An Introduction to MCMC for Machine Learning

Christophe Andrieu
Dept. of Mathematics, Statistics Group, University of Bristol
University Walk, Bristol BS8 1TW, UK
C.Andrieu@bristol.ac.uk

Nando de Freitas
Dept. of Computer Science, University of British Columbia
2366 Main Mall, Vancouver, BC V6T 1Z4, Canada
nando@cs.ubc.ca

Arnaud Doucet
Dept. of Electrical and Electronic Engineering, University of Melbourne
Parkville, Victoria 3052, Australia
doucet@ee.mu.oz.au

Michael I. Jordan
Depts. of Computer Science and Statistics, University of California at Berkeley
387 Soda Hall, Berkeley, CA 94720-1776, USA
jordan@cs.berkeley.edu

September 10, 2001

Abstract. This purpose of this introductory paper is threefold. First, it introduces the Monte Carlo method with emphasis on probabilistic machine learning. Second, it reviews the main building blocks of modern Markov chain Monte Carlo simulation, thereby providing and introduction to the remaining papers of this special issue. Lastly, it discusses new interesting research horizons.

Keywords: Markov chain Monte Carlo, MCMC, sampling, stochastic algorithms.

1. Introduction

A recent survey places the Metropolis algorithm among the ten algorithms that have had the greatest influence on the development and practice of science and engineering in the 20th century (Beichl and Sullivan 2000). This algorithm is an instance of a large class of sampling algorithms, known as Markov chain Monte Carlo (MCMC). These algorithms have played a significant role in statistics, econometrics, physics and computing science over the last two decades. There are several high-dimensional problems, such as computing the volume of a convex body in d dimensions, for which MCMC simulation is the only known general approach for providing a solution within a reasonable time (polynomial in d) (Dyer, Frieze and Kannan 1991, Jerrum and Sinclair 1996).

While convalescing from an illness in 1946, Stan Ulam was playing solitaire. It, then, occurred to him to try to compute the chances that a particular solitaire laid out with 52 cards would come out successfully (Eckhard 1987). After attempting exhaustive combinatorial calculations, he decided to go for the more practical approach of laying out several solitaires at random and then observing and counting the number of successful plays. This idea of selecting a statistical sample to approximate a hard combinatorial problem by a much simpler problem is at the heart of modern Monte Carlo simulation.

Stan Ulam soon realised that computers could be used in this fashion to answer questions of neutron diffusion and mathematical physics. He contacted John Von Neumann, who understood the great potential of this idea. Over the next few years, Ulam and Von Neumann developed many Monte Carlo algorithms, including importance sampling and rejection sampling. Enrico Fermi in the 1930's also used Monte Carlo in the calculation of neutron diffusion, and later designed the FERMIAC, a Monte Carlo mechanical device that performed calculations (Anderson 1986). In the 1940's Nick Metropolis, a young physicist, designed new controls for the state-of-the-art computer (ENIAC) with Klari Von Neumann, John's wife. He was fascinated with Monte Carlo methods and this new computing device. Soon he designed an improved computer, which he named the MANIAC in the hope that computer scientists would stop using acronyms. During the time he spent working on the computing machines, many mathematicians and physicists (Fermi, Von Neumann, Ulam, Teller, Richtmyer, Bethe, Feynman and Gamow) would go to him with their work problems. Eventually in 1949, he published the first public document on Monte Carlo simulation with Stan Ulam (Metropolis and Ulam 1949). This paper introduces, among other ideas, Monte Carlo particle methods, which form the basis of modern sequential Monte Carlo methods such as bootstrap filters, condensation, and survival of the fittest algorithms (Doucet, de Freitas and Gordon 2001). Soon after, he proposed the Metropolis algorithm with the Tellers and the Rosenbluths (Metropolis, Rosenbluth, Rosenbluth, Teller and Teller 1953).

Many papers on Monte Carlo simulation appeared in the physics literature after 1953. From an inference perspective, the most significant contribution was the generalisation of the Metropolis algorithm by Hastings in 1970. Hastings and his student Peskun showed that Metropolis and the more general Metropolis-Hastings algorithms are particular instances of a large family of algorithms, which also includes the Boltzmann algorithm (Hastings 1970, Peskun 1973). They studied the optimality of these algorithms and introduced the formulation of the Metropolis-Hastings algorithm that we adopt in this paper. In the

1980's, two important MCMC papers appeared in the fields of computer vision and artificial intelligence (Geman and Geman 1984, Pearl 1987). Despite the existence of a few MCMC publications in the statistics literature at this time, it is generally accepted that it was only in 1990 that MCMC made the first significant impact in statistics (Gelfand and Smith 1990). In the neural networks literature, the publication of (Neal 1996) was particularly influential.

In the introduction to this special issue, we focus on describing algorithms that we feel are the main building blocks in modern MCMC programs. We should emphasize that in order to obtain the best results out of this class of algorithms, it is important that we do not treat them as black boxes, but instead try to incorporate as much domain specific knowledge as possible into their design. MCMC algorithms typically require the design of proposal mechanisms to generate candidate hypotheses. Many existing machine learning algorithms can be adapted to become proposal mechanisms (de Freitas, Højen-Sørensen, Jordan and Russell 2001). This is often essential to obtain MCMC algorithms that converge quickly. In addition to this, we believe that the machine learning community can contribute significantly to the solution of many open problems in the MCMC field. For this purpose, we have outlined several "hot" research directions at the end of this paper. Finally, readers are encouraged to consult the excellent texts of (Chen, Shao and Ibrahim 2001, Gilks, Richardson and Spiegelhalter 1996, Liu 2001, Meyn and Tweedie 1993, Robert and Casella 1999) and review papers by (Besag, Green, Hidgon and Mengersen 1995, Brooks 1998, Diaconis and Saloff-Coste 1998, Jerrum and Sinclair 1996, Neal 1993, Tierney 1994) for more information on MCMC.

The remainder of this paper is organised as follows. In Part 2, we outline the general problems and introduce simple Monte Carlo simulation, rejection sampling and importance sampling. Part 3 deals with the introduction of MCMC and the presentation of the most popular MCMC algorithms. In part 4, we describe some important research frontiers. To make the paper more accessible, we make no notational distinction between distributions and densities until the section on reversible jump MCMC.

2. MCMC Motivation

MCMC techniques are often applied to solve integration and optimisation problems in large dimensional spaces. These two types of problem play a fundamental role in machine learning, physics, statistics, econometrics and decision analysis. The following are just some examples.

1. **Bayesian inference and learning**: Given some unknown variables $x \in \mathcal{X}$ and data $y \in \mathcal{Y}$, the following typically intractable integration problems are central to Bayesian statistics

   a) *Normalisation*: To obtain the posterior $p(x|y)$ given the prior $p(x)$ and likelihood $p(y|x)$, the normalising factor in Bayes' theorem needs to be computed

   $$p(x|y) = \frac{p(y|x)p(x)}{\int_{\mathcal{X}} p(y|x')p(x')dx'}.$$

   b) *Marginalisation*: Given the joint posterior of $(x, z) \in \mathcal{X} \times \mathcal{Z}$, we may often be interested in the marginal posterior

   $$p(x|y) = \int_{\mathcal{Z}} p(x, z|y)dz.$$

   c) *Expectation*: The objective of the analysis is often to obtain summary statistics of the form

   $$\mathbb{E}_{p(x|y)}(f(x)) = \int_{\mathcal{X}} f(x)p(x|y)dx$$

   for some function of interest $f : \mathcal{X} \to \mathbb{R}^{n_f}$ integrable with respect to $p(x|y)$. Examples of appropriate functions include the conditional mean, in which case $f(x) = x$, or the conditional covariance of x where $f(x) = xx' - \mathbb{E}_{p(x|y)}(x)\mathbb{E}'_{p(x|y)}(x)$.

2. **Statistical mechanics**: Here, one needs to compute the partition function Z of a system with states s and Hamiltonian $E(s)$

   $$Z = \sum_s \exp\left[-\frac{E(s)}{kT}\right],$$

   where k is the Boltzmann's constant and T denotes the temperature of the system. Summing over the large number of possible configurations is prohibitively expensive (Baxter 1982). Note that the problems of computing the partition function and the normalising constant in statistical inference are analogous.

3. **Optimisation**: The goal of optimisation is to extract the solution that minimises some objective function from a large set of feasible solutions. In fact, this set can be continuous and unbounded. In general, it is too computationally expensive to compare all the solutions to find out which one is optimal.

4. **Penalised likelihood model selection**: This task typically involves two steps. First, one finds the maximum likelihood (ML) estimates for each model separately. Then one uses a penalisation term (for example MDL, BIC or AIC) to select one of the models. The problem with this approach is that the initial set of models can be very large. Moreover, many of those models are of not interest and, therefore, computing resources are wasted.

Although we have emphasized integration and optimisation, MCMC also plays a fundamental role in the simulation of physical systems. This is of great relevance in nuclear physics and computer graphics (Chenney and Forsyth 2000, Kalos and Whitlock 1986, Veach and Guibas 1997).

2.1. The Monte Carlo Principle

The idea of Monte Carlo simulation is to draw an i.i.d. set of samples $\{x^{(i)}\}_{i=1}^{N}$ from a target density $p(x)$ defined on a high-dimensional space $\mathcal{X}$ (*e.g.* the set of possible configurations of a system, the space on which the posterior is defined, or the combinatorial set of feasible solutions). These N samples can be used to approximate the target density with the following empirical point-mass function

$$p_N(x) = \frac{1}{N}\sum_{i=1}^{N} \delta_{x^{(i)}}(x),$$

where $\delta_{x^{(i)}}(x)$ denotes the delta-Dirac mass located at $x^{(i)}$. Consequently, one can approximate the integrals (or very large sums) $I(f)$ with tractable sums $I_N(f)$ that converge as follows

$$I_N(f) = \frac{1}{N}\sum_{i=1}^{N} f(x^{(i)}) \quad \xrightarrow[N\to\infty]{a.s.} \quad I(f) = \int_{\mathcal{X}} f(x)p(x)dx.$$

That is, the estimate $I_N(f)$ is unbiased and by the strong law of large numbers, it will almost surely $(a.s.)$ converge to $I(f)$. If the variance (in the univariate case for simplicity) of $f(x)$ satisfies $\sigma_f^2 \triangleq \mathbb{E}_{p(x)}\left(f^2(x)\right) - I^2(f) < \infty$, then the variance of the estimator $I_N(f)$ is equal to $\mathrm{var}(I_N(f)) = \frac{\sigma_f^2}{N}$ and a central limit theorem yields convergence in distribution of the error

$$\sqrt{N}\left(I_N(f) - I(f)\right) \underset{N\to\infty}{\Longrightarrow} \mathcal{N}(0, \sigma_f^2),$$

where $\Longrightarrow$ denotes convergence in distribution (Robert and Casella 1999, Section 3.2). The advantage of Monte Carlo integration over deterministic integration arises from the fact that the former positions the integration grid (samples) in regions of high probability. The N samples can also be used to obtain a maximum of the objective function $p(x)$ as follows

$$\widehat{x} = \underset{x^{(i)};i=1,\ldots,N}{\arg\max} \quad p\left(x^{(i)}\right)$$

However, we will show later that it is possible to construct simulated annealing algorithms that allow us to sample approximately from a distribution whose support is the set of global maxima.

When $p(x)$ has standard form, *e.g.* Gaussian, it is straightforward to sample from it using easily available routines. However, when this is not the case, we need to introduce more sophisticated techniques based on rejection sampling, importance sampling and MCMC.

2.2. Rejection Sampling

We can sample from a distribution $p(x)$, which is known up to a proportionality constant, by sampling from another easy-to-sample proposal distribution $q(x)$ that satisfies $p(x) \leq Mq(x)$, $M < \infty$, using the accept/reject procedure describe in Figure 1 (see also Figure 2). The

Set $i = 1$

Repeat until $i = N$

1. Sample $x^{(i)} \sim q(x)$ and $u \sim \mathcal{U}_{(0,1)}$.
2. If $u < \frac{p\left(x^{(i)}\right)}{Mq\left(x^{(i)}\right)}$ then accept $x^{(i)}$ and increment the counter i by 1. Otherwise, reject.

Figure 1. Rejection sampling algorithm. Here, $u \sim \mathcal{U}_{(0,1)}$ denotes the operation of sampling a uniform random variable on the interval $(0, 1)$.

accepted $x^{(i)}$ can be easily shown to be sampled with probability $p(x)$ (Robert and Casella 1999, page 49). This simple method suffers from severe limitations. It is not always possible to bound $p(x)/q(x)$ with a reasonable constant M over the whole space $\mathcal{X}$. If M is too large, the acceptance probability

$$\Pr(x \text{ accepted}) = \Pr\left(u < \frac{p(x)}{Mq(x)}\right) = \frac{1}{M}$$

will be too small. This makes the method impractical in high-dimensional scenarios.

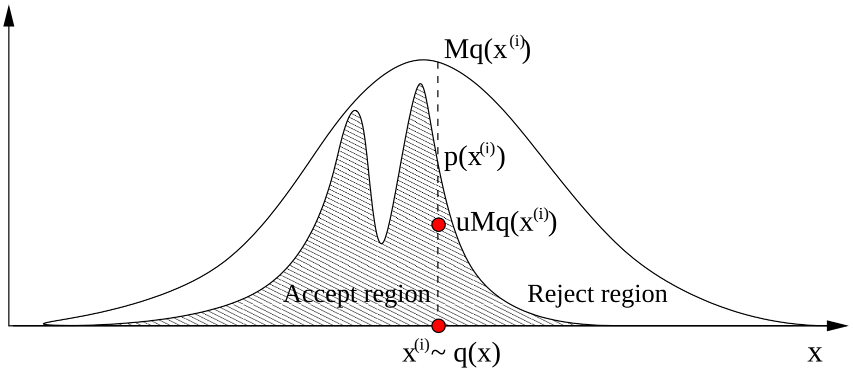

Figure 2. Rejection sampling: sample a candidate $x^{(i)}$ and a uniform variable u. Accept the candidate sample if $uMq(x^{(i)}) < p(x^{(i)})$, otherwise reject it.

2.3. Importance Sampling

Importance sampling is an alternative "classical" solution that goes back to the 1940's; see for example (Geweke 1989, Rubinstein 1981). Let us introduce, again, an arbitrary importance proposal distribution $q(x)$ such that its support includes the support of $p(x)$. Then we can rewrite $I(f)$ as follows

$$I(f) = \int f(x)\, w(x)\, q(x)\, dx$$

where $w(x) \triangleq \frac{p(x)}{q(x)}$ is known as the *importance weight*. Consequently, if one can simulate N i.i.d. samples $\{x^{(i)}\}_{i=1}^{N}$ according to $q(x)$ and evaluate $w(x^{(i)})$, a possible Monte Carlo estimate of $I(f)$ is

$$\widehat{I}_N(f) = \sum_{i=1}^{N} f\left(x^{(i)}\right) w(x^{(i)})$$

This estimator is unbiased and, under weak assumptions, the strong law of large numbers applies, that is $\widehat{I}_N(f) \underset{N\to\infty}{\xrightarrow{a.s.}} I(f)$. It is clear that this integration method can also be interpreted as a sampling method where the posterior density $p(x)$ is approximated by

$$\widehat{p}_N(x) = \sum_{i=1}^{N} w(x^{(i)})\delta_{x^{(i)}}(x)$$

and $\widehat{I}_N(f)$ is nothing but the function $f(x)$ integrated with respect to the empirical measure $\widehat{p}_N(x)$.

