Stat 8931 Fall 2005 Class Notes

Some Markov Chain Theory

Version of October 28, 2005

Contents

1 Applied Measure Theory **3**

2 Conditional Probability and Kernels **4**

3 General Markov Chains **6**

4 Kernel Operations **7**
4.1 Left Multiplication by a Measure 7
4.2 Right Multiplication by a Function 8
4.3 Multiplication of Kernels 8
4.4 The Identity Kernel 9

5 Special Types of Markov Chains **9**
5.1 Stationary Markov Chains and Invariant Measures 9
5.2 Reversible Markov Chains 10

6 Unnormalized Densities and Measures **11**

7 The Metropolis Update **12**
7.1 Algorithm 12
7.2 Proof 13
7.3 Examples 15
7.4 Turning an Update into a Markov Chain 15
7.5 Choosing the Proposal Distribution 16
7.6 History 17
7.7 Variable-at-a-Time Metropolis 19

8 Combining Updates I **20**
8.1 Composition 20
8.2 Composition versus Reversibility 21
8.3 State Independent Mixing 22
8.4 Subsampling a Markov Chain 23

8.4.1 Fixed Interval 23
8.4.2 Random Interval 24

9 The Metropolis-Hastings Update 25
9.1 Algorithm 25
9.2 Langevin Diffusion 25

10 The Gibbs Update 27
10.1 Algorithm 27
10.2 The Block Gibbs Update 28
10.3 The Generalized Gibbs Update 28
10.4 Proof 28
10.5 The Gibbs Sampler 29
10.6 Examples 29

11 The Swendsen-Wang Algorithm 30
11.1 The Ising Model 30
11.1.1 The Basic Model 30
11.1.2 Other Boundary Conditions 32
11.1.3 Phase Transitions 33
11.2 The Potts Model 34
11.3 Naive Metropolis and Gibbs 35
11.4 Swenden-Wang 36
11.5 Lessons Learned 40

12 Annealing and Tempering 41
12.1 Random Search Optimization 41
12.2 Simple Random Search Optimization 41
12.3 Adaptive Random Search Optimization 42
12.4 Simulated Annealing 42
12.5 Parallel Tempering 45
12.6 Serial Tempering 46

13 Monte Carlo Likelihood 46
13.1 Methods 46
13.2 Unknown Normalizing Constant Models 47
13.2.1 MCLA 50
13.2.2 MCNR 51
13.2.3 MCSA 54
13.2.4 MCMM 57

13.2.5 The Acid Test 58
13.3 Missing Data Models 58
13.3.1 MCLA 59
13.3.2 MCNR 62
13.3.3 MCSA 62
13.3.4 MCEM 62
13.4 Unknown Normalizing Constant Missing Data Models 64
13.4.1 MCSA 66
13.4.2 MCEM 66
13.5 Umbrella Sampling 66

1 Applied Measure Theory

If X is a random element of some probability space having measure P, then we write

$$E\{g(X)\} = \int g(x)P(dx) \tag{1}$$

whenever g is a real-valued function such that the expectation exists. From an applied point of view, we can regard (1) as a convenient shorthand. Whatever is meant by the left hand side is whatever is meant by the right hand side.

If X is a discrete random variable taking values in a set S with probability mass function f, then (1) means

$$E\{g(X)\} = \sum_{x \in S} g(x)f(x).$$

If X is a continuous random variable with probability density function f, then (1) means

$$E\{g(X)\} = \int_{-\infty}^{+\infty} g(x)f(x)\,dx.$$

If X is a continuous random vector taking values in $\mathbb{R}^3$ with probability density function f, then (1) means

$$E\{g(X)\} = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} g(x_1, x_2, x_3)f(x_1, x_2, x_3)\,dx_1\,dx_2\,dx_3.$$

If X is a random variable that is neither discrete nor continuous, for example, X is an exponential random variable with rate parameter λ right censored

at T (meaning we observe the exponential random variable or T, whichever is smaller), then (1) means

$$E\{g(X)\} = \int_0^T g(x)\lambda e^{-\lambda x}\,dx + g(T)e^{-\lambda T}.$$

So the integral in (1) doesn't necessarily mean integration in the sense of calculus. It may mean summation or a combination of integration and summation. And even if it does mean integration in the sense of calculus, it may mean a double, triple, or higher integral.

So measure-theoretic notation is valuable even in applied situations because it allows us to cover all the special cases with one notation. But isn't the left hand side of (1) and all those other equations good enough as a common notation? Not really, because it is too vague. The right hand side of (1) clearly indicates the measure P and clearly indicates that what expectation means depends (through P) on the probability model in question. The left hand side doesn't. This clarity will become more apparent as we go along.

We shall not need the abstract measure-theoretic definition of measures like P. It will be enough to know that when outside an integral, a measure is a set function, a map from subsets A of the state space to probabilities $P(A)$. It is a deep theorem that such functions determine abstract integrals like the right hand side of (1). But to apply measure theory, one only needs to know that *probability is a special case of expectation*

$$P(A) = E\{I_A(X)\} = \int_A P(dx), \tag{2}$$

where I_A denotes the *indicator function* of the set A,

$$I_A(x) = \begin{cases} 1, & x \in A \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

2 Conditional Probability and Kernels

In master's level probability theory much is made of the operation of finding conditional probabilities from joint probabilities. So much is made of it that students in such courses can be forgiven their impression that that is what conditional probabilities really are: thingummies derived from joint probabilities by an operation they have learned. This notion is too limited for general probability theory and too limited for the study of Markov chains.

The general view is actually much simpler. A conditional probability is just a variable probability that depends on some other variable. Often this other variable is written "behind the bar" so we write $P(A|y)$ meaning that the measure $P(\,\cdot\,|y)$ is a (possibly) different measure for each different y. If Q is another measure on a state space where y takes values then the "product" of P and Q is the "joint measure" and we can calculate expectations.

$$E\{g(X,Y)\} = \iint g(x,y)P(dx|y)Q(dy). \tag{4}$$

As in the case of only one abstract (measure-theoretic) integral, this iterated integral may mean summation, integration or a combination of the two. Each of the iterated integrals on the right hand side of (4) (with respect to x and with respect to y) is an expectation just like any of the special cases in the preceding section. After we have done the integral with respect to x (which is such an operation, ordinary integration, summation, or a combination of the two, as the case may be), we are left with another such integral with respect to y.

