Probabilistic Inference Using Markov Chain Monte Carlo Methods

Radford M. Neal

Technical Report CRG-TR-93-1
Department of Computer Science
University of Toronto

E-mail: radford@cs.toronto.edu

25 September 1993

Abstract

Probabilistic inference is an attractive approach to uncertain reasoning and empirical learning in artificial intelligence. Computational difficulties arise, however, because probabilistic models with the necessary realism and flexibility lead to complex distributions over high-dimensional spaces.

Related problems in other fields have been tackled using Monte Carlo methods based on sampling using Markov chains, providing a rich array of techniques that can be applied to problems in artificial intelligence. The "Metropolis algorithm" has been used to solve difficult problems in statistical physics for over forty years, and, in the last few years, the related method of "Gibbs sampling" has been applied to problems of statistical inference. Concurrently, an alternative method for solving problems in statistical physics by means of dynamical simulation has been developed as well, and has recently been unified with the Metropolis algorithm to produce the "hybrid Monte Carlo" method. In computer science, Markov chain sampling is the basis of the heuristic optimization technique of "simulated annealing", and has recently been used in randomized algorithms for approximate counting of large sets.

In this review, I outline the role of probabilistic inference in artificial intelligence, present the theory of Markov chains, and describe various Markov chain Monte Carlo algorithms, along with a number of supporting techniques. I try to present a comprehensive picture of the range of methods that have been developed, including techniques from the varied literature that have not yet seen wide application in artificial intelligence, but which appear relevant. As illustrative examples, I use the problems of probabilistic inference in expert systems, discovery of latent classes from data, and Bayesian learning for neural networks.

Acknowledgements

I thank David MacKay, Richard Mann, Chris Williams, and the members of my Ph.D committee, Geoffrey Hinton, Rudi Mathon, Demetri Terzopoulos, and Rob Tibshirani, for their helpful comments on this review. This work was supported by the Natural Sciences and Engineering Research Council of Canada and by the Ontario Information Technology Research Centre.

Contents

Index to Examples

Type of model	*Definition*	*Statistical Inference*	*Gibbs Sampling*	*Metropolis Algorithm*	*Stochastic Dynamics*	*Hybrid Monte Carlo*
Gaussian distribution	9	15, 19	64		83, 84	
Latent class model	10	21	51	POS	POS	POS
Belief network	10	*	50	POS	NA	NA
Multi-layer perceptron	12	16, 19, 22, 35	INF	58	77	81
2D Ising model	26	NA	49	57	NA	NA
Lennard-Jonesium	27	NA	INF	57	76	POS

NA – Not applicable, INF – Probably infeasible, POS – Possible, but not discussed

* Statistical inference for the parameters of belief networks is quite possible, but this review deals only with inference for the values of discrete variables in the network.

1. Introduction

Probability is a well-understood method of representing uncertain knowledge and reasoning to uncertain conclusions. It is applicable to low-level tasks such as perception, and to high-level tasks such as planning. In the Bayesian framework, learning the probabilistic models needed for such tasks from empirical data is also considered a problem of probabilistic inference, in a larger space that encompasses various possible models and their parameter values. To tackle the complex problems that arise in artificial intelligence, flexible methods for formulating models are needed. Techniques that have been found useful include the specification of dependencies using "belief networks", approximation of functions using "neural networks", the introduction of unobservable "latent variables", and the hierarchical formulation of models using "hyperparameters".

Such flexible models come with a price however. The probability distributions they give rise to can be very complex, with probabilities varying greatly over a high-dimensional space. There may be no way to usefully characterize such distributions analytically. Often, however, a sample of points drawn from such a distribution can provide a satisfactory picture of it.

In particular, from such a sample we can obtain *Monte Carlo* estimates for the expectations of various functions of the variables. Suppose $X = \{X_1, \ldots, X_n\}$ is the set of random variables that characterize the situation being modeled, taking on values usually written as $x_1, \ldots, x_n$, or some typographical variation thereon. These variables might, for example, represent parameters of the model, hidden features of the objects modeled, or features of objects that may be observed in the future. The expectation of a function $a(X_1, \ldots, X_n)$ — it's average value with respect to the distribution over X — can be approximated by

$$\langle a \rangle = \sum_{\tilde{x}_1} \cdots \sum_{\tilde{x}_n} a(\tilde{x}_1, \ldots, \tilde{x}_n)\, P(X_1 = \tilde{x}_1, \ldots, X_n = \tilde{x}_n) \tag{1.1}$$

$$\approx \frac{1}{N} \sum_{t=0}^{N-1} a(x_1^{(t)}, \ldots, x_n^{(t)}) \tag{1.2}$$

where $x_1^{(t)}, \ldots, x_n^{(t)}$ are the values for the t-th point in a sample of size N. (As above, I will often distinguish variables in summations using tildes.) Problems of prediction and decision can generally be formulated in terms of finding such expectations.

Generating samples from the complex distributions encountered in artificial intelligence applications is often not easy, however. Typically, most of the probability is concentrated in regions whose volume is a tiny fraction of the total. To generate points drawn from the distribution with reasonable efficiency, the sampling procedure must search for these relevant regions. It must do so, moreover, in a fashion that does not bias the results.

Sampling methods based on *Markov chains* incorporate the required search aspect in a framework where it can be proved that the correct distribution is generated, at least in the limit as the length of the chain grows. Writing $X^{(t)} = \{X_1^{(t)}, \ldots, X_n^{(t)}\}$ for the set of variables at step t, the chain is defined by giving an initial distribution for $X^{(0)}$ and the transition probabilities for $X^{(t)}$ given the value for $X^{(t-1)}$. These probabilities are chosen so that the distribution of $X^{(t)}$ converges to that for X as t increases, and so that the Markov chain can feasibly be simulated by sampling from the initial distribution and then, in succession, from the conditional transition distributions. For a sufficiently long chain, equation (1.2) can then be used to estimate expectations.

Typically, the Markov chain explores the space in a "local" fashion. In some methods, for example, $x^{(t)}$ differs from $x^{(t-1)}$ in only one component of the state — e.g.. it may differ with respect to x_i, for some i, but have $x_j^{(t)} = x_j^{(t-1)}$ for $j \neq i$. Other methods may change all components at once, but usually by only a small amount. Locality is often crucial to the feasibility of these methods. In the Markov chain framework, it is possible to guarantee that such step-by-step local methods eventually produce a sample of points from the globally-correct distribution.

My purpose in this review is to present the various realizations of this basic concept that have been developed, and to relate these methods to problems of probabilistic reasoning and empirical learning in artificial intelligence. I will be particularly concerned with the potential for Markov chain Monte Carlo methods to provide computationally feasible implementations of Bayesian inference and learning. In my view, the Bayesian approach provides a flexible framework for representing the intricate nature of the world and our knowledge of it, and the Monte Carlo methods I will discuss provide a correspondingly flexible mechanism for inference within this framework.

Historical development. Sampling methods based on Markov chains were first developed for applications in statistical physics. Two threads of development were begun forty years ago. The paper of Metropolis, *et al* (4:1953) introduced what is now known as the *Metropolis algorithm*, in which the next state in the Markov chain is chosen by considering a (usually small) change to the state, and accepting or rejecting this change based on how the probability of the altered state compares to that of the current state. Around the same time, Alder and Wainwright (5:1959) developed the "molecular dynamics" method, in which new states are found by simulating the dynamical evolution of the system. I will refer to this technique as the *dynamical* method, since it can in fact be used to sample from any differentiable probability distribution, not just distributions for systems of molecules. Recently, these two threads have been united in the *hybrid Monte Carlo* method of Duane, Kennedy, Pendleton, and Roweth (5:1987), and several other promising approaches have also been explored.

A technique based on the Metropolis algorithm known as *simulated annealing* has been widely applied to optimization problems since a paper of Kirkpatrick, Gelatt, and Vecchi (6:1983). Work in this area is of some relevance to the sampling problems discussed in this review. It also relates to one approach to solving the difficult statistical physics problem of *free energy* estimation, which is equivalent to the problem of comparing different models in the Bayesian framework. The Metropolis algorithm has also been used in algorithms for approximate counting of large sets (see (Aldous, 7:1993) for a review), a problem that can also be seen as a special case of free energy estimation.

Interest in Markov chain sampling methods for applications in probability and statistics has recently become widespread. A paper by Geman and Geman (4:1984) applying such methods to image restoration has been influential. More recently, a paper by Gelfand and Smith (4:1990) has sparked numerous applications to Bayesian statistical inference. Work in these areas has up to now relied almost exclusively on the method of *Gibbs sampling*, but other methods from the physics literature should be applicable as well.

Probability has been applied to problems in artificial intelligence from time to time over the years, and underlies much related work in pattern recognition (see, for example, (Duda and Hart, 2:1973)). Only recently, however, has probabilistic inference become prominent, a development illustrated by the books of Pearl (2:1988), concentrating on methods applicable to expert systems and other high-level reasoning tasks, and of Szeliksi (2:1989), on low-level vision. Much of the recent work on "neural networks", such as that described by Rumelhart, McClelland, and the PDP Research Group (2:1986), can also be regarded as statistical inference for probabilistic models.

Applications in artificial intelligence of Markov chain Monte Carlo methods could be said to have begun with the work on optimization using simulated annealing. This was followed by the work on computer vision of Geman and Geman (4:1984) mentioned above, along with work on the "Boltzmann machine" neural network of Ackley, Hinton, and Sejnowski (2:1985). In the Boltzmann machine, the Gibbs sampling procedure is used to make inferences relating to a particular situation, and also when learning appropriate network parameters from empirical data, within the maximum likelihood framework. Pearl (4:1987) introduced Gibbs sampling for "belief networks", which are used to represent expert knowledge of a probabilistic form. I have applied Gibbs sampling to maximum likelihood learning of belief networks (Neal, 2:1992b).

True Bayesian approaches to learning in an artificial intelligence context have been investigated only recently. Spiegelhalter and Lauritzen (2:1990), Hanson, Stutz, and Cheeseman (2:1991), MacKay (2:1991, 2:1992b), Buntine and Weigend (2:1991), and Buntine (2:1992) have done interesting work using methods other than Monte Carlo. I have applied Markov chain Monte Carlo methods to some of the same problems (Neal, 2:1992a, 2:1992c, 2:1993a).

Though these applications to problems in artificial intelligence are still in their infancy, I believe the Markov chain Monte Carlo approach has great potential as a widely applicable computational strategy, which is particularly relevant when problems are formulated in the Bayesian framework.

Outline of this review. In Section 2, which follows, I discuss probabilistic inference and its applications in artificial intelligence. This topic can be divided into inference using a specified model, and statistical inference concerning the model itself. In both areas, I indicate where computational problems arise for which Monte Carlo methods may be appropriate. I also present some basic concepts of statistical physics which are needed to understand the algorithms drawn from that field. This section also introduces a number of running examples that will be used to illustrate the concepts and algorithms.

In Section 3, I define more precisely the class of problems for which use of Monte Carlo methods based on Markov chains is appropriate, and discuss why these problems are difficult to solve by other methods. I also present the basics of the theory of Markov chains, and discuss recently developed theoretical techniques that may allow useful analytical results to be derived for the complex chains encountered in practical work.

Section 4 begins the discussion of the algorithms themselves by presenting the Metropolis, Gibbs sampling, and related algorithms. These most directly implement the idea of sampling using Markov chains, and are applicable to the widest variety of systems. Section 5 then discusses the dynamical and hybrid Monte Carlo algorithms, which are applicable to continuous systems in which appropriate derivatives can be calculated. Section 6 reviews simulated annealing, free energy estimation, techniques for assessing and improving the accuracy of estimates, and the potential for parallel implementations. These topics apply to all the algorithms discussed.

I conclude in Section 7 by discussing possibilities for future research concerning Markov chain Monte Carlo algorithms and their applications. Finally, I have included a comprehensive, though hardly exhaustive, bibliography of work in the area.

2. Probabilistic Inference for Artificial Intelligence

Probability and statistics provide a basis for addressing two crucial problems in artificial intelligence — how to *reason in the presense of uncertainty*, and how to *learn from experience.*

This statement is, of course, controversial. Many workers in artificial intelligence have argued that probability is an inappropriate, or at least an incomplete, mechanism for representing the sort of uncertainty encountered in everyday life. Much work in machine learning is based on non-statistical approaches. Some of the arguments concerning these issues may be seen in a paper by Cheeseman (2:1988) and the accompanying discussion. A book by Pearl (2:1988) is a detailed development and defence of the use of probability as a representation of uncertainty. In this review, I will take it as given that the application of probability and statistics to problems in artificial intelligence is of sufficient interest to warrant investigating the computational problems that these applications entail.

The role of probabilistic inference in artificial intelligence relates to long-standing controversies concerning the interpretation of probability and the proper approach to statistical inference. (Barnett (9:1982) gives a balanced presentation of the various views.) In the *frequency* interpretation, a probability represents the long-run frequency of an event in a repeatable experiment. It is meaningless with this interpretation to use probabilities in reference to unique events. We cannot, for example, ask what is the probability that Alexander the Great played the flute, or that grandmother would enjoy receiving a cribbage set as a gift. Such questions do make sense if we adopt the *degree of belief* interpretation, under which a probability represents the degree to which we believe that the given evidence, together with our prior opinions as to what is reasonable, warrant belief in the proposition in question. This interpretation of probability is natural for applications in artificial intelligence.

Linked to these different views of probability are different views on how to find probabilistic models from empirical data — in artificial intelligence terms, how to learn from experience. As a simple example, suppose that we have flipped a coin ten times, and observed that eight of these times it landed head-up. How can we use this data to develop a probabilistic model of how the coin lands? The frequency interpretation of probability views the parameters of the model — in this case, just the "true" probability that the coin will land head-up — as fixed constants, about which it makes no sense to speak in terms of probability. These constants can be estimated by various *frequentist* statistical procedures. We might, for example, employ a procedure that estimates the probability of heads to be the observed frequency, 8/10 in the case at hand, though this is by no means the only possible, or reasonable, procedure.

The degree of belief interpretation of probability leads instead to a *Bayesian* approach to statistical inference. In this framework, uncertainty concerning the parameters of the model is expressed by means of a probability distribution over the possible parameter values. This distribution is updated using *Bayes' rule* as new information arrives. Mathematically, this is a process of probabilistic inference similar to that used to deal with a particular case using a fully specified model, a fact which we will see is very convenient computationally. In the coin-tossing example above, a typical Bayesian inference procedure would produce a probability distribution for the "true probability of heads" in which values around 8/10 are more likely than those around, say, 1/10.

In this review, I will be primarily concerned with models where probability is interpreted as a degree of belief, and with statistical inference for such models using the Bayesian approach,

x_1	x_2	x_3	$P(x_1, x_2, x_3)$
CLEAR	RISING	DRY	0.40
CLEAR	RISING	WET	0.07
CLEAR	FALLING	DRY	0.08
CLEAR	FALLING	WET	0.10
CLOUDY	RISING	DRY	0.09
CLOUDY	RISING	WET	0.11
CLOUDY	FALLING	DRY	0.03
CLOUDY	FALLING	WET	0.12

X_1 = Sky clear or cloudy in the morning
X_2 = Barometer rising or falling in the morning
X_3 = Dry or wet in the afternoon

Figure 2.1: The joint distribution for a model of the day's weather.

but the algorithms described are also applicable to many problems that can be formulated in frequentist terms. A number of texts on probability and statistics adopt the Bayesian approach; I will here mention only the introductory books of Schmitt (9:1969) and Press (9:1989), and the more advanced works of DeGroot (9:1970), Box and Tiao (9:1973), and Berger (9:1985). Unfortunately, none of these are ideal as an introduction to the subject for workers in artificial intelligence. Pearl (2:1988) discusses Bayesian probabilistic inference from this viewpoint, but has little material on statistical inference or empirical learning.

2.1 Probabilistic inference with a fully-specified model

The least controversial applications of probability are to situations where we have accurate estimates of all relevant probabilities based on numerous observations of closely parallel cases. The frequency interpretation of probability is then clearly applicable. The degree of belief interpretation is not excluded, but for reasonable people the degree of belief in an event's occurrence will be very close to its previously-observed frequency.

In this section, I discuss how such probabilistic models are formulated, and how they are used for inference. Mathematically, this material applies also to models for unrepeatable situations where the probabilities are derived entirely from subjective assessments — the crucial point is that however the model was obtained, it is considered here to be specified fully and with certainty.

Joint, marginal, and conditional probabilities. A fully-specified probabilistic model gives the *joint probability* for every conceivable combination of values for the variables used to characterize some situation of interest. Let these random variables be $X_1, \ldots, X_n$, and, for the moment, assume that each takes on values from some discrete set. The model is then specified by the values of the joint probabilities, $P(X_1 = x_1, \ldots, X_n = x_n)$, for every possible assignment of values, $x_1, \ldots, x_n$, to the variables. I will generally abbreviate such notation to $P(x_1, \ldots, x_n)$ when the random variables involved are clear from the names of the arguments. Figure 2.1 gives the joint distribution for a model of the day's weather.

From these joint probabilities, *marginal probabilities* for subsets of the variables can be found by summing over all possible combinations of values for the other variables. For example, if only the variables $X_1, \ldots, X_m$ are relevant to our problem, we would be interested in the marginal probabilities

$$P(x_1, \ldots, x_m) \;=\; \sum_{\tilde{x}_{m+1}} \cdots \sum_{\tilde{x}_n} P(x_1, \ldots, x_m, \tilde{x}_{m+1}, \ldots, \tilde{x}_n) \tag{2.1}$$

I will often use notation in which the above formula can be written instead as follows, where $\mathcal{A} = \{1, \ldots, m\}$ and $\mathcal{B} = \{m+1, \ldots, n\}$:

$$P(\{x_i : i \in \mathcal{A}\}) \;=\; \sum_{\{\tilde{x}_j : j \in \mathcal{B}\}} P(\{x_i : i \in \mathcal{A}\}, \{\tilde{x}_j : j \in \mathcal{B}\}) \tag{2.2}$$

That is, when "$\{x_i : i \in \mathcal{A}\}$" occurs in a probability statement, it is equivalent to listing all the x_i for $i \in \mathcal{A}$, and when such an expression occurs in a summation, it represents a multiple summation over all possible combinations of values for these variables.

From the joint distribution of Figure 2.1 we can calculate that $P(\text{CLEAR, RISING}) = 0.47$ and $P(\text{CLOUDY}) = 0.35$.

Conditional probabilities for one subset of variables, X_i for $i \in \mathcal{A}$, given values for another (disjoint) subset, X_j for $j \in \mathcal{B}$, are defined as ratios of marginal probabilities:

$$P(\{x_i : i \in \mathcal{A}\} \mid \{x_j : j \in \mathcal{B}\}) \;=\; \frac{P(\{x_i : i \in \mathcal{A}\}, \{x_j : j \in \mathcal{B}\})}{P(\{x_j : j \in \mathcal{B}\})} \tag{2.3}$$

For the example of Figure 2.1, we can calculate that

$$P(\text{DRY} \mid \text{CLEAR, RISING}) \;=\; \frac{P(\text{DRY, CLEAR, RISING})}{P(\text{CLEAR, RISING})} \;=\; \frac{0.40}{0.47} \;\approx\; 0.85$$

Seen as a statement about long-run frequencies, this says that of all those mornings with clear sky and rising barometer, the proportion which were followed by a dry afternoon was about 0.85. Interpreted in terms of degree of belief, it says that if we know that the sky is clear and the barometer is rising in the morning, and we know nothing else of relevance, then the degree of belief we should have that the weather will be dry in the afternoon is about 0.85.

The *expected value* or *expectation* of a function of a random variable is its average value with respect to the probability distribution in question. The expectation of $a(X)$, written as $\langle a \rangle$ or $E[a(X)]$, is defined as

$$\langle a \rangle \;=\; E[a(X)] \;=\; \sum_{\tilde{x}} a(\tilde{x}) P(\tilde{x}) \tag{2.4}$$

One can also talk about the expectation of a function conditional on certain variables taking on certain values, in which case $P(\tilde{x})$ above is replaced by the appropriate conditional probability.

In this review, I will usually use $\langle \cdot \rangle$ to denote expectation for variables that we are really interested in, and $E[\,\cdot\,]$ to denote expectation for variables that are part of a Monte Carlo procedure for estimating the interesting expectations.

Probability for continuous variables. Models in which the variables take on continuous values are common as well. The model is then specified by giving the joint probability density function for all the variables, which I will write using the same notation as for probabilities, trusting context to make clear my meaning. The probability density function for X is defined to be such that $\int_A P(x)\,dx$ is the probability that X lies in the region A. Marginal densities are obtained as in equation (2.2), but with the summations replaced by integrations. Conditional densities are defined as in equation (2.3). Expectations are also defined as for discrete variables, but with integrals replacing summations.

A model may contain both discrete variables are continuous variables. In such cases, I will again use the notation $P(x_1, \ldots, x_n)$, referring in this case to numbers that are hybrids of probabilities and probability densities.

Not all distributions on continuous spaces can be specified using probability densities of the usual sort, because some distributions put a non-zero probability mass in an infinitesimal region of the space, which would correspond to an infinite probability density. In this review, I will handle cases of this sort using the *delta function*, $\delta(x, y)$, which is defined for real arguments by the property that for any continuous function, $f(\,\cdot\,)$:

$$\int_{-\infty}^{+\infty} f(\tilde{x})\,\delta(\tilde{x}, y)\,d\tilde{x} \;=\; \int_{-\infty}^{+\infty} f(\tilde{x})\,\delta(y, \tilde{x})\,d\tilde{x} \;=\; f(y) \tag{2.5}$$

Clearly, no actual real-valued function has these properties, and hence expressions containing delta functions must be manipulated with care to avoid fallacies — to be valid, they must be true when $\delta(x, y)$ is regarded as the limit of actual real-valued functions that are ever more peaked about the line $x = y$ and zero away from it.

As an example, if a variable, X, with range $(0, 1)$ has the probability density given by $P(x) = (1/2) + (1/2)\delta(1/3, x)$, then its distribution has half of the total probability spread uniformly over $(0, 1)$ and half concentrated at the point $1/3$.

For discrete x and y, it will be convenient to define $\delta(x, y)$ to be zero if $x \neq y$ and one if $x = y$. This allows some formulas to be written that apply regardless of whether the variables are continuous or discrete. The following analogue of equation (2.5) holds:

$$\sum_{\tilde{x}} f(\tilde{x})\,\delta(\tilde{x}, y) \;=\; \sum_{\tilde{x}} f(\tilde{x})\,\delta(y, \tilde{x}) \;=\; f(y) \tag{2.6}$$

Conditioning on observed data. Typically, we are interested in conditioning with respect to the variables whose values we know from observation, a process that can be viewed as updating our beliefs in the light of new information. We can then use these conditional probabilities in making decisions.

The model is generally specified in such a way that calculation of the full joint probabilities is feasible, except perhaps for an unknown normalizing factor required to make them sum to one. If we could calculate marginal probabilities easily (even up to an unknown factor) we could use equation (2.3) to calculate conditional probabilities (with any unknown factor cancelling out). Alternatively, from the conditional probabilities for all combinations of values for the unobserved variables, given the observed values, we could find the conditional probabilities for just the variables we are interested in by summing over the others, as follows:

$$\begin{aligned} & P(\{x_i : i \in \mathcal{A}\} \mid \{x_j : j \in \mathcal{B}\}) \\ & \quad = \sum_{\{\tilde{x}_k \,:\, k \in \mathcal{C}\}} P(\{x_i : i \in \mathcal{A}\},\, \{\tilde{x}_k : k \in \mathcal{C}\} \mid \{x_j : j \in \mathcal{B}\}) \end{aligned} \tag{2.7}$$

In the weather example, suppose that we observe the sky to be cloudy in the morning, but that we have no barometer. We wish to calculate the probability that it will be wet in the afternoon, and intend to go on a picnic if this probability is no more than 0.3. Here, we must sum over the possible values for X_2, the barometer reading. The calculation may be made as follows:

$$\begin{aligned} P(\text{WET} \,|\, \text{CLOUDY}) &= P(\text{WET}, \text{RISING} \,|\, \text{CLOUDY}) + P(\text{WET}, \text{FALLING} \,|\, \text{CLOUDY}) \\ &= 0.11/0.35 \;+\; 0.12/0.35 \;\approx\; 0.66 \end{aligned}$$

We decide not to go for a picnic.

Unfortunately, neither the calculation of a marginal probability using equation (2.2) nor the calculation for a conditional probability of equation (2.7) are feasible in more complex situations, since they require time exponential in the number of variables summed over.

Such sums (or integrals) can, however, be estimated by Monte Carlo methods. Another way to express the conditional probability of equations (2.3) and (2.7) is

$$\begin{aligned} &P(\{x_i : i \in \mathcal{A}\} \mid \{x_j : j \in \mathcal{B}\}) \;=\; \\ &\qquad \sum_{\{\tilde{x}_i : i \in \mathcal{A}\}} \sum_{\{\tilde{x}_k : k \in \mathcal{C}\}} P(\{\tilde{x}_i : i \in \mathcal{A}\}, \{\tilde{x}_k : k \in \mathcal{C}\} \mid \{x_j : j \in \mathcal{B}\}) \cdot \prod_{i \in \mathcal{A}} \delta(x_i, \tilde{x}_i) \end{aligned} \tag{2.8}$$

This expression has the form of an expectation with respect to the distribution for the X_i and X_k conditional on the known values of the X_j. For discrete X_i, it can be evaluated using the Monte Carlo estimation formula of equation (1.2), provided we can obtain a sample from this conditional distribution. This procedure amounts to simply counting how often the particular combination of X_i values that we are interested in appears in the sample, and using this observed frequency as our estimate of the corresponding conditional probability.

Here is yet another expression for the conditional probability:

$$\begin{aligned} &P(\{x_i : i \in \mathcal{A}\} \mid \{x_j : j \in \mathcal{B}\}) \;=\; \\ &\qquad \sum_{\{\tilde{x}_k : k \in \mathcal{C}\}} P(\{\tilde{x}_k : k \in \mathcal{C}\} \mid \{x_j : j \in \mathcal{B}\})\; P(\{x_i : i \in \mathcal{A}\} \mid \{\tilde{x}_k : k \in \mathcal{C}\}, \{x_j : j \in \mathcal{B}) \end{aligned} \tag{2.9}$$

This, too, has the form of an expectation with respect to the distribution conditional on the known values of the X_j. In Monte Carlo estimation based on this formula, rather than count how often the values for X_i we are interested in show up, we average the conditional probabilities, or probability densities, for those X_i. This method still works for real-valued X_i, for which we would generally never see an exact match with any particular x_i.

Typically, we will wish to estimate these conditional probabilities to within some absolute error tolerance. In the picnicking example, for instance, we may wish to know the probability of rain to within ± 0.01. If we are interested in rare but important events, however, a relative error tolerance will be more appropriate. For example, in computing the probability of a nuclear reactor meltdown, the difference between the probabilities 0.3 and 0.4 is not significant, since neither is acceptable, but the difference between a probability of 10^{-5} and one of 10^{-6} may be quite important.

Model specification. When the number of variables characterizing a situation is at all large, describing their joint distribution by explicitly giving the probability for each combination of values is infeasible, due to the large number of parameters required. In some simple cases a more parsimonious specification is easily found. For variables that are *independent*, for example, $P(x_1, \ldots, x_n) = P(x_1) \cdots P(x_n)$ and the distribution can be specified by giving the values of $P(x_i)$ for all i and x_i.

In more complex cases, it can be a challenge to find a model that captures the structure of the problem in a compact and computationally tractable form. *Latent* (or *hidden*) variables are often useful in this regard (see, for example, (Everitt, 9:1984)). These variables are not directly observable, and perhaps do not even represent objectively identifiable attributes of the situation, but they do permit the probabilities for the *observable* (or *visible*) variables to be easily specified as a marginal distribution, with the latent variables summed over. In addition to their practical utility, these models have interest for artificial intelligence because the latent variables can sometimes be viewed as abstract features or concepts.

Models are often characterized as either *parametric* or *non-parametric*. These terms are to some degree misnomers, since all models have parameters of some sort or other. The distinguishing characteristic of a "non-parametric" model is that these parameters are sufficiently numerous, and employed in a sufficiently flexible fashion, that they can be used to approximate any of a wide class of distributions. The parameters also do not necessarily represent any meaningful aspects of reality. In contrast, a parametric model will generally be capable of representing only a narrow class of distributions, and its parameters will often have physical interpretations.

By their nature, non-parametric models are virtually never specified in detail by hand. They are instead learned more or less automatically from training data. In contrast, a parametric model with physically meaningful parameters might sometimes be specified in full by a knowledgeable expert.

We can also distinguish between models that define a joint probability distribution for all observable variables and those that define only the conditional distributions for some variables given values for some other variables (or even just some characteristics of these conditional distributions, such as the expected value). The latter are sometimes referred to as *regression* or *classification* models, depending on whether the variables whose conditional distributions they model are continuous or discrete.

Example: Gaussian distributions. The *Gaussian* or *Normal* distribution is the archetype of a parametric probability distribution on a continuous space. It is extremely popular, and will be used later as a model system for demonstrating the characteristics of Markov chain Monte Carlo algorithms.

The univariate Gaussian distribution for a real variable, X, has the following probability density function:

$$P(x) \;=\; \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-(x-\mu)^2/2\sigma^2\right) \tag{2.10}$$

Here, μ and σ^2 are the parameters of the distribution, with μ being the *mean* of the distribution, equal to $\langle x \rangle$, and σ^2 being the *variance*, equal to $\langle (x - \langle x \rangle)^2 \rangle$. The square root of the variance is the *standard deviation*, given by σ.

The multivariate generalization of the Gaussian distribution, for a vector X, of dimensionality n, has probability density

$$P(x) \;=\; (2\pi)^{-n/2}(\det\Sigma)^{-1/2} \exp\left(-\tfrac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right) \tag{2.11}$$

The mean of the distribution is given by the vector μ, while the variance is generalized to the *covariance matrix*, Σ, which is symmetric, and equal to $\langle (x-\mu)(x-\mu)^T \rangle$.

In low dimensions, the family of Gaussian distributions is too simple to be of much intrinsic interest from the point of view of artificial intelligence. As the dimensionality, n, increases, however, the number of parameters required to specify an arbitrary Gaussian distribution grows as n^2, and more parsimonious representations become attractive. *Factor analysis* involves searching for such representations in terms of latent variables (see (Everitt, 9:1984)).

Example: Latent class models. The simplest latent variable models express correlations among the observable variables by using a single *latent class* variable that takes on values from a discrete (and often small) set. The observable variables are assumed to be independent given knowledge of this class variable. Such models are commonly used in exploratory data analysis for the social sciences. Responses to a survey of opinions on public policy issues might, for example, be modeled using a latent class variable with three values that could be interpreted as "conservative", "liberal", and "apolitical". Latent class models have been used in artificial intelligence contexts by Cheeseman, *et al* (2:1988), Hanson, Stutz, and Cheeseman (2:1991), Neal (2:1992c), and Anderson and Matessa (2:1992), though not always under this name.

Due to the independence assumption, the joint distribution for the class variable, C, and the visible variables, $V_1, \ldots, V_n$, can be written as

$$P(c, v_1, \ldots, v_n) \quad = \quad P(c) \prod_{j=1}^{n} P(v_j \mid c) \tag{2.12}$$

The model is specified by giving the probabilities on the right of this expression explicitly, or in some simple parametric form. For example, if there are two classes (0 and 1), and the V_i are binary variables (also taking values 0 and 1), we need only specify $\alpha = P(C = 1)$ and the $\beta_{\ell j} = P(V_j = 1 \mid C = \ell)$. The joint probabilities can then be expressed as

$$P(c, v_1, \ldots, v_n) \quad = \quad \alpha^c (1-\alpha)^{1-c} \prod_{j=1}^{n} \beta_{cj}^{v_j} (1-\beta_{cj})^{1-v_j} \tag{2.13}$$

The above specification requires only $2n+1$ numbers, many fewer than the $2^n - 1$ numbers needed to specify an arbitrary joint distribution for the observable variables. Such a reduction is typical when a parametric model is used, and is highly desirable if we in fact have good reason to believe that the distribution is expressible in this restricted form. A latent class model with many more than just two classes could be employed as a non-parametric model, since as the number of classes increases, any distribution can be approximated arbitrarily closely.

The marginal distribution for the observable variables in a latent class model is given by

$$P(v_1, \ldots, v_n) \quad = \quad \sum_{\tilde{c}} P(\tilde{c}, v_1, \ldots, v_n) \tag{2.14}$$

Since only one latent variable is involved, these marginal probabilities can easily be computed (assuming the number of classes is manageable). In fact, conditional probabilities for any set of observable variables given the values for any other set can also be calculated without difficulty. There is thus no need to use Monte Carlo methods for probabilistic inference when the model is of this simple sort, provided it is fully specified.

Example: Belief networks. More complex latent variable models can be constructed using *belief networks*, which can also be used for models where all variables are observable. These networks, which are also referred to as *Bayesian networks*, *causal networks*, *influence di-*

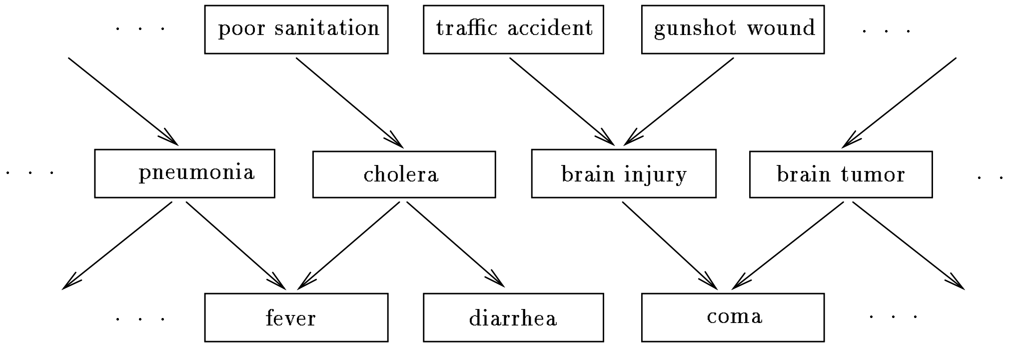


Figure 2.2: A fragment of a hypothetical belief network for medical diagnosis. The full network would contain many more variables and many more connections than are shown here.

agrams, and *relevance diagrams*, have been developed for expert systems applications by Pearl (2:1988) and others (see (Oliver and Smith, 2:1990)). For a tutorial on their use in such applications, see (Charniak, 2:1991). They can also be viewed as non-parametric models to be learned from empirical data (Neal, 2:1992b).

A belief network expresses the joint probability distribution for a set of variables, with an ordering, $X_1, \ldots, X_n$, as the product of the conditional distributions for each variable given the values of variables earlier in the ordering. Only a subset of the variables preceding X_i, its *parents*, $\mathcal{P}_i$, are relevant in specifying this conditional distribution. The joint probability can therefore be written as

$$P(x_1, \ldots, x_n) \;=\; \prod_{i=1}^{n} P(x_i \mid \{x_j : j \in \mathcal{P}_i\}) \tag{2.15}$$

Note that some variables will have no parents (i.e. $\mathcal{P}_i$ will be empty), in which case the conditional probability above will be just the marginal probability for that variable. The latent class model of equation (2.12) can be regarded as a simple belief network in which the class variable has no parents, and is the only parent of all the visible variables.

When a variable has many parents, various ways of economically specifying its conditional probability have been employed, giving rise to various types of belief network. For example, conditional distributions for binary variables can be specified by the "noisy-OR" method (Pearl, 2:1988) or the "logistic" (or "sigmoid") method (Spiegelhalter and Lauritzen, 2:1990, Neal, 2:1992b)). For the latter, the probabilities are as follows:

$$P(X_i = 1 \mid \{x_j : j \in \mathcal{P}_i\}) \;=\; \sigma\Big(\sum_{j \in \mathcal{P}_i} w_{ij} x_j\Big) \tag{2.16}$$

where $\sigma(z) = 1/(1+\exp(-z))$, and the w_{ij} are parameters of the model. Of course, not all conditional distributions can be put in this form.

The structure of a belief network can be represented as a directed acyclic graph, with arrows drawn from parents to children. Figure 2.2 shows the representation of a fragment of a hypothetical belief network intended as a parametric model for medical diagnosis. The variables here are all binary, representing the presence or absence of the stated condition, and are ordered from top to bottom (with no connections within a layer). Arrows out of "traffic accident" and "gunshot wound" indicate that these are relevant in specifying the conditional probability of "brain injury". The lack of an arrow from "poor sanitation" to "brain injury" indicates that the former is not relevant when specifying the conditional probability of "brain injury" given the variables preceding it. For the model to be fully specified, this graphical structure must, of course, be accompanied by actual numerical values for the relevant conditional probabilities, or for parameters that determine these.

The diseases in the middle layer of this belief network are mostly latent variables, invented by physicians to explain patterns of symptoms they have observed in patients. The symptoms in the bottom layer and the underlying causes in the top layer would generally be considered observable. Neither classification is unambiguous — one might consider microscopic observation of a pathogenic microorganism as a direct observation of a disease, and, on the other hand, "fever" could be considered a latent variable invented to explain why some patients have consistently high thermometer readings.

In any case, many of the variables in such a network will not, in fact, have been observed, and inference will require a summation over all possible combinations of values for these unobserved variables, as in equation (2.7). To find the probability that a patient with certain symptoms has cholera, for example, we must sum over all possible combinations of other diseases the patient may have as well, and over all possible combinations of underlying causes. For a complex network, the number of such combinations will be enormous. For some networks with sparse connectivity, exact numerical methods are nevertheless feasible (Pearl, 2:1988, Lauritzen and Spiegelhalter, 2:1988). For general networks, Markov chain Monte Carlo methods are an attractive approach to handling the computational difficulties (Pearl, 4:1987).

Example: Multi-layer perceptrons. The most widely-used class of "neural networks" are the *multi-layer perceptron* (or *backpropagation*) networks (Rumelhart, Hinton, and Williams, 2:1986). These networks can be viewed as modeling the conditional distributions for an output vector, Y, given the various possible values of an input vector, X. The marginal distribution of X is not modeled, so these networks are suitable only for regression or classification applications, not (directly, at least) for applications where the full joint distribution of the observed variables is required. Multi-layer perceptrons have been applied to a great variety of problems. Perhaps the most typical sorts of application take as input sensory information of some type and from that predict some characteristic of what is sensed. (Thodberg (2:1993), for example, predicts the fat content of meat from spectral information.)

Multi-layer perceptrons are almost always viewed as non-parametric models. They can have a variety of architectures, in which "input", "output", and "hidden" units are arranged and connected in various fashions, with the particular architecture (or several candidate architectures) being chosen by the designer to fit the characteristics of the problem. A simple and common arrangement is to have a layer of input units, which connect to a layer of hidden units, which in turn connect to a layer of output units. Such a network is shown in Figure 2.3. Architectures with more layers, selective connectivity, shared weights on connections, or other elaborations are also used.

The network of Figure 2.3 operates as follows. First, the input units are set to their observed values, $x = \{x_1, \ldots, x_m\}$. Values for the hidden units, $h = \{h_1, \ldots, h_p\}$, and for the output units, $o = \{o_1, \ldots, o_n\}$, are then computed as functions of x as follows:

$$h_k(x) = f\big(u_{k0} + \sum_j u_{kj} x_j\big) \tag{2.17}$$

$$o_l(x) = g\big(v_{l0} + \sum_k v_{lk} h_k(x)\big) \tag{2.18}$$

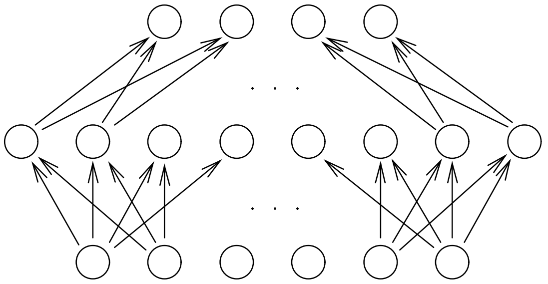


Figure 2.3: A multi-layer perceptron with one layer of hidden units. The input units at the bottom are fixed to their values for a particular case. The values of the hidden units are then computed, followed by the values of the output units. The value of a unit is a function of the the weighted sum of values received from other units connected to it via arrows.

Here, u_{kj} is the weight on the connection from input unit j to hidden unit k, with u_{k0} being a "bias" weight for hidden unit k. Similarly, the v_{lk} are the weights on connections into the output units. The functions f and g are used to compute the activity of a hidden or output unit from the weighted sum over its connections. Generally, the hidden unit function, f, and perhaps g as well, are non-linear, with $f(z) = \tanh(z)$ being a common choice. This non-linearity allows the hidden units to represent "features" of the input that are useful in computing the appropriate outputs. The hidden units thus resemble latent variables, with the difference that their values can be found with certainty from the inputs (in this sense, they are not "hidden" after all).

The conditional distribution for $Y = \{Y_1, \ldots, Y_n\}$ given $x = \{x_1, \ldots, x_m\}$ is defined in terms of the values of the output units computed by the network when the input units are set to x. If the Y_l are real-valued, for example, independent Gaussian distributions with means of $o_l(x)$ and some predetermined "noise" variance, σ^2, might be appropriate. The conditional distribution would then be

$$P(y \mid x) \;=\; \prod_{l=1}^{n} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(o_l(x) - y_l)^2}{2\sigma^2}\right) \tag{2.19}$$

Note that the computations required for the above can be performed easily in time proportional to the number of connections in the network. There is hence no need to use Monte Carlo methods with these networks once their weights have been determined.

2.2 Statistical inference for model parameters

The models described above are fully specified only when the values of certain *model parameters* are fixed — examples are the parameters α and $\beta_{\ell j}$ for the latent class model, and the weights u_{kj} and v_{lk} for a multi-layer perceptron. Determining these parameters from empirical data is a task for statistical inference, and corresponds to one concept of learning in artificial intelligence. The frequentist approach to statistics addresses this task by attempting to find procedures for *estimating* the parameters that can be shown to probably produce "good" results, regardless of what the true parameters are. Note that this does *not* imply that the values actually found in any particular instance are probably good — indeed, such a statement is generally meaningless in this framework. In contrast, the Bayesian approach reduces statistical inference to probabilistic inference by defining a joint distribution for both the parameters and the observable data. Conditional on the data actually observed, *posterior* probability distributions for the parameters and for future observations can then be obtained.

Statistical inference is applicable only when the potentially observable variables come in groups of similar structure — each applying to a particular *case* — with the distributions of the variables in different cases being related in some way. The values seen for the cases that have been observed — the *training cases*, in machine learning terminology — can then tell us something about the distribution for the unseen cases. The simplest assumption, made for all the examples in this review, is that, given the values of the model parameters, the observable (and latent) variables for one case are independent of the variables for the other cases, and the distributions of these variables are the same for all cases. On this assumption, if $X_i = \{X_{i1}, \ldots, X_{in}\}$ are the variables for case i, and $\theta = \{\theta_1, \ldots, \theta_p\}$ are the model parameters, we can write the distribution of the variables for all cases as

$$P(x_1, x_2, \ldots \mid \theta) \;=\; \prod_i P(x_i \mid \theta) \;=\; \prod_i P(x_{i1}, \ldots, x_{in} \mid \theta_1, \ldots, \theta_p) \qquad (2.20)$$

with $P(x_{i1}, \ldots, x_{in} \mid \theta_1, \ldots, \theta_p)$ being a function only of the model parameters and of the values x_{ij}, not of i itself. The number of cases is considered indefinite, though in any particular instance we will be concerned only with whatever number of cases have been observed, plus whatever number of unobserved cases we would like to make predictions for. (Note that the variables used to express the models of the preceding section will in this section acquire an additional index, to distinguish the different cases. Also, while in the previous section the model parameters were considered fixed, and hence were not explicitly noted in the formulas, in this section the distribution for the data will be shown explicitly to depend on the parameter values.[1])

I will use coin tossing as a simple illustrative problem of statistical inference. In this example, each case, X_i, consists of just one value, representing the result of tossing a particular coin for the i-th time, with $X_i = 1$ representing heads, and $X_i = 0$ representing tails. We model the coin as having a "true" probability of landing heads given by a single real number, θ, in the interval $[0, 1]$. The probability of a particular series of tosses, $x_1, \ldots, x_C$, is then

$$P(x_1, \ldots, x_C \mid \theta) \;=\; \prod_i \theta^{x_i}(1-\theta)^{1-x_i} \;=\; \theta^{C_1}(1-\theta)^{C_0} \qquad (2.21)$$

where $C_1 = \sum_i x_i$, i.e. the number of the x_i that are one (heads), and $C_0 = C - C_1$, i.e. the number that are zero (tails).

The machine learning literature distinguishes between *supervised* and *unsupervised* learning. Supervised learning can be seen as statistical inference for a regression or classification model, in which only the conditional distributions of certain variables are modeled, whereas unsupervised learning can (on one interpretation, at least) be seen as statistical inference for a model that defines the joint probability of all observable variables.

Maximum likelihood inference. The probability that a model with particular parameters values assigns to the data that has actually been observed (with any unobserved variables being summed over) is called the *likelihood*. For example, if cases $X_1, \ldots, X_C$ have been observed in their entirety (and nothing else has been observed), then the likelihood is

[1] I show this dependence by writing the parameter as if it were a variable on whose value we are conditioning. This is fine for Bayesians. Others may object on philosophical grounds, but will likely not be confused.

$$L(\theta \mid x_1, \ldots, x_C) \;=\; P(x_1, \ldots, x_C \mid \theta) \;=\; \prod_{i=1}^{C} P(x_i \mid \theta) \tag{2.22}$$

The likelihood is regarded as a function of the model parameters, with given data, and is considered significant only up to multiplication by an arbitrary factor. It encapsulates the relative abilities of the various parameter values to "explain" the observed data, which may be considered a measure of how plausible the parameter values are in light of the data. In itself, it does *not* define a probability distribution over parameter values, however — for that, one would need to introduce some measure on the parameter space as well.[2]

The widely used *maximum likelihood* procedure estimates the parameters of the model to be those that maximize the likelihood given the observed data. In practice, the equivalent procedure of maximizing the log of the likelihood is usually found to be more convenient.

For the coin tossing example, the log likelihood function given data on C flips, obtained from equation (2.21), is

$$\log L(\theta \mid x_1, \ldots, x_C) \;=\; C_1 \log(\theta) \;+\; C_0 \log(1-\theta) \tag{2.23}$$

The maximum likelihood estimate for the "true" probability of heads is easily found to be $\hat{\theta} = C_1/C$, i.e. the frequency of heads in the observed flips.

For a large class of models, the maximum likelihood procedure has the frequentist justification that it converges to the true parameter values in the limit as the number of observed cases goes to infinity. This is not always the case, however, and even when it is, the quality of such estimates when based on small amounts of data may be poor. One way to address such problems is to choose instead the parameters that maximize the log likelihood plus a penalty term, which is intended to bias the result away from "overfitted" solutions that model the noise in the data rather the true regularities. This is the *maximum penalized likelihood* method. The magnitude of the penalty can be set by hand, or by the method of *cross validation* (for example, see (Efron, 9:1979)).

Naively, at least, predictions for unobserved cases in this framework are done using the single estimate of the parameters found by maximizing the likelihood (or penalized likelihood). This is not always very reasonable. For the coin tossing example, if we flip the coin three times, and each time it lands heads, the maximum likelihood estimate for the probability of heads is one, but the resulting prediction that on the next toss the coin is certain to land head-up is clearly not warranted.

Example: Univariate Gaussian. Suppose that $X_1, \ldots, X_C$ are independent, and that each has a univariate Gaussian distribution with the same parameters, μ and σ, with σ being known, but μ not known. We can estimate μ by maximum likelihood. From equation (2.10), the likelihood function can be found:

$$L(\mu \mid x_1, \ldots, x_C) \;=\; \prod_{i=1}^{C} P(x_i \mid \mu) \;=\; \prod_{i=1}^{C} \frac{1}{\sqrt{2\pi}\sigma} \exp\big(-(x_i-\mu)^2/2\sigma^2\big) \tag{2.24}$$

Taking the logarithm, for convenience, and discarding terms that do not involve μ, we get:

[2] The definition of likelihood given here is that used by careful writers concerned with foundational issues. Unfortunately, some Bayesians have taken to using "likelihood" as a synonym for "probability", to be used only when referring to observed data. This has little practical import within the Bayesian school, but erases a distinction important to those of some other schools who are happy to talk about the "likelihood that $\theta = 0$", but who would never talk about the "probability that $\theta = 0$".

$$\log L(\mu \mid x_1, \ldots, x_C) \;=\; -\frac{1}{2\sigma^2} \sum_{i=1}^{C} (x_i - \mu)^2 \tag{2.25}$$

The value of μ that maximizes this is the arithmetic average of the observed values:

$$\hat{\mu} \;=\; \frac{1}{C} \sum_{i=1}^{C} x_i \;=\; \bar{x} \tag{2.26}$$

One virtue of this estimate, from the frequentist perspective, is that it is *unbiased* — for any true value of μ, the expected value of the estimate, $\hat{\mu}$, is equal to the true value, the expectation being taken with respect to the distribution that μ defines for $X_1, \ldots, X_C$.

Example: Multi-layer perceptrons. The log of the likelihood for the parameters of the multi-layer perceptron of Figure 2.3 (i.e. for the weight matrices u and v), given the training cases $(x_1, y_1), \ldots, (x_C, y_C)$, is

$$\log L(u,\, v \mid (x_1, y_1), \ldots, (x_C, y_C)) \;=\; \log P(y_1, \ldots, y_C \mid x_1, \ldots, x_C,\, u,\, v) \tag{2.27}$$

$$=\; -\sum_{i=1}^{C} \sum_{l=1}^{n} \frac{(o_l(x_i) - y_{il})^2}{2\sigma^2} \tag{2.28}$$

where terms that do not depend on u or v have been omitted, as they are not significant. Note that the functions $o_l(\,\cdot\,)$ do depend on u and v (see equations (2.17) and (2.18)). The above expression does not quite have the form of (2.22) because the network does not attempt to model the marginal distribution of the X_i.

The objective of conventional neural network training is to minimize an "error" function which is proportional to the negative of the above log likelihood. Such training can thus be viewed as maximum likelihood estimation. Since the focus is solely on the conditional distribution for the Y_i, this is an example of supervised learning.

A local maximum of the likelihood of equation (2.28) can be found by gradient-based methods, using derivatives of $\log L$ with respect to the u_{kj} and v_{lk} obtained by the "backpropagation" method, an application of the chain rule (Rumelhart, Hinton, and Williams, 2:1986). The likelihood is typically a very complex function of the weights, with many local maxima, and an enormous magnitude of variation. Perhaps surprisingly, simple gradient-based methods are nevertheless capable of finding good sets of weights in which the hidden units often compute non-obvious features of the input.

Multi-layer perceptrons are sometimes trained using "weight decay". This method can be viewed as maximum penalized likelihood estimation, with a penalty term proportional to minus the sum of the squares of the weights. This penalty encourages estimates in which the weights are small, and is found empirically to reduce overfitting.

Bayesian inference. Bayesian statistical inference requires an additional input not needed by frequentist procedures such as maximum likelihood — a *prior* probability distribution for the parameters, $P(\theta_1, \ldots, \theta_p)$, which embodies our judgement, before seeing any data, of how plausible it is that the parameters could have values in the various regions of parameter space. The introduction of a prior is the crucial element that converts statistical inference into an application of probabilistic inference.

The need for a prior is also, however, one of the principal reasons that some reject the use of the Bayesian framework. Partly, this is because the prior can usually be interpreted only as an expression of degrees of belief, though in uncommon instances it could be derived from observed frequencies (in which case, use of the Bayesian apparatus would be uncontroversial). It may also be maintained that the choice of prior is subjective, and therefore objectionable, at least when the problem appears superficially to be wholely objective. (There may be less objection if a subjective prior is based on expert opinion, or otherwise introduces relevant new information.) Bayesians divide into two schools on this last point. Some seek ways of producing "objective" priors that represent complete ignorance about the parameters. Others, while finding such priors useful on occasion, regard the quest for complete objectivity as both unnecessary and unattainable. There will be no need to resolve this debate here.

When we combine a prior distribution for the parameters with the conditional distribution for the observed data, we get a joint distribution for all quantities related to the problem:

$$\begin{aligned} P(\theta_1, \ldots, \theta_p, x_1, x_2, \ldots) &= P(\theta_1, \ldots, \theta_p)\, P(x_1, x_2, \ldots \mid \theta_1, \ldots, \theta_p) && (2.29) \\ &= P(\theta) \prod_i P(x_i \mid \theta) && (2.30) \end{aligned}$$

From this, we can derive *Bayes' rule* for the *posterior* distribution of the parameters, given observed values for $X_1, \ldots, X_C$:

$$P(\theta \mid x_1, \ldots, x_C) \;=\; \frac{P(\theta, x_1, \ldots, x_C)}{P(x_1, \ldots, x_C)} \;=\; \frac{P(\theta) \prod_{i=1}^{C} P(x_i \mid \theta)}{\int P(\tilde{\theta}) \prod_{i=1}^{C} P(x_i \mid \tilde{\theta})\, d\tilde{\theta}} \qquad (2.31)$$

The posterior can also be expressed as a proportionality in terms of the likelihood:

$$P(\theta \mid x_1, \ldots, x_C) \;\propto\; P(\theta)\, L(\theta \mid x_1, \ldots, x_C) \qquad (2.32)$$

This shows how the introduction of a prior converts the expressions of relative plausibility contained in the likelihood into an actual probability distribution over parameter space.