Some proposal distributions $q(x)$ will obviously be preferable to others. An important criterion for choosing an optimal proposal distribution is to find one that minimises the variance of the estimator $\widehat{I}_N(f)$. The variance of $f(x)w(x)$ with respect to $q(x)$ is given by

$$\mathrm{var}_{q(x)}\left(f(x)w(x)\right) = \mathbb{E}_{q(x)}\left(f^2(x)w^2(x)\right) - I^2(f) \tag{8}$$

The second term on the right hand side does not depend on $q(x)$ and hence we only need to minimise the first term, which according to Jensen's inequality has the following lower bound

$$\mathbb{E}_{q(x)}\left(f^2(x)w^2(x)\right) \geq \left(\mathbb{E}_{q(x)}\left(|f(x)|w(x)\right)\right)^2 = \left(\int |f(x)|p(x)dx\right)^2$$

This lower bound is attained when we adopt the following *optimal importance distribution*

$$q^{\star}(x) = \frac{|f(x)|p(x)}{\int |f(x)|p(x)dx}$$

The optimal proposal is not very useful in the sense that it is not easy to sample from $|f(x)|p(x)$. However, it tells us that high sampling efficiency is achieved when we focus on sampling from $p(x)$ in the important regions where $|f(x)|p(x)$ is relatively large; hence the name importance sampling.

This result implies that importance sampling estimates can be super-efficient. That is, for a a given function $f(x)$, it is possible to find a distribution $q(x)$ that yields an estimate with a lower variance than when using a perfect Monte Carlo method, *i.e.* with $q(x) = p(x)$. This property is often exploited to evaluate the probability of rare events in communication networks (Smith, Shafi and Gao 1997). There the quantity of interest is a tail probability (bit error rate) and hence $f(x) = \mathbb{I}_E(x)$ where $\mathbb{I}_E(x) = 1$ if $x \in E$ and 0 otherwise (see Figure 3). One could estimate the bit error rate more efficiently by sampling according to $q(x) \propto \mathbb{I}_E(x)p(x)$ than according to $q(x) = p(x)$. That is, it is wasteful to propose candidates in regions of no utility. In many applications, the aim is usually different in the sense that one wants to have a good approximation of $p(x)$ and not of a particular integral with respect to $p(x)$, so we often seek to have $q(x) \simeq p(x)$.

As the dimension of the x increases, it becomes harder to obtain a suitable $q(x)$ from which to draw samples. A sensible strategy is to adopt a parameterised $q(x,\theta)$ and to adapt θ during the simulation. *Adaptive importance sampling* appears to have originated in the

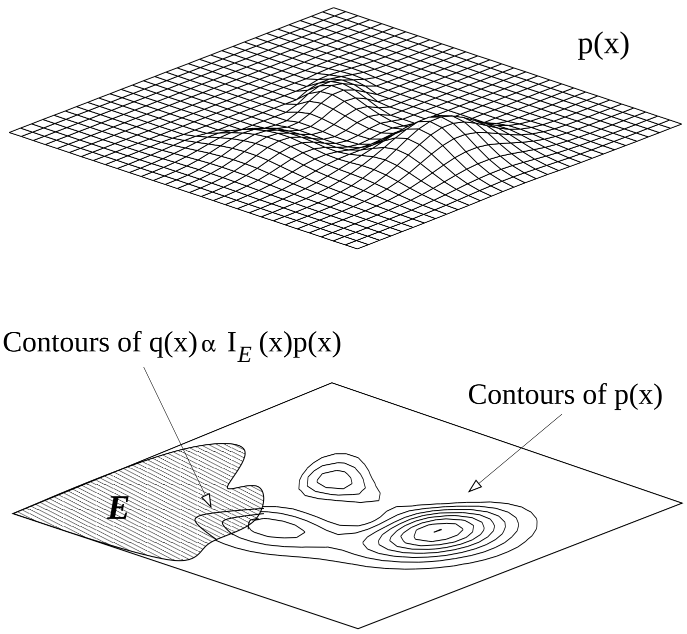

Figure 3. Importance sampling: one should place more importance on sampling from the state space regions that matter. In this particular example one is interested in computing a tail probability of error (detecting infrequent abnormalities).

structural safety literature (Bucher 1988), and has been extensively applied in the communications literature (Al-Qaq, Devetsikiotis and Townsend 1995, Remondo, Srinivasan, Nicola, van Etten and Tattje 2000). This technique has also been exploited recently in the machine learning community (de Freitas, Niranjan, Gee and Doucet 2000, Cheng and Druzdzel 2000, Ortiz and Kaelbling 2000, Schuurmans and Southey 2000). A popular adaptive strategy involves computing the derivative of the first term on the right hand side of equation (1)

$$D(\theta) = \mathbb{E}_{q(x,\theta)}\left(f^2(x)w(x,\theta)\frac{\partial w(x,\theta)}{\partial \theta}\right)$$

and then updating the parameters as follows

$$\theta_{t+1} = \theta_t - \alpha\frac{1}{N}\sum_{i=1}^{N} f^2(x^{(i)})w(x^{(i)},\theta_t)\frac{\partial w(x^{(i)},\theta_t)}{\partial \theta_t}$$

where α is a learning rate and $x^{(i)} \sim q(x,\theta)$. Other optimisation approaches that make use of the Hessian are also possible.

When the normalising constant of $p(x)$ is unknown, it is still possible to apply the importance sampling method by rewriting $I(f)$ as follows:

$$I(f) = \frac{\int f(x)\, w(x)\, q(x)\, dx}{\int w(x)\, q(x)\, dx}$$

where $w(x) \propto \frac{p(x)}{q(x)}$ is now only known up to a normalising constant. The Monte Carlo estimate of $I(f)$ becomes

$$\widetilde{I}_N(f) = \frac{\frac{1}{N}\sum_{i=1}^{N} f\left(x^{(i)}\right) w(x^{(i)})}{\frac{1}{N}\sum_{j=1}^{N} w\left(x^{(i)}\right)} = \sum_{i=1}^{N} f\left(x^{(i)}\right) \widetilde{w}(x^{(i)})$$

where $\widetilde{w}(x^{(i)})$ is a normalised importance weight. For N finite, $\widetilde{I}_N(f)$ is biased (ratio of two estimates) but asymptotically, under weak assumptions, the strong law of large numbers applies, that is $\widetilde{I}_N(f) \underset{N\to\infty}{\xrightarrow{a.s.}} I(f)$. Under additional assumptions a central limit theorem can be obtained (Geweke 1989). The estimator $\widetilde{I}_N(f)$ has been shown to perform better than $\widehat{I}_N(f)$ in some setups under squared error loss (Robert and Casella 1999).

If one is interested in obtaining M *i.i.d.* samples from $\widehat{p}_N(x)$, then an asymptotically $(N/M \to \infty)$ valid method consists of resampling M times according to the discrete distribution $\widehat{p}_N(x)$. This procedure results in M samples $\widetilde{x}^{(i)}$ with the possibility that $\widetilde{x}^{(i)} = \widetilde{x}^{(j)}$ for $i \neq j$. This method is known as *sampling importance resampling (SIR)* (Rubin 1988). After resampling, the approximation of the target density is

$$\widetilde{p}_M(x) = \frac{1}{M}\sum_{i=1}^{M} \delta_{\widetilde{x}^{(i)}}(x) \tag{13}$$

The resampling scheme introduces some additional Monte Carlo variation. It is, therefore, not clear whether the SIR procedure can lead to practical gains in general. However, in the sequential Monte Carlo setting described in Section 4.3, it is essential to carry out this resampling step.

We conclude this section by stating that even with adaptation, it is often impossible to obtain proposal distributions that are easy to sample from and good approximations at the same time. For this reason, we need to introduce more sophisticated sampling algorithms based on Markov chains.

3. MCMC Algorithms

MCMC is a strategy for generating samples $x^{(i)}$ while exploring the state space $\mathcal{X}$ using a Markov chain mechanism. This mechanism is constructed so that the chain spends more time in the most important regions. In particular, it is constructed so that the samples $x^{(i)}$ mimic samples drawn from the target distribution $p(x)$. (We reiterate that we use MCMC when we cannot draw samples from $p(x)$ directly, but can evaluate $p(x)$ up to a normalising constant.)

It is intuitive to introduce Markov chains on finite state spaces, where $x^{(i)}$ can only take s discrete values $x^{(i)} \in \mathcal{X} = \{x_1, x_2, \ldots, x_s\}$. The stochastic process $x^{(i)}$ is called a Markov chain if

$$p(x^{(i)}|x^{(i-1)}, \ldots, x^{(1)}) = T(x^{(i)}|x^{(i-1)}),$$

The chain is *homogeneous* if $T \triangleq T(x^{(i)}|x^{(i-1)})$ remains invariant for all i, with $\sum_{x^{(i)}} T(x^{(i)}|x^{(i-1)}) = 1$ for any i. That is, the evolution of the chain in a space $\mathcal{X}$ depends solely on the current state of the chain and a fixed transition matrix.

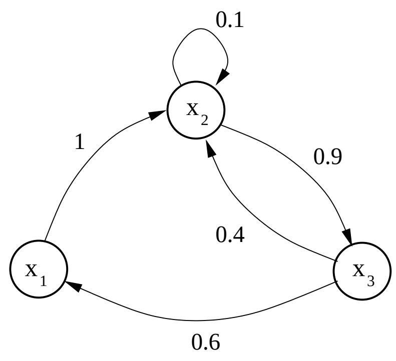

Figure 4. Transition graph for the Markov chain example with $\mathcal{X} = \{x_1, x_2, x_3\}$.

As an example, consider a Markov chain with three states ($s = 3$) and a transition graph as illustrated in Figure 4. The transition matrix for this example is

$$T = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0.1 & 0.9 \\ 0.6 & 0.4 & 0 \end{bmatrix}$$

If the probability vector for the initial state is $\mu(x^{(1)}) = (0.5, 0.2, 0.3)$, it follows that $\mu(x^{(1)})T = (0.2, 0.6, 0.2)$ and, after several iterations (multiplications by T), the product $\mu(x^{(1)})T^t$ converges to $p(x) = (0.2, 0.4, 0.4)$. No matter what initial distribution $\mu(x^{(1)})$ we use, the chain will stabilise at $p(x) = (0.2, 0.4, 0.4)$. This stability result plays a fundamental role in MCMC simulation. For any starting point, the chain will convergence to the invariant distribution $p(x)$, as long as T is a stochastic transition matrix that obeys the following properties:

1. *Irreducibility*: For any state of the Markov chain, there is a positive probability of visiting all other states. That is, the matrix T cannot be reduced to separate smaller matrices, which is also the same as stating that the transition graph is connected.

2. *Aperiodicity*: The chain should not get trapped in cycles.

A sufficient, but not necessary, condition to ensure that a particular $p(x)$ is the desired invariant distribution is the following reversibility (detailed balance) condition

$$p(x^{(i)})T(x^{(i-1)}|x^{(i)}) = p(x^{(i-1)})T(x^{(i)}|x^{(i-1)}).$$

Summing both sides over $x^{(i-1)}$, gives us

$$p(x^{(i)}) = \sum_{x^{(i-1)}} p(x^{(i-1)})T(x^{(i)}|x^{(i-1)}).$$

MCMC samplers are irreducible and aperiodic Markov chains that have the target distribution as the invariant distribution. One way to design these samplers is to ensure that detailed balance is satisfied. However, it is also important to design samplers that converge quickly. Indeed, most of our efforts will be devoted to increasing the convergence speed.

Spectral theory gives us useful insights into the problem. Notice that $p(x)$ is the left eigenvector of the matrix T with corresponding eigenvalue 1. In fact, the Perron-Frobenius theorem from linear algebra tells us that the remaining eigenvalues have absolute value less than 1. The second largest eigenvalue, therefore, determines the rate of convergence of the chain, and should be as small as possible.

The concepts of irreducibility, aperiodicity and invariance can be better appreciated once we realise the important role that they play in our lives. When we search for information on the World-Wide Web, we typically follow a set of links (Berners-Lee, Cailliau, Luotonen, Nielsen and Secret 1994). We can interpret the webpages and links, respectively, as the nodes and directed connections in a Markov chain transition graph. Clearly, we (say, the random walkers on the Web) want to avoid getting trapped in cycles (aperiodicity) and want to be able to access all the existing webpages (irreducibility). Let us consider, now, the popular information retrieval algorithm used by the search engine Google, namely PageRank (Page, Brin, Motwani and Winograd

1998). PageRank requires the definition of a transition matrix with two components $T = L + E$. L is a large link matrix with rows and columns corresponding to web pages, such that the entry $L_{i,j}$ represents the normalised number of links from web page i to web page j. E is a uniform random matrix of small magnitude that is added to L to ensure irreducibility and aperiodicity. That is, the addition of noise prevents us from getting trapped in loops, as it ensures that there is always some probability of jumping to anywhere on the Web. From our previous discussion, we have

$$p(x^{(i+1)})\,[L+E] = p(x^{i})$$

where, in this case, the invariant distribution (eigenvector) $p(x)$ represents the rank of a webpage x. Note that it is possible to design more interesting transition matrices in this setting. As long as one satisfies irreducibility and aperiodicity, one can incorporate terms into the transition matrix that favour particular webpages or that bias the search in useful ways.

In continuous state spaces, the transition matrix T becomes an integral kernel K and $p(x)$ becomes the corresponding eigenfunction

$$\int p(x^{(i)})K(x^{(i+1)}|x^{(i)})dx^{(i)} = p(x^{(i+1)}).$$

The kernel K is the conditional density of $x^{(i+1)}$ given the value $x^{(i)}$. It is a mathematical representation of a Markov chain algorithm. In the following subsections we describe various of these algorithms.

3.1. The Metropolis-Hastings Algorithm

The *Metropolis-Hastings* (MH) algorithm is the most popular MCMC method (Hastings 1970, Metropolis et al. 1953). In later sections, we will see that most practical MCMC algorithms can be interpreted as special cases or extensions of this algorithm.

An MH step of invariant distribution $p(x)$ and proposal distribution $q(x^\star|\,x)$ involves sampling a candidate value $x^\star$ given the current value x according to $q(x^\star|\,x)$. The Markov chain then moves towards $x^\star$ with acceptance probability $\mathcal{A}(x, x^\star) = \min\{1, [p(x)q(x^\star|x)]^{-1}\,p(x^\star)q(x|x^\star)\}$, otherwise it remains at x. The pseudo-code is shown in Figure 5, while Figure 6 shows the results of running the MH algorithm with a Gaussian proposal distribution $q(x^\star|x^{(i)}) = \mathcal{N}(x^{(i)}, 100)$ and a bimodal target distribution $p(x) \propto 0.3\exp\left(-0.2x^2\right) + 0.7\exp\left(-0.2(x-10)^2\right)$ for 5000 iterations. As expected, the histogram of the samples approximates the target distribution.