Markov chain theory is somewhat eccentric in its notation. We don't use the "bar" notation for conditional probability, insisting on rewriting (4) as

$$E\{g(X,Y)\} = \iint g(x,y)P(y,dx)Q(dy), \tag{5}$$

the only difference between the two being that one has $P(dx|y)$ where the other has $P(y,dx)$ and this is a mere notational quibble. Nevertheless the difference is important. If you can't recognize that (5) means the same thing as (4), then you can't understand the Markov chain literature.

Probabilities are nonnegative and integrate (abstractly, meaning ordinary integration, summation, or a combination of the two) to one so we have

$$Q(B) \ge 0, \qquad \text{for all } B$$

and

$$\int Q(dy) = 1$$

and also

$$P(y,A) \ge 0, \qquad \text{for all } y \text{ and all } A$$

and

$$\int P(y,dx) = 1, \qquad \text{for all } y.$$

But for complete generality, we want to drop this restriction. A *signed measure* is an arbitrary real-valued set function. It can be thought of as a linear combination of probability measures

$$\mu(A) = \alpha_1 P_1(A) - \alpha_2 P_2(A)$$

where P_1 and P_2 are probability measures and α_1 and α_2 are positive real constants (it is another deep theorem that all signed measures can be written this way). A *kernel* K is an arbitrary function[1] of two variables a point x and a set A. An arbitrary kernel K is *Markov* if all of the measures are probability measures

$$K(x, A) \ge 0, \qquad \text{for all } x \text{ and all } A \tag{6a}$$

$$\int K(x, dy) = 1, \qquad \text{for all } x. \tag{6b}$$

And K is *sub-Markov* if (6a) holds and (6b) with $=$ replaced by $\le$ holds.

3 General Markov Chains

A *Markov chain* is a sequence X_1, X_2, ... of random elements of some space, called the *state space* of the Markov chain, that has the *Markov property*

the future is independent of the past given the present,

which in notation is that

$$E\{g(X_{n+1}, X_{n+2}, \ldots) | X_1, \ldots, X_n\} = E\{g(X_{n+1}, X_{n+2}, \ldots) | X_n\}$$

for any function g (of an infinite number of random elements (the entire future after time n) for which the expectation exists. Using the Markov property and the iterated expectation theorem we have

$$\begin{aligned}
&E\{g(X_{n+1}, X_{n+2}, \ldots) | X_1, \ldots, X_n\} \\
&\qquad = E\bigl\{E[g(X_{n+1}, X_{n+2}, \ldots) | X_1, \ldots, X_{n+1}] \bigm| X_1, \ldots, X_n\bigr\} \\
&\qquad\qquad = E\bigl\{E[g(X_{n+1}, X_{n+2}, \ldots) | X_{n+1}] \bigm| X_1, \ldots, X_n\bigr\} \\
&\qquad\qquad\qquad = E\{h(X_{n+1}) | X_1, \ldots, X_n\} \\
&\qquad\qquad\qquad\qquad = E\{h(X_{n+1}) | X_n\}
\end{aligned}$$

[1]Not completely arbitrary, the measure-theoretic definition is that $K(x, \cdot)$ is a signed measure for each fixed x and $K(\cdot, A)$ is a measurable function for each fixed A.

where we have defined

$$h(X_{n+1}) = E[g(X_{n+1}, X_{n+2}, \ldots)|X_{n+1}].$$

Thus we see that the "one-step transition probabilities" $E\{h(X_{n+1})|X_n\}$ for arbitrary functions h and integers n are sufficient to determine everything about the Markov chain.

Let Q be the marginal distribution of X_1 (the *initial distribution* of the Markov chain, and let P_n be the Markov kernel that gives the distribution of X_n given X_{n-1} for $n = 2, 3, \ldots$. Then we can calculate the so-called "finite-dimensional distributions" of the Markov chain as

$$\begin{aligned} &E\{g(X_1, \ldots, X_n)\} \\ &\qquad = \iint \cdots \int Q(dx_1) P_2(x_1, dx_2) \cdots P_n(x_{n-1}, dx_n) g(x_1, \ldots, x_n) \end{aligned}$$

(and it is another deep theorem of measure theory that the finite-dimensional distributions determine a unique infinite-dimensional distribution for the whole sequence).

The P_n are called the transition probability kernels, and the Markov chain is said to have *stationary transition probabilities* if P_n does not depend on n. In MCMC we are only interested in Markov chains with stationary transition probabilities, and in a bit we shall adopt the convention that "Markov chain" implies "stationary transition probabilities" unless otherwise explicitly stated. But we continue a bit with general Markov chains.

4 Kernel Operations

4.1 Left Multiplication by a Measure

If μ is a general signed measure and K a general kernel, we define the notation $\mu K = \nu$ to mean

$$\nu(A) = \int \mu(dx) K(x, A), \qquad \text{for all } A.$$

When we specialize this definition to probability measures and Markov kernels we get the following interpretation. If μ is the marginal distribution of X_n and P is the kernel that gives the conditional distribution of X_{n+1} given X_n, then $\nu = \mu P$ is the marginal distribution of X_{n+1}. This is essentially what (5) says when we replace X and Y in (5) with X_{n+1} and X_n. Thus

left multiplication of the transition probability kernel by the marginal distribution at time n shifts forward in time to the marginal distribution at time $n + 1$.

4.2 Right Multiplication by a Function

If g is a general function on the state space and K a general kernel, we define the notation $h = Kg$ to mean

$$h(x) = \int K(x, dy)g(y), \qquad \text{for all } x.$$

When we specialize this definition to Markov kernels kernels we get the following interpretation. If P is the kernel that gives the conditional distribution of X_{n+1} given X_n, then $h = Pg$ is the conditional expectation of $g(X_{n+1})$ given X_n

$$h(X_n) = E\{g(X_{n+1})|X_n\}.$$

Thus

right multiplication of the transition probability kernel by a function gives the lag one conditional expectation of that function.

