the exchange algorithm, MAVM and most standard Metropolis–Hastings algorithms.

As a result of early terminations version a) was roughly five times cheaper per iteration than our implementation of version b). Even with this advantage the number of effective samples per Gibbs-sampling-like sweep through the system was around ten times worse for version a). The efficiency of version b) was $4{\times}10^{-5}$ effective samples per Gibbs sweep, about half that obtained by SAVM. The latent history efficiency computations included the Gibbs-like sweeps needed to compute the u's, while SAVM was given its random numbers for free. This makes the real computation time of the methods comparable. Which method is actually faster will come down to implementation details. However, both the bridged exchange and MAVM algorithms are clearly faster than either variant of latent histories on this Ising problem.

We might hope that latent histories will perform better on other problems. A more realistic reason to be excited by latent histories is that they might be a much better route to approximate methods than any of the previous algorithms in this chapter. We return to this issue in section 5.9. First we turn to the original motivation for developing latent histories: constructing slice samplers.

5.8 Slice sampling doubly-intractable distributions

All of the algorithms for doubly-intractable distributions described so far are based on Metropolis–Hastings. Such algorithms always require step-size parameters that need to be set well. Overly large step-sizes lead to almost all proposals' being rejected; overly small step-sizes give lengthy random-walk-like exploration of the posterior over θ. It would be nice to be able to use a self-tuning method like slice sampling (subsection 2.4.2), which has less-important or even no step-size parameters.

5.8.1 Latent histories

The latent history representation naturally allows slice sampling. It also has the property, like the exchange algorithm, that small steps in θ are always accepted, so moves will always be possible. The trick is to identify the variables in the joint distribution (figure 5.12) as θ and U, while thinking of the X quantities as just a deterministic function of these variables.

Conditioned on the U variables any standard slice sampling algorithm (subsection 2.4.2) can be applied to the parameter θ. The stationary distribution $p(\theta|U)$ is simply the prior over the parameters conditioned on $x_0(\theta, U) = y$. Any point not satisfying the constraint is off the slice and discarded by the slice sampling procedure. This description gives a slice sampler corresponding to version a) of latent history algorithm 5.7. A slice sampling implementation of version b) sets the stationary distribution to $T(y \leftarrow x_{-1}; \theta)p(\theta)$. After each iteration U is (effectively) updated in a block-Gibbs update. This involves resetting a lazy u_t generator as in the Metropolis–Hastings algorithm.

5.8.2 MAVM

Standard slice sampling cannot be applied to the θ parameter in the MAVM joint distribution, because this would involve computing $\mathcal{Z}(\theta)$. As with Metropolis–Hastings the auxiliary variables must be changed simultaneously. It turns out that, with an appropriate definition of a slice, algorithm 5.4 can be converted into a slice sampler.

We first describe the new algorithm. Ostensibly we do just use a standard slice sampling procedure (subsection 2.4.2) to update the parameters θ. However, the range of auxiliary heights $0 < h < H$ is defined in terms of the following quantity:

$$H(X, \theta) = p(\theta, y)\frac{p(X|\theta, y)}{q(X; \theta, y)}\mathcal{Z}(\hat{\theta}) = f(y; \theta)p(\theta)\prod_{k=0}^{K}\frac{f_{k+1}(x_k; \hat{\theta}, \theta)}{f_k(x_k; \hat{\theta}, \theta)} \ . \tag{5.48}$$

After drawing the slice-sampling auxiliary quantity $h \sim \text{Uniform}[0, H(X, \theta)]$ we consider new parameter settings using any of the normal slice sampling procedures. However, whenever a new setting of the parameters θ' is considered a new setting of auxiliary X' variables is generated in the same way as algorithm 5.4. Then, as usual, slice membership is assertained by checking $H(X', \theta') \geq h$.

We now show that this procedure satisfies detailed balance with respect to the desired stationary distribution. As in subsection 2.4.2 we use the label S for all rejected points and other ancillary points generated while exploring the slice in a stepping-out procedure. The probability of starting at a setting (X, θ), generating a particular setting of

h, generating intermediate quantities S and finally an acceptable new pair (X', θ') is

$$\begin{aligned} & p(\theta|y)\, p(X|\theta, y) \cdot p(h|X, \theta, y) \cdot q(S; \theta, h) \cdot q(\theta'; S, h)\, q(X'; \theta', y) \\ &= p(\theta|y)\, p(X|\theta, y) \cdot \frac{q(X; \theta, y)}{p(X|\theta, y)\, p(\theta, y)\, \mathcal{Z}(\hat{\theta})} \cdot q(S; \theta, h) \cdot q(\theta'; S, h)\, q(X'; \theta', y) \\ &= \frac{1}{p(y)\, \mathcal{Z}(\hat{\theta})}\, q(\theta'; S, h)\, q(S; \theta, h)\, q(X; \theta, y)\, q(X'; \theta', y). \end{aligned} \tag{5.49}$$

All of the slice-sampling bracketing procedures mentioned in subsection 2.4.2 ensure that

$$q(\theta'; S, h)\, q(S; \theta, h) = q(\theta; S, h)\, q(S; \theta', h). \tag{5.50}$$

Therefore equation (5.49) is invariant to exchanging $(X, \theta) \leftrightarrow (X', \theta')$ and the overall procedure satisfies detailed balance.