1. Initialise $x^{(0)}$.
2. For $i = 0$ to $N-1$
   - Sample $u \sim \mathcal{U}_{[0,1]}$.
   - Sample $x^{\star} \sim q(x^{\star}|x^{(i)})$.
   - If $u < \mathcal{A}(x^{(i)}, x^{\star}) = \min\left\{1, \frac{p(x^{\star})q(x^{(i)}|x^{\star})}{p(x^{(i)})q(x^{\star}|x^{(i)})}\right\}$

     $x^{(i+1)} = x^{\star}$

     else

     $x^{(i+1)} = x^{(i)}$

Figure 5. Metropolis-Hastings algorithm.

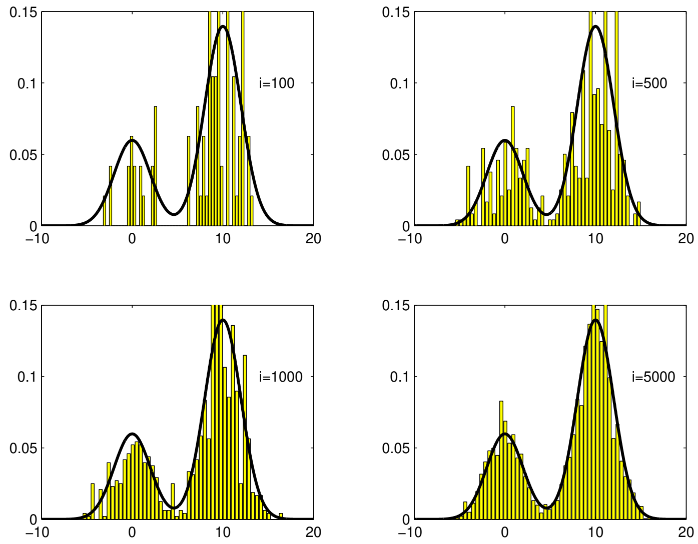

Figure 6. Target distribution and histogram of the MCMC samples at different iteration points.

The MH algorithm is very simple, but it requires careful design of the proposal distribution $q(x^{\star}|x)$. In subsequent sections, we will see that many MCMC algorithms arise by considering specific choices of this distribution. In general, it is possible to use suboptimal inference and learning algorithms to generate data-driven proposal distributions.

The transition kernel for the MH algorithm is

$$K_{\mathrm{MH}}(x^{(i+1)}|x^{(i)}) = q(x^{(i+1)}|x^{(i)})\mathcal{A}(x^{(i)}, x^{(i+1)}) + \delta_{x^{(i)}}(x^{(i+1)})r(x^{(i)}),$$

where $r(x^{(i)})$ is the term associated with rejection

$$r(x^{(i)}) = \int_{\mathcal{X}} q(x^\star|x^{(i)}) \left(1 - \mathcal{A}(x^{(i)}, x^\star)\right) dx^\star.$$

It is fairly easy to prove that the samples generated by MH algorithm will mimic samples drawn from the target distribution asymptotically. By construction, K_{MH} satisfies the detailed balance condition

$$p(x^{(i)})K_{\mathrm{MH}}(x^{(i+1)}|x^{(i)}) \;=\; p(x^{(i+1)})K_{\mathrm{MH}}(x^{(i)}|x^{(i+1)})$$

and, consequently, the MH algorithm admits $p(x)$ as invariant distribution. To show that the MH algorithm converges, we need to ensure that there are no cycles (aperiodicity) and that every state that has positive probability can be reached in a finite number of steps (irreducibility). Since the algorithm always allows for rejection, it follows that it is aperiodic. To ensure irreducibility, we simply need to make sure that the support of $q(\cdot)$ includes the support of $p(\cdot)$. Under these conditions, we obtain asymptotic convergence (Tierney 1994, Theorem 3, page 1717). If the space $\mathcal{X}$ is small (for example, bounded in $\mathbb{R}^n$), then it is possible to use minorisation conditions to prove uniform (geometric) ergodicity (Meyn and Tweedie 1993). It is also possible to prove geometric ergodicity using Foster-Lyapunov drift conditions (Meyn and Tweedie 1993, Roberts and Tweedie 1996).

The *independent sampler* and the *Metropolis algorithm* are two simple instances of the MH algorithm. In the independent sampler the proposal is independent of the current state, $q(x^\star|x^{(i)}) = q(x^\star)$. Hence, the acceptance probability is

$$\mathcal{A}(x^{(i)}, x^\star) = \min\left\{1, \frac{p(x^\star)q(x^{(i)})}{p(x^{(i)})q(x^\star)}\right\} = \min\left\{1, \frac{w(x^\star)}{w(x^{(i)})}\right\}.$$

This algorithm is close to importance sampling, but now the samples are correlated since they result from comparing one sample to the other. The Metropolis algorithm assumes a symmetric random walk proposal $q(x^\star|x^{(i)}) = q(x^{(i)}|x^\star)$ and, hence, the acceptance ratio simplifies to

$$\mathcal{A}(x^{(i)}, x^\star) = \min\left\{1, \frac{p(x^\star)}{p(x^{(i)})}\right\}.$$

Some properties of the MH algorithm are worth highlighting. Firstly, the normalising constant of the target distribution is not required.

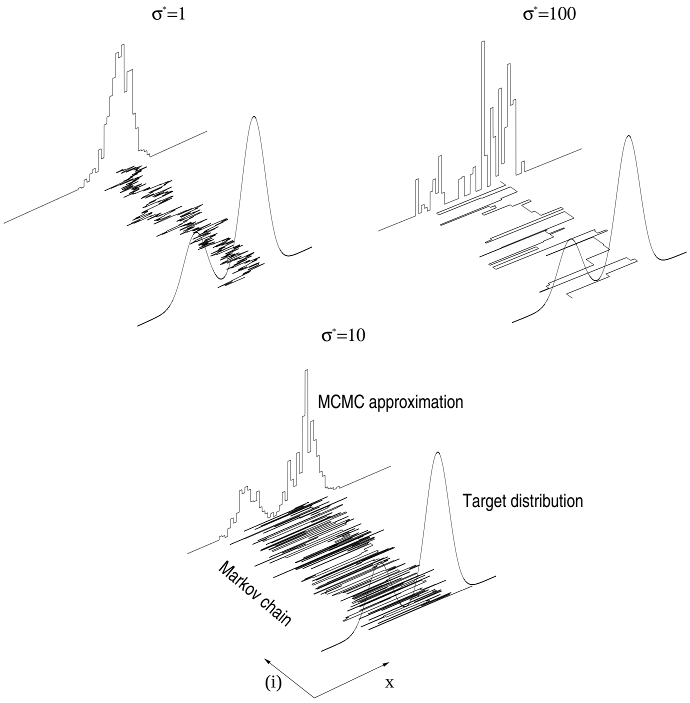

Figure 7. Approximations obtained using the MH algorithm with three Gaussian proposal distributions of different variances.

We only need to know the target distribution up to a constant of proportionality. Secondly, although the pseudo-code makes use of a single chain, it is easy to simulate several independent chains in parallel. Lastly, the success or failure of the algorithm often hinges on the choice of proposal distribution. This is illustrated in Figure 7. Different choices of the proposal standard deviation $\sigma^{\star}$ lead to very different results. If the proposal is too narrow, only one mode of $p(x)$ might be visited. On the other hand, if it is too wide, the rejection rate can be very high, resulting in high correlations. If all the modes are visited while the acceptance probability is high, the chain is said to "mix" well.

3.2. Simulated annealing for global optimization

Let us assume that instead of wanting to approximate $p(x)$, we want to find its global maximum. For example, if $p(x)$ is the likelihood or posterior distribution, we often want to compute the ML and maximum *a posteriori* (MAP) estimates. As mentioned earlier, we could run a Markov chain of invariant distribution $p(x)$ and estimate the global mode by

$$\widehat{x} = \underset{x^{(i)};i=1,\ldots,N}{\arg\max} \quad p\left(x^{(i)}\right).$$

This method is inefficient because the random samples only rarely come from the vicinity of the mode. Unless the distribution has large probability mass around the mode, computing resources will be wasted exploring areas of no interest. A more principled strategy is to adopt *simulated annealing* (Geman and Geman 1984, Kirkpatrick, Gelatt and Vecchi 1983, Van Laarhoven and Arts 1987). This technique involves simulating a non-homogeneous Markov chain whose invariant distribution at iteration i is no longer equal to $p(x)$, but to

$$p_i(x) \propto p^{1/T_i}(x),$$

where T_i is a decreasing cooling schedule with $\lim_{i\to\infty} T_i = 0$. The reason for doing this is that, under weak regularity assumptions on $p(x)$, $p^{\infty}(x)$ is a probability density that concentrates itself on the set of global maxima of $p(x)$. The simulated annealing involves, therefore, just a minor modification of standard MCMC algorithms as shown in Figure 8. The results of applying annealing to the example of the previous section are shown in Figure 9.

1. Initialise $x^{(0)}$ and set $T_0 = 1$.
2. For $i = 0$ to $N - 1$
   - Sample $u \sim \mathcal{U}_{[0,1]}$.
   - Sample $x^{\star} \sim q(x^{\star}|x^{(i)})$.
   - If $u < \mathcal{A}(x^{(i)}, x^{\star}) = \min\left\{1, \frac{p^{\frac{1}{T_i}}(x^{\star})q(x^{(i)}|x^{\star})}{p^{\frac{1}{T_i}}(x^{(i)})q(x^{\star}|x^{(i)})}\right\}$

     $x^{(i+1)} = x^{\star}$

     else

     $x^{(i+1)} = x^{(i)}$
   - Set T_{i+1} according to a chosen cooling schedule.

Figure 8. General simulated annealing algorithm.

To obtain efficient annealed algorithms, it is again important to choose suitable proposal distributions and an appropriate cooling schedule. Many of the negative simulated annealing results reported in the

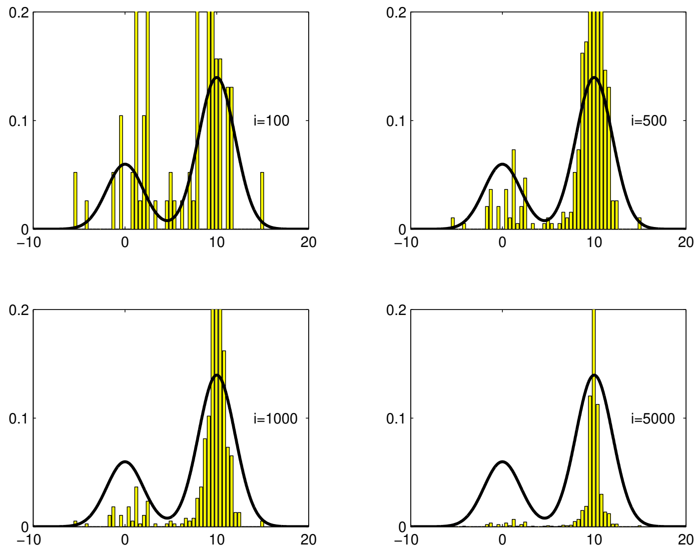

Figure 9. Discovering the modes of the target distribution with the simulated annealing algorithm.

literature often stem from poor proposal distribution design. In some complex variable and model selection scenarios arising in machine learning, one can even propose from complex reversible jump MCMC kernels (Section 3.7) within the annealing algorithm (Andrieu, de Freitas and Doucet 2000a). If one defines a joint distribution over the parameter and model spaces, this technique can be used to search for the best model (according to MDL or AIC criteria) and ML parameter estimates simultaneously.

Most convergence results for simulated annealing typically state that if for a given T_i, the homogeneous Markov transition kernel mixes quickly enough, then convergence to the set of global maxima of $p(x)$ is ensured for a sequence $T_i = (C \ln(i + T_0))^{-1}$, where C and T_0 are problem-dependent. Most of the results have been obtained for finite spaces (Geman and Geman 1984, Van Laarhoven and Arts 1987) or compact continuous spaces (Haario and Sacksman 1991). Some results for non-compact spaces can be found in (Andrieu, Breyer and Doucet 1999).

3.3. Mixtures and Cycles of MCMC Kernels

A very powerful property of MCMC is that it is possible to combine several samplers into mixtures and cycles of the individual samplers (Tierney 1994). If the transition kernels K_1 and K_2 have invariant distribution $p(\cdot)$ each, then the *cycle hybrid kernel* K_1K_2 and the *mixture hybrid kernel* $\nu K_1 + (1-\nu)K_2$, for $0 \leq \nu \leq 1$, are also transition kernels with invariant distribution $p(\cdot)$.

Mixtures of kernels can incorporate global proposals to explore vast regions of the state space and local proposals to discover finer details of the target distribution (Andrieu, de Freitas and Doucet 2000b, Andrieu and Doucet 1999, Robert and Casella 1999). This will be useful, for example, when the target distribution has many narrow peaks. Here, a global proposal locks into the peaks while a local proposal allows one to explore the space around each peak. For example, if we require a high-precision frequency detector, one can use the fast Fourier transform (FFT) as a global proposal and a random walk as local proposal (Andrieu and Doucet 1999). Similarly, in kernel regression and classification, one might want to have a global proposal that places the bases (kernels) at the locations of the input data and a local random walk proposal that perturbs these in order to obtain better fits (Andrieu et al. 2000b). However, mixtures of kernels also play a big role in many other samplers, including the reversible jump MCMC algorithm (Section 3.7). The pseudo-code for a typical mixture of kernels is shown in Figure 10.

1. Initialise $x^{(0)}$.
2. For $i = 0$ to $N-1$
   - Sample $u \sim \mathcal{U}_{[0,1]}$.
   - If $u < \nu$

     Apply the MH algorithm with a global proposal.
   - else

     Apply the MH algorithm with a random walk proposal.

Figure 10. Typical Mixture of MCMC kernels.

Cycles allow us to split a multivariate state vector into components (blocks) that can be updated separately. Typically the samplers will mix more quickly by blocking highly correlated variables. A block MCMC sampler, using b_j to indicate the j-th block, n_b to denote the number of blocks and $x^{(i+1)}_{-[b_j]} \triangleq \{x^{(i+1)}_{b_1}, x^{(i+1)}_{b_2}, \ldots, x^{(i+1)}_{b_{j-1}}, x^{(i)}_{b_{j+1}}, \ldots, x^{(i)}_{b_{n_b}}\}$, is shown in Figure 11. The transition kernel for this algorithm is given by the following expression

$$K_{\text{MH-Cycle}}(x^{(i+1)}|x^{(i)}) = \prod_{j=1}^{n_b} K_{\text{MH(j)}}(x_{b_j}^{(i+1)}|x_{b_j}^{(i)}, x_{-[b_j]}^{(i+1)})$$

where $K_{\text{MH(j)}}$ denotes the j-th MH algorithm in the cycle.