A simple prior density for the coin tossing example is $P(\theta) = 1$, i.e. the uniform distribution on the interval $[0, 1]$. The corresponding posterior after observing C flips can be obtained by substituting equation (2.21) into (2.31):

$$P(\theta \mid x_1, \ldots, x_C) \;=\; \frac{\theta^{C_1}(1-\theta)^{C_0}}{\int_0^1 \tilde{\theta}^{C_1}(1-\tilde{\theta})^{C_0}\, d\tilde{\theta}} \;=\; \frac{(C+1)!}{C_0!\, C_1!}\, \theta^{C_1}(1-\theta)^{C_0} \qquad (2.33)$$

Here, I have used the well-known "beta" integral: $\int_0^1 x^a(1-x)^b\, dx = a!b!/(a+b+1)!$. As C grows large, this posterior distribution becomes highly concentrated around the maximum likelihood estimate, $\hat{\theta} = C_1/C$.

The Bayesian framework can provide a *predictive* distribution for an unobserved case, X_{C+1}, given the values observed for $X_1, \ldots, X_C$:

$$P(x_{C+1} \mid x_1, \ldots, x_C) \;=\; \int P(x_{C+1} \mid \tilde{\theta})\, P(\tilde{\theta} \mid x_1, \ldots, x_C)\, d\tilde{\theta} \qquad (2.34)$$

Such distributions, or similar conditional distributions in which some of the variables for case $C+1$ are also known, are what are generally needed when making decisions relating to new cases. Note that the Bayesian predictive distribution is not based on a single estimate for the parameters, but is instead an average of the predictions using all possible values of the parameters, with each prediction weighted by the probability of the parameters having those values. This reasonable idea of averaging predictions leads almost inevitably to the Bayesian approach, since it requires that a measure be defined on the parameter space, which can then be interpreted as a Bayesian prior distribution.

For the coin tossing example, the predictive probability of heads on flip number $C+1$, given data on the first C flips, is

$$
\begin{aligned}
P(X_{C+1} = 1 \mid x_1, \ldots, x_C) &= \int_0^1 \tilde{\theta} \cdot \frac{(C+1)!}{C_0!\,C_1!}\, \tilde{\theta}^{C_1}(1-\tilde{\theta})^{C_0}\, d\tilde{\theta} && (2.35) \\
&= \frac{(C+1)!}{C_0!\,C_1!} \cdot \frac{C_0!\,(C_1+1)!}{(C+2)!} && (2.36) \\
&= \frac{C_1+1}{C+2} && (2.37)
\end{aligned}
$$

Note that this predictive distribution never assigns zero probability to either heads or tails, reflecting the fact that values of θ that assign non-zero probability to both results cannot be completely ruled out with any finite amount of data (unless they were ruled out from the start by the prior). Different priors for θ will, of course, give different predictive distributions, whose form will not in general be as simple as that above.

A Bayesian using the above approach to prediction will in theory almost never be working with a fully-specified model derived from empirical data, of the sort discussed in the Section 2.1 — there will instead always be some degree of uncertainty in the parameter values. In practice, if the posterior distribution is very strongly peaked, the predictive distribution of equation (2.34) can be replaced by the prediction made using the values of the parameters at the peak, with a negligible loss of accuracy, and a considerable reduction in computation. Situations where the amount of training data is not sufficient to produce such a strong peak are by no means uncommon, however. It is in such situations that one might expect the Bayesian predictions to be better than the predictions obtained using any single estimate for the model parameters.

Evaluation of the integrals over parameter space in equations (2.31) and (2.34) can be very demanding. Note that the predictive distribution of equation (2.34) can be viewed as the expectation of $P(x_{C+1} \mid \theta)$ with respect to the posterior distribution for θ. The Monte Carlo estimation formula of equation (1.2) thus applies. For problems of interest in artificial intelligence, the parameter space often has very high dimensionality, and the posterior distribution is very complex. Obtaining a Monte Carlo estimate may then require use of Markov chain sampling methods. As we will see, these methods require only that we be able to calculate probability densities for parameter values up to an unknown constant factor; we therefore need not evaluate the integral in the denominator of equation (2.31).

Monte Carlo techniques can also be used to evaluate whether the prior distribution chosen reflects our actual prior beliefs. Even before any data has been observed, we can find the *prior predictive* distribution for a set of data items, $X_1, \ldots, X_C$:

$$
P(x_1, \ldots, x_C) = \int P(\tilde{\theta}) \cdot \prod_{i=1}^{C} P(x_i \mid \tilde{\theta})\, d\tilde{\theta} \qquad (2.38)
$$

If we have a sample of data sets generated according to this distribution, we can examine them to determine whether they are representative of what our prior beliefs lead us to expect. If they are not, then the prior distribution, or perhaps the entire model, is in need of revision.

Generating a value from the prior predictive distribution can be done by first generating a value, θ, from the prior parameter distribution, and then generating values for $X_1, \ldots, X_C$ from their distribution conditional on θ. Even for quite complex models, these are often easy operations. When this is not the case, however, Markov chain sampling can be useful for this task (as is illustrated in (Szeliski, 2:1989, Chapter 4)).

Example: Univariate Gaussian. Suppose that $X_1, \ldots, X_C$ are independent Gaussian variables with the same unknown mean, μ, but known variance, σ^2. Let the prior distribution for μ be Gaussian with mean μ_0 and variance σ_0^2. Using equations (2.32) and (2.10), the posterior distribution for μ given values for the X_i can be found as follows:

$$\begin{aligned} P(\mu \mid x_1, \ldots, x_C) &\propto P(\mu) \prod_{i=1}^{C} P(x_i \mid \mu) && (2.39) \\ &\propto \exp(-(\mu-\mu_0)^2/2\sigma_0^2) \prod_{i=1}^{C} \exp(-(x_i-\mu)^2/2\sigma^2) && (2.40) \\ &\propto \exp(-(\mu-\mu_1)^2/2\sigma_1^2) && (2.41) \end{aligned}$$

where $1/\sigma_1^2 = 1/\sigma_0^2 + C/\sigma^2$ and $\mu_1 = (\mu_0/\sigma_0^2 + C\bar{x}/\sigma^2)/\sigma_1^2$, with $\bar{x} = C^{-1}\sum_i x_i$. Thus the posterior distribution for μ is a univariate Gaussian with a variance that decreases as the amount of data increases, and with a mean that combines information from the data with information from the prior, eventually converging on the maximum likelihood estimate of equation (2.26).

The predictive distribution for a new data item, X_{C+1}, can be obtained by applying equation (2.34). The result is a Gaussian distribution with mean μ_1 and variance $\sigma_1^2 + \sigma^2$.

As with the coin-tossing example, the analytical tractability seen here results from a convenient choice of prior distribution, which will not always be a reasonable representation of actual prior beliefs.

Example: Multi-layer perceptrons. To perform Bayesian inference for a multi-layer perceptron, we must decide on a prior for the parameters u and v. A simple candidate is

$$P(u, v) = \prod_{k,j} \frac{1}{\sqrt{2\pi}\sigma_u} \exp\big(-u_{kj}^2/2\sigma_u^2\big) \cdot \prod_{l,k} \frac{1}{\sqrt{2\pi}\sigma_v} \exp\big(-v_{lk}^2/2\sigma_v^2\big) \qquad (2.42)$$

i.e. independent Gaussian distributions for each weight, with variance σ_u^2 for the input-hidden weights, and σ_v^2 for the hidden-output weights. This prior makes small weights more likely than large weights, with the degree of bias depending on the values of σ_u and σ_v. In this respect, it resembles the penalized likelihood method of "weight decay" described previously.

We can evaluate whether this prior captures our beliefs in respect of some particular problem by generating a sample from the distribution over functions defined by this distribution over weights. (This is essentially the prior predictive distribution if the outputs are assumed to be observed without noise.) Figure 2.4 shows two such samples, with different values of σ_u, for a problem with one real-valued input and one real-valued output. As can be seen, the value of σ_u controls the smoothness of the functions. Note in this respect that as the number of hidden units in the network increases, it becomes *possible* for the function defined by the network to be very ragged, but *typical* functions generated using this prior do *not* become more and more ragged. (However, as the number of hidden units increases, σ_v must decrease in proportion to the square root of this number to maintain the output scale.) I discuss this point further in (Neal, 2:1993b).

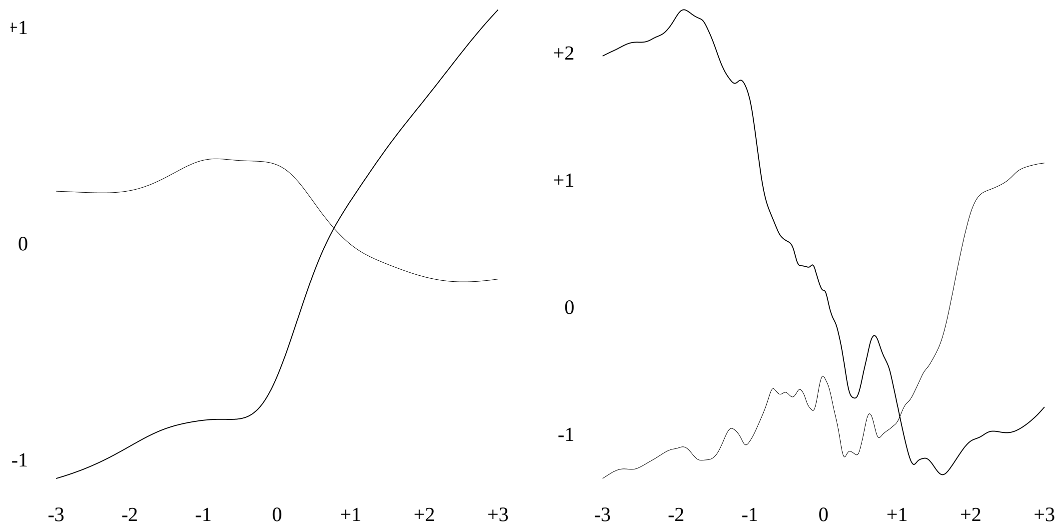


Figure 2.4: Samples from the prior distribution over functions implied by the prior of equation (2.42), (a) with $\sigma_u = 1$, (b) with $\sigma_u = 10$. Two functions generated from each prior are shown, with the one-dimensional input, x, plotted on the horizontal axis, and the single output, $o(x)$, plotted on the vertical axis. The hidden unit activation function was tanh; the output unit activation function was the identity. Networks with 1000 hidden units were used, and σ_v was set to $1/\sqrt{1000}$.

When the prior is combined with the multi-layer perceptron's typically complex likelihood function (equation (2.28)) a correspondingly complex posterior distribution for the weights results. It is thus appropriate to consider Markov chain Monte Carlo methods as a means of performing Bayesian inference for these networks (Neal, 2:1992a, 2:1993a), though other approaches are also possible, as is discussed in Section 3.2.

Statistical inference with unobserved and latent variables. In the previous section, I have assumed that when case i is seen, we find out the values of all the variables, $X_{i1}, \ldots, X_{in}$, that relate to it. This will not always be so, certainly not when some of these are latent variables. Theoretically, at least, this is not a problem in the Bayesian and maximum likelihood frameworks — one need only work with marginal probabilities obtained by summing over the possible values of these unknown variables.

Maximum likelihood or maximum penalized likelihood learning procedures for models with latent variables can be found in (Ackley, Hinton, and Sejnowski, 2:1985), for "Boltzmann machines", in (Cheeseman, *et al*, 2:1988), for latent class models, and in (Neal, 2:1992b), for belief networks. The procedures for Boltzmann machines and belief networks use Markov chain Monte Carlo methods to estimate the gradient of the log likelihood. The procedure of Cheeseman, *et al* for latent class models is based on the *EM algorithm* (Dempster, Laird, and Rubin, 9:1977), which is widely applicable to problems of maximum likelihood inference with latent or other unobserved variables.

Bayesian inference is again based on the posterior distribution, which, generalizing equation (2.31), can be written as

$$P(\theta \mid \{x_{ij} : i \in \{1,\ldots,C\},\, j \in \mathcal{K}_i\}) \;=\; \frac{P(\theta) \prod\limits_{i=1}^{C} \sum\limits_{\{x_{ij}:j\notin\mathcal{K}_i\}} P(x_{i1},\ldots,x_{in} \mid \theta)}{\int P(\tilde{\theta}) \prod\limits_{i=1}^{C} \sum\limits_{\{x_{ij}:j\notin\mathcal{K}_i\}} P(x_{i1},\ldots,x_{in} \mid \tilde{\theta})\, d\tilde{\theta}} \tag{2.43}$$

where $\mathcal{K}_i$ is the set of indices of variables for case i whose values are known from observation. (I have here had to depart from my conventional placement of tildes over the unknown x_{ij}.)

Computationally, when this posterior is used to make predictions, the required summations over the unobserved variables can be done as part of the same Monte Carlo simulation that is used to evaluate the integrals over parameter space. This convenient lack of any fundamental distinction between the variables applying to a particular case and the parameters of the model is a consequence of statistical inference being reduced to probabilistic inference in the Bayesian framework.

For some models, the predictive distribution given by the integral of equation (2.34) can be found analytically, or the model might be specified directly in terms of this distribution, with no mention of any underlying parameters. If all the variables in the training cases have known values, finding the predictive distribution for a new case will then be easy. If there are latent or other unobserved variables, however, computing the predictive distribution will require a summation or integration over the possible values these variables could take on in all the training cases where they were not observed. Monte Carlo methods may then be required.

Example: Latent class models. Since a latent class model defines the joint distribution of all observable variables, statistical inference for such a model can be seen as an unsupervised learning procedure. Bayesian inference for latent class models is discussed by Hanson, Stutz, and Cheeseman (2:1991) and by myself (Neal, 2:1992c). I will use the two-class model for binary data of equation (2.13) to illustrate the concepts. At least three computational approaches are possible with this model, in which different sets of variables are integrated or summed over analytically.

Assuming we use a prior distribution in which α and the $\beta_{\ell j}$ are all independent, the joint distribution for the parameters of this model and all the variables relating to C cases can be written as

$$P(\alpha,\, \beta,\, c,\, v) \;=\; P(\alpha) \cdot \prod_{\ell,j} P(\beta_{\ell j}) \cdot \prod_{i=1}^{C} \left(\alpha^{c_i} (1-\alpha)^{1-c_i} \prod_{j=1}^{n} \beta_{c_i j}^{v_{ij}} (1-\beta_{c_i j})^{1-v_{ij}} \right) \tag{2.44}$$

From this joint distribution, the conditional distribution given the observed values of the V_i, could be obtained, and sampled from by Markov chain techniques. The predictive distribution for a new case could then be found by Monte Carlo methods, as an expectation with respect to this distribution.

For this particular model it is possible instead to analytically integrate out the parameters, provided the priors on α and the $\beta_{\ell j}$ are of a certain form — in particular, this is possible when the priors are uniform on $[0,1]$, i.e. $P(\alpha) = 1$ and $P(\beta_{\ell j}) = 1$. Using the "beta" integral, as in equation (2.33), we can obtain the marginal distribution for just the observable and class variables for the various cases, as follows:

$$P(c,\, v) \;=\; \frac{r_0!\; r_1!}{(C+1)!} \prod_{\ell=0}^{1} \prod_{j=1}^{n} \frac{s_{\ell j0}!\; s_{\ell j1}!}{(r_\ell+1)!} \tag{2.45}$$

where $r_\ell = \sum_{i=1}^{C} \delta(\ell, c_i)$ and $s_{\ell j b} = \sum_{i=1}^{C} \delta(\ell, c_i)\delta(b, v_{ij})$. Sampling from the distribution for the C_i given values for the V_i can again be done using Markov chain methods, and the results used to find the predictive distribution for a new case.

Alternatively, we can sum over the possible values for each of the class variables (which are independent given the values of the parameters) obtaining the marginal distribution for the parameters and the observable variables alone:

$$P(\alpha, \beta, v) \;=\; P(\alpha) \cdot \prod_{\ell, j} P(\beta_{\ell j}) \cdot \prod_{i=1}^{C} \sum_{\tilde{c}_i=0}^{1} \left(\alpha^{\tilde{c}_i} (1-\alpha)^{1-\tilde{c}_i} \prod_{j=1}^{n} \beta_{\tilde{c}_i j}^{v_{ij}} (1-\beta_{\tilde{c}_i j})^{1-v_{ij}} \right) \tag{2.46}$$

From this, one can obtain a posterior parameter distribution, which Hanson, *et al* (2:1991) approximate and then use to make predictions for new cases using equation (2.34). One could also try sampling from this posterior using Markov chain methods.

Expressing priors using hyperparameters. Just as latent variables are sometimes a useful tool for expressing the distribution of observable variables for a particular case, so too can the prior distribution for the parameters of a model sometimes be most conveniently expressed using additional *hyperparameters*. For example, the prior for a set of parameters $\theta_1, \ldots, \theta_p$ might be represented as a marginal distribution using a hyperparameter, α, as follows:

$$P(\theta_1, \ldots, \theta_p) \;=\; \int P(\tilde{\alpha}, \theta_1, \ldots, \theta_p)\, d\tilde{\alpha} \;=\; \int P(\theta_1, \ldots, \theta_p \mid \tilde{\alpha})\, P(\tilde{\alpha})\, d\tilde{\alpha} \tag{2.47}$$

This technique can be extended to any number of levels; the result is sometimes referred to as a *hierarchical model*. The dependency relationships amongst hyperparameters, parameters, latent variables, and observed variables, can often be conveniently expressed in the belief network formalism, as is done by Thomas, Spiegelhalter, and Gilks (4:1992).

To give a simple example, suppose the observable variables are the weights of various dogs, each classified according to breed, and that θ_k is the mean weight for breed k, used to specify a Gaussian distribution for weights of dogs of that breed. Rather than using the same prior for each θ_k, independently, we could instead give each a Gaussian prior with a mean of α, and then give α itself a prior as well. The effect of this hierarchical structure can be seen by imagining that we have observed dogs of several breeds and found them all to be surprisingly heavy. Rather than stubbornly persisting with our underestimates for every new breed we encounter, we will instead adjust our idea of how heavy dogs are in general by changing our view of the likely value of the hyperparameter α. We will then start to expect even dogs of breeds that we have never seen before to be heavier than we would have expected at the beginning.

Models specified using hyperparameters can easily be accommodated by Monte Carlo methods, provided only that the probabilities $P(\theta \mid \alpha)$ can be easily calculated, at least up to a factor that does not depend on either θ or α.

Example: Multi-layer perceptrons. Often, we will have difficulty deciding on good values for σ_u and σ_v in the prior over networks weights of equation (2.42) — either because we are ignorant about the nature of the function underlying the data, or because we have difficulty visualizing the effects of these parameters with high-dimensional inputs and outputs. It is then sensible to treat σ_u and σ_v as hyperparameters, with rather broad prior distributions. If the problem turns out to require a function that is smooth only on a small scale, a large value of σ_u will be found, permitting large input-hidden weights, while, for some other problem, the same hierarchical model might lead to a small value for σ_u, appropriate for a much smoother function. Similarly, when σ_v is a hyperparameter, the model can adapt to the overall scale of the outputs. This topic is discussed in more detail by MacKay (2:1991, 2:1992b).

2.3 Bayesian model comparison

Section 2.1 dealt with probabilistic inference given a model in which the parameters were fully specified. Section 2.2 discussed inference when the parameters of the model were unknown, but could be inferred on the basis of the information in a set of training cases. I will now consider inference when even the form of the model is uncertain. This is a rather open-ended problem, since generally we, or a learning program, will be able to come up with any number of possible models, often of an increasingly elaborate nature — we can, for example, contemplate modeling data using multi-layer perceptron networks with various numbers of hidden layers, arranged in a variety of architectures.

I will assume here, however, that we have reduced the problem to that of comparing a fairly small number of models, all of which we regard *a priori* as having a reasonable chance of being the truth, or at least of being a useful approximation to the truth. For simplicity, I will deal with only two models, M_A and M_B, with prior probabilities $P(M_A)$ and $P(M_B)$ (which sum to one). Presumably these prior probabilities will be roughly equal, since we wouldn't bother to even consider a model that we felt was highly implausible to start with. This is true regardless of whether one model is more elaborate than the other (i.e. is specified using a larger number of parameters). "Occam's razor" — the principle of avoiding unnecessary complexity — is implicitly embodied in the Bayesian framework through the effect of each model's prior on its parameters, so there is generally no need to incorporate a further bias toward simplicity using the priors on the models themselves. See (MacKay, 2:1991, 2:1992a) and (Jeffreys and Berger, 9:1992) for discussion of this point.

Suppose that model M_A has parameters $\theta = \{\theta_1, \ldots, \theta_p\}$, with prior distribution $P(\theta \mid M_A)$, while model M_B has a different set of parameters, $\phi = \{\phi_1, \ldots, \phi_q\}$, with prior $P(\phi \mid M_B)$. For each model, the parameters determine the probabilities for the observable variables in each case, as $P(x_i \mid \theta, M_A)$, and $P(x_i \mid \phi, M_B)$. The probability of the entire training set under each model is given by

$$P(x_1, \ldots, x_C \mid M_A) \;=\; \int P(\tilde{\theta} \mid M_A) \prod_{i=1}^{C} P(x_i \mid \tilde{\theta}, M_A)\, d\tilde{\theta} \tag{2.48}$$

$$P(x_1, \ldots, x_C \mid M_B) \;=\; \int P(\tilde{\phi} \mid M_B) \prod_{i=1}^{C} P(x_i \mid \tilde{\phi}, M_B)\, d\tilde{\phi} \tag{2.49}$$

The posterior model probabilities can then be found by Bayes' rule:

$$P(M_A \mid x_1, \ldots, x_C) \;=\; \frac{P(M_A)\, P(x_1, \ldots, x_C \mid M_A)}{P(M_A)\, P(x_1, \ldots, x_C \mid M_A) + P(M_B)\, P(x_1, \ldots, x_C \mid M_B)} \tag{2.50}$$

and similarly for $P(M_B \mid x_1, \ldots, x_C)$.

The predictive probability for a new case is the mixture of the predictions of the two models, weighted by their posterior probabilities:

$$\begin{aligned} P(x_{C+1} \mid x_1, \ldots, x_C) \;=\; & P(x_{C+1} \mid x_1, \ldots, x_C, M_A)\, P(M_A \mid x_1, \ldots, x_C) \\ & + P(x_{C+1} \mid x_1, \ldots, x_C, M_B)\, P(M_B \mid x_1, \ldots, x_C) \end{aligned} \tag{2.51}$$

The predictive probabilities given each model are obtained as in equation (2.34). Often, the information in the training data is sufficient to make the posterior probability for one model be very much greater than that for the other. We can then simply ignore the predictions of the improbable model, accepting the overwhelmingly more probable one as being "true".

For the coin flipping example, we could contemplate a very simple model, M_A, with no parameters, that simply states that the probability of the coin landing head-up is one-half, and a more complex model, M_B, that has a parameter for the "true" probability of heads, which is given a uniform prior. When examining a coin of somewhat disreputable provenance, it may be reasonable to assign these two models roughly equal prior probabilities, say $P(M_A) = P(M_B) = 1/2$. This choice of models and of the prior over models embodies a belief that it is plausible that the probabilities of heads and tails might be exactly equal (or equal for all practical purposes), while we have no particular reason to think that the coin might be biased so as to land heads, say, 37.2% of the time.

After we have flipped the coin C times, with results $x_1, \ldots, x_C$, we can compare how probable these two models are in light of this data. Suppose that of these flips, C_1 landed heads and C_0 landed tails. The probability of the observed data under the two models will then be as follows:

$$P(x_1, \ldots, x_C \mid M_A) = 2^{-C} \tag{2.52}$$

$$P(x_1, \ldots, x_C \mid M_B) = \frac{C_0!\, C_1!}{(C+1)!} \tag{2.53}$$

where the probability under M_B is found by integrating equation (2.21) with respect to θ.

For ten flips, with $C_1 = 6$ and $C_0 = 4$, equation (2.50) gives the result $P(M_A) = 0.693$, showing that the simpler model can be favoured even when the data could be "explained" better by the more complex model, if its parameter were set appropriately. With $C_1 = 8$ and $C_0 = 2$, however, we find that $P(M_A) = 0.326$, showing that the more complex model can be favoured when the evidence for it is strong.

In this simple example, the integrations required to calculate the probability of the training data under each model could be done analytically, but for more complex models, this will generally not be the case. Note that the required probabilities (in equations (2.48) and (2.49)) correspond to the denominator of equation (2.31), whose evaluation was not required for inference with respect to a single model. Typically, these probabilities will be extremely small, since any *particular* data set of significant size will have low probability, even under the correct model. We are interested in the *relative* magnitude of these very small probabilities, as given by the two models being compared. In Section 6.2, techniques for finding such ratios are discussed. We will see that even though such calculations involve more than simply finding a Monte Carlo estimate for an expectation, they are nevertheless possible using a series of Monte Carlo simulations.

The Bayesian model comparison framework is used by MacKay (2:1991, 2:1992a, 2:1992b), to compare different interpolation models and different architectures for multi-layer perceptrons, and by Hanson, Stutz, and Cheeseman (2:1991), to compare latent class models with different numbers of classes, and different hierarchical structure.

2.4 Statistical physics

Historically, Monte Carlo methods based on Markov chains were first developed for performing calculations in statistical physics, and interesting methods of general applicability continue to be developed in by workers in this area. Techniques from statistical physics of a theoretical nature have also been applied to analogous problems in statistical inference. Here, I will briefly outline the essential concepts and vocabulary required to understand the literature in this area. This material is covered in innumerable texts on statistical physics, such as that of Thompson (9:1988).

Microstates and their distributions. A complete *microscopic* description of a physical system specifies its state (more precisely, its *microstate*) in complete detail. For example, the microstate of a quantity of some substance would include a specification of the position and velocity of every molecule of which it is composed. In contrast, a *macroscopic* description specifies only the system's *macrostate*, which is sufficient to determine its macroscopically observable properties. In the above example, the macrostate might be specified by the temperature, volume, and mass of the substance. Whereas the macroscopic state is easily observed, the exact microstate is essentially unknowable, and hence must be treated in terms of probabilities. One of the goals of statistical physics is to relate these two levels of description.

Every possible microstate, s, of the system has some definite *energy*, $E(s)$, which may also be a function of the external environment (for instance, the applied magnetic field). If the system is isolated, then this energy is fixed, say at E_0, and the assumption is generally made that all microstates with that energy are equally likely (and all those with a different energy are impossible). For a system with a continuous state, we thus have $P(s) = Z^{-1}\delta(E_0, E(s))$, for some normalization constant Z. This uniform distribution over states of a given energy is known as the *microcanonical* distribution.

We can also consider systems that are not isolated, but instead exchange energy with a much larger reservoir that maintains the system at a constant *temperature*. The system's energy can then fluctuate, and it is assumed that the probability of the system being in microstate s, given that the temperature is T, is

$$P(s) \quad = \quad \frac{1}{Z}\exp\big(-E(s)/T\big) \tag{2.54}$$

(using suitable units for temperature). Here, $Z = \sum_{\tilde{s}} \exp(-E(\tilde{s})/T)$ is the normalization constant needed to make the distribution sum (or integrate) to one. This distribution is known as the *canonical* (or *Gibbs*, or *Boltzmann*) distribution over microstates. It is with such distributions that we will primarily be concerned.

The physical systems commonly studied can be of arbitrary size — in microscopic terms, the dimension of the state variable, s, can be made arbitrarily large, with the energy, $E(s)$, being defined for any size of system. An *intensive* quantity is one whose value is independent of system size, such as temperature. *Extensive* quantities, such as energy, grow with system size. If the system's interactions are local, this growth will be linear for large systems, and the values of extensive quantities per unit of system size will reach limiting values.

The characteristics of the system in this *thermodynamic limit* of macroscopic size are of the most interest in statistical physics. Most macroscopic properties of a system, such as the energy per unit of system size, can be expressed as expectations with respect to the canonical distribution. In the thermodynamic limit, the fluctuations in these quantities

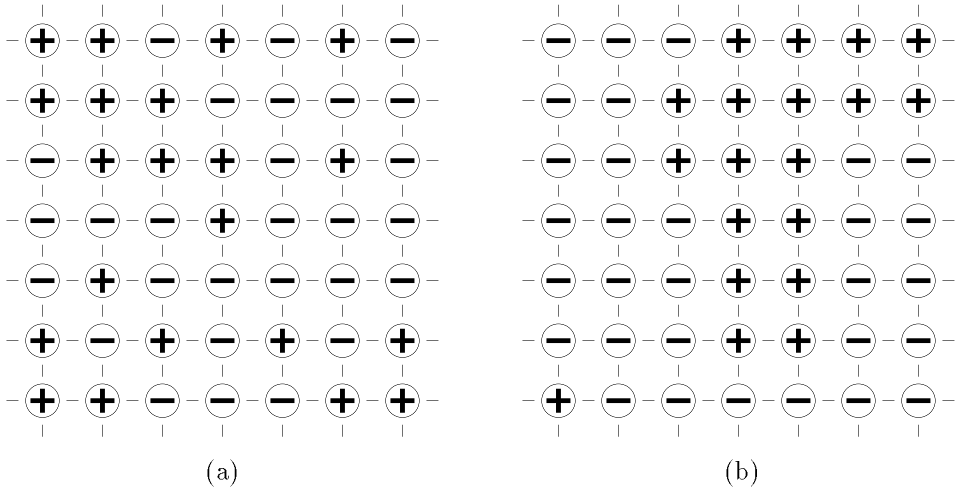


Figure 2.5: A two-dimensional Ising system. Spin variables are shown as circles, with lines connecting neighboring spins. A typical high-temperature state is shown in (a), while (b) shows a lower-temperature state, in which there are large clusters of like spins.

become negligible, allowing their average values to be identified with the apparently stable values obtained from macroscopic observations.

Example: The 2D Ising model. The *Ising* model of ferromagnetism is a much-studied system with a discrete microstate (see (Cipra, 9:1987) for a tutorial). A two-dimensional Ising system consists of a 2D array of "spin" variables, S_i, taking on values of $+1$ or -1, as illustrated in Figure 2.5. The system's energy is given by

$$E(s) \;=\; -\sum_{(i,j)\in\mathcal{N}} J s_i s_j \;-\; \sum_i H s_i \tag{2.55}$$

where $\mathcal{N}$ is the set of pairs of indexes for nearest vertical and horizontal neighbors in the 2D array. The constant J controls how strong a tendency there is for neighboring spins to be the same. H gives the strength of an external magnetic field that biases the spins in one direction or the other. The overall "magnetization" of the system is

$$M(s) \;=\; \sum_i s_i \tag{2.56}$$

Only systems of finite size can be simulated on a computer (a toroidal topology is generally used, to avoid boundaries), but it is the extrapolation of the results to the thermodynamic limit that is of interest. These system exhibits a phase transition at a critical temperature. Above this temperature, there are local regions of matched spins, but no long-range order, and hence there is zero magnetization when the external field is zero; below the critical temperature, over half the spins in the entire system acquire a consistent orientation, producing a non-zero magnetization even when the external field is zero.

The 2D Ising model happens to be the same as the simplest of the image models investigated by Geman and Geman (4:1984) and others. Here, the "spins" are interpreted as pixels of a black and white image; their interaction in the energy function models the tendency of images to contain large black and large white areas. Generalized to allow J and H to vary from spin to spin, and to allow interactions between any two spins, the Ising model becomes the "Boltzmann machine" of Ackley, Hinton, and Sejnowski (2:1985).

Example: Lennard-Jonesium. A simple model of molecular interactions based on the *Lennard-Jones potential* has also been the subject of numerous computer investigations, starting in the early days of the Monte Carlo method (Wood and Parker, 4:1957). The state of this system is composed of position vectors, $\mathbf{q}_i$, and momentum vectors, $\mathbf{p}_i$, for some number of molecules of a hypothetical substance sometimes dubbed "Lennard-Jonesium", which resembles argon. As is typical in statistical physics, interest centres on systems with very large numbers of molecules, though only much smaller systems can actually be simulated.

The energy for the system, denoted by H in this context, is

$$H(\mathbf{q},\mathbf{p}) \;=\; \sum_{i\neq j} 4\varepsilon \left[\left(\frac{\sigma}{|\mathbf{q}_i-\mathbf{q}_j|}\right)^{12} - \left(\frac{\sigma}{|\mathbf{q}_i-\mathbf{q}_j|}\right)^{6} \right] + \sum_i \frac{|\mathbf{p}_i|^2}{2\mu} \tag{2.57}$$

where ε, σ, and μ are arbitrary positive constants. The first term in the expression for H is known as the *potential energy*; it depends only on the positions of the molecules, and will be denoted by $E(\mathbf{q})$. The second term is the *kinetic energy*; it depends only on the momenta of the molecules, and will be denoted by $K(\mathbf{p})$. The form of the potential energy is designed so that nearby molecules will be attracted to each other, under the influence of the 6-th power term, but will not be able to approach too closely, due to the 12-th power term. Distant molecules have little effect on each other.

The canonical distribution for the complete state, $(\mathbf{q},\mathbf{p})$, will be

$$P(\mathbf{q},\mathbf{p}) \;=\; \frac{1}{Z_H}\exp\bigl(-H(\mathbf{q},\mathbf{p})/T\bigr) \tag{2.58}$$

$$\;=\; \left[\frac{1}{Z_E}\exp\bigl(-E(\mathbf{q})/T\bigr)\right] \cdot \left[\frac{1}{Z_K}\exp\bigl(-K(\mathbf{p})/T\bigr)\right] \tag{2.59}$$

The distributions for $\mathbf{q}$ and for $\mathbf{p}$ are thus independent. That for $\mathbf{p}$ is simply a multivariate Gaussian, and can be dealt with analytically. Consequently, Monte Carlo simulation is often used only to sample from the distribution for $\mathbf{q}$, determined by the potential energy, $E(\mathbf{q})$.

Eliminating aspects of the problem that can be solved analytically is a generally useful technique. Interestingly, though, we will see later that it can also be computationally useful to introduce extra variables such as $\mathbf{p}$.

Free energy and phase transitions. The normalization factor, Z, for the canonical distribution of equation (2.54) is known as the *partition function*. (As written, it appears to be a constant, but it becomes a function if one considers varying T, or the environmental variables that enter implicitly into $E(s)$.) The *free energy* of the system is defined as $F = -T\log(Z)$. The following relationship is easily derived:

$$F \;=\; \langle E\rangle - TS \tag{2.60}$$

where $\langle E\rangle$ is the expectation of the energy, and $S = -\sum_{\tilde{s}} P(\tilde{s})\log(P(\tilde{s}))$ is the *entropy*. The free energy and entropy are extensive quantities.[3]

[3] I am here ignoring certain distinctions that would be important for physical applications. For example, one clearly gets different "free energies" for Lennard-Jonesium depending on whether one looks at the space of both position and momentum coordinates, and uses the total energy, $H(\mathbf{q},\mathbf{p})$, or instead looks only at the position coordinates, and uses only the potential energy, $E(\mathbf{q})$. Additional complications arise from the fact that the molecules are not distinguishable.

These closely related quantities play important roles in statistical physics. In particular, *phase transitions* such as melting and boiling, which occur as the temperature or other environmental variables change, can be identified by discontinuities in the derivatives of the free energy per unit of system size, in the thermodynamic limit. Much effort has been devoted to overcoming the unfortunate fact that it is the behaviour in the vicinity of such phase transitions that is both of the greatest scientific interest and also the most difficult to elucidate using Monte Carlo simulations. In particular, calculation of the free energy is not straightforward, and usually requires a whole series of Monte Carlo runs.

Correspondence with probabilistic inference. To relate the formalism of statistical physics to problems of probabilistic inference, one need only regard the joint values of the random variables in a probabilistic inference problem as possible microstates of an imaginary physical system. Note that *any* probability distribution over these variables that is nowhere zero can be considered a canonical distribution (equation (2.54)) with respect to an energy function $E(s) = -T\log(P(s)) - T\log(Z)$, for any convenient choice of Z and T. States with zero probability can be accommodated if the energy is allowed to be infinite. Usually, we set $T = 1$, and drop it from the equations.

In particular, Bayesian inference for model parameters can be represented by an imaginary physical system in which the microstate corresponds to the set of unknown parameters, $\theta_1, \ldots, \theta_p$. Given a training set of complete observations, $x_1, \ldots, x_C$, the following energy function is generally easy to compute:

$$E_C(\theta) \;=\; -\log\big(P(\theta)\,P(x_1,\ldots,x_C \mid \theta)\big) \;=\; -\log\left(P(\theta)\prod_{i=1}^{C} P(x_i \mid \theta)\right) \tag{2.61}$$

Taking $T = 1$, the partition function will then be

$$Z_C \;=\; \int \exp\big(-E_C(\tilde\theta)\big)\,d\tilde\theta \;=\; \int P(\tilde\theta)\prod_{i=1}^{C} P(x_i \mid \tilde\theta)\,d\tilde\theta \;=\; P(x_1,\ldots,x_C) \tag{2.62}$$

The canonical distribution for this system (equation (2.54)) is identical to the posterior parameter distribution (equation (2.31)):

$$P(\theta \mid x_1,\ldots,x_C) \;=\; \frac{P(\theta)\prod_{i=1}^{C} P(x_i \mid \theta)}{P(x_1,\ldots,x_C)} \;=\; \frac{1}{Z_C}\exp\big(-E(\theta)\big) \tag{2.63}$$

Predicting future observations by calculating expectations with respect to this posterior (as in equation (2.34)) is thus reduced to finding expectations for functions of the microstate of this imaginary physical system.

The value of the partition function, Z_C, is the probability of the training data given the model being considered. This is the crucial quantity needed for Bayesian model comparison using equation (2.50). The problem of model comparison is thus reduced to that of calculating the partition function for a physical system, or equivalently, the free energy or the entropy. The partition function can also be used to express predictive probabilities, as follows:

$$P(x_{C+1} \mid x_1,\ldots,x_C) \;=\; \frac{P(x_1,\ldots,x_C,x_{C+1})}{P(x_1,\ldots,x_C)} \;=\; \frac{Z_{C+1}}{Z_C} \tag{2.64}$$

Calculating predictive probabilities this way, using the methods for estimating Z_{C+1}/Z_C described in Section 6.2, will be preferable to straightforward Monte Carlo estimation of $\langle P(x_{C+1} \mid \theta)\rangle_C$ whenever the value x_{C+1} is most likely to occur in conjunction with values of

θ that have low probability, and hence would seldom be visited during a standard simulation. This will sometimes be the case for rare events.

The connections between probabilistic inference and statistical physics have been noted by several authors. They motivated Ackley, Hinton, and Sejnowski (2:1985) to give the name "Boltzmann machine" to the probabilistic neural network they developed. More recently, methods from statistical physics have been used to analyse the generalization capabilities of neural networks (for a review, see (Watkin, Rau, and Biehl, 2:1993)). For this work, it is useful to define statistical problems of indefinitely large "size" in a way that leads to an appropriate "thermodynamic limit" as the size increases. This can be done by letting both the number of cases in the training set and the number of variables in the model increase in tandem with system size. Empirical work using Markov chain Monte Carlo methods has been a useful adjunct to the theoretical work done in this framework (see, for example, (Seung, Sompolinsky, and Tishby, 2:1992)).

3. Background on the Problem and its Solution

Three tasks involving complex distributions were described in the preceding section: probabilistic inference for unobserved variables using a fully-specified model, Bayesian statistical inference for unknown model parameters based on training data, and simulation of physical systems with a given energy function. In this section, I more explicitly characterize problems of this sort, and discuss why simple approaches to solving them are not adequate. I also present the essential theory of Markov chains needed for the development of the Monte Carlo methods that we hope will be able to solve these problems.

3.1 Definition of the problem

The problems that we will principally address take the form of finding the expectation of some function with respect to a probability distribution on some discrete or continuous space. Examples include the computation of conditional probabilities as expressed in equations (2.8) and (2.9), and of predictive probabilities given by equation (2.34).

The problem of calculating an expectation. To formalize this class of problems, suppose we have a state variable, $X = \{X_1, \ldots, X_n\}$, whose components may be discrete, or continuous, or a mixture of both. The dimensionality, n, is often large — problems with a few hundred dimensions are typical, and those with a few million dimensions may be contemplated. Suppose that the probability distribution for this variable is given by an unnormalized probability mass/density function $f(x)$. Our goal is to find the expectation of $a(X)$ with respect to this probability distribution. For the discrete case:

$$\langle a \rangle \;=\; \sum_{\tilde{x}_1} \cdots \sum_{\tilde{x}_n} a(\tilde{x}_1, \ldots, \tilde{x}_n)\, P(\tilde{x}_1, \ldots, \tilde{x}_n) \tag{3.1}$$

$$\;=\; \frac{\sum_{\tilde{x}_1} \cdots \sum_{\tilde{x}_n} a(\tilde{x}_1, \ldots, \tilde{x}_n)\, f(\tilde{x}_1, \ldots, \tilde{x}_n)}{\sum_{\tilde{x}_1} \cdots \sum_{\tilde{x}_n} f(\tilde{x}_1, \ldots, \tilde{x}_n)} \tag{3.2}$$

If the components of X are continuous, the sums above will be replaced by integrals; if some components are continuous and some discrete, we will have a mixture of sums and integrals.

We assume that both $f(x)$ and $a(x)$ can feasibly be evaluated for any given value of x. For some of the algorithms discussed, we also assume that $f'(x)$ exists and can be computed. In many applications, however, evaluating $f(x)$ or $f'(x)$ takes a significant amount of time, so we will wish to minimize the number of such evaluations. When X is multidimensional, $f(x_1, \ldots, x_n)$ sometimes has a "local" structure that allows its value to be quickly re-computed after a change to only one of the x_i.

Problems of varying difficulty. For the problems we wish to address, $f(x)$ varies greatly, with most of the probability being concentrated in regions of the state space that occupy a tiny fraction of the whole, and whose location is not known *a priori*. This crucial characteristic means that any method of solution must somehow search for regions of high probability. The shape of the relevant regions is also important — a convoluted shape will require continual search, in order to find all its nooks and crannies.

Typically, we will require an estimate of $\langle a \rangle$ that satisfies an *absolute* error bound, i.e. that is likely to be within $\pm\varepsilon$ of the true value, for some ε that does not depend on the magnitude of $\langle a \rangle$. We will also assume that $\langle a \rangle$ is dominated by contributions from $a(x)P(x)$ in the region where $P(x)$ is large; this allows us to obtain a reasonable estimate of $\langle a \rangle$ by sampling from

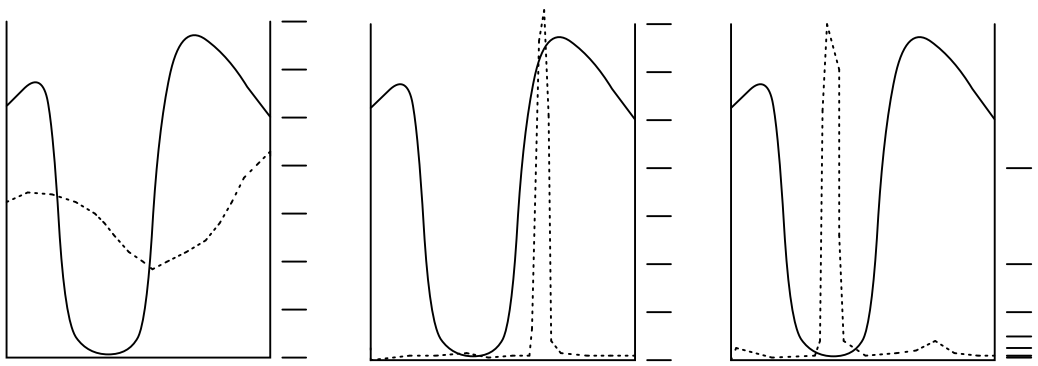


Figure 3.1: Three types of estimation problem. In each case, we wish to estimate the value of $\langle a \rangle = \int a(\tilde{x}) f(\tilde{x})\, d\tilde{x} \Big/ \int f(\tilde{x})\, d\tilde{x}$, where $a(x)$ is given by the solid line, and $f(x)$ by the dotted line. The required accuracy is indicated by the divisions to the right of each graph. In case (a), the probability is substantial over much of the domain, and $a(x)$ does not vary much in relation to the absolute error tolerance. The expectation could be adequately estimated by Monte Carlo integration using points drawn uniformly from the domain. This method would be inefficient for case (b), where the probability varies greatly. Estimation could in this case be done using points drawn from the distribution defined by $f(x)$. This would not work well for case (c), however, where the error tolerance is relative to the expectation value, and the dominant contribution comes from regions of low probability.

the distribution $P(x)$. In fact, the same sample can be used to estimate the expectation of a number of functions satisfying this requirement.

Problems of Bayesian model comparison and of computing the probabilities of rare events are of a different nature. In these cases, we require an estimate satisfying a *relative* error bound, and the variation in $a(x)P(x)$ may be dominated by regions where $P(x)$ is small. These problems are formally similar to that of estimating the free energy of a physical system; they are discussed in Section 6.2.

Figure 3.1 illustrates this range of estimation problems. In (a), the probability varies little across the domain, making a search for high-probability regions unnecessary. This problem is too easy to arouse our interest here. A problem more typical of those we are interested in is shown in (b) — though real problems would likely have more dimensions and exhibit more extreme variations. Monte Carlo methods based on a sample from the distribution defined by $f(x)$ will work well here. In (c), the primary contribution to $\langle a \rangle$ comes from regions where $f(x)$ is small. Using a sample from the distribution given by $f(x)$ would still be adequate for this problem if the error tolerance were as in (b). Here, however, the tolerance is relative to the value of $\langle a \rangle$. When most of the probability is in regions where $a(x)$ is nearly zero, this tolerance can become arbitrarily small, and the methods of Section 6.2 will be required.

For many problems of the sort shown in Figure 3.1(b) it is not possible to obtain a sample of *independent* points from the distribution defined by $f(x)$ — the most one can hope for is to obtain a sample of *dependent* points from a distribution that is *close* to that defined by $f(x)$, using Markov chain methods. In some cases, theoretical bounds on the convergence rate of the Markov chain may be obtainable — if not now, then with further research — and these could be used to guarantee that estimates found in this way will be approximately correct, with high probability. For the most challenging problems, involving elements of difficult combinatorial search, such guarantees will probably not be available. The probability distribution defined by $f(x)$ may in these cases not only be concentrated in a tiny volume of the parameter space, of unknown location, but also be distributed across this space in a complex pattern, perhaps as a huge number of separate peaks or ridges. This is a common situation in statistical physics. It is also seen when doing Bayesian statistical inference for a latent class model with many classes, for the multi-layer perceptron, and generally for any model that attempts to divine complex structure in the data, with many alternative structures working reasonably well.

A single realization of reasonable length of a Markov chain designed to sample from such a complex distribution may not cover anywhere near the entire region of high probability. To make inferences about $\langle a \rangle$ based on such a sample, we must implicitly rely on an assumption that $a(x)$ does not vary much from one "typical" region to another "typical" region, and hence a sample drawn from only one such region will be adequate. Reliance on this assumption can be eliminated by generating a number of independent realizations of the Markov chain, assuming that these chains truly do reach (or closely approach) the equilibrium distribution in the allotted time. Unfortunately, when dependencies within a single chain are high, it will generally also be difficult to confirm that equilibrium has been reached. These issues are discussed further in Section 6.3.

3.2 Approaches to solving the problem

Insight into the difficulty of these problems can be gained by considering various approaches to solving them. A number of methods for performing Bayesian statistical computations are reviewed by Smith (9:1991). Also of interest is the collection edited by Flournoy and Tsutakawa (9:1991) and the book by Tanner (9:1991). Many common computational methods are applicable only to easier problems than are addressed here, however, such as when the parameter space is of low dimension.

One way of classifying the methods to be considered is by the degree of search they employ — from no searching, to a single search at the beginning, to a continual search in the case of the Markov chain methods. I start with the first category.

Numerical methods. Perhaps the most obvious approach to solving the problem is direct numerical calculation. In particular, when the state space is finite, we could in theory perform the summations of equation (3.2) explicitly. For the problems we are interested in, however, this is computationally infeasible, since it would require time exponential in the dimensionality of X. When the state space is continuous, numerical evaluation of the corresponding integrals using a simple product rule would be similarly infeasible.

The method of numerical integration in high dimensional spaces recently developed by Woźniakowski (9:1991) also appears to be inapplicable. This method is good in an average sense, on the assumption that the functions to be integrated are drawn from a particular distribution. It appears that this distribution is not a good model for the class of problems discussed here, in which the integrand is close to zero over much of the space, but very much larger in a small region of unknown location.

Rejection sampling. We could estimate $\langle a \rangle$ of equation (3.2) by the Monte Carlo formula of equation (1.2) if we could only sample from the distribution defined by $f(x)$. Though we assume that this is not possible by direct methods, for some problems it may be possible to use the technique of *rejection sampling* (see, for example, (Ripley, 1:1987, Section 3.2) or (Devroye, 9:1986, Section II.3)) to produce a sample of independent points drawn from $f(x)$ by generating points from another, more tractable distribution, and then "rejecting" some points in order to produce a sample from $f(x)$.

To apply this method, we must be able to generate points from a distribution with density proportional to some function, $g(x)$, such that for some constant, c, we can guarantee that $f(x) \leq cg(x)$ for all x. To generate a point from the distribution defined by $f(x)$, we generate a point, x^*, from $g(x)$, and "accept" x^* as our generated point with probability $f(x)/cg(x)$. If we do not accept x^*, then we generate another such point from $g(x)$, repeating the procedure until a point is finally accepted.

One can prove that this procedure does indeed sample from exactly the distribution defined by $f(x)$. The efficiency of the procedure depends on how often points are rejected, which in turn depends on how well $g(x)$ approximates $f(x)$ — in the extreme case, if we had $g(x) = f(x)$, we could use $c = 1$ and never have to reject a point. For easy problems, we might hope to do only a small factor worse than this, but for complex problems, it may be impossible to find an appropriate function $g(x)$ and constant c for which we can prove that $f(x) \leq cg(x)$, except perhaps by chosing $g(x)$ to be very diffuse and c to be very large, which would lead to a very low acceptance rate.

Rejection sampling is therefore not a feasible method for the difficult problems treated in this review. However, "adaptive rejection sampling" can be a useful component of a Markov chain method, as is discussed in Section 4.1.

Simple importance sampling. *Importance sampling* is a another fundamental technique in Monte Carlo estimation, which we will later see has applications in connection with Markov chain sampling methods. However, simple importance sampling methods that do not incorporate any search for high probability regions also fail when applied to the problems addressed here.

To estimate an expectation using importance sampling, we first choose some probability mass/density function, $g(x)$, not necessarily normalized, from which we can easily sample, and which we hope approximates the distribution of interest. Unlike the case with rejection sampling, there are no absolute requirements for how well $g(x)$ approximates $f(x)$, except that $g(x)$ must not be zero anywhere $f(x)$ is non-zero. We can then express the expectation of $a(X)$ with respect to the distribution defined by $f(x)$, denoted by $\langle a \rangle$, in terms of expectations with respect to $g(x)$, denoted by $E_g[\,\cdot\,]$, as follows:

$$\langle a \rangle = \sum_{\tilde{x}} a(\tilde{x}) f(\tilde{x}) \Big/ \sum_{\tilde{x}} f(\tilde{x}) \tag{3.3}$$

$$= \sum_{\tilde{x}} a(\tilde{x}) \frac{f(\tilde{x})}{g(\tilde{x})} g(\tilde{x}) \Big/ \sum_{\tilde{x}} \frac{f(\tilde{x})}{g(\tilde{x})} g(\tilde{x}) \tag{3.4}$$

$$= E_g\Big[a(X)\frac{f(X)}{g(X)}\Big] \Big/ E_g\Big[\frac{f(X)}{g(X)}\Big] \tag{3.5}$$

Monte Carlo estimation of the expectations with respect to g gives the following (as in (Hastings, 4:1970)):

$$\langle a \rangle \approx \sum_{t=0}^{N-1} a(x^{(t)}) \frac{f(x^{(t)})}{g(x^{(t)})} \Big/ \sum_{t=0}^{N-1} \frac{f(x^{(t)})}{g(x^{(t)})} \tag{3.6}$$

where $x^{(0)}, \ldots, x^{(N-1)}$ are drawn from the distribution defined by g. The method averages the values of a at the sample points, weighting each value according to how the sampling distribution departs from the desired distribution at that point.

If $g = f$, equation (3.6) reduces to the simple Monte Carlo estimation formula of equation (1.2), but we assume that this choice is not available. If g is a good approximation to f, equation (1.2) will still yield a good estimate of $\langle a \rangle$. For the problems of interest, however, guessing such an approximation *a priori* is very difficult. If g is not close to f, the result will be poor, since it is then likely that only a few of the points selected will be in regions where f is large. The value of the weighting factor, $f(x^{(t)})/g(x^{(t)})$, for these points will be much larger than for the other points, effectively reducing the size of the sample to just these few points. Even worse, it could be that *none* of the points selected lie in regions where f is large. In this case, not only might the estimate of equation (3.6) be very inaccurate, the data themselves might provide no warning of the magnitude of the error.

Methods based on finding the mode. Clearly, any effective method of solving these problems must in some way search for the high-probability states. Perhaps the most straightforward way of doing this is to search once for a point where the probability is at least locally maximized, and then use some approximation around this mode to evaluate expectations of functions with respect to the original distribution.