This derivation closely follows that of standard slice sampling. This is possible because conditioned on h, the decision to reject intermediate parameters in S is the same whether generating S forwards from θ or backwards from θ'. In contrast the acceptability of a proposal in the exchange algorithm cannot be summarized by a scalar slice height h. A move depends on giving a new data set to the old parameter, which will have different probabilities depending on whether the move started at θ or θ'. We don't currently see any way to define a slice sampler for the exchange representation.

As noted in subsection 5.6.2, even very small moves in θ can be rejected when using the MAVM auxiliary system. The usual slice bracketing procedures work on an assumption that nearby regions will always be accepted and exponentially shrink towards the current point. Unless a large number of bridging levels are used this behavior could lead to overly small step sizes. Care would be needed in applying stock slice-sampling code.

5.9 Discussion

We established in subsection 5.1.1 that global information like that provided by exact sampling must be found as part of a strictly valid MCMC algorithm for doubly-intractable distributions. Given this, it seems likely that the exchange algorithm performs about as well as is possible. On the problems we have tried the acceptance rate approaches that of Metropolis–Hastings with moderate K, leaving exact sampling as the dominant cost. We would generally recommend this algorithm if a valid MCMC procedure is required. Møller *et al.* (2004) pointed out that good deterministic approximations to $p(\theta|y)$ could make SAVM perform very well. Our results firmly support the use of bridging, and we would recommend always considering MAVM instead.

There is an unresolved issue concerning doubly-intractable distributions when learning

from N i.i.d. data points:

$$p(y|\theta) = \prod_{n=1}^{N} p(y^{(n)}|\theta) = \frac{1}{\mathcal{Z}(\theta)^N} \prod_{n=1}^{N} f\big(y^{(n)};\theta\big). \tag{5.51}$$

All of the known valid MCMC algorithms for $p(\theta|y)$ require drawing an exact sample from $p(y|\theta)$. This means drawing a fantasy containing N data sets $x = \{x^{(n)}\}_{n=1}^{N}$. But perhaps there is enough information about $\mathcal{Z}(\theta)$ in a single exact sample from the data distribution? Drawing N exact samples may be prohibitively expensive with large data sets. Whether this problem can be avoided is an open question.

Sadly it is not possible to draw exact samples from all distributions; approximate relaxations must be used for some situations. Section 5.2 observed that a bad choice of approximations applied to the inner loop of standard Metropolis–Hastings can have surprising and disastrous consequences on the quality of the approximate samples. The valid algorithms discussed in this chapter are an alternative target for approximation. The obvious relaxation of methods requiring exact sampling from the data distribution is to use a brief run of MCMC, possibly starting from the observed data. Møller *et al.* (2004) report that a fixed length of sampling can be more inefficient than exact sampling for similar levels of performance. However, this idea would be worth exploring further in cases where exact sampling is not possible.

A natural relaxation of the latent history representation is truncating the history to some finite length of time K. This would require specifying an explicit prior distribution for x_{-K}, perhaps an approximation to the original model. One could optionally replace the stationary Markov chain operator T with a sequence of operators T_k bridging between the approximate distribution used for x_{-K} and the target model distribution. This finite model could be simulated using the latent history algorithms with explicit computations, or any standard MCMC technique.

We believe that truncated latent history representations are a promising direction for future research in approximate inference for doubly-intractable distributions. Running finite Markov chains within the exchange algorithm or MAVM is an inner-loop approximation, somewhat like those explored in section 5.2. The consequences of approximating MCMC procedures is hard to predict — it is possible that very unreasonable inferences will result. In contrast a truncated latent history representation is a model on which we can apply standard MCMC algorithms. The model will not describe exactly the same beliefs as the original distribution, but all of the inferences performed on it will be self-consistent. Thus if prior draws from the truncated model look reasonable it is likely that the inferences will also be sensible.

This line of reasoning is really suggesting that we throw out the original doubly-intractable model and replace it with a more tractable latent variable model. If the model was originally introduced as an *ad hoc* way to introduce dependencies, then us-

ing an alternative that can be treated consistently may be preferable. In particular we can look at draws from the prior and see if they actually reflect our beliefs. Heckerman *et al.* (2000) follow a related thought process. They discuss circumstances in which undirected graphical models may not be a natural representation and provide an alternative replacement.

Changing the model doesn't seem attractive when an undirected model was derived directly from physical considerations. An example of such a model is recent work on learning protein structure (Podtelezhnikov *et al.*, 2007), which learns the parameters of an energy function using contrastive divergence (Hinton, 2002). A full Bayesian treatment of these parameters with MCMC seems daunting; exact sampling from the configurational-distribution of a protein chain seems infeasibly difficult. However, using the wrong model could still be used as an approximation technique and doing so may be more stable than inner-loop approximations.

This chapter has not addressed computing marginal likelihoods $p(y|\mathcal{M})$ for Bayesian model comparison of different undirected graphical models[4]. Evidently the standard methods discussed in chapter 4 will apply in theory, although such computations are likely to be demanding.

[4] A *triply*-intractable problem?

Chapter 6

Summary and future work

The Markov chain Monte Carlo method introduced by Metropolis *et al.* (1953) was one of the earliest applications of electronic computers. Today it continues to be an important technique for approximate numerical integration in a growing number of applications. Chapter 1 provided an introduction to the challenges involved, while chapter 2 described some of the current literature along with some minor extensions.