1. Initialise $x^{(0)}$.
2. For $i = 0$ to $N-1$
   - Sample the block $x_{b_1}^{(i+1)}$ according to an MH step with proposal distribution $q_1(x_{b_1}^{(i+1)}|x_{-[b_1]}^{(i+1)}, x_{b_1}^{(i)})$ and invariant distribution $p(x_{b_1}^{(i+1)}|x_{-[b_1]}^{(i+1)})$.
   - Sample the block $x_{b_2}^{(i+1)}$ according to an MH step with proposal distribution $q_2(x_{b_2}^{(i+1)}|x_{-[b_2]}^{(i+1)}, x_{b_2}^{(i)})$ and invariant distribution $p(x_{b_2}^{(i+1)}|x_{-[b_2]}^{(i+1)})$.

$$\vdots$$

   - Sample the block $x_{b_{n_b}}^{(i+1)}$ according to an MH step with proposal distribution $q_{n_b}(x_{b_{n_b}}^{(i+1)}|x_{-[b_{n_b}]}^{(i+1)}, x_{b_{n_b}}^{(i)})$ and invariant distribution $p(x_{b_{n_b}}^{(i+1)}|x_{-[b_{n_b}]}^{(i+1)})$.

Figure 11. Cycle of MCMC kernels - block MH algorithm.

Obviously, choosing the size of the blocks poses some trade-offs. If one samples the components of a multi-dimensional vector one-at-a-time, the chain may take a very long time to explore the target distribution. This problem gets worse as the correlation between the components increases. Alternatively, if one samples all the components together, then the probability of accepting this large move tends to be very low.

A popular cycle of MH kernels, known as *Gibbs sampling* (Geman and Geman 1984), is obtained when we adopt the full conditional distributions $p(x_j|x_{-j}) = p(x_j|x_1, \ldots, x_{j-1}, x_{j+1}, \ldots, x_n)$ as proposal distributions (for notational simplicity, we have replaced the index notation b_j with j). The following section describes it in more detail.

3.4. The Gibbs sampler

Suppose we have an n-dimensional vector x and the expressions for the full conditionals $p(x_j|x_1,\ldots,x_{j-1},x_{j+1},\ldots,x_n)$. In this case, it is often advantageous to use the following proposal distribution for $j=1,\ldots,n$

$$q(x^\star|x^{(i)}) = \begin{cases} p(x_j^\star|x_{-j}^{(i)}) & \text{If } x_{-j}^\star = x_{-j}^{(i)} \\ 0 & \text{Otherwise.} \end{cases}$$

The corresponding acceptance probability is:

$$\begin{aligned} \mathcal{A}(x^{(i)},x^\star) &= \min\left\{1, \frac{p(x^\star)q(x^{(i)}|x^\star)}{p(x^{(i)})q(x^\star|x^{(i)})}\right\} \\ &= \min\left\{1, \frac{p(x^\star)p(x_j^{(i)}|x_{-j}^{(i)})}{p(x^{(i)})p(x_j^\star|x_{-j}^\star)}\right\} \\ &= \min\left\{1, \frac{p(x_{-j}^\star)}{p(x_{-j}^{(i)})}\right\} = 1. \end{aligned}$$

That is, the acceptance probability for each proposal is one and, hence, the deterministic scan Gibbs sampler algorithm is often presented as shown in Figure 12.

Since the Gibbs sampler can be viewed as a special case of the MH algorithm, it is possible to introduce MH steps into the Gibbs sampler. That is, when the full conditionals are available and belong to the family of standard distributions (Gamma, Gaussian, etc.), we will draw the new samples directly. Otherwise, we can draw samples with MH steps embedded within the Gibbs algorithm. For $n=2$, the Gibbs sampler is also known as the data augmentation algorithm, which is closely related to the expectation maximisation (EM) algorithm (Dempster, Laird and Rubin 1977, Tanner and Wong 1987).

Directed acyclic graphs (DAGS) are one of the best known application areas for Gibbs sampling (Pearl 1987). Here, a large-dimensional joint distribution is factored into a directed graph that encodes the conditional independencies in the model. In particular, if $x_{pa(j)}$ denotes the parent nodes of node x_j, we have

$$p(x) = \prod_j p(x_j|x_{pa(j)}).$$

It follows that the full conditionals simplify as follows

$$p(x_j|x_{-j}) = p(x_j|x_{pa(j)}) \prod_{k\in ch(j)} p(x_k|x_{pa(k)})$$

1. Initialise $x_{0,1:n}$.
2. For $i = 0$ to $N-1$
   – Sample $x_1^{(i+1)} \sim p(x_1|x_2^{(i)}, x_3^{(i)}, \ldots, x_n^{(i)})$.
   – Sample $x_2^{(i+1)} \sim p(x_2|x_1^{(i+1)}, x_3^{(i)}, \ldots, x_n^{(i)})$.

   $\vdots$

   – Sample $x_j^{(i+1)} \sim p(x_j|x_1^{(i+1)}, \ldots, x_{j-1}^{(i+1)}, x_{j+1}^{(i)}, \ldots, x_n^{(i)})$.

   $\vdots$

   – Sample $x_n^{(i+1)} \sim p(x_n|x_1^{(i+1)}, x_2^{(i+1)}, \ldots x_{n-1}^{(i+1)})$.

Figure 12. Gibbs Sampler.

where $ch(j)$ denotes the children nodes of x_j. That is, we only need to take into account the parents, the children and the children's parents. This set of variables is known as the *Markov blanket* of x_j. This technique forms the basis of the popular software package for Bayesian updating with Gibbs sampling (BUGS) (Gilks, Thomas and Spiegelhalter 1994). Sampling from the full conditionals, with the Gibbs sampler, lends itself naturally to the construction of general purpose MCMC software. It is sometimes convenient to block some of the variables to improve mixing (Jensen, Kong and Kjærulff 1995, Wilkinson and Yeung 2001).

3.5. Monte Carlo EM

The *EM algorithm* (Baum, Petrie, Soules and Weiss 1970, Dempster et al. 1977) is a standard algorithm for ML and MAP point estimation. If $\mathcal{X}$ contains visible and hidden variables $x = \{x_v, x_h\}$, then a local maximum of the likelihood $p(x_v|\theta)$ given the parameters θ can be found by iterating the following two steps:

1. E step: Compute the expected value of the complete log-likelihood function with respect to the distribution of the hidden variables

$$Q(\theta) = \int_{\mathcal{X}_h} \log\left(p(x_h, x_v|\theta)\right) p(x_h|x_v, \theta^{(\text{old})}) dx_h,$$

where $\theta^{(\text{old})}$ refers to the value of the parameters at the previous time step.

2. M step: Perform the following maximisation $\theta^{(\text{new})} = \arg\max_\theta Q(\theta)$.

In many practical situations, the expectation in the E step is either a sum with an exponentially large number of summands or an intractable integral (Ghahramani 1995, Ghahramani and Jordan 1995, McCulloch 1994, Pasula, Russell, Ostland and Ritov 1999, Utsugi 2001); see also (Dellaert, Seitz, Thorpe and Thrun 2001: this issue). A solution is to introduce MCMC to sample from $p(x_h|x_v, \theta^{(\text{old})})$ and replace the expectation in the E step with a small sum over the samples, as shown in Figure 13. Convergence of this algorithm is discussed in (Sherman, Ho and Dalal 1999), while (Levine and Casella 2001) is a good recent review.

1. Initialise $(x_h^{(0)}, \theta^{(0)})$ and set $i = 0$.
2. Iteration i of EM
   - Sample $\{x_h^{(j)}\}_{j=1}^{N_i}$ with any suitable MCMC algorithm. For example, one could use an MH algorithm with acceptance probability

     $$\mathcal{A} = \min\left\{1, \frac{p(x_v|x_h^\star, \theta^{(i-1)})p(x_h^\star|\theta^{(i-1)})q(x_h^{(j)}|x_h^\star)}{p(x_v|x_h^{(j)}, \theta^{(i-1)})p(x_h^{(j)}|\theta^{(i-1)})q(x_h^\star|x_h^{(j)})}\right\}$$

   - **E step**: Compute

     $$\widehat{Q}(\theta) = \frac{1}{N_i}\sum_{j=1}^{N_i} \log p(x_h^{(j)}, x_v|\theta)$$

   - **M step**: Maximise $\theta^{(i)} = \arg\max_\theta \widehat{Q}(\theta)$.
3. $i \leftarrow i + 1$ and go to 2.

Figure 13. MCMC-EM algorithm.

To improve the convergence behaviour of EM, namely to escape low local minima and saddle points, various authors have proposed stochastic approaches that rely on sampling from $p(x_h|x_v, \theta^{(\text{old})})$ in the E step and then performing the M step using these samples. The method is known as stochastic EM (SEM) when we draw only one sample (Celeux and Diebolt 1985) and Monte Carlo EM (MCEM) when several samples are drawn (Wei and Tanner 1990). There are several annealed variants (such as SAEM) that become more deterministic as the number of iterations increases (Celeux and Diebolt 1992). The are also very efficient algorithms for marginal MAP estimation (SAME) (Doucet, Godsill and Robert 2000). One wishes sometimes that Metropolis had succeeded in stopping the proliferation of acronyms!

3.6. Auxiliary Variable Samplers

It is often easier to sample from an augmented distribution $p(x,u)$, where u is an auxiliary variable, than from $p(x)$. Then, it is possible to obtain marginal samples $x^{(i)}$ by sampling $(x^{(i)},u^{(i)})$ according to $p(x,u)$ and, subsequently, ignoring the samples $u^{(i)}$. This very useful idea was proposed in the physics literature (Swendsen and Wang 1987). Here, we will focus on two well-known examples of auxiliary variable methods, namely hybrid Monte Carlo and slice sampling.

3.6.1. *Hybrid Monte Carlo*

Hybrid Monte Carlo (HMC) is an MCMC algorithm that incorporates information about the gradient of the target distribution to improve mixing in high dimensions. We describe here the "leapfrog" HMC algorithm outlined in (Duane, Kennedy, Pendleton and Roweth 1987, Neal 1996) focusing on the algorithmic details and not on the statistical mechanics motivation. Assume that $p(x)$ is differentiable and everywhere strictly positive. At each iteration of the HMC algorithm, one takes a predetermined number (L) of deterministic steps using information about the gradient of $p(x)$. To explain this in more detail, we first need to introduce a set of auxiliary "momentum" variables $u \in \mathbb{R}^{n_x}$ and define the extended target density

$$p(x,u) = p(x)\mathcal{N}(u;0,I_{n_x}).$$

Next, we need to introduce the n_x-dimensional gradient vector $\Delta(x) \triangleq \partial \log p(x)/\partial x$ and a fixed step-size parameter $\rho > 0$.

In the HMC algorithm, we draw a new sample according to $p(x,u)$ by starting with the previous value of x and generating a Gaussian random variable u. We then take L "frog leaps" in u and x. The values of u and x at the last leap are the proposal candidates in the MH algorithm with target density $p(x,u)$. Marginal samples from $p(x)$ are obtained by simply ignoring u. Given $(x^{(i-1)},u^{(i-1)})$, the algorithm proceeds as illustrated in Figure 14.

When only one deterministic step is used, *i.e.* $L=1$, one obtains the *Langevin algorithm*, which is a discrete time approximation of a

1. Initialise $x^{(0)}$.
2. For $i = 0$ to $N-1$
   - Sample $v \sim \mathcal{U}_{[0,1]}$ and $u^{\star} \sim \mathcal{N}(0, I_{n_x})$.
   - Let $x_0 = x^{(i)}$ and $u_0 = u^{\star} + \rho\Delta(x_0)/2$.
   - For $l = 1, \ldots, L$, take steps
   $$\begin{aligned} x_l &= x_{l-1} + \rho u_{l-1} \\ u_l &= u_{l-1} + \rho_l \Delta(x_l) \end{aligned}$$
   where $\rho_l = \rho$ for $l < L$ and $\rho_L = \rho/2$.
   - If $v < \mathcal{A} = \min\left\{1, \frac{p(x_L)}{p(x^{(i)})}\exp\left(-\frac{1}{2}(u_L^{\mathsf{T}} u_L - u^{\star\mathsf{T}} u^{\star})\right)\right\}$
   $$(x^{(i+1)}, u^{(i+1)}) = (x_L, u_L)$$
   else
   $$(x^{(i+1)}, u^{(i+1)}) = (x^{(i)}, u^{\star})$$

Figure 14. Hybrid Monte Carlo.

Langevin diffusion process. The Langevin algorithm is a special case of MH where the candidate satisfies

$$x^{\star} = x_0 + \rho u_0 = x^{(i-1)} + \rho(u^{\star} + \rho\Delta(x^{(i-1)})/2)$$

with $u^{\star} \sim \mathcal{N}(0, I_{n_x})$.

The choice of the parameters L and ρ poses simulation tradeoffs. Large values of ρ result in low acceptance rates, while small values require many leapfrog steps (expensive computations of the gradient) to move between two nearby states. Choosing L is equally problematic as we want it to be large to generate candidates far from the initial state, but this can result in many expensive computations. HMC, therefore, requires careful tuning of the proposal distribution. It is more efficient, in practice, to allow a different step size ρ for each of the coordinates of x (Ishwaran 1999).

3.6.2. *The Slice Sampler*

The *slice sampler* (Damien, Wakefield and Walker 1999, Higdon 1998, Wakefield, Gelfand and Smith 1991) is a general version of the Gibbs sampler. The basic idea of the slice sampler is to introduce an auxiliary variable $u \in \mathbb{R}$ and construct an extended target distribution $p^{\star}(x,u)$, such that

$$p^{\star}(x,u) = \begin{cases} 1 & \text{if } 0 \leq u \leq p(x) \\ 0 & \text{otherwise.} \end{cases}$$

It is then straightforward to check that

$$\int p^{\star}(x,u)du = \int_{0}^{p(x)} du = p(x).$$

Hence, to sample from $p(x)$ one can sample from $p^{\star}(x,u)$ and then ignore u. The full conditionals are of this augmented model are

$$\begin{aligned} p(u|\, x) &= \mathcal{U}_{[0,p(x)]}(u) \\ p(x|\, u) &= \mathcal{U}_{A}(x) \end{aligned}$$

where $A = \{x; p(x) \geq u\}$. If A is easy to identify then the algorithm is straightforward to implement, as shown in Figure 15.

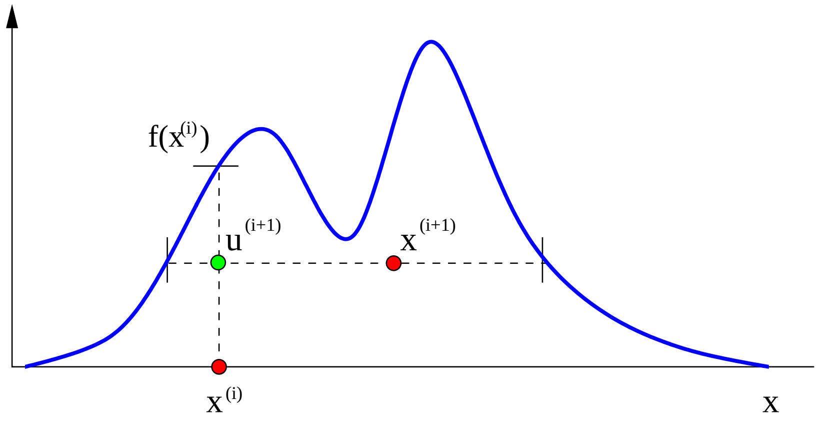

Figure 15. Slice sampling: given a previous sample, we sample a uniform variable $u^{(i+1)}$ between 0 and $f(x^{(i)})$. One then samples $x^{(i+1)}$ in the interval where $f(x) \geq u^{(i+1)}$.