One method used for continuous spaces approximates the distribution by a multivariate Gaussian centred on the mode found. This is equivalent to using a quadratic approximation for the log of the probability density, and hence requires that we evaluate the second derivatives of the log probability density at the mode (the *Hessian* matrix). At least for problems where the dimensionality of the space is only moderately high (a few hundred, say), the amount of computation required for this will often be tolerable.

Having found the Gaussian that approximates the distribution, we need then to evaluate the expectations of whatever functions are of interest. For functions that are approximately linear in the region where the probability density is significant, the expectation can be approximated as the value of the function at the mode. The distribution of the function value will be Gaussian, with a variance that can be calculated from the Hessian and the gradient of the function at the mode. The expectation of a non-linear function with respect to the approximating Gaussian can be evaluated by simple Monte Carlo methods, or by numerical integration.

Rather than simply accept whatever error is introduced by the Gaussian approximation, one can instead obtain unbiased estimates via importance sampling, as described above, using the approximating Gaussian density as the sampling distribution, g, of equation (3.5). An approximating distribution other than a Gaussian could also be used. In particular, the heavier tails of a Student t distribution may be beneficial.

There is no doubt that these methods are often useful. A theoretical reason for expecting this to be so is that for many statistical inference problems the posterior parameter distribution approaches a Gaussian distribution asymptotically, as the size of the training set increases. Nevertheless, methods based on finding the mode are not universally applicable.

One obvious difficulty is that the distribution of interest may have more than one mode. It is also quite possible that the mode or modes are not at all representative of the distribution as a whole. This is well illustrated by statistical physics. The lowest energy, maximum probability, molecular configuration for a substance is generally a perfect crystal. This is not, however, a good starting point for determining the properties of the substance at a temperature above its melting point, where an enormous number of more disordered configurations of somewhat higher energy overwhelm the influence of the few highly-ordered configurations. Analogous effects are often important in Bayesian inference.

The asymptotically Gaussian form of the posterior parameter distribution for a particular model is also perhaps not as relevant as might appear. If we wish to obtain the maximum benefit from a large data set, we should not stick to simple models, where this Gaussian form may have been reached, but should consider more complex models as well, up to the point where increasing the complexity gives no further return with the amount of data available. At this point, the Gaussian approximation is unlikely to be adequate, and the posterior distribution may well have a shape that can be adequately explored only by Monte Carlo methods based on Markov chains.

On the other hand, the methods discussed in this section have the considerable advantage that the "free energy" needed for model comparison can be computed with no more effort than is needed to find expectations, in sharp contrast to the Markov chain methods.

Example: Multi-layer perceptrons. MacKay (2:1991, 2:1992b) has developed an approach to Bayesian learning for multi-layer perceptrons based on Gaussian approximations, which has been further extended and applied to practical problems by Thodberg (2:1993) and MacKay (2:1993). Buntine and Weigend (2:1991) discuss a related approach.

One problem with using a Gaussian approximation for this task is that the posterior distribution for the network weights usually has a large number of modes. MacKay handles this situation by finding many (though nowhere near all) modes, using many runs of the optimization procedure, and then selecting the best of these by treating the region of parameter space in the vicinity of each mode as a separate "model".

The hyperparameters controlling the distribution of the weights (σ_u and σ_v in equation (2.42)) present a different problem — the posterior distribution of the weights together with these hyperparameters is not close to being Gaussian. MacKay therefore employs the Gaussian approximation only for the distribution of the weights conditional on fixed values of the hyperparameters. Different hyperparameter values are treated much as different models would be, with those hyperparameter values being selected that lead to the highest probability for the training data, after integrating over the weight space. (This is a slight departure from the true Bayesian solution, in which the hyperparameters would be integrated over as well.)

It is interesting how MacKay compensates for the weaknesses of the Gaussian approximation method by exploiting its strength in model comparison. It is certainly not clear that this is always possible, however.

More sophisticated methods. We have seen that it can be hopeless to try to evaluate expectations with respect to a complex distribution without searching for regions of high probability, and that methods based on searching once for a mode and then approximating the distribution in its vicinity have limitations. Monte Carlo methods based on Markov chains can be viewed as combining sampling with a continual search for large regions of high probability, in a framework that is guaranteed to produce the correct distribution in the limit as the length of the chain increases.

Other approaches involving a more sophisticated search have been tried. For example, Evans (9:1991) describes an adaptive version of importance sampling. In this method, a class of importance sampling functions thought to contain one appropriate for the problem is defined, and an initial function from within the class is chosen. Various characteristics of the true distribution that can be expressed as expectations are then estimated using this importance sampler, and a new importance sampler from within the class is chosen that matches these characteristics. This procedure is iterated until it stabilizes, and the final importance sampler is then used to estimate the expectations of interest.

If the initial importance sampler is sufficiently bad, this adaptive procedure may not work. To handle this, Evans proposes "chaining" through a series of problems. To start, it is assumed that a good importance sampler can be found for an easy problem. This easy problem is then transformed into the problem of interest in small steps. At each step, a good importance sampler is found by the adaptive method, using the importance sampler found for the previous step as the starting point. This technique resembles simulated annealing (see Section 6.1) and some methods used in free energy estimation (see Section 6.2).

It seems likely that adaptive methods of this sort will perform better than Markov chain methods on at least some problems of moderate difficulty. My principal focus in this review is on the most difficult problems, for which, I believe, the methods based on Markov chains are at present the only feasible approach.

3.3 Theory of Markov chains

I present here the essential theory required in developing Monte Carlo methods based on Markov chains. The most fundamental result, which is here given a simple proof, is that certain Markov chains converge to a unique invariant distribution, and can be used to estimate expectations with respect to this distribution.

The theory of Markov chains is well developed; references to a few of the books in the area may be found in Section 3 of the bibliography. Much of the elaboration of the theory can be avoided for our purposes, however, since we are not interested in discovering the properties of some arbitrary given Markov chain, but rather in constructing a Markov chain having the properties we desire.

Basic definitions. A *Markov chain* is a series of random variables, $X^{(0)}, X^{(1)}, X^{(2)}, \ldots$, in which the influence of the values of $X^{(0)}, \ldots, X^{(n)}$ on the distribution of $X^{(n+1)}$ is mediated entirely by the value of $X^{(n)}$. More formally,

$$P(x^{(n+1)} \mid x^{(n)}, \{x^{(t)} : t \in \mathcal{E}\}) \quad = \quad P(x^{(n+1)} \mid x^{(n)}) \tag{3.7}$$

where $\mathcal{E}$ is any subset of $\{0, \ldots, n-1\}$. The indexes, $t = 0, 1, 2, \ldots$, are often viewed as representing successive "times". The $X^{(t)}$ have a common range, the *state space* of the Markov chain. I will for the moment assume that the state space is finite, but countably infinite and continuous state spaces will be discussed later.[4]

A Markov chain can be specified by giving the marginal distribution for $X^{(0)}$ — the *initial probabilities* of the various states — and the conditional distributions for $X^{(n+1)}$ given the possible values for $X^{(n)}$ — the *transition probabilities* for one state to follow another state. I will write the initial probability of state x as $p_0(x)$, and the transition probability for state

[4]A continuous "time" parameter is accommodated by the more general notion of a *Markov process*. Here, in common with many others, I use the term "Markov chain" to refer to a Markov process with a discrete time parameter, a nomenclature that makes good metaphorical sense. Unfortunately, there are other authors who use the term to refer to a Markov process with a discrete state space, and still others who use it to refer to what is generally known as a "homogeneous" Markov chain (see below).

x' at time $n+1$ to follow state x at time n as $T_n(x, x')$. If the transition probabilities do not depend on the time, the Markov chain is said to be *homogeneous* (or *stationary*) and the transition probabilities will be written simply as $T(x, x')$.

Using the transition probabilities, one can find the probability of state x occurring at time $n+1$, denoted by $p_{n+1}(x)$, from the corresponding probabilities at time n, as follows:

$$p_{n+1}(x) \;=\; \sum_{\tilde{x}} p_n(\tilde{x})\, T_n(\tilde{x}, x) \tag{3.8}$$

Given the initial probabilities, p_0, this determines the behaviour of the chain at all times.

The probabilities at time n can also be regarded as a row vector, $\mathbf{p}_n$, and the transition probabilities at time n as a matrix, $\mathbf{T}_n$, or just $\mathbf{T}$, if the chain is homogeneous. (Such matrices, in which all elements are non-negative, and rows sum to one, are call *stochastic* matrices.) Equation (3.8) can then be written as $\mathbf{p}_{n+1} = \mathbf{p}_n \mathbf{T}_n$. For a homogeneous chain, $\mathbf{T}^k$ (the k-th power of the matrix $\mathbf{T}$) gives the "k-step" transition probabilities, which will also be written $T^k(x, x')$, and we will have $\mathbf{p}_n = \mathbf{p}_0 \mathbf{T}^n$.

Invariant distributions. An *invariant* (or *stationary*) distribution over the states of a Markov chain is one that persists forever once it is reached. More formally, the distribution given by the probabilities $\pi(x)$ is invariant with respect to the Markov chain with transition probabilities $T_n(x, x')$ if, for all n,

$$\pi(x) \;=\; \sum_{\tilde{x}} \pi(\tilde{x})\, T_n(\tilde{x}, x) \tag{3.9}$$

Equivalently, the vector $\boldsymbol{\pi}$ represents an invariant distribution if and only if $\boldsymbol{\pi} = \boldsymbol{\pi}\mathbf{T}_n$ for all n. For a homogeneous chain, of course, we have just the single condition that $\boldsymbol{\pi} = \boldsymbol{\pi}\mathbf{T}$. A Markov chain can have more than one invariant distribution. If $\mathbf{T}$ is the identity matrix, for example, then any distribution is invariant. A finite Markov chain always has at least one invariant distribution.

We are interested in constructing Markov chains for which the distribution we wish to sample from, given by π, is invariant. Often, we will use *time reversible* homogeneous Markov chains that satisfy the more restrictive condition of *detailed balance* — that if a transition occurs from a state picked according to the probabilities given by π, then the probability of that transition being from state x to state x' is the same as the probability of it being from state x' to state x. In other words, for all x,

$$\pi(x)\, T(x, x') \;=\; \pi(x')\, T(x', x) \tag{3.10}$$

This implies that π is an invariant distribution, since

$$\sum_{\tilde{x}} \pi(\tilde{x})\, T(\tilde{x}, x) \;=\; \sum_{\tilde{x}} \pi(x)\, T(x, \tilde{x}) \;=\; \pi(x) \sum_{\tilde{x}} T(x, \tilde{x}) \;=\; \pi(x) \tag{3.11}$$

It is possible for a distribution to be invariant without detailed balance holding. For example, the uniform distribution on the state space $\{0, 1, 2\}$ is invariant with respect to the homogeneous Markov chain with transition probabilities $T(0,1) = T(1,2) = T(2,0) = 1$ and all others zero, but detailed balance does not hold.

Ergodic Markov chains. For our purposes, it is not enough merely to find a Markov chain with respect to which the distribution we wish to sample from is invariant. We also require that the Markov chain be *ergodic* — that the probabilities at time n, $p_n(x)$, converge to this invariant distribution as $n \to \infty$, regardless of the choice of initial probabilities $p_0(x)$. Clearly, an ergodic Markov chain can have only one invariant distribution, which is also referred to as its *equilibrium* distribution. Some Markov chains "converge" not to a single distribution, but rather to a cycle of distributions. These *periodic* chains are not ergodic by this definition.[5]

The question of when a Markov chain is ergodic has been asked and answered in many different ways in the literature, with most ways employing a fair amount of theoretical apparatus. I give here a simple proof that a large class of Markov chains are ergodic. (An elementary proof of a similar result can also be found in (Kemeny and Snell, 3:1960, Theorem 4.1.4.))

FUNDAMENTAL THEOREM. *If a homogeneous Markov chain on a finite state space with transition probabilities $T(x, x')$ has π as an invariant distribution and*

$$\nu \;=\; \min_x \min_{x':\pi(x')>0} T(x,x')/\pi(x') \;>\; 0 \tag{3.12}$$

then the Markov chain is ergodic, i.e., regardless of the initial probabilities, $p_0(x)$

$$\lim_{n\to\infty} p_n(x) \;=\; \pi(x) \tag{3.13}$$

for all x. A bound on the rate of convergence is given by

$$|\pi(x) - p_n(x)| \;\leq\; (1-\nu)^n \tag{3.14}$$

Furthermore, if $a(x)$ is any real-valued function of the state, then the expectation of a with respect to the distribution p_n, written $E_n[a]$, converges to its expectation with respect to π, written $\langle a \rangle$, with

$$|\langle a \rangle - E_n[a]| \;\leq\; (1-\nu)^n \max_{x,\,x'} |a(x) - a(x')| \tag{3.15}$$

PROOF. The basis of the proof is a demonstration that the distribution at time n can be expressed as a "mixture" of the invariant distribution and another arbitrary distribution, with the invariant distribution's proportion of the mixture approaching one as n approaches infinity. This growth occurs because the invariant portion can never shrink, while the non-invariant portion keeps producing extra invariant bits, due to condition (3.12).

Specifically, we will see that the distribution at time n can be written as

$$p_n(x) \;=\; [1-(1-\nu)^n]\,\pi(x) \;+\; (1-\nu)^n\, r_n(x) \tag{3.16}$$

with r_n being a valid probability distribution. Note that $\nu \leq 1$, since we cannot have $\pi(x') < T(x,x')$ for all x'. The above formula can be satisfied for $n = 0$ — just set $r_0(x) = p_0(x)$. If it holds for $n = \bar{n}$, then

$$p_{\bar{n}+1}(x) \;=\; \sum_{\tilde{x}} p_{\bar{n}}(\tilde{x})\, T(\tilde{x}, x) \tag{3.17}$$

$$= \; [1-(1-\nu)^{\bar{n}}] \sum_{\tilde{x}} \pi(\tilde{x})\, T(\tilde{x},x) \;+\; (1-\nu)^{\bar{n}} \sum_{\tilde{x}} r_{\bar{n}}(\tilde{x})\, T(\tilde{x},x) \tag{3.18}$$

[5] The reader should be warned that various definitions of the term "ergodic" and its qualified forms are current. To some authors, the defining property is that the initial state is eventually forgotten. To others, it is that long time averages converge independently of the initial state. Above, I have required that the distribution at a single time converge independently of the initial state. In general, none of these are equivalent. Some definitions also exclude chains having "transient" states with zero equilibrium probability.

$$= \; [1-(1-\nu)^{\bar{n}}]\,\pi(x) \;+\; (1-\nu)^{\bar{n}} \sum_{\tilde{x}} r_{\bar{n}}(\tilde{x})\,\big[T(\tilde{x},x) - \nu\pi(x) + \nu\pi(x)\big] \tag{3.19}$$

$$= \; [1-(1-\nu)^{\bar{n}}]\,\pi(x) \;+\; (1-\nu)^{\bar{n}}\,\nu\,\pi(x) + (1-\nu)^{\bar{n}} \sum_{\tilde{x}} r_{\bar{n}}(\tilde{x})\,\big[T(\tilde{x},x) - \nu\pi(x)\big] \tag{3.20}$$

$$= \; [1-(1-\nu)^{\bar{n}+1}]\,\pi(x) \;+\; (1-\nu)^{\bar{n}+1} \sum_{\tilde{x}} r_{\bar{n}}(\tilde{x})\,\frac{T(\tilde{x},x) - \nu\pi(x)}{1-\nu} \tag{3.21}$$

$$= \; [1-(1-\nu)^{\bar{n}+1}]\,\pi(x) \;+\; (1-\nu)^{\bar{n}+1}\, r_{\bar{n}+1}(x) \tag{3.22}$$

where $r_{\bar{n}+1}(x) = \sum_{\tilde{x}} r_{\bar{n}}(\tilde{x})\,[T(\tilde{x},x) - \nu\pi(x)]/(1-\nu)$. From (3.12), we find that $r_{\bar{n}+1}(x) \geq 0$ for all x. One can also easily show that $\sum_x r_{\bar{n}+1}(x) = 1$. The $r_{\bar{n}+1}(x)$ therefore define a probability distribution, establishing (3.16) for $n = \bar{n}+1$, and, by induction, for all n.

Using (3.16), we can now show that (3.14) holds:

$$\begin{aligned}
|\pi(x) - p_n(x)| \;&=\; |\,\pi(x) \;-\; [1-(1-\nu)^n]\,\pi(x) \;-\; (1-\nu)^n\, r_n(x)\,| && (3.23)\\
&=\; |\,(1-\nu)^n\,\pi(x) \;-\; (1-\nu)^n\, r_n(x)\,| && (3.24)\\
&=\; (1-\nu)^n\,|\pi(x) - r_n(x)| && (3.25)\\
&\leq\; (1-\nu)^n && (3.26)
\end{aligned}$$

We can show (3.15) similarly:

$$\begin{aligned}
|\langle a\rangle - E_n[a]| \;&=\; \big|\,\textstyle\sum_{\tilde{x}} a(\tilde{x})\,\pi(\tilde{x}) \;-\; \sum_{\tilde{x}} a(\tilde{x})\,p_n(\tilde{x})\,\big| && (3.27)\\
&=\; \big|\,\textstyle\sum_{\tilde{x}} a(\tilde{x})\,[(1-\nu)^n\pi(\tilde{x}) \;-\; (1-\nu)^n r_n(\tilde{x})]\,\big| && (3.28)\\
&=\; (1-\nu)^n\,\big|\textstyle\sum_{\tilde{x}} a(\tilde{x})\,\pi(\tilde{x}) \;-\; \sum_{\tilde{x}} a(\tilde{x})\,r_n(\tilde{x})\,\big| && (3.29)\\
&\leq\; (1-\nu)^n \max_{x,\,x'} |a(x) - a(x')| && (3.30)
\end{aligned}$$

This completes the proof.

As phrased, the above theorem applies only to homogeneous Markov chains. Many of the algorithms we will discuss are based on Markov chains that are not homogeneous. They are, however, of a simple cyclic type, in which the transition matrices repeat after some period, d, with $\mathbf{T}_n = \mathbf{T}_{n+d}$. If we look at the state only at times that are multiples of d, we will thus see a homogeneous Markov chain, with transition matrix $\mathbf{T}_0\mathbf{T}_1\cdots\mathbf{T}_{d-1}$. We can then try to show that this chain converges to the desired distribution. If the desired distribution is also invariant with respect to all the $\mathbf{T}_n$ individually, we can use states from any of the times in computing Monte Carlo estimates.

Similarly, it may be that for a homogeneous Markov chain, condition (3.12) does not hold for the one-step transition probabilities, T, but does hold for the k-step transition probabilities, T^k. This is sufficient to guarantee convergence, at a rate bounded as in (3.14) and (3.15), but with the exponent n replaced by $\lfloor n/k \rfloor$.

The theorem as stated guarantees only that at large times the distribution will be close to the invariant distribution. It does not say how dependent the states at different times might be, and hence does not guarantee that the average value of a function over a long period of time converges to the function's expected value. In fact, however, the state of an ergodic Markov chain at time m will be nearly independent of the state at time n when $m \gg n$. To see this, imagine restarting the chain at time n from each of the possible states. Since the chain is ergodic, we will in each case have nearly reached the invariant distribution by time m, showing that the distribution at time m is (almost) independent of whatever state the chain was in at time n. It is therefore valid to use a Monte Carlo estimation formula such as equation (1.2) with a series of values taken from a single realization of an ergodic Markov chain with the desired invariant distribution. For a detailed proof of a similar result, see (Kemeny and Snell, 3:1960, Theorem 4.2.1).

The amount of computational effort required to produce a good Monte Carlo estimate using the states generated by a Markov chain will depend on three factors: first, the amount of computation required to simulate each transition; second, the time for the chain to converge to the equilibrium distribution, which gives the number of states that must be discarded from the beginning of the chain; third, the number of transitions needed to move from one state drawn from the equilibrium distribution to another state that is almost independent, which determines the number of states taken from the chain at equilibrium that are needed to produce an estimate of a given accuracy. The latter two factors are related — as we have just seen, fast convergence to the equilibrium distribution from any state guarantees fast movement from one state to an independent state. In practice, though, we chose the initial state from some particular distribution, so it is possible for the time required to reach equilibrium to be less than the time required to move about the distribution once equilibrium is reached; the reverse situation is also possible. These issues are discussed further in Section 6.3.

Unfortunately, while it will often be easy to show that a Markov chain we are interested in is ergodic by showing that ν in equation (3.12) is greater than zero, the readily obtainable lower bounds on ν are generally so small that the guarantees on the the rates of convergence given by (3.14) and (3.15) will be of no practical use. More sophisticated theoretical analysis, or, failing that, empirical testing, are therefore essential in applications of Markov chain sampling.

Analysing convergence using eigenvalues. Convergence results similar to that proved above can be shown in a number of other ways as well. Here, I will briefly describe a method using the eigenvalues of the transition matrix that is often seen in the literature. Further details on this and other classical approaches can be found in (Iosifescu, 3:1980).

A *(left) eigenvector* of a matrix, $\mathbf{M}$, is a non-zero row vector, $\mathbf{v}$, that satisfies $\mathbf{vM} = \lambda\mathbf{v}$, for some (possibly complex) number, λ, the *eigenvalue* associated with $\mathbf{v}$. The vector of probabilities, $\boldsymbol{\pi}$, for an invariant distribution of a homogeneous Markov chain is clearly an eigenvector of its transition matrix, $\mathbf{T}$, with eigenvalue one, since $\boldsymbol{\pi}\mathbf{T} = \boldsymbol{\pi}$.

For most stochastic matrices, one can find a complete set of linearly independent eigenvectors. (In this informal presentation, I will ignore the exceptions, matrices that are "defective".) These eigenvectors can be chosen so that they fall into three classes, as follows:

1) One or more eigenvectors with elements that are real, positive, and sum to one, associated with the eigenvalue one.

2) Eigenvectors other than the above that are associated with eigenvalues of magnitude one.

3) Eigenvectors with elements that sum to zero that are associated with eigenvalues of magnitude less than one.

The subspace spanned by the eigenvectors in class (1) contains the invariant distributions. We are interested in the case where there is only one invariant distribution, and hence only one such eigenvector. When eigenvectors in class (2) are present, the Markov chain is periodic. We wish to exclude these chains as well.

We are left with the case of a single eigenvector with eigenvalue one, giving the probabilities for the invariant distribution, and a number of eigenvectors with eigenvalues of magnitude less than one. We can order the eigenvalues as $1 > |\lambda_2| \geq |\lambda_3| \geq \cdots$, with the associated eigenvectors being $\boldsymbol{\pi}, \mathbf{v}_2, \mathbf{v}_3, \ldots$ Using these eigenvectors as a basis, we can write the vector representing the initial probability distribution for the Markov chain as

$$\mathbf{p}_0 \;=\; \boldsymbol{\pi} + a_2\mathbf{v}_2 + a_3\mathbf{v}_3 + \cdots \tag{3.31}$$

The distribution over states after the n-th step of the Markov chain will then be

$$\mathbf{p}_n \;=\; \mathbf{p}_0\mathbf{T}^n \;=\; \boldsymbol{\pi}\mathbf{T}^n + a_2\mathbf{v}_2\mathbf{T}^n + a_3\mathbf{v}_3\mathbf{T}^n + \cdots \tag{3.32}$$

$$\;=\; \boldsymbol{\pi} + \lambda_2^n a_2\mathbf{v}_2 + \lambda_3^n a_3\mathbf{v}_3 + \cdots \tag{3.33}$$

As n increases, $\mathbf{p}_n$ will therefore converge to $\boldsymbol{\pi}$, with the asymptotic rate of convergence being determined by the magnitude of the second-largest eigenvalue, λ_2.

The eigenvalues of the transition matrix of a reversible Markov chain are guaranteed to be real. For this case, a non-asymptotic bound on the time required for convergence, independently of the initial distribution, has been given by Sinclair (1:1993, Proposition 2.1):

$$\max_x \frac{|p_n(x) - \pi(x)|}{\pi(x)} \;\leq\; \frac{|\lambda_2|}{\min\limits_x \pi(x)} \tag{3.34}$$

If $T(x,x) \geq 1/2$ for all x, the eigenvalues will all be non-negative, and λ_2 will be just the largest eigenvalue less than one.

Recent work on analysing convergence. Recently, methods for bounding the convergence rate of a Markov chain have been developed that allow useful convergence guarantees to be proved for some of the complex Markov chains that arise in practical sampling problems.

The work of Sinclair and Jerrum (7:1989) is representative; references to other recent work along the same lines may be found in Section 3 of the bibliography. Sinclair and Jerrum define the *conductance* of a homogeneous, reversible Markov chain with equilibrium state probabilities, $\pi(x)$, that are nowhere zero to be

$$\Phi \;=\; \min_{S\,:\,0<\pi(S)\leq 1/2} \left[\frac{1}{\pi(S)} \sum_{x\in S,\, x'\notin S} \pi(x)\,T(x,x')\right] \tag{3.35}$$

where $\pi(S) = \sum_{x\in S}\pi(x)$. The quantity in brackets is just the probability of the chain exiting S in one step if it is started in a state picked from S according to the equilibrium probabilities. Sinclair and Jerrum then show, by bounding the magnitude of the second-largest eigenvalue of the transition matrix, that regardless of the initial distribution for the chain, its convergence rate is bounded as follows:

$$\max_x \frac{|p_n(x) - \pi(x)|}{\pi(x)} \;\leq\; \frac{\left(1 - \Phi^2/2\right)^n}{\min\limits_x \pi(x)} \tag{3.36}$$

provided that $T(x,x) \geq 1/2$ for all x. (Any chain can easily be modified to satisfy this requirement, by at each step having it repeat the current state with probability $1/2$, and otherwise do whatever it normally would.)

To obtain from this a useful bound on the convergence rate of a Markov chain with complex structure, one must be able to bound the conductance of the chain. Jerrum and Sinclair (4:1989) show how this can sometimes be done for chains of practical interest by defining "canonical paths" connecting every pair of states, with the path from $\bar{x}$ to $\bar{x}'$ being given a "weight" of $\pi(\bar{x})\pi(\bar{x}')$. Suppose that these paths can be defined in such a fashion that for any x and x', the total weight of paths that include a transition from x to x' is no more the $b\pi(x)T(x,x')$, for some constant b. It is not hard to show that in this case the conductance of the Markov chain is at least $1/2b$ — for any set of states, S, with $\pi(S) \leq 1/2$, the total weight of paths from states in S to states not in S must be at least $\pi(S)/2$; since this cannot exceed b times the sum of $\pi(x)T(x,x')$ for $x \in S$ and $x' \notin S$, the latter must be at least $\pi(S)/2b$, giving the bound of $1/2b$ on the conductance.

Markov chains with infinite state spaces. In many applications we will need to sample from distributions over countably infinite or continuous spaces, and hence will wish to construct Markov chains having these as state spaces. New phenomena are possible in such cases — for example, there are Markov chains on infinite state spaces that have no invariant distribution. The analysis may be more difficult, and care must be taken to avoid fallacies.

When the state space is countably infinite, the definition given for a Markov chain can remain unchanged, as can the detailed balance condition that can be used to show the invariance of a distribution. The fundamental theorem proved above remains valid, after changing "maximum" and "minimum" to "least upper bound" and "greatest lower bound", and adding the condition that the function $a(x)$ be bounded.

When the state space is continuous, we can interpret $p_0(x)$ as the probability density for the initial state at x, and $T(x,x')$ as the probability density for a transition to state x' from a given state x. The state distributions at successive times can still be found as in equation (3.8), with the obvious replacement of the summation by an integral, and the detailed balance condition of equation (3.10) remains valid. The k-step transition probabilities can be found by a continuous generalization of a matrix multiply, as follows:

$$T^k(x,x') \;=\; \int T(x,\tilde{x})\,T^{k-1}(\tilde{x},x')\,d\tilde{x} \tag{3.37}$$

Even with the amendments of the previous paragraph, however, the fundamental theorem does not quite hold as stated, because the transition in the proof from equation (3.25) to (3.26) relies on the fact that probabilities are never greater than one, whereas probability densities can be greater than one. Fortunately, the expectations of bounded functions still converge as given by equation (3.15), justifying the use of the chain for Monte Carlo estimation.

Complications can arise, however. Often, when we deal with continuous spaces, we will use Markov chains whose initial state distributions and transition probability densities must be expressed using delta functions. For example, we might always start in the same state, x_0. The initial probability density would then be given by $p_0(x) = \delta(x_0, x)$. For some methods, the transition probabilities include the possibility of "rejecting" a candidate move, in which case the old state is retained. This corresponds to a transition distribution with a non-zero probability mass concentrated at the old state. When dealing with such transition probabilities defined using delta functions, the detailed balance condition must be interpreted with care, since delta functions do not have real numerical values, and cannot be compared as if they did. Detailed balance must in these cases be verified by showing that for any two regions A and B,

$$\int_A \int_B \pi(x)\, T_n(x,x')\, dx'\, dx \;=\; \int_B \int_A \pi(x')\, T_n(x',x)\, dx\, dx' \tag{3.38}$$

i.e. that the probability of a transition from somewhere in A to somewhere in B is the same as that of a reverse transition.

Finally, one should note that there are Markov chains on infinite state spaces, some of which are useful, that converge to an invariant distribution, but for which the fundamental theorem stated above does not apply (ν in condition (3.12) being zero), and for which, indeed, no bound on the time required for convergence can be given that is independent of the initial state distribution. For example, consider the Markov chain with state space $\{0,1,2,\ldots\}$ and with $T(0,0)=1$, $T(x,x-1)=1$ for $x>0$, and all other transition probabilities zero. This chain clearly converges to an invariant distribution with $\pi(0)=1$, but one can say nothing of how long this might take unless some constraint is placed on the distribution for the initial state.

Random walks. The properties of simple Markov chains that perform *random walks* shed light on the behaviour of the more complex Markov chains that will be discussed later in this review. As a simple example, consider a homogeneous Markov chain that has the integers as its state space, starts in state 0, and uses the following transition probabilities:

$$T(x,x') \;=\; \begin{cases} \frac{1}{2} & \text{if } x'=x \\ \frac{1}{4} & \text{if } x'=x\pm 1 \\ 0 & \text{otherwise} \end{cases} \tag{3.39}$$

In each time step, this chain stays at its currently position with probability one half, and otherwise moves one step to the left or one step to the right, with the two directions being equally likely.

After n steps, what can we say about the state of the chain, X_n? From symmetry, it is clear that $E[X_n]=0$ — the random walk is equally likely to have taken us some distance to the right as it is to have taken us the same distance to the left — but how far from state 0 are we likely to be? We can get an indication of this by computing $E[X_n^2]$, as follows:

$$E[X_n^2] \;=\; \sum_{\tilde{x}} \tilde{x}^2 p_n(\tilde{x}) \;=\; \sum_{\tilde{x}} \tilde{x}^2 \left[\tfrac{1}{2}p_{n-1}(\tilde{x}) + \tfrac{1}{4}p_{n-1}(\tilde{x}-1) + \tfrac{1}{4}p_{n-1}(\tilde{x}+1)\right] \tag{3.40}$$

$$=\; \tfrac{1}{2}E[X_{n-1}^2] + \tfrac{1}{4}E[(X_{n-1}+1)^2] + \tfrac{1}{4}E[(X_{n-1}-1)^2] \tag{3.41}$$

$$=\; E[X_{n-1}^2] + \tfrac{1}{2} \tag{3.42}$$

Since $E[X_0^2]=0$, we see that $E[X_n^2]=n/2$. Thus, although after n iterations we are likely to have travelled a total distance of around $n/2$, we nevertheless are likely at that time to be a distance of only around $\sqrt{n/2}$ from the starting point. This comes about because the direction at each step is chosen independently at random, with the result that many of the steps end up cancelling each other. This "square root" phenomenon occurs generally with random walks, including random walks over continuous state spaces, of any dimensionality.

Unconfined random walks such as the one just considered are not of interest for sampling, since they do not have invariant distributions (as is clear from the fact that $E[X_n^2]$ goes to infinity as n increases). However, chains resembling random walks can have invariant distributions if they are biased appropriately or are confined to a finite state space. Many of

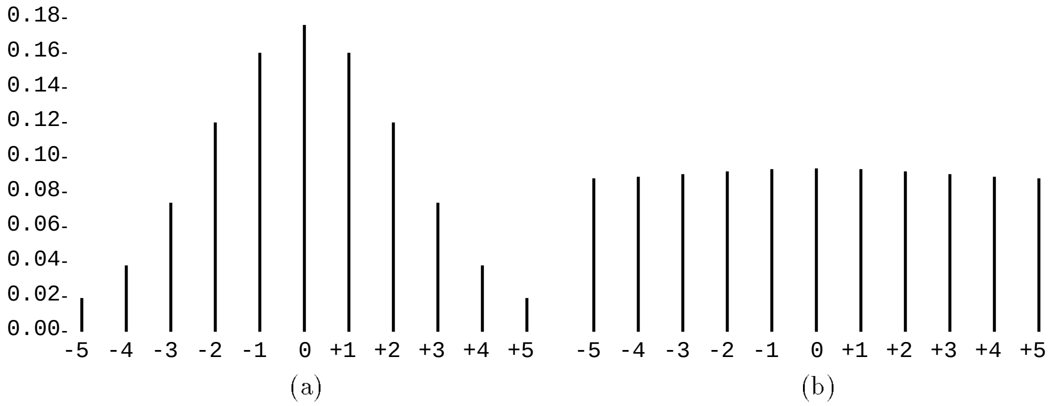


Figure 3.2: A random walk on the state space $\{-5, -4, \ldots, +4, +5\}$, with transition probabilities as in equation (3.39), except that the state stays the same whenever it would otherwise move out of bounds. The plots show the distribution of states (a) at time $t = 10$, and (b) at $t = 100$. The probabilities in (b) are almost uniform, ranging from a high of 0.0938 at 0 to a low of 0.0881 at ± 5.

the Markov chain sampling methods described later operate in random walk fashion when exploring a region of state space where the probability is almost uniform; they may also move in a random walk in certain directions, where the probability varies only slowly, even while being tightly confined in other directions.

Because moving a distance L via a random walk takes a number of iterations proportional to L^2, the performance of random walk sampling procedures will be worse than one might naively expect. Figure 3.2 illustrates this with respect to a random walk with the same transitions as equation (3.39), but with the modification that whenever the chain attempts to move outside the range -5 to $+5$, it instead stays in the current state. It is easy to verify that the uniform distribution over $\{-5, -4, \ldots, +4, +5\}$ is invariant with respect to this random walk, and that the random walk is ergodic. We should therefore be able to sample from the uniform distribution by simulating the random walk for some sufficiently large number of iterations. One might think that around 10 iterations would be sufficient to approach this equilibrium distribution, since the random walk has a probability of 1/2 of making a move in each iteration, and it can reach any position in the state space in only 5 steps, starting from the initial state of 0. As shown in Figure 3.2(a), however, the state distribution after 10 iterations is actually far from uniform. We should expect this from the general "square root rule" for distance travelled in a random walk — the uniform distribution should be approached only once the random walk has gone on long enough that any point in the state space is within the distance range that is likely to have been reached by that time. For this example, this suggests that around $10^2 = 100$ iterations will be needed. Figure 3.2(b) shows that the state distribution at that time is indeed close to uniform. A similar amount of time is required for the chain to move from one state at equilibrium to another independent state.

Constructing Markov chains. In a sampling application, our goal is to find an ergodic Markov chain that converges to a desired invariant distribution, at as fast a rate as possible. It is often convenient to construct the transition probabilities for such a chain from a set of base transition probabilities, given by $B_1, \ldots, B_s$, each of which leaves the desired distribution invariant, but which may not be ergodic individually. For example, each B_k might change only some subset of the variables making up the state.

One approach is to build a homogeneous Markov chain in which the transition probabilities are mixtures of the base transitions, with proportions given by α_k, where $\alpha_k > 0$ and $\sum_k \alpha_k = 1$:

$$T(x, x') = \sum_k \alpha_k B_k(x, x') \tag{3.43}$$

It is easy to show that if a distribution over states is invariant with respect to each of the B_k, then it is also invariant with respect to T. Furthermore, if each of the B_k satisfy detailed balance (equation (3.10)), T will as well.

An alternative is to construct a nonhomogeneous Markov chain by applying each of the transitions in turn. That is, for every $a \geq 0$, and each k with $1 \leq k \leq s$:

$$T_{as+k-1}(x, x') = B_k(x, x') \tag{3.44}$$

Clearly, a distribution invariant with respect to each B_k is also invariant with respect to all the T_n. Another way of looking at this construction is that it defines a homogeneous Markov chain with transition matrix $\mathbf{T} = \mathbf{B}_1 \cdots \mathbf{B}_s$. Again, if a distribution is invariant with respect to all the $\mathbf{B}_k$, it is also invariant with respect to $\mathbf{T}$. Unlike the case with mixtures, though, even when all the $\mathbf{B}_k$ satisfy detailed balance, $\mathbf{T}$ generally does not. This fact is usually of no particular importance, but there are some methods of theoretical analysis that apply only to reversible Markov chains (those satisfying detailed balance). For this reason one might sometimes wish to use the chain with transition matrix $\mathbf{T} = \mathbf{B}_1 \cdots \mathbf{B}_{s-1}\mathbf{B}_s\mathbf{B}_s\mathbf{B}_{s-1} \cdots \mathbf{B}_1$, which does satisfy detailed balance if each of the $\mathbf{B}_k$ do.

One might expect the Markov chain of equation (3.44) to perform better than that of equation (3.43), since the former ensures that each B_k is applied equally often. Consider the (common) case where component X_k can be changed only by B_k. If each iteration applies a B_k selected at random, a particular X_k might, by chance, be neglected for a fairly long time. On the other hand, applying the B_k in order can sometimes result in a chain that is not ergodic. Mezie (4:1981) has proposed an intermediate strategy, in which the B_k are applied in an order that is randomly selected at the start of each cycle. These issues are discussed further in Section 4.4.

One reason to combine a number of B_k is to produce a chain that is ergodic — this is clearly necessary, for example, if each B_k changes only some of the variables. Ergodicity may be verified by confirming that condition (3.12) holds, in which case a bound on the rate of convergence is obtained as well. For finite, homogeneous chains, a sufficient condition for obtaining a bound using the fundamental theorem is that for some k, $T^k(x, x') > 0$ for all x and x'. The Markov chain is then said to be *regular*.

A weaker condition is that for all x and x', there is a k such that $T^k(x, x') > 0$. A Markov chain satisfying this is said to be *irreducible*. Irreducibility is not sufficient to guarantee ergodicity, as defined here, since it does not exclude the possibility that the chain is periodic. This possibility is ruled out if at every step there is a non-zero probability of the chain remaining in its current state, which can easily be ensured by mixing the transitions with the identity. (In fact, a weaker condition is sufficient — that at least one state have a non-zero probability of being repeated.)

Even if an ergodic chain has been constructed, it may still be desirable to combine it with other B_k, since these may help speed convergence. It is easy to see that if at least one of the B_k is ergodic, then the mixture, T, of equation (3.43) is ergodic as well. This is true also for the chain of equation (3.44), in which the B_k are applied in sequence, provided each B_k has a non-zero probability of leaving the state unchanged. (Without this last condition, counterexamples exist in which $\mathbf{B}_1$ and $\mathbf{B}_2$ are both ergodic, with the same invariant distribution, but $\mathbf{T} = \mathbf{B}_1\mathbf{B}_2$ is not ergodic.)

In Sections 4 and 5, which follow, a number of base transitions will be described, and will be combined in various standard ways. The reader should keep in mind that more elaborate combinations of these methods are quite possible, and may well be necessary when tackling practical problems.

4. The Metropolis and Gibbs Sampling Algorithms

In this section, I describe two classes of Markov chain Monte Carlo algorithms that are applicable to both discrete and continuous systems. These methods also form the basis for the hybrid Monte Carlo algorithm described in Section 5.

4.1 Gibbs sampling

The *Gibbs sampler*, also known as the *heatbath* algorithm, is conceptually the simplest of the Markov chain sampling methods, but has come into prominence only recently, with the work of Geman and Geman (4:1984) and Gelfand and Smith (4:1990). It is widely applicable to problems where the variables take on values from a small finite set, or have conditional distributions of a parametric form that can easily be sampled from.

The Gibbs sampling algorithm. Suppose we wish to sample from the joint distribution for $X = \{X_1, \ldots, X_n\}$ given by $P(x_1, \ldots, x_n)$, where the range of the X_i may be either continuous or discrete. The Gibbs sampler does this by repeatedly replacing each component with a value picked from its distribution conditional on the current values of all other components. This process can be seen as generating a realization of a Markov chain that is built from a set of base transition probabilities B_k, for $k = 1, \ldots, n$, with

$$B_k(x, x') \;=\; P(x'_k \mid \{x_i : i \neq k\}) \cdot \prod_{i \neq k} \delta(x_i, x'_i) \tag{4.1}$$

I.e. B_k leaves all the components except x_k unchanged, and draws a new x_k from its distribution conditional on the current values of all the other components. This is assumed to be a feasible operation.

These base transitions are usually applied in sequence, as in equation (3.44), though at each step we could instead pick a B_k at random from some pre-specified distribution, as in equation (3.43). To complete the definition of the Markov chain, we also must specify some initial distribution, $p_0(x)$, but the hope is that this choice will not be critical.

When the B_k are applied in sequence, the algorithm can be described as simulating a homogeneous Markov chain, $X^{(0)}$, $X^{(1)}$, $X^{(2)}$, ..., with transition matrix $T = B_1 B_2 \cdots B_n$. The procedure for generating $X^{(t)}$ from $X^{(t-1)}$ can be expressed as follows:

$$\begin{array}{l}
\text{Pick } X_1^{(t)} \text{ from the distribution for } X_1 \text{ given } x_2^{(t-1)},\, x_3^{(t-1)}, \ldots, x_n^{(t-1)}. \\
\text{Pick } X_2^{(t)} \text{ from the distribution for } X_2 \text{ given } x_1^{(t)},\, x_3^{(t-1)}, \ldots, x_n^{(t-1)}. \\
\qquad\qquad\qquad\qquad\vdots \\
\text{Pick } X_i^{(t)} \text{ from the distribution for } X_i \text{ given } x_1^{(t)}, \ldots, x_{i-1}^{(t)},\, x_{i+1}^{(t-1)}, \ldots, x_n^{(t-1)}. \\
\qquad\qquad\qquad\qquad\vdots \\
\text{Pick } X_n^{(t)} \text{ from the distribution for } X_n \text{ given } x_1^{(t)},\, x_2^{(t)}, \ldots, x_{n-1}^{(t)}.
\end{array}$$

Note that the new value for X_{i-1} is used immediately when picking the next value for X_i.

To show that the Gibbs sampling algorithm works, we must first verify that all the B_k leave the desired distribution invariant. Intuitively, this is clear. Since B_k leaves the components x_i for $i \neq k$ unchanged, the desired marginal distribution for these components is certainly invariant. Furthermore, the conditional distribution for x_k in the new state given the other components is defined to be that which is desired. Together, these ensure that if we started from the desired distribution, the joint distribution for all the X_i after B_k is applied must also be the desired distribution.

This can be seen more formally as follows, for discrete X_i:

$$
\begin{aligned}
\sum_{\tilde{x}} P(\tilde{x})\, B_k(\tilde{x}, x) &= \sum_{\tilde{x}} P(\tilde{x}_k \mid \{\tilde{x}_i : i \neq k\})\, P(\{\tilde{x}_i : i \neq k\}) \\
&\qquad \cdot P(x_k \mid \{\tilde{x}_i : i \neq k\}) \prod_{i \neq k} \delta(\tilde{x}_i, x_i) && (4.2) \\
&= P(x_k \mid \{x_i : i \neq k\})\, P(\{x_i : i \neq k\}) \sum_{\tilde{x}_k} P(\tilde{x}_k \mid \{x_i : i \neq k\}) && (4.3) \\
&= P(x) && (4.4)
\end{aligned}
$$

Here, use has been made of equation (2.6). The proof for continuous X_i is analogous. One can also prove invariance by showing that detailed balance (equation (3.10)) holds.

We must also show that the Markov chain built from the B_k is ergodic. This will be so, regardless of whether we apply the B_k in sequence or select one at random, provided that all the conditional probabilities used to define the B_k in equation (4.1) are non-zero. This guarantees that there is a non-zero probability (or probability density) for the chain to move from any state to any other state in n steps (one full iteration), with each step changing one component. The chain is thus irreducible, and, since it has a non-zero probability of remaining in the current state, it is also ergodic. If some of the conditional probabilities in (4.1) are zero, however, a Markov chain built from these transitions may or may not be ergodic. These cases must be analysed on an individual basis.

Under suitable conditions, therefore, the state of the Markov chain used in Gibbs sampling will at long times have the desired distribution. By simulating this chain we can thus produce a sample of values for use in a Monte Carlo estimation procedure. A number of questions remain, however. Should we sample states from a single, long realization of the chain, or from many, shorter realizations? How can we judge the accuracy of the estimates obtained? What initial state distribution should we use? Such questions will be deferred to Section 6.3, as they apply equally to the other methods that will be discussed.

Random variate generation for Gibbs sampling. The basic operation used in the Gibbs sampling algorithm is the generation of a random value for some component of the state, X_i, from its conditional distribution given the current values of all the other components, X_j, for $j \neq i$. The speed of the algorithm depends crucially on whether this operation can be done quickly.

For discrete components that take on values from a small set, the usual approach is to simply calculate the joint probabilities (perhaps up to an unknown common factor) of all the states in which X_i takes on its various possible values, while the other X_j remain fixed at their current values. The conditional distribution for X_i is then found by normalizing these probabilities so they sum to one, and a new value for X_i is picked from this distribution. (That the calculation of joint probabilities required for this is feasible is one of the assumptions made in Section 3.1, where the class of problems being addressed was defined.)

For components taking on values from an infinite set, additional structure is clearly required if a feasible generation procedure is to be found. One possibility is that the conditional distribution for X_i has a standard parametric form for which a good method of generating random variates has been developed. Before applying such a procedure, of course, the actual values of the distribution parameters will have to be computed; these will depend on the current values of X_j for $j \neq i$. (Devroye, 9:1986) is a comprehensive reference on methods of generating values from standard distributions. The techniques employed in these methods are many and varied, and may well be applicable even if the form of the distribution is non-standard. Developing such a customized procedure could involve considerable effort, however.

A "black box" generation procedure based on rejection sampling that is applicable to a large class of continuous univariate distributions is described by Gilks and Wild (9:1992). It requires only that the log of the probability density be a concave function, and that the density and its derivative can be calculated at any given point (perhaps up to an unknown factor). Gilks and Wild argue that for Bayesian hierarchical models these conditions will often be satisfied. They also report that, for a particular example, an average of only three evaluations of the density and its derivative were required in order to generate a value from the conditional distributions required for Gibbs sampling. Gilks (9:1992) has modified this method to permit its use when the derivative of the log probability density is unavailable.

Another technique of random variate generation that has wide applicability is the generalized ratio-of-uniforms method, described by Wakefield, Gelfand, and Smith (9:1991).

Note that a component that is updated in one step of the Gibbs sampler need not consist of a single discrete or real value. Indeed, it is generally desirable to group many values into a single component, provided that an efficient procedure is available for generating values from the distribution of this multivariate component conditional on the rest of the state.

Applications of Gibbs sampling. Not all problems can reasonably be tackled using Gibbs sampling. Of the examples used in this review, the simulation of Lennard-Jonesium and Bayesian learning for multi-layer perceptrons both seem inappropriate for Gibbs sampling, as the conditional distributions they give rise to are complex and multimodal. For these problems, the Metropolis algorithm of the next section is more appropriate.

Gibbs sampling can be used for a great many problems, however. As seen above, it may be particularly appropriate for systems of discrete variables that take on values from a small set, and for systems of continuous variables where the required conditional distributions are of standard forms for which good sampling methods have been developed. Even in such favourable cases, some of the methods to be described later may explore the state space faster than Gibbs sampling does, but the Gibbs sampler certainly has an attractive simplicity.

Example: The 2D Ising model. The two-dimensional Ising model provides a simple example of Gibbs sampling. To simulate this system, we first select values for the spins, S_i, according to some initial distribution. A natural choice is to set each spin to $+1$ or -1 with equal probability, independently. We then visit the various spins repeatedly, either in some pre-defined order, or by picking a spin at random each time. When spin S_i is visited, a new value for it is chosen from the conditional distribution defined by the other spins.

This conditional distribution is derived from the canonical distribution (equation (2.54)) with respect to the Ising energy function of equation (2.55). It can be written as

$$P(S_i = +1 \mid \{s_j : j \neq i\}) = \frac{\exp\big(-E(s^+)/T\big)}{\exp\big(-E(s^+)/T\big) + \exp\big(-E(s^-)/T\big)} \tag{4.5}$$

$$= \sigma\Big(2\Big(H + \sum_{j\,:\,(i,j)\in\mathcal{N}} J s_j\Big)\Big/T\Big) \tag{4.6}$$

where $\sigma(z) = 1/(1+\exp(-z))$ and s^+ has $s_i = +1$ (and other s_j as currently set) while s^- has $s_i = -1$. Since each spin has only four neighbors, the above probability can easily be calculated, in time that is independent of the total number of spins. The choice for S_i can then be made by drawing a random number uniformly from the interval $[0,1)$ and setting S_i to $+1$ if this number is less than the calculated probability, and to -1 otherwise.

It is informative to compute a bound on the rate of convergence of the Gibbs sampler for this model. For simplicity, let us take H to be zero and T to be one. If the spins are visited in order, we can view the algorithm as simulating a homogeneous Markov chain, in which each transition corresponds to a full sweep that visits each spin once. The transition probabilities for this chain can be found from equation (4.6), as follows, noting that $1 - \sigma(x) = \sigma(-x)$:

$$T(s, s') \;=\; \prod_i \sigma\Big(2s'_i \sum_{j\,:\,(i,j)\in\mathcal{N}} J s_j^{(i)}\Big) \tag{4.7}$$

where $s_j^{(i)}$ equals s'_j for $j < i$ and s_j for $j > i$. Each factor in the above expression must be at least $\sigma(-8J)$, since each spin has only four neighbors, and $\sigma(z)$ increases monotonically with z. This gives a lower bound on $T(s, s')$ of $[\sigma(-8J)]^K$, where K is the number of spins. To get a lower bound for ν of equation (3.12), we need an upper bound on the equilibrium state probabilities as well. It is easy to see that the states of highest probability are the two where all the S_i are equal, which have energy $-2KJ$. However, to find the absolute probability of these states under the canonical distribution, we would need to know the value of the normalization constant, Z, in equation (2.54), which is not easily found. Of course, the probability of the most probable state cannot exceed one, so we do get the bound $\nu > [\sigma(-8J)]^K$, showing that the fundamental theorem applies, and that convergence is therefore guaranteed in the limit. The upper bound this gives on the number of steps needed to reach a good approximation to the canonical distribution grows exponentially with K, however, and is too large to be of practical use in realistic cases.

It is of course possible that a more sophisticated analysis might yield a better bound. Typically, however, systems such as this are simulated without any formal guarantees that they will converge within the allotted time. Empirical tests such as those discussed in Section 6.3 are used to gain confidence in the accuracy of the results.

Example: Belief networks. Another application of Gibbs sampling that has received considerable attention is that of inference in belief networks. This application was introduced by Pearl (4:1987), and is reviewed by York (4:1992). In this problem, we assume that the parameters of the belief network have been fully specified, either by an expert, or as a result of a previous learning phase. Furthermore, some of the variables described by the belief network have been now been observed. Our task is to sample from the conditional distribution for the remaining variables.

We start the Gibbs sampling procedure by fixing the observed variables to their known values, and setting the unobserved variables arbitrarily. We then repeatedly visit each unobserved variable in turn, each time randomly selecting a new value for the variable from its conditional distribution given the current values of the other variables. From the joint distribution of equation (2.15), we see that the conditional distribution for X_k is as follows:

$$P(x_k \mid \{x_i : i \neq k\}) \;=\; \frac{P(x_k \mid \{x_i : i \in \mathcal{P}_k\}) \prod\limits_{j\,:\,k\in\mathcal{P}_j} P(x_j \mid x_k, \{x_i : i \in \mathcal{P}_j - \{k\}\})}{\sum\limits_{\tilde{x}_k} P(\tilde{x}_k \mid \{x_i : i \in \mathcal{P}_k\}) \prod\limits_{j\,:\,k\in\mathcal{P}_j} P(x_j \mid \tilde{x}_k, \{x_i : i \in \mathcal{P}_j - \{k\}\})} \tag{4.8}$$

(The denominator in this expression is simply the factor required to normalize these probabilities, and is computed naturally as the numerator is computed for the various possible x_k.) The conditional probabilities in equation (4.8) are assumed to be known explicitly, or to be computable from some parametric formula, such as equation (2.16).

For a belief network with binary variables, we might be interested in the probability that a particular unobserved variable is 1, given the values of the observed variables. One way of estimating this probability is to simply compute the average value of this binary variable as the Gibbs sampling procedure is run (ignoring the initial passes, prior to when the equilibrium distribution is thought to have been approached). This corresponds to expressing the probability as an expectation in the form of equation (2.8). As pointed out by Pearl (4:1987), however, more accurate results for a given size sample are obtained by expressing this probability as an expectation in the form of equation (2.9) — that is, by averaging the *probability* that the value is one at each iteration (obtainable using equation (4.8)), rather than the value itself. This general technique is discussed further in Section 6.3.

In many applications of belief networks, some of the conditional probabilities used in Gibbs sampling will be zero, reflecting the presence of deterministic constraints. For example, a belief network used for medical diagnosis might specify that it is simply impossible for a male patient to be pregnant. These zero probabilities sometimes render the Markov chain defined by the Gibbs sampling procedure non-ergodic, a problem discussed by York (4:1992).