In chapter 3 we explored the promising ideas of using multiple particles and multiple proposals in Markov chain simulations. As with other authors we found that modest improvements are possible, largely derived from the basic length-scale information given by a few samples from a distribution. However drawing multiple proposals can easily cost more to compute than the statistical benefit. It remains to be seen if implementations leveraging caching of intermediate results or parallel computations can gain advantage in real applications. Other authors in the literature remain hopeful that they can, and our pivot-based approaches seem to bring unique benefits to this area of the field.

Chapter 4 investigated methods for computing normalizing constants. One of these, AIS is actually an importance sampling method rather than MCMC, and nested sampling offers a new sampling paradigm in its own right. However, both of these algorithms use Markov chains in their practical implementation. We extended existing techniques for nested sampling on discrete distributions. We then compared three fundamentally different algorithms based on the same Markov chain. Our theoretical and empirical results show that the algorithms perform very differently and that no method is appropriate for all distributions. It would seem wise to use more than one algorithm as standard practice. We proposed one practical approach to realize this: a method for tuning AIS based on preliminary runs of nested sampling.

Chapter 5 serves to highlight the limitations of MCMC. Sampling from doubly-intractable distributions is exceedingly difficult, yet these distributions are found in a large class of 'undirected' statistical models and have received a great deal of at-

tention in recent years. We have introduced three new algorithms for this problem: MAVM, the exchange algorithm and latent history samplers. These improve the existing state of the art, remove the need for separate deterministic approximations and offer new directions for approximating this difficult problem. Exploring truncations of latent histories is a possible area for further work.

This thesis has concentrated on investigating new Markov chains and new algorithms for the better use of existing MCMC operators. These have been investigated empirically and sometimes theoretically on simple problems designed to demonstrate the algorithms' key properties. One limitation of the presentation here has been the focus on stationary distributions — little has been said about the chains' formal convergence rates. This is partly because making meaningful statements about convergence for general statistical problems seems difficult (section 2.3). Also, wherever possible, we have reduced new algorithms to the Metropolis–Hastings algorithm applied to an auxiliary system. This passes some of the burden of deriving rates of convergence onto an established literature.

While we have focused on MCMC methodology and proposed a number of novel algorithms, ultimately the computational advantages of these methods has to be proven on challenging real-world applications. This is an area I hope to spend more time on in my future research.

Bibliography

D. H. Ackley, G. E. Hinton, and T. J. Sejnowski. A learning algorithm for Boltzmann machines. *Cognitive Science*, 9(1):147–169, 1985. (page 128)

J. Albert. Bayesian selection of log-linear models. Technical Report Working Paper 95-15, Duke University, Institute of Statistics and Decision Sciences, 1995. (page 124)

N. Balakrishnan and A. Clifford Cohen. *Order statistics & inference: estimation methods*. Academic Press, San Diego, 1991. ISBN 0120769484. (page 93)

T. Bayes. An essay towards solving a problem in the doctrine of chances. *Philosophical Transations*, 53:370–418, 1763. Communicated by Richard Price, in a letter to John Canton. Also available edited by G. A. Barnard in Biometrika 45(3/4):293–315, December 1958. (page 20)

M. J. Beal. *Variational algorithms for approximate Bayesian inference*. PhD thesis, Gatsby computational neuroscience unit, University College London, 2003. (page 104)

M. J. Beal and Z. Ghahramani. The variational Bayesian EM algorithm for incomplete data: with application to scoring graphical model structures. In J. M. Bernardo, M. J. Bayarri, J. O. Berger, A. P. Dawid, D. Heckerman, A. F. M. Smith, and M. West, editors, *Bayesian statistics 7*, pages 453–464. Oxford University Press, 2003. (page 82)

M. Beltrán, J. García-Bellido, J. Lesgourgues, A. R. Liddle, and A. Slosar. Bayesian model selection and isocurvature perturbations. *Physical Review D*, 71(6):063532, 2005. (pages 109, 118, and 119)

C. H. Bennett. Efficient estimation of free energy differences from Monte Carlo data. *Journal of Computational Physics*, 22(2):245–268, October 1976. (page 117)

B. A. Berg and T. Neuhaus. Multicanonical ensemble: a new approach to simulate first-order phase transitions. *Phys. Rev. Lett.*, 68(1):9–12, January 1992. (page 50)

J. Besag. Spatial interaction and the statistical analysis of lattice systems. *Journal of the Royal Statistical Society*, 36(2):192–236, 1974. (page 124)

J. Besag and P. J. Green. Spatial Statistics and Bayesian Computation. *Journal of the Royal Statistical Society B*, 55(1):25–37, 1993. (page 37)

J. Besag, P. Green, D. Higdon, and K. Mengersen. Bayesian computation and stochastic systems. *Statistical Science*, 10(1):3–41, 1995. (page 77)

A. Beskos, O. Papaspiliopoulos, G. O. Roberts, and P. Fearnhead. Exact and computationally efficient likelihood-based estimation for discretely observed diffusion processes (with discussion). *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68:333–382, 2006. (page 140)

C. Bischof and M. Bücker. Computing derivatives of computer programs. In *Modern Methods and Algorithms of Quantum Chemistry*, volume 3, pages 315–327, 2000. (page 41)

C. M. Bishop. *Pattern recognition and machine learning*. Springer-Verlag, New York, 2006. ISBN 0387310738. (page 20)

O. Cappé, A. Guillin, J.-M. Marin, and C. P. Robert. Population Monte Carlo. *Journal of Computational & Graphical Statistics*, 13(4):907–929, December 2004. (pages 9, 58, 59, 60, and 61)