It can be difficult to identify A. It is then worth introducing several auxiliary variables (Damien et al. 1999, Higdon 1998). For example assume that

$$p(x) \propto \prod_{l=1}^{L} f_l(x),$$

where the $f_l(\cdot)$'s are positive functions, not necessarily densities. Let us introduce L auxiliary variables $(u_1, \ldots, u_L)$ and define

$$p^{\star}(x, u_1, \ldots, u_L) \propto \prod_{l=1}^{L} \mathbb{I}_{[0,f_l(x)]}(u_l).$$

Then one can also check that $\int p^{\star}(x, u_1, \ldots, u_L) du_1 \ldots du_L = p(x)$ as

$$\int p^{\star}(x, u_1, \ldots, u_L) du_1 \ldots du_L \propto \int \prod_{l=1}^{L} \mathbb{I}_{[0, f_l(x)]}(u_l) du_1 \ldots du_L = \prod_{l=1}^{L} f_l(x).$$

The slice sampler to sample from $p^{\star}(x, u_1, \ldots, u_L)$ proceeds as shown in Figure 16. Algorithmic improvements and convergence results are presented in (Mira 1999, Neal 2000).

1. For $l = 1, \ldots, L$
   - Sample $u_l^{(i)} \sim \mathcal{U}_{\left[0, f_l\left(x^{(i-1)}\right)\right]}(u_l)$.
2. Sample $x^{(i)} \sim \mathcal{U}_{A^{(i)}}(x)$ where $A^{(i)} = \left\{ x; f_l(x) \geq u_l^{(i)}, l = 1, \ldots, L \right\}$.

Figure 16. Slice Sampler.

3.7. Reversible jump MCMC

In this section, we attack the more complex problem of model selection. Typical examples include estimating the number of neurons in a neural network (Andrieu, de Freitas and Doucet 1999a, Holmes and Mallick 1998, Rios Insua and Müller 1998), the number of splines in a multivariate adaptive splines regression (MARS) model (Holmes and Denison 2001: this issue), the number of sinusoids in a noisy signal (Andrieu and Doucet 1999), the number of lags in an autoregressive process (Troughton and Godsill 1998), the number of components in a mixture (Richardson and Green 1997), the number of levels in a change-point process (Green 1995), the number of components in a mixture of factor analysers (Fokoué and Titterington 2001: this issue), the appropriate structure of a graphical model (Friedman and Koller 2001, Giudici and Castelo 2001: this issue) or the best set of input variables (Lee 2001: this issue).

Given a family of M models $\{\mathcal{M}_m; m = 1, \ldots, N\}$, we will focus on constructing ergodic Markov chains admitting $p(m, x_m)$ as the invariant distribution. For simplicity, we avoid the treatment of nonparametric model averaging techniques; see for example (Escobar and West 1995, Green and Richardson 2000).

Up to this section, we have been comparing densities in the acceptance ratio. However, if we are carrying out model selection, then comparing the densities of objects in different dimensions has no meaning. It is like trying to compare spheres with circles. Instead, we have to be more formal and compare distributions $P(dx) = \Pr(x \in dx)$ under a common measure of volume. The distribution $P(dx)$ will be assumed to admit a density $p(x)$ with respect to a measure of interest, e.g. Lebesgue in the continuous case: $P(dx) = p(x)dx$. The acceptance ratio will now include the ratio of the densities and the ratio of the measures (Radon Nikodym derivative). The latter gives rise to a Jacobian term. To compare densities point-wise, we need, therefore, to map the two models to a common dimension as illustrated in Figure 17.

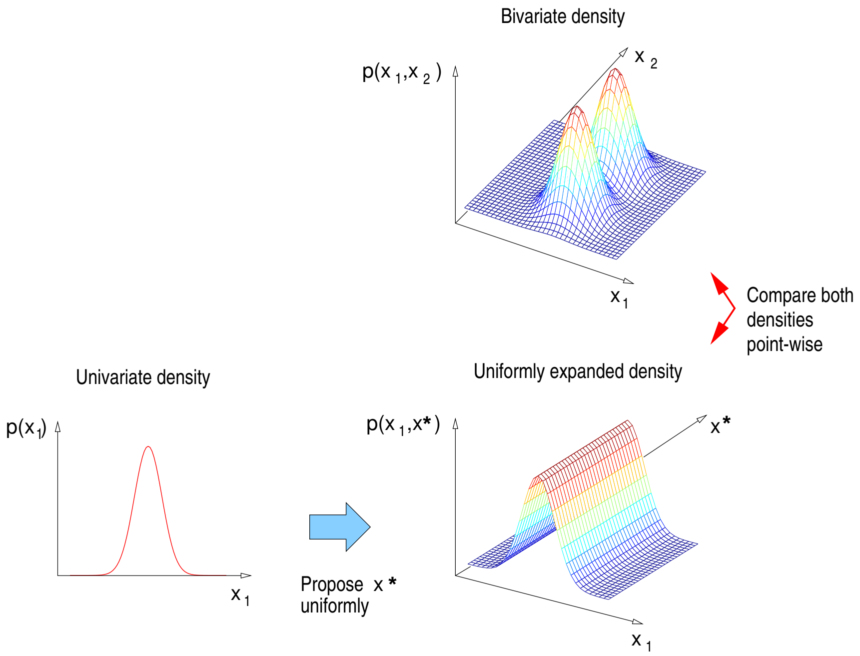

Figure 17. To compare a 1D model against a 2D model, we first have to map the first model so that both models have common measure (area in this case).

The parameters $x_m \in \mathcal{X}_m$ (*e.g.* $\mathcal{X}_m = \mathbb{R}^{n_m}$) are model dependent. Hence, to find the right model and parameters we could sample over the model indicator and the product space $\prod_{m=1}^{M} \mathcal{X}_m$ (Carlin and Chib 1995). Recently, Green introduced a strategy that avoids this expensive search over the full product space (Green 1995). In particular one samples on a much smaller union space $\mathcal{X} \triangleq \cup_{m=1}^{M} \{m\} \times \mathcal{X}_m$. The full target distribution defined in this space is given by

$$p(k, dx) = \sum_{m=1}^{M} p(m, dx_m) \mathbb{I}_{\{m\} \times \mathcal{X}_m}(k, x).$$

That is, the probability of k being equal to m and x belonging to an infinitesimal set centred around x_m is $p(m, dx_m)$. By marginalisation, we obtain the probability of being in subspace $\mathcal{X}_m$.

Green's method allows the sampler to jump between the different subspaces. To ensure a common measure, it requires the extension of each pair of communicating spaces, $\mathcal{X}_m$ and $\mathcal{X}_n$, to $\overline{\mathcal{X}}_{m,n} \triangleq \mathcal{X}_m \times \mathcal{U}_{m,n}$ and $\overline{\mathcal{X}}_{n,m} \triangleq \mathcal{X}_n \times \mathcal{U}_{n,m}$. It also requires the definition of a deterministic, differentiable, invertible dimension matching function $f_{n \to m}$ between $\overline{\mathcal{X}}_{m,n}$ and $\overline{\mathcal{X}}_{n,m}$,

$$(x_m, u_{m,n}) = f_{n \to m}(x_n, u_{n,m}) = (f^x_{n \to m}(x_n, u_{n,m}), f^u_{n \to m}(x_n, u_{n,m})).$$

We define $f_{m \to n}$ such that $f_{m \to n}(f_{n \to m}(x_n, u_{n,m})) = (x_n, u_{n,m})$. The choice of the extended spaces, deterministic transformation $f_{m \to n}$ and proposal distributions for $q_{n \to m}(\cdot |\, n, x_n)$ and $q_{m \to n}(\cdot |\, m, x_m)$ is problem dependent and needs to be addressed on a case by case basis.

If the current state of the chain is (n, x_n), we move to (m, x_m) by generating $u_{n,m} \sim q_{n \to m}(\cdot | n, x_n)$, ensuring that we have reversibility $(x_m, u_{m,n}) = f_{n \to m}(x_n, u_{n,m})$, and accepting the move according to the probability ratio

$$\mathcal{A}_{n \to m} = \min\left\{1, \frac{p(m, x^\star_m)}{p(n, x_n)} \times \frac{q(n|m)}{q(m|n)} \times \frac{q_{m \to n}(u_{m,n}|m, x^\star_m)}{q_{n \to m}(u_{n,m}|n, x_n)} \times \mathcal{J}_{f_{n \to m}}\right\},$$

where $x^\star_m = f^x_{n \to m}(x_n, u_{n,m})$ and $\mathcal{J}_{f_{n \to m}}$ is the Jacobian of the transformation $f_{n \to m}$ (when only continuous variables are involved in the transformation)

$$\mathcal{J}_{f_{m \to n}} = \left| \det \frac{\partial f_{n \to m}(x_m, u_{m,n})}{\partial (x_m, u_{m,n})} \right|.$$

To illustrate this, assume that we are concerned with sampling the locations μ and number k of components of a mixture. For example we might want to estimate the locations and number of basis functions in kernel regression and classification, the number of mixture components in a finite mixture model, or the location and number of segments in a segmentation problem. Here, we could define a merge move that combines two nearby components and a split move that breaks a component into two nearby ones. The merge move involves randomly selecting a component (μ_1) and then combining it with its closest neighbour (μ_2) into a single component μ, whose new location is

$$\mu = \frac{\mu_1 + \mu_2}{2}$$

The corresponding split move that guarantees reversibility, involves splitting a randomly chosen component as follows

$$\begin{aligned}\mu_1 &= \mu - u_{n,m}\beta \\ \mu_2 &= \mu + u_{n,m}\beta\end{aligned}$$

where β is a simulation parameter and, for example, $u_{n,m} \sim \mathcal{U}_{[0,1]}$. Note that to ensure reversibility, we only perform the merge move if $\|\mu_1 - \mu_2\| < 2\beta$. The acceptance ratio for the split move is

$$\mathcal{A}_{split} = \min\left\{1, \frac{p(k+1, \mu_{k+1})}{p(k, \mu_k)} \times \frac{\frac{1}{k+1}}{\frac{1}{k}} \times \frac{1}{p(u_{n,m})} \times \mathcal{J}_{split}\right\},$$

where $1/k$ denotes the probability of choosing, uniformly at random, one of the k components. The Jacobian is

$$\mathcal{J}_{split} = \left|\frac{\partial(\mu_1, \mu_2)}{\partial(\mu, u_{n,m})}\right| = \begin{vmatrix} 1 & 1 \\ -\beta & \beta \end{vmatrix} = 2\beta.$$

Similarly, for the merge move, we have

$$\mathcal{A}_{merge} = \min\left\{1, \frac{p(k-1, \mu_{k-1})}{p(k, \mu_k)} \times \frac{\frac{1}{k-1}}{\frac{1}{k}} \times \mathcal{J}_{merge}\right\},$$

where $\mathcal{J}_{merge} = 1/2\beta$.

Reversible jump is a mixture of MCMC kernels (moves). In addition, to the split and merge moves, we could have other moves such as birth of a component, death of a component and a simple update of the locations. The various moves are carried out according to the mixture probabilities $(b_k, d_k, m_k, s_k, u_k)$, as shown in Figure 18. In fact, it is the flexibility of including so many possible moves that can make reversible jump a more powerful model selection strategy than schemes based on model selection using a mixture indicator or diffusion processes using only birth and death moves (Stephens 1997). However, the problem with reversible jump MCMC is that engineering reversible moves is a very tricky, time-consuming task.

1. Initialisation: set $(k^{(0)}, \mu^{(0)})$.
2. For $i = 0$ to $N-1$
   - Sample $u \sim \mathcal{U}_{[0,1]}$.
   - If $(u \leq b_k)$
     - then "birth" move.
     - else if $(u \leq b_k + d_k)$ then "death" move.
     - else if $(u \leq b_k + d_k + s_k)$ then "split" move.
     - else if $(u \leq b_k + d_k + s_k + m_k)$ then "merge" move.
     - else update.

     End If.
   - Sample other parameters.

Figure 18. Generic reversible jump MCMC.

4. The MCMC Frontiers

4.1. Convergence and Perfect Sampling

Determining the length of the Markov chain is a difficult task. In practice, one often discards an initial set of samples (*burn-in*) to avoid starting biases. In addition, one can apply several graphical and statistical tests to assess, roughly, if the chain has stabilised (Robert and Casella 1999, Chapter 8). In general, none of these tests provide entirely satisfactory diagnostics.

Several theoreticians have tried to bound the *mixing time*; that is, the minimum number of steps required for the distribution of the Markov chain K to be close to the target $p(x)$. (Here, we present a, by no means exhaustive, summary of some of the available results.) If we measure closeness with the total variation norm $\Delta_x(t)$, where

$$\Delta_x(t) = \|K^{(t)}(\cdot|x) - p(\cdot)\| = \frac{1}{2}\int \left(K^{(t)}(y|x) - p(y)\right) dy,$$

then the mixing time is

$$\tau_x(\epsilon) = \min\left\{t : \Delta_x(t') \leq \epsilon \text{ for all } t' \geq t\right\}.$$

If the state space $\mathcal{X}$ is finite and reversibility holds true, then the transition operator K ($Kf(x) = \sum K(y|x)f(y)$) is self adjoint on $L_2(p)$.

That is,

$$\langle Kf|g\rangle = \langle f|Kg\rangle,$$

where f and g are real functions and we have used the bra-ket notation for the inner product $\langle f|g\rangle = \sum f(x)g(x)p(x)$. This implies that K has real eigenvalues

$$1 = \lambda_1 > \lambda_2 \geq \lambda_3 \geq \cdots \geq \lambda_{|\mathcal{X}|} > -1$$

and an orthonormal basis of real eigenfunctions f_i, such that $Kf_i = \lambda_i f_i$. This spectral decomposition and the Cauchy-Schwartz inequality allow us to obtain a bound on the total variation norm

$$\Delta_x(t) \leq \frac{1}{2\sqrt{p(x)}}\lambda_\star^t,$$

where $\lambda_\star = \max(\lambda_2, |\lambda_{|\mathcal{X}|}|)$ (Diaconis and Saloff-Coste 1998, Jerrum and Sinclair 1996). This classical result give us a geometric convergence rate in terms of eigenvalues. Geometric bounds have also been obtained in general state spaces using the tools of regeneration and Lyapunov-Foster conditions (Meyn and Tweedie 1993).