Dagum and Luby (2:1993) have shown that in general the problem of probabilistic inference for belief networks is computationally intractable, in the sense that the worst case computation time must increase more than polynomially in the size of the network, assuming certain widely-believed conjectures to be true. One therefore cannot hope to in all cases obtain results of assured reliability using the Gibbs sampler. However, Dagum and Chavez (2:1993) have used the methods of Sinclair and Jerrum (7:1989) to demonstrate that Gibbs sampling can form the basis for a polynomial-time approximation algorithm for belief networks that have a limited degree of "dependence" — a measure how much the forward conditional probabilities for variables in the network can change depending on the values of the parent variables.

Example: Latent class models. In (Neal, 2:1992c), I have shown how Gibbs sampling can be used for Bayesian inference with latent class models, such as that defined by equation (2.13). Lavine and West (2:1992) handle a related model for real-valued data by using Gibbs sampling as well, but in a somewhat different fashion, which will also be illustrated here using the model of equation (2.13), with uniform prior distributions for the parameters α and β.

Recall that in this model, for each case, i, there are n visible variables, $V_{i1}, \ldots, V_{in}$, and a latent class variable C_i. Assume that the values of the visible variables have been observed for $C-1$ cases, $V_1, \ldots, V_{C-1}$, and that some of the visible variables have been observed for a further case, C. The task is to predict the values of the unobserved variables in case C. (I must apologize that my notational conventions lead to "C" being used here with two entirely unrelated meanings!)

The most direct approach to applying Gibbs sampling to this problem, which is used by Lavine and West (2:1992), is to simulate the distribution for all the unknown variables and parameters, using conditional distributions derived from the joint distribution of equation (2.44). For given values of the parameters, the variables for one case are independent of those for other cases, and for given values of the variables in all cases, the parameters are all independent of each other. Noting these and some other independence relations, the relevant conditional distributions are seen to be determined by the following proportionalities:

$$P(c_i \mid \alpha,\, \beta,\, v) \;\;\propto\;\; \alpha^{c_i}\,(1-\alpha)^{1-c_i} \prod_j \beta_{c_i j}^{v_{ij}}\,(1-\beta_{c_i j})^{1-v_{ij}} \tag{4.9}$$

$$P(v_{ij} \mid \beta,\, c) \;\;\propto\;\; \beta_{c_i j}^{v_{ij}}\,(1-\beta_{c_i j})^{1-v_{ij}} \tag{4.10}$$

$$P(\alpha \mid c) \;\;\propto\;\; \prod_i \alpha^{c_i}\,(1-\alpha)^{1-c_i} \;=\; \alpha^{r_1}\,(1-\alpha)^{r_0} \tag{4.11}$$

$$P(\beta_{\ell j} \mid c,\, v) \;\;\propto\;\; \prod_{i\,:\,c_i=\ell} \beta_{\ell j}^{v_{ij}}(1-\beta_{\ell j})^{1-v_{ij}} \;=\; \beta_{\ell j}^{s_{\ell j1}}(1-\beta_{\ell j})^{s_{\ell j0}} \tag{4.12}$$

where $r_\ell = \sum\limits_{i=1}^{C} \delta(\ell, c_i)$ and $s_{\ell j b} = \sum\limits_{i=1}^{C} \delta(\ell, c_i)\delta(b, v_{ij})$.

Generating values from the conditional distributions for the C_i and V_{ij} given above is straightforward, as these variables have only two possible values, whose relative probabilities are easily obtained from the formulas. The conditional distributions for the real-valued parameters α and β_{cj} are both of the standard "Beta" form, for which efficient random variate generation procedures are known (Devroye, 9:1986). The Gibbs sampling procedure can thus be feasibly implemented.

The conditional distribution for the V_{ij} shown above can be used to sample values for the unobserved variables in case C, which we are trying to predict. This direct approach to prediction is conceptually simple, but as noted above for belief networks, it is better to obtain a predictive distribution by averaging the conditional distributions for these variables. It also turns out that any unknown variables in the training cases can simply be ignored when computing conditional distributions for the c_i and $\beta_{\ell j}$, eliminating any need to pick particular values for them.

In (Neal, 2:1992c), I have applied Gibbs sampling to this model using the joint distribution for the visible and class variables alone given by equation (2.45), which was obtained by analytically integrating over the parameters of the model (again, assuming uniform priors). In this scheme, the conditional distributions required are as follows:

$$P(c_k \mid v, \{c_i \,:\, i \neq k\}) \;\;\propto\;\; \frac{\bar{r}_{c_k}+1}{C+1} \cdot \prod_j \frac{\bar{s}_{c_k j v_{kj}}+1}{\bar{r}_{c_k}+1} \tag{4.13}$$

$$P(v_{kj} \mid c_k,\, \{c_i \,:\, i \neq k\},\, \{v_{ij} \,:\, i \neq k\}) \;\;\propto\;\; \frac{\bar{s}_{c_k j v_{kj}}+1}{\bar{r}_{c_k}+1} \tag{4.14}$$

where here $\bar{r}_\ell = \sum\limits_{i\,:\,i\neq k} \delta(\ell, c_i)$ and $\bar{s}_{\ell j b} = \sum\limits_{i\,:\,i\neq k} \delta(\ell, c_i)\delta(b, v_{ij})$.

The comments above regarding the need to actually sample values for the unknown V_{ij} apply here as well. It is plausible to think that the elimination of the model parameters in this scheme gives it an advantage over the preceding scheme in terms of the speed of convergence to the equilibrium distribution and the degree of dependence between the states sampled.

The joint distribution for just the parameters and visible variables of this model (equation (2.46)), obtained by summing over the possible values of the class variables, does not lead to a feasible scheme for Gibbs sampling, as sampling from the conditional distributions for the parameters is quite intractable.

Variations on Gibbs sampling. Due to the simple nature of the Gibbs sampling algorithm, most options in its implementation involve the general issues of Markov chain Monte Carlo estimation discussed in Section 6.3, the methods used to generate values from the conditional distributions discussed above, or issues specific to a particular application. Two variations on the generic Gibbs sampling algorithm are worth mentioning here, however, along with the somewhat hard to classify "hit-and-run" scheme. An "approximate" form of Gibbs sampling is described as a variation on the Metropolis algorithm in Section 4.3. The "over-relaxed" method of Adler (4:1981) could also be regarded as a variation on Gibbs sampling.

Gelfand and Smith (4:1990) point out that convergence of the Gibbs sampler may be improved by exploiting conditional distributions involving less than the full set of variables. For example, let $X = \{X_1, X_2, X_3\}$, and assume that we can feasibly sample from the distribution for each X_i given values for the other X_j. We could therefore implement the usual Gibbs sampling algorithm, in which we pick new values for each component in turn, given the values for the others. If we assume that we can also feasibly sample from the distribution for X_2 given X_1 alone, however, we could instead proceed as follows: pick a new x_1 given the current x_2 and x_3, pick a new x_2 from the distribution conditional on just x_1, pick a new x_3 given the current x_1 and x_2, repeat until convergence. Gelfand and Smith found that exploiting reduced conditional distributions in this fashion speeds convergence, though not dramatically so for the problems they tried.

Tanner and Wong (4:1987) introduced a "data augmentation" algorithm that is essentially a special case of the Gibbs sampler, in which X consists of only two components, X_1 and X_2. This paper made a valuable contribution in pointing out the utility of Monte Carlo methods of this sort for Bayesian inference, but the particular algorithm they describe is sub-optimal. Though phrased differently in the paper, it may be expressed as follows. Each iteration of the algorithm will produce a pool of m values for X_1, which, for the final iteration, will be used as the basis for Monte Carlo estimates. To generate this pool of values we first generate m values for X_2 by, in each case, randomly selecting one of the X_1 values from the pool of the previous iteration, independently, with replacement, and picking from the distribution for X_2 given that value for X_1. We then generate the new pool of m values for X_1 by picking from the conditional distributions for X_1 given each of these m values for X_2.

It may seem that this procedure is merging information in the pool in a desirable way, but this is an illusion. In fact, the path by which each final value is computed is determined independently of the results of the computation, and hence is no different from a Gibbs sampling run. Furthermore, in each iteration, a fraction of approximately $1/e$ of the X_1 values in the pool will not be selected for use in picking any X_2, and hence will have no effect on subsequent iterations. Their computation is therefore a waste of effort. It is better to simply perform m independent runs of the Gibbs sampler.

The *hit-and-run* algorithm, described by Bélisle, Romeijn, and Smith, (4:1993), can be seen as a form of Gibbs sampling in which the coordinate system used to represent the state in terms of components is chosen at random for each iteration. The state space is assumed to be contained in an Euclidean space of some dimension. At each iteration, a direction in this space is chosen at random from some fixed distribution, and a new state is chosen from among those that can be reached by travelling in this direction, or the reverse direction, starting from the current point, with the probability of chosing a particular such state being proportional to the desired equilibrium probability for that state. This choice of a new state is the same as would be made by ordinary Gibbs sampling if the direction in question were one of the coordinate axes.

This scheme results in detailed balance being satisfied for *any* distribution of directions, though not all distributions will lead to an ergodic Markov chain. With a uniform distribution for the directions, the scheme is referred to as the "Hypersphere directions algorithm", and is ergodic whenever the region of non-zero probability density is bounded. When the directions are confined to those along the coordinate axes — the "Coordinate directions method" — the algorithm is equivalent to ordinary Gibbs sampling, and may not be ergodic if the region with non-zero probability density is not connected.

For the hit-and-run method to be feasible, it must be possible to sample from the distributions along all the possible directions. One application where this can be done is that of sampling uniformly from the interior of a polyhedron. In many cases, however, sampling along arbitrary directions is not easy, and the ordinary Gibbs sampling algorithm will be preferable.

Goodman and Sokal (4:1989, Section VII) discuss a more general concept of "partial resampling", of which the hit-and-run algorithm may be seen as a special case.

4.2 The Metropolis algorithm

The classic paper of Metropolis, Rosenbluth, Rosenbluth, Teller, and Teller (4:1953) was the first to employ Markov chain sampling, in the form now known as the *Metropolis algorithm*. This algorithm has since been applied extensively to problems in statistical physics; indeed, in this literature, the "Monte Carlo method" is almost synonymous with use of the Metropolis algorithm.

The Metropolis algorithm shares many of the characteristics of the Gibbs sampler, but is more generally applicable, as it avoids any need to sample from difficult distributions. It can be applied to problems where the state is either continuous or discrete, as long as it is possible to compute the ratio of the probabilities, or probability densities, of two states.

The Metropolis algorithm. Suppose that we wish to sample from the joint distribution for $X = \{X_1, \ldots, X_n\}$. The Metropolis algorithm does this by repeatedly considering randomly generated changes to the components of X, accepting or rejecting these changes based on how they affect the probability of the state. This process can be seen as the operation of a Markov chain built from a set of base transition probabilities, B_k, for $k = 1, \ldots, n$. The way transition B_k operates to generate a new state, x', from the current state, x, can be described as follows:

1) Select a *candidate state*, x^*, in which all components other than the k'th are the same as in x, while x_k^* is picked at random from a *proposal distribution*, which may depend on x, given by the probabilities $S_k(x, x_k^*)$.

2) *Accept* this candidate state with probability $A(x, x^*)$; otherwise, *reject* it, and retain the current state. In detail, this can be done by generating a random number, u, from the uniform distribution on $[0, 1)$, and then setting the next state as follows:

$$x' \;=\; \begin{cases} x^* & \text{if } u < A(x, x^*) \\ x & \text{otherwise} \end{cases} \tag{4.15}$$

The probabilities for the proposal distribution, $S_k(x, x_k^*)$, must, of course, be non-negative, and must satisfy $\sum_{\tilde{x}_k} S_k(x, \tilde{x}_k) = 1$. For the moment, we will also require that S_k satisfy a the following symmetry condition:

$$S_k(x, x_k^*) = S_k(x^*, x_k), \text{ whenever } x_i^* = x_i \text{ for all } i \neq k \tag{4.16}$$

In many applications, $S_k(x, x_k^*)$ depends only on x_k, not on x_i for $i \neq k$, but proposal distributions of the more general form are permissible.

The acceptance probability, $A(x, x^*)$, can have various forms, one of which is defined below. Note that when the candidate state is rejected, the current state becomes the new state, and is included again in any time averages that are being computed.

More formally (but perhaps less intelligibly), the transition probabilities are as follows:

$$\begin{aligned} B_k(x, x') &= S_k(x, x'_k)\, A(x, x') \prod_{i \neq k} \delta(x_i, x'_i) \\ &\quad + \delta(x, x') \big[1 - \sum_{\tilde{x}} S_k(x, \tilde{x}_k)\, A(x, \tilde{x}) \prod_{i \neq k} \delta(x_i, \tilde{x}_i)\big] \end{aligned} \tag{4.17}$$

The first term is the probability of proposing a change in component k from x_k to x'_k, and then accepting the proposed change. The second term accounts for the possibility of rejecting the candidate state, and therefore remaining in the current state. When X_k is a continuous variable, the sum in this term will be replaced by an integral. Note that there is no need to actually compute the value of this term.

For the moment, only the following *Metropolis* acceptance function will be used:

$$A(x, x') = \min\big(1,\, P(x')/P(x)\big) \tag{4.18}$$

When $P(x)$ is defined as the canonical distribution with respect to some energy function, $E(x)$, as in equation (2.54), this acceptance function can be written as:

$$A(x, x') = \min\big(1,\, \exp\big(-(E(x') - E(x))/T\big)\big) \tag{4.19}$$

Note that evaluating the acceptance probability does not require knowledge of the partition function, Z.

As was the case with the Gibbs sampler, we may apply the base transitions, B_k, in sequence, or we may at each step pick one at random. We again need to choose some distribution, $p_0(x)$, for the initial state in the chain.

We can prove that $P(x)$ is an invariant distribution for the Markov chain used in the Metropolis algorithm by showing that detailed balance (equation (3.10)) holds for each of the B_k, with respect to any two states, x and x'. If $x_i \neq x'_i$ for some $i \neq k$, detailed balance certainly holds, since the transition probabilities are both zero. If $x = x'$, detailed balance also holds trivially. Otherwise, detailed balance can be verified as follows, for symmetric S_k, and the Metropolis acceptance function (equation (4.18)):

$$\begin{aligned} P(x)\, B_k(x, x') &= P(x)\, S_k(x, x'_k)\, A(x, x') && (4.20) \\ &= S_k(x, x'_k) \min\big(P(x), P(x')\big) && (4.21) \\ &= S_k(x', x_k) \min\big(P(x'), P(x)\big) && (4.22) \\ &= P(x')\, S_k(x', x_k)\, A(x', x) && (4.23) \\ &= P(x')\, B_k(x', x) && (4.24) \end{aligned}$$

The Markov chain will be ergodic as long as $S_k(x, x'_k)$ is non-zero for all x'_k (including $x'_k = x_k$), and $P(x)$ is non-zero for all x. This guarantees that any new value for X_k has a non-zero probability of being proposed, and then accepted. In n steps, there is thus a non-zero probability of moving from any state to any other state. The Metropolis algorithm is often used when these criteria are not satisfied, however. Ergodicity must then be shown by somewhat more specific arguments.

Choice of state representation and proposal distribution. The distributions used to propose candidate states will clearly have a large effect on how well the Metropolis algorithm operates. Prior to choosing these proposal distributions, an even more fundamental decision must be made concerning how the full state, x, is to be represented in terms of components, $x_1, \ldots, x_n$.

One option is to not divide the state into components at all, giving what I will call a *global* Metropolis algorithm. All transitions in the Markov chain are then of the same type; in each, a candidate for the next state is proposed according to the probabilities given by a single function, $S(x, x')$.

As assumed in the presentation above, however, it is more common to divide the state into components, with each component, X_k, corresponding to a base transition, B_k, of the Markov chain, which updates only X_k. I will refer to schemes of this sort as *local* Metropolis algorithms. The components may themselves be multidimensional, as is illustrated below in the case of Lennard-Jonesium. Often, the components will all be of similar type, and it will be appropriate to use the same form of proposal distribution for all of them.

When possible, it is desirable to use a decomposition of the state for which a ratio of the form $P(x')/P(x)$ in which x and x' differ in only one component can be computed in much less time than can $P(x)$ for an arbitrary x. This ratio is needed to calculate the acceptance probability, $A(x, x')$. It may also be desirable to choose a decomposition in which the components of the state are nearly independent. Dependencies amongst components tend to slow exploration of state space, since major changes to the state may be possible only via a long series of local changes, each of which respects the dependencies. In a system of independent variables, state space may be explored very rapidly, as long as the proposal distribution for each component individually provides rapid exploration of that component's states.

For components where the set of possible values is finite, a proposal distribution that gives equal probabilities to all possible values is the obvious choice. Another commonly-chosen option is a uniform distribution over all possible values except the current one. The latter choice sometimes leads to faster exploration of state space, but if the B_k are applied in order systematically, it can sometimes produce slower convergence instead, or even cause the Markov chain to be non-ergodic. These issues are discussed further in Section 4.4.

For real-valued components, many forms for the proposal distribution for component k, $S_k(x, x'_k)$, are reasonable. A Gaussian distribution centred on the current x_k is one obvious choice. A Cauchy distribution, advocated by Szu and Hartley (6:1987), is similar, but with the possible advantage that its heavy tails allow large changes to be tried on occasion. Both these choices satisfy the symmetry condition (4.16). Both also have a non-zero probability of proposing any change to the component, and hence produce a chain that is ergodic, as long as the desired equilibrium distribution, $P(x)$, is never zero. A uniform distribution on some interval centred on x_k is another possibility. It, too, satisfies the symmetry requirement, but it does not give non-zero probability to all new values for the component. It nevertheless leads to a Markov chain that is ergodic, as long as $P(x)$ is never zero, since any value can still be reached eventually. For continuous multi-dimensional components, the analogous possibilities are multivariate Gaussian or Cauchy distributions, and distributions that are uniform over a hypercube or hypersphere. All these proposal distributions have width parameters which will have to be set either on the basis of *a priori* knowledge of the problem or by trial and error (see also Sections 4.4 and 7.1).

For some problems, as in (Kirkpatrick, Gelatt, and Vecchi, 6:1983), the state may have more structure than just a set of component values, and there may be constraints on allowable states, leading to proposal distributions that are specific to the problem. Constraints on valid states can also be imposed by considering states violating the constraints to have zero probability (infinite energy). Any attempted move to an invalid state will then be rejected (at which point the old state must be counted again in forming estimates, as with any other rejection).

Applications of the Metropolis algorithm. The Metropolis algorithm has proved to be a flexible tool that is applicable to a wide range of problems. However, when the required conditional distributions can be sampled from easily, the Gibbs sampler may be preferred, and when it is difficult to decompose the state into local components that are not too dependent, the dynamical algorithms of the next section may be more attractive.

Example: The 2D Ising model. The Metropolis algorithm has been extensively applied to the simulation of the 2D Ising model. It is natural for this problem to treat each spin as a separate component. Usually, the proposal distribution for a change to spin i is $S_i(s, s'_i) = \delta(-s_i, s'_i)$ — i.e. the candidate state is always obtained by flipping spin i to its opposite value. The probability of accepting this s' is found from equations (4.19) and (2.55):

$$A(s, s') \;=\; \min\Big(1,\ \exp\Big(2s'_i\Big(\sum_{j\,:\,(i,j)\in\mathcal{N}} Js_j \;+\; H\Big)\Big/T\Big)\Big) \tag{4.25}$$

Computing this acceptance probability requires looking only at the four spins that are neighbors of spin i. If the decomposition of the state into individual spins did not permit this local computation, the method would be much less attractive. Note, however, that for low values of T, neighboring spins become very dependent, forming large clusters with identical orientation. This has prompted attempts to find a better representation for the state of this system, as in the Swendsen-Wang algorithm described in Section 4.3 below.

Whether the Metropolis algorithm or the Gibbs sampler is better for simulating the Ising system is discussed in Section 4.4.

Example: Lennard-Jonesium. The Metropolis algorithm is also widely used to simulate systems of molecules with energy functions such as the Lennard-Jones potential of equation (2.57). Typically, the goal is to sample from the distribution for the position coordinates of the molecules, with the independently-varying momenta handled separately by analytical methods. This distribution is defined by the potential energy, $E(\mathbf{q})$:

$$E(\mathbf{q}) \;=\; \sum_{i\neq j} 4\varepsilon\left[\left(\frac{\sigma}{|\mathbf{q}_i - \mathbf{q}_j|}\right)^{12} - \left(\frac{\sigma}{|\mathbf{q}_i - \mathbf{q}_j|}\right)^{6}\right] \tag{4.26}$$

with $\mathbf{q}_i = \{q_{i1}, q_{i2}, q_{i3}\}$ being the position vector for the i'th molecule.

The three coordinates for one molecule are usually treated as a single component. A new candidate value for such a component might typically be selected from the uniform distribution over the inside of a sphere of some radius, centred on the molecule's current position. The radius of this sphere is adjusted by trial and error to be as large as possible while keeping the rejection rate for moves reasonably low.

Computing the exact change in energy produced by a candidate move (needed to find the acceptance probability) is not a local operation with this model, since every molecule interacts with every other in equation (4.26). Since the influence of distant molecules is very small, however, a cut-off is sometimes introduced to allow local recomputation. Also, if the substance being simulated is in a liquid or gaseous phase, the distributions of molecules in distant regions will be almost independent, and hence their interactions will not slow the approach of the system to equilibrium. When these two locality properties hold, the time required to simulate such a system grows only linearly with its volume.

It is illuminating to consider alternative ways of applying the Metropolis algorithm to this system. One possibility would be to treat the coordinates of each molecule as separate components of the state. Each coordinate would then be updated independently, perhaps using a proposal distribution that was uniform over some interval centred on the current coordinate value. However, since the potential energy function is rotationally symmetric, this division into coordinates will be arbitrary, and hence one would not expect to gain anything by operating on this finer scale. In particular, to maintain an acceptably low rejection rate, the width of the interval from which candidate states are selected in this scheme will have to be similar to the diameter of the sphere used to select candidate states in the usual scheme. This finer division of the state will therefore not lead an appreciably faster exploration of state space.

In the other direction, one could consider treating the entire state as a single component, updating the positions of *all* the molecules at every step. However, if reasonable rejection rates are to be maintained with this scheme, the amount by which each molecule is moved will have to be much smaller than the amount each would be able to move if their positions were updated independently. Indeed, the size of the proposed changes will have to become ever smaller as the volume of the system being simulated increases. The reasons for this are discussed in Section 4.4.

Example: Multi-layer perceptrons. Gibbs sampling is not a feasible means of implementing Bayesian inference for complex multi-layer perceptron networks, because it is extremely difficult to sample from the distribution for one weight conditional on the current values of the other weights (and the training data). It is possible to apply the Metropolis algorithm to this problem, however, though it can be rather slow.

When using the Metropolis algorithm to sample from the posterior distribution for the weights of a neural network, it is natural to treat each weight as a separate component of the state. This decomposition will be most effective if the change in the probability of a state when just one weight is changed can be computed "locally" at little cost. It would also be nice if the weights were largely independent, as this would lead to rapid convergence if the width of each weight's proposal distribution were appropriately set. These desirable characteristics are sometimes partially present, especially with simple networks, but they are largely absent in large, complex networks.

Some pairs of weights will be independent — for example, two weights on connections into different output units make independent contributions to the likelihood (and will be independent with respect to a simple prior as well). However, two weights into the same unit, as well as any two weights into hidden units, are likely to be highly dependent. Accordingly, even with carefully selected proposal distributions the rate of convergence will be slowed by these dependencies.

Regardless of whether the weights are independent, a gain in computation time will be possible if the change in likelihood when a single weight is updated can be found using only local computations. For a simple network with the architecture shown in Figure 2.3 this can indeed be done, by saving, for each training case, the total input into each hidden and output unit, using the current values for the weights. Before a weight is updated, the contribution that its connection makes to the unit it feeds into can be subtracted from this saved sum; after the change, the new contribution can be added. In this way, the new value for the unit can be found with a constant amount of effort, regardless of how many other connections feed into it. If the weight updated is one into a hidden unit, the effect of the hidden unit's new value on the output units can be found in similar fashion, requiring further time proportional to the number of output units.

In a network with several hidden layers, however, this scheme helps only for the two layers following the connection whose weight was changed. Changing a weight on a connection into a hidden unit has an effect on *all* units in the layer after that hidden unit, so from there on, a full re-computation is required. Accordingly, we can expect to see only modest gains from local computation when applying the Metropolis algorithm to Bayesian learning for a general multi-layer perceptron network.

4.3 Variations on the Metropolis algorithm

Quite a few variations on the basic Metropolis algorithm have been proposed. Those variations that make use of the derivative of the energy function are discussed in Section 5. Here, I describe other variations that may be generally applicable.

Other acceptance functions. Functions other than that of equation (4.18) can be used for the probability of accepting a candidate state in the Metropolis algorithm. A whole family of acceptance functions for which detailed balance holds is given by Hastings (4:1970). The only such alternative that appears to have found wide use is the following, which I will call the *Boltzmann* acceptance function:

$$A(x,x') \;=\; P(x') \,/\, (P(x)+P(x')) \tag{4.27}$$

For distributions defined in terms of an energy function, this is

$$\begin{aligned} A(x,x') &= \exp\big(-E(x')/T\big) \,/\, \big(\exp\big(-E(x)/T\big)+\exp\big(-E(x')/T\big)\big) && (4.28)\\ &= 1 \,/\, \big(1+\exp\big((E(x')-E(x))/T\big)\big) && (4.29)\end{aligned}$$

Use of this acceptance function is equivalent to forgetting which of x and x' is the current state, and then selecting between them at random, according to their relative probabilities (Boltzmann probabilities, when the distribution is defined by an energy function).

We can verify that detailed balance holds when this acceptance function is used in conjunction with a symmetrical proposal distribution. For the nontrivial case where $x_k \neq x'_k$, but $x_i = x'_i$ for $i \neq k$, we have:

$$\begin{aligned} P(x)\,S_k(x,x'_k)\,A(x,x') &= S_k(x,x'_k)\,P(x)P(x')\,/\,(P(x)+P(x')) && (4.30)\\ &= S_k(x',x_k)\,P(x')P(x)\,/\,(P(x')+P(x)) && (4.31)\\ &= P(x')\,S_k(x',x_k)\,A(x',x) && (4.32)\end{aligned}$$

The Boltzmann acceptance function was known to be valid from the earliest days of the Metropolis algorithm (Wood, 1:1985), but was regarded as inferior to the standard Metropolis acceptance function, since it leads to fewer changes being accepted. In fact, as discussed in Section 4.4, neither of these acceptance functions is superior to the other in all contexts.

When the components of the state have only two possible values, as with the Ising model, use of the Boltzmann acceptance function of equation (4.27) makes the Metropolis algorithm identical to Gibbs sampling, assuming that candidate states are selected by flipping the value of one component.

Generalization to non-symmetric proposal distributions. Hastings (4:1970) generalized the Metropolis algorithm to allow its use with proposal distributions that do not satisfy the symmetry condition (4.16). Detailed balance is preserved by altering the acceptance function to compensate for the bias the asymmetry introduces. The Metropolis acceptance function (equation (4.18)) can be modified as follows:

$$A_k(x, x') \;=\; \min\big(1,\ P(x')S_k(x', x_k) \,/\, P(x)S_k(x, x'_k)\big) \tag{4.33}$$

The acceptance probability may now depend on k, since the proposal distribution may. Detailed balance can be verified to hold as follows (for the nontrivial case where $x_k \neq x'_k$, but $x_i = x'_i$ for $i \neq k$):

$$\begin{aligned} P(x)\, S_k(x, x'_k)\, A_k(x, x') &= \min\big(P(x)S_k(x, x'_k),\ P(x')S_k(x', x_k)\big) && (4.34)\\ &= \min\big(P(x')S_k(x', x_k),\ P(x)S_k(x, x'_k)\big) && (4.35)\\ &= P(x')\, S_k(x', x_k)\, A_k(x', x) && (4.36) \end{aligned}$$

Other acceptance functions, such as that of equation (4.27), may also be modified to work with non-symmetric proposal distributions. Note that to use a non-symmetric proposal distribution, it is necessary in general that one be able to calculate the numerical value of the probability of the particular candidate state chosen having been proposed. This is not necessary when the proposal distribution is symmetric.

The Gibbs sampling algorithm can be viewed as a special case of this generalized Metropolis algorithm, in which the proposal distribution for component k is just its conditional distribution given the current values of the other components — that is, $S_k(x, x'_k) = P(x'_k \mid \{x_i : i \neq k\})$. Noting that $P(x) = P(x_k \mid \{x_i : i \neq k\})\, P(\{x_i : i \neq k\})$, and that $A_k(x, x')$ is used only when $x_i = x'_i$ for $i \neq k$, the acceptance probability from equation (4.33) is seen to be

$$A_k(x, x') = \min\left[1,\ \frac{P(x'_k \mid \{x'_i : i \neq k\})\, P(\{x'_i : i \neq k\})\, P(x_k \mid \{x'_i : i \neq k\})}{P(x_k \mid \{x_i : i \neq k\})\, P(\{x_i : i \neq k\})\, P(x'_k \mid \{x_i : i \neq k\})}\right] = 1 \tag{4.37}$$

Hence the new "candidate" state is always accepted, giving the usual Gibbs sampling procedure.

Taking this view, Tierney (1:1991a) suggests an approximate form of Gibbs sampling that produces exactly unbiased results. (Ritter and Tanner (4:1992) described an approximate version of Gibbs sampling that is not exact.) In this method, one uses some proposal distribution, $S_k(x, x'_k)$, that may be only an approximation to $P(x'_k \mid \{x_i : i \neq k\})$, in conjunction with the acceptance function of equation (4.33). The better the approximation used, the lower the rejection rate will be, but exact results will be obtained (eventually) even with poor approximations.

The "rejectionless" method. In some applications of the Metropolis algorithm, the frequency with which candidate states are accepted is very low. The time spent in generating these rejected states can sometimes be eliminated by drawing from the distribution for the next *accepted* state, and from the distribution for how many rejections would occur before such an acceptance. Algorithms based on this idea are described by Bortz, Kalos, and Lebowitz (4:1975) and by Greene and Supowit (4:1986).

This *rejectionless* method applies only to global Metropolis algorithms. Note, however, that a local Metropolis algorithm in which the component to be updated is chosen at random can be viewed as a global Metropolis algorithm, with a proposal distribution that changes a randomly selected component. I will assume here that the state space is finite; generalizations to infinite state spaces are probably not of practical use.

For a global Metropolis algorithm, the probability that we will accept a transition from the current state, x, to a new state, x', is

$$w(x') \quad = \quad S(x, x')A(x, x') \tag{4.38}$$

where $S(x, x')$ is the probability of proposing x' as a candidate state, and $A(x, x')$ is the probability of accepting this candidate. The total probability of accepting a move from state x is $W = \sum_{x'} w(x')$. The distribution for the number of iterations, n, that x remains the current state is therefore as follows:

$$P(n) \quad = \quad W(1-W)^{n-1} \tag{4.39}$$

Rather than actually simulate these rejections, we can simply generate a random integer from the above distribution, and consider the current state to be repeated that many times. Alternatively, if the only motive is to find the time average of some function of state, the current state can simply be given a weight in this average equal to the expected number of repetitions, which is $1/W$.

Having decided how many times the current state is repeated, we pick a new state, chosing x' with probability $w(x')/W$.

The usefulness of this method depends on whether these operations can be performed efficiently for the system being simulated. Bortz, Kalos, and Lebowitz (4:1975) give an algorithm for the Ising system, with a proposal distribution that generates a candidate state by flipping a spin chosen at random. In this case, the number of possible candidate states is quite large, but there are only ten possible values of $w(x')$, resulting from two possible states of the flipped spin, and five possible environments of neighboring spins. This allows the algorithm to be efficiently implemented. They report that it is much faster than the standard Metropolis algorithm at low temperatures.

Greene and Supowit (4:1986) deal with systems where the number of possible values of $w(x')$ may be larger, but in which the number of possible candidate moves is relatively small. They use an incrementally-maintained tree structure to speed the selection of a new state.

Even when the rejectionless method can be implemented efficiently, however, it will not necessarily be effective in speeding up exploration. It is possible to imagine systems in which significant changes in state are rare, but in which there are always trivial moves that have a high probability of acceptance. Unless the latter can be identified and handled separately, they will render use of the rejectionless method pointless.

The generalized Swendsen-Wang algorithm. Swendsen and Wang (4:1987) developed a specialized algorithm for simulating Ising systems that is a spectacular improvement on previous methods, especially for simulations near the point of a phase transition. Here, I will describe a generalization of this algorithm due to Edwards and Sokal (4:1988). Formally, this generalization can be applied to a wide variety of systems, though the result is not always of practical use.

Assume that the distribution of interest over the state variable X can be expressed as follows:

$$P(x) \;=\; \frac{1}{Z}\prod_{b=1}^{B} W_b(x) \tag{4.40}$$

where Z is the appropriate normalization constant, and each of the factors is bounded, with $0 \leq W_b(x) \leq 1$. This formulation is equivalent to expressing the distribution using an energy function that is the sum of B terms, each of which is non-negative.

Edwards and Sokal express this model using auxiliary variables, $Y_1, \ldots, Y_B$, which take values in the interval $[0,1]$. The joint probability for the original state and these new variables is defined to be

$$P(x,y) \;=\; \frac{1}{Z}\prod_{b=1}^{B} \Theta\big(W_b(x) - y_b\big) \tag{4.41}$$

where $\Theta(z)$ is one for $z \geq 0$ and zero for $z < 0$. The marginal distribution for X in this model is then

$$\int_0^1 \cdots \int_0^1 \frac{1}{Z}\prod_{b=1}^{B} \Theta\big(W_b(x) - y_b\big)\, dy_B \cdots dy_1 \;=\; \frac{1}{Z}\prod_{b=1}^{B}\int_0^1 \Theta\big(W_b(x) - y_b\big)\, dy_b \tag{4.42}$$

$$=\; \frac{1}{Z}\prod_{b=1}^{B} W_b(x) \tag{4.43}$$

which is the same as the distribution for X in the original formulation of the model in equation (4.40).

Any scheme for sampling from the joint distribution of X and Y defined by equation (4.41) will therefore solve the original problem. In particular, we can try Gibbs sampling, alternately choosing new values for the auxiliary variables, Y_b, given the current x, and choosing a new value for X given the current y_b. Conditional on the current x, the Y_b are independent, with distributions

$$P(y_b \mid x) \;=\; \begin{cases} 1/W_b(x) & \text{if } 0 \leq y_b < W_b(x) \\ 0 & \text{otherwise} \end{cases} \tag{4.44}$$

This is easily sampled from, assuming $W_b(x)$ can be computed. Whether it is easy to sample from the conditional distribution for X given the y_b depends on the details of the model.

To see how large gains are possible with this approach, consider the Ising model with no external field, which can be described with the following energy function, obtained from the energy of equation (2.55) by adding a constant (which does not affect the resulting canonical distribution):

$$E(s) \;=\; \sum_{(i,j)\in\mathcal{N}} J(1 - s_i s_j) \tag{4.45}$$

The corresponding canonical distribution can be written as

$$P(s) \;=\; \frac{1}{Z} \prod_{(i,j)\in\mathcal{N}} W_{ij}(s), \quad \text{where } W_{ij}(s) \;=\; \begin{cases} 1 & \text{if } s_i = s_j \\ \exp(-2J) & \text{if } s_i \neq s_j \end{cases} \tag{4.46}$$

Now introduce auxiliary variables, Y_{ij}, taking values in $[0,1]$, with the joint distribution

$$P(s,y) \;=\; \frac{1}{Z} \prod_{(i,j)\in\mathcal{N}} \Theta\big(W_{ij}(s) - y_{ij}\big) \tag{4.47}$$

$$\;=\; \frac{1}{Z} \prod_{(i,j)\in\mathcal{N}} \left\{ \begin{array}{ll} 1 & \text{if } s_i = s_j \\ \Theta\big(\exp(-2J) - y_{ij}\big) & \text{if } s_i \neq s_j \end{array} \right\} \tag{4.48}$$

All we really need to known about the Y_{ij} are the values of the binary variables $D_{ij} = 1 - \Theta\big(\exp(-2J) - Y_{ij}\big)$. A value of one for D_{ij} can be seen as a "bond" between spins i and j. The conditional distributions for the D_{ij} needed for Gibbs sampling can be found as in equation (4.44):

$$P(D_{ij} = 1 \mid s) \;=\; P(Y_{ij} > \exp(-2J) \mid s) \;=\; \begin{cases} 1 - \exp(-2J) & \text{if } s_i = s_j \\ 0 & \text{if } s_i \neq s_j \end{cases} \tag{4.49}$$

Gibbs sampling for the D_{ij} thus consists of independently placing bonds between adjacent matching spins with probability $1 - \exp(-2J)$, and omitting bonds between adjacent spins that differ in value.

The conditional distribution for the spin values, S, given the bond variables, D_{ij}, is

$$P(s \mid d) \;\propto\; \prod_{(i,j)\in\mathcal{N}} \left\{ \begin{array}{ll} 1 & \text{if } s_i = s_j \text{ or } d_{ij} = 0 \\ 0 & \text{if } s_i \neq s_j \text{ and } d_{ij} = 1 \end{array} \right\} \tag{4.50}$$

Sampling from this distribution takes the form of a "percolation" problem, in which spins are grouped into clusters based on which bonds are present. The spins in each cluster are then all assigned the same new value, with $+1$ and -1 being equally likely. This choice of spin value is done independently for each cluster.

This procedure can be efficiently implemented, and is capable of flipping large clusters of spins in a single iteration. This is especially advantageous when simulating the Ising system at a temperature near the point of its phase transition, when large clusters of identical spins are present, but the system has not yet acquired a dominant orientation. Algorithms that pick new values for each spin locally take a long time to explore configurations in this situation, as the boundaries between clusters drift only slowly, via a random walk.

The success of this approach with the Ising and related models naturally encourages one to try applying it in other contexts. It appears applicable to certain image models that are closely related to the Ising system, as are some discussed by Geman and Geman (4:1984). Whether it will be useful for other statistical problems remains to be seen (see (Besag and Green, 1:1993) for some discussion).

Other variations of the Metropolis algorithm. Other variations on the Metropolis algorithm may be found in the papers in Section 4 of the bibliography. I will briefly describe two here.

Kennedy and Kulti (4:1985) and Bhanot and Kennedy (4:1985) have attempted to make use of "noisy" acceptance probabilities. This approach is potentially useful when such noisy approximations are easier to compute than the exact acceptance probabilities. For Bayesian inference problems, fast approximations can be computed based on the log likelihood for a randomly selected subset of the training cases (scaled up to give an estimate of what the full likelihood would be). Unfortunately, it appears difficult to fully control the potential error introduced by these techniques.

Goodman and Sokal (4:1989) provide a comprehensive review of *multigrid* methods. This class of algorithms is applicable to problems where the variables have a geometric structure, in which one can move between coarse-grained and fine-grained representations. Szeliski (2:1989) applies such methods to computer vision.

4.4 Analysis of the Metropolis and Gibbs sampling algorithms

In this section, I discuss and compare the performance of Gibbs sampling and of various forms of the Metropolis algorithm. These algorithms can be analysed from several points of view. I start with a largely intuitive look at sampling from a system of continuous variables with a Gaussian distribution. Following this, I discuss how the effort required to obtain an adequate sample scales with "system size". I then briefly review work concerning the "optimal" choice of acceptance function.

The Metropolis and Gibbs sampling algorithms for a multivariate Gaussian. The multivariate Gaussian distribution provides a simple model for problems with a continuous state space, in which the computational characteristics of the algorithms can easily be analysed and visualized. This analysis provides a guide to how rapidly the region around a local maximum in many continuous distributions will be explored, as such local maxima are often approximately Gaussian (i.e. the log of the probability density is approximately quadratic).

I will consider three algorithms here:

- Gibbs sampling, with components of the state updated systematically in turn.
- The local Metropolis algorithm, in which components of the state are updated systematically in turn, using some pre-defined proposal distribution centred on the current value, whose width may be different for each component.
- The global Metropolis algorithm, in which all components are updated simultaneously, using some pre-defined spherically-symmetric proposal distribution centred on the current state.

The n-dimensional state space will of course be represented in some coordinate system. By a "component", above, is meant either a single such coordinate, or perhaps a small group of related coordinates.

The operation of each of these algorithms is sensitive to some coordinate transformations, and insensitive to others. Gibbs sampling is invariant under translation and scaling of the coordinate system, but is sensitive to rotation. The local Metropolis algorithm is invariant under translation, but is sensitive to scaling and rotation. The global Metropolis algorithm is invariant under translation and rotation, but sensitive to scaling. Which algorithm is better can therefore depend on whether a good coordinate system can be found; for the Metropolis algorithm, this choice must be coordinated with the choice of proposal distribution.

For systems with local interactions, it is natural to use a separate set of coordinates to describe each sub-system. As discussed in Section 4.2 with respect to Lennard-Jonesium, this may lead to some components of the state being almost independent. Gibbs sampling is then an attractive method, provided, of course, that the required conditional distributions can be efficiently sampled from. Indeed, if the components were completely independent, every pass of the Gibbs sampling algorithm would produce an independent state. The local Metropolis algorithm also performs well in this situation, provided the proposal distributions are tailored to the scale of changes likely to be accepted for each component.

For a multivariate Gaussian, there is always a rotation of the coordinate system that results in the different coordinates of the state being independent. However, in a high-dimensional problem, finding this transformation may not be at all easy. (For further discussion of this topic, see Section 7.1.) Accordingly, for some problems we may have to live with a coordinate system that is far from ideal. This may be the case for Bayesian learning of multi-layer perceptrons, for example, as discussed in Section 4.2.

This situation is illustrated in Figure 4.1, for a bivariate Gaussian. The probability density is in this case given by equation (2.11) with the covariance matrix

$$\Sigma \;=\; \begin{bmatrix} \sigma_1^2 & \rho\sigma_1\sigma_2 \\ \rho\sigma_1\sigma_2 & \sigma_2^2 \end{bmatrix} \tag{4.51}$$

where σ_1 and σ_2 are the standard deviations of the marginal distributions for X_1 and X_2, and ρ is the correlation between X_1 and X_2. In the illustration, $\sigma_1 = \sigma_2 = 1$ and $\rho = 0.99$. Without loss of generality, we can take the mean, μ, to be zero. Note that if the coordinate system were rotated by $\pi/4$, the correlation would disappear; the two components would then have independent Gaussian distributions with standard deviations $\varsigma_1 = 0.10$ and $\varsigma_2 = 1.41$. We assume that we do not know enough to do this, however.

Figure 4.1(a) shows the performance of Gibbs sampling in this situation. The conditional distribution for X_1 given the current value of X_2, which is needed for Gibbs sampling, is Gaussian with mean $(\rho\sigma_1/\sigma_2)x_2$ and standard deviation $\sigma_1\sqrt{1-\rho^2} \approx 0.14$. The conditional distribution for X_2 given the current value of x_1 is analogous. The amount by which the state changes in a single Gibbs sampling pass is thus usually rather small — of the same order as the standard deviation in the *most* confined direction ($\varsigma_1 = 0.10$). Movement in the less confined direction (for which the standard deviation is $\varsigma_2 = 1.41$) is therefore fairly slow. Many more iterations will be needed to move from one point to another point that is largely independent than would be required if X_1 and X_2 were nearly independent. Furthermore, in the initial stages of the simulation, a large number of moves will be needed to reach the equilibrium distribution, if the initial state is far from the mean.

The local Metropolis algorithm behaves similarly with proposal distributions that are centred on the current value and are narrow enough that the acceptance probability is reasonably large. When the correlation is high, one might just as well use the global Metropolis algorithm, changing all the coordinates at once. To keep the acceptance rate reasonably high, the size of a global change must be limited to roughly the standard deviation in the most confined direction. The behaviour of the global Metropolis algorithm using such a proposal distribution, with an acceptance rate of about 56%, is illustrated in Figure 4.1(b). It is roughly similar to the behaviour with Gibbs sampling, though the occasional rejections slow the search somewhat.

Notice that in this example both Gibbs sampling and the Metropolis algorithm explore the less confined direction by a random walk. This further slows the rate at which independent states are visited. If the simulation takes steps of size about ς_1, one might expect that

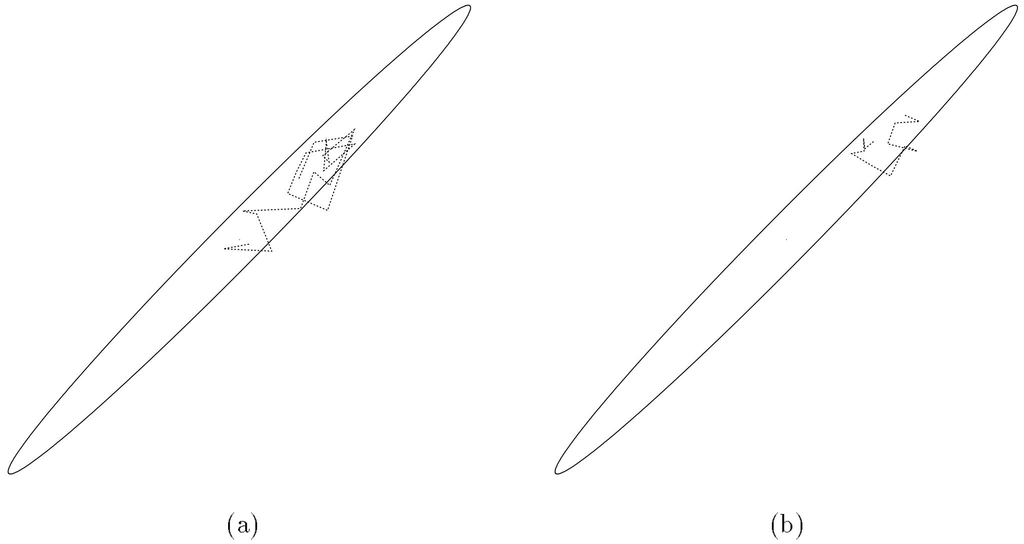


Figure 4.1: Sampling from a bivariate Gaussian distribution, (a) using Gibbs sampling, with one iteration choosing new values for first the horizontal and then the vertical coordinate, (b) using the global Metropolis algorithm, with a symmetric Gaussian proposal distribution with standard deviation 0.15, centred on the current state. The path followed by twenty iterations is shown in each case (note that for some iterations of the Metropolis algorithm the candidate state was rejected). The ellipses show the one standard deviation contour of the distribution, for which $\sigma_1 = \sigma_2 = 1$, and $\rho = 0.99$.

to move anywhere along the less confined direction, which extends a distance of about ς_2, would take about ς_2/ς_1 iterations. In fact, however, in k iterations, the simulation will usually move only around a distance $\varsigma_1\sqrt{k}$, as a consequence of the random walk nature of the moves. One must simulate about $(\varsigma_2/\varsigma_1)^2$ iterations in order to reach a state that is largely independent of the starting point; in the example of Figure 4.1, this is about 200 iterations.

This analysis gives a valid picture of the performance of these algorithms in high-dimensional problems. For two-dimensional problems such as that of Figure 4.1, however, one can do much better with the global Metropolis algorithm by using a proposal distribution that considers large changes to the state, of the order of the standard deviation in the *less* confined direction. In two dimensions, the probability that a jump in a random direction from a region of high probability will end up in another region of high probability, and hence be accepted, is inversely proportional to the size of the jump. On the other hand, the distance moved is directly proportional to the size. These two factors cancel, with the result that the total distance travelled in a given number of steps is roughly independent of the size of changes proposed, for any size between ς_1 and ς_2. If the distance is travelled in a random walk with many small steps, however, the net distance moved is usually much less than if only a few large steps were taken. It is therefore best to make large changes, despite the very low acceptance rate that results. However, in higher-dimensional problems where there is more than one orthogonal direction in which the standard deviation is small, the acceptance probability goes down at a higher rate as the size of a jump increases, and the strategy of taking large jumps no longer works well. This illustrates the danger of relying too much on low-dimensional examples for guidance in high-dimensional problems.

Scaling with system size. Since we are interested in using the Metropolis and Gibbs sampling methods to sample from distributions over high-dimensional spaces, we would like to know how their behaviour scales as the number of dimensions increases. As a model for this, we can ask how they behave when applied to a composite system that consists of many independent replicas of some simple system. Suppose the simple system has state variable X_0 and energy function $E_0(x_0)$. The composite system consisting of N replicas of this system will have state variable $X = \{X_1, \ldots, X_N\}$, with all the X_i being identical in form to X_0, and energy function $E(x) = E_0(x_1) + \cdots + E_0(x_N)$.

Of course, we would not bother to simulate the composite system if we knew that it had this structure, since the system's properties can be found more easily by simulating just the simple system. However, studying this case may provide insight into less trivial systems where the X_i are only weakly dependent. Furthermore, a system of interest might take this form, with the X_i nearly independent, only if its state were to be decomposed into components using some coordinate system other than the natural one. In high dimensional problems, it may not be easy to find this good coordinate system.

Consider simulating the composite system using Gibbs sampling or a local Metropolis method, in which the changes in state affect only one of the X_i (or part of one X_i), with each X_i being visited in turn. Each of the X_i will then evolve completely independently, in exactly the same way as they would in a simulation of a single instance of the simple system. As the size of the composite system increases, the computation time required bring the simulation to equilibrium, and, once it has reached equilibrium, the computation time needed to move from one point to a nearly independent point, will both simply be proportional to N, reflecting the fact that each full iteration of the simulation must update each of the N replicas.

This linear scaling is obtained using knowledge of how to decompose the state into independent components. If we don't know enough to do this, it may be more appropriate to use a global Metropolis algorithm, in which the candidate state proposed, x^*, differs in all components. Let us also assume that the proposed changes in each component are independent and identically distributed. We can then ask how the probability of accepting a move scales as N increases, while the proposal distributions remain fixed. This acceptance probability depends on the difference in energy between the current state and the candidate state. The expected value of this difference once the simulation has reached equilibrium can be expressed in terms of the expected difference in energy when using the same proposal distribution with the simple system, as follows:

$$\big\langle E(x^*) - E(x) \big\rangle \;=\; \sum_i \big\langle E_0(x_i^*) - E_0(x_i) \big\rangle \;=\; N \big\langle E_0(x_0^*) - E_0(x_0) \big\rangle_0 \tag{4.52}$$

Thus the average magnitude of the change in energy gets larger in direct proportion to system size. Furthermore, from the Law of Large Numbers, we know that the distribution will become concentrated in the vicinity of this average. If $\big\langle E_0(x_0^*) - E_0(x_0) \big\rangle_0$ is positive, the acceptance probability will therefore decline exponentially as N increases.

But might $\big\langle E_0(x_0^*) - E_0(x_0) \big\rangle_0$ be negative? If so, the acceptance probability would instead approach one as N increases. Unfortunately, this is not possible, as is shown by Creutz (5:1988). To see this, first note that the following relation holds at equilibrium when candidate states are generated using a symmetrical proposal distribution, $S(x, x^*)$:

$$\left\langle \frac{P(x^*)}{P(x)} \right\rangle \;=\; \sum_{\tilde{x}} P(\tilde{x}) \sum_{\tilde{x}^*} S(\tilde{x}, \tilde{x}^*) \, \frac{P(\tilde{x}^*)}{P(\tilde{x})} \;=\; \sum_{\tilde{x}^*} P(\tilde{x}^*) \sum_{\tilde{x}} S(\tilde{x}^*, \tilde{x}) \;=\; 1 \tag{4.53}$$

Applying this to the simple system defined by E_0, we get

$$\begin{aligned} 1 \;=\; \big\langle P_0(x_0^*)/P_0(x_0)\big\rangle_0 \;&=\; \big\langle \exp\big(-\big(E_0(x_0^*) - E_0(x_0)\big)/T\big)\big\rangle_0 &(4.54)\\ &\geq\; \exp\big(-\big(\big\langle E_0(x_0^*) - E_0(x_0)\big\rangle_0\big)/T\big) &(4.55)\end{aligned}$$

The last step uses Jensen's Inequality, and the fact that exp is a concave function. By taking the log of both sides, we can now conclude that $\big\langle E_0(x_0^*) - E_0(x_0)\big\rangle_0 \geq 0$. Equality will hold only in trivial cases.

The time required to sample using the global Metropolis algorithm will therefore grow exponentially with system size if we use a fixed proposal distribution. Of course, this is not the best strategy. Instead, we can change the proposal distribution as N increases so as to keep the expected change in energy constant. Often, for example, the candidate state is obtained from a proposal distribution centred on the current state. In this case, reducing the width of the proposal distribution will result in smaller changes in state, and hence smaller changes in energy, assuming that the energy is a smooth function. Reducing the size of the proposed changes will increase the number of accepted moves required to move to an independent point, however. According, the time required to generate an adequate sample will increase more than linearly with system size. The form of this scaling will depend on the details of the problem and of the sampling method.

Choice of acceptance function. A number of papers have addressed the question of whether the standard "Metropolis" acceptance function of equation (4.18) or the alternative "Boltzmann" acceptance function of equation (4.27) is better. For systems whose components have only two states, such as the Ising system, the latter choice is equivalent to Gibbs sampling.