B. P. Carlin and S. Chib. Bayesian model choice via Markov chain Monte Carlo methods. *Journal of the Royal Statistical Society B (Methodological)*, 57(3):473–484, 1995. (page 139)

G. Casella and C. P. Robert. Rao-Blackwellisation of Sampling Schemes. *Biometrika*, 83(1):81–94, 1996. (page 35)

S. Chib. Marginal likelihood from the Gibbs output. *Journal of the American Statistical Association*, 90(432):1313–1321, December 1995. (page 84)

A. M. Childs, R. B. Patterson, and D. J. C. MacKay. Exact sampling from nonattractive distributions using summary states. *Physical Review E*, 63:036113, 2001. (pages 54 and 134)

A. Christen and C. Fox. MCMC using an approximation. *Journal of Computational and Graphical Statistics*, 14(4):795–810, 2005. (page 31)

J. A. Christen and C. Fox. A self-adjusting multi-scale MCMC algorithm, 2006. Poster presented at Valencia Bayesian meeting, Benidorm 2006. Pre-print available from jac@cimat.mx; code available from `http://www.cimat.mx/~jac/twalk/`. (page 58)

P. Clifford *et al.* Discussion on the meeting on the Gibbs sampler and other Markov chain Monte Carlo methods. *Journal of the Royal Statistical Society. Series B (Methodological)*, 55(1):53–102, 1993. (page 30)

M. K. Cowles, N. Best, K. Vines, and M. Plummer. R-CODA 0.10-5, 2006. Available from `http://www-fis.iarc.fr/coda/`. (pages 64 and 152)

R. T. Cox. Probability, frequency and reasonable expectation. *American Journal of Physics*, 14(1):1–13, January 1946. (pages 20 and 120)

R. V. Craiu and C. Lemieux. Acceleration of the multiple-try Metropolis algorithm using antithetic and stratified sampling. *Statistics and Computing*, 2007. (page 79)

P. Damien, J. Wakefield, and S. Walker. Gibbs sampling for Bayesian non-conjugate and hierarchical models by using auxiliary variables. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, 61(2):331–344, 1999. (page 40)

S. Della Pietra, V. J. Della Pietra, and J. D. Lafferty. Inducing features of random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(4):380–393, 1997. (page 124)

P. Dellaportas and J. J. Forster. Markov chain Monte Carlo model determination for hierarchical and graphical log-linear models. *Biometrika*, 86(3):615–633, 1999. (pages 124 and 132)

L. Devroye. *Non-uniform random variate generation*. Springer-Verlag, New York, 1986. ISBN 0-387-96305-7. Out of print. Available from `http://cg.scs.carleton.ca/~luc/rnbookindex.html`. (page 22)

A. Dobra, C. Tebaldi, and M. West. Data augmentation in multi-way contingency tables with fixed marginal totals. *Journal of statistical planning and inference*, 136 (2):355–372, 2006. (page 124)

P. Dostert, Y. Efendiev, T. Y. Hou, and W. Luo. Coarse-gradient Langevin algorithms for dynamic data integration and uncertainty quantification. *Journal of Computational Physics*, 217:123–142, 2006. (page 31)

S. Duane, A. D. Kennedy, B. J. Pendleton, and D. Roweth. Hybrid Monte Carlo. *Physics Letters B*, 195(2):216–222, September 1987. (page 41)

D. Edwards and T. Havránek. A fast procedure for model search in multidimensional contingency tables. *Biometrika*, 72(2):339–351, August 1985. (page 132)

R. G. Edwards and A. D. Sokal. Generalizations of the Fortuin-Kasteleyn-Swendsen-Wang representation and Monte Carlo algorithm. *Physical Review*, 38:2009–2012, 1988. (pages 37 and 39)

J. Ferkinghoff-Borg. *Monte Carlo methods in complex systems*. Ph.D. Thesis, Graduate School of Biophysics Niels Bohr Institute and Rise National Laboratory Faculty of Science, University of Copenhagen, May 2002. (page 52)

C. M. Fortuin and P. W. Kasteleyn. On the random-cluster model. I. Introduction and relation to other models. *Physica*, 57:536–564, 1972. (page 37)

D. Frenkel. Speed-up of Monte Carlo simulations by sampling of rejected states. *Proceedings of the National Academy of Sciences*, 101(51):17571–17575, 2004. (pages 34 and 35)

D. Frenkel and B. Smit. *Understanding molecular simulation.* Academic Press Inc., 2nd edition, 2001. (pages 65 and 67)

A. E. Gelfand and A. F. M. Smith. Sampling-based approaches to calculating marginal densities. *Journal of the American Statistical Association*, 85(410):398–409, 1990. (page 33)

A. Gelman and X.-L. Meng. Simulating normalizing constants: from importance sampling to bridge sampling to path sampling. *Statistical Science*, 13(2):163–185, 1998. (pages 86 and 120)

A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin. *Bayesian Data Analysis, Second Edition.* Chapman & Hall/CRC, 2003. ISBN 1-58488-388-X. (page 20)

D. Geman and S. Geman. Stochastic relaxation, Gibbs distribution and Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:721–741, 1984. (pages 30 and 124)

J. Geweke. Getting it right: joint distribution tests of posterior simulators. *Journal of the American Statistical Association*, 99(467):799–804, 2004. (pages 36 and 83)

C. J. Geyer. Markov chain Monte Carlo maximum likelihood. In *Computing Science and Statistics: Proceedings of the 23rd Symposium on the Interface*, pages 156–163, 1991. (page 139)