4.3 Multiplication of Kernels

If K_1 and K_2 are general kernels, we define the notation $K_3 = K_1K_2$ to mean

$$K_3(x, A) = \int K_1(x, dy)K_2(y, A), \qquad \text{for all } x \text{ and all } A.$$

If P_2, P_3, ... are the transition probability kernels of a general Markov chain, then P_nP_{n+1} is the conditional distribution of X_{n+1} given X_{n-1}. Why do we write $K_3 = K_1K_2$ rather than in the other order $K_3 = K_2K_1$? So it works with our other two notations! If μ is the marginal distribution of X_{n-1}, then μP_nP_{n+1} is the marginal distribution of X_{n+1}, and multiplication is associative

$$(\mu P_n)P_{n+1} = \mu(P_nP_{n+1}).$$

Similarly, if g is a general function on the state space, then $P_nP_{n+1}g$ is the conditional expectation of $g(X_{n+1})$ given X_{n-1}, and multiplication is associative

$$(P_nP_{n+1})g = P_n(P_{n+1}g).$$

Because of the associativity of multiplication (all three multiplication operations), we can omit the parentheses. If P is the transition probability kernel of a Markov chain with stationary transition probabilities, then P^n, meaning $PP \cdots P$ with n factors, is the conditional distribution of X_{n+1} given X_1. If μ is the initial measure of the Markov chain, then μP^n is the marginal distribution of X_{n+1}.

4.4 The Identity Kernel

The *identity kernel* on the state space is defined by

$$I(x, A) = \begin{cases} 1, & x \in A \\ 0, & \text{otherwise} \end{cases}$$

Comparing with (3) we recognize the right hand side: $I(x, A)$ means the same thing as $I_A(x)$. Of course, I and I_A do not mean the same thing (one is a kernel, the other a function). The function I_A can be written $I(\,\cdot\,, A)$.

It is easily checked that I is the identity element for kernel multiplication: $KI = IK = K$ for all kernels K. To evaluate KI we need to understand that $I(x, \,\cdot\,)$ is the probability measure concentrated at the point x, so for any function g

$$\int I(x, dy) g(y) = g(x).$$

So I also behaves as the identity in $Ig = g$ and $\mu I = \mu$.

Not only is I as important to "kernel theory" as zero is important to arithmetic (and for the same reasons), but I is also the transition probability kernel of a Markov chain, a rather useless Markov chain that goes nowhere and does nothing, satisfying $X_1 = X_2 = \cdots$ with probability one (all the variables are exact copies of the first). This fact will mostly be important as a source of simple counterexamples.

5 Special Types of Markov Chains

5.1 Stationary Markov Chains and Invariant Measures

If P is a Markov kernel, then π is an *invariant measure* for P if $\pi P = \pi$ (also called *stationary distribution* and *equilibrium distribution*).

We now impose the convention that "Markov chain" implies "with stationary transition probabilities" unless otherwise explicitly noted. Then if π is the initial distribution and P the transition probability kernel of a

Markov chain and $\pi P = \pi$, then we also have $\pi P^n = \pi$ for all n, so all of the X_n have the same marginal distribution (thus justifying the name "stationary"). Moreover, it is easily checked that the joint distribution of the k-tuple $(X_{n+1}, \ldots, X_{n+k})$ does not depend on k, which makes the Markov chain (by definition) a *strictly stationary stochastic process*.

Invariant measures need not exist, for example, consider the Markov chain on the integers that always moves one step to the right ($x_{n+1} = x_n + 1$ with probability one). If an invariant measure exists, it need not be unique, for example, consider the Markov chain with identity transition kernel ($\mu I = \mu$ for all measures μ).

When we do MCMC we always assume (prove, hopefully) that an invariant measure exists, is unique, is the one we want, and the Markov chain converges rapidly to equilibrium.

5.2 Reversible Markov Chains

If P is a Markov kernel, then P is *reversible* with respect to a measure π if

$$\iint \pi(dx) P(x, dy) g(x, y) = \iint \pi(dx) P(x, dy) g(y, x) \tag{7}$$

whenever g is such that the integrals exist (g is bounded, for example).

Plugging $g(x, y) = I_A(y)$ into (7) we get

$$\int \pi(dx) P(x, A) = \int_A \int \pi(dx) P(x, dy) = \int_A \pi(dx) = \pi(A),$$

which is $\pi P = \pi$. Hence P reversible with respect to π implies π is invariant for P.

The reason the notion is called "reversible" (sometimes "time reversible") is that when π is used as the initial distribution, so the chain is stationary, (7) has the interpretation that the joint distribution of the pair (X_n, X_{n+1}) is the same as the joint distribution of the pair (X_{n+1}, X_n) with the order reversed. From this one easily shows that the k-tuple $(X_{n+1}, \ldots, X_{n+k})$ has the same joint distribution as the k-tuple $(X_{n+k}, \ldots, X_{n+1})$ with the order reversed.

Thus we say the *stationary* chain with reversible kernel P looks the same (in distribution) running forward or backward. Note well that a *non-stationary* chain with kernel P will *not* look the same running forward or backward.

The main use of reversibility in MCMC is constructing kernels that have a specified invariant distribution. Given π, find a P such that P is reversible

w. r. t. (with respect to) π. It turns out that this is a much easier problem than: given π find a P that *preserves* π (meaning π is invariant for P). The reason is that $\pi P = \pi$ is a difficult integral equation to solve for P given π, whereas (7) although even more difficult if considered as an integral equation to solve, is quite trivial when considered as a symmetry condition to check (swapping x and y in the argument of g doesn't change anything). The reversibility condition is sometimes written

$$\pi(dx)P(x, dy) = \pi(dy)P(y, dx)$$

meaning that if we hit both sides with $g(x, y)$ and integrate, we get the same thing, regardless of what function g we use (so long as the expectations exist).

6 Unnormalized Densities and Measures

Let π be a probability measure and let λ be a measure that dominates π, meaning π has a density f with respect to λ

$$\pi(A) = \int_A f(x)\lambda(dx).$$

In most cases λ is Lebesgue measure, so $\lambda(dx)$ is just plain dx and f is the ordinary probability density function (but x is usually a vector although the notation doesn't explicitly indicate this and the integral is a multiple integral, because MCMC is never needed for one-dimensional problems). In most of the remaining cases λ is counting measure on a finite set S, so the integral becomes an ordinary sum

$$\pi(A) = \sum_{x \in A} f(x)$$

and f is the ordinary probability mass function.

A useful notion in MCMC is the *unnormalized density* which means a function $h = cf$, where $0 < c < \infty$. Of course, h determines f because f integrates to one, so h integrates to c

$$c = \int h(x)\lambda(dx)$$

and $f = h/c$.

The reason for introducing this notion is that it turns out that most methods of specifying MCMC algorithms need only specify unnormalized

densities and this turns out to be very convenient (in Bayesian inference, for example, the unnormalized posterior is likelihood times prior, which requires no computation to specify).

Similarly, we say that $\eta = c\pi$ is an unnormalized probability measure (which means no more than a positive, finite measure). The unnormalized density h is the density w. r. t. λ of the unnormalized measure η. We can say we are specifying either h or η.