This question might appear to have been answered by Peskun (4:1973), who showed that, for any fixed proposal distribution, the Metropolis acceptance function is optimal amongst all acceptance functions in a large class. This result applies to forms of the Metropolis algorithm that can be expressed in terms of a single proposal distribution, which potentially may change any part of the state. This includes what I have called "global" Metropolis algorithms, in which the candidate state generally differs in all components, and also those "local" Metropolis algorithms in which the component to be updated is selected at random, since these can be rephrased in terms of a single proposal distribution. The performance criterion used by Peskun is the "asymptotic variance" of a Monte Carlo estimate of the expectation of some function based on states generated by the Markov chain, defined as the limit of N times the variance of the estimate after N iterations, as N goes to infinity. The results of Peskun (4:1973, 4:1981) also imply that it is best to use a proposal distribution that gives zero probability to the current state.

Peskun's results do not really settle the issue, however, for two reasons. First, asymptotic variance is an appropriate criterion for assessing performance only if very precise estimates are desired. This is often not the case. In many problems, even a single state drawn from the equilibrium distribution would be highly informative, and much of the uncertainty in the results may be due to doubts about whether the Markov chain has truly converged to this equilibrium distribution, rather than to the variance in the estimator that arises from use of a finite sample.

Perhaps more importantly, the result simply does not apply to the most commonly used method — the local Metropolis algorithm with components updated in a fixed order. Indeed, in this context, using the standard Metropolis acceptance function along with a proposal distribution that gives zero probability to the current state can produce a chain that is not even ergodic.

To see this, consider a system of binary variables in which all states have the same probability. Suppose that the B_k are applied in order systematically, and that S_k gives probability one to the candidate in which the value of the k-th variable is flipped. Since all states have equal probability, all the candidate moves will be accepted when using the Metropolis acceptance function. A single pass over the variables will thus change them all; the next pass will change them all back to their original values. Other states are never seen.[6] The problem remains in the less trivial system of three binary variables where all states have the same probability except for (0, 1, 0) and (1, 0, 1), whose probabilities may be arbitrary. For a real-life example of the problem, see (Friedberg and Cameron, 4:1970).

For most problems, using the Metropolis acceptance function with systematic updating does produce an ergodic chain, but the asymptotic variance may still be higher than is obtained using the same method with the Boltzmann acceptance function. This is found empirically to be the case for an Ising system by Cunningham and Meijer (4:1976), and also by Peskun (4:1981). Further theoretical results bearing on these questions have been obtained by Frigessi, di Stefano, Hwang, and Sheu (4:1993). The issues still remain unclear, though it appears that common opinion favours using the Metropolis acceptance function in most circumstances.

[6] In fact, for this system, the Metropolis acceptance function produces a non-ergodic chain even if the component to update is chosen at random — the number of components having the value 1 will always be even at even iterations and odd at odd iterations. Time averages turn out correctly however, and the problem does not persist in the less trivial example mentioned next.

5. The Dynamical and Hybrid Monte Carlo Methods

In this section, I describe Markov chain sampling methods that derive from the "molecular dynamics" approach (Alder and Wainwright, 5:1959), which was developed concurrently with the Metropolis algorithm as a means of simulating physical systems. These methods are widely applicable to problems with continuous state variables, provided appropriate derivatives can be calculated. (If discrete variables are present as well, one could use a Markov chain that combines these methods with those of the previous section.)

Dynamical sampling is based on a physical analogy. The gradient of the potential energy for a physical system with respect to its configuration coordinates defines the "force" that acts to change this configuration, via its effect on the system's momentum. When a physical system is in contact with a heat reservoir, it also experiences influences that can be modeled as being random. These dynamical and stochastic effects together result in the system visiting states with a frequency given by its canonical distribution.

Simulating the dynamics of such a physical system therefore provides a way of sampling from the canonical distribution. Dynamical simulation also allows one to observe in detail how the system behaves as it visits states with this distribution. Through much of the literature, the latter motive for dynamical simulation has been dominant, the general view apparently being that when the only purpose is to sample from the canonical distribution, methods based on the Metropolis algorithm are to be preferred.

Recently, however, it has become clear that, at least for some problems, the dynamical method can be faster than a direct approach using the Metropolis algorithm, largely because the dynamical method avoids the random walk behaviour inherent in simple forms of the Metropolis algorithm. Indeed, it can be advantageous to invent an artificial dynamics for problems that have no connection with any real physical system. It is convenient to retain the physical terminology even in such cases, however. Accordingly, in this section I will talk only of canonical distributions defined by an energy function, as in equation (2.54). Any distribution that is nowhere zero can be put in this form, so this is largely a matter of representation.

5.1 The stochastic dynamics method

I will first discuss methods that straightforwardly implement dynamical simulation along with a connection to a heat reservoir. The dynamical aspect derives from the early work on "molecular dynamics" (Alder and Wainwright, 5:1959). On its own, dynamical simulation samples from the microcanonical distribution for states at a given total energy. Rather surprisingly, it was apparently not until much later, with the work of Andersen (5:1980), that the idea of introducing a stochastic element in order to instead sample from the canonical distribution was thought of. (This delay may perhaps be explained by the fact that for many applications in statistical physics either distribution may be used, since they are equivalent in the thermodynamic limit.)

Phase space. Suppose that we wish to sample from the canonical distribution for a set of real variables, $Q = \{Q_1, \ldots, Q_n\}$, with respect to the potential energy function, $E(q)$, assumed to be differentiable with respect to the q_i. Taking the temperature to be one, for convenience, this canonical distribution is

$$P(q) \;=\; \frac{1}{Z_E} \exp\big(-E(q)\big) \tag{5.1}$$

where Z_E is a constant such that the distribution integrates to one. For a real molecular dynamics problem, the Q_i might be the position coordinates for the molecules being simulated (three coordinates for each particle). For a problem in Bayesian statistical inference, the Q_i might be the unknown model parameters. I will refer to the Q_i as the "position" variables even when such usage is strictly metaphorical.

We introduce another set of real variables, $P = \{P_1, \ldots, P_n\}$, one P_i for each Q_i, with a kinetic energy function, $K(p) = \frac{1}{2}\sum_i p_i^2$. The canonical distribution for these variables is

$$P(p) \;=\; \frac{1}{Z_K}\exp\big(-K(p)\big) \;=\; (2\pi)^{-n/2}\exp\big(-\tfrac{1}{2}\sum_i p_i^2\big) \tag{5.2}$$

I.e. the P_i are independent, and have Gaussian distributions with zero mean and unit variance. For a molecular dynamics problem, the P_i have real significance — they are the components of the momentum for the various particles. For other problems, the P_i are introduced solely to allow the problem to be given a dynamical formulation. They will nevertheless be referred to as the "momentum" variables.

The combination of the position and momentum variables is known as *phase space*. The total energy function for points in phase space, known as the *Hamiltonian*, is

$$H(q,p) \;=\; E(q) + K(p) \;=\; E(q) \;+\; \tfrac{1}{2}\sum_i p_i^2 \tag{5.3}$$

The canonical distribution over phase space defined by this energy function is

$$\begin{aligned} P(q,p) \;&=\; \frac{1}{Z_H}\exp\big(-H(q,p)\big) && (5.4)\\ &=\; \left[\frac{1}{Z_E}\exp\big(-E(q)\big)\right]\cdot\left[\frac{1}{Z_K}\exp\big(-K(p)\big)\right] && (5.5)\\ &=\; P(q)\,P(p) && (5.6)\end{aligned}$$

The distribution for Q from which we wish to to sample, given by equation (5.1), is therefore just the marginal distribution for Q with respect to the canonical distribution over phase space. If we can find some way of sampling from this distribution over phase space, we can obtain a sample of values for Q by just ignoring the values we obtained for P.

Hamiltonian dynamics. We can use the Hamiltonian function to define a dynamics on phase space, in which the q_i and p_i are regarded as functions of a "time" parameter, τ, satisfying the following dynamical equations:

$$\begin{aligned} \frac{dq_i}{d\tau} \;&=\; +\frac{\partial H}{\partial p_i} \;=\; p_i && (5.7)\\ \frac{dp_i}{d\tau} \;&=\; -\frac{\partial H}{\partial q_i} \;=\; -\frac{\partial E}{\partial q_i} && (5.8)\end{aligned}$$

For a real physical system, τ will represent actual physical time. In statistical problems, the "time" is entirely artificial. This continuous time parameter, written as τ, should be distinguished from the discrete time parameter of a Markov chain, written as t.

The value of H (the total energy) is conserved as q_i and p_i evolve through time according to the above dynamics. This is easily seen as follows:

$$\frac{dH}{d\tau} \;=\; \sum_i\left[\frac{\partial H}{\partial q_i}\frac{dq_i}{d\tau} + \frac{\partial H}{\partial p_i}\frac{dp_i}{d\tau}\right] \;=\; \sum_i\left[\frac{\partial H}{\partial q_i}\frac{\partial H}{\partial p_i} - \frac{\partial H}{\partial p_i}\frac{\partial H}{\partial q_i}\right] \;=\; 0 \tag{5.9}$$

The dynamics also preserves the volumes of regions of phase space — i.e. if we follow how the points in some region of volume V move according to the dynamical equations, we find that the region where these points end up after some given period of time also has volume V. We can see this by looking at the divergence of the motion in phase space:

$$\sum_i \left[\frac{\partial}{\partial q_i}\left(\frac{dq_i}{d\tau}\right) + \frac{\partial}{\partial p_i}\left(\frac{dp_i}{d\tau}\right)\right] \;=\; \sum_i \left[\frac{\partial H}{\partial q_i \partial p_i} - \frac{\partial H}{\partial p_i \partial q_i}\right] \;=\; 0 \tag{5.10}$$

This result is known as Liouville's theorem.

Finally, the dynamics can be followed backward in time, with a result that is the inverse of following the dynamics forward in time for the same duration.

Together, these properties imply that the canonical distribution of equation (5.4) is invariant with respect to any transformation that consists of following Hamiltonian dynamics for some pre-specified period of time. To see this, consider any small region in phase space, of volume δV, where the total energy, H, is H_0. The probability of being in this region after the transformation is just the probability of being at some point that maps to this region before the transformation. Due to Liouville's theorem and the invertibility of the transformation, the region containing such points also has volume δV, and due to energy conservation, points in this region also have total energy H_0. It follows that if the probability density before the transformation is given by the canonical distribution, which depends only on H, then the transformed points will also have a canonical distribution.

Clearly, however, following Hamiltonian dynamics does not sample points from the canonical distribution ergodically, since the total energy remains constant. At most, it may sample from the microcanonical distribution for a given energy. Later we will see how to introduce stochastic transitions that can move the system to states of different total energy.

The "leapfrog" discretization. In practice, we cannot follow the dynamics defined by equations (5.7) and (5.8) exactly, but must instead discretize these equations using some non-zero time step, inevitably introducing some error. It is, however, both possible and desirable to use a discretization for which Liouville's theorem still holds exactly. This ensures that the discretization errors will not lead to such qualitatively wrong behaviours as convergence of all points to an attracting point. Preserving phase space volume will also be crucial for the hybrid Monte Carlo method of Section 5.2.

The *leapfrog* discretization is a commonly used scheme that preserves phase space volume and is also time reversible (which will also prove important for the hybrid Monte Carlo method). A single leapfrog iteration calculates approximations to the position and momentum, $\hat{q}$ and $\hat{p}$, at time $\tau+\epsilon$ from $\hat{q}$ and $\hat{p}$ at time τ as follows:

$$\hat{p}_i(\tau+\tfrac{\epsilon}{2}) \;=\; \hat{p}_i(\tau) \;-\; \frac{\epsilon}{2}\frac{\partial E}{\partial q_i}\big(\hat{q}(\tau)\big) \tag{5.11}$$

$$\hat{q}_i(\tau+\epsilon) \;=\; \hat{q}_i(\tau) \;+\; \epsilon\,\hat{p}_i(\tau+\tfrac{\epsilon}{2}) \tag{5.12}$$

$$\hat{p}_i(\tau+\epsilon) \;=\; \hat{p}_i(\tau+\tfrac{\epsilon}{2}) \;-\; \frac{\epsilon}{2}\frac{\partial E}{\partial q_i}\big(\hat{q}(\tau+\epsilon)\big) \tag{5.13}$$

This consists of a half-step for the p_i, a full step for the q_i, and another half-step for the q_i. (One can instead do a half-step for the q_i, a full step for the p_i, and another half-step for the q_i, but this is usually slightly less convenient.) To follow the dynamics for some period of time, $\Delta\tau$, a value of ϵ that is thought to be small enough to give acceptable error is chosen, and equations (5.11) to (5.13) are applied for $\Delta\tau/\epsilon$ iterations in order to reach the target time. When this is done, the last half-step for p_i in one iteration will be immediately followed by the first half-step for p_i in the next iteration. All but the very first and very last such half-steps can therefore be merged into full steps starting at times $\tau + k\epsilon + \epsilon/2$, which "leapfrog" over the steps for the q_i that start at times $\tau + k\epsilon$.

For smooth energy functions, as ϵ approaches zero, the error in following the dynamics for a time $\Delta\tau$ using the leapfrog method approaches zero (while the number of iterations required to do this approaches infinity). The magnitude of the error in the coordinates at the end of the trajectory is $O(\epsilon^2)$.

To see informally why the error should be of this order, consider first the *local error* in a half-step for p that starts with exact values for $p(\tau)$ and $q(\tau)$. This is

$$\begin{aligned}
\left|\hat{p}(\tau+\tfrac{\epsilon}{2}) - p(\tau+\tfrac{\epsilon}{2})\right| &= \left|\left[p(\tau) - \tfrac{\epsilon}{2}F(\tau)\right] - \left[p(\tau) - \int_0^{+\frac{\epsilon}{2}} F(\tau+\delta)\,d\delta\right]\right| && (5.14)\\
&= \left| -\tfrac{\epsilon}{2}F(\tau) + \int_0^{+\frac{\epsilon}{2}} \left(F(\tau) + O(\delta)\right) d\delta\right| && (5.15)\\
&= \left| -\tfrac{\epsilon}{2}F(\tau) + \tfrac{\epsilon}{2}F(\tau) + O(\epsilon^2)\right| && (5.16)\\
&= O(\epsilon^2) && (5.17)
\end{aligned}$$

where $F(\tau) = E'(q(\tau))$. The error in a full step for p that starts with exact values is of smaller order:

$$\begin{aligned}
\left|\hat{p}(\tau+\epsilon) - p(\tau+\epsilon)\right| &= \left|\left[p(\tau) - \epsilon F(\tau+\tfrac{\epsilon}{2})\right] - \left[p(\tau) - \int_{-\frac{\epsilon}{2}}^{+\frac{\epsilon}{2}} F(\tau+\tfrac{\epsilon}{2}+\delta)\,d\delta\right]\right| && (5.18)\\
&= \Big| -\epsilon F(\tau+\tfrac{\epsilon}{2}) && (5.19)\\
&\qquad + \int_{-\frac{\epsilon}{2}}^{+\frac{\epsilon}{2}} \left(F(\tau+\tfrac{\epsilon}{2}) + \delta F'(\tau+\tfrac{\epsilon}{2}) + O(\delta^2)\right) d\delta\Big| && (5.20)\\
&= \left| -\epsilon F(\tau+\tfrac{\epsilon}{2}) + \epsilon F(\tau+\tfrac{\epsilon}{2}) + O(\epsilon^3)\right| && (5.21)\\
&= O(\epsilon^3) && (5.22)
\end{aligned}$$

A full step for q starting with exact values for $q(\tau)$ and $p(\tau+\frac{\epsilon}{2})$ also gives an error that is $O(\epsilon^3)$.

Of course, we are really interested in the *global error* that takes into account the fact that all but the first half-step of the trajectory starts with the inexact values for p and q from the previous step. Simulating the dynamics for a time $\Delta\tau$ using a stepsize of ϵ requires $O(1/\epsilon)$ full steps. As one would informally expect, the errors in each step accumulate additively, giving a total error that is $O(\epsilon^2)$ for the whole series of full steps for q and p. Adding in the $O(\epsilon^2)$ error from the half steps at the beginning and end of the trajectory leaves the total error at $O(\epsilon^2)$.

Actually, the error in the trajectory is, in itself, of no significance for sampling applications. What does matter is the error in the value of the total energy, H, and this will generally be of the same order. Hence, whereas for the true dynamics, H remains constant, it will *not* be exactly constant when the dynamics are simulated using the leapfrog method, except in the limit as ϵ approaches zero. This will introduce some systematic error into the sampling.

It is, however, easy to see that the leapfrog method is exactly time-reversible — i.e. that applying it with a negative value of ϵ undoes the effect of applying it with the corresponding positive ϵ — and that it preserves phase space volume. In fact, each of steps (5.11) to (5.13) preserves phase space volume individually, since they each increment or decrement

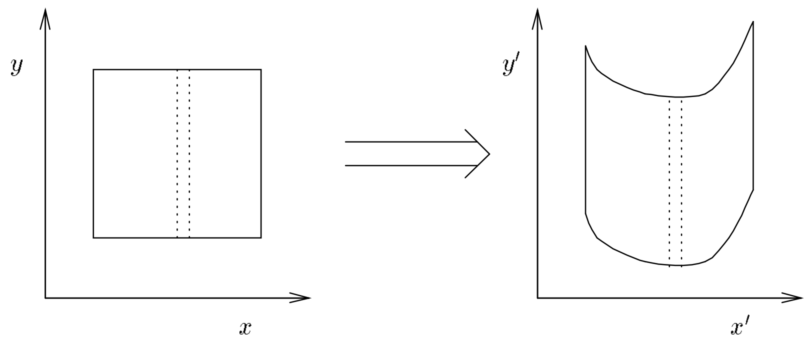


Figure 5.1: Illustraton of volume conservation by a shear transformation. The region on the left is transformed according to $x' = x$, $y' = y + f(x)$, for some function $f(\cdot)$. That this transformation preserves the volume can be seen by visualizing what happens to a narrow vertical strip of the original region — it may be shifted vertically, but its length and width remain the same, and it does not end up overlapping other such strips.

one of the q_i or p_i by an amount that depends only on the *other* coordinates, producing a "shear" transformation that does not alter a region's volume. (This property of shear transformations is illustrated in Figure 5.1.) Note that this relies on the Hamiltonian being the sum of a potential energy term, which is a function only of position coordinates, and a kinetic energy term, which is a function only of momentum coordinates. For Hamiltonians where the position and momentum interact, Liouville's theorem still holds for the true dynamics, but not for the leapfrog discretization (except in the limit as the stepsize goes to zero).

I have assumed above that the arithmetic in equations (5.11) to (5.13) is exact. When floating-point arithmetic is used, the leapfrog method will generally not be exactly time reversible, since the identity $(a+b)-b = a$ does not hold in floating-point arithmetic. Exact time-reversibility can be achieved by representing the q_i and p_i as fixed-point numbers, and performing the increments and decrements of equations (5.11) to (5.13) using fixed-point arithmetic. Exact conservation of phase space volume is then ensured by the fact that each discretized, fixed-point state corresponds to the same volume in real-valued state-space.

The advantages of a fixed-point representation are discussed by Earn and Tremaine (5:1992). For our purposes, however, the effects of floating-point round-off errors may be minor, as they are probably overwhelmed by the stochastic transitions that will be discussed next.

Stochastic dynamics. If we ignore the changes in H caused by discretization errors, simulating Hamiltonian dynamics for some pre-defined period of time, using the leapfrog discretization with some pre-defined stepsize, leaves the canonical distribution over phase space invariant. Such deterministic transitions can be alternated with stochastic transitions that are capable of changing H, and that also leave the canonical distribution invariant, in order to produce an ergodic Markov chain that samples from the canonical distribution.

It is convenient to use stochastic transitions that change only the P_i, as then $E(q)$ need not be re-evaluated. The simplest choice is to just replace all the P_i by values picked from their distribution conditional on the current q_i. (This can be seen as a Gibbs sampling step.) Since the P_i and Q_i are independent in the canonical distribution of equation (5.4), the conditional distribution for the P_i is just their marginal distribution (equation (5.2)), which is Gaussian, and easy to sample from. This is similar to the approach of Andersen (5:1980), except that he replaces only one of the P_i, which being chosen at random.

In physical terms, a stochastic transition can be viewed as representing the effect on the system of contact with a heat reservoir. Such contact can be of varying strength, suggesting that we might also try controlling how large an effect the stochastic transitions have relative to that of the dynamical transitions. For stochastic transitions that completely replace the P_i with newly chosen values, such control can be exercised by adjusting the duration of the dynamical trajectories simulated between stochastic transitions. An alternative is to simulate only short trajectories (perhaps just a single leapfrog iteration), but to use stochastic transitions of the following form:

$$p_i' \;=\; \alpha p_i \;+\; (1-\alpha^2)^{1/2} n_i \tag{5.23}$$

where n_i is drawn from a Gaussian distribution with zero mean and unit variance. Here, α is a parameter that controls how much effect the stochastic transitions have, with $0 \leq \alpha < 1$. One can easily show that the canonical distribution is invariant under the above transition.

Setting α to zero in equation (5.23) gives the previous method of just replacing the momenta. If this is done after every leapfrog iteration, there is a large random walk aspect to the motion, which is generally undesirable. When α is set to a value only slightly less than one, the momenta are only slightly altered in each stochastic transition, much reducing this effect. These issues will be discussed further later. Use of a weak connection to the heat reservoir also relates to the technique of simulated annealing discussed in Section 6.1.

In addition to allowing points in phase space with different total energies to be visited, stochastic transitions also help ensure that all points of a given energy are accessible, i.e. that the Markov chain is ergodic. Cases where the chain is still not ergodic do exist, however (see Section 5.4).

Other discretization schemes. The leapfrog discretization is not the only one that preserves phase-space volume. A number of other "canonical" or "symplectic" discretizations having this property (in fact, strong properties) have been identified in recent years. References to this work may be found in Section 5 of the bibliography. Some of these discretizations are accurate to a higher order than the leapfrog method, though they require more evaluations of the energy gradient for each iteration.

It is also possible to construct composite discretizations by applying a series of leapfrog steps (or steps of other symplectic methods), each of which operates on a subset of the state variables, or is based on a subset of the terms in the energy function.

Sexton and Weingarten (5:1992) discuss such a scheme, applicable when the potential energy can be written as $E(q) = E_0(q) + E_1(q)$, where the derivatives of E_0 can be calculated much more quickly than those of E_1. These characteristics can be exploited by using an elaboration of the leapfrog discretization in which each iteration consists of the following series of updates to q and p:

$$\begin{array}{l} p \;\leftarrow\; p \;-\; (\epsilon/2)\,\partial E_1/\partial q \\[4pt] \text{Repeat } n \text{ times:} \left\{ \begin{array}{lcl} p & \leftarrow & p \;-\; (\epsilon/2n)\,\partial E_0/\partial q \\ q & \leftarrow & q \;+\; (\epsilon/n)\,p \\ p & \leftarrow & p \;-\; (\epsilon/2n)\,\partial E_0/\partial q \end{array} \right. \\[4pt] p \;\leftarrow\; p \;-\; (\epsilon/2)\,\partial E_1/\partial q \end{array} \tag{5.24}$$

With $n = 1$, this is equivalent to the usual leapfrog method. Using a larger value for n reduces the discretization error associated with E_0, without significantly increasing the computation time (since $\partial E_0 / \partial q$ is assumed to be easy to calculate). This scheme might be useful when sampling from a Bayesian posterior distribution, with E_0 being minus the log of the prior (usually a simple function), and E_1 minus the log of the likelihood.

Many workers in the neural network field favour "on-line" learning methods, in which parameters are updated on the basis of each training case in turn, rather than on the basis of the entire training set. They argue that most training sets contain considerable redundancy, with many training cases contributing only a small increment of information about the parameters. In such circumstances, computing the likelihood for each iteration using all the training cases seems wasteful.

In the context of dynamical simulation, this philosophy might lead us to write the potential energy for a Bayesian inference problem in the following form:

$$E(q) \;=\; \frac{1}{C}\sum_{i=1}^{C}\Big[P_0(q) + CL_i(q)\Big] \;=\; \frac{1}{C}\sum_{i=1}^{C} E_i(q) \tag{5.25}$$

where P_0 is minus the log of the prior, and L_i is minus the log of the likelihood for training case i. We can now use a composite discretization for the dynamics in which each full iteration is composed of leapfrog iterations for $E_1, \ldots, E_C, E_C, \ldots, E_1$ in succession. Note that this combination preserves time reversibility, which will be important for the hybrid Monte Carlo method. If the training set were completely redundant, so that all the E_i were actually identical, and equal to the full E, we could use the same stepsize for each of these component iterations as we could in the ordinary leapfrog method. We would then be able to move a given distance while using only a fraction $1/C$ of the computing time required with the usual method (assuming that the computation is dominated by the time required to evaluate the L_i). In practice, we would likely have to use a somewhat smaller stepsize, but a large gain might still be possible.

For problems with a "local" structure, it may be interesting to consider composite discretizations in which each component step changes only a subset of the variables. This could be useful if recomputation after a change of state confined to variables in such a subset can be done locally at much less cost than is required to recompute after a change to all variables.

Applications of the stochastic dynamics method. Dynamical methods have historically been applied primarily to the simulation of classical physical systems, perhaps because in these applications the dynamics has a direct interpretation. Recently, there have been important applications to quantum chromodynamics, using a dynamics that is artificial. Other applications outside the traditional domain, such as to statistical inference, may be equally attractive.

Example: Lennard-Jonesium. The archetypal application of these methods is to the "molecular dynamics" simulation of real or hypothetical substances — Lennard-Jonesium being an example of the latter. The partial derivatives of the energy of equation (2.57) with respect to the position coordinates of the molecules are straightforward to obtain, allowing the dynamics to be simulated by the leapfrog or other methods. As with the applications of the Metropolis algorithm to this system, the very small effects of distant molecules are often ignored, in which case the computations are "local", and the computation time scales linearly with the size of the system, assuming that the time to settle to equilibrium and the significance of the error introduced by a finite discretization are independent of system size.

It appears that most such simulations are done without any stochastic element, and hence sample from the microcanonical distribution at the original total energy. However, it was systems such as this that motivated Andersen (5:1980) to develop the stochastic dynamics method for sampling from the canonical distribution.

Example: Multi-layer perceptrons. In (Neal, 2:1993a), I report experiments with applying the stochastic dynamics method to Bayesian learning for multi-layer perceptrons. The backpropagation algorithm readily provides the derivatives of the log likelihood of equation (2.28), to which are added the easily-obtained derivatives of the log prior probability of equation (2.42), giving the derivatives of the energy needed to perform the dynamical simulations. In these experiments, the prior weight variances, σ_u^2 and σ_v^2, and the noise variance in the likelihood, σ^2, were considered unknown, which introduces some extra complexity which will not be described here. (Automatic determination of these variances is, however, one of the main advantages of Bayesian over conventional learning in this application.)

When the stochastic transitions of equation (5.23) are used, the algorithm is similar to the batch version of the standard "backprop with momentum" method of neural network training, except that the latter lacks the stochastic element. A value of α very close to one (0.999) appears to be desirable in order to avoid random walk behaviour. The experiments in (Neal, 2:1993a) show that, with a careful choice of stepsize, this method can produce a good approximation to the true Bayesian result in a reasonable amount of time. Further work is needed to explore the potential of the stochastic dynamics method and compare its merits for this problem with those of the hybrid Monte Carlo method described in the next section.

5.2 The hybrid Monte Carlo algorithm

A Markov chain based on stochastic dynamics will sample from the correct distribution only in the limit as the stepsize used in discretizing the dynamics goes to zero. The bias introduced by using a non-zero stepsize is eliminated in the *hybrid Monte Carlo* method of Duane, Kennedy, Pendleton, and Roweth (5:1987), which can also be seen as a form of the Metropolis algorithm.

The hybrid Monte Carlo algorithm. Like the stochastic dynamics method, the hybrid Monte Carlo algorithm samples points in phase space by means of a Markov chain in which stochastic and dynamical transitions alternate. The stochastic transitions can be any of the types used for the stochastic dynamics method, such as the update of equation (5.23). Typically, the momenta are replaced with new values via Gibbs sampling (equivalently, by using equation (5.23) with $\alpha = 0$); this form of stochastic transition will be assumed for the hybrid Monte Carlo method unless otherwise stated. The dynamical transitions in the hybrid Monte Carlo method are also similar to those in the stochastic dynamics method, but with two changes — first, a random decision is made for each transition whether to simulate the dynamics forward in time or backward in time; second, the point reached by following the dynamics is only a candidate for the new state, to be accepted or rejected based on the change in total energy, as in the Metropolis algorithm. If the dynamics were simulated exactly, the change in H would always be zero, and the new point would always be accepted. When the dynamics is simulated with a non-zero stepsize, H may change, and moves will occasionally be rejected. These rejections exactly eliminate the bias introduced by inexact simulation.

In detail, given values for the magnitude of the leapfrog stepsize, ϵ_0, and the number of leapfrog steps, L, the dynamical transitions of the hybrid Monte Carlo algorithm operate as follows:

1) Randomly choose a direction, λ, for the trajectory, with the two values $\lambda = +1$, representing a forward trajectory, and $\lambda = -1$, representing a backward trajectory, being equally likely.

2) Starting from the current state, $(q, p) = (\hat{q}(0), \hat{p}(0))$, perform L leapfrog steps with a stepsize of $\epsilon = \lambda\epsilon_0$, resulting in the state $(\hat{q}(\epsilon L), \hat{p}(\epsilon L)) = (q^*, p^*)$.

3) Regard (q^*, p^*) as a candidate for the next state, as in the Metropolis algorithm, accepting it with probability $A((q,p),\, (q^*,p^*)) = \min\big(1,\, \exp(-(H(q^*,p^*) - H(q,p)))\big)$, and otherwise letting the new state be the same as the old state.

The values used for ϵ_0 and for L may be chosen at random from some fixed distribution. Indeed, as discussed in Section 5.4, choosing at least one of them at random may sometimes be necessary in order to ensure that the chain is ergodic. Generally, it is good for the number of leapfrog steps, L, to be reasonably large, as this reduces the random walk aspect of the motion.

To prove that the hybrid Monte Carlo method leaves the canonical distribution over phase space invariant, we could proceed by showing that the distribution used to propose candidate states in the above procedure satisfies the the general symmetry condition required for the Metropolis algorithm to be valid. However, it seems more illuminating to show directly that detailed balance holds between any two small regions of phase space, R and R', where R' is the image of R under the mapping produced by L forward leapfrog steps, with stepsize $+\epsilon_0$. Due to time reversibility, R is the image of R' under the mapping produced by L backward leapfrog steps, with stepsize $-\epsilon_0$. Since the leapfrog steps conserve phase space volume, if the volume of R is δV, the volume of R' is δV as well. We assume that R is small enough that at all points within it the total energy can be considered to have the same value, $H(R)$, and similarly for R'.

The probability of a transition from R to R' occurring, when the starting point has the canonical distribution, is therefore

$$\frac{1}{Z_H}\exp\big(-H(R)\big)\,\delta V \cdot \frac{1}{2} \cdot \min\big(1,\, \exp\big(-(H(R') - H(R))\big)\big) \tag{5.26}$$

The first factor above is the probability of starting at a point in R. The second factor, $\frac{1}{2}$, is the probability of deciding to simulate the dynamics forward in time, leading to a point in R'. The third factor is the probability of accepting that move. Similarly, the probability of a transition from R' to R is

$$\frac{1}{Z_H}\exp\big(-H(R')\big)\,\delta V \cdot \frac{1}{2} \cdot \min\big(1,\, \exp\big(-(H(R) - H(R'))\big)\big) \tag{5.27}$$

The two probabilities are readily seen to be equal, showing that detailed balance holds. (Here, I have assumed that the image of R under the forward mapping does not overlap its image under the backward mapping, but taking this possibility into account would not alter the result.)

As pointed out by Duane, *et al* (5:1987), it is not necessary for the Hamiltonian used in simulating the dynamics to be the same as that which defines the canonical distribution from which we wish to sample. The algorithm remains valid as long as the correct Hamiltonian is used when deciding whether to accept a candidate state. Of course, generating candidate states by following trajectories based on a drastically wrong Hamiltonian will lead to a very low acceptance rate. One might, however, deliberately use a slightly "wrong" Hamiltonian in an attempt to improve the acceptance rate by compensating for the inexact simulation of the dynamics.

Variations on the hybrid Monte Carlo algorithm. Two variations on the basic hybrid Monte Carlo algorithm have recently been developed.

Horowitz (5:1991) introduced a modified form of the algorithm in which the momenta are not completely replaced in the stochastic transitions, but are instead perturbed only slightly, using the update of equation (5.23) with α close to one. This is coupled with the use of very short trajectories, perhaps consisting of only a single leapfrog step.

In the formulation of hybrid Monte Carlo I have given above, this procedure would be valid, but rather pointless — in my description, the direction in which a trajectory is simulated (i.e. the sign of the stepsize) is chosen at random each time, so keeping the old momenta largely intact will not prevent the algorithm from undertaking a random walk between trajectories (which, up to a point, is less desirable than persevering in the same direction). However, Horowitz modifies the dynamical transitions to operate as follows. First, simulate a dynamical trajectory with positive stepsize, then negate the momenta, then accept or reject the resulting state in the usual Metropolis fashion. Second, unconditionally negate the momenta. It is easy to see that both the first and second of these steps satisfy detailed balance, so the procedure is valid.

The purpose of the two (possible) negations of the momenta is to achieve the final effect that the momenta are negated only when a trajectory is rejected. The whole procedure would be pointless if it were combined with the usual stochastic transitions, in which the old momenta are completely forgotten anyway. When the momenta are only slightly perturbed, however, the modified procedure ensures that random walk behaviour occurs only as a consequence of rejections, which one may hope to keep to a low level.

This modified algorithm will probably not perform vastly better than the standard algorithm. When using trajectories with only a single leapfrog step, the rejection rate will be determined by the $O(\epsilon^3)$ local error of the leapfrog method, rather than by the $O(\epsilon^2)$ global error that applies to the standard algorithm (with long trajectories). However, to achieve an effect equivalent to a single long trajectory of the standard algorithm, the modified algorithm must proceed the same distance without a rejection occurring. This will require a number of steps proportional to $1/\epsilon$, so the probability of a rejection being avoided during the process will be $O(\epsilon^2)$, just as for the standard algorithm. However, this argument does not exclude the possibility that the modified algorithm might be a modest improvement over the standard algorithm in some circumstances. For a multivariate Gaussian, the modified method may be most appropriate when $\sigma_{\max}/\sigma_{\min}$ is not too large. This ratio controls the number of leapfrog iterations in a trajectory of optimal length; as it becomes larger, producing the effect of a long trajectory with the modified method will necessitate use of a smaller stepsize than for the standard method, in order to ensure that reversals in direction do not occur more frequently than is optimal.

I have developed a "windowed" form of the hybrid Monte Carlo algorithm (Neal, 5:1993). One motivation for this variation is the observation that the usual hybrid Monte Carlo algorithm appears somewhat wasteful, in that of the L new states visited along the trajectory, only the one state at the end has any chance of being chosen as the next state in the chain. If additional states along the trajectory could be considered as candidate states, one might expect that the probability of a new state being accepted would increase.

The windowed algorithm achieves this by employing an "accept window", consisting of the last W states in the trajectory, and a "reject window" consisting of the first W states, with $1 \leq W \leq L+1$. Whereas the standard algorithm decides whether to accept or reject a move to the single state at the end of the trajectory, based on the ratio of the probabilities of this new state and of the current state, the windowed algorithm decides instead whether to move to the accept window or to remain in the reject window, based on the ratio of the *sum* of the probabilities of *all* states in the accept window to the sum of the probabilities of all states in the reject window. Having in this way decided from which window the next state will come, a particular state from among those in that window is then chosen based on their relative probabilities.

In order to ensure that detailed balance still holds when this procedure is used, it is necessary that the current state be positioned at random within the reject window. This is arranged by performing from zero to $W-1$ of the L leapfrog steps in the opposite of the primary direction of the trajectory, starting from the current state. The remaining leapfrog steps are performed as usual, also starting from the current state.

The use of accept and reject windows reduces the effect of variations in H along the trajectory, and is seen empirically to increase the probability of acceptance for a given stepsize. As will be discussed in Section 6.3, it also becomes possible to utilize all the states in both windows when estimating the expectation of a function of the state. When $W \geq (L+1)/2$, all states along the trajectory will be in at least one of these windows, and all will then contribute to the estimate. On the other hand, use of windows requires that time be spent computing parts of the trajectory in the reject window that precede the current state, as well as parts of the trajectory in the accept window that turn out to extend past the state that is accepted. One must also evaluate the energy for all states in both windows, rather than for just the single state at the end of the trajectory.

Applications of the hybrid Monte Carlo method. The hybrid Monte Carlo algorithm was devised for use in simulations of quantum chromodynamics. The algorithm has a wide range of other potential applications, however, including to physical simulations of other sorts, and to problems of probabilistic reasoning and statistical inference. These include inference for belief networks in which the variables are continuous, and Bayesian learning of latent class models using the formulation of equation (2.46), in which the discrete variables are summed over. However, as yet, the only application to Bayesian inference that has been published is for multi-layer perceptrons, as described below.

For problems where the hybrid Monte Carlo algorithm is appropriate, use of the stochastic dynamics method will usually be reasonable as well. In comparing these options, the hybrid Monte Carlo method offers the assurance that the results are (asymptotically) unbiased. Using too large a stepsize with hybrid Monte Carlo produces the easily diagnosed problem of too high a rejection rate. With the stochastic dynamics method, the same mistake might introduce a degree of bias that is unacceptable, but which goes unnoticed.

As pointed out by Toussaint (1:1989), two other factors in choosing between the algorithms are the degree of accuracy required and the size of the system. As the required accuracy increases, the stochastic dynamics method must be run with an increasingly small stepsize in order to reduce bias. In contrast, once the rejection rate for the hybrid Monte Carlo method is acceptably low, there is no reason to reduce the stepsize any further, regardless of accuracy requirements. On the other hand, as the size of the system increases, the stepsize for the stochastic dynamics method can generally stay unchanged, but, as discussed further below, the stepsize for the hybrid Monte Carlo method will have to decrease with increasing system size in order to maintain a low rejection rate. The tradeoff between these factors will depend on the particular application.

Example: Multi-layer perceptrons. I have experimented with applying the hybrid Monte Carlo method to Bayesian learning for a multi-layer perceptron with the architecture of Figure 2.3 (Neal, 2:1992a, 2:1993a). As described for the stochastic dynamics method, the derivatives required to compute the dynamical transitions for the hybrid Monte Carlo method are readily obtained by backpropagation. The performance of the algorithm is fairly sensitive to the choice of stepsize, with the optimal stepsize being not much below a value at which the rejection rate becomes unacceptably high. Quite long trajectories (1000 leapfrog steps) were found to be desirable. A number of variations on the basic technique are described in the references, including use of re-parameterizations in which the prior is flat, and of simulated annealing. A clear picture of which techniques give the best results has not yet been attained, but it is clear that use of the hybrid Monte Carlo method for this application is feasible for problems of moderate size.

5.3 Other dynamical methods

Here, I will discuss other Markov chain methods that are "dynamical" in the sense that they make use of the derivative of the energy function (the "force").

The Langevin Monte Carlo method. In the hybrid Monte Carlo method, after picking new values for the momentum variables from the canonical distribution, one can choose to select a candidate state by following the dynamics for only a single leapfrog iteration (equations (5.11) to (5.13)), accepting or rejecting the resulting state based on the change in the total energy. This procedure is known as the *Langevin Monte Carlo* method (from the relationship to the "Langevin equation", described below). As is the case with the hybrid Monte Carlo method in general, it produces exactly unbiased results.

Though in a sense it is a special case of hybrid Monte Carlo, the Langevin Monte Carlo method has rather different performance characteristics. Consider the family of Langevin Monte Carlo methods obtained by varying the stepsize, ϵ, and the family of hybrid Monte Carlo methods obtained by varying ϵ while keeping the length of the trajectory fixed (varying the number of leapfrog steps to compensate for the change in ϵ). The error in the total energy that results when a single leapfrog step is used to find a candidate state in the Langevin Monte Carlo method will depend on the local error of the leapfrog method, which is $O(\epsilon^3)$ (see Section 5.1). In contrast, the error in the total energy in the hybrid Monte Carlo method with fixed trajectory length is determined by the global error, which is $O(\epsilon^2)$. Accordingly, for a given limit on the rejection rate, the Langevin Monte Carlo method can be used with a larger stepsize than can the hybrid Monte Carlo method. On the other hand, the Langevin Monte Carlo method has the disadvantage that, unlike the hybrid Monte Carlo method, it must explore the state space via a random walk. This difference is illustrated below in Section 5.4.

The *smart Monte Carlo* method of Rossky, Doll, and Friedman (5:1978) is equivalent to the Langevin Monte Carlo method, though it is expressed in somewhat different fashion.

The uncorrected Langevin method. We can consider omitting the acceptance test in the Langevin Monte Carlo method, simply accepting all the proposed moves. In this case, it is not necessary to compute the final values of the momentum variables at the end of the leapfrog step, since they will immediately be replaced by new values chosen from the canonical distribution at the start of the next iteration. In fact, there is no reason to explicitly represent the momenta at all. The effects on the position coordinates of picking new momenta and then applying one leapfrog step can simply be expressed by the following form of the *Langevin equation*:

$$q'_i \;=\; q_i \;-\; \frac{\epsilon^2}{2}\frac{\partial E}{\partial q_i}(q) \;+\; \epsilon n_i \tag{5.28}$$

where the n_i are independently chosen from a Gaussian distribution of mean zero and variance one.

From the Langevin equation, it is clear that moving a given distance under the influence of the "force", $-\partial E/\partial q$, will require a number of iterations proportional to ϵ^{-2}. Moving a given distance as a result of the random term in the equation (e.g. when exploring a region of constant energy) will require a number of iterations that is also of that order, as in N iterations, the distance travelled by the random walk will be only about $\epsilon\sqrt{N}$, not ϵN, due to the continual randomization of direction.

Use of the Langevin equation without an acceptance test may appear to be a special case of the uncorrected stochastic dynamics method, but this is not really so. Stochastic dynamics is justified on the basis that it produces the correct distribution in the limit as the stepsize approaches zero, *while the length of the trajectories remains fixed.* In the uncorrected Langevin method, the length of the trajectory is equal to the stepsize, so the justification does not apply. Indeed, for some discretizations other than the leapfrog method, that are valid for stochastic dynamics, the corresponding "Langevin" method is invalid.[7]

However, one can see that simulating a Markov chain using the transitions of equation (5.28) results in a Boltzmann distribution for the q_i, in the limit of small ϵ, by considering the effect that reducing ϵ would have in an application of the corresponding Langevin Monte Carlo method. As seen above, for a Langevin simulation to move a given distance in state space, the number of iterations will have to grow as ϵ^{-2} as the stepsize decreases, while the rejection rate for the Langevin Monte Carlo method will decrease as ϵ^3. The total number of rejections expected as the Langevin Monte Carlo method explores the entire state space will therefore be proportional to ϵ. For a small enough ϵ, the probability of *any* moves being rejected will be negligible, and hence omitting the rejection test will be unlikely to affect the results.

Other approaches. The *force bias* method of Rao, Pangali, and Berne (5:1979) is similar to the Langevin Monte Carlo method, in that it uses the derivative of the energy at the current point (the "force") to generate a candidate state for the Metropolis algorithm. Unlike the Langevin Monte Carlo method, however, the state is not found by a procedure involving momentum variables. Instead a candidate state is chosen from within some region around the current state, with a bias toward states that lie in the direction of the force, as evaluated at the current state. This state is then accepted or rejected using the acceptance function of equation (4.33), which accounts for the fact that such a proposal distribution is not necessarily symmetrical. The behaviour of this method should be similar to that of the Langevin Monte Carlo method.

[7] Consider the discretization $\hat{q}_i(\tau+\epsilon) \;=\; \hat{q}_i(\tau) + \epsilon\hat{p}(\tau)$, $\hat{p}_i(\tau+\epsilon) \;=\; \hat{p}_i(\tau) - \epsilon(\partial E/\partial q_i)(\hat{q}_i(\tau))$, for which the corresponding "Langevin" method is $q'_i \;=\; q_i + \epsilon n_i$ — i.e. we simply add noise to the q_i, ignoring the energy function entirely!

Nosé dynamics (Nosé, 5:1984) is a method for sampling from the canonical distribution without introducing a stochastic element. In this approach, the system of interest is extended with additional position and momentum variables. A Hamiltonian for this extended system is then defined, and the system simulated using some discretization of Hamiltonian dynamics. Assuming ergodicity, this procedure will sample from (close to) the microcanonical distribution for the extended system. The extended system is designed so that the marginal distribution of the original variables with respect to this microcanonical distribution is the original canonical distribution, from which we wished to sample.

5.4 Analysis of the hybrid Monte Carlo algorithm

Work on analysing the performance of the hybrid Monte Carlo algorithm has concentrated on explicating its behaviour when sampling from a Gaussian distribution. Typically, the hybrid Monte Carlo algorithm is applied to more complex problems, but performance on Gaussian distributions may be indicative of general performance, at least in the negative sense that a method slow to sample from a Gaussian distribution will be slow to explore locally Gaussian regions of a more complex distribution.

The hybrid Monte Carlo algorithm for a univariate Gaussian. A univariate Gaussian distribution with mean zero and variance σ^2 can be represented as the marginal distribution for a "position" variable Q with respect to the canonical distribution given by the following Hamiltonian:

$$H(q,p) \;=\; \frac{1}{2\sigma^2}\,q^2 \;+\; \frac{1}{2}\,p^2 \tag{5.29}$$

A single iteration of the leapfrog method (equations (5.11) to (5.13)) applied with this Hamiltonian produces a linear mapping from the point in phase space at time τ to the point at time $\tau+\epsilon$, which can be written in matrix form as follows:

$$\begin{bmatrix} q(\tau+\epsilon) \\ p(\tau+\epsilon) \end{bmatrix} \;=\; \begin{bmatrix} 1-\epsilon^2/2\sigma^2 & \epsilon \\ -(\epsilon/\sigma^2)(1-\epsilon^2/4\sigma^2) & 1-\epsilon^2/2\sigma^2 \end{bmatrix} \begin{bmatrix} q(\tau) \\ p(\tau) \end{bmatrix} \tag{5.30}$$

When $\epsilon > 2\sigma$, repeated application of this transformation results in rapid divergence of q, p, and H. When $\epsilon < 2\sigma$, however, the leapfrog method is stable, and the transformation matrix above can be written as $M = S^{-1}RS$, where

$$S = \begin{bmatrix} \sigma^{-1}\sqrt{1-\epsilon^2/4\sigma^2} & 0 \\ 0 & 1 \end{bmatrix}, \quad R = \begin{bmatrix} 1-\epsilon^2/2\sigma^2 & (\epsilon/\sigma)\sqrt{1-\epsilon^2/4\sigma^2} \\ -(\epsilon/\sigma)\sqrt{1-\epsilon^2/4\sigma^2} & 1-\epsilon^2/2\sigma^2 \end{bmatrix} \tag{5.31}$$

The matrix S represents a rescaling of q; the matrix R can be recognized as a rotation through an angle of $\arccos(1-\epsilon^2/2\sigma^2)$. The effect of each leapfrog iteration in this case is therefore to move some distance around an ellipse in phase space. This is illustrated in Figure 5.2.

Several conclusions may be drawn from this analysis. First, when the hybrid Monte Carlo method is used with long trajectories, consisting of many leapfrog iterations, it is essential that the stepsize be small enough to ensure stability, as otherwise the value of H will rapidly diverge, and the probability of acceptance will be extremely low. Second, as long as the stepsize is in the stable region, the error in H, at least for this example, does *not* grow larger and larger as the number of leapfrog iterations is increased. Instead, H oscillates as the current point moves around the ellipse, as seen in Figure 5.2. Of course, in this univariate example, simulating a long trajectory that goes around this ellipse many times would simply

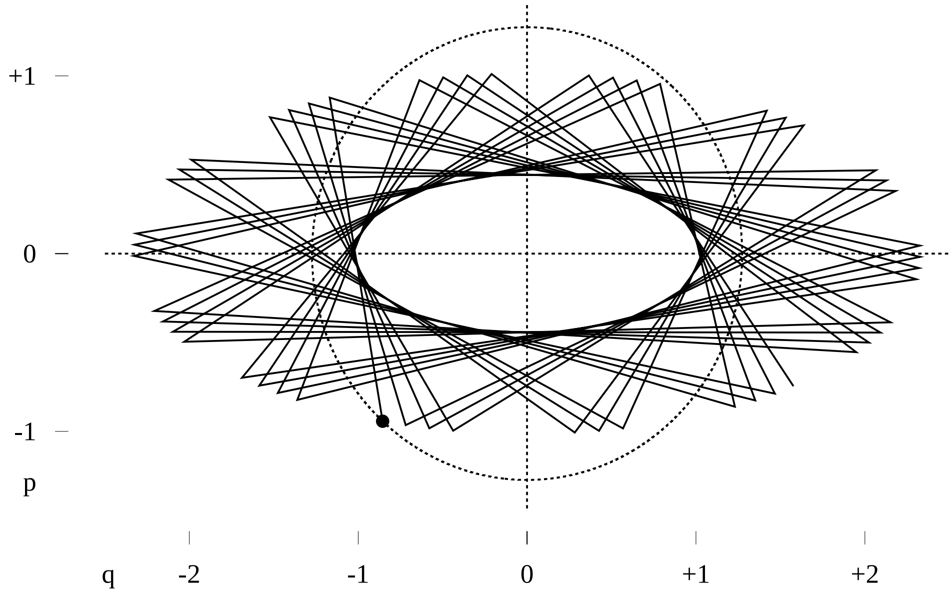


Figure 5.2: Phase space depiction of the operation of the leapfrog method applied to the Hamiltonian of equation (5.29), with $\sigma = 1$. Fifty leapfrog steps with $\epsilon = 1.8$ are shown, starting at the point marked in the lower left quadrant. The circle on which the true trajectory lies is shown as well. Note that the value of H at any point in this diagram is simply half the square of the distance from the origin.

be a waste of time, but we will see below that long trajectories are useful in multivariate problems.

Mackenzie (5:1989) has pointed out that the hybrid Monte Carlo method can fail to be ergodic for problems of this sort. This will occur if the stepsize and the number of leapfrog iterations used are such that the trajectory returns exactly to its starting point. Note that whether this happens does not depend on the value randomly chosen for the momentum before each trajectory. If the trajectory returns close to its starting point, but not exactly so, the method will be ergodic, but may take a very long time to reach equilibrium. These phenomena do not occur with the Langevin Monte Carlo method, in which each trajectory consists of only a single leapfrog iteration. Indeed, it is easy to see that the randomization of the momentum in the Langevin Monte Carlo method is sufficient to guarantee ergodicity for any Hamiltonian.

As Mackenzie notes, this potential problem with the hybrid Monte Carlo method can easily be avoided by chosing the stepsize at random from from within some small range.

The hybrid Monte Carlo algorithm for a multivariate Gaussian. The system studied above, with the Hamiltonian of equation (5.29), can be generalized to a system with several position variables (and the same number of momentum variables), with the following Hamiltonian:

$$H(q,p) \quad = \quad \sum_i \frac{1}{2\sigma_i^2}\, q_i^2 \; + \; \sum_i \frac{1}{2}\, p_i^2 \tag{5.32}$$

In the canonical distribution this defines, the marginal distribution for Q is a multivariate

Gaussian in which the components are independent, with component Q_i having mean zero and standard deviation σ_i. Since the operation of the hybrid Monte Carlo algorithm is invariant with respect to translations and rotations of the coordinate system, its behaviour when applied to the above system will also tell us its behaviour when it is used to sample from any multivariate Gaussian distribution, in which the means may not be zero, and the components may be correlated.

When the leapfrog method is used with the Hamiltonian of equation (5.32), there is no interaction between the different pairs of position and momentum variables. Each such pair, (q_i, p_i), evolves independently of the other pairs, since the value of $\partial E/\partial q_i$ does not depend on q_j for $j \neq i$. Once the end of the trajectory is reached, however, a decision to accept or reject the final state is made based on the total error in H, which is the sum of the errors due to the inaccuracies in the simulation with respect to each pair of variables. A large error resulting from inaccurate simulation with respect to *any* of these coordinate pairs will likely lead to the trajectory being rejected.

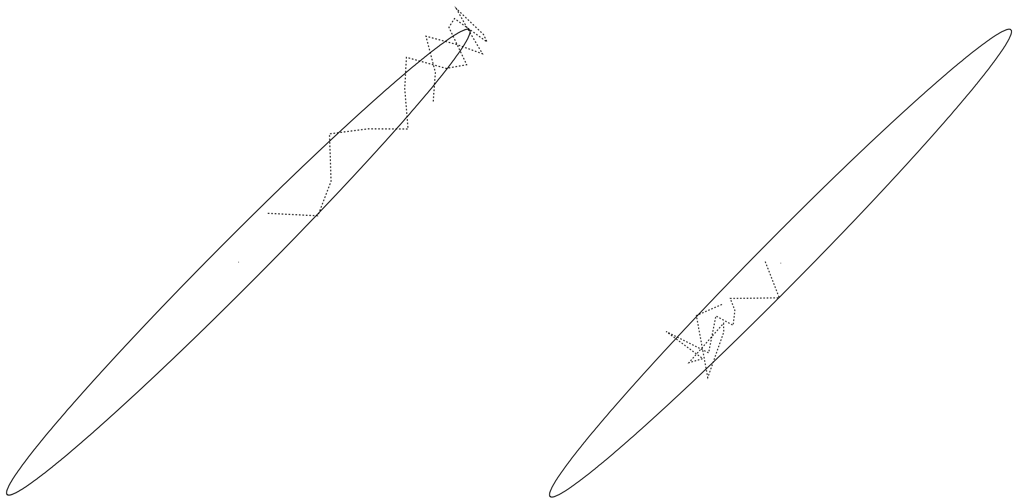
The total error in H is dominated by the error relating to those q_i for which σ_i small. Certainly, if each trajectory consists of many leapfrog iterations, we must be sure to use a stepsize less than $2\sigma_{\min}$, where $\sigma_{\min} = \min \sigma_i$, in order to ensure stability; the optimal stepsize will be somewhat smaller than this. The same advise applies when the hybrid Monte Carlo algorithm is used to sample from a general multivariate Gaussian distribution, except that $\sigma_{\min}$ is in this case the smallest standard deviation in any direction, which in general is less than the standard deviation along any of the coordinate axes.