C. J. Geyer. Practical Markov chain Monte Carlo. *Statistical Science*, 7(4):473–483, November 1992. (page 32)

W. R. Gilks. Derivative-free adaptive rejection sampling for Gibbs sampling. In J. Bernardo, J. Berger, A. P. Dawid, and A. F. M. Smith, editors, *Bayesian Statistics 4*. Oxford University Press, 1992. (page 22)

W. R. Gilks and P. Wild. Adaptive rejection sampling for Gibbs sampling. *Applied Statistics*, 41(2):337–348, 1992. (page 22)

W. R. Gilks, G. O. Roberts, and E. I. George. Adaptive direction sampling. *The Statistician*, 43(1):179–9, 1994. Special Issue: Conference on Practical Bayesian Statistics, 1992 (3). (page 57)

V. K. Gore and M. R. Jerrum. The Swendsen-Wang process does not always mix rapidly. In *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, pages 674–681. ACM Press New York, NY, USA, 1997. (page 117)

V. K. Gore and M. R. Jerrum. The Swendsen–Wang process does not always mix rapidly. *Journal of Statistical Physics*, 97(1):67–86, 1999. (page 117)

P. J. Green. Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika*, 82(4):711–732, December 1995. (pages 55 and 119)

F. Hamze and N. de Freitas. Hot coupling: a particle approach to inference and normalization on pairwise undirected graphs. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems (NIPS*18): Proceedings of the 2005 Conference*. MIT Press, 2006. (page 119)

W. K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1), April 1970. (pages 27, 49, 50, and 55)

D. Heckerman, D. M. Checkering, C. Meek, R. Rounthwaite, and C. Kadie. Dependency networks for inference, collaborative filtering, and data visualization. *Journal of Machine Learning Research (JMLR)*, 1:49–75, 2000. (page 161)

G. E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14(8):1771–1800, August 2002. (pages 124, 127, 131, 132, and 161)

G. E. Hinton, S. Osindero, and Y.-W. Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18:1527–1554, 2006. (page 153)

J. D. Hobby. A user's manual for MetaPost. Technical Report 162, AT&T Bell Laboratories, Murray Hill, New Jersey, 1992. (page 4)

K. S. V. Horn. Constructing a logic of plausible inference: a guide to Cox's theorem. *International Journal of Approximate Reasoning*, 34(1):3–24, 2003. (page 20)

M. Huber. Exact sampling and approximate counting techniques. In *30th ACM Symposium on the Theory of Computing*, pages 31–40, 1998. (page 54)

M. Huber. A bounding chain for Swendsen–Wang. *Random Structures & Algorithms*, 22(1):53–59, 2002. (page 55)

Y. Iba. Population Monte Carlo algorithms. *Transactions of the Japanese Society for Artificial Intelligence*, 16(2):279–286, 2001a. (page 58)

Y. Iba. Extended ensemble Monte Carlo. *International Journal of Modern Physics C*, 12(05):623–656, 2001b. (page 44)

W. Janke and S. Kappler. Multibondic cluster algorithm for Monte Carlo simulations of first-order phase transitions. *Physical Review Letters*, 74(2):212–215, 1995. (page 109)

C. Jarzynski. Equilibrium free-energy differences from nonequilibrium measurements: a master-equation approach. *Physical Review E*, 56(5):5018–5035, November 1997. (page 46)

E. T. Jaynes. *Probability theory: the logic of science: principles and elementary applications vol 1.* Cambridge University Press, April 2003. (page 120)

H. Jeffreys. *Theory of probability.* Oxford University Press, 3rd edition edition, 1961. Republished in Oxford Classic Texts in the Physical Sciences, 1998. (page 82)

M. H. Kalos and P. A. Whitlock. *Monte Carlo methods, volume I: basics.* John Wiley, 1986. ISBN 0-471-89839-2. (page 34)

R. E. Kass and A. E. Raftery. Bayes Factors. *Journal of the American Statistical Association*, 90(430):773–795, 1995. (page 82)

R. E. Kass, B. P. Carlin, A. Gelman, and R. M. Neal. Markov chain Monte Carlo in practice: a roundtable discussion. *American Statistician*, 52(2):93–100, 1998. (page 36)

S. Kirkpatrick, C. D. Gelatt Jr., and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983. (page 42)

M. Kuss and C. E. Rasmussen. Assessing approximate inference for binary Gaussian process classification. *Journal of Machine Learning Research*, 6:1679–1704, 2005. (pages 82 and 104)

S. L. Lauritzen and D. J. Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems (with discussion). *Journal of the Royal Statistical Society, Series B*, 50(2):157–224, 1988. (page 18)

J. S. Liu. Metropolized independent sampling with comparisons to rejection sampling and importance sampling. *Statistics and Computing*, 6(2):113–119, 1996. (page 49)

J. S. Liu. *Monte Carlo strategies in scientific computing.* Springer, 2001. ISBN 0387952306. (page 31)

J. S. Liu, W. H. Wong, and A. Kong. Covariance structure of the Gibbs sampler with applications to the comparisons of estimators and augmentation schemes. *Biometrika*, 81(1):27–40, 1994. (page 33)

J. S. Liu, F. Liang, and W. H. Wong. The use of multiple-try method and local optimization in Metropolis sampling. *Journal of American Statistical Association*, 95(449):121–134, 2000. (pages 60, 62, 64, and 65)