Note that π is invariant for P if and only if η is invariant for P. Similarly, P is reversible w. r. t. π if and only if it is reversible w. r. t. η.

7 The Metropolis Update

7.1 Algorithm

Given an unnormalized density h with respect to λ, the Metropolis update makes a random change to the state that preserves the distribution having this unnormalized density. Thus, if iterated, it produces a Markov chain with h as the unnormalized density of the equilibrium distribution.

Let q be any function on the product of the state space with itself such that

- $q(x, y) = q(y, x)$ for all x and y,
- $q(x, \cdot\,)$ is a probability density w. r. t. λ for all x, and
- it is possible to simulate random realizations from $q(x, \cdot\,)$ for all x.

The *Metropolis update* of the state moves from a state x to the state x^* according to the following procedure

- [The Proposal] Simulate y from $q(x, \cdot\,)$.
- [The Odds Ratio] Calculate

$$r = \frac{h(y)}{h(x)} \tag{8}$$

- [Metropolis Rejection] With probability $\min(r, 1)$ set $x^* = y$, otherwise set $y = x$.

In the last step we say we "accept the proposal" when we set $x^* = y$ and otherwise we say we "reject the proposal."

The update is undefined when $h(x) = 0$, but $h(y) = 0$ is allowed. Such a proposal gives $r = 0$ so we "accept" the proposal with probability zero. If the update is used as the transition probability mechanism of a Markov chain, then so long as $h(x_1) > 0$ each iterate will be well-defined and satisfy $h(x_n) > 0$ as well.

Computer code for the update looks something like this, supposing `rq(x)` simulates a random variate having density $q(x, \cdot)$ and `runif` simulates a uniform $(0, 1)$ random variate

```
y = rq(x);
r = h(y) / h(x);
if (runif() < r)
    x = y;
```

Or one can save a call to `runif` sometimes with

```
y = rq(x);
r = h(y) / h(x);
if (r < 1 && runif() < r)
    x = y;
```

(the value of `x` at the end is what is denoted x^* in the mathematical description).

Note well that when the "Metropolis rejection" step "rejects" the state does not change (we have $x^* = x$). So a Markov chain that is produced by iterating a Metropolis update over and over, has many steps when the state does not change. This is part of preserving π. Any attempt to avoid this "rejection" only ruins the algorithm. The state not changing in "rejection" steps is not a bug, it's a feature. It's what makes the algorithm get the correct stationary distribution with only trivial calculations.

7.2 Proof

We claim this update defines a kernel P that is reversible w. r. t. η. First we define the kernel. Let

$$a(x, y) = \min\left(\frac{h(y)}{h(x)}, 1\right)$$

be the probability that the "Metropolis rejection" step is "accepted" (executes the assignment $x = y$). Then

$$P(x, A) = \int_A q(x, y)a(x, y)\lambda(dy) + I(x, A)\left(1 - \int q(x, y)a(x, y)\lambda(dy)\right).$$

This is a little complicated, but sensible if we take it one bit at a time. We can move from x to A by proposing $y \in A$ and accepting (that's the first term) or by having $x \in A$ originally (that's what the $I(x, A)$ is in there for) and proposing some y that is rejected (that's the term in the big parens).

Now

$$\iint \eta(dx) P(x, dy) g(x, y) = \iint h(x) q(x, y) a(x, y) g(x, y) \lambda(dx) \lambda(dy) + \int h(x) g(x, x) \left(1 - \int q(x, y) a(x, y) \lambda(dy) \right) \lambda(dx)$$

and the second term on the right hand side (the second line of the display) is obviously unchanged if the two arguments of g are swapped (since they are both x). Thus we only need to show that the first term is unchanged if we replace $g(x, y)$ by $g(y, x)$. So

$$\begin{aligned} &\iint h(x) q(x, y) a(x, y) g(x, y) \lambda(dx) \lambda(dy) \\ &\qquad = \iint h(y) q(y, x) a(y, x) g(y, x) \lambda(dy) \lambda(dx) \\ &\qquad\qquad = \iint h(y) q(x, y) a(y, x) g(y, x) \lambda(dy) \lambda(dx) \end{aligned}$$

the first inequality being interchange of dummy variables and the other being the symmetry requirement $q(x, y) = q(y, x)$. Thus in order to finish the proof it is enough to show that

$$h(x) a(x, y) = h(y) a(y, x), \qquad \text{for all } x \text{ and all } y. \tag{9}$$

To prove that, assume without loss of generality that $h(x) \geq h(y)$. Then

$$a(x, y) = \frac{h(y)}{h(x)} \leq 1$$

and

$$a(y, x) = 1$$

so

$$h(x) a(x, y) = h(x) \cdot \frac{h(y)}{h(x)} = h(y)$$

and

$$h(y) a(y, x) = h(y)$$

so (9) does indeed hold, and the Metropolis update does indeed preserve η because its kernel is reversible w. r. t. η.

7.3 Examples

The standard example is to make $q(x, \cdot\,)$ be a nondegenerate multivariate normal distribution centered at x. Then

$$q(x, y) = \phi(x - y)$$

where ϕ is multivariate normal centered at zero. Hence q does indeed have the required symmetry property. This is the update used by the `metrop` function in the `mcmc` contributed package for R. The variance-covariance matrix of the normal proposal can be anything so long as it is nonsingular. In order that the Markov chain have stationary transition probabilities, the proposal distribution must not change. We must use the same variance-covariance matrix for all proposals.

Clearly, the particular form of the normal distribution plays no role in the preceding example. We could replace the normal ϕ above by any density having point symmetry about zero

$$\phi(x) = \phi(-x), \text{for all } x.$$

There are not many such symmetric multivariate distributions that can be simulated so as to make suitable proposals. The only other obviously correct possibility is uniform on any region having a center of symmetry with x being the center. The regions could be balls, ellipsoids, or boxes. It is not clear that any of these are better than normal proposals.

7.4 Turning an Update into a Markov Chain

One makes a Markov chain (with stationary transition probabilities, our default assumption) by executing the same random mechanism over and over and letting X_n be the state after the n-th execution. This random mechanism is associated with a transition probability kernel P. The chain starts at X_0 which can have any distribution (the "initial distribution" μ).

> MCMC involves running a Markov chain with transition probability kernel P and invariant distribution π.

That's all there is to it. There's a Markov chain. You run it. And you use averages over the run to estimate (approximate, calculate, whatever) properties of π.