The need to limit the stepsize to approximately $\sigma_{\min}$ slows the rate at which the algorithm can explore the distribution in other directions. For the least confined direction, where the standard deviation is $\sigma_{\max}$, roughly $\sigma_{\max}/\sigma_{\min}$ leapfrog iterations will be needed to move across the region of high probability. This ratio is thus a measure of how difficult the problem is.

Figure 5.3 illustrates the operation of the hybrid Monte Carlo algorithm on a bivariate Gaussian distribution. In this example, $\sigma_{\min} = 0.10$ and $\sigma_{\max} = 1.41$. The stepsize used was $\epsilon = 0.15$, which gives a reasonable acceptance rate. In Figure 5.3(a), a single trajectory consisting of twenty leapfrog iterations is shown. Note that the path taken does *not* resemble a random walk. Instead, the trajectory proceeds one way along the less confined direction, until it is "reflected" at one end. (The point where this reflection occurs depends on the magnitude of the momentum in that direction, which was chosen at random before the start of the trajectory.)

Figure 5.3(b) shows twenty successive trajectories that each consist of a single leapfrog iteration, as are used in the Langevin Monte Carlo method. Between each leapfrog step, new values for the momentum variables are chosen at random. Consequently, the less confined direction is in this case explored via a random walk.

This illustrates the crucial role that long trajectories play in the hybrid Monte Carlo method. A trajectory consisting of around $\sigma_{\max}/\sigma_{\min}$ leapfrog iterations, with a stepsize around $\sigma_{\min}$, will produce a candidate state for the hybrid Monte Carlo method which is almost independent of the start state, and which has a good chance of being accepted. In contrast, to reach an independent state via a random walk using the Langevin Monte Carlo method, in which trajectories consist of a single leapfrog iteration, will require approximately $(\sigma_{\max}/\sigma_{\min})^2$ iterations. For difficult problems, where $\sigma_{\max}/\sigma_{\min}$ is large, this difference will be more important than the ability of the Langevin Monte Carlo method to use a somewhat larger



(a) (b)

Figure 5.3: Sampling from a bivariate Gaussian distribution, (a) using the hybrid Monte Carlo method with a trajectory consisting of twenty leapfrog iterations, (b) using the Langevin Monte Carlo method (i.e. hybrid Monte Carlo with a trajectory consisting of just one leapfrog iteration). One trajectory is shown in (a), twenty in (b) (except some rejected trajectories are not shown), representing the same number of function evaluations. In both cases, the leapfrog method was used with a stepsize of 0.15. The distribution sampled from is the same as that of Figure 4.1. Note that only the course of the position variables is depicted; the momentum variables are not shown.

stepsize (resulting from the difference between the local and the global discretization errors of the leapfrog method).

Scaling with system size. The manner in which the performance of the Langevin Monte Carlo and hybrid Monte Carlo methods scales with system size has been investigated using general arguments by several authors (e.g. Creutz, 5:1988), and analysed in detail for the multivariate Gaussian by Kennedy and Pendleton (5:1991). As in Section 4.4, we are here concerned with performance on a system consisting of N independent replicas of a sub-system.

The conclusion of these studies is that for the Langevin Monte Carlo algorithm — that is, for hybrid Monte Carlo used with trajectories consisting of a single leapfrog iteration — the computational effort required to move from one state at equilibrium to an approximately independent state is proportional to $N^{4/3}$. This is accomplished using a stepsize proportional to $N^{-1/6}$. When the Hybrid Monte Carlo algorithm is instead used with trajectories long enough that the end point is approximately independent of the start point, the scaling behaviour improves to $N^{5/4}$, with the stepsize used in this case being proportional to $N^{-1/4}$.

In contrast, the effort required with uncorrected stochastic dynamics simply grows as N, though when the sub-systems are not truly independent, one might sometimes wonder whether the bias grows as well. The scaling of the hybrid Monte Carlo method can be made closer to linear by using discretization schemes of higher order than the leapfrog method, but since the associated constant factor may be larger, this will not necessarily be advantageous in practice.

6. Extensions and Refinements

This section covers several topics that apply generally to all or most of the methods described in the preceding sections. I discuss ways of reaching equilibrium more rapidly using simulated annealing, of estimating free energy differences using a series of simulations, of assessing and reducing the error in Monte Carlo estimates, and of exploiting parallel hardware.

6.1 Simulated annealing

Markov chain sampling methods of all types may fail to converge to their equilibrium distribution within a reasonable time if movement through state space is inhibited by regions of high energy (low probability). The simulation may instead remain in a region of relatively high energy and/or small volume, because movement to a region of lower energy and/or larger volume can take place only via passage through states of very high energy, an event that is unlikely to occur in a simulation run of limited length.

This problem can sometimes be overcome using *simulated annealing*. This method was introduced for optimization problems by Kirkpatrick, Gelatt, and Vecchi (6:1983) and by Černý (6:1985). In such applications, the goal is to sample from the canonical distribution for a system at a temperature of zero, in which the probability is concentrated on the state or states of minimal energy. For applications to probabilistic inference, we wish instead to sample from a canonical distribution that reproduces some distribution of interest, usually defined using an energy function that gives the desired distribution at $T = 1$. Annealing has occasionally been used to more easily reach equilibrium at a non-zero temperature in physical simulations, and its use for this purpose is standard practice with the "Boltzmann machine" of Ackley, Hinton, and Sejnowski (2:1985). However, most of the literature on annealing assumes that the goal is to find a minimum energy state by cooling to a temperature of zero. Results pertaining to such optimization applications do not always apply when the final temperature is non-zero.

The concept of annealing. Simulated annealing is inspired by an analogy with metallurgy, in which slow cooling (annealing) is used to produce metal that is tougher than that which results from rapid cooling (quenching). When a Markov chain simulation is used to sample from the canonical distribution given some energy function (equation (2.54)), the analogous procedure is to gradually reduce the temperature, T, from a high initial value to the value at which we wish to sample. The high initial temperature reduces the effect of energy barriers, allowing free movement through state space. The hope is that as the temperature is then reduced, the simulation will end up on the right side of the energy barriers, that is, on the side where it is more likely to be under the canonical distribution at the final temperature.

Annealing cannot perform miracles, however. When applied to complex optimization problems, it is generally too much to expect that a practical annealing procedure will converge to precisely the global energy minimum — the hope is merely that a state of close to minimal energy will be found. Similarly, in cases where reaching equilibrium at some non-zero temperature is difficult, one can hope that annealing will increase the chances of reaching a region that is typical of the equilibrium distribution, but one generally cannot guarantee this.

These hopes are based on the idea that the distribution at higher temperatures is a good guide to the distribution at lower temperatures. This may often be the case, but it will not

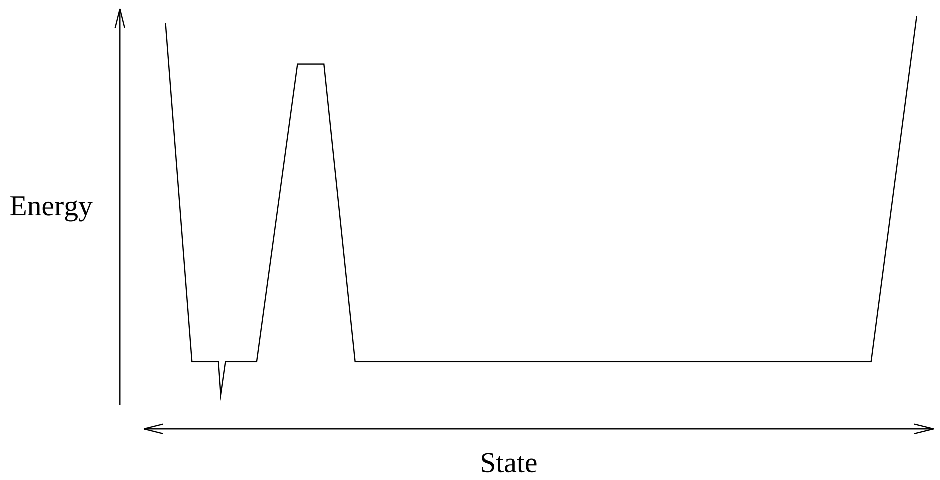


Figure 6.1: A problem for which annealing does not work well. The state here is one-dimensional, with the energy of a state as shown. At low temperatures an energy barrier confines the simulation for long periods of time to either the small region on the left or the larger region on the right. The small region contains the global minimum, but annealing will likely direct the simulation to the larger region, since the tiny region in the vicinity of the global minimum has negligible influence on the distribution at the high temperatures where the energy barrier can easily be traversed.

always be. Figure 6.1 shows a problem where this assumption fails. The distribution at high temperatures is here dominated by the *size* of the two low-energy regions, not by the depths of their lowest points. Since the global minimum happens to be in the smaller region, annealing will in this case direct the simulation *away* from the region where the equilibrium distribution will be concentrated at very low temperatures.

Such examples are cause for caution in evaluating the prospects of success using annealing, but they should not be taken as discouraging its general use, as no realistic procedure can guarantee good results in all cases.

Note that annealing can be used with all Markov Chain sampling procedures, even those, such as Gibbs sampling, that are not normally expressed in terms of energy. To sample at a temperature of T, one simply alters the probabilities (or probability densities) of all states by raising them to the power $1/T$. (They must then be re-normalized to again sum or integrate to one.)

Annealing schedules. The success of simulated annealing depends crucially on the schedule of temperatures used in the simulation. The temperature used in generating the state at iteration t will be denoted T_t. Discussion is usually confined to schedules in which the temperature is non-increasing, i.e. for which $T_{t+1} \leq T_t$.

The initial temperature, T_1, is generally chosen to be high enough that the canonical distribution is close to uniform. When the Metropolis algorithm is used with a fixed distribution for candidate moves, a related criterion is that the temperature be high enough that the rejection rate is very low.

Many people use schedules in which the temperature stays fixed for a number of iterations, in the apparent belief that it is necessary, or at least desirable, to approach equilibrium at the present temperature before lowering it further. In my view, this reflects an incorrect intuition. Instead, one should usually adopt the strategy of decreasing the temperature slightly after every iteration. With such smaller temperature reductions, the degree to which the system is no longer in equilibrium after a reduction will be lessened, which should at least compensate for the fact that only one iteration is done at each temperature. Furthermore, the basic assumption behind the use of annealing is that the distribution at a higher temperature is a good guide to the distribution at a lower temperature. This assumption is more likely to be true for small changes in temperature than for large changes.

A number of pre-determined annealing schedules have been proposed, in which the schedule of temperatures is fixed before the simulation is begun. Geman and Geman (4:1984) analyse the following logarithmic schedule:

$$T_t \;=\; T_1/\log(t), \qquad \text{for } t > 1 \tag{6.1}$$

They prove that, under certain conditions, there is a value for T_1 such that use of this schedule guarantees that the distribution will eventually be concentrated on the set of points where the energy reaches its global minimum.

It is instructive to see informally why this should be so. Consider the case where the system is simulated using the Metropolis algorithm and the state is decomposed into n components, with each iteration of the algorithm visiting these components in turn, selecting a candidate change to the component from a uniform distribution. Suppose that the maximum difference in energy between states is ΔE. Then the probability of moving to any state in iteration t when using the schedule of equation (6.1) will be at least

$$\prod_{k=1}^{n} \frac{1}{m_k} \exp(-\Delta E/T_t) \;=\; t^{-n\Delta E/T_1}/M \tag{6.2}$$

where m_k is the number of possible values for component k, and $M = \prod_k m_k$. The total number of times any particular state is visited as the number of iterations goes to infinity will therefore be at least $\sum_t t^{-n\Delta E/T_1}/M$, which is infinite if T_1 is chosen to be at least $n\Delta E$. Accordingly, with such a choice for T_1 it is not possible for the system to be trapped forever in a bad local minimum. As the temperature approaches zero, moreover, the system will spend an increasing fraction of its time at the global minimum (or minima). Hajeck (6:1988) gives a similar result with a better constant. The argument can also be used to show asymptotic convergence with a logarithmic schedule for a variety of other simulation schemes.

It should be clear from this explanation that these results are of no practical interest. Essentially, they say that simulated annealing with a logarithmic annealing schedule retains enough traces of exhaustive search to guarantee asymptotic convergence, but if exhaustive search were a realistic option, we would not be using simulated annealing anyway. In optimization applications, these logarithmic schedules are often rejected on the grounds that they require too many iterations to reach a temperature near zero. Even if this were not so, however, asymptotic convergence provides no reason to think that the simulation will be near the global minimum after some reasonable finite number of iterations, even if the temperature has at that time declined to an appropriate level. The irrelevance of these results is even more clear for applications where the goal is not to find the global minimum, but rather to sample from the equilibrium distribution at some non-zero temperature

It is more common in practice to use a geometric annealing schedule, of the form

$$T_{t+1} \;=\; \alpha T_t \tag{6.3}$$

with $0 < \alpha < 1$. T_1 and α are generally set by intuition or trial and error.

Adaptive annealing schedules base the temperature for the next iteration on characteristics of the system observed in earlier iterations. The aim is to spend more time at what turn out to be the critical temperatures, when the fundamental features of the low-temperature state are being established. In physical systems, these are the temperatures where phase transitions occur.

One heuristic with this motivation is to aim for a constant rate of entropy reduction. This method was apparently first used by Otten and van Ginneken (6:1984). Recall that the entropy of a distribution is $S = \langle -\log P(s)\rangle$. For a canonical distribution (equation (2.54)), the entropy will be a function of temperature, and one can show that $dS/dT = C/T$, where the *heat capacity*, C, is defined as $C = d\langle E\rangle/dT$. One can also show that $C = (\langle E^2\rangle - \langle E\rangle^2)\,/\,T^2$. An approximately constant rate of entropy reduction can thus be achieved using an annealing schedule of the following form:

$$T_t - T_{t+1} \quad \propto \quad T\,/\,C \quad = \quad T_t^3\,/\,\big(\langle E^2\rangle_t \, - \, \langle E\rangle_t^2\big) \tag{6.4}$$

where $\langle\,\cdot\,\rangle_t$ denotes expectation with respect to the canonical distribution at temperature T_t. The need to estimate these expectations might provide a reason to keep the system at a fixed temperature for a number of iterations (despite the arguments to the contrary given above).

Another heuristic has been proposed by Salamon, Nulton, Harland, Pedersen, Ruppeiner, and Liao (6:1988). They argue in favour of using an annealing schedule that results in "constant thermodynamic speed" — one in which the system stays the same distance from true equilibrium at every stage. They interpret distance from equilibrium in terms of the degree to which the mean energy departs from its mean in the equilibrium distribution (though it is quite possible for a system to have reached the equilibrium energy distribution but still be far from equilibrium in other respects). On this basis, they find that the annealing schedule should be of the following form:

$$T_t - T_{t+1} \quad \propto \quad T\,/\,\big(\tau\sqrt{C}\big) \tag{6.5}$$

where C is as above, and τ is a time constant describing how rapidly the equilibrium energy is approached, which can also be estimated from the data. From their empirical tests, Salamon, *et al*, conclude that annealing schedules incorporating a factor of τ^{-1}, as above, generally perform better than those without this factor. Unfortunately, the schedule of equation (6.4) was not included in their comparison, though they mention it in their discussion, so the relative merits of these two heuristics are unclear.

Annealing with the Metropolis algorithm. When annealing a system of real-valued variables simulated using the Metropolis algorithm, it seems reasonable that as the temperature is reduced, the width of the proposal distribution used to select candidate states should be reduced as well, since at low temperatures the increase in energy that can be tolerated while keeping the rejection rate acceptably low is smaller than at high temperatures.

To find a reasonable approach to scaling with temperature, let us assume that most moves are rejected because in one or more directions the current state is near the bottom of a locally-quadratic bowl in the energy function. Large changes to the state in these directions will move the state up the sides of such a bowl, increasing the energy, and likely leading to rejection. For example, suppose there is just one component to the state, and the energy is given by $E(x) = x^2/2$. The canonical distribution at temperature T with this energy function is $P(x) \propto \exp(-x^2/2T)$, which is a Gaussian with standard deviation $T^{1/2}$. Once the state distribution has approached equilibrium, we are therefore likely to be about a distance of $T^{1/2}$ from the center of this bowl. From this starting point, changes of a magnitude around $T^{1/2}$ have a good chance of moving to a lower energy state, and of therefore being accepted. Changes of more than a few times this magnitude are likely to greatly increase the energy, and therefore be rejected.

Accordingly, it makes sense to scale the magnitude of the candidate changes by $T^{1/2}$, if one wishes to keep the acceptance rate approximately constant. As discussed in Section 4.4, maintaining a reasonably high acceptance rate is expected to be good policy in high-dimensional problems, though not always in lower dimensional problems.

The candidate moves for the Metropolis algorithm might change only a single component at a time, or they might change all components at once. In either case, a Gaussian distribution centred at the current state with a standard deviation proportional to $T^{1/2}$ is an obvious choice for the proposal distribution that scales in the fashion recommended above. For an optimization application, one can show that with this distribution the logarithmic annealing schedule of equation (6.1) is guaranteed to eventually settle to the set of global minima.

Szu and Hartley (6:1987) advocate a "fast" simulated annealing method, in candidate states differ from the current state in all components, with the offset, Δx, from the current state drawn from a multi-dimensional Cauchy distribution. This distribution has the following density:

$$P(\Delta x) \;\; \propto \;\; W^{-n}\big(1 + (|\Delta x|/W)^2\big)^{-(n+1)/2} \tag{6.6}$$

where W is a width parameter, and n is the dimensionality of the state space. Efficient methods for generating values from this distribution are known.[8] Szu and Hartley show that with this choice for the candidate selection distribution, asymptotic convergence to the set of global minima is guaranteed if the width of the distribution at iteration t is decreased according to the following schedule (while the temperature also goes to zero):

$$W_t \;\; = \;\; W_1 \,/\, t \tag{6.7}$$

This result comes about because the tails of the Cauchy distribution drop off polynomially with distance from the origin, rather than exponentially, as for the Gaussian distribution. This introduces a greater element of exhaustive search into the algorithm.

Ingber (6:1989) advocates a "very fast" simulated annealing method, in which changes to the components are drawn independently (but then applied together), from a distribution whose tails drop off as $|\Delta x_i|^{-1}$. (For this distribution to be proper, the range of the state must be confined to a bounded interval.) He shows that asymptotic convergence is guaranteed if the width of these distributions decreases according to the schedule

$$W_t \;\; = \;\; K \exp(-ct^{1/n}) \tag{6.8}$$

where K and c are positive constants and n is the dimensionality of the state space.

It appears that both Szu and Hartley and Ingber envision that the temperature used in deciding whether to accept a move may decrease according to a schedule that is not tied to the above schedules for reducing the width parameter, W_t. This contrasts with the argument I give above for setting $W_t \propto T_t^{1/2}$. At least in part, this difference may be due to the focus

[8] One method is to generate independent Gaussian variables $Y_1, \ldots, Y_n$, with zero mean and standard deviation W, and another independent Gaussian variable, S, with zero mean and unit variance, and then let $X_i = Y_i/S$. See also (Devroye, 9:1986, Chapter 11, Section 1.1).

of these authors on optimization problems.[9]

It is important to realize that the terms "fast" and "very fast" used to describe these methods refer properly only to the schedule by which the width of the proposal distribution (and perhaps the temperature) is reduced. There is no guarantee that these methods actually converge to the global minimum or its vicinity any faster than other methods. As discussed above, results on asymptotic convergence are of no practical interest. Accordingly, even when the associated proposal distributions are used, such results do not constitute a good justification for using the "fast" and "very fast" schedules in preference to others. Nevertheless, the idea of using a heavy-tailed proposal distribution seems reasonable, if it is done in such a way as to sacrifice only a modest constant factor in local exploration in return for the possibility of a substantial gain from large moves.

In this respect, one should note that there is a significant difference between the "fast" and "very fast" algorithms with respect to the dependency between the changes proposed to different components of the state. With the multi-dimensional Cauchy distribution used in the "fast" algorithm, the changes to different components are not independent — a large change in one component is usually accompanied by large changes in other components, likewise, when one component changes only slightly, other components also are likely to change only slightly. This property ensures that there is a significant probability of considering changes that are entirely local (i.e. that involve small changes in all components), while at the same time permitting occasional large changes to the entire state. In contrast, the "very fast" algorithm selects changes independently for each component. This produces a significantly different effect in high-dimensional problems. If the selection distribution for each component gives significant probability to a large change, then every move will be very likely to involve a large change to at least one component, effectively eliminating any element of local search from the algorithm. If the probability of making a large change to a component is reduced to avoid this, then almost all the large changes considered will affect only one or a few components. Which of these methods is better presumably depends on the problem.

Annealing with dynamical methods. Annealing can be implemented in conjunction with dynamical methods of simulation by explicitly scaling the potential energy function, or by explicitly varying the distribution of the momenta used in the stochastic transitions. However, annealing can also very naturally be added to the stochastic dynamics method in an implicit way if the stochastic transitions of equation (5.23) are being used. All that is required is to use a value for α that is close to one, and to choose initial values for each of the momentum variables from a Gaussian distribution of mean zero and variance T_0, where T_0 is, roughly, the desired initial temperature.

If the momentum variables did not interact with the position variables, the variance of the momentum variables would gradually decline until it reached a value of one, corresponding to a temperature of one, as this is the equilibrium distribution for the transitions of equation (5.23). The momentum does interact with the position variables, of course, and its effect is to impose a temperature on the rest of the system equal to its present variance. The gradual "cooling" of the momentum variables leads to the entire system being annealed, though not necessarily at the same rate as if the momentum variables were isolated, as the

[9]Note that these authors use the word "temperature" to refer both to the width of the proposal distribution, and to the parameter controlling move acceptance probability (with occasional disambiguating qualifications). In particular, the schedules of equations (6.7) and (6.8) are referred to as "annealing" schedules, and W as a "temperature" parameter. I feel this usage has the potential to introduce confusion into the literature, and should therefore be avoided.

interactions allow potential energy to be converted into kinetic energy. From analogies with statistical physics, one can see that the annealing schedule implicitly being pursued is that of constant entropy reduction (until such time as the temperature approaches $T = 1$).

A system initialized to a temperature of T_0 in this way will often get *hotter* initially, due to the initial state of the position variables having a higher potential energy than the equilibrium value at T_0. It is also possible that the temperature might instead drop rapidly in the first few iterations, if the initial state of the position variables had atypically low energy. To avoid such possibilities, one can re-initialize the momentum variables after a few preliminary iterations, thereby re-establishing the desired initial temperature.

This scheme can be used with the hybrid Monte Carlo method as well, but an additional elaboration is needed in order for the current temperature to have the proper effect on the probability of accepting a trajectory. Details may be found in (Neal, 4:1992a).

Applying annealing to Bayesian inference. In Bayesian inference problems, the energy function will contain some terms representing the prior and other terms representing the likelihood. Straightforward application of annealing will lead to the simulation initially exploring regions of parameters space with little regard to either the prior or the likelihood, as the effects of terms of both kinds will be scaled down by the high temperature. This seems undesirable, since if the prior truly represents prior beliefs about where the parameter values are likely to lie, it should be used to guide the search even in its initial stages. (An exception would arise if the prior itself introduced barriers to free movement through the parameter space, but this would be unusual.)

Exempting the prior from the effect of temperature is easily accomplished when annealing is done explicitly — just use the modified canonical distribution

$$P(s) \;=\; \frac{1}{Z}\exp\bigl(-E_P(s) - E_L(s)/T\bigr) \tag{6.9}$$

where $E_P(s)$ is the portion of the energy derived from the prior, and $E_L(s)$ is the portion derived from the likelihood.

When annealing is done implicitly using a dynamical method, as described above, it is not so clear how to prevent annealing from affecting the prior. However, an equivalent effect is achieved in (Neal, 2:1992a) by transforming to a parameterization in which the prior is flat. This is always possible, though not always convenient. A flat prior contributes nothing to the energy, and hence cannot be affected by the current temperature.

Other techniques related to annealing. Several techniques for speeding the approach to equilibrium have recently been developed that, like simulated annealing, involve the system being simulated at several temperatures. Unlike annealing, however, these methods do not impose a monotonically decreasing schedule of temperatures. Instead, they either allow the system to move between temperatures in a random walk, or simultaneously perform simulations at a range of temperatures.

In the *simulated tempering* method of Marinari and Parisi (6:1992), a variable is added to the state whose values index a set of possible temperatures, including the one of interest and others that are higher. This variable is updated in Metropolis fashion during the simulation, with changes accepted or rejected according to an "energy" function that includes the scaling effect of the temperature, and also contains an artificial weighting term, intended to approximately equalize the time spent at each temperature. (These weights might be set on the basis of preliminary simulations.) The temperature thus moves in a random walk, occasionally taking on the value for which samples are desired, while at other times moving to higher values at which energy barriers can be surmounted.

Rather than explicitly represent the temperature index in the simulated tempering algorithm, one could instead represent only the usual system state, but perform the simulation using the marginal probabilities of states found by summing the simulated tempering probabilities for all the possible temperature indexes. The result would be essentially the same as an algorithm of Berg and Celik (6:1992). Considered more directly, their method samples from a *multicanonical* distribution, in which states of equal energy have equal probability, and the relative probability of states with different energies is adjusted so as to produce an approximately uniform distribution of energies over some range. (As with simulated tempering, suitable factors for this adjustment could be found from preliminary simulations.) When this system is simulated, it performs a random walk over states with various energies, avoiding being trapped in local low-energy minima. By regarding the multicanonical distribution as an importance sampler, estimates of expectations with respect to some canonical distribution of interest may be obtained by applying equation (3.5).

Wang and Swedsen (6:1988) investigate a different approach to combining high-temperature and low-temperature simulations. In their *replica Monte Carlo method*, a number of copies of an Ising system at various temperatures are simultaneously simulated, using both standard Metropolis moves that affect only one of the replicas, and additional moves that transfer information between replicas at nearby temperatures. These latter moves allow states found in the free exploration done in the high temperature replicas to move down to the low temperature replicas. The method Wang and Swendsen describe is specific to the Ising model, but similar ideas may well be applicable to other systems. Geyer (6:1991) describes a similar method, involving multiple "Metropolis-coupled" chains, and Frantz and Freeman (6:1990) describe a related technique.

All these algorithms have close connections with the methods of free energy estimation discussed in the next section.

6.2 Free energy estimation

As discussed in Section 2.4, the free energy of a system is of great interest in statistical physics. In computing science, algorithms for approximate counting of large sets require that similar computations be carried out. Section 8 of the bibliography contains a number of papers on the problem drawn from these fields. Methods for free energy estimation can also be applied to Bayesian model comparison and to the estimation of rare event probabilities. Such applications are not yet widespread, but the importance of this topic, especially for Bayesian inference, seems clear.

Recall that the free energy of a system with energy function $E(s)$, at temperature T, is defined as $F = -T\log(Z)$, where $Z = \int \exp(-E(s)/T)\,ds$ is the partition function. All the methods discussed here in fact calculate only the *difference* in the free energy of two systems with the same state space but different energy functions, $E_0(s)$ and $E_1(s)$, or of the same system at two different temperatures, T_0 and T_1, or of two systems differing in both energy function and temperature. The partition functions for the two systems will in all cases be denoted by Z_0 and Z_1.

The free energy difference is what is required for Bayesian model comparison, where each system corresponds to a different model for the data, since from this difference one can easily calculate the *ratio* of the probabilities of the observed data under the two models.

Multiplying this by the ratio of the prior probabilities for the models gives the desired ratio of posterior probabilities. The requirement that the state spaces of the two systems be the same can often be accommodated by introducing dummy parameters into one of the models. In some cases, however, it will be easiest to compare models by calculating the *absolute* free energy of each, which can be done by calculating the difference in free energy from some reference system whose free energy can be obtained analytically.

Actually, rather than directly computing the free energy, we will instead focus on computing the difference $\log(Z_1) - \log(Z_0)$, or, equivalently, the ratio Z_1/Z_0. Knowing $\log(Z_1) - \log(Z_0)$, we can easily compute the free energy difference for systems at the same temperature, though not for systems at different temperatures, unless we also know the absolute free energy for one of the systems. This last point is not important in a statistical context, where the interesting systems all have $T = 1$.

Estimating the free energy using simple importance sampling. Before beginning the general discussion, I will first show how the free energy of a system can be estimated when a good importance sampling function is available. Recall from Section 3.2 that in simple importance sampling we find expectations with respect to a distribution proportional to $f(x)$ by sampling from a distribution proportional to $g(x)$. For good results, the distribution defined by $g(x)$ must be close to that defined by $f(x)$. Using such an importance sampling function, the ratio of the partition functions for $f(x)$ and $g(x)$ may be expressed as follows:

$$\frac{Z_f}{Z_g} \;=\; \sum_{\tilde{x}} f(\tilde{x}) \Big/ \sum_{\tilde{x}} g(\tilde{x}) \tag{6.10}$$

$$= \sum_{\tilde{x}} \frac{f(\tilde{x})}{g(\tilde{x})} g(\tilde{x}) \Big/ \sum_{\tilde{x}} g(\tilde{x}) \tag{6.11}$$

$$= E_g\big[f(X)/g(X)\big] \tag{6.12}$$

Accurate results can be obtained by estimating the above expectation using simple Monte Carlo methods only if the ratio $f(x)/g(x)$ does not exhibit extreme variations. If such a well-matched importance sampler can be found, and if the value of Z_g for it can be calculated analytically, then one can find the absolute value of Z_f.

For the problems with which this review is primarily concerned, such good importance sampling functions cannot be found *a priori*. However, this method of calculating the free energy may be applied in connection with Markov chain methods, particularly Gibbs sampling, by constructing an importance sampler from points obtained from one or more simulations of a Markov chain that converges to the distribution defined by $f(x)$. If these points are $x^{(0)}, \ldots, x^{(N-1)}$, we can use the importance sampling distribution given by

$$g(x) \;=\; \sum_{t=0}^{N-1} \frac{1}{N}\, T(x^{(t)}, x) \tag{6.13}$$

where $T(x, x')$ is the transition probability function for the Markov chain. While simulating the chain requires only that one be able to sample from this transition distribution, to use the above importance sampler one must also be able to calculate the probability of choosing a particular point. This is possible at least in some applications of Gibbs sampling. When it is not possible, one could try replacing $T(x^{(t)}, x)$ in equation (6.13) by some other distribution centred on the sample point.

This method will give good results only if the points $x^{(0)}, \ldots, x^{(N-1)}$, completely "cover" the distribution defined by $f(x)$. Informally, every region of high probability under $f(x)$ must be only one transition away from a point in the sample. This will not be possible for very high dimensional problems with a sample of any reasonable size (unless a single iteration is nearly sufficient to move anywhere in the distribution), but it may be true for some problems of only moderate dimensionality for which Markov chain methods are nevertheless appropriate.

This method is related to the "Gibbs stopper" convergence diagnostic of Ritter and Tanner (4:1992), discussed in Section 6.3.

The need for intermediate systems. Returning to the problem of finding free energy differences for general systems, one could try calculating Z_1/Z_0 directly via Monte Carlo simulation by expressing it as follows:

$$\frac{Z_1}{Z_0} = \frac{1}{Z_0}\int \exp\big(E_0(s)/T_0 - E_1(s)/T_1\big)\,\exp\big(-E_0(s)/T_0\big)\,ds \tag{6.14}$$

$$= \big\langle \exp\big(-\Delta(s)\big)\big\rangle_0 \tag{6.15}$$

where $\Delta(s) = E_1(s)/T_1 - E_0(s)/T_0$ and $\langle\cdot\rangle_0$ denotes expectation with respect to the canonical distribution for system 0, defined by the energy E_0 and temperature T_0. (This formula is actually the same as equation (6.12), but in a different guise.) One could also use an analogous expression involving an expectation with respect to system 1.

Unfortunately, this method works only if the energy functions and temperatures of the two systems are nearly the same. Otherwise, $\exp(-\Delta(s))$ may take on very large values in regions that have very low probability under the canonical distribution for system 0. A Monte Carlo estimate of the expectation in equation (6.15) will in such a case be very inaccurate.

We will see that there may be better ways of estimating Z_1/Z_0 than by using equation (6.15), but virtually all such methods are likewise feasible only for systems that are sufficiently similar. To find Z_1/Z_0 for two arbitrary systems, we can build a path between them via a series of intermediate systems, labeled by the numbers $0 = \alpha_0 < \alpha_1 < \cdots < \alpha_{n-1} < \alpha_n = 1$. We can then express Z_1/Z_0 as follows:

$$\frac{Z_1}{Z_0} = \frac{Z_1}{Z_{\alpha_{n-1}}}\,\frac{Z_{\alpha_{n-1}}}{Z_{\alpha_{n-2}}}\cdots\frac{Z_{\alpha_2}}{Z_{\alpha_1}}\,\frac{Z_{\alpha_1}}{Z_0} \tag{6.16}$$

Provided that system α_i is in each case sufficiently similar to system α_{i+1}, each of the above factors can be found by Monte Carlo methods, using the analog of equation (6.15), or one of the other methods described below.

The series of intermediate systems needed can be constructed in various ways. If systems 0 and 1 differ only in their energy functions, for example, system α could be given the energy function

$$E_\alpha(s) = (1-\alpha)E_0(s) + \alpha E_1(s) \tag{6.17}$$

If the systems instead differ in temperature, a series of systems at intermediate temperatures can be constructed similarly.

For statistical problems, system 1 could be defined to have as its canonical distribution the posterior distribution for the model parameters given a set of training cases, as in equation (2.63). System α for $\alpha < 1$ could then be defined in like fashion, but using only a fraction α of the full training set, with system 0 corresponding to the prior distribution.

Typically, Z_0 will be easy to find analytically, allowing an absolute value for the probability of the data under the model to be found. (For the system of equation (2.63), $Z_0 = 1$.) Other paths connecting the posterior distribution with a tractable reference system, such as to one at infinite temperature, may be possible as well. One could also attempt a direct comparison of two models with the same parameter space using the intermediate systems of equation (6.17).

Data from each of the intermediate systems needed to compute the ratios in equation (6.16) must be obtained via Monte Carlo simulations. Performing each such simulation independently may be wasteful, however. It will generally be better to start at one end of the path (at either system 0 or system 1), and then proceed step-by-step to the other end, using the final state from each simulation as the initial state for the next simulation. When the systems differ only in their temperature, and we start at the high temperature end, this procedure is identical to simulated annealing, which we saw in Section 6.1 may be desirable in any case. Note, however, that for the purposes of free energy estimation, we must take care to remain close to the equilibrium distribution at all times, a matter that is of less concern when annealing is used simply to speed convergence.

The reference system, and the series of intermediate systems linking the reference system to the system of interest, need not be defined *a priori*, before any data has been collected. Instead, one can use a sample of states from the system of interest to select a simpler system that is assured of being close enough to the current system to permit accurate estimation of the corresponding free energy difference. The process is then repeated for this simpler system, and so on, terminating in a system simple enough that its free energy may be calculated directly.

This method, which goes back at least to Jerrum, Valiant, and Vazirani (7:1986), is used by Dagum and Chavez (2:1993) to estimate the probability of rare events modelled using a belief network; similar methods should be applicable to other models. Suppose we wish to find the conditional probability that a binary variable X has the value 1, given the observed values for other variables, Z, and given that $Y_1, \ldots, Y_k$ are also unobserved. If we are content with a reasonable absolute error bound for this estimate, a single Gibbs sampling simulation will give an adequate estimate. If we require the estimated probability to have a small relative error, however, we must employ the methods of this section, since the required absolute accuracy may then be very high if the probability of X being 1 is very low.

One approach would be to define intermediate systems in which $Y_1, \ldots, Y_k$ were successively set to predetermined values, say to 0 in each case. We could then express the probability that $X = 1$ in conjunction with the observed data as follows:

$$\begin{aligned} P(X=1,z) \;&=\; \frac{P(X=1,z)}{P(X=1,Y_1=0,z)} \cdot \frac{P(X=1,Y_1=0,z)}{P(X=1,Y_1=0,Y_2=0,z)} \cdots \\ &\quad \frac{P(X=1,Y_1=0,\ldots,Y_{k-1}=0,z)}{P(X=1,Y_1=0,\ldots,Y_k=0,z)} \cdot P(X=1,Y_1=0,\ldots,Y_k=0,z) \end{aligned} \tag{6.18}$$

Each of the ratios above would be estimated using data from a Gibbs sampling simulation of the less constrained system; the final factor, in which all the variables have fixed values, is directly computable. We could estimate $P(z)$ in the same way, and from the ratio of the two, obtain $P(X = 1 \mid z)$.

This will fail, however, if $P(X = 1, Y_1 = 0, \ldots, Y_i = 0 \mid X = 1, Y_1 = 0, \ldots, Y_{i-1} = 0)$ turns out to be very small, for some i. The frequency with which Y_i takes on the value 0 in

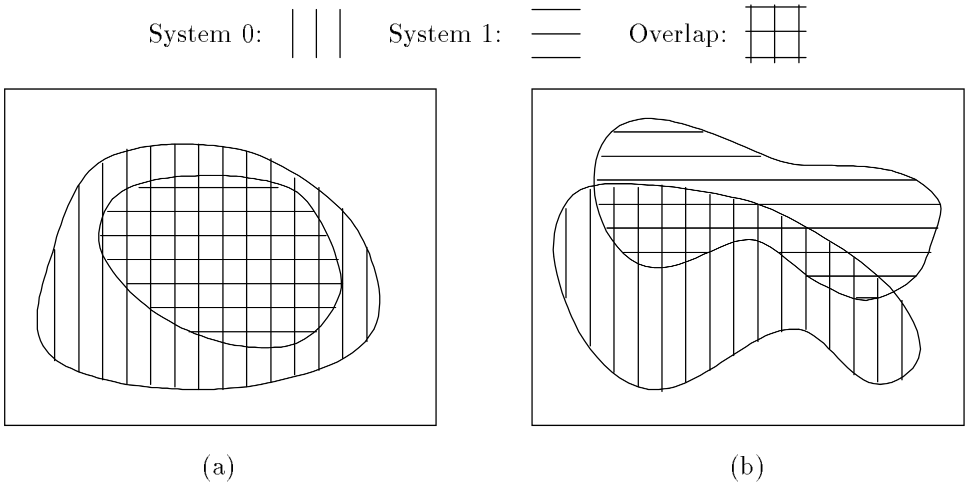


Figure 6.2: Estimating a free energy difference by distribution overlap, for systems where the energy of a state is either zero or infinity. The true answer is given by $Z_1/Z_0 = \mathrm{A(System1)/A(System0)}$, where A(System0) is the area of state space accessible under system 0, and similarly for A(System1). The Monte Carlo estimate based on equation (6.15) will be approximately A(Overlap)/A(System0), where A(Overlap) is the area of the region accessible to both systems. This is accurate when, as in (a), the region accessible to system 1 is a subset of that accessible to system 0. When this is not so, as in (b), estimating Z_1/Z_0 as (A(Overlap)/A(System0)) / (A(Overlap)/A(System1)) is better.

a Gibbs sampling simulation with variables up to Y_{i-1} fixed to 0 will then provide a very poor estimate of the corresponding ratio. This problem can be avoided by deciding on the value to which Y_i will be fixed only after data from the simulation is available; we then just choose the value that has shown up more often, ensuring that we have adequate data for estimating the corresponding ratio.

The acceptance ratio method. Accurate Monte Carlo estimation of the expectation in equation (6.15) requires that the canonical distribution for system 0, given by $P_0(s)$, assign reasonably high probability to all the regions where $\exp(-\Delta(s))P_0(s)$ is large, which are just those regions that have high probability under the canonical distribution for system 1. This is most easily visualized for systems where the energy of a state is always either zero or infinity, giving canonical distributions that are uniform over some subset of the whole state space. The condition for a Monte Carlo estimate based on equation (6.15) to be accurate is then just that the region of non-zero probability under system 1 be a subset of that under system 0, as illustrated in Figure 6.2(a). If system 1 can move to regions where system 0 cannot, as in Figure 6.2(b), no amount of sampling from the distribution for system 0 will give an accurate estimate. For systems with energies that are never actually infinite, an estimate based on equation (6.15) will always converge eventually, but only very slowly if some regions with high probability under system 1 have low probability under system 0.

When the space visited by system 1 is not contained in that visited by system 0, a better estimate than that of equation (6.15) can be obtained by sampling from *both* systems, rather than from system 0 alone. For the systems of Figure 6.2(b), this allows us to estimate both the fraction of the region accessible to system 0 that is also accessible to system 1, and the fraction of the region accessible to system 1 that is also accessible to system 0. From these quantities, we can then find Z_1/Z_0, which in this case is just the ratio of the sizes of the regions accessible to the two systems. This method of estimation will be accurate as long as the two regions overlap significantly.

The *acceptance ratio* method of Bennett (7:1976) extends this concept to general systems. It can be visualized by considering an extended system with state space (S, W), where S represents the state variables of systems 0 and 1, and W is an additional state variable that takes on the value 0 or 1. The energy function for this extended system is defined to be

$$E_*(s, w) \;=\; \begin{cases} E_0(s)/T_0 & \text{if } w = 0 \\ E_1(s)/T_1 - C & \text{if } w = 1 \end{cases} \tag{6.19}$$

where C is a constant whose role will be made clear in a moment. The temperature of the extended system is taken to be one.

We could simulate the canonical distribution for this extended system, using the same transitions to alter S as we would have used for a simulation of system 0 or 1, and picking new values for W using Gibbs sampling. The conditional distributions needed for Gibbs sampling are

$$P(W = 1 \mid s) \;=\; \frac{\exp(-E_*(s,1))}{\exp(-E_*(s,0)) + \exp(-E_*(s,1))} \;=\; \sigma\big(-\Delta(s) + C\big) \tag{6.20}$$

$$P(W = 0 \mid s) \;=\; \frac{\exp(-E_*(s,0))}{\exp(-E_*(s,0)) + \exp(-E_*(s,1))} \;=\; \sigma\big(+\Delta(s) - C\big) \tag{6.21}$$

where $\sigma(z) = 1/(1+\exp(-z))$ and, as before, $\Delta(s) = E_1(s)/T_1 - E_0(s)/T_0$. While this simulation runs, we could observe the proportion of time the extended system spends in states with $W = 1$, versus $W = 0$. This would let us estimate

$$\frac{P_*(W=1)}{P_*(W=0)}\exp(-C) \;=\; \frac{\int \exp\big(-E_1(s)/T_1 + C\big)\big/ Z_*\, ds}{\int \exp\big(-E_0(s)/T_0\big)\big/ Z_*\, ds}\exp(-C) \;=\; \frac{Z_1}{Z_0} \tag{6.22}$$

Of course, this only works if both $P_*(W = 1)$ and $P_*(W = 0)$ are appreciably greater than zero, as otherwise the simulation might never sample one or the other of these, giving an estimate for Z_1/Z_0 of either zero or infinity. This can always be arranged by adjusting the constant C. Furthermore, for the simulation to produce a useful sample in a reasonable amount of time, the distributions of S for the two systems must overlap appreciably, as only in the overlap region will changes in W have a significant probability of occurring.

Rather than actually simulating such an extended system, Bennett instead proposes that systems 0 and 1 each be simulated, and data gathered on what the probability *would be* of a transition from $W = 0$ to $W = 1$, and vice versa, if the extended system were to be simulated. As a consequence of detailed balance holding, the ratio of these transition probabilities also gives the ratio $P_*(W = 1)/P_*(W = 0)$, allowing Z_1/Z_0 to be estimated as

$$\frac{Z_1}{Z_0} \;=\; \frac{\big\langle \sigma\big(-\Delta(s) + C\big)\big\rangle_0}{\big\langle \sigma\big(+\Delta(s) - C\big)\big\rangle_1}\exp(-C) \tag{6.23}$$

The data required for making such an estimate can easily be gathered for various values of C during a single pair of runs, after which a value for this constant can be selected that results in both transition probabilities being non-negligible. Of course, if the distributions for the two systems have no significant overlap, no value of C will give a good estimate, and one or more intermediate systems will have to be introduced.

Voter (7:1984) generalizes the acceptance ratio method by considering Metropolis moves that not only change W (thereby switching energy functions), but also simultaneously change S by adding or subtracting a displacement vector. This allows the free energy difference between a pair of dissimilar systems to be computed with a single application of the acceptance ratio method, with no intermediate systems, provided that the distribution for each system is fairly concentrated, with the offset between the areas of concentration known in advance, or obtainable from preliminary runs.

Thermodynamic integration. An alternative to the use of estimation formulas based on distribution overlap such as (6.15) or (6.23) is to find the derivative of $\log(Z)$ at a number of points along a path between systems 0 and 1, and then apply some numerical integration procedure to find $\log(Z_1)-\log(Z_0)$. This method of *thermodynamic integration* corresponds to the way free energy differences for actual physical systems are measured experimentally.

If the systems differ only in their energy functions (their temperatures being the same), the required derivatives can be estimated as follows:

$$\frac{d\log(Z_\alpha)}{d\alpha} = \frac{1}{Z_\alpha}\frac{dZ_\alpha}{d\alpha} = \frac{1}{Z_\alpha}\frac{d}{d\alpha}\int \exp\left(-E_\alpha(s)/T\right) ds \tag{6.24}$$

$$= -\int \frac{1}{T}\frac{dE_\alpha(s)}{d\alpha} \cdot \frac{1}{Z_\alpha}\exp\left(-E_\alpha(s)/T\right) ds \tag{6.25}$$

$$= -\frac{1}{T}\left\langle \frac{dE_\alpha(s)}{d\alpha}\right\rangle_\alpha \tag{6.26}$$

If the intermediate systems are defined by the energy function of equation (6.17), we will have $dE_\alpha(s)/d\alpha = E_1(s) - E_0(s)$. For this and other typical choices, the derivative does not vary greatly with s, and hence Monte Carlo estimation of the expectation in equation (6.26) is feasible.

We can proceed similarly when the systems differ in temperature instead. Defining $\beta = 1/T$, we can compute the following at various points from β_0 to β_1:

$$\frac{d\log(Z_\beta)}{d\beta} = \frac{1}{Z_\beta}\frac{dZ_\beta}{d\beta} = \frac{1}{Z_\beta}\frac{d}{d\beta}\int \exp\left(-\beta E(s)\right) ds \tag{6.27}$$

$$= -\int E(s) \cdot \frac{1}{Z_\beta}\exp\left(-\beta E(s)\right) ds \tag{6.28}$$

$$= -\left\langle E(s)\right\rangle_\beta \tag{6.29}$$

For each intermediate system, with parameter α_i or β_i, the expectation of equation (6.26) or (6.29) is computed by averaging over the states found by continuing the Markov chain with this new parameter for some number of steps. The *slow growth* method, described by Basho, Singh, Langridge, and Kollman (7:1987) is a limiting case of this, in which the number of intermediate systems is very large, but only one step of the Markov chain is done for each system. Integrating the derivative of equation (6.26), estimating each expectation using the single state generated at each value of α, gives the following formula:

$$\log(Z_1) - \log(Z_0) \approx \sum_{i=0}^{n-1} -\frac{1}{T}\left(E_{\alpha_{i+1}}(s^{(i)}) - E_{\alpha_i}(s^{(i)})\right) \tag{6.30}$$

where $s^{(i)}$ is the state found by performing one transition in the Markov chain designed to converge to the canonical distribution for E_{α_i}, starting from the state produced for the previous system, $s^{(i-1)}$. The α_i must be spaced sufficiently closely that performing a single transition at each value is sufficient to maintain the chain near equilibrium. The slow growth method can be applied similarly for systems differing in temperature.

As with the acceptance ratio method, computation of free energy differences using thermodynamic integration requires, in general, that a whole series of Monte Carlo simulations be performed. Unlike the acceptance ratio method, however, the number of intermediate simulations needed is not obviously related to the overlap in the distributions they sample. Instead, for an accurate estimate, it is necessary only that the points where the derivative is evaluated adequately represent the curve being integrated. If one is willing to make some assumptions concerning how smooth the derivatives are as a function of α or β, the number of simulations required may be many fewer than are needed to estimate the free energy by the acceptance ratio method. In particular, one would not expect the number of points required for thermodynamic integration with a simulated physical system to grow with system size, whereas the number of points needed does grow for the acceptance ratio method, since the energies of a state in the two systems, and hence also the difference in energies, are extensive quantities.

Other methods. The acceptance ratio and thermodynamic integration methods are representative of two general approaches to free energy estimation. Many other methods have been investigated, including a number that are particular to certain types of physical systems. Here, I will briefly describe two that are generally applicable.

As mentioned above, the thermodynamic integration method can sometimes give good results using fewer intermediate systems than the acceptance ratio method, provided an assumption of smoothness is justified. Bennett (7:1976) has proposed an *interpolation* method which, like the acceptance ratio method, is based on measuring the overlap in the canonical distributions for the two systems being compared, but which, by utilizing a smoothness assumption, can allow computation of free energy differences even when the actual data collected does not exhibit any overlap.

Like the acceptance ratio method, the interpolation method utilizes data from Monte Carlo simulations of both of the systems being compared. During each simulation, a histogram of values for $\Delta(s)$ is accumulated. If both simulations record a significant number of counts in some overlap region, Z_1/Z_0 can be estimated by comparing the distributions where they overlap (the acceptance ratio formula (6.23) can be seen as one implementation of this). If there is no significant region of overlap, a value can still be obtained by projecting both distributions into the gap. This projection is more in the nature of interpolation than extrapolation, because the projections are not independent, but are rather related in a form that is fixed up to the value of Z_1/Z_0, which is being estimated. Each projection is thus constrained by the data on both sides.

The *umbrella sampling* method of Torrie and Valleau (7:1977) is another approach to improving the efficiency of free energy estimation using distribution overlap. It is based on the expression for Z_1/Z_0 given by equation (6.15), but estimates the expectation in this formula using importance sampling, rather than by simulating the canonical distribution for system 0 itself. The importance sampling distribution used, $P_*(s)$, has the form

$$P_*(s) \quad = \quad \frac{1}{Z_*} W\big(\Delta(s)\big) \exp\big(-E_0(s)/T_0\big) \tag{6.31}$$

where $W(\,\cdot\,)$ is a weighting function that is chosen by trial and error, on the basis of preliminary simulations, so as to produce a distribution that covers the ranges of $\Delta(s)$ that would be seen in the canonical distributions of both system 0 and system 1. Intermediate values for $\Delta(s)$ must also be covered, if the simulation is to reach equilibrium and produce an adequate sample in a reasonable period of time. Mezei (7:1987) gives a heuristic procedure for determining a good weighting function automatically.

Applying the importance sampling formula (3.5) to equation (6.15), we get

$$\frac{Z_1}{Z_0} \;=\; \big\langle \exp\left(-\Delta(s)\right)\big\rangle_0 \;=\; \frac{\big\langle \exp\left(-\Delta(s)\right) P_0(s) / P_*(s)\big\rangle_*}{\big\langle P_0(s)/P_*(s)\big\rangle_*} \tag{6.32}$$

$$= \frac{\big\langle \exp\left(-\Delta(s)\right) / \,W\big(\Delta(s)\big)\big\rangle_*}{\big\langle 1 \,/\, W\big(\Delta(s)\big)\big\rangle_*} \tag{6.33}$$

Use of intermediate systems may still be convenient, although, in theory, an appropriate weighting function could allow the free energy for even quite dissimilar systems to be determined from a single simulation run. One cannot escape the inherent difficulties of the problem in this way, however, as this single umbrella sampling simulation would still have to sample from regions of state space intermediate between those typical of the two systems. The time required to gather adequate statistics in a single such simulation run could therefore be comparable to that needed for the whole series of simulation runs used by the other methods.

Umbrella sampling and other importance sampling schemes have also been used for other estimation problems where the canonical distribution does not adequately sample the regions of interest. It is related to the *multicanonical* method discussed at the end of Section 6.1.

6.3 Error assessment and reduction

In this section, I discuss the problem of assessing the error of a Monte Carlo estimate obtained from data generated by Markov chain methods. I also discuss techniques for reducing this error, and the choice of an overall strategy for obtaining good estimates with realistic error indications. A variety of techniques for reducing the error of Monte Carlo estimates may also be found in the texts by Hammersley and Handscomb (1:1964), Kalos and Whitlock (1:1986), and Ripley (1:1987).

Somewhat ironically, the error assessment procedures described here are nearly all frequentist in nature — they say how frequently the Monte Carlo procedure will produce an estimate close to the true value, not how probable it is that the true value in a particular instance is close to the estimate that has been obtained. The possibility of using Bayesian procedures instead is discussed in Section 7.1.