A. P. Lyubartsev, A. A. Martsinovski, S. V. Shevkunov, and P. N. Vorontsov-Velyaminov. New approach to Monte Carlo calculation of the free energy: method of expanded ensembles. *J. Chem. Phys.*, 96(3):1776–1783, 1992. (page 43)

D. MacKay. Nested sampling explanatory illustrations, 2004. Available from `http://www.inference.phy.cam.ac.uk/bayesys/box/nested.pdf`. (page 91)

D. J. C. MacKay. *Information theory, inference, and learning algorithms*. Cambridge University Press, 2003. Available from `http://www.inference.phy.cam.ac.uk/mackay/itila/`. (pages 20, 23, 28, 41, 57, 82, 107, and 130)

E. Marinari and G. Parisi. Simulated tempering: a new Monte Carlo scheme. *Europhysics Letters*, 19(6):451–458, 1992. (page 43)

P. Marjoram, J. Molitor, V. Plagnol, and S. Tavaré. Markov chain Monte Carlo without likelihoods. *Proceedings of the National Academy of Sciences*, 100(26):15324–15328, 2003. (pages 153 and 154)

A. McCallum and B. Wellner. Toward conditional models of identity uncertainty with application to proper noun coreference. In *IJCAI Workshop on Information Integration on the Web*, 2003. (page 124)

I. R. McDonald and K. Singer. Machine calculation of thermodynamic properties of a simple fluid at supercritical temperatures. *J. Chem. Phys.*, 47(11):4766–4772, 1967. (page 88)

N. Metropolis. The beginning of the Monte Carlo method. *Los Alamos Science*, 15: 125–130, 1987. (page 21)

N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. Equation of state calculation by fast computing machines. *J. Chem Phys*, 21:1087, 1953. (pages 3, 28, 55, and 162)

J. Møller, A. N. Pettitt, K. K. Berthelsen, and R. W. Reeves. An efficient Markov chain Monte Carlo method for distributions with intractable normalising constants. Technical Report R-2004-02, Department of Mathematical Sciences, Aalborg University, 2004. (pages 122, 125, 127, 144, 149, 150, 151, 159, and 160)

J. Møller, A. N. Pettitt, R. Reeves, and K. K. Berthelsen. An efficient Markov chain Monte Carlo method for distributions with intractable normalising constants. *Biometrika*, 93(2):451–458, 2006. (pages 122, 125, 144, 146, and 149)

P. Mukherjee, D. Parkinson, and A. R. Liddle. A nested sampling algorithm for cosmological model selection. *The Astrophysical Journal*, 638:L51, 2006. (page 109)

I. Murray and Z. Ghahramani. Bayesian learning in undirected graphical models: approximate MCMC algorithms. In M. Chickering and J. Halpern, editors, *Proceedings of the 20th Annual Conference on Uncertainty in Artificial Intelligence*, pages 392–399. AUAI Press Arlington, Virginia, United States, 2004. (pages 125 and 129)

I. Murray and E. Snelson. A pragmatic Bayesian approach to predictive uncertainty. In J. Quiñonero-Candela, I. Dagan, B. Magnini, and F. D'Alché-Buc, editors, *Machine Learning Challenges. Evaluating Predictive Uncertainty, Visual Object Classification, and Recognising Textual Entailment.: First PASCAL Machine Learning Challenges Workshop, Southampton, UK, April 11–13, 2005, Revised Selected Papers.*, Springer Lecture Notes in Computer Science. 2006. (page 55)

I. Murray, Z. Ghahramani, and D. J. C. MacKay. MCMC for doubly-intractable distributions. In R. Dechter and T. S. Richardson, editors, *Proceedings of the 22nd Annual Conference on Uncertainty in Artificial Intelligence (UAI-06)*, pages 359–366. AUAI Press, 2006a. (page 123)

I. Murray, D. MacKay, Z. Ghahramani, and J. Skilling. Nested sampling for Potts models. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems (NIPS*18): Proceedings of the 2005 Conference*, pages 947–954. MIT Press, 2006b. (pages 89 and 97)

R. M. Neal. Bayesian training of backpropagation networks by the hybrid Monte Carlo method. Technical Report CRG-TR-92-1, Connectionist Research Group, Department of Computer Science, University of Toronto, April 1992. (page 41)

R. M. Neal. Probabilistic inference using Markov chain Monte Carlo methods. Technical Report CRG-TR-93-1, Department of Computer Science, University of Toronto, September 1993. (pages 41, 120, and 129)

R. M. Neal. Sampling from multimodal distributions using tempered transitions. Technical Report 9421, Department of Statistics, University of Toronto, October 1994. (pages 46 and 110)

R. M. Neal. Suppressing random walks in Markov chain Monte Carlo using ordered overrelaxation. Technical Report No. 9508, Department of Statistics, University of Toronto, 1995. (page 68)

R. M. Neal. Sampling from multimodal distributions using tempered transitions. *Statistics and Computing*, 6(4):353–366, 1996a. (pages 46, 47, 48, 110, and 148)

R. M. Neal. *Bayesian learning for neural networks*. Number 118 in Lecture Notes in Statistics. Springer-Verlag, 1996b. (page 41)

R. M. Neal. Markov chain Monte Carlo methods based on 'slicing' the density function. Technical Report 9722, Department of Statistics, University of Toronto, 1997. (page 39)

R. M. Neal. Annealed importance sampling. Technical Report No. 9805, Department of Statistics, University of Toronto, 1998a. (pages 45 and 110)