Right now, we only know one way to make a kernel P that preserves a specified stationary distribution π, the Metropolis update. So right now our

recipe for doing MCMC is to iterate the same Metropolis update over and over.

But we shall soon (Sections 7.7 and 10) meet two other methods of making what we call *elementary updates*, the "indivisible atoms" of MCMC mechanisms, and we shall also soon (Section 8) meet methods of combining elementary updates to make a "combined kernel" P that preserves a specified π. Then everything we said here will apply to these more general update mechanisms. Given π you construct a random mechanism described by a kernel P (elementary or combined) that preserves π. Then one makes a Markov chain (with stationary distribution π) by executing this random mechanism over and over.

7.5 Choosing the Proposal Distribution

The wonderful feature of the Metropolis algorithm, that *any* proposal works, leaves us with a difficult problem of too many choices. Some proposals will work better than others (will produce more accurate Monte Carlo approximation in the same amount of computer time). Which do we choose?

In general, there is very little one can say. The possible problems that MCMC can be used to solve include every possible probability problem, not to mention every possible integration problem (thus going far beyond probability and statistics). This class of problems is so general, that nothing can be said at this level of generality.

Most discussions in the literature focus on the *acceptance rate* in the Metropolis rejection step (the proportion of Metropolis proposals that are accepted) as a guide. Moreover, they focus on a particular class of proposals (for example, multivariate normal Metropolis proposals). With such simplification of the problem, our choices seem much simpler. How do we adjust the variance matrix of the (multivariate normal) proposal so as to get good performance, and how does the acceptance rate indicate good performance?

It is important to understand that higher acceptance rate is not necessarily good.

- If the unnormalized density h is continuous, then one can always make the acceptance rate as close to one as one pleases by making the proposal variance very very small so $h(y)$ and $h(x)$ are nearly equal and the odds ratio is nearly one.

  But such "baby steps" take a very long time to get anywhere.

- Conversely, consider "giant steps" with very large proposal variance.

In order for h to be integrable, it must go to zero at infinity, thus if the current position x is from the equilibrium distribution and y is very far from x we will generally have $h(y) \ll h(x)$ and the odds ratio is nearly zero.

The "giant steps are bad" part of the argument is not so clear as the "baby steps are bad" part, but it is clear that we do not want an acceptance rate so low that there are very few acceptances in the entire run of the Markov chain that we are willing to do. It is clear that we don't want an acceptance rate that is either zero or one, and intuitively obvious that we don't want an acceptance rate that is nearly zero or one either.

Thus it seems that we have something of a "Goldilocks problem" (we don't want the porridge too hot or two cold as in the children's story of Goldilocks and the three bears). We want an acceptance rate somewhere between zero and one. Surprisingly, it is possible to work out the theoretically optimal acceptance rate for some very simple problems. Gelman et al. (1996) considered the problem of sampling the multivariate normal distribution (which, of course, does not need MCMC but is simple enough to analyze theoretically) and showed that an acceptance rate of about 20% was right for normal proposal Metropolis (the optimal rate goes to 23.4% and the dimension of the state space goes to infinity).

In a quite different situation Geyer and Thompson (1995) came to a similar conclusion, that a 20% acceptance rate is about right, But they also warned that a 20% acceptance rate could be very wrong and produced an example where a 20% acceptance rate was impossible and attempting to reduce the acceptance rate below 70% would keep the sampler from ever visiting part of the state space.

The 20% magic number must be considered like other rules of thumb we teach in intro courses (like $n > 30$ means means normal approximation is valid). It is not at all clear that the focus on *acceptance rate* as the sole criterion of goodness of proposal makes any sense. Even if one decides to focus on acceptance rate, we have no theory that tells us what acceptance rate to use in general.

For more on acceptance rates, see the discussion at the end of Section 9.2.

7.6 History

The Metropolis update was invented by Metropolis, Rosenbluth, Rosenbluth, Teller, and Teller (1953) at the dawn of the computer age, when Los Alamos National Laboratory was one of the few places that had a computer.

Metropolis et al. (1953) did not use the Metropolis update described here but the variable-at-a-time updates described later. Metropolis et al. (1953) were not doing probability and statistics, but simulating the thermodynamic equilibrium distribution of a physical system (a liquid in equilibrium with its vapor phase). This is natural. Physical systems are continuous time Markov processes. To find out about their equilibrium distribution, let them come into equilibrium, and then run them for a long time and average. That's what thermodynamic equilibrium distribution means. The *tour de force* involved in the paper is the realization of Metropolis et al. (1953) that the physical system need not be simulated exactly, but that any continuous time *or discrete time* Markov process having the *same equilibrium distribution* could be used instead (because, after all, it will, by definition, give the same answer). The other contribution of Metropolis et al. (1953) was understanding the reversibility argument for finding a Markov chain with a specified equilibrium distribution.

Despite never having been forgotten in computational physics and chemistry, and despite being known in spatial statistics which used Markov random field models similar to those used in statistical physics (for example, Ising and Potts models, explained in Section 11.1 and 11.2 below), being explained in Ripley (1987), for example, the Metropolis algorithm was little used in statistics until after MCMC became popular in the form of the Gibbs sampler following the appearance of Gelfand and Smith (1990). It wasn't until several years later that Clifford (1993), discussing a Royal Statistical Society meeting in which three papers on MCMC were read, said

> Surely it has to be recognized that the Gibbs sampler attained prominence by accident. Currently, there are many statisticians trying to reverse out of this *cul-de-sac*. To use the Gibbs sampler, we have to be good at manipulating conditional distributions and ingenious in sampling, so this rather brings back the mystique of the statisticians. The same equilibrium can be attained by using a Markov chain in which sites are updated one at at time by the Metropolis method. This only involves calculating ratios of densities to accept or reject the sampled state at each site. It is very much easier to programme ....

I couldn't have said it better myself. For Americans, "reverse out of this *cul-de-sac*" means back out of this dead end street. The only thing to be added is that the simple Metropolis update (not the variable-at-a-time update Clifford specifically refers to) is also good. I can tell a story about an experienced MCMC user who in 2005, although well aware of the Metropolis

update, having used it in previous work, went through incredible contortions trying to do an approximate Gibbs update. When I pointed out after his talk (fortunately a "work in progress" talk) that Metropolis would be easy, he said "I'm embarrassed." A week later Metropolis was working well for him. I could tell a dozen other stories where experienced MCMC users asked me for help when the only real problem is that they were trying to use Gibbs instead of Metropolis. It may add to "the mystique of the statisticians" but it makes many simple problems impossibly hard.