The next logical step is to bound the second eigenvalue. There are several inequalities (Cheeger, Poincaré, Nash) from differential geometry that allows us to obtain these bounds (Diaconis and Saloff-Coste 1998). For example, one could use Cheeger's inequality to obtain the following bound

$$1 - 2\Phi \leq \lambda_2 \leq 1 - \frac{\Phi^2}{2},$$

where Φ is the *conductance* of the Markov chain

$$\Phi = \min_{0<p(S)<1/2; S\subset\mathcal{X}} \frac{\sum_{x\in S, y\in S^c} p(x)K(y|x)}{p(S)}$$

Intuitively, one can interpret this quantity as the readiness of the chain to escape from any small region S of the state space and, hence, make rapid progress towards equilibrium (Jerrum and Sinclair 1996).

These mathematical tools have been applied to show that simple MCMC algorithms (mostly Metropolis) run in time that is polynomial in the dimension d of the state space, thereby escaping the exponential curse of dimensionality. Polynomial time sampling algorithms have been obtained in the following important scenarios:

1. Computing the volume of a convex body in d dimensions, where d is large (Dyer et al. 1991).

2. Sampling from log-concave distributions (Applegate and Kannan 1991).

3. Sampling from truncated multivariate Gaussians (Kannan and Li 1996).

4. Computing the permanent of a matrix (Jerrum, Sinclair and Vigoda 2000).

The last problem is equivalent to sampling matchings from a bipartite graph; a problem that manifests itself in many ways in machine learning (*e.g.*, stereo matching and data association).

Although the theoretical results are still far from the practice of MCMC, they will eventually provide better guidelines on how to design and choose algorithms. Already, some results tell us, for example, that it is not wise to use the independent Metropolis sampler in high dimensions (Mengersen and Tweedie 1996).

A remarkable recent breakthrough was the development of algorithms for *perfect sampling.* These algorithms are guaranteed to give us an independent sample from $p(x)$ under certain restrictions. The two major players are *coupling from the past* (Propp and Wilson 1998) and Fill's algorithm (Fill 1998). From a practical point of view, these algorithms are still limited and, in many cases, computationally inefficient. However, some steps are being taken towards obtaining more general perfect samplers; for example perfect slice samplers (Casella, Mengersen, Robert and Titterington 1999).

4.2. Adaptive MCMC

If we look at the chain on the top right of Figure 7, we notice that the chain stays at each state for a long time. This tells us that we should reduce the variance of the proposal distribution. Ideally, one would like to automate this process of choosing the proposal distribution as much as possible. That is, one should use the information in the samples to update the parameters of the proposal distribution so as to obtain a distribution that is either closer to the target distribution, that ensures a suitable acceptance rate, or that minimises the variance of the estimator of interest. However, one should not allow adaptation to take place infinitely often in a naive way because this can disturb the stationary distribution. This problem arises because by using the past information infinitely often, we violate the Markov property of the transition kernel. That is, $p(x^{(i)}|x^{(0)}, x^{(1)}, \ldots, x^{(i-1)})$ no longer simplifies to $p(x^{(i)}|x^{(i-1)})$. In particular, Gelfand and Sahu (1994) present a pathological example, where the stationary distribution is disturbed despite the fact that each participating kernel has the same stationary distribution.

To avoid this problem, one could carry out adaptation only during an initial fixed number of steps, and then use standard MCMC simulation to ensure convergence to the right distribution. Two methods for doing this are presented in (Gelfand and Sahu 1994). The first is based on the idea of running several chains in parallel and using sampling-importance resampling (Rubin 1988) to multiply the kernels that are doing well and suppress the others. In this approach, one uses an approximation to the marginal density of the chain as proposal. The second method simply involves monitoring the transition kernel and changing one of its components (for example the proposal distribution) so as to improve mixing. A similar method that guarantees a particular acceptance rate is discussed in (Browne and Draper 2000).

There are, however, a few adaptive MCMC methods that allow one to perform adaptation continuously without disturbing the Markov property, including delayed rejection (Tierney and Mira 1999), parallel chains (Gilks and Roberts 1996) and regeneration (Gilks, Roberts and Sahu 1998, Mykland, Tierney and Yu 1995). These methods are, unfortunately, inefficient in many ways and much more research is required in this exciting area.

4.3. Sequential Monte Carlo and Particle Filters

Sequential Monte Carlo (SMC) methods allow us to carry out on-line approximation of probability distributions using samples (particles). They are very useful in scenarios involving real-time signal processing, where data arrival is inherently sequential. Furthermore, one might wish to adopt a sequential processing strategy to deal with non-stationarity in signals, so that information from the recent past is given greater weighting than information from the distant past. Computational simplicity in the form of not having to store all the data might also constitute an additional motivating factor for these methods.

In the SMC setting, we assume that we have an initial distribution, a dynamic model and measurement model

$$\begin{aligned} &p\left(x_0\right) \\ &p\left(x_t|x_{0:t-1}, y_{1:t-1}\right) \quad \text{for } t \geq 1 \\ &p\left(y_t|x_{0:t}, y_{1:t-1}\right) \quad \text{for } t \geq 1 \end{aligned}$$

We denote by $x_{0:t} \triangleq \{x_0, ..., x_t\}$ and $y_{1:t} \triangleq \{y_1, ..., y_t\}$, respectively, the states and the observations up to time t. Note that we could assume Markov transitions and conditional independence to simplify the model; $p\left(x_t|x_{0:t-1}, y_{1:t-1}\right) = p\left(x_t|x_{t-1}\right)$ and $p\left(y_t|x_{0:t}, y_{1:t-1}\right) = p\left(y_t|x_t\right)$. However, this assumption is not necessary in the SMC framework.

Our aim is to estimate recursively in time the posterior $p\left(x_{0:t}|\, y_{1:t}\right)$ and its associated features including the marginal distribution $p\left(x_t|\, y_{1:t}\right)$,

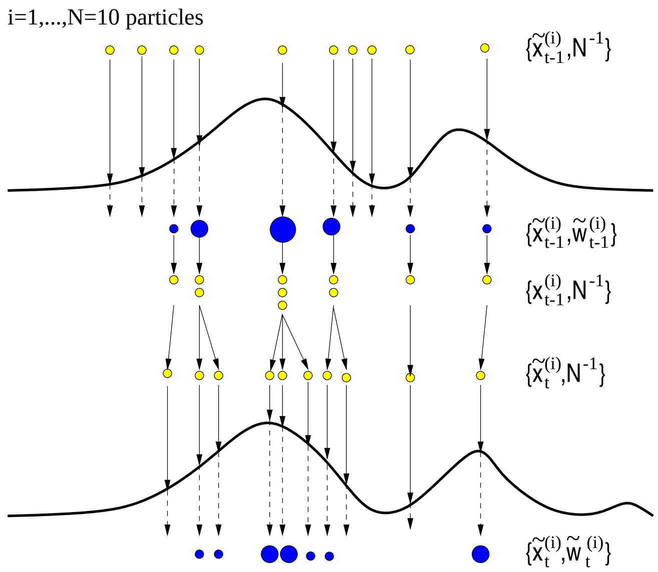

Figure 19. In this example, the bootstrap filter starts at time $t-1$ with an unweighted measure $\{\widetilde{\mathbf{x}}^{(i)}_{t-1}, N^{-1}\}$, which provides an approximation of $p(\mathbf{x}_{t-1}|y_{1:t-2})$. For each particle we compute the importance weights using the information at time $t-1$. This results in the weighted measure $\{\widetilde{\mathbf{x}}^{(i)}_{t-1}, \widetilde{w}^{(i)}_{t-1}\}$, which yields an approximation $p(\mathbf{x}_{t-1}|y_{1:t-1})$. Subsequently, the resampling step selects only the "fittest" particles to obtain the unweighted measure $\{\mathbf{x}^{(i)}_{t-1}, N^{-1}\}$, which is still an approximation of $p(\mathbf{x}_{t-1}|y_{1:t-1})$. Finally, the sampling (prediction) step introduces variety, resulting in the measure $\{\widetilde{\mathbf{x}}^{(i)}_{t}, N^{-1}\}$, which is an approximation of $p(\mathbf{x}_{t}|y_{1:t-1})$.

known as the *filtering distribution*, and the expectations

$$I(f_t) = \mathbb{E}_{p(x_{0:t}|y_{1:t})}[f_t(x_{0:t})]$$

A generic SMC algorithm is depicted in Figure 19. Given N particles $\{x^{(i)}_{0:t-1}\}_{i=1}^N$ at time $t-1$, approximately distributed according to the distribution $p(x_{0:t-1}|y_{1:t-1})$, SMC methods allow us to compute N particles $\{x^{(i)}_{0:t}\}_{i=1}^N$ approximately distributed according to the posterior $p(x_{0:t}|y_{1:t})$, at time t. Since we cannot sample from the posterior directly, the SMC update is accomplished by introducing an appropriate importance proposal distribution $q(x_{0:t})$ from which we can obtain samples. The samples are then appropriately weighted.

In generic SMC simulation, one needs to extend the current paths $\{x^{(i)}_{0:t-1}\}_{i=1}^N$ to obtain new paths $\{\widetilde{x}^{(i)}_{0:t}\}_{i=1}^N$ using the proposal distribution $q(\widetilde{x}_{0:t}|y_{1:t})$ given by

$$q(\widetilde{x}_{0:t}|y_{1:t}) = \int q(\widetilde{x}_{0:t}|x_{0:t-1}, y_{1:t})p(x_{0:t-1}|y_{1:t-1})dx_{0:t-1}.$$

<u>*Sequential importance sampling step*</u>

- For $i = 1, ..., N$, sample from the transition priors

$$\widetilde{x}_t^{(i)} \sim q_t\left(\widetilde{x}_t | x_{0:t-1}^{(i)}, y_{1:t}\right)$$

and set

$$\widetilde{x}_{0:t}^{(i)} \triangleq \left(\widetilde{x}_t^{(i)}, x_{0:t-1}^{(i)}\right)$$

- For $i = 1, ..., N$, evaluate and normalize the importance weights

$$w_t^{(i)} \propto \frac{p\left(y_t | \widetilde{x}_t^{(i)}\right) p\left(\widetilde{x}_t^{(i)} | x_{0:t-1}^{(i)}, y_{1:t-1}\right)}{q_t\left(\widetilde{x}_t^{(i)} | x_{0:t-1}^{(i)}, y_{1:t}\right)}.$$

<u>*Selection step*</u>

- Multiply/Discard particles $\left\{\widetilde{x}_{0:t}^{(i)}\right\}_{i=1}^N$ with high/low importance weights $w_t^{(i)}$ to obtain N particles $\left\{x_{0:t}^{(i)}\right\}_{i=1}^N$.

Figure 20. Simple SMC algorithm at time t. For filtering purposes, there is no need for storing or resampling the past trajectories.

To make this integral tractable, we only propose to modify the particles at time t, and leave the past trajectories intact. Consequently

$$q(\widetilde{x}_{0:t}|y_{1:t}) = p(x_{0:t-1}|y_{1:t-1})q(\widetilde{x}_t|x_{0:t-1}, y_{1:t})$$

The samples from $q(\cdot)$, must be weighted by the importance weights

$$\begin{aligned} w_t &= \frac{p(\widetilde{x}_{0:t}|y_{1:t})}{q(\widetilde{x}_{0:t}|y_{1:t})} = \frac{p(x_{0:t-1}|y_{1:t})}{p(x_{0:t-1}|y_{1:t-1})} \frac{p(\widetilde{x}_t|x_{0:t-1}, y_{1:t})}{q(\widetilde{x}_t|x_{0:t-1}, y_{1:t})} \\ &\propto \frac{p\left(y_t|\widetilde{x}_t\right) p\left(\widetilde{x}_t|x_{0:t-1}, y_{1:t-1}\right)}{q_t\left(\widetilde{x}_t|x_{0:t-1}, y_{1:t}\right)}. \end{aligned} \tag{22}$$

From equation (22), we note that the optimal importance distribution is

$$q(\widetilde{x}_t|x_{0:t-1}, y_{1:t}) = p(\widetilde{x}_t|x_{0:t-1}, y_{1:t}).$$

(When using this proposal, one might still encounter difficulties if the ratio of the first two terms of equation (22) differs significantly from 1 (Andrieu, Doucet and Punskaya 2001, Pitt and Shephard 1999).) The optimal importance distribution can be difficult to evaluate. One can adopt, instead, the transition prior as proposal distribution

$$q(\widetilde{x}_t|x_{0:t-1}, y_{1:t}) = p\left(\widetilde{x}_t|x_{0:t-1}, y_{1:t-1}\right)$$

in which case the importance weights are given by the likelihood function

$$w_t \propto p\left(y_t|\widetilde{x}_t\right).$$

This simplified version of SMC has appeared under many names, including condensation (Isard and Blake 1996), survival of the fittest (Kanazawa, Koller and Russell 1995) and the bootstrap filter (Gordon, Salmond and Smith 1993). The importance sampling framework allows us to design more principled and "clever" proposal distributions. For instance, one can adopt suboptimal filters and other approximation methods that make use of the information available at time t to generate the proposal distribution (Doucet, Godsill and Andrieu 2000, de Freitas et al. 2000, Pitt and Shephard 1999, van der Merwe, Doucet, de Freitas and Wan 2000). In fact, in some restricted situations, one may interpret the likelihood as a distribution in terms of the states and sample from it directly. In doing so, the importance weights become equal to the transition prior (Fox, Thrun, Burgard and Dellaert 2001).

After the importance sampling step, a selection scheme associates to each particle $\widetilde{x}_{0:t}^{(i)}$ a number of "children", say $N_i \in \mathbb{N}$, such that $\sum_{i=1}^{N} N_i = N$. This selection step is what allows us to track moving target distributions efficiently by choosing the fittest particles. There are various selection schemes in the literature, but their performance varies in terms of $var\left[N_i\right]$ (Doucet et al. 2001).

An important feature of the selection routine is that its interface only depends on particle indices and weights. That is, it can be treated as a black-box routine that does not require any knowledge of what a particle represents (*e.g.*, variables, parameters, models). This enables one to implement variable and model selection schemes straightforwardly. The simplicity of the coding of complex models is, indeed, one of the major advantages of these algorithms.

It is also possible to introduce MCMC steps of invariant distribution $p(x_{0:t}|y_{1:t})$ on each particle (Andrieu, de Freitas and Doucet 1999b, Gilks and Berzuini 1998, MacEachern, Clyde and Liu 1999). The basic idea is that if the particles are distributed according to the posterior distribution $p(x_{0:t}|y_{1:t})$, then applying a Markov chain transition kernel $K(x_{0:t}^{\star}|x_{0:t})$, with invariant distribution $p(\cdot|y_{1:t})$ such that $\int K(x_{0:t}^{\star}|x_{0:t})p(x_{0:t}|y_{1:t}) = p(x_{0:t}^{\star}|y_{1:t})$, still results in a set of particles distributed according to the posterior of interest. However, the new particles might have been moved to more interesting areas of the state-space. In fact, by applying a Markov transition kernel, the total variation of the current distribution with respect to the invariant distribution can only decrease. Note that we can incorporate any of the standard MCMC methods, such as the Gibbs sampler, MH algorithm and reversible jump MCMC, into the filtering framework, but we no longer require the kernel to be ergodic.