R. M. Neal. Suppressing random walks in Markov chain Monte Carlo using ordered overrelaxation. In M. I. Jordan, editor, *Learning in graphical models*, pages 205–228. Kluwer Academic Publishers, 1998b. (pages 68 and 69)

R. M. Neal. Markov chain sampling methods for Dirichlet process mixture models. Technical Report 9815, Department of Statistics, University of Toronto, September 1998c. (page 55)

R. M. Neal. Erroneous results in "Marginal likelihood from the Gibbs output", 1999. Available from `http://www.cs.toronto.edu/~radford/chib-letter.html`. (page 84)

R. M. Neal. Annealed importance sampling. *Statistics and Computing*, 11(2):125–139, 2001. (pages 45, 49, 103, and 110)

R. M. Neal. Slice sampling. *Annals of Statistics*, 31(3):705–767, 2003. (pages 39, 40, 59, 79, and 107)

R. M. Neal. Taking bigger Metropolis steps by dragging fast variables. Technical Report No. 0411, Department of Statistics, University of Toronto, October 2004a. (page 144)

R. M. Neal. Improving asymptotic variance of MCMC estimators: non-reversible chains are better. Technical Report 0406, Department of Statistics, University of Toronto, July 2004b. (page 73)

R. M. Neal. Estimating ratios of normalizing constants using linked importance sampling. Technical Report No. 0511, Department of Statistics, University of Toronto, 2005. (pages 119 and 143)

M. E. J. Newman and R. M. Ziff. Fast Monte Carlo algorithm for site or bond percolation. *Physical Review E*, 64(016706), June 2001. (page 37)

M. A. Newton and A. E. Raftery. Approximate Bayesian inference with the weighted likelihood bootstrap. *Journal of the Royal Statistical Society, Series B (Methodological)*, 56(1):3–48, 1994. (page 83)

A. O'Hagan. Monte Carlo is fundamentally unsound. *The Statistician*, 36(2/3):247–249, 1987. In special issue: Practical Bayesian Statistics. (page 120)

O. Papaspiliopoulos and G. O. Roberts. Retrospective MCMC methods for Dirichlet process hierarchical models, 2005. Submitted manuscript. (page 140)

J. Pearl. Evidential reasoning using stochastic simulation of causal models. *Artificial Intelligence*, 32(2):245–257, 1987. (page 34)

P. Peskun. Optimum Monte-Carlo sampling using Markov chains. *Biometrika*, 60(3): 607–612, 1973. (pages 29 and 31)

A. A. Podtelezhnikov, Z. Ghahramani, and D. L. Wild. Learning about protein hydrogen bonding by minimizing contrastive divergence. *Proteins: Structure, Function, and Bioinformatics*, 66:588–599, 2007. (page 161)

J. G. Propp and D. B. Wilson. Exact sampling with coupled Markov chains and applications to statistical mechanics. *Random Structures and Algorithms*, 9(1&2): 223–252, 1996. (pages 52 and 55)

Y. Qi and T. P. Minka. Hessian-based Markov chain Monte-Carlo algorithms. In *First Cape Cod Workshop on Monte Carlo Methods*, September 2002. (pages 65 and 67)

Z. S. Qin and J. S. Liu. Multipoint Metropolis method with application to hybrid Monte Carlo. *Journal of Computational Physics*, 172:827–840, 2001. (page 80)

A. E. Raftery, M. A. Newton, J. M. Satagopan, and P. N. Krivitsky. Estimating the integrated likelihood via posterior simulation using the harmonic mean identity. In J. M. Bernardo, M. J. Bayarri, J. O. Berger, A. P. Dawid, D. Heckerman, A. F. M. Smith, and M. West, editors, *Bayesian Statistics 8, Proceedings of the Valencia / ISBA 8th World Meeting on Bayesian Statistics*. Oxford University Press, 2007. (page 83)

C. E. Rasmussen and Z. Ghahramani. Bayesian Monte Carlo. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems (NIPS*15): Proceedings of the 2002 Conference*. MIT Press, 2003. (page 120)

C. Ritter and M. A. Tanner. Facilitating the Gibbs sampler: the Gibbs stopper and the griddy-Gibbs sampler. *Journal of the American Statistical Association*, 87(419): 861–868, 1992. (page 30)

C. P. Robert and G. Casella. *Monte Carlo statistical methods*. Springer Texts in Statistics. Springer, 2nd edition, 2004. ISBN 0-387-21239-6. (pages 58, 59, and 61)

M. Seeger. Low rank updates for the Cholesky decomposition. Technical report, September 2005. `http://www.kyb.tuebingen.mpg.de/bs/people/seeger/papers/cholupdate.pdf`. (page 18)

J. Seward and N. Nethercote. Using valgrind to detect undefined value errors with bit-precision. In *Proceedings of the USENIX'05 Annual Technical Conference*, April 2005. (page 4)

R. Shaw, M. Bridges, and M. P. Hobson. Clustered nested sampling: efficient Bayesian inference for cosmology, 2007. Preprint available from `http://arxiv.org/pdf/astro-ph/0701867`. (pages 109, 116, 118, and 119)

D. S. Sivia and J. Skilling. *Data analysis: a Bayesian tutorial*. Oxford Science Publications, 2006. ISBN 0-19-856832-0. (page 20)