7.7 Variable-at-a-Time Metropolis

As mentioned in the history, Metropolis et al. (1953) actually used a variant of the update described in Section 7.1 that most people nowadays think of as the plain Metropolis update. We call the variant described in this section "variable-at-a-time" Metropolis.

In this section subscripts denote components of the state vector *not* different variables of a Markov chain. Everything remains the same as described in Section 7.1 except that the proposal only changes one component of the state, which is a vector $x = (x_1, \ldots, x_d)$. The proposal is of the form

$$y = (y_1, \ldots, y_d) = (x_1, \ldots, x_{j-1}, y_j, x_{j+1}, \ldots, x_d),$$

that is, we have $y_i = x_i$ except for y_j, which is a new random variate. The proposal density $q_j(x, y_j)$ must thus be a density w. r. t. a measure λ_j which is on the space in which x_j and y_j take values, and the unnormalized density h of the desired stationary distribution must be a density w. r. t. the product measure $\lambda = \lambda_1 \times \cdots \times \lambda_d$. The symmetry requirement is now

- $q_j(x, y_j) = q_j(y, x_j)$ for all x and y.

Then everything goes through unchanged (the odds ratio and Metropolis rejection are the same). The proof in Section 7.2 can be altered for this variable-at-a-time update—the changes are mostly notational—but, since this update is a special case of the Metropolis-Hastings-Green update we shall omit the details.

A variable-at-a-time Metropolis update does not, by itself, make a good Markov chain. Since it only changes one variable, it can never sample the equilibrium distribution. There are many ways to combine different updates that preserve the same stationary distribution to make a "combined update" that also preserves that distribution and does make a good Markov chain. These ways deserve a section of their own, which follows.

8 Combining Updates I

8.1 Composition

It is a trivial observation that if each term of a product of kernels preserves a stationary distribution, then so does the product, in mathematical notation

$$\pi P_i = \pi, \qquad \text{for all } i$$

implies

$$\pi P_1 P_2 \ldots P_d = \pi,$$

the proof being the aforementioned *multiplication is associative* (applied to kernel operations).

This idea is most frequently called "fixed scan" in the literature, the image being one of making a "scan" over the possible choices in fixed order.

The reason we like the term *composition* is because that is what it is mathematically in several senses. One can think of kernels as linear operators on the space of signed measures (a kernel K mapping $\mu \mapsto \mu K$), and then this is composition of operators

$$P_1 P_2 \cdots P_d = P_d \circ P_{d-1} \circ \cdots \circ P_1$$

(note the reversal of order). One can also think of kernels as linear operators on the space of functions (a kernel K mapping $g \mapsto Kg$), and then this is composition of operators

$$P_1 P_2 \cdots P_d = P_1 \circ P_2 \circ \cdots \circ P_d$$

(note the non-reversal of order). One can also think of kernels as linear operators on the space of kernels, in which case there are two kinds $M \mapsto MK$ and $M \mapsto KM$, and the two kinds correspond to the two compositions above.

This recognition of composition makes the proof obvious (composition of operators is always associative, whether or not the operators have anything to do with Markov chains). Whichever one prefers, "composition" or "fixed scan," it is important to understand the method clearly and its trivial proof. Note that it is our measure-theoretic notation and our knowledge of kernel operations that make everything trivial. If we insisted on master's level theory with integrals and sums written our explicitly and lots of case splitting, this could be made arbitrarily messy and confusing.

8.2 Composition versus Reversibility

A Metropolis update (plain or variable-at-a-time) is reversible (more precisely, its kernel is reversible w. r. t. π). But a composition of reversible kernels is, in general, not reversible. The "reverse" of the composition $P_1 P_2 \cdots P_d$ is the composition $P_d P_{d-1} \cdots P_1$ and these kernels are reversible if and only if they are the same:

$$P_1 P_2 \cdots P_d = P_d P_{d-1} \cdots P_1.$$

One way to make them the same is to use a *palindromic* composition[2] that is obviously the same when reversed, such as $P_1 P_2 P_3 P_1 P_3 P_2 P_1$. Of course, a palindromic composition can also be called a *palindromic fixed scan*.

The connection between reversibility and composition is curious and of some theoretical importance. Theory for reversible Markov chains is much sharper and better understood than for non-reversible Markov chains. All known methods of MCMC that apply to complicated problems use the reversibility principle and have "elementary" updates (the "atoms" from which combined updates are formed) that are reversible.

The other methods of combining updates that we shall study (various forms of "mixing") *also* preserve reversibility. The only source of non-reversibility is non-palindromic composition. One can hold two attitudes toward this.

(i) Since palindromic composition is easy and has sharper theory, use it.

(ii) Since palindromic compositions are generally twice as long as need be, and since non-reversibility only makes the theory somewhat messier and less sharp, which may not be of any practical concern, why bother?

(iii) Moreover, since non-reversible MCMC, at least in theory, can be more efficient than reversible MCMC, non-palindromic composition may do some good.

Although there is some literature on non-reversible MCMC (see Mira and Geyer, 2000, and other works cited therein), little progress has been made on the only kind of reversibility that arises in practice, which is from non-palindromic composition. In particular, it is not known whether point (iii) above, is of any practical importance.

[2] A *palindrome* is a phrase that reads the same forwards and backwards, such as "Able was I ere I saw Elba."

Your humble author holds the view (i) above, but admits the case is not overwhelming.

8.3 State Independent Mixing

If multiplication of kernels provides one method of combining, perhaps addition provides another? Addition of kernels is well defined and obvious, but does not correspond to any probabilistic operation. The sum of Markov kernels is not Markov (the sum integrates to two, not one).

However, a *convex combination* of kernels

$$q_1 P_1 + q_2 P_2 + \cdots + q_d P_d$$

where the P_i are Markov kernels and the q_i are nonnegative real numbers that sum to one (and do not depend on x or A in the kernels), does correspond to a probabilistic operation. This combined update proceeds as follows.

- Choose an index j at random, choosing j with probability q_j.
- Update the state using the mechanism with kernel P_j.

We call this state independent mixing to contrast with state dependent mixing, which will be covered later. *State independent* means the q_j do not depend on the state x. That

$$\pi(q_1 P_1 + q_2 P_2 + \cdots + q_d P_d) = \pi$$

is again a trivial matter of algebra (and the fact that the q_i sum to one), and again it is our use of measure-theoretic notation that makes it trivial. It is crucial that the q_i do not depend on x or A. As we shall see, state dependent mixing is rather less trivial.