Assessing error in simple Monte Carlo estimates. Recall the basic Monte Carlo estimation formula:

$$\langle a\rangle \;=\; \int a(\tilde{x})\,P(\tilde{x})\,d\tilde{x} \;\approx\; \frac{1}{N}\sum_{t=0}^{N-1} a(x^{(t)}) \;=\; \frac{1}{N}\sum_{t=0}^{N-1} a^{(t)} \;=\; \bar{a} \tag{6.34}$$

where the $x^{(t)}$ each have marginal distributions given by $P(\,\cdot\,)$. The above average, $\bar{a}$, is an unbiased estimate of $\langle a\rangle$ — i.e. $E[\bar{a}] = \langle a\rangle$, where $E[\cdot]$ denotes expectation with respect to the possible realizations of the Monte Carlo procedure.

If the $x^{(t)}$ are *independent*, the Law of Large Numbers guarantees that $\bar{a}$ converges almost certainly to $\langle a\rangle$ as N increases, whenever $\langle a\rangle$ exists. More can be said when the variance,

$\sigma^2 = \langle (a - \langle a \rangle)^2 \rangle$ also exists. In this case, the variance of the estimate $\bar{a}$, found using N independent points, will be

$$\mathrm{Var}[\bar{a}] \;=\; E[(\bar{a} - E[\bar{a}])^2] \;=\; E[(\bar{a} - \langle a \rangle)^2] \;=\; \sigma^2 \,/\, N \tag{6.35}$$

Furthermore, the Central Limit Theorem guarantees that, asymptotically, the distribution of $\bar{a}$ is Gaussian. From equation (6.35), one can see that the amount of effort required to obtain a Monte Carlo estimate of a given accuracy is not directly related to the dimensionality of the space, in contrast to standard numerical integration procedures. However, since the standard error, $\sqrt{\mathrm{Var}[\bar{a}]}$, goes down only as $\sqrt{N}$, obtaining very precise estimates is laborious. Fortunately, typical applications in statistics and artificial intelligence do not require high precision.

In practice, equation (6.35) does not allow one to assess the variance of $\bar{a}$ because σ^2 is not known. The usual frequentist estimate for σ^2 is

$$\sigma^2 \;\approx\; \frac{1}{N-1} \sum_{t=0}^{N-1} \left(a^{(t)} - \bar{a}\right)^2 \tag{6.36}$$

By substituting this estimate for σ^2 into equation (6.35), one can obtain an estimate of the variance of the Monte Carlo estimate of $\langle a \rangle$:

$$\mathrm{Var}[\bar{a}] \;\approx\; \frac{1}{N(N-1)} \sum_{t=0}^{N-1} \left(a(t) - \bar{a}\right)^2 \tag{6.37}$$

This formula is sometimes naively regarded as providing a complete solution to the error assessment problem. This is not the case, as without further information, one can give no bound on the error in the estimate of σ^2 given by equation (6.36). When this estimate is seriously in error, it is possible that equation (6.37) will give a wildly optimistic impression of the accuracy of $\bar{a}$ as an estimate for $\langle a \rangle$. This will likely occur, for example, if $a(x)$ takes on extremely large values in a region of very low probability. If no points from this region are included in the sample used to compute $\bar{a}$, this estimate could be very far from the true value of $\langle a \rangle$ without equation (6.37) providing any indication of this fact.

Assessing the error of an importance sampling estimate found using equation (3.6) is more difficult. Hastings (4:1970) gives an approximate formula for the variance of this estimator. As with equation (6.37), any estimate of the error based only on the data points can in some circumstances seriously overstate the accuracy of the result obtained. Such misleading error assessments will often occur when the importance sampling distribution used is very poor.

Assessing error in Markov Chain Monte Carlo estimates. Markov chain sampling will produce a series of *dependent* values, $x^{(0)}, \ldots, x^{(N-1)}$. In these circumstances, $\bar{a}$ from equation (6.34) is still an unbiased estimate of $\langle a \rangle$, provided we discard the states generated by the early part of the chain, before the equilibrium distribution has been reached.[10] However, the variance of this estimate is not given by equation (6.35) — typically, the true variance will be larger, though the reverse is possible as well. The problem of estimating this variance has been discussed by many authors, in many fields (see the references in Section 8 of the bibliography).

[10] However, as discussed in (Diggle, 8:1990, Section 3.6), $\bar{a}$ is not always the *best* unbiased estimate of $\langle a \rangle$. Due to the dependencies, points near the two ends of the series contain more information than points in the middle. Consequently, an average that weights the points near the end more heavily can have lower variance than $\bar{a}$. As this effect is quite small, however, I will not consider estimators other than $\bar{a}$ here.

Batching is a simple, intuitive way of obtaining a better assessment of the variance of $\bar{a}$. The N sample points are divided into k "batches", each consisting of $m = N/k$ consecutive sample points. (I will here assume that N is a multiple of k.) Estimates of $\langle a \rangle$ are found using the data in each batch, with the estimate from batch i being

$$\bar{a}_i \;=\; \frac{1}{m} \sum_{t=im}^{im+m-1} a^{(t)} \tag{6.38}$$

If m is large enough, these estimates will be nearly independent, as most of the sample points contributing to each will be distant from each other in the Markov chain, and hence, assuming ergodicity, nearly independent. The estimate of $\langle a \rangle$ based on the entire data set (equation (6.34)) can be expressed as the average of the batch estimates:

$$\bar{a} \;=\; \frac{1}{k} \sum_{i=0}^{k-1} \bar{a}_i \tag{6.39}$$

If the batch estimates, $\bar{a}_i$, are indeed independent, then the variance of $\bar{a}$ can be estimated as in equation (6.37):

$$\mathrm{Var}[\bar{a}] \;\approx\; \frac{1}{k(k-1)} \sum_{i=0}^{k-1} \big(\bar{a}_i - \bar{a}\big)^2 \tag{6.40}$$

The worth of this estimate depends crucially on the choice of batch size, m. If m is too small, so that the $\bar{a}_i$ are not really independent, then the variance estimate will likely be too small, leading to an unjustified confidence in the accuracy of $\bar{a}$ as an estimate of $\langle a \rangle$. On the other hand, if m is set larger than is necessary to make the $\bar{a}_i$ (almost) independent, the smaller value for k that results leads to a larger variance in the estimate of the variance itself. For small k, this will be cause for significant doubt concerning whether $\bar{a}$ is accurate even if the estimated variance from equation (6.40) is small.

One way to cope with this problem of choosing an appropriate batch size is to plot the estimated variance of $\bar{a}$ as calculated from equation (6.40) for a series of increasing batch sizes, m. If it weren't for the noise in the variance estimates, which increases with large m (small k), these estimates would approach the true value in the limit as m increased. From the plot, one may be able to visually pick out an approximation to this limiting value despite the noise. This is essentially the method used by Friedberg and Cameron (4:1970) (see also (Morales and Nuevo, 8:1990)).

A perhaps more sophisticated approach to estimating the accuracy of $\bar{a}$ is to use methods from time series analysis (for a readable introduction, see (Diggle, 8:1990)). The variance of $\bar{a}$ can be expressed in terms of the *autocovariance function* for the time series $a^{(t)} = a\big(x^{(t)}\big)$, which is defined as follows:

$$\gamma(s) \;=\; E\Big[\big(a^{(t)} - \langle a \rangle\big)\,\big(a^{(t+s)} - \langle a \rangle\big)\Big] \tag{6.41}$$

The *autocorrelation function* is defined as $\rho(s) = \gamma(s)/\sigma^2$. The definition for $\gamma(s)$ given here assumes, as before, that $E[a^{(t)}] = \langle a \rangle$. The definition is independent of the choice of t provided that the time series is *stationary* — that it looks the same from any starting point. This will be the case when the $x^{(t)}$ come from an ergodic Markov chain from which we have discarded the initial portion, prior to when the equilibrium distribution was reached.

Note that $\gamma(0) = \sigma^2$, $\gamma(s) = \gamma(-s)$, and $-\sigma^2 \le \gamma(s) \le +\sigma^2$. If the $x^{(t)}$, and hence the $a^{(t)}$, are independent, we have $\gamma(s) = 0$ for $s \ne 0$.

The variance of $\bar{a}$ can be expressed in terms of the autocovariance as follows:

$$\begin{aligned}
\mathrm{Var}[\bar{a}] \;&=\; E\big[(\bar{a}-\langle a\rangle)^2\big] \;=\; E\Big[\Big(\frac{1}{N}\sum_{t=0}^{N-1}\big(a^{(t)}-\langle a\rangle\big)\Big)^2\Big] && (6.42)\\
&=\; \frac{1}{N^2}\sum_{t,t'=0}^{N-1} E\big[\big(a^{(t)}-\langle a\rangle\big)\big(a^{(t')}-\langle a\rangle\big)\big] && (6.43)\\
&=\; \frac{1}{N^2}\sum_{t,t'=0}^{N-1}\gamma(t'-t) && (6.44)\\
&=\; \frac{1}{N}\sum_{-N<s<N}(1-|s|/N)\gamma(s) && (6.45)
\end{aligned}$$

If the $a^{(t)}$ are independent, this reduces to equation (6.35). For dependent sample points, it is possible for $\gamma(s)$ to be negative, in which case the variance can be *less* than with a sample of independent points. Though it is sometimes possible to arrange for short-range correlations to be negative when using Markov chain sampling, long-range correlations are typically positive, and for difficult problems contribute the most to the variance.

For large N, the variance from equation (6.45) can be written as follows:

$$\mathrm{Var}[\bar{a}] \;=\; \frac{1}{N}\Big[\sigma^2+2\sum_{s=1}^{\infty}\gamma(s)\Big] \;=\; \frac{\sigma^2}{N/\tau} \qquad (6.46)$$

where $\tau = 1+2\sum_1^\infty \rho(s)$ can be seen as the number of dependent sample points from the Markov chain are needed to give the equivalent of one independent point. (Note that it is possible, though not typical, for τ to be less than one.)

An estimate for the variance of $\bar{a}$ can be obtained from equation (6.45) by substituting estimates for $\gamma(s)$. Hannan proposed the following (see (Ripley, 1:1987, Chapter 6)):

$$\mathrm{Var}[\bar{a}] \;\approx\; \frac{N}{(N-L)(N-L+1)}\sum_{-L<s<L}\Big(1-\frac{|s|}{N}\Big)\Big(\frac{1}{N-|s|}\sum_t a^{(t)}a^{(t+s)} \;-\; \bar{a}^2\Big) \qquad (6.47)$$

where L is chosen so that $\gamma(s)\approx 0$ for $s\geq L$. Making a sensible choice for L on this basis is essential; too large a value for L introduces a large amount of noise, as the estimates for the autocovariances for large lags are based on little data. Other estimates of this general type are also possible, in which the estimates of $\gamma(s)$ for different s are weighted in various ways. Straatsma, Berendsen, and Stam (8:1986) use all the autocovariances, but estimate those at longer lags by fitting an exponential, rather than by noisy individual estimates.

Rather than work directly with the autocovariance function, we can instead work with the *spectrum* of the time series, defined as

$$f(\omega) \;=\; \sum_{s=-\infty}^{+\infty}\gamma(s)\cos(s\omega) \;=\; \Big[\sigma^2+2\sum_{s=1}^{\infty}\gamma(s)\cos(s\omega)\Big] \qquad (6.48)$$

From equation (6.46), we see that for large N

$$\mathrm{Var}[\bar{a}] \;=\; \frac{f(0)}{N} \qquad (6.49)$$

From an estimate of the spectrum, $f(\omega)$, we can thus obtain an estimate of $\mathrm{Var}[\bar{a}]$. Note, however, that since $f(\omega) = f(-\omega)$, a minimum or maximum will be located at $f(0)$ (unless the spectrum is flat, as it is when the points are independent). Standard methods for smoothing an estimate of the spectrum tend to flatten such peaks or troughs, and hence are not good when the focus is specifically on $f(0)$. Heidelberger and Welch (8:1981) avoid this problem by instead fitting a polynomial to the log of the spectrum in the vicinity of $f(0)$.

One can also proceed by fitting a parametric model to the time series, such as an autoregressive model of some order. Once the parameters of such a model have been estimated, equation (6.46) can typically be evaluated analytically to obtain a corresponding estimate for $\text{Var}[\bar{a}]$. Of course, the accuracy of this estimate will depend on whether the model used is adequate.

All the time series methods can be applied to a series of batch means, $\bar{a}_i$, rather than to the values $a^{(t)}$ themselves. In this context, the batches need not be large enough for the $\bar{a}_i$ to be approximately independent, as dependence will be handled by whatever time series method is used. The motive is instead to reduce the amount of storage and computation required. The volume of data can also be reduced simply by looking at only every mth iteration. This discards useful information, but avoids the need to calculate $a(x^{(t)})$ for every sample point. It may therefore be desirable when this calculation is expensive.

The methods described above can be adapted to handle estimation from R independent realizations of the chain, each assumed to start from the equilibrium distribution, and which for simplicity I will here assume are all of the same length. The estimate of $\langle a \rangle$ in this case is simply the average of $a(x)$ for all sample points. Expressed another way, if $\bar{a}_r$ is the average from the rth run, then the grand estimate is $\bar{a} = (1/R) \sum_r \bar{a}_r$.

The batch method for evaluating the variance of an estimate is easily adapted to handle multiple runs — one simply pools the batches from all the runs, and applies equation (6.40). It is possible to treat the data from each run as a single batch, and this certainly guarantees that the batches are independent (assuming each run did indeed reach equilibrium before the data was taken). This may not be the best way to proceed, however, as the variance estimate may then be based on a smaller number of batches than it might have been.

For the time series methods, data from all the runs can be used to estimate the autocovariance function or the spectrum, and from this one can estimate $\text{Var}[\bar{a}_r]$, which is the same for all r, given that all runs are the same length. The variance of the grand estimate can then be estimated using the fact that $\text{Var}[\bar{a}] = (1/R)\text{Var}[\bar{a}_r]$.

Typically, we are interested in estimating the expectation of more than one function of state, perhaps quite a large number if we wish to plot posterior or predictive distributions for various variables. In the literature, each such estimation problem is generally considered independently. Note that it is quite possible for the series of values of one function at successive iterations to be highly dependent, while the values of another function at successive iterations are almost independent. Estimating the expectation of the first function will then be harder than estimating the expectation of the second function. However, when there are long-range dependencies in the value of one function of state, it would be prudent not to conclude too hastily that some other function of state has no long-range dependencies, even if that is how the data concerning that function alone might otherwise be interpreted.

Diagnosing convergence. The methods of the previous section all assume that we have discarded an initial portion of each run, of sufficient length that the states in the remaining portion all come from the desired equilibrium distribution, or, more realistically, from distributions that are sufficiently good approximations to the desired distribution. In this section, I will discuss methods of choosing how much of each chain to delete in order to accomplish this — i.e. how to decide when the chain has (approximately) *converged* to the equilibrium distribution.

Technically speaking, convergence is a property of the Markov chain, regarded as a sequence of random variables, not of any particular realization of the chain. A Markov chain, $X^{(0)}, X^{(1)}, X^{(2)}, \ldots$, has converged by time t if the marginal distribution of $X^{(t)}$ is the same as that for all $X^{(t')}$ with $t' > t$, this distribution being, of course, the equilibrium distribution that the chain was designed to sample from. The chain has "almost" converged at time t if the distribution of $X^{(t)}$ and all subsequent states is "close" to the equilibrium distribution in some relevant sense.

In practice, however, one often speaks of a particular realization of a chain having "converged" by time t, while perhaps also saying that at time t another realization of the same chain has not converged. This makes no sense according to the above definition, but it is nevertheless a reasonable statement, even if imprecisely phrased.

As an example, consider sampling from a distribution with two modes, with the function a being close to 0 near one mode and close to 1 near the other. Assume that the region around the mode where a is close to 1 contains almost all of the probability, but that the chain starts at the mode where a is close to 0. Assume also that the Markov chain only rarely jumps between the regions around these modes. In a typical run, a will start out close to 0, and stay close to 0 for a rather long time. When a jump to the vicinity of the main mode finally occurs, a will change to a value close to 1, and probably remain close to 1 for a very long time, perhaps the rest of the run. In calculating the expectation of a, we would like to discard the early portion as not being representative of the value of a in the equilibrium distribution. Suppose we have determined theoretically that by $t = 1000$, a realization of the chain is very likely to have reached the main mode. We might decide to discard this amount from the chain. However, if, in an actual realization, the transition to the main mode occurs at $t = 500$, it seems wasteful to discard the 500 following states, waiting for the occurrence of an event that we know has already occurred. Conversely, in the unlikely event that the realization stays in the minor mode past $t = 1000$, it seems bad to blindly include these atypical values in the average.

Accordingly we will generally attempt to diagnose "convergence" for a particular realization of a chain (or for a set of realizations). The notion that convergence should be conditional on the observed realization could perhaps be formalized by the concept of a "convergence diagnostic" — a series of integer functions, $D_n(x^{(0)}, \ldots, x^{(n-1)})$, such that

$$P(X^{(t')} = x^{(t')} \mid D_n(X^{(0)}, \ldots, X^{(n-1)}) = t) \quad \approx \quad \pi(x^{(t')}), \quad \text{for all } t' \geq t \qquad (6.50)$$

where $\pi(x)$ is the equilibrium distribution. Having observed the realization $x^{(0)}, \ldots, x^{(n-1)}$, we would use D_n to determine how much of the realization to discard, confident that if the diagnostic satisfied equation (6.50) the resulting estimates will be nearly unbiased. This definition of a convergence diagnostic could be extended to handle sets of realizations. No such formal concept is employed with the methods described below, however.

Since the state, x, is typically of high dimension, and hence hard to examine directly, most methods for diagnosing convergence are based on monitoring one or a few scalar functions of the state, such as the functions whose expectation we are interested in estimating. Perhaps the most commonly-employed method is to simply plot these functions for increasing values of t, and visually judge when equilibrium behaviour has been reached.

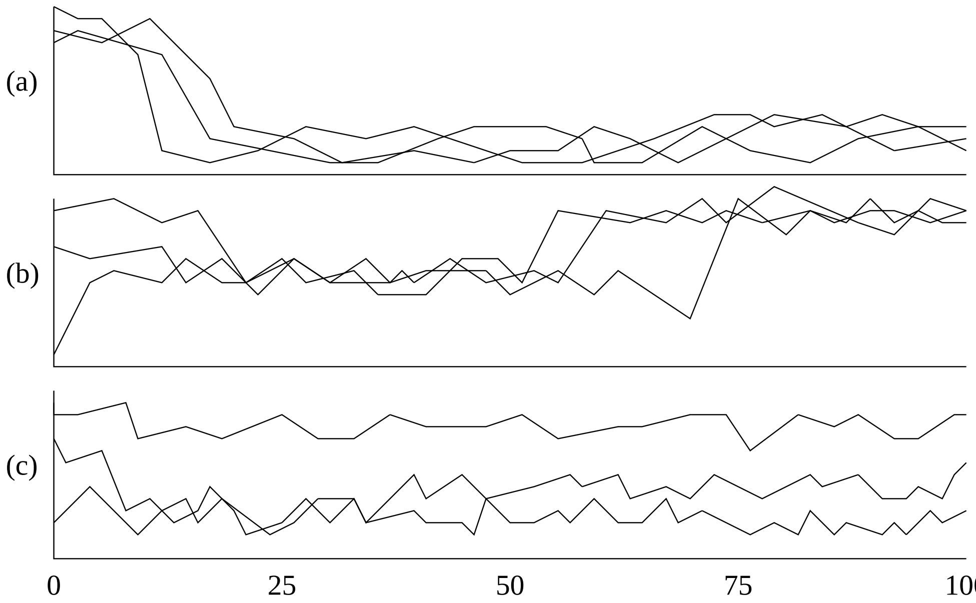


Figure 6.3: Realizations of three hypothetical Markov chains. Each of (a), (b), and (c) plot a scalar function of state versus time for three simulation runs, for different Markov chains. In (a), the runs have apparently all converged by about iteration 40. In (b), the runs at first appear to have converged after about iteration 25, but around iteration 50 this is seen to have been illusory; true convergence may, perhaps, have occurred around iteration 75. In (c), there is no evidence that any of the runs have converged within the first 100 iterations.

Figure 6.3 shows three hypothetical plots of this sort for a single function of state, for three realizations each of three different Markov chains. The behaviour seen is Figure 6.3(a) is what one hopes for — the runs show signs of all having converged after about 25 iterations. However, without some theoretical analysis, which is generally not available, there is no absolute guarantee that this apparent convergence is real. Figure 6.3(b) illustrates this possibility. If only 50 iterations had been simulated in each run, we might well have thought that the runs in (b) had all converged at about iteration 25, and that we should estimate the expectation by the average over iterations 25 to 50. With data from 100 iterations available, it is clear that this would have been a mistake, as it appears that the runs do not reach convergence until at least iteration 75. Figure 6.3(c) shows a case where the data give no reason to believe that any of the runs have converged within the duration of the simulations. (Note, however, that we cannot say with certainty that the runs in Figure 6.3(c) have *not* converged. Each could be sampling from the true equilibrium distribution, but moving about the state space so slowly that they appear to be sampling from different distributions when viewed over only 100 iterations.)

If only one of the runs in Figure 6.3(c) had been simulated, however, we might well have thought it had converged by iteration 50, or even before. This illustrates the added power of multiple runs in revealing when apparent evidence for convergence from a single run is in fact illusory. A single run from Figure 6.3(b) would also provide less information than is apparent from multiple runs, as one cannot tell from one run whether the change in function value after iteration 50 indicates that equilibrium had not been reached before then, or simply that there is a high degree of dependence between successive states. From the fact that all the runs in the figure behave similarly, however, one can conclude that equilibrium cannot have been reached before iteration 75 (or, rather, that would be a safe conclusion if it were based on somewhat more than three runs).

When more than one realization of the chain has been simulated, some authors recommend plotting the value of the function averaged across realizations versus time, looking for the point where this average appears to have stabilized. This approach would work satisfactorily for cases (a) and (b) in Figure 6.3, but in case (c) averaging across realizations would discard highly relevant data. While it is clear from the plots of individual runs in case (c) that we should not consider any of them to have converged, a plot of the average across runs would give every appearance of convergence by around iteration 50.

If we are monitoring two functions of state, $a(x)$ and $b(x)$, it may be that the distribution of a, say, appears to have converged even though that of b shows no signs of having reached an equilibrium. We might in this situation decide that we are in a position to estimate $\langle a \rangle$, using the values of a from the time of apparent convergence on. However, unless we understand why a should have converged even when b apparently has not, this may be dangerous, as if b has not converged, the states being visited are still not fully representative of the equilibrium distribution, and hence could well possess atypical values of a. In order to reveal such situations, it may be useful to monitor certain functions of state that are of fundamental significant to the process (such as the energy), even when the values of these functions are of no interest in themselves.

Hypothesis tests have been proposed by a number of authors for determining whether convergence has really been attained after some specified number of iterations. Ripley (1:1987, Section 6.1) reviews several that are based on values averaged over a number of runs. Geweke (8:1992) gives one based on the time average over an initial portion of the supposedly converged sequence versus a final portion. These approaches leave to the user's informal judgement the guess of a time at which the chain *may* have converged, addressing only the question of whether this guess was correct. It is not clear how such a guess is to be made, however, except perhaps on the basis of preliminary runs — basing the guess on the same data as is used to test it is not valid. It is also not clear that the hypothesis testing framework is appropriate for this task, in which the goal is not to reject the hypothesis, but rather to accept it.

A method of diagnosing convergence, known as the "Gibbs stopper", that is based on the entire state has been proposed by Ritter and Tanner (4:1992) (see also the comments by Cui, Tanner, Sinha, and Hall and by Gelfand following (Gelman and Rubin, 8:1993) and (Geyer, 8:1993)). Suppose that a large number, M, of independent realizations of the chain are simulated concurrently. This is the context in which the method was originally proposed, though it can also be applied using many samples from a single run. Denote the state variables from these runs at iteration t by $x^{(i,t)}$, for $i = 1, \ldots, M$. We can use these points to construct an approximation, $\hat{\pi}^{(t)}$, to the equilibrium distribution, as follows:

$$\hat{\pi}^{(t)}(x) \quad = \quad \frac{1}{M} \sum_{i=1}^{M} T(x^{(i,t)}, x) \tag{6.51}$$

where T gives the transition probabilities for the Markov chain. When the points from the next iteration, $x^{(i,t+1)}$, have been selected, we can compute the ratios between their unnormalized probabilities, given by $f(x)$, as used to derive the Markov chain, and their probability under $\hat{\pi}^{(t)}$:

$$w_i \quad = \quad f(x^{(i,t+1)}) \,/\, \hat{\pi}^{(t)}(x^{(i,t+1)}) \tag{6.52}$$

If the chain has converged by iteration t, we expect that the approximation $\pi^{(t)}$ will be good, and that the w_i will all be approximately equal. If $f(x)$ is the normalized probability under the desired distribution, all the w_i should in fact be close to 1, and if this is so, one can be sure that no areas of significant probability have been missed. Unfortunately, $f(x)$ is not normalized in typical applications, and this additional check is hence unavailable.

For this diagnostic to be of practical use, it must be feasible to calculate the transition probabilities, $T(x, x')$. Recall that these transition probabilities are usually built using a set of basic transitions. If each such transition changes a single component of x, and they are applied in a deterministic sequence, then the probability for moving from x to x' is just the product of the required basic transition probabilities. For Gibbs sampling, these will often be computable without great difficulty.

Use of this diagnostic also requires that it be possible to "cover" the entire distribution with a reasonable number of points. Otherwise, the w_i will not be nearly equal even once convergence has occurred. Whether it is possible to cover the distribution will depend both on the volume of the region of significant probability and on the characteristics of the Markov chain whose transition probabilities are used in equation (6.51) (which typically is the same as is used for the simulation). Unfortunately, for the hardest problems, where a good convergence diagnostic would be most useful, it is unlikely that this condition will be satisfied.

Choosing a good initial state distribution. The time required for convergence to the equilibrium distribution will often be greatly affected by the distribution used to select the initial state of the Markov chain. The initial distribution used may also affect how useful multiple realizations of the Markov chain are in diagnosing convergence — for this purpose, initial distributions concentrated at a single point appear less desirable.

The ideal initial distribution is the distribution we wish to sample from itself — except that if picking an initial state from this distribution were feasible, we would generally not be using a Markov chain method anyway. However, in some problems of moderate difficulty, it may be possible to generate values from the desired distribution, but only at a large computational cost, perhaps by using the rejection method (see Section 3.2). It might then make sense to generate a few such values and use them as as starting states for realizations of a Markov chain under which the desired distribution is invariant, thereby obtaining more information from each expensively-generated initial point. In this scheme, convergence of the Markov chain is immediate, so there is no need to discard an initial segment of the chain. It is also not essential that the Markov chain be ergodic.

For a Bayesian inference problem, where states consist of values for the model parameters and the desired equilibrium distribution is the posterior, one possible initial distribution is the prior distribution for the parameters, which presumably reflects our best guess as to what the parameters might be, given that we have as yet made no sense of the observed data. Prior distributions typically have simple, standard forms, from which it is easy to generate random values. Another possibility along the same lines is to always start at the parameter values with maximum prior probability density.

Alternatively, one might start with the parameters that have maximum posterior probability density, though finding such "MAP" parameter values may require a significant amount of work. One could also try to approximate the posterior distribution based on information available using the MAP parameters, such as the second derivatives of the probability density at that point, and then pick an initial value for the Markov chain from this approximation to the posterior. Gelman and Rubin (8:1993) advocate drawing the initial state from an "overdispersed" approximation to the distribution found in this way. Note that these methods break down when there are many modes, or when the mode is not representative of the full distribution. Both of these situations are common for problems in artificial intelligence and statistical physics.

A more elaborate scheme for constructing an approximation for use as an initial distribution is advocated by Goutsias (8:1991), in the context of image restoration. The model in this case resembles the 2D Ising system, in which the distribution is determined by interactions between pairs of nearby variables. This distribution can be approximated by one in which the interactions are expressed in terms of conditional probabilities for "later" variables given "earlier" variables, as in a belief network. Unlike the original distribution, sampling from this belief network approximation is easy. Goutsias selects such an approximation that minimizes the error as measured by "relative entropy". Alexandrowicz (9:1971) constructs similar approximations (for a different purpose) based on a dual measure of error.

Finally, simulated annealing may be viewed as a procedure for finding a good initial state for the Markov chain at the desired final temperature.

Choosing a good estimator of the expectation. The same quantity can often be expressed as an expectation in various ways, as in equations (2.8) and (2.9). The variances of such alternative Monte Carlo estimates may differ.

Kalos and Whitlock (1:1986, section 4.2) show that for Monte Carlo estimation using independent points, forms such as (2.9) are always better. Similar points have been made by other authors as well (e.g. Pearl (4:1987), Gelfand and Smith (4:1990)). As a specific example, suppose that the state is decomposed into components X_1 and X_2, and that we are interested in the expectation of some function $a(x_1, x_2)$. Define the function $b(x_1) = \langle a(x_1, X_2)\rangle$ — i.e. the expectation over the second argument of a with the first argument fixed at x_1. Clearly $\langle b\rangle = \langle a\rangle$. One can show, however, that the variance of b is less than the variance of a (or equal, in degenerates cases). Equation (6.35) relates these variances directly to the variance of the corresponding Monte Carlo estimate for a sample of independent points. Therefore, rather than evaluate $\langle a\rangle$ by averaging the values $a(x_1^{(t)}, x_2^{(t)})$, it is better to instead average the values $b(x_1^{(t)})$ — provided, of course, that $b(x_1)$ can be calculated efficiently.

If the estimates are based on a sample generated using a Markov chain, with the $x^{(t)}$ being dependent, it is possible to construct examples in which the estimate using b is worse than the estimate using a.[11] Nevertheless, it seems likely that in practice calculating part of the expectation analytically will generally improve the result.

Kalos and Whitlock also show how this idea can be used to improve the estimates produced using the Metropolis algorithm. Rather than taking an average over the current state at each step, one can instead take a weighted average of the current and candidate state at each step, using as weighting factors the probability of their being the next state in the chain. For the generalized hybrid Monte Carlo algorithm using windows of states (Neal,

[11] Consider a system with three binary variables, x_1, x_2, x_3, with all state probabilities equal. Let the probabilities for the transitions $000 \leftrightarrow 001$, $010 \leftrightarrow 011$, $100 \leftrightarrow 110$, and $101 \leftrightarrow 111$ be 0.999; the other possible transitions split the remaining probability. Let $a(x_1, x_2, x_3)$ have the value $+1$ in states 000, 100, 010, and 101, and the value -1 in the other states; let $b(x_2, x_3) = \langle a(X_1, x_2, x_3)\rangle$. Suppose we wish to estimate $\langle a\rangle$ from a sample of two consecutive states from the equilibrium distribution of this chain. The estimate obtained by averaging the values of a in the two states is very likely to be the correct value of zero; that obtained by averaging the values of b is very likely to be $\pm 1/2$.

5:1993), this idea can be extended to allow all states in the accept and reject windows of each trajectory to contribute to the estimate.

Strategies for Markov chain Monte Carlo. I will conclude this section by discussing how the issues dealt with above affect the choice of a strategy for allocating computation time between simulation runs and for making use of the results. The papers of Gelman and Rubin (8:1993) and of Geyer (8:1993) and the accompanying discussion provide a view of the spectrum of recent opinion on these issues amongst statisticians using Markov chain sampling for Bayesian inference.

In my view, no single strategy is best for all problems, but one can identify a set of "typical" problems for artificial intelligence applications for which one particular strategy seems better than the others. The following problem characteristics appear relevant to choosing an appropriate strategy; in each case I have indicated what I will consider the "typical" case:

1) What form is the distribution being sampled from expected to take? How many modes may there be — just one? a few? a huge number? Will most of the probability mass be in the vicinity of these modes? I take the presence of numerous modes to be typical, and assume that the dominant probability mass may not be in the vicinity of these modes.

2) Is there significant doubt concerning whether the Markov chain will reach its equilibrium distribution in a reasonable length of time? Or, on the other hand, are the initial distribution and the transition probabilities so good that quite rapid convergence is expected? I assume that in typical situations there is significant uncertainty concerning how long it will take for the simulation to converge.

3) How does the anticipated time to reach equilibrium compare to the anticipated time to move from one point drawn from the equilibrium distribution to another nearly independent point? I assume that typically the time to reach convergence is at least as long as the time to move about the distribution once convergence is reached.

4) Is it feasible to "cover" the entire distribution with a reasonable number of sample points? Here, a distribution is "covered" if the areas "near" to sample points contain almost all the probability mass, where "near" is defined in terms of the transitions of the chain. I assume that typically one cannot cover the region.

5) What is the purpose of the analysis — interpretation for scientific insight? Prediction with the need for an accurate indication of uncertainty? A "best guess" point prediction? I assume that we typically will wish to make a prediction with an indication of uncertainty, but might settle for a "best guess" if that is all that is reasonably available.

The "typical" situation described above for artificial intelligence problems is also reasonably typical of statistical physics problems, but not of all problems in statistical applications.

The following basic strategies can be distinguished:

a) Simulate one very long realization of the Markov chain. This method is advocated by Geyer (8:1993).

b) Simulate a small number of relatively long chains. This is advocated by Gelman and Rubin (8:1993).

c) Simulate a large number of relatively short chains. This is the method used by Gelfand and Smith (4:1990).

Strategy (b) fades smoothly into strategy (c) as the number of runs increases. I assume that in strategy (b) the number of runs is on the order of ten — enough that it is unlikely that the runs seen are all atypical.

Strategy (a) will be appropriate when there is substantial doubt that equilibrium can be reached in a shorter run, and, as is typical, there is no sure way of verifying that a shorter run that superficially appears to have reached equilibrium has actually done so. The best bet may then be to devote all the computation time to simulating a single chain, in order to maximize the chances of reaching the true equilibrium distribution. (This assumes that the initial distribution is poor, so that it is unlikely that equilibrium could be more quickly reached from some other initial state.) This situation could arise either because the problem being tackled is extraordinarily difficult — often the case in statistical physics — or because the time allowed for the computation is very limited — as might be the case, for example, in a real-time image-processing application. Of course, in these circumstances, one may not be able to obtain a good indication of how inaccurate the result might be.

Strategy (a) might also be appropriate when there is very little doubt that equilibrium can be reached in a time much shorter than the total allotted. If the time to reach equilibrium is at least as long as the time required to move to a nearly independent point in the equilibrium distribution, then a single run will be the most efficient method.

Strategy (c) might be appropriate when rapid convergence to equilibrium is either guaranteed on theoretical grounds, or can be verified to have occurred to a satisfactory degree of certainty (the ability to "cover" the distribution might be helpful here). This strategy would be especially advantageous if the time for convergence was less than the time required to move from one point of the equilibrium distribution to an independent point. This situation could arise if the initial state distribution used were very good, but the distribution being sampled from was multimodal, with movement from the area around one mode to that around another being infrequent. Practical examples of this nature do not spring to mind, however.

The most extreme form of strategy (c) is to use only the final state of each run in making estimates. In virtually all cases, however, it will be better to use at least a few states from the end of the run, unless calculating the functions of state we are interested in is very expensive.

For the typical problem I outline above, as well as many statistical problems, I feel that strategy (b) is most appropriate. As discussed earlier, the information from multiple runs is useful in determining when the runs have convergenced. In typical problems, it will be prudent to base this decision on as much information as possible. Note in this regard that though information from multiple runs is also available when using strategy (c), diagnostics for convergence based solely on the observed data *not* infallible, and accordingly it is generally not prudent to perform only short runs on the assumption that any convergence difficulties will be detectable in this way. In comparison with strategy (a), the multiple runs of strategy (b) may be somewhat less efficient, since one must discard an initial portion of all of them, but the increased assurance that the true equilibrium distribution has been reached seems worth the cost for most applications.

Gelman and Rubin (8:1993) provide a detailed recipe for estimation using strategy (b). They focus, however, on relatively easy problems, for which a good initial distribution can be found. In more difficult problems, more informal methods may be appropriate. Simply comparing the estimates obtained from different runs together with their estimated standard errors can reveal convergence problems. It is worth emphasizing again, however, that it is possible for such estimates from different runs to be consistent even when all the runs are far from equilibrium.

What should we do when using strategy (b) if the diagnostics do not confirm that equilibrium has been reached? (One may, of course, simply extend each run further, hoping the problem will go away, but an end must be declared sometime.) Two situations are possible:

1) Some or all runs have not reached regions typical of the true equilibrium distribution, though they may have reached "local" equilibrium.

2) The runs have reached the equilibrium distribution, but the rate at which they move about this distribution is so slow that this fact cannot be diagnosed. (Each run has sampled from only a limited region, and hence appears different from the others.)

Situations (1) and (2) generally cannot be distinguished from the data. Accordingly, good results of known reliability cannot be expected, but an estimate could be obtained by discarding the portion of each run prior to when local equilibrium appears to have been reached, and then pooling the remaining samples. The result will be a valid estimate for situation (2), but of course one will not know this to be the case. In many statistical contexts, this procedure would not be considered acceptable (though perhaps there will be no alternative), but expectations are realistically lower in some artificial intelligence applications.

6.4 Parallel implementation

Many of the applications of Markov chain Monte Carlo methods are of the open-ended sort in which arbitrarily large amounts of computer power can be usefully employed. It is thus natural to explore the possibility of using multiple processors in parallel for these problems. Unfortunately, sampling using Markov chains appears to be inherently serial in nature — in general, one must have the state at time t before the state at time $t+1$ can be computed.

Typically, however, there is substantial potential for using parallelism in ways that skirt this basic fact. I will outline some such ways here, but only briefly, and with few references, as much of the work in this area is specific to particular problems or machine architectures.

Easy ways to exploit parallelism. Two areas in which parallelism can often be easily employed are in performing multiple simulations of independent Markov chains, and in computing the transitions in a single Markov chain.

As was discussed in Section 6.3, multiple runs are often advisable in order to verify that the number of iterations used is sufficient to reach the equilibrium distribution of the Markov chain. In Section 6.2, we saw that the free energy difference between dissimilar systems may have to be estimated as the sum of free energy differences between members of a series of intermediate systems. In both cases, the multiple runs required can trivially be performed simultaneously on multiple processors, though in the case of free energy estimation, one would have to forgo the potential gain from using the final state of the simulation for one intermediate system as the initial state for the simulation of the next.

Gains from parallelism may also be possible when calculating the energy of a state, needed to make accept/reject decisions with the Metropolis algorithm, or the derivative of the energy, needed for dynamical simulations. In Bayesian inference problems, the energy calculation will often be dominated by the calculation of the likelihood (equation (2.22)), in which the probability of each training case enters independently, allowing these contributions to be computed in parallel. The energy function for the Lennard-Jones system (equation (2.57)) is also the sum of a number of independent terms (one for each pair of molecules), providing considerable scope for parallelism.

The conditional distributions needed for Gibbs sampling in a system of discrete variables are often computed from the relative joint probabilities of the states in which the variable in question takes on its various possible values. This provides scope for parallelism regardless of whether the computation of the joint probability of a state can itself be parallelized.

Parallel updates in systems with local interactions. As seen in many of the examples in this review, the state of a system is often decomposed into components, and the transition matrix into a product of base transitions, one for each component. When these base transitions are locally determined, it is possible to perform large subsets of them in parallel, as the new value for one component in the subset does not effect the transitions for other components in the subset.

The 2D Ising model provides a simple example, which is discussed by Heermann and Burkitt (1:1992). Imagine the array of spins as a checkerboard, with "black" and "white" spins alternating both horizontally and vertically. When the system is simulated with either Gibbs sampling or the Metropolis algorithm, the distribution from which the new value of a spin is chosen depends only on the values of its four neighbors, which are of the opposite colour. It is therefore possible to pick new values for *all* the spins of one colour in parallel, with exactly the same result as if they were picked in sequence. The spins of the opposite colour can then be chosen in the same fashion. In this way, a full transition of the Markov chain can be performed in time independent of the size of the system, assuming sufficient processors are available. This approach to parallelism is appropriate when many processors with appropriate communication links are available.

A more coarse-grained approach, which also exploits the locality of the problem, is to divide the 2D array into large contiguous areas, one per processor. Transitions within each area can be computed independently of other areas, except for the need to communicate the states of spins at the boundaries between areas. These boundary spins make up a relatively small fraction of the total.

Gibbs sampling for a belief network may also permit some degree of parallelism, the amount depending on how sparsely connected the network is. Pearl (4:1987, 2:1988, Section 4.4.3) describes a distributed algorithm for finding and exploiting this parallelism. Each variable in the network is assumed to be associated with a separate processor, which communicates directly only with the processors for "nearby" variables, as determined by the connections present in the network. Using such local communication, it is possible for the processors to coordinate in such a way that a number of processors can simultaneously select new values for the variables they control, while being assured that the other variables on which this selection is based are not being updated simultaneously.

7. Directions for Research

Despite the long history of Markov chain Monte Carlo methods, the field has by no means become static. Indeed, there is currently a resurgence of activity, spurred by new applications in statistics, artificial intelligence, and quantum field theory. Here, I will touch on some areas of research that I believe are most promising, most interesting, or most challenging. In the course of this review, the reader will undoubtably have noticed various other areas in which work remains to be done as well.

7.1 Improvements in the algorithms

Following their development in the 1950's, the basic algorithms were applied for a number of years with relatively few changes. Recently, a new vitality has been evident, illustrated by the development of new methods for free energy estimation, the Swendsen-Wang algorithm for Ising systems, and the hybrid Monte Carlo method. However, there is still scope for improvement with respect to several fundamental aspects of the algorithms, some of which I will briefly indicate here.

Adapting parameters of the algorithms. A simple problem, which may superficially appear rather easy, is that of adapting the parameters of the algorithms to match the problem being solved. For example, it would be nice to adjust the proposal distribution in the Metropolis algorithm so as to make moves that are as large as is possible while maintaining a reasonable acceptance rate. For the hybrid Monte Carlo algorithm, this would take the form of adapting the stepsize for the dynamical simulations.

It is easy enough to keep track of the acceptance rate for recent moves, and it would not be hard to use this information to adjust the stepsize so that it converges to near the optimal value. Unfortunately, doing so undermines the proofs of correctness for the algorithms. Indeed, situations where such adaptation results in incorrect sampling can easily be constructed.

Currently, we have several choices, none of which are entirely satisfactory. We can try to guess a good stepsize *a priori*, but a bad choice might be very inefficient. We can randomly choose a new stepsize from some fixed distribution at every move, and thereby be sure of sometimes using a good stepsize, at the cost of often using a bad one. We can do a preliminary run to find a good stepsize, and then fix the stepsize at that value for the final run, but this works only if the same stepsize is good throughout the course of the simulation. Finally, we can adapt the stepsize, and hope that the error introduced by this adaptation is not too large.

How else might this problem be tackled? One possibility for the hybrid Monte Carlo method is to choose the stepsize at random, but terminate trajectories with bad stepsizes early, before they have cost much in computation time. As I mention in (Neal, 5:1993), this can be valid if done properly. Perhaps other such loopholes exist.

Choosing a coordinate system. The performance of many of the algorithms discussed in this review can depend strongly on the coordinate system used. For example, Gibbs sampling converges in one iteration if all the components of the state are independent, but can take much longer, for the same distribution, if the state is instead represented using components that are highly correlated, as may be the case if the coordinate system is rotated. The Gibbs sampler is, however, invariant under translation (except, inevitably, for the effect of the initial state distribution). For the dynamical algorithms, efficiency (with the optimal stepsize) is greatest when the second derivative of the energy is of similar magnitude in all directions. This property is independent of translation and rotation, but changes if the coordinates are rescaled by different amounts in different directions.

For some problems, an appropriate coordinate system can be chosen *a priori*. For example, in a molecular simulation where each molecule interacts with only a few others in its vicinity, it seems reasonable to use the positions and momenta of the molecules as coordinates, since many of these will be essentially independent. In other problems, the choice of coordinate system is more difficult. For example, the weights on different connections in a multi-layer perceptron can be highly dependent, raising doubts as to whether the obvious coordinate system is in this case the best. The best coordinate system may also be different for different parts of the state space.

Bennett (5:1975) describes a way of finding a good coordinate system for dynamical simulations (by the equivalent means of finding a good form for the kinetic energy). His method can be applied adaptively, as the simulation runs, but one must then deal with the general problems of adaptation mentioned above. One potential source of ideas for finding a good coordinate system is the literature on optimization, in which similar problems are addressed. Skilling (9:1992) has pursued this approach with respect to conjugate gradient methods; the stochastic multigrid methods reviewed by Goodman and Sokal (4:1989) can also be viewed in this light.

Eliminating random walks. We have seen that the hybrid Monte Carlo method, used with long trajectories, avoids much of the random walk behaviour characteristic of simpler versions of the Metropolis algorithm. Consequently, with N energy gradient evaluations, the hybrid Monte Carlo algorithm can move a distance proportional to N, while a random walk algorithm will likely have traversed only a distance proportional to $\sqrt{N}$.

It is natural to ask whether other ways of achieving this result might be possible. For example, is the introduction of "momentum" variables essential? A method that could be applied to systems with discrete variables would be particularly interesting, as there the hybrid Monte Carlo method is not applicable.

Is there a better approach? There are at least two respects in which any Monte Carlo method based on Markov chain sampling will be sub-optimal.

From the point of view of search, basing the next state on only the current state, as a Markov chain does, can clearly lead to a bad choice, such as one of the states already visited. The inefficiencies of the random walks that many of the Markov chain methods engage in can be seen as one manifestation of this problem. It seems quite conceivable that a better way of choosing states could be found.

The Monte Carlo estimation procedure itself is also fundamentally sub-optimal. According to Bayesian theory, the conclusion one draws from given information, such as the results of evaluating a function with various arguments, should not depend on *why* that information was gathered, but only on the information itself. The Monte Carlo estimation formula of equation (2) violates this dictum, as it depends crucially on the states used having been chosen at random.

Can we therefore hope to find some entirely different method that can handle, better, the same problems that can be tackled by Markov chain Monte methods? I believe so, but the problem is not trivial. An "improved" search procedure will not be beneficial if it requires much more time to choose a state, and this state turns out to usually be only slightly better than one chosen at random. Replacing frequentist Monte Carlo estimates with Bayesian inferences runs the risk of an infinite regress, if the resulting problem of Bayesian computation is as difficult as that which we set out to solve originally.

The potential benefits of a new approach are substantial, however. As well as perhaps producing a faster algorithm, a careful Bayesian analysis might also reveal exactly what assumptions we are implicitly making when we decide that some Markov chain Monte Carlo method will likely be able to solve a given problem, even though we lack a rigorous proof of rapid convergence, or when we decide that a particular Monte Carlo run has actually produced a reasonable estimate.

7.2 Scope for applications

Markov chain Monte Carlo methods have long been a standard tool in statistical physics. In recent years, applications to quantum chromodynamics and to the simulation of proteins and nucleic acids have been particularly prominent. Markov chain methods are now also becoming standard tools for performing probabilistic inference in statistical applications, and they should be even more relevant for the complex probabilistic models that are, in my opinion, the most promising approach to tackling many problems in artificial intelligence. More speculatively, it is interesting to ask whether any operations in the brain, or other biological systems, can usefully be viewed as Monte Carlo implementations of probabilistic inference.

Applications to statistical inference. Gibbs sampling has proved to be a flexible tool for Bayesian inference. For some problems, it has allowed realistic models to be used when these would previously not have been feasible, or could have been made feasible only by the expenditure of considerable effort to develop specialized analytical or numerical techniques. One promising avenue for future research in this area is to apply the other Markov chain sampling methods described in this review to statistical problems. Another is to further exploit the capabilities provided by the Markov chain methods by developing and applying new models that would previously have been dismissed as impractical. One challenge is to find ways of expressing prior beliefs in complex domains.

To illustrate the possibilities in this respect, consider the common practice of restricting attention to models whose number of parameters is small in relation to the amount of data available. While this is certainly necessary if good results are to be obtained with procedures such as maximum likelihood, the practice makes no sense from a Bayesian perspective — the choice of which model or models to consider is a matter of *prior* belief, and should not depend on how much data is later collected. There are two difficulties with using a complex model to analyse a small data set, however. First, one must be careful to use a prior for the parameters of the model that accurately reflects prior beliefs, including in some form the same biases that might otherwise have lead one use a simple model of a particular type. Second, one must be able to deal computationally with a complex model when the amount of data is not sufficient to produce a posterior that is approximately Gaussian, a problem for which Monte Carlo methods are relevant. If these two difficulties can be addressed, use of a complex model to analyse a small data set would potentially be able to bring out the maximum amount of information in the data, and, perhaps more importantly, fully indicate the degree of uncertainty in the conclusions drawn.

Applications to artificial intelligence. The most straightforward applications of Markov chain methods in artificial intelligence are to inference for a particular situation using a specified probabilistic model. For models expressed using belief networks, based on information obtained from experts, Markov chain methods have been reasonably well developed, at least as a research area.

Probabilistic models that are formally and computationally similar to these may also be able to provide a sound basis for more speculative work in artificial intelligence. The "Copycat" system of Mitchell and Hofstadter (2:1990) provides an example of work that might be improved by a probabilistic formulation. Copycat is intended to model high-level perception and analogy-making, within an idealized microworld. When presented with a situation requiring interpretation, Copycat attempts to build perceptual structures that bring out the deep similarities between superficially dissimilar objects, allowing it to then make an analogy. These structures are built by the interaction of competing and cooperating local agents, whose actions are in part stochastic. While its design is based on certain general principles, the specifics of Copycat's implementation are rather *ad hoc*, making it difficult to determine which aspects of the system are essential, and to what extent the same ideas would be expected to work in other domains. It seems possible that a more transparent system of similar capabilities might be built in which the situation presented is modeled using a belief network. Higher levels in this network would contain latent variables representing the deep aspects of the situation that are common to all its components. Lower levels would express how these aspects are realized in each object. Since such a network would be highly interconnected, inference would likely require the use of Markov chain methods.

A further advantage of representing knowledge in probabilistic form is that we can then envision learning such representations from empirical data, using Bayesian, maximum likelihood, or other statistical procedures. In particular, models with latent variables where relationships are expressed via belief networks appear capable of representing the structure of many domains, from vision to natural language. I believe that Bayesian inference using Monte Carlo methods is a promising approach to learning such structures.

A model for biological processes? Finally, we can ask whether it might be useful to view neural or other biological systems as implementations of probabilistic inference using Markov chain Monte Carlo methods.

In some respects, this seems quite plausible. Organisms certainly need some way of coping with an uncertain environment if they are to survive and reproduce, and the world is certainly complex enough that models for which Markov chain methods would be appropriate might be necessary. It is easy to envision how a biological system could use replicated sub-systems to produce the moderate number of sample points required to form a Monte Carlo estimate. (This is probably more reasonable than using time averages.) Replication is an ubiquitous feature of biological systems, and might be necessary in any case to ensure fault-tolerance.

In other respects, however, envisioning biological implementations of Markov chain methods is more difficult. The generation of random variates required for Gibbs sampling and the consideration and possible rejection of candidate moves for the Metropolis algorithm both require types of computation that are perhaps far-fetched in a biological context. Methods based on stochastic dynamics seem more natural, however. The Langevin method seems particularly promising, as it does not require the representation of momentum variables.

A conceptual problem with viewing biological learning in Bayesian terms is that the Bayesian posterior distribution is based on the entirety of a given training set. Even if one assumes that a set of training cases is stored in episodic memory, it seems implausible that they are all attended to simultaneously during learning. It appears necessary to instead consider some form of "on-line" learning, in which the learning procedure sees only a single training case at each step. One idea for reconciling this with the requirements of Bayesian inference by Markov chain sampling is to make the amount of noise in the gradient of the log likelihood that results from using only a single training case match the amount that would need to be introduced in any case to supply the stochastic element in the dynamics.

8. Annotated Bibliography

Works are referred to by author, section, and year; for example, (Metropolis, *et al*, 4:1953) is found in Section 4 below. I have included a number of works of interest that are not referenced in the text. I have had to refrain from including many of the large number of papers that merely detail particular applications in statistical physics and in statistical inference.

1. General Works on Monte Carlo Methods

Included here are reviews, monographs, texts, compilations, and historical accounts concerning Monte Carlo methods, particularly those based on Markov chains.

Abraham, F. F. (1986) "Computational statistical mechanics: Methodology, applications and supercomputing", *Advances in Physics*, vol. 35, pp. 1-111.

Reviews the Metropolis and dynamical methods of simulation, and contains extensive material on their applications in statistical physics.

Besag, J. and Green, P. J. (1993) "Spatial statistics and Bayesian computation" (with discussion), *Journal of the Royal Statistical Society B*, vol. 55, pp. 25-37 (discussion, pp. 53-102).

Reviews progress in Markov chain Monte Carlo methods, including discussion of antithetic methods, and the use of auxiliary variables, as in the Swendsen-Wang algorithm.

Binder, K. (1979) "Introduction: Theory and 'technical' aspects of Monte Carlo simulations", in K. Binder (editor) *Monte Carlo Methods in Statistical Physics*, Berlin: Springer-Verlag.

A general introduction to the use of Monte Carlo methods based on the the Metropolis algorithm in statistical physics. Discussions of the "rejectionless" method and of error estimation are of particular interest.

Ciccotti, G., Frenkel, D., McDonald, I. R. (1987) *Simulation of Liquids and Solids: Molecular Dynamics and Monte Carlo Methods in Statistical Mechanics*, Amsterdam: North-Holland.

An annotated collection of reprints of original papers, including (Metropolis, *et al*, 4:1953), (Bennett, 7:1976), and (Torrie and Valleau, 7:1977).