4.4. The Machine Learning Frontier

The machine learning frontier is characterised by large dimensional models, massive datasets and many and varied applications. Massive datasets pose no problem in the SMC context. However, in batch MCMC simulation it is often not possible to load the entire dataset into memory. A few solutions based on importance sampling have been proposed recently (Ridgeway 1999), but there is still great room for innovation in this area.

Despite the auspicious polynomial bounds on the mixing time, it is an arduous task to design efficient samplers in high dimensions. The combination of sampling algorithms with either gradient optimisation or exact methods has proved to be very useful. Gradient optimisation is inherent to Langevin algorithms and hybrid Monte Carlo. These algorithms have been shown to work with large dimensional models such as neural networks (Neal 1996) and Gaussian processes (Barber and Williams 1997). Information about derivatives of the target distribution also forms an integral part of many adaptive schemes, as discussed in Section 2.3. Recently, it has been argued that the combination of MCMC and variational optimisation techniques can also lead to more efficient sampling (de Freitas et al. 2001).

The combination of exact inference with sampling methods within the framework of Rao-Blackwellisation (Casella and Robert 1996) can also result in great improvements. Suppose we can divide the hidden variables x into two groups, u and v, such that $p(x) = p(v|u)p(u)$ and, conditional on u, the conditional posterior distribution $p(v|u)$ is analytically tractable. Then we can easily marginalise out v from the posterior, and only need to focus on sampling from $p(u)$, which lies in a space of reduced dimension. That is, we sample $u^{(i)} \sim p(u)$ and then use exact inference to compute

$$p(v) = \frac{1}{N} \sum_{i=1}^{N} p(v|u^{(i)})$$

By identifying "troublesome" variables and sampling them, the rest of the problem can often be solved easily using exact algorithms such as Kalman filters, HMMs or junction trees. For example, one can apply this technique to sample variables that eliminate loops in graphical models and then compute the remaining variables with efficient analytical algorithms (Jensen et al. 1995, Wilkinson and Yeung 2001). Other application areas include dynamic Bayesian networks (Doucet, de Freitas, Murphy and Russell 2000), conditionally Gaussian models (Carter and Kohn 1994, De Jong and Shephard 1995, Doucet 1998) and model averaging for graphical models (Friedman and Koller 2001: this issue). The problem of how to automatically identify which variables should be sampled, and which can be handled analytically is still open. An interesting development is the augmentation of high dimensional models with low dimensional artificial variables. By sampling only the artificial variables, the original model decouples into simpler, more tractable submodels (Albert and Chib 1993, Andrieu, de Freitas and Doucet 2001, Wood and Kohn 1998); see also (Holmes and Denison 2001: this issue). This strategy allows one to map probabilistic classification problems to simpler regression problems.

The design of efficient sampling methods most of the times hinges on awareness of the basic building blocks of MCMC (mixtures of kernels, augmentation strategies and blocking) and on careful design of the proposal mechanisms. The latter requires domain specific knowledge and heuristics. There are great opportunities for combining existing sub-optimal algorithms with MCMC in many machine learning problems. Some areas that are already benefiting from sampling methods include:

1. **Computer vision**: Tracking (Isard and Blake 1996, Ormoneit, Lemieux and Fleet 2001), stereo matching (Dellaert, Seitz, Thorpe and Thrun 2001: this issue), colour constancy (Forsyth 1999), restoration of old movies (Morris, Fitzgerald and Kokaram 1996) and segmentation (Clark and Quinn 1999, Kam 2000, Tu and Zhu 2001).

2. **Web statistics**: Estimating coverage of search engines, proportions belonging to specific domains and the average size of web pages (Bar-Yossef, Berg, Chien, Fakcharoenphol and Weitz 2000).

3. **Speech and audio processing**: Signal enhancement (Godsill and Rayner 1998, Vermaak, Andrieu, Doucet and Godsill 1999).

4. **Probabilistic graphical models**: For example (Gilks et al. 1994, Wilkinson and Yeung 2001) and several papers in this issue.

5. **Regression and classification**: Neural networks and kernel machines (Andrieu, de Freitas and Doucet 1999a, Holmes and Mallick 1998, Neal 1996, Müller and Rios Insua 1998), Gaussian processes

(Barber and Williams 1997), CART (Denison, Mallick and Smith 1998) and MARS (Holmes and Denison 2001: this issue).

6. **Computer graphics**: Light transport (Veach and Guibas 1997) and sampling plausible solutions to multi-body constraint problems (Chenney and Forsyth 2000).
7. **Data association**: Vehicle matching in highway systems (Pasula et al. 1999) and multitarget tracking (Bergman 1999).
8. **Decision theory**: Partially observable Markov decision Processes (POMDPs) (Thrun 2000, Salmond and Gordon 2001), abstract Markov policies (Bui, Venkatesh and West 1999) and influence diagrams (Bielza, Müller and Rios Insua 2001).
9. **First order probabilistic logic**: (Pasula and Russell 2001).
10. **Genetics and Molecular biology**: DNA microarray data (West, Nevins, Marks, Spang and Zuzan 2001), cancer gene mapping (Newton and Lee 2000), protein alignment (Neuwald, Liu, Lipman and Lawrence 1997) and linkage analysis (Jensen et al. 1995).
11. **Robotics**: Robot localisation and map building (Fox et al. 2001).
12. **Classical mixture models**: Mixtures of independent factor analysers (Utsugi 2001) and mixtures of factor analysers (Fokoué and Titterington 2001: this issue).

We hope that this review will be a useful resource to people wishing to carry out further research at the interface between MCMC and machine learning. For conciseness, we have skipped many interesting ideas, including tempering and coupling. For more details, we advise the readers to consult the references at the end of this paper.

Acknowledgements

We would like to thank Robin Morris, Kevin Murphy, Mark Paskin, Sekhar Tatikonda and Mike Titterington.

References

Al-Qaq, W. A., Devetsikiotis, M. and Townsend, J. K. (1995). Stochastic gradient optimization of importance sampling for the efficient simulation of digital communication systems, *IEEE Transactions on Communications* **43**(12): 2975–2985.

Albert, J. and Chib, S. (1993). Bayesian analysis of binary and polychotomous response data, *Journal of the American Statistical Association* **88**(422): 669–679.

Anderson, H. L. (1986). Metropolis, Monte Carlo, and the MANIAC, *Los Alamos Science* **14**: 96–108.

Andrieu, C. and Doucet, A. (1999). Joint Bayesian detection and estimation of noisy sinusoids via reversible jump MCMC, *IEEE Transactions on Signal Processing* **47**(10): 2667–2676.

Andrieu, C., Breyer, L. A. and Doucet, A. (1999). Convergence of simulated annealing using Foster-Lyapunov criteria, *Technical Report CUED/F-INFENG/TR 346*, Cambridge University Engineering Department.

Andrieu, C., de Freitas, N. and Doucet, A. (1999a). Robust full Bayesian learning for radial basis networks, To appear in *Neural Computation* Vol 13 No 10.

Andrieu, C., de Freitas, N. and Doucet, A. (1999b). Sequential MCMC for Bayesian model selection, *IEEE Higher Order Statistics Workshop*, Caesarea, Israel, pp. 130–134.

Andrieu, C., de Freitas, N. and Doucet, A. (2000a). Reversible jump MCMC simulated annealing for neural networks, *Uncertainty in Artificial Intelligence*, Morgan Kaufmann, pp. 11–18.

Andrieu, C., de Freitas, N. and Doucet, A. (2000b). Robust full Bayesian methods for neural networks, *in* S. Solla, T. Leen and K.-R. Müller (eds), *Advances in Neural Information Processing Systems 12*, MIT Press, pp. 379–385.

Andrieu, C., de Freitas, N. and Doucet, A. (2001). Rao-blackwellised particle filtering via data augmentation, Under review.

Andrieu, C., Doucet, A. and Punskaya, E. (2001). Sequential Monte Carlo methods for optimal filtering, *in* A. Doucet, N. de Freitas and N. J. Gordon (eds), *Sequential Monte Carlo Methods in Practice*, Springer-Verlag.

Applegate, D. and Kannan, R. (1991). Sampling and integration of near log-concave functions, *Proceedings of the Twenty Third Annual ACM Symposium on Theory of Computing*, pp. 156–163.

Bar-Yossef, Z., Berg, A., Chien, S., Fakcharoenphol, J. and Weitz, D. (2000). Approximating aggregate queries about web pages via random walks, *International Conference on Very Large Databases*, pp. 535–544.

Barber, D. and Williams, C. K. I. (1997). Gaussian processes for Bayesian classification via hybrid Monte Carlo, *in* M. C. Mozer, M. I. Jordan and T. Petsche (eds), *Advances in Neural Information Processing Systems*, Vol. 9, MIT Press, pp. 340–346.

Baum, L. E., Petrie, T., Soules, G. and Weiss, N. (1970). A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains, *Annals of Mathematical Statistics* **41**: 164–171.

Baxter, R. J. (1982). *Exactly Solved Models in Statistical Mechanics*, Academic Press.

Beichl, I. and Sullivan, F. (2000). The Metropolis algorithm, *Computing in Science & Engineering* **2**(1): 65–69.

Bergman, N. (1999). *Recursive Bayesian Estimation: Navigation and Tracking Applications*, PhD thesis, Department of Electrical Engineering, Linköping University, Sweeden.

Berners-Lee, T., Cailliau, R., Luotonen, A., Nielsen, H. F. and Secret, A. (1994). The World-Wide Web, *Communications of the ACM* **10**(4): 49–63.

Besag, J., Green, P. J., Hidgon, D. and Mengersen, K. (1995). Bayesian computation and stochastic systems, *Statistical Science* **10**(1): 3–66.

Bielza, C., Müller, P. and Rios Insua, D. (2001). Monte Carlo methods for decision analysis with applications to influence diagrams, to appear in Management Science.

Brooks, S. P. (1998). Markov chain Monte Carlo method and its application, *The Statistician* **47**(1): 69–100.

Browne, W. J. and Draper, D. (2000). Implementation and performance issues in the Bayesian and likelihood fitting of multilevel models, *Computational Statistics* **15**: 391–420.

Bucher, C. G. (1988). Adaptive sampling — an iterative fast Monte Carlo procedure, *Structural Safety* **5**: 119–126.

Bui, H. H., Venkatesh, S. and West, G. (1999). On the recognition of abstract Markov policies, *National Conference on Artificial Intelligence (AAAI-2000)*.

Carlin, B. P. and Chib, S. (1995). Bayesian Model choice via MCMC, *Journal of the Royal Statistical Society Series B* **57**: 473–484.

Carter, C. K. and Kohn, R. (1994). On Gibbs sampling for state space models, *Biometrika* **81**(3): 541–553.

Casella, G. and Robert, C. P. (1996). Rao-Blackwellisation of sampling schemes, *Biometrika* **83**(1): 81–94.

Casella, G., Mengersen, K. L., Robert, C. P. and Titterington, D. M. (1999). Perfect slice samplers for mixtures of distributions, *Technical Report BU-1453-M*, Department of Biometrics, Cornell University.

Celeux, G. and Diebolt, J. (1985). The SEM algorithm: A probabilistic teacher algorithm derived from the EM algorithm for the mixture problem, *Computational Statistics Quarterly* **2**: 73–82.

Celeux, G. and Diebolt, J. (1992). A stochastic approximation type EM algorithm for the mixture problem, *Stochastics and stochastics reports* **41**: 127–146.

Chen, M. H., Shao, Q. M. and Ibrahim, J. G. (eds) (2001). *Monte Carlo Methods for Bayesian Computation*, Springer-Verlag.

Cheng, J. and Druzdzel, M. J. (2000). AIS-BN: An adaptive importance sampling algorithm for evidential reasoning in large bayesian networks, *Journal of Artificial Intelligence Research*.

Chenney, S. and Forsyth, D. A. (2000). Sampling plausible solutions to multi-body constraint problems, *SIGGRAPH*, pp. 219–228.

Clark, E. and Quinn, A. (1999). A data-driven Bayesian sampling scheme for unsupervised image segmentation, *IEEE International Conference on Acoustics, Speech, and Signal Processing*, Vol. 6, Arizona, pp. 3497–3500.

Damien, P., Wakefield, J. and Walker, S. (1999). Gibbs sampling for Bayesian non-conjugate and hierarchical models by auxiliary variables, *Journal of the Royal Statistical Society B* **61**(2): 331–344.

de Freitas, N., Højen-Sørensen, P., Jordan, M. I. and Russell, S. (2001). Variational MCMC, *in* J. Breese and D. Koller (eds), *Uncertainty in Artificial Intelligence*, Morgan Kaufmann, pp. 120–127.

de Freitas, N., Niranjan, M., Gee, A. H. and Doucet, A. (2000). Sequential Monte Carlo methods to train neural network models, *Neural Computation* **12**(4): 955–993.

De Jong, P. and Shephard, N. (1995). Efficient sampling from the smoothing density in time series models, *Biometrika* **82**(2): 339–350.

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

Denison, D. G. T., Mallick, B. K. and Smith, A. F. M. (1998). A Bayesian CART algorithm, *Biometrika* **85**: 363–377.

Diaconis, P. and Saloff-Coste, L. (1998). What do we know about the Metropolis algorithm?, *Journal of Computer and System Sciences* **57**: 20–36.

Doucet, A. (1998). On sequential simulation-based methods for Bayesian filtering, *Technical Report CUED/F-INFENG/TR 310*, Department of Engineering, Cambridge University.

Doucet, A., de Freitas, N. and Gordon, N. J. (eds) (2001). *Sequential Monte Carlo Methods in Practice*, Springer-Verlag.

Doucet, A., de Freitas, N., Murphy, K. and Russell, S. (2000). Rao blackwellised particle filtering for dynamic Bayesian networks, *in* C. Boutilier and M. Godszmidt (eds), *Uncertainty in Artificial Intelligence*, Morgan Kaufmann Publishers, pp. 176–183.

Doucet, A., Godsill, S. and Andrieu, C. (2000). On sequential Monte Carlo sampling methods for Bayesian filtering, *Statistics and Computing* **10**(3): 197–208.

Doucet, A., Godsill, S. J. and Robert, C. P. (2000). Marginal maximum a posteriori estimation using MCMC, *Technical Report CUED/F-INFENG/TR 375*, Cambridge University Engineering Department.

Duane, S., Kennedy, A. D., Pendleton, B. J. and Roweth, D. (1987). Hybrid Monte Carlo, *Physics Letters B* **195**(2): 216–222.

Dyer, M., Frieze, A. and Kannan, R. (1991). A random polynomial-time algorithm for approximating the volume of convex bodies, *Journal of the ACM* **1**(38): 1–17.

Eckhard, R. (1987). Stan Ulam, John Von Neumann and the Monte Carlo method, *Los Alamos Science* **15**: 131–136.

Escobar, M. D. and West, M. (1995). Bayesian density estimation and inference using mixtures, *Journal of the American Statistical Association* **90**: 577–588.

Fill, J. A. (1998). An interruptible algorithm for perfect sampling via Markov chains, *The Annals of Applied Probability* **8**(1): 131–162.