What we are calling mixing here is more often called "random scan" in the literature, the image being one of making a "scan" over the possible choices in random order. But from our point of view (the "update" point of view), this term is highly misleading. A state independent mixing update does not do a "scan." Rather it executes the mechanism associated with exactly one P_j (chosen randomly).

It is only slightly less trivial that the mixing can be over an infinite set of choices. Say P_z is for each z a Markov kernel that preserves π and Q is an arbitrary measure on the space where z lives. The mixture kernel

associated with the mechanism that chooses z at random according to Q and then executes the mechanism having kernel P_z is

$$P_{\text{mix}}(x, A) = \int Q(dz) P_z(x, A)$$

and $\pi P_{\text{mix}} = \pi$ written out in full is

$$\pi(A) = \int \pi(dx) \int Q(dz) P_z(x, A)$$

The Fubini theorem (change of order of doing double integrals) says

$$\begin{aligned} \int \pi(dx) \int Q(dz) P_z(x, A) &= \int Q(dz) \int \pi(dx) P_z(x, A) \\ &= \int Q(dz) \pi(A) \\ &= \pi(A) \end{aligned}$$

The main use of this mixing over an infinite choice of possibilities is the so-called *hit-and-run* sampler (what this has to do with leaving the scene of an auto accident has always been a mystery to me, perhaps any catchy short name is good, even if it makes no sense) proposed by Bélisle et al. (1993) and further studied by Chen and Schmeiser (1993).

In this sampler, the state space is Euclidean. The random choice is a direction. Then one (conceptually) does a change-of-variable so that only one coordinate changes along that direction and updates that coordinate variable using a variable-at-a-time Metropolis (or Metropolis-Hastings or Gibbs) update.

This algorithm has no particular virtues, but it is good to know that it fits into our general notions about combining updates, which shows these notions are general enough to encompass a wide variety of schemes.

8.4 Subsampling a Markov Chain

Although rarely thought of as belonging in this section (combining updates), a subsampled Markov chain is just a special case of composition.

8.4.1 Fixed Interval

If P is a Markov kernel, then P^k is also a Markov kernel, and P^k is clearly the k-fold composition of P with itself. If X_1, X_2, ... is a Markov chain with kernel P, then X_k, X_{2k}, ... is a Markov chain with kernel P^k.

This operation is commonly called "subsampling" the Markov chain. The original justification was that the subsampling lessened the autocorrelation in the chain, but this is wrong headed. As Elizabeth Thompson once eloquently summed it up,

> you don't get a better answer by throwing away data.

The first formal proof of this (for reversible chains only) was given by Geyer (1992). Then MacEachern and Berliner (1994) gave a much simpler proof that also applied to non-reversible chains.

Geyer (1992) also gave a finer analysis. If one considers not only the cost (computing time, memory usage, whatever) of generating the Markov chain, but also the cost of using the samples generated and notices that a subsampled chain will have cost of generation proportional to kn where k is the spacing and n the number of iterates in the subsampled chain, but the cost of using samples will be proportional to n. So subsampling may be effective, but only if the cost of "using" is large compared to the cost of "sampling." The theorems that forbid subsampling mentioned above assume zero cost of "using."

The point is not that one should do such an analysis (the analysis would generally take longer than just going ahead and doing something suboptimal). The point is to be aware of the issue. Before these theorems, subsampling sounded plausible to many MCMC experts. Even if one considers costs, most users' intuitions about subsampling are far too favorable to it. Most users guess that intervals like 100 or 1000 are a good idea when a formal analysis would say something more like 3 or 5 is optimal.

Moreover, subsampling also competes with the method of batch means. If the only point of subsampling is to avoid excessive memory use, then one can batch instead of subsample and get the full accuracy possible without using more memory.

8.4.2 Random Interval

Fixed interval subsampling can be "part of the problem not part of the solution" because it can convert an effective sampler into a useless one. This most often happens when the original sampler is periodic, but can also happen when the original sampler is only "nearly" periodic. Subsampling using an interval that is a multiple of the period (or "near" period) can destroy all of the good properties of the sampler (even mere irreducibility).

But subsampling at a *random* interval has no such drawbacks. Surprisingly, this idea seems unknown to MCMC practitioners, although it is a

major theoretical tool in the best Markov chain theory book (Meyn and Tweedie, 1993).

Since it is hardly used, we need say no more about it other than to mention that it also fits into our general notions about combining updates. It is a mixture of updates, which are themselves compositions. The random choice selects a random non-negative integer j, and the update executes the mechanism associated with P^j, where P^0 is another notation for the identity kernel I. So this is state independent mixing from a possibly infinite mixture. This again shows our notions about combining are general enough to encompass a wide variety of schemes.

9 The Metropolis-Hastings Update

9.1 Algorithm

Hastings (1970) proposed a variant of the Metropolis which makes the symmetry requirement unnecessary. Everything is the same as described in Section 7.1 except that the requirement

- $q(x, y) = q(y, x)$ for all x and y

is *dropped* and replaced by the much weaker

- $q(x, y)$ can be evaluated for all x and y.

Of course, without the symmetry requirement, the algorithm is no longer correct, but Hastings found that the simple change of replacing the Metropolis definition of r in (8) by

$$r = \frac{h(y)q(y, x)}{h(x)q(x, y)} \tag{10}$$

restores correctness.

Then everything goes through unchanged (use this r in the Metropolis rejection and everything else works the same). The proof in Section 7.2 can be altered for this Metropolis-Hastings update, but, since this update is a special case of the Metropolis-Hastings-Green update we shall omit the details.

9.2 Langevin Diffusion

Grenander and Miller (1994) proposed using a continuous time rather than a discrete time Markov process for simulation. They were not the first

to do this, as we shall later see in the context of spatial point processes. The problem with this is that a computer can't do continuous time. One must use a discrete-time approximation. But then one is not actually doing the process one is theorizing about. It turns out that discretizing a continuous time process like this is highly problematic (Roberts and Tweedie, 1996); the convergence properties of the continuous time process need not correspond to those of the discrete time approximation.

But there is a much simpler way to see this is problematic. Let us introduce the term *exact* MCMC for running a Markov chain that has *exactly* the claimed stationary distribution. Of course, the "exact" here ignores both rounding errors in computer arithmetic (which is a different kind of discrete approximation) and any defects of the random number generators that are raw materials for our MCMC algorithms. Generally, statisticians ignore rounding errors because theory about that lies in a different discipline: numerical analysis. They also, for the most part ignore defects in random number generators, because currently available random number generators are so good that whatever defects there may be are negligible compared to MCMC convergence issues. Thus "exact" MCMC is the most one tries to achieve. We claim it should also be the least. Since "exact" MCMC is so easy to achieve (just correctly implement Metropolis, Metropolis-Hastings, Gibbs, Metropolis-Hastings-Green, with correct combining of updates), there is no excuse for not being "exact."