Gelman, A. (1993) "Iterative and non-iterative simulation algorithms", to appear in *Computing Science and Statistics: Proceedings of the 24th Symposium on the Interface.*

Reviews rejection sampling, importance sampling, the Metropolis algorithm, and Gibbs sampling, pointing out how the first two as well as the latter two can be viewed as iterative in nature.

Gordon, R. (1980) "Monte Carlo methods for cooperative ISING models", in G. Karreman (editor) *Cooperative Phenomena in Biology*, New York: Pergamon Press.

Discusses the Metropolis algorithm as applied to discrete systems. Several variations are described, including the "rejectionless" method, which avoids the problem of long runs of the same state. Describes applications in chemistry and biology.

Hammersley, J. M. and Handscomb, D. C. (1964) *Monte Carlo Methods*, London: Chapman and Hall.

A classic text covering Monte Carlo methods, including the Metropolis algorithm.

Heermann, D. W. (1990) *Computer Simulation Methods in Theoretical Physics*, 2nd edition, Berlin:

Springer-Verlag.

An introductory treatment of both molecular dynamics and Monte Carlo methods, and their applications in statistical physics.

Heermann, D. W. and Burkitt, A. N. (1992) "Parallel algorithms for statistical physics problems", in K. Binder (editor) *The Monte Carlo Method in Condensed Matter Physics*, Berlin: Springer-Verlag.

Reviews methods for exploiting parallel hardware in statistical physics computations, primarily those based on Markov Chain Monte Carlo methods, both those of the Metropolis type, and those based on dynamical simulation.

Kalos, M. H. and Whitlock, P. A. (1986) *Monte Carlo Methods, Volume I: Basics*, New York: John Wiley.

An introduction to Monte Carlo sampling methods, including the Metropolis algorithm, and their uses in numerical integration and the solution of integral equations.

Kennedy, A. D. (1990) "The theory of hybrid stochastic algorithms", in P. H. Damgaard, *et al* (editors) *Probabilistic Methods in Quantum Field Theory and Quantum Gravity*, New York: Plenum Press.

Reviews the basics of Monte Carlo integration using Markov chains, and of the Metropolis algorithm in particular, and then discusses "hybrid" algorithms that employ dynamical methods, reviewing attempts to analyse their performance for the simple case of a multivariate Gaussian distribution.

Ripley, B. D. (1987) *Stochastic Simulation*, New York: John Wiley.

Discusses simulation methods, including the Metropolis algorithm. Has material on random number generation, methods for reducing variance, and assessing error in estimates.

Sheykhet I. I. and Simkin, B. Y. (1990) "Monte Carlo method in the theory of solutions", *Computer Physics Reports*, vol. 12, pp. 67-133.

Reviews the Metropolis Monte Carlo algorithm and its variants, including the "force bias" and "smart Monte Carlo" methods, techniques for free energy estimation, and the problems of applying these methods to the simulation of liquids.

Sinclair, A. (1993) *Algorithms for Random Generation and Counting: A Markov Chain Approach*, Boston: Birkhäuser.

Develops randomized algorithms for the approximate counting of large sets, using methods related to those used for free energy estimation, implemented using sampling via Markov chains. Covers the same work as (Sinclair and Jerrum, 7:1989) and (Jerrum and Sinclair, 4:1989). Includes an appendix reviewing more recent work.

Smith, A. F. M. and Roberts, G. O. (1993) "Bayesian computation via the Gibbs sampler and related Markov chain Monte Carlo methods" (with discussion), *Journal of the Royal Statistical Society B*, vol. 55, pp. 3-23 (discussion, pp. 53-102).

Discusses recent developments in Gibbs sampling and the Metropolis algorithm, and their application to statistical problems.

Swendsen, R. H., Wang, J-S., and Ferrenberg, A. M. (1992) "New Monte Carlo methods for improved efficiency of computer simulations in statistical mechanics", in K. Binder (editor) *The Monte Carlo Method in Condensed Matter Physics*, Berlin: Springer-Verlag.

Reviews several recent improvements in Metropolis type Monte Carlo methods, including the

Swendsen-Wang algorithm and the replica Monte Carlo method, and discusses "histogram" techniques for making use of data from one or several simulations.

Tierney, L. (1991a) "Markov chains for exploring posterior distributions", Technical Report No. 560, School of Statistics, University of Minnesota.

Presents methods related to the Metropolis algorithm and Gibbs sampler, reviews applicable results from the theory of Markov Chains, and discusses implementation issues, all from the viewpoint of applying Markov chain sampling to Bayesian inference for statistical problems.

Tierney, L. (1991b) "Exploring posterior distributions using Markov chains", in E. M. Keramidas (editor), *Computing Science and Statistics: Proceedings of the 23rd Symposium on the Interface*, pp. 563-570, Interface Foundation.

Similar to (Tierney, 1:1991a), but shorter.

Toussaint, D. (1989) "Introduction to algorithms for Monte Carlo simulations and their application to QCD", *Computer Physics Communications*, vol. 56, pp. 69-92.

Reviews the Metropolis, molecular dynamics, and hybrid Monte Carlo methods, and discusses their application to quantum chromodynamics.

Valleau, J. P. and Torrie, G. M. (1977) "A guide to Monte Carlo for statistical mechanics: 2. Byways", in B. J. Berne (editor) *Statistical Mechanics, Part A: Equilibrium Techniques (Modern Theoretical Chemistry, Volume 5)*, New York: Plenum Press.

Includes a review of methods for estimating free energy differences, and of applications of Monte Carlo methods to quantum mechanical calculations.

Valleau, J. P. and Whittington, S. G. (1977) "A guide to Monte Carlo for statistical mechanics: 1. Highways", in B. J. Berne (editor) *Statistical Mechanics, Part A: Equilibrium Techniques (Modern Theoretical Chemistry, Volume 5)*, New York: Plenum Press.

An introduction to the Metropolis Monte Carlo method, including a discussion of the best choice of transition matrix and of rate of convergence.

Wood, W. W. (1985) "Early history of computer simulations in statistical mechanics", in *Molecular-Dynamics Simulation of Statistical-Mechanical Systems (Proceedings of the International School of Physics "Enrico Fermi", Course 97)*, Amsterdam: North-Holland.

Recounts the early history of attempts to simulate statistical mechanical systems using the Monte Carlo and molecular dynamics methods.

2. Probabilistic Inference for Artificial Intelligence

I list here works on applications of probabilistic inference to problems that may be regarded as being in the domain of artificial intelligence. See also (Geman and Geman, 4:1984) and (Pearl, 4:1987).

Ackley, D. H., Hinton, G. E., and Sejnowski, T. J. (1985) A learning algorithm for Boltzmann machines, *Cognitive Science*, vol. 9, pp. 147-169.

Uses Gibbs sampling with simulated annealing to sample probability distributions represented by a neural network with symmetric connections. Maximum likelihood estimates for the weights on connections are found using "positive" and "negative" phase simulations.

Anderson, J. R. and Matessa, J. R. (1992) "Explorations of an incremental, Bayesian algorithm for categorization", *Machine Learning*, vol. 9, pp. 275-308.

Describes an incremental algorithm for latent class analysis based on Bayesian methods, and explores its uses in machine learning and in psychological modeling.

Buntine, W. (1992) "Learning classification trees", *Statistics and Computing*, vol. 2, pp. 63-73.

Presents Bayesian methods for inferring classification trees from data. Includes a Bayesian derivation of a common tree growing heuristic, and computational methods for averaging over all possible prunings of a tree, as well as over different tree structures.

Buntine, W. L. and Weigend, A. S. (1991) "Bayesian back-propagation", *Complex Systems*, vol. 5, pp. 603-643.

Discusses Bayesian forms of backpropagation learning for multi-layer perceptron networks using the Gaussian approximation method.

Charniak, E. (1991) "Bayesian networks without tears", *AI Magazine*, vol. 12, pp.50-63.

A tutorial introduction to the use of belief networks to express probabilistic knowledge.

Cheeseman, P. (1988) "An inquiry into computer understanding" (with discussion), *Computational Intelligence*, vol. 4, pp. 57-142. Further discussion appears in vol. 6, pp. 179-192.

Advocates the use of probability and Bayesian statistical inference for uncertain reasoning and learning in artificial intelligence. Includes numerous replies by other workers of varied opinions.

Cheeseman, P., Kelly, J., Self, M., Stutz, J., Taylor, W., and Freeman, D. (1988) AutoClass: A Bayesian classification system, *Proceedings of the Fifth International Conference on Machine Learning*.

Uses latent class models in a machine learning context, estimating their parameters by maximum penalized likelihood (which the authors view as a "maximum *a posteriori* probability" approximation to Bayesian inference).

Dagum, P. and Luby, M. (1993) "Approximating probabilistic inference in Bayesian belief networks is NP-hard", *Artificial Intelligence*, vol. 60, pp. 141-153.

Shows, assuming certain widely-believed conjectures in complexity theory, that one cannot, in the worst case, approximate the conditional distribution for one variable in a belief network conditioned on a value for another variable in an amount of time that is bounded by a polynomial in the size of the network and the reciprocal of the approximation error. This still applies if one allows randomized algorithms that deliver such an approximation with high probability.

Duda, R. O. and Hart, P. E. (1973) *Pattern Classification and Scene Analysis*, New York: John Wiley.

A frequently referenced work that presents the basics of decision theory for classification models, and discusses learning such models by maximum likelihood and Bayesian methods, in supervised and unsupervised contexts.

Hanson, R., Stutz, J., and Cheeseman, P. (1991) "Bayesian classification with correlation and inheritance", presented at the 12th International Joint Conference on Artificial Intelligence, Sydney, Australia, August 1991.

Applies the Gaussian approximation method to Bayesian inference for latent class models.

Lauritzen, S. L. and Spiegelhalter, D. J. (1988) Local computations with probabilities on graphical structures and their application to expert systems (with discussion), *Journal of the Royal Statistical Society B*, vol. 50, pp. 157-224.

Presents exact, deterministic methods of computing conditional probabilities in belief networks that are feasible for networks with sparse connectivity.

Lavine, M. and West, M. (1992) "A Bayesian method for classification and discrimination", *Canadian Journal of Statistics*, vol. 20, pp. 451-461.

Describes how Gibbs sampling may be used to perform Bayesian inference for normal-mixture models, such as in latent class modeling of real-valued data (though this application is not the principle motivation of the authors).

MacKay, D. J. C. (1991) *Bayesian Methods for Adaptive Models*, Ph.D thesis, California Institute of Technology.

Develops Bayesian methods for interpolation and classification, particularly using multi-layer perceptrons as models. Emphasizes the Bayesian approach to selecting which model (e.g. which network architecture) best fits the data.

MacKay, D. J. C. (1992a) "Bayesian interpolation", *Neural Computation*, vol. 4, pp. 415-447.

Illustrates Bayesian inference in the context of interpolation, with emphasis on the Bayesian approach to model selection.

MacKay, D. J. C. (1992b) "A practical Bayesian framework for backpropagation networks", *Neural Computation*, vol. 4, pp. 448-472.

Applies the Gaussian approximation method to Bayesian inference for neural networks, including the comparison of different network architectures.

MacKay, D. J. C. (1993) "Bayesian non-linear modeling for the energy prediction competition", preprint.

Applies the techniques of (Mackay, 2:1992b) to two prediction problems relating to energy usage in buildings.

Mitchell, M. and Hofstadter, D. R. (1990) "The emergence of understanding in a computer model of concepts and analogy-making", *Physica D*, vol. 42, pp. 322-334.

Presents a computational model for high-level perception and analogy in which local agents compete and cooperate in constructing a representation of a situation. The model is implemented in an idealized microworld.

Neal, R. M. (1992a) "Bayesian training of backpropagation networks by the hybrid Monte Carlo method", Technical Report CRG-TR-02-1, Dept. of Computer Science, University of Toronto.

Applies the hybrid Monte Carlo algorithm with simulated annealing to problems of inference and prediction in a Bayesian formulation of neural network learning.

Neal, R. M. (1992b) "Connectionist learning of belief networks", *Artificial Intelligence*, vol. 56, pp. 71-113.

Develops maximum likelihood learning procedures using Gibbs sampling for two types of belief network, and compares the learning performance of these networks with that of Boltzmann machines.

Neal, R. M. (1992c) "Bayesian mixture modeling", in C. R. Smith, G. J. Erickson, and P. O. Neudorfer (editors) *Maximum Entropy and Bayesian Methods: Proceedings of the 11th International Workshop on Maximum Entropy and Bayesian Methods of Statistical Analysis, Seattle, 1991*, Dordrecht: Kluwer Academic Publishers.

Applies Gibbs sampling to Bayesian inference for latent class models, including models with a countably infinite number of latent classes.

Neal, R. M. (1993a) "Bayesian learning via stochastic dynamics", in C. L. Giles, S. J. Hanson, and J. D. Cowan (editors) *Advances in Neural Information Processing Systems 5*, pp. 475-482, San Mateo, California: Morgan Kaufmann.

Compares stochastic dynamics and hybrid Monte Carlo implementations of Bayesian learning for multi-layer perceptrons with traditional learning methods. Shows that the Bayesian methods can find good values for "weight decay" constants without the need for a "validation" set.

Neal, R. M. (1993b) "Priors for infinite networks", in preparation.

Shows how to define prior distributions over the weights in a neural network in such a fashion that as the number of hidden units goes to infinity the distribution over functions computed by the network reaches a sensible limit.

Oliver, R. M. and Smith, J. Q., editors (1990) *Influence Diagrams, Belief Nets and Decision Analysis* (proceedings of a conference entitled 'Influence diagrams for decision analysis, inference, and prediction', Berkeley, USA, 1988), Chichester, England: John Wiley.

A collection of papers on methods for expressing probability distributions using belief networks, and the extension of this idea to networks with nodes representing decisions.

Pearl, J. (1988) *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, San Mateo, California: Morgan Kaufmann.

A wide-ranging book on the philosophy of probabilistic reasoning, formalisms for specifying probabilistic models, and algorithms for performing probabilistic inference. Contains extensive material on graphical structures, including "belief networks", that can represent a set of independence relations between variables, and, with the addition of numerical parameters, their joint distribution.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986) "Learning representations by back-propagating errors, *Nature*, vol. 323, pp. 533-536.

Describes the "backpropagation" algorithm for neural network learning, one of whose advantages is the ability to discover features in the input, represented by the values of "hidden units".

Rumelhart, D. E., McClelland, J. L., and the PDP Research Group (1986) *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Volume 1: Foundations*, Cambridge, Massachusetts: MIT Press.

Describes influential early work in the "parallel distributed processing" framework, now more commonly referred to as "connectionism" or "neural networks". Includes papers on "backpropagation" and "Boltzmann machines".

Seung, H. S., Sompolinsky, H., and Tishby, N. (1992) "Statistical mechanics of learning from examples", *Physical Review A*, vol. 45, pp. 6056-6091.

Investigates the generalization properties of neural networks using the analytical tools of statistical mechanics, supplemented by numerical simulations using the Metropolis algorithm.

Spiegelhalter, D. J. and Lauritzen, S. L. (1990) "Sequential updating of conditional probabilities on directed graphical structures", *Networks*, vol. 20, pp. 579-605.

Discusses Bayesian learning of the parameters in a belief network. The methods discussed are exact when the priors satisfy certain independence requirements and the training cases are completely observed. Approximations are proposed to handle more general problems.

Szeliski, R. (1989) *Bayesian Modeling of Uncertainty in Low-level Vision*, Boston: Kluwer.

Applies Bayesian methods to modeling surfaces that have been observed using noisy sensors, using prior distributions favouring some degree of smoothness. Implementation of such models using Gibbs sampling is discussed, including use of the multigrid method. The Bayesian approach to estimating the degree of smoothness is discussed as well.

Thodberg, H. H. (1993) "Ace of Bayes: Application of neural networks with pruning", submitted to *IEEE Transactions on Neural Networks.*

Reviews and extends the method for Bayesian learning of neural networks of MacKay (2:1992b), and applies it to a practical problem.

Watkin, T. L. H., Rau, A., and Biehl, M. (1993) "The statistical mechanics of learning a rule", to appear in *Reviews of Modern Physics.*

Reviews work on applying the formalism of statistical mechanics to problems of learning.

3. Theory of Markov Chains

A number of reviews and monographs in Section 1 also cover the theory of Markov chains (e.g. Kennedy, 1:1990, Sinclair, 1:1993, Tierney, 1:1991a). See also (Sinclair and Jerrum, 7:1989).

Diaconis, P. and Stroock, D. (1991) "Geometric bounds for eigenvalues of Markov chains", *Annals of Applied Probability*, vol. 1, pp.36-61.

Develops bounds on the second largest eigenvalue of the transition matrix of a reversible Markov chain, and relates these to bounds on the convergence rate of the chain.

Feller, W. (1968) *An Introduction to Probability Theory and Its Applications*, Third Edition, New York: John Wiley.

Includes a section that is a classic reference on Markov chains.

Fill, J. A. (1991) "Eigenvalue bounds on convergence to stationarity for nonreversible Markov chains, with an application to the exclusion process", *Annals of Applied Probability*, vol. 1, pp. 62-87.

Discusses, extends, and applies a number of bounds on the convergence rate of Markov chains, including non-reversible chains.

Iosifescu, M. (1980) *Finite Markov Processes and Their Applications*, Chichester: John Wiley.

A fairly readable treatment of finite Markov chains, including their analysis using eigenvalues of the transition matrix, the "fundamental matrix", and "ergodic coefficients".

Kemeny, J. G. and Snell, J. L. (1960) *Finite Markov Chains*, (reprinted, 1976), New York: Springer-Verlag.

A readable exposition of the theory of finite, homogeneous Markov chains.

Lawler, G. F. and Sokal, A. D. (1988) "Bounds on the L^2 spectrum for Markov chains and Markov processes: A generalization of Cheeger's inequality", *Transactions of the American Mathematical Society*, vol. 309, pp. 557-580.

Gives bounds on the second largest eigenvalue of a Markov chain transition matrix, for both reversible and non-reversible chains, using a concept related to the "conductance" of Sinclair and Jerrum (7:1989).

Mihail, M. (1989) "Conductance and convergence of Markov chains — A combinatorial treatment of expanders", *30th Annual Symposium on Foundations of Computer Science, 1989*, IEEE.

Gives bounds on the convergence rates of Markov chains in terms of "conductance" that apply to non-reversible chains. The results are proved without reference to eigenvalues.

4. The Metropolis and Gibbs Sampling Algorithms

These methods and their applications are also discussed in many of the reviews of Section 1, of which (Toussaint, 1:1989), (Sheykhet and Simkin, 1:1990), and (Tierney, 1:1991a,b) are recent accounts covering a variety of viewpoints.

Adler, S. L. (1981) "Over-relaxation method for the Monte Carlo evaluation of the partition function for multiquadratic actions", *Physical Review D*, vol. 23, pp. 2901-2904.

Describes a Markov chain Monte Carlo algorithm inspired by "over-relaxed" optimization methods that is applicable to systems in which all the conditional distributions for one variable given values for the others are Gaussian. (Despite the title, the method is not directly aimed at evaluating the paritition function, but rather at sampling from the corresponding canonical distribution.)

Bélisle, C. J. P., Romeijn, H. E., and Smith, R. L. (1993) "Hit-and-run algorithms for generating multivariate distributions", *Mathematics of Operations Research*, vol. 18, pp. 255-266.

Describes an algorithm for sampling from a bounded distribution over Euclidean space using a Markov chain that chooses new states from those along a randomly-chosen direction. Shows that the Markov chain satisfies the detailed balance condition, and gives conditions for it to be ergodic.

Bhanot, G. and Kennedy, A. D. (1985) "Bosonic lattice gauge theory with noise", *Physics Letters B*, vol. 157, pp. 70-76.

Applies the method of Kennedy and Kulti (4:1985), showing how unbiased estimates can often be obtained via Taylor series expansions.

Boender, C. G. E., Caron, R. J., McDonald, J. F., Rinnooy Kan, A. H. G., Romeijn, H. E., Smith, R. L., Telgen, J., and Vorst, A. C. F. (1991) "Shake-and-bake algorithms for generating uniform points on the boundary of bounded polyhedra", *Operations Research*, vol. 39, pp. 945-954.

Applies what are in fact variations on the Metropolis algorithm to the problem of generating points uniformly distributed over the surface of a polyhedron.

Bortz, A. B., Kalos, M. H., and Lebowitz, J. L. (1975) "A new algorithm for Monte Carlo simulation of Ising spin systems", *Journal of Computational Physics*, vol. 17, pp. 10-18.

Describes a simulation method for dynamical or equilibrium studies in which time spent in generating rejected transitions is avoided by sampling directly from the distribution for the next accepted transition. The method is applied to the Ising system.

Cunningham, G. W. and Meijer, P. H. E. (1976) "A comparison of two Monte Carlo methods for computations in statistical mechanics", *Journal of Computational Physics*, vol. 20, pp. 50-63.

Compares the performance of the Metropolis algorithm using the standard Metropolis acceptance function of equation (4.18) with that using the Boltzmann acceptance function of equation (4.27).

Edwards, R. G. and Sokal, A. D. (1988) "Generalization of the Fortuin-Kasteleyn-Swendsen-Wang representation and Monte Carlo algorithm", *Physical Review D*, vol. 38, pp. 2009.

Generalizes the algorithm of Swendsen and Wang (4:1987) for Ising models to any model in which the unnormalized probability of a state can be expressed as the product of a number of bounded functions of the state. Auxiliary variables are introduced for each such factor, and the algorithm proceeds by alternately chosing new values for the auxiliary and for the original variables.

Goodman, J. and Sokal, A. D. (1989) "Multigrid Monte Carlo method. Conceptual foundations", *Physical Review D*, vol. 40, pp. 2035-2071.

Reviews Markov chain sampling methods in which updates are performed on both fine-grained and coarse-grained representations of the state.

Friedberg, R. and Cameron, J. E. (1970) "Test of the Monte Carlo method: Fast simulation of a small Ising lattice", *Journal of Chemical Physics*, vol. 52, pp. 6049-6058.

Investigates possible sources of error in the Monte Carlo simulation of a 4×4 Ising system using a vectorized version of the Metropolis algorithm.

Frigessi, A., di Stefano, P., Hwang, C-R., and Sheu, S-J. (1993) "Convergence rates of the Gibbs sampler, the Metropolis algorithm, and other single-site updating dynamics", *Journal of the Royal Statistical Society*, vol. 55, pp. 205-219.

Analyses the asymptotic rate of convergence of the Metropolis and Gibbs sampling algorithms, especially as applied to the Ising model.

Gelfand, A. E. and Smith, A. F. M. (1990) "Sampling-based approaches to calculating marginal densities", *Journal of the American Statistical Association*, vol. 85, pp. 398-409.

Discusses applications to Bayesian statistical inference of the Gibbs sampler and two other Monte Carlo algorithms. The methods are illustrated using a variety of models with standard parametric forms, most with small to moderate numbers of parameters.

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

Develops "Markov random fields" as image models, and shows how, for given observed data, Gibbs sampling can be used to generate a sample of images from the posterior distribution produced by the model, or, in conjunction with simulated annealing, to find the most probable image.

Greene, J. W. and Supowit, K. J. (1986) "Simulated annealing without rejected moves", *IEEE Transactions on Computer-Aided Design*, vol. 5, pp. 221-228.

Describes a simulation method in which time spent on rejected moves is avoided by sampling from the distribution for the next accepted move, along with an implementation appropriate when the total number of possible moves is relatively small.

Hastings, W. K. (1970) "Monte Carlo sampling methods using Markov chains and their applications", *Biometrika*, vol. 57, pp. 97-109.

Generalizes the Metropolis algorithm, and discusses its use for statistical problems. Includes discussion of error estimation, and of the use of a Markov chain method as an importance sampler.

Hills, S. E. and Smith, A. F. M. (1992) "Parameterization issues in Bayesian inference" (with discussion), in J. M. Bernardo, J. O. Berger, A. P. Dawid, and A. F. M. Smith (editors), *Bayesian Statistics 4*, pp. 227-246, Oxford University Press.

Discusses techniques for reparameterizing problems to aid computation, including reparameterizations that remove dependencies between variables that slow the convergence of Gibbs sampling.

Jerrum, M. and Sinclair, A. (1989) "Approximating the permanent", *SIAM Journal of Computing*, vol. 18, pp. 1149-1178.

Applies the technique for analysing the convergence rate of a Markov chain of (Sinclair and Jerrum, 7:1989) to show that a method based on the Metropolis algorithm can be used to approximate the "permanent" of a dense 0-1 matrix in polynomial time.

Kennedy, A. D. and Kuti, J. (1985) "Noise without noise: A new Monte Carlo Method", *Physical Review Letters*, vol. 54, pp. 2473-2476.

Describes an adaptation of the Metropolis algorithm that, in some cases, allows unbiased results to be obtained using only an unbiased estimate of the ratio of the probabilities of two states, rather than the exact value.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E. (1953) "Equation of state calculations by fast computing machines", *Journal of Chemical Physics*, vol. 21, pp. 1087-1092.

The classic paper on Monte Carlo sampling using Markov chains, introducing what is now known as the "Metropolis algorithm", and applying it to a problem in statistical physics.

Mezei, M. (1981) "On the selection of the particle to be perturbed in the Monte Carlo method", *Journal of Computational Physics*, vol. 39, pp. 128-136.

Proposes a form of the Metropolis method in which components of the state are updated in an order chosen randomly for each cycle, and compares it to updating the components in a fixed order, and to updating components chosen at random indepedently. The proof of ergodicity for the method (Theorem 1) is flawed — a system in which all states have equal probabilities provides a counterexample.

Pearl, J. (1987) "Evidential reasoning using stochastic simulation of causal models, *Artificial Intelligence*, vol. 32, pp. 245-257.

Gives a simulation method (which is, in fact, Gibbs sampling) for inference in "belief networks", used to express probabilistic knowledge in expert systems. A parallel version of the algorithm is described as well.

Peskun, P. H. (1973) "Optimum Monte-Carlo sampling using Markov chains", *Biometrika*, vol. 60, pp. 607-612.

Shows that in constructing Markov chains using the method of Hastings (4:1970), the standard Metropolis acceptance criterion is optimal from the point of view of maximizing the asymptotic precision of the estimates.

Peskun, P. H. (1981) "Guidelines for chosing the transition matrix in Monte Carlo methods using Markov chains", *Journal of Computational Physics*, vol. 40, pp. 327-344.

Discusses the choice of transition matrix for Markov chains constructed using Hastings' method in order to maximize asymptotic precision, proposes approximate guidelines, and gives some empirical results.

Ritter, C. and Tanner, M. A. (1992) "Facilitating the Gibbs sampler: The Gibbs stopper and the Griddy-Gibbs sampler", *Journal of the American Statistical Association*, vol. 87, pp. 861-868.

Proposes a method for diagnosing convergence of the Gibbs sampler, and a technique for doing approximate Gibbs sampling when the conditional distributions are difficult to sample from exactly.

Swendsen, R. H. and Wang, J-S. (1987) "Nonuniversal critical dynamics in Monte Carlo simulations", *Physical Review Letters*, vol. 58, pp. 86-88.

Presents a method of simulating Ising systems (and their generalizations) that exploits a mapping to a percolation model. The new method is much faster than standard single-spin methods near a phase transition.

Tanner, M. A. and Wong, W. H. (1987) "The calculation of posterior distributions by data augmentation" (with discussion), *Journal of the American Statistical Association*, vol. 82, pp. 528-550.

Describes an early method for simulating posterior distributions derived from partially observed data that is essentially a special case of Gibbs sampling.

Thomas, A., Spiegelhalter, D. J., and Gilks, W. R. (1992) "BUGS: A program to perform Bayesian inference using Gibbs sampling", in J. M. Bernardo, J. O. Berger, A. P. Dawid, and A. F. M. Smith (editors), *Bayesian Statistics 4*, pp. 837-842, Oxford University Press.

Describes a program for general Bayesian inference using Gibbs sampling in which the model is expressed using the belief network formalism.

Wood, W. W. and Parker, F. R. (1957) "Monte Carlo equation of state of molecules interacting with the Lennard-Jones potential. I. A supercritical isotherm at about twice the critical temperature", *Journal of Chemical Physics*, vol. 27, pp. 720-733.

Applies the Metropolis algorithm to the simulation of a system of Lennard-Jones molecules.

York, J. (1992) "Use of the Gibbs sampler in expert systems", *Artificial Intelligence*, vol. 56, pp. 115-130.

Discusses the application of Gibbs sampling to distributions defined by belief networks. Proposes several methods of dealing with networks where some states have very low, or zero, probability.

5. *The Dynamical and Hybrid Monte Carlo Methods*

These methods are also discussed in many of the reviews of Section 1, particularly (Toussaint, 1:1989) and (Kennedy, 1:1990).

Alder, B. J. and Wainwright, T. E. (1959) "Studies in molecular dynamics. I. General method", *Journal of Chemical Physics*, vol. 31, pp. 459-466.

A classic early paper on use of the molecular dynamics method in statistical physics.

Andersen, H. C. (1980) "Molecular dynamics simulations at constant pressure and/or temperature", *Journal of Chemical Physics*, vol. 72, pp. 2384-2393.

Shows how the molecular dynamics method can be extended to sample from other than the microcanonical ensemble. In particular, the total energy is allowed to vary by introducing simulated collisions in which the momentum of a particle is refreshed from the canonical distribution.

Bennett, C. H. (1975) "Mass tensor molecular dynamics", *Journal of Computational Physics*, vol. 19, pp. 267-279.

Discusses dynamical simulations in which the kinetic energy is given the form $\frac{1}{2}\mathbf{p}^T M^{-1}\mathbf{p}$, of which the usual $\frac{1}{2}\mathbf{p}^T\mathbf{p}$ is a special case. This effectively rescales the coordinates, and, for a well chosen M, can improve the speed at which state space is explored.

Berendsen, H. J. C. and van Gunsteren W. F. (1986) "Practical algorithms for dynamic simulation", in *Molecular-Dynamics Simulation of Statistical-Mechanical Systems (Proceedings of the*

International School of Physics "Enrico Fermi", Course 97), Amsterdam: North-Holland.

Discusses methods for integrating dynamical equations, concluding that the "leapfrog" method is often best, though other methods can be more accurate.

Creutz, M. (1988) "Global Monte Carlo algorithms for many-fermion systems", *Physical Review D*, vol. 38, pp. 1228-1238.

Investigates the properties of the hybrid Monte Carlo and Langevin Monte Carlo methods.

Creutz, M. and Gocksch, A. (1989) "Higher-order hybrid Monte Carlo algorithms", *Physical Review Letters*, vol. 63, pp. 9-12.

Develops integration methods accurate to arbitrarily high order that preserve phase space volume, and applies them to the hybrid Monte Carlo method.

Duane, S. and Kogut, J. B. (1985) "Hybrid stochastic differential equations applied to quantum chromodynamics", *Physical Review Letters*, vol. 55, pp. 2774-2777.

Applies a combination of the molecular dynamics and Langevin methods to calculations in quantum chromodynamics.

Duane, S., Kennedy, A. D., Pendleton, B. J., and Roweth, D. (1987) "Hybrid Monte Carlo", *Physics Letters B*, vol. 195, pp. 216-222.

The original paper showing how the dynamical and Metropolis methods can be combined to eliminate the bias due to the use of a finite stepsize in the dynamical simulations.

Earn, D. J. D. and Tremaine, S. (1992) "Exact numerical studies of Hamiltonian maps: Iterating without roundoff error", *Physica D*, vol. 56, pp. 1-22.

Discusses the advantages of simulating the dynamics of a Hamiltonian system with fixed-point arithmetic, which allows reversibility and phase space volume to be preserved exactly even when round-off error is accounted for.

Horowitz, A. M. (1991) "A generalized guided Monte Carlo algorithm", *Physics Letters B*, vol. 268, pp. 247-252.

Introduces a variation on the Hybrid Monte Carlo algorithm in which the momenta are not completely replaced after each dynamical trajectory.

Kennedy, A. D. and Pendleton, B. (1991) "Acceptances and autocorrelations in hybrid Monte Carlo", *Nuclear Physics B (Proc. Suppl.)*, vol. 20, pp. 118-121.

Gives a detailed theoretical analysis of the behaviour of the hybrid Monte Carlo algorithm applied to a multivariate Gaussian distribution.

Mackenzie, P. B. (1989) "An improved hybrid Monte Carlo method", *Physics Letters B*, vol. 226, pp. 369-371.

Points out the desirability of varying the trajectory length in the hybrid Monte Carlo algorithm in order to avoid situations where a fixed length matches the period of some oscillation in the system.

Neal, R. M. (1993) "An improved acceptance procedure for the hybrid Monte Carlo algorithm", to appear in the *Journal of Computational Physics.*

Presents a generalization of the hybrid Monte Carlo algorithm in which the acceptance probability is increased by considering transitions to be between windows of several states at the beginning and end of the trajectory, rather than between single states.

Nosé, S. (1984) "A unified formulation of the constant temperature molecular dynamics methods", *Journal of Chemical Physics*, vol. 81, pp. 511-519.

Describes a class of deterministic dynamical algorithms that sample from a system's canonical distribution by sampling from the microcanonical distribution for an extended system.

Rao, M., Pangali, C., and Berne, B. J. (1979) "On the force bias Monte Carlo simulation of water: methodology, optimization and comparison with molecular dynamics", *Molecular Physics*, vol. 37, pp.1773-1798.

Discusses a form of the Metropolis algorithm in which the candidate states proposed are offset from the current state according to a distribution biased in the direction of the force, as evaluated at the current state. This candidate state is then accepted or rejected according to a criterion that takes into account the non-symmetrical form of this proposal distribution.

Rossky, P. J., Doll, J. D., and Friedman, H. L. (1978) "Brownian dynamics as smart Monte Carlo simulation", *Journal of Chemical Physics*, vol. 69, pp. 4628-4633.

Uses Langevin dynamics to generate candidate states for the Metropolis algorithm, which are then accepted or rejected by a criterion that accounts for the non-symmetrical form of this proposal distribution.

Ruth, R. D. (1983) "A canonical integration technique", *IEEE Transactions on Nuclear Science*, vol. NS-30, pp. 2669-2671.

Develops high order "canonical" or "symplectic" integration methods, which preserve phase space volume.

Sanz-Serna, J. M. (1992) "Symplectic integrators for Hamiltonian problems: An overview", *Acta Numerica 1992*, pp. 243-286.

Reviews numerical methods for simulating Hamiltonian systems that respect the "symplectic" character of the dynamics, which (among other things) ensures that phase space volume is preserved.

Sexton, J. C. and Weingarten, D. H. (1992) "Hamiltonian evolution for the hybrid Monte Carlo algorithm", *Nuclear Physics B*, vol. 380, pp. 665-677.

Discusses methods of simulating Hamiltonian dynamics which, as required by hybrid Monte Carlo, are reversible and preserve phase space volume. In particular, it discusses ways of exploiting situations where some terms in the potential energy are more easily calculated than others, as well as methods that are accurate to higher order than the leapfrog method.

6. Simulated Annealing and Related Techniques

Papers on the use of simulated annealing in optimization are very numerous. Here, I mostly include only works that are relevant to the use of simulated annealing in speeding the approach to equilibrium at a finite temperature. Annealing is also discussed in (Geman and Geman, 4:1984).

Berg, B. A. and Celik, T. (1992) "New approach to spin-glass simulations", *Physical Review Letters*, vol. 69, pp. 2292-2295.

Shows that simulations that sample from a "multicanonical" distribution, in which all energies in some range are of roughly equal probability, can reach the equilibrium distribution of a difficult system more quickly than can a direct simulation of the canonical distribution.

Bertsimas, D. and Tsitsiklis, J. (1993) "Simulated annealing", *Statistical Science*, vol. 8, pp. 10-15.

Reviews work on simulated annealing, including convergence results and their relevance (or lack thereof), work on computational complexity, and empirical evaluations.

Černý, V. (1985) "Thermodynamical approach to the travelling salesman problem: An efficient simulation algorithm", *Journal of Optimization Theory and Applications*, vol. 45, pp. 41-51.

Applies the concept of annealing using the Metropolis algorithm to the problem of finding good solutions to the travelling salesman problem.

Frantz, D. D. and Freeman, D. L. (1990) "Reducing quasi-ergodic behavior in Monte Carlo simulations by J-walking: Applications to atomic clusters", *Journal of Chemical Physics*, vol. 93, pp. 2769-2784.

Presents a method of speeding convergence and reducing correlations in a Markov chain simulation by occasionally attempting moves to a state sampled from a higher-temperature distribution. Discusses methods of minimizing the systematic error that is introduced when the points from the higher-temperature distribution are not independent, as when they come from another Markov chain simulation.

Grieg, D. M., Porteous, B. T., and Seheult, A. H. (1989) "Exact maximum *a posteriori* estimation for binary images", *Journal of the Royal Statistical Society B*, vol. 51, pp. 271-279.

Compares the exact maximum *a posteriori* estimates for a simple image reconstruction problem with those found using simulated annealing, showing that the latter are sometimes far from the exact estimate, and that the exact estimate is sometimes undesirable in any case.

Hajek, B. (1988) "Cooling schedules for optimal annealing", *Mathematics of Operations Research*, vol. 13, pp. 311-329.

Proves that simulated annealing converges asymptotically to the set of global optima when a logarithmic annealing schedule is used, with a constant given by the greatest depth of a local minimum.

Ingber, L. (1989) "Very fast simulated re-annealing", *Mathematical and Computer Modelling*, vol. 12, pp. 967-973.

Advocates a particular form of distribution for selecting candidate moves in the simulated annealing algorithm as applied to optimization, and the use of re-annealing with new parameters determined by the situation at the current minimum value.

Geyer, C. J. (1991) "Markov chain Monte Carlo maximum likelihood", in E. M. Keramidas (editor), *Computing Science and Statistics: Proceedings of the 23rd Symposium on the Interface*, pp. 156-163, Interface Foundation.

Discusses various aspects of Markov chain Monte Carlo estimation. Of particular interest is the description of "Metropolis-coupled Markov chains" in Section 3, a technique related to simulated annealing.

Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P. (1983) "Optimization by simulated annealing", *Science*, vol. 220, pp. 671-680.

The influential paper that introduced the idea of using the Metropolis algorithm in conjunction with simulated annealing to find good solutions to difficult optimization problems.

Marinari, E. and Parisi, G. (1992) "Simulated tempering: A new Monte Carlo Scheme", *Europhysics Letters*, vol. 19, pp. 451-458.

Describes a scheme for sampling in systems where equilibrium is difficult to reach, in which the

temperature is made part of the state, allowing the system to surmount energy barriers at times when the temperature is high.

Szu, H. and Hartley, R. (1987) "Fast simulated annealing", *Physics Letters A*, vol. 122, pp. 157-162.

Advocates use of a multivariate Cauchy distribution to select candidate moves in the simulated annealing algorithm as applied to optimization.

Salamon, P., Nulton, J. D., Harland, J. R., Pedersen, J., Ruppeiner, G. and Liao, L. (1988) "Simulated annealing with constant thermodynamic speed", *Computer Physics Communications*, vol. 49, pp. 423-428.

Advocates using an adaptive annealing schedule in which the simulation stays a constant distance from the equilibrium distribution at the current temperature.

Otten, R. H. J. M. and van Ginneken, L. P. P. P. (1984) "Floorplan design using simulated annealing", *IEEE International Conference on Computer-Aided Design (ICCAD-84)*.

A concise exposition of simulated annealing, its implementation with a particular cooling strategy, and its application to a problem in computer-aided design.

Wang, J-S. and Swendsen, R. H. (1988) "Low-temperature properties of the $\pm J$ Ising spin glass in two dimensions", *Physical Review B*, vol. 38, pp. 4840-4844.

Describes a technique for reaching the equilibrium distribution of an Ising system at low temperature by simultaneously simulating a number of replicas of the system at various temperatures and introducing Metropolis moves that transfer information between the replicas.

7. Free Energy Estimation

For reviews of free energy estimation methods, see also (Valleau and Torrie, 1:1977) and (Sheykhet and Simkin, 1:1990). The book by Sinclair (1:1993) is also relevant to this topic.

Aldous, D. (1993) "Approximate counting via Markov chains", *Statistical Science*, vol. 8, pp. 16-19.

Reviews work on approximate counting of large sets using methods based on uniform sampling via Markov chains. This problem is a special case of free energy estimation, and the solutions discussed are related to methods developed for that problem.

Bash, P. A., Singh, U. C., Langridge, R., and Kollman, P. A. (1987), "Free energy calculations by computer simulation", *Science*, vol. 236, pp. 564-568.

Discusses methods of free energy computation, including the "slow growth" method, and applies them to the calculation of the free energy of solvation for various organic molecules.

Bennett, C. H. (1976) "Efficient estimation of free energy differences from Monte Carlo data", *Journal of Computational Physics*, vol. 22, pp. 245-268.

An in-depth treatment of the problem of estimating free energy differences from Monte Carlo data. Derives the acceptance ratio method, discusses allocation of simulation time between the systems, and presents an interpolation method for handling systems whose distributions do not overlap.

Frenkel, D. (1986) "Free-energy computation and first-order phase transitions", in *Molecular-Dynamics Simulation of Statistical-Mechanical Systems (Proceedings of the International School of Physics "Enrico Fermi", Course 97)*, Amsterdam: North-Holland.

Reviews many techniques used in free energy estimation, including methods based on particle

insertion, on overlap in distributions, and on thermodynamic integration.

Jerrum, M. R., Valiant, L. G., and Vazirani, V. V. (1986) "Random generation of combinatorial structures from a uniform distribution", *Theoretical Computer Science*, vol. 43, pp. 169-188.

Gives reductions between the problems of generating structures uniformly from a family of sets, and approximately counting the size of such sets.

Mezei, M. (1987) "Adaptive umbrella sampling: Self-consistent determination of the non-Boltzmann bias", *Journal of Computational Physics*, vol. 68, pp. 237-248.

Presents a heuristic procedure for automatically determining the weighting function required for the umbrella sampling method of free energy estimation.

Sinclair, A. and Jerrum, M. (1989) "Approximate counting, uniform generation, and rapidly mixing Markov chains", *Information and Computation*, vol. 82, pp. 93-133.

A paper of interest for two reasons. It discusses how algorithms for approximate counting of sets of structures that are "self-reducible" can be used to sample almost uniformly from such sets, and, conversely, that algorithms for sampling almost uniformly from such sets can be used to approximately count them. This work is related to methods for free energy estimation. The paper also presents a technique for showing that complex Markov chains are "rapidly mixing" — i.e. that they converge in a time that is small in relation to the size of the problem.

Torrie, G. M. and Valleau, J. P. (1977) "Nonphysical sampling distributions in Monte Carlo free-energy estimation: Umbrella sampling", *Journal of Computational Physics*, vol. 23, pp. 187-199.

Shows how to estimate the difference in free energy of two systems using data from a single simulation run that has been designed to sample from a distribution that overlaps the canonical distributions for both systems.

Voter, A. F. (1985) "A Monte Carlo method for determining free-energy differences and transition state theory rate constants", *Journal of Chemical Physics*, vol. 82, pp. 1890-1899.

Extends the "acceptance ratio" method of Bennett (7:1976) by considering moves that translate the state vector at the same time as they switch energy functions, thereby allowing the free energy difference for some pairs of systems to be found directly, without the need for intermediate systems.

8. Error Assessment and Reduction

These references concern strategies for performing Markov chain Monte Carlo sampling and for assessing the accuracy of the resulting estimates. This topic is also discussed in a number of the reviews in Section 1, particularly (Ripley, 1:1987, Chapter 6).

Diggle, P. J. (1990) *Time Series: A Biostatistical Introduction*, Oxford: Clarendon Press.

A readable introduction to theory and algorithms for analysing time series. Section 3.6 discusses the application of these methods to assessing the error in an estimate of the mean of a stationary time series.

Gelman, A. and Rubin, D. B. (1993) "Inference from iterative simulation using multiple sequences" (with discussion), *Statistical Science*, vol. 7, pp. 457-511

Advocates use of multiple runs in order to properly assess errors in iterative simulations, and proposes a detailed implementation of this strategy, based on the construction of an "overdispersed" approximation to the distribution for use as an initial distribution for the Markov chain.

Geweke, J. (1992) "Evaluating the accuracy of sampling-based approaches to the calculation of posterior moments" (with discussion), in J. M. Bernardo, J. O. Berger, A. P. Dawid, and A. F. M. Smith (editors), *Bayesian Statistics 4*, pp. 169-193, Oxford University Press.

Applies spectral analysis to diagnosing convergence and assessing error in Gibbs sampling runs.

Geyer, C. J. (1993) "Practical Markov chain Monte Carlo" (with discussion), *Statistical Science*, vol. 7, pp. 473-511.

Advocates use of a single long run in forming estimates from Markov chain Monte Carlo simulation and in assessing their accuracy.

Goutsias, J. (1991) "Unilateral approximation of Gibbs random field images", *CVGIP: Graphical Models and Image Processing*, vol. 53, pp. 240-257.

Shows how a Gibbs random field can be approximated by a "mutually-compatible" Gibbs random field (which may be viewed as a belief network). The approximation can be used to initialize a Gibbs sampling simulation used to sample from the original Gibbs random field model.

Heidelberger, P. and Welch, P. D. (1981) "A spectral method for confidence interval generation and run length control in simulations", *Communications of the ACM*, vol. 24, pp. 233-245.

Presents a method for assessing error in estimates derived from simulation data based on estimating the spectral density at zero frequency, using methods tailored to that task.

Law, A. M. (1983) "Statistical analysis of simulation output data", *Operations Research*, vol. 31, pp. 983-1029.

Discusses the assessment of error in estimates obtained via simulation in an operations research context. The methods presented for "steady state" simulations are relevant to Markov chain Monte Carlo methods.

Morales, J. J., Nuevo M. J., and Rull, L. F. (1990) "Statistical error methods in computer simulations", *Journal of Computational Physics*, vol. 89, pp. 432-438.

Compares a batch method of assessing error with a method based on estimating the autocorrelation function.

Straatsma, T. P., Berendsen, J. J. C., and Stam, A. J. (1986) "Estimation of statistical errors in molecular simulation calculations", *Molecular Physics*, vol. 57, pp. 89-95.

Presents a method of assessing error in estimates derived from correlated series, based on estimating the autocorrelation function, and advocates its use in preference to batching methods.

Roberts, G. O. (1992) "Convergence diagnostics of the Gibbs sampler", in J. M. Bernardo, J. O. Berger, A. P. Dawid, and A. F. M. Smith (editors), *Bayesian Statistics 4*, pp. 775-782, Oxford University Press.

Proposes a method of diagnosing convergence of a single run of a reversible Markov chain, as measured by a particular metric for judging closeness to the equilibrium distribution.

9. Background References

This section lists works on statistical inference, computatonal statistics, statistical physics, random variate generation, and other miscellaneous topics.

Alexandrowicz, Z. (1971) "Stochastic models for the statistical description of lattice systems", *Journal of Chemical Physics*, vol. 55, pp. 2765-2779.

Presents an approach to statistical physics problems in which the system is approximated by a probabilistic model for which sampling can easily be performed, without the need for methods based on Markov chains.

Barnett, V. (1982) *Comparative Statistical Inference*, Second Edition, New York: John Wiley.

Presents and compares the conceptual bases of frequentist and Bayesian statistical inference and of decision theory.

Berger, J. O. (1985) *Statistical Decision Theory and Bayesian Analysis*, New York: Springer-Verlag.

A comprehensive treatment of Bayesian statistics and decision theory, including discussion of foundational issues, as well as technical development. Discusses hierarchical priors specified using hyperparameters, but has little on non-parametric models.

Box, G. E. P. and Tiao, G. C. (1973) *Bayesian Inference in Statistical Analysis*, New York: John Wiley.

A good exposition of Bayesian theory and practice, though mostly confined to traditional sorts of statistical problems.

Cipra, B. A. (1987) "An introduction to the Ising Model", *American Mathematical Monthly*, vol. 94, pp. 937-959.

A tutorial introduction to the Ising model of ferromagnetism, concentrating on its mathematical, and particularly combinatorial, aspects.

DeGroot, M. H. (1970) *Optimal Statistical Decisions*, New York: McGraw-Hill.

An exposition of Bayesian statistics and decision theory at an intermediate level of difficulty, including discussion of the conceptual basis of probability and utility, and with extensive discussion of decision problems, but concentrating on simple parametric distributions.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977) "Maximum likelihood from incomplete data via the EM algorithm" (with discussion), *Journal of the Royal Statistical Society B*, vol. 39, pp. 1-38.

Presents a unified view of a class of algorithms for finding maximum likelihood estimates for models with latent variables, or in other situations where some variables were not observed.

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

A comprehensive reference on methods of generating random numbers drawn from standard distributions.

Efron, B. (1979) "Computers and the theory of statistics: Thinking the unthinkable", *SIAM Review*, vol. 21, pp. 460-480.

Discusses a number of computer-intensive methods for statistical analysis, including the jacknife, the bootstrap, and cross-validation.

Evans, M. (1991) "Adaptive importance sampling and chaining", in N. Flournoy and R. K. Tsutakawa, *Statistical Multiple Integration: Proceeding of a Joint Summer Research Conference Held at Humboldt University, June 17-23, 1989*, Series in Contemporary Mathematics, no. 115, Providence, Rhode Island: American Mathematical Society.

Describes how, starting from a poor, but not completely unacceptable importance sampler, one can find successively better importance samplers, and how an acceptable initial importance sampler can found by "chaining" from an easier problem.

Everitt, B. S. (1984) *An Introduction to Latent Variable Models*, London: Chapman and Hall.

Discusses latent variable models of several sorts, including factor analysis, where both the latent and the observable variables are continuous, and latent class / latent profile analysis, where the latent variable is discrete.

Flournoy, N. and Tsutakawa, R. K. (1991) *Statistical Multiple Integration: Proceeding of a Joint Summer Research Conference Held at Humboldt University, June 17-23, 1989*, Series in Contemporary Mathematics, no. 115, Providence, Rhode Island: American Mathematical Society.

A collection of papers on methods of performing integrations over high-dimensional spaces, such as those required for performing Bayesian inference.

Gilks, W. R. (1992) "Derivative-free adaptive rejection sampling for Gibbs sampling", in J. M. Bernardo, J. O. Berger, A. P. Dawid, and A. F. M. Smith (editors), *Bayesian Statistics 4*, pp. 641-649, Oxford University Press.

Modifies the method of Gilks and Wild (4:1992) to permit its use when the log of the probability density can be calculated, but not its derivative.

Gilks, W. R. and Wild, P. (1992) "Adaptive rejection sampling for Gibbs sampling", *Applied Statistics*, vol. 41, pp. 337-348.

Describes a "black box" method for generating a random value from a continuous univariate distribution for which the log probability density function is concave, and it and its derivative can be calculated. Argues that this method will often be applicable to Gibbs sampling for hierarchical Bayesian models.

Jeffreys, W. H. and Berger, J. O. (1992) "Ockham's razor and Bayesian analysis", *American Scientist*, vol. 80, pp. 64-72. See also the discussion in vol. 80, pp. 212-214.

Explains the automatic Ockham's razor effects of Bayesian inference, with reference to a scientific hypothesis testing situation.

Press, S. J. (1989) *Bayesian Statistics: Principles, Models, and Applications*, New York: John Wiley.

An introduction to the concepts, implementation, applications, and history of Bayesian inference.

Schmitt, S. A. (1969) *Measuring Uncertainty: An Elementary Introduction to Bayesian Statistics*, Reading, Massachussets: Addison-Wesley.

A readable introduction to Bayesian statistics and decision theory, concentrating on simple parametric models and traditional statistical applications, and avoiding sophisticated mathematics.

Skilling, J. (1992) "Bayesian numerical analysis", to appear in W. T. Grandy and P. Milonni (editors) *Physics and Probability*, Cambridge University Press.

Develops deterministic and Monte Carlo methods for performing Bayesian calculations, centred around use of the conjugate gradient method.

Smith, A. F. M. (1991) "Bayesian computational methods", *Philosophical Transactions of the Royal Society of London A*, vol. 337, pp. 369-386.

Surveys methods for performing the high-dimensional integrations required for Bayesian statistical inference, including analytic approximations, numerical integration, and Monte Carlo integration using importance sampling and Gibbs sampling.

Tanner, M. A. (1991) *Tools for Statistical Inference: Observed Data and Data Augmentation Methods*, Lecture Notes in Statistics, vol. 67, New York: Springer-Verlag.

Discusses computational methods for maximum likelihood and Bayesian inference, including inference for models with latent or other unobserved variables.

Thompson, C. J. (1988) *Classical Equilibrium Statistical Mechanics*, Oxford: Clarendon Press.

An introductory, though rather fast-paced, text with a relatively high proportion of material that may be of interest to non-physicists. Topics covered include the elements of macroscopic thermodynamics, microscopic descriptions using the canonical and other ensembles, the relationship between microscopic and macroscopic descriptions, and phase transitions.

Wakefield, J. C., Gelfand, A. E., and Smith, A. F. M. (1991) "Efficient generation of random variates via the ratio-of-uniforms method", *Statistics and Computing*, vol. 1, pp. 129-133.

Describes a generalization of the "ratio-of-uniforms" method for random generation from univariate and multivariate distributions, and discusses optimal translations to improve the acceptance probability of the method.

Woźniakowski (1991) "Average case complexity of multivariate integration", *Bulletin of the American Mathematical Society*, vol. 24, pp. 185-194.

Gives a way of deterministically choosing points for integrating a function in a high-dimensional space that, like Monte Carlo methods, avoids the exponential growth with dimension that would result from using a simple grid. The accuracy bound in only achieved on average, however, for a particular distribution over functions.