Forsyth, D. A. (1999). Sampling, resampling and colour constancy, *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 300–305.

Fox, D., Thrun, S., Burgard, W. and Dellaert, F. (2001). Particle filters for mobile robot localization, *in* A. Doucet, N. de Freitas and N. J. Gordon (eds), *Sequential Monte Carlo Methods in Practice*, Springer-Verlag.

Gelfand, A. E. and Sahu, S. K. (1994). On Markov chain Monte Carlo acceleration, *Journal of Computational and Graphical Statistics* **3**: 261–276.

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

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

Geweke, J. (1989). Bayesian inference in econometric models using Monte Carlo integration, *Econometrica* **24**: 1317–1399.

Ghahramani, Z. (1995). Factorial learning and the EM algorithm, *in* G. Tesauro, D. S. Touretzky and J. Alspector (eds), *Advances in Neural Information Processing Systems*, Vol. 7, pp. 617–624.

Ghahramani, Z. and Jordan, M. (1995). Factorial Hidden Markov Models, *Technical Report 9502*, MIT Artificial Intelligence Lab, MA.

Gilks, W. R. and Berzuini, C. (1998). Monte Carlo inference for dynamic Bayesian models, Unpublished. Medical Research Council, Cambridge, UK.

Gilks, W. R. and Roberts, G. O. (1996). Strategies for improving MCMC, *in* W. R. Gilks, S. Richardson and D. J. Spiegelhalter (eds), *Markov chain Monte Carlo in Practice*, Chapman & Hall, pp. 89–114.

Gilks, W. R., Richardson, S. and Spiegelhalter, D. J. (eds) (1996). *Markov Chain Monte Carlo in Practice*, Chapman and Hall, Suffolk.

Gilks, W. R., Roberts, G. O. and Sahu, S. K. (1998). Adaptive Markov chain Monte Carlo through regeneration, *Journal of the American Statistical Association* **93**: 763–769.

Gilks, W. R., Thomas, A. and Spiegelhalter, D. J. (1994). A language and program for complex Bayesian modelling, *The Statistician* **43**: 169–178.

Godsill, S. J. and Rayner, P. J. W. (eds) (1998). *Digital Audio Restoration : A Statistical Model Based Approach*, Springer-Verlag.

Gordon, N. J., Salmond, D. J. and Smith, A. F. M. (1993). Novel approach to nonlinear/non-Gaussian Bayesian state estimation, *IEE Proceedings-F* **140**(2): 107–113.

Green, P. J. (1995). Reversible jump Markov chain Monte Carlo computation and Bayesian model determination, *Biometrika* **82**: 711–732.

Green, P. J. and Richardson, S. (2000). Modelling heterogeneity with and without the Dirichlet process, Department of Statistics, Bristol University.

Haario, H. and Sacksman, E. (1991). Simulated annealing process in general state space, *Advances in Applied Probability* **23**: 866–893.

Hastings, W. K. (1970). Monte Carlo sampling methods using Markov chains and their Applications, *Biometrika* **57**: 97–109.

Higdon, D. M. (1998). Auxiliary variable methods for Markov chain Monte Carlo with application, *Journal of American Statistical Association* **93**(442): 585–595.

Holmes, C. C. and Mallick, B. K. (1998). Bayesian radial basis functions of variable dimension, *Neural Computation* **10**(5): 1217–1233.

Isard, M. and Blake, A. (1996). Contour tracking by stochastic propagation of conditional density, *European Conference on Computer Vision*, Cambridge, UK, pp. 343–356.

Ishwaran, H. (1999). Application of hybrid Monte Carlo to Bayesian generalized linear models: Quasicomplete separation and neural networks, *Journal of Computational and Graphical Statistics* **8**: 779–799.

Jensen, C. S., Kong, A. and Kjærulff, U. (1995). Blocking-Gibbs sampling in very large probabilistic expert systems, *International Journal of Human-Computer Studies* **42**: 647–666.

Jerrum, M. and Sinclair, A. (1996). The Markov chain Monte Carlo method: an approach to approximate counting and integration, *in* D. S. Hochbaum (ed.), *Approximation Algorithms for NP-hard Problems*, PWS Publishing, pp. 482–519.

Jerrum, M., Sinclair, A. and Vigoda, E. (2000). A polynomial-time approximation algorithm for the permanent of a matrix, *Technical Report TR00-079*, Electronic Colloquium on Computational Complexity.

Kalos, M. H. and Whitlock, P. A. (1986). *Monte Carlo Methods*, John Wiley & Sons.

Kam, A. H. (2000). *A General Multiscale Scheme for Unsupervised Image Segmentation*, PhD thesis, Department of Engineering, Cambridge University, Cambridge, UK.

Kanazawa, K., Koller, D. and Russell, S. (1995). Stochastic simulation algorithms for dynamic probabilistic networks, *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*, Morgan Kaufmann, pp. 346–351.

Kannan, R. and Li, G. (1996). Sampling according to the multivariate normal density, *37th Annual Symposium on Foundations of Computer Science*, IEEE, pp. 204–212.

Kirkpatrick, S., Gelatt, C. D. and Vecchi, M. P. (1983). Optimization by simulated annealing, *Science* **220**: 671–680.

Levine, R. and Casella, G. (2001). Implementations of the Monte Carlo EM algorithm, To appear in the Journal of Computational and Graphical Statistics.

Liu, J. S. (ed.) (2001). *Monte Carlo Strategies in Scientific Computing*, Springer-Verlag.

MacEachern, S. N., Clyde, M. and Liu, J. S. (1999). Sequential importance sampling for nonparametric Bayes models: the next generation, *Canadian Journal of Statistics* **27**: 251–267.

McCulloch, C. E. (1994). Maximum likelihood variance components estimation for binary data, *Journal of the American Statistical Association* **89**(425): 330–335.

Mengersen, K. L. and Tweedie, R. L. (1996). Rates of convergence of the Hastings and Metropolis algorithms, *The Annals of Statistics* **24**: 101–121.

Metropolis, N. and Ulam, S. (1949). The Monte Carlo method, *Journal of the American Statistical Association* **44**(247): 335–341.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H. and Teller, E. (1953). Equations of state calculations by fast computing machines, *Journal of Chemical Physics* **21**: 1087–1091.

Meyn, S. P. and Tweedie, R. L. (1993). *Markov Chains and Stochastic Stability*, Springer-Verlag, New York.

Mira, A. (1999). *Ordering, Slicing and Splitting Monte Carlo Markov Chains*, PhD thesis, School of Statistics, University of Minnesota.

Morris, R. D., Fitzgerald, W. J. and Kokaram, A. C. (1996). A sampling based approach to line scratch removal from motion picture frames, *IEEE International Conference on Image Processing*, pp. 801–804.

Müller, P. and Rios Insua, D. (1998). Issues in Bayesian analysis of neural network models, *Neural Computation* **10**: 571–592.

Mykland, P., Tierney, L. and Yu, B. (1995). Regeneration in Markov chain samplers, *Journal of the American Statistical Association* **90**: 233–241.

Neal, R. M. (1993). Probabilistic inference using markov chain monte carlo methods, *Technical Report CRG-TR-93-1*, Dept. of Computer Science, University of Toronto.

Neal, R. M. (1996). *Bayesian Learning for Neural Networks*, Lecture Notes in Statistics No. 118, Springer-Verlag, New York.

Neal, R. M. (2000). Slice sampling, *Technical Report No. 2005*, Department of Statistics, University of Toronto.

Neuwald, A. F., Liu, J. S., Lipman, D. J. and Lawrence, C. E. (1997). Extracting protein alignment models from the sequence database, *Nucleic Acids Research* **25**(9): 1665–1677.

Newton, M. A. and Lee, Y. (2000). Inferring the location and effect of tumor suppressor genes by instability-selection modeling of allelic-loss data, *Biometrics* **56**: 1088–1097.

Ormoneit, D., Lemieux, C. and Fleet, D. (2001). Lattice particle filters, *Uncertainty in Artificial Intelligence*, Morgan Kaufmann.

Ortiz, L. E. and Kaelbling, L. P. (2000). Adaptive importance sampling for estimation in structured domains, *in* C. Boutilier and M. Godszmidt (eds), *Uncertainty in Artificial Intelligence*, Morgan Kaufmann Publishers, pp. 446–454.

Page, L., Brin, S., Motwani, R. and Winograd, T. (1998). The PageRank citation ranking: Bringing order to the Web, Stanford Digital Libraries Working Paper.

Pasula, H. and Russell, S. (2001). Approximate inference for first-order probabilistic languages, *International Joint Conference on Artificial Intelligence*, Seattle.

Pasula, H., Russell, S., Ostland, M. and Ritov, Y. (1999). Tracking many objects with many sensors, *International Joint Conference on Artificial Intelligence*, Stockholm.

Pearl, J. (1987). Evidential reasoning using stochastic simulation, *Artificial Intelligence* **32**: 245–257.

Peskun, P. H. (1973). Optimum Monte-Carlo sampling using Markov chains, *Biometrika* **60**(3): 607–612.

Pitt, M. K. and Shephard, N. (1999). Filtering via simulation: Auxiliary particle filters, *Journal of the American Statistical Association* **94**(446): 590–599.

Propp, J. and Wilson, D. (1998). Coupling from the past: a user's guide, *in* D. Aldous and J. Propp (eds), *Microsurveys in Discrete Probability, DIMACS Series in Discrete Mathematics and Theoretical Computer Science*.

Remondo, D., Srinivasan, R., Nicola, V. F., van Etten, W. C. and Tattje, H. E. P. (2000). Adaptive importance sampling for performance evaluation and parameter optimization of communications systems, *IEEE Transactions on Communications* **48**(4): 557–565.

Richardson, S. and Green, P. J. (1997). On Bayesian analysis of mixtures with an unknown number of components, *Journal of the Royal Statistical Society B* **59**(4): 731–792.

Ridgeway, G. (1999). *Generalization of Boosting Algorithms and Applications of Bayesian Inference for Massive Datasets*, PhD thesis, Department of Statistics, University of Washington.

Rios Insua, D. and Müller, P. (1998). Feedforward neural networks for nonparametric regression, *in* D. K. Dey, P. Müller and D. Sinha (eds), *Practical Nonparametric and Semiparametric Bayesian Statistics*, Springer Verlag, pp. 181–191.

Robert, C. P. and Casella, G. (1999). *Monte Carlo Statistical Methods*, Springer-Verlag, New York.

Roberts, G. and Tweedie, R. (1996). Geometric convergence and central limit theorems for multidimensional Hastings and Metropolis algorithms, *Biometrika* **83**: 95–110.

Rubin, D. B. (1988). Using the SIR algorithm to simulate posterior distributions, *in* J. M. Bernardo, M. H. DeGroot, D. V. Lindley and A. F. M. Smith (eds), *Bayesian Statistics 3*, Oxford University Press, Cambridge, MA, pp. 395–402.

Rubinstein, R. Y. (ed.) (1981). *Simulation and the Monte Carlo Method*, John Wiley and Sons.

Salmond, D. and Gordon, N. (2001). Particles and mixtures for tracking and guidance, *in* A. Doucet, N. de Freitas and N. J. Gordon (eds), *Sequential Monte Carlo Methods in Practice*, Springer-Verlag.

Schuurmans, D. and Southey, F. (2000). Monte Carlo inference via greedy importance sampling, *in* C. Boutilier and M. Godszmidt (eds), *Uncertainty in Artificial Intelligence*, Morgan Kaufmann Publishers, pp. 523–532.

Sherman, R. P., Ho, Y. K. and Dalal, S. R. (1999). Conditions for convergence of Monte Carlo EM sequences with an application to product diffusion modeling, *Econometrics Journal* **2**(2): 248–267.

Smith, P. J., Shafi, M. and Gao, H. (1997). Quick simulation: A review of importance sampling techniques in communications systems, *IEEE Journal on Selected Areas in Communications* **15**(4): 597–613.

Stephens, M. (1997). *Bayesian Methods for Mixtures of Normal Distributions*, PhD thesis, Department of Statistics, Oxford University, England.

Swendsen, R. H. and Wang, J. S. (1987). Nonuniversal critical dynamics in Monte Carlo simulations, *Physical Review Letters* **58**(2): 86–88.

Tanner, M. A. and Wong, W. H. (1987). The calculation of posterior distributions by data augmentation, *Journal of the American Statistical Association* **82**(398): 528–550.

Thrun, S. (2000). Monte Carlo POMDPs, *in* S. Solla, T. Leen and K.-R. Müller (eds), *Advances in Neural Information Processing Systems 12*, MIT Press, pp. 1064–1070.

Tierney, L. (1994). Markov chains for exploring posterior distributions, *The Annals of Statistics* **22**(4): 1701–1762.

Tierney, L. and Mira, A. (1999). Some adaptive Monte Carlo methods for Bayesian inference, *Statistics in Medicine* **18**: 2507–2515.

Troughton, P. T. and Godsill, S. J. (1998). A reversible jump sampler for autoregressive time series, *International Conference on Acoustics, Speech and Signal Processing*, Vol. IV, pp. 2257–2260.

Tu, Z. W. and Zhu, S. C. (2001). Image segmentation by data driven Markov Chain Monte Carlo, *International Computer Vision Conference*.

Utsugi, A. (2001). Ensemble of independent factor analyzers with application to natural image analysis, *Neural Processing Letters* **14**(1): 49–60.

van der Merwe, R., Doucet, A., de Freitas, N. and Wan, E. (2000). The unscented particle filter, *Technical Report CUED/F-INFENG/TR 380*, Cambridge University Engineering Department.

Van Laarhoven, P. J. and Arts, E. H. L. (1987). *Simulated Annealing: Theory and Applications*, Reidel Publishers, Amsterdam.

Veach, E. and Guibas, L. J. (1997). Metropolis light transport, *SIGGRAPH*, Vol. 31, pp. 65–76.

Vermaak, J., Andrieu, C., Doucet, A. and Godsill, S. J. (1999). Non-stationary Bayesian modelling and enhancement of speech signals, *Technical Report CUED/F-INFENG/TR*, Cambridge University Engineering Department.

Wakefield, J. C., Gelfand, A. E. and Smith, A. F. M. (1991). Efficient generation of random variates via the ratio-of-uniforms methods, *Statistics and Computing* **1**: 129–133.

Wei, G. C. G. and Tanner, M. A. (1990). A Monte Carlo implementation of the EM algorithm and the poor man's data augmentation algorithms, *Journal of the American Statistical Association* **85**(411): 699–704.

West, M., Nevins, J. R., Marks, J. R., Spang, R. and Zuzan, H. (2001). Bayesian regression analysis in the "large p, small n" paradigm with application in DNA microarray studies, Department of Statistics, Duke University.

Wilkinson, D. J. and Yeung, S. K. H. (2001). Conditional simulation from highly structured gaussian systems, with application to blocking-MCMC for the Bayesian analysis of very large linear models, *Statistics and Computing* **11**: To appear.

Wood, S. and Kohn, R. (1998). A Bayesian approach to robust binary nonparametric regression, *Journal of the American Statistical Association* **93**(441): 203–213.