Grenander and Miller (1994) were not "exact" and in hindsight, since the Roberts and Tweedie (1996) paper, this is a mistake. They were not the first to make this mistake. Many sorta-kinda-but-not-exactly MCMC algorithms have been devised and published, but now we know that there is no sorta-kinda-but-not-exactly Markov chain convergence theory that justifies them. Fortunately, Besag (1994) in his discussion of Grenander and Miller (1994) pointed out how to fix their algorithm. Simply consider each of their iterates as a mere *proposal* in a Metropolis-Hastings update which must be followed by a Metropolis rejection step. Now we are doing "exact" MCMC and there are no problems.

The actual Langevin diffusion proposal is multivariate normal but does not have the symmetry property of a Metropolis proposal (which requires the mean be the current state x). It proposes y to be multivariate normal with mean

$$x + \frac{\epsilon}{2} \nabla h(x)$$

and variance-covariance matrix ϵ times the identity. Here ϵ is some "small" number that is the discrete time step length (as $\epsilon \to 0$ we get closer and closer

to continuous time) and $\nabla h(x)$ is the gradient (vector of partial derivatives) of h evaluated at the point x.

When we consider this as a Metropolis-Hastings update there is no reason for ϵ to be small; the update is valid for all positive ϵ. As in all Metropolis-Hastings we adjust the "tuning parameter" (here ϵ) so that we get an acceptance rate that is not too large and not too small. There is no reason to make ϵ as small as possible. In fact, this is the worst thing you can do. Making ϵ very small guarantees the algorithm will make only very small steps and have very slow convergence.

It is interesting that a true continuous time Langevin diffusion (if it could actually be done) would be non-reversible. It drifts in the direction $\nabla h(x)$. The time-reversed process has the opposite drift. But the Metropolis-Hastings corrected discrete time approximation is reversible (like any Metropolis-Hastings update). So another lesson in this story is that the only way we know how to get "exact" MCMC is using reversibility.

Roberts and Rosenthal (1998) show that an acceptance rate of about 50% is optimal for the Langevin diffusion approximation Metropolis-Hastings algorithm, more precisely, they show that for problems in which the equilibrium distribution has IID components (and hence for which MCMC is unnecessary) that the optimal acceptance rate goes to 57.4% as the dimension of the state space goes to infinity. They also discuss some extensions of their theory to slightly more complicated problems than IID ones, but do not have an extension to completely general equilibrium distributions. Thus, as we saw for simple Metropolis in Section 7.5, there is no general theory for setting acceptance rates. Nor is there any general theory that says that *acceptance rates* are the right quantity to look at to adjust the proposal of a Metropolis-Hastings update.

There is no theory at all about adjusting proposals in a combined update (variable-at-a-time Metropolis, and the like).

10 The Gibbs Update

In this section subscripts denote components of the state vector *not* different variables of a Markov chain.

10.1 Algorithm

Given a desired stationary distribution π whose state is a vector $x = (x_1, \ldots, x_d)$, update one variable, say x_j, by giving it a random realization

from its conditional distribution given "the rest" $(x_1, \ldots, x_{j-1}, x_{j+1}, \ldots, x_d)$, this conditional distribution being derived from the joint distribution π.

10.2 The Block Gibbs Update

For any subset J of $D = \{1, \ldots, d\}$, let x_J denote the tuple formed from the variables x_j, $j \in J$. A *block Gibbs* update gives x_J a random realization from its conditional distribution given "the rest" $x_{D \setminus J}$, this conditional distribution being derived from the joint distribution π.

10.3 The Generalized Gibbs Update

Given any function of the state $g(x)$, a *generalized Gibbs* update gives x a random realization from its conditional distribution given $g(x)$, this conditional distribution being derived from the joint distribution π.

Clearly, a block Gibbs update is the special case obtained when $g(x) = x_J$, and an ordinary Gibbs update is the special case of block Gibbs obtained when $J = \{j\}$. Conversely, generalized Gibbs is the special case of ordinary Gibbs obtained when one does a change of variable so that one of the variables is $g(x)$.

10.4 Proof

The proof is trivial: marginal times conditional equals joint. Additional mathematical detail can hardly make the argument any more convincing, but for those who want to see it, let P be the conditional distribution of X given $g(X)$ for a generalized Gibbs update, and let Q be the marginal distribution of $g(X)$, both marginal and conditional being derived from π. If the current state X has distribution π, then $g(X)$ has distribution Q, and a generalized Gibbs update of the current state has distribution

$$\int Q(dy) P(y, A) = \pi(A)$$

because when we integrate out $y = g(x)$ we get the marginal of the other variable which is the joint distribution π (because the "other" variable is X).

This is the sort of argument that shows that "applied measure theory" is worth knowing. The joint distribution of X and $g(X)$ is degenerate, hence impossible to handle using the tools of master's level theory. Measure theory (rigorous or not) has no problem. The distributions exist. We have notation for them. The argument goes right through.

10.5 The Gibbs Sampler

The so-called *Gibbs sampler* is an MCMC algorithm using only Gibbs updates. A single Gibbs update does not, by itself, make a good Markov chain. Since (if ordinary) it only changes one variable, it can never sample the equilibrium distribution. One needs to combine the Gibbs updates using any of the combining methods discussed in Section 8.

10.6 Examples

Suppose we observe IID normal data assumed to be two-parameter normal and we want to do a Bayesian analysis. The log likelihood is

$$l(\mu, \lambda) = -\frac{\lambda}{2} \sum_{i=1}^{n} (x_i - \mu)^2 + \frac{n}{2} \log(\lambda)$$

where μ is the mean of the normal distribution and λ is the precision (inverse variance). We adopt independent priors on the parameters: Normal(m, l^{-1}) for μ and Gamma(a, b) for λ. Then a log unnormalized posterior is

$$l(\mu, \lambda) - \frac{l}{2}(\mu - m)^2 + (a - 1) \log \lambda - b\lambda$$

Since a conditional probability density is merely the joint density renormalized, when unnormalized densities are allowed

> there is no difference between (unnormalized) joint and conditional except that the joint is considered a function of all the variables and the conditional a function