J. Skilling. Nested sampling. In R. Fischer, R. Preuss, and U. von Toussaint, editors, *Bayesian inference and maximum entropy methods in science and engineering*, AIP Conference Proceeedings 735, pages 395–405, 2004. (page 88)

J. Skilling. Nested sampling for general Bayesian computation. *Bayesian Analysis*, 1 (4):833–860, 2006. (pages 88 and 96)

J. Skilling. Nested sampling for Bayesian computations. In J. M. Bernardo, M. J. Bayarri, J. O. Berger, A. P. Dawid, D. Heckerman, A. F. M. Smith, and M. West, editors, *Bayesian Statistics 8, Proceedings of the Valencia / ISBA 8th World Meeting on Bayesian Statistics*. Oxford University Press, 2007. (pages 88 and 97)

J. Skilling and D. J. C. MacKay. Slice sampling — a binary implementation. *Annals of Statistics*, 31(3), 2003. In discussion: Slice Sampling, Radford M. Neal. (page 40)

G. R. Smith. *The measurement of free energy by Monte Carlo computer simulation*. PhD thesis, University of Edinburgh, September 1995. (pages 52 and 121)

A. Sokal. Monte Carlo methods in statistical mechanics: foundations and new algorithms, 1996. Lectures at the Cargèse Summer School on "Functional Integration: Basics and Applications". (page 21)

D. J. Spiegelhalter, A. Thomas, N. G. Best, and W. R. Gilks. *BUGS examples volumes 1 and 2*, 1996. Available from `http://www.mrc-bsu.cam.ac.uk/bugs/`. (page 30)

D. Stern, T. Graepel, and D. J. C. MacKay. Modelling uncertainty in the game of go. In L. K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems (NIPS*17): Proceedings of the 2004 Conference*. MIT Press, 2005. (page 37)

K. Stormark. Multiple proposal strategies for Markov chain Monte Carlo. MSc thesis, Department of Mathematical Sciences, Norwegian University of Science and Technology, 2006. (page 79)

R. H. Swendsen and J.-S. Wang. Replica Monte Carlo simulation of spin-glasses. *Physical Review Letters*, 57(21):2607–2609, 1986. (page 139)

R. H. Swendsen and J. S. Wang. Nonuniversal critical dynamics in Monte Carlo simulations. *Physical Review Letters*, 58(2):86–88, 1987. (pages 36 and 37)

C. J. F. Ter Braak. A Markov chain Monte Carlo version of the genetic algorithm differential evolution: easy Bayesian computing for real parameter spaces. *Statistical Computing*, pages 239–249, 2006. (page 58)

L. Tierney. Markov chains for exploring posterior distributions. *The Annals of Statistics*, 22(4):1701–1728, 1994. (page 25)

L. Tierney and A. Mira. Some adaptive Monte Carlo methods for Bayesian inference. *Statistics in Medicine*, 18(17–18):2507–2515, 1999. (page 59)

H. Tjelmeland. Using all Metropolis–Hastings proposals to estimate mean values. Technical Report 4/2004, Department of Mathematical Sciences, Norwegian University of Science and Technology, 2004. (pages 34 and 79)

Z. Tu and S.-C. Zhu. Image segmentation by data-driven Markov chain Monte Carlo. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5):657–673, 2002. (page 29)

J.-S. Wang and R. H. Swendsen. Low-temperature properties of the ±J Ising spin glass in two dimensions. *Physical Review B*, pages 4840–4844, September 1988. (page 45)

L. Wang, J. Liu, and S. Z. Li. MRF parameter estimation by MCMC method. *Pattern recognition*, 33:1919–1925, 2000. (page 131)

G. R. Warnes. *The normal kernel coupler: an adaptive MCMC method for efficiently sampling from multi-modal distributions*. PhD thesis, Department of Biostatistics, University of Washington, 2000. (page 59)

M. Welling and S. Parise. Bayesian random fields: the Bethe-Laplace approximation. In *Proceedings of the 22nd Annual Conference on Uncertainty in Artificial Intelligence (UAI-06)*. AUAI Press, 2006. (page 127)

W. Wiegerinck. Variational approximations between mean field theory and the junction tree algorithm. In *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence*, pages 626–633. Morgan Kaufmann, 2000. (page 130)

D. B. Wilson. Annotated bibliography of perfectly random sampling with Markov chains. In *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, volume 41. American Mathematical Society, 1998. Updated version available online `http://dbwilson.com/exact/`. (page 52)

J. Winn and C. M. Bishop. Variational message passing. *Journal of Machine Learning Research*, 6:661–694, 2005. (page 130)

O. Winther and A. Krogh. Teaching computers to fold proteins. *Physical Review E*, 70 (3):30903, 2004. (page 124)

C. Yanover and Y. Weiss. Approximate inference and protein folding. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems (NIPS*15): Proceedings of the 2002 Conference*. MIT Press, 2003. (page 124)

J. S. Yedidia, W. T. Freeman, and Y. Weiss. Constructing free-energy approximations and generalized belief propagation algorithms. *IEEE Transactions on Information Theory*, 51(7):2282–2312, July 2005. (page 130)

Y. Yu and Q. Cheng. MRF parameter estimation by an accelerated method. *Pattern Recognition Letters*, 24:1251–1259, 2003. (page 131)

X. Zhu and Z. Ghahramani. Towards semi-supervised classification with Markov random fields. Technical Report CMU-CALD-02-106, School of Computer Science, Carnegie Mellon University, June 2002. (pages 124, 134, 135, and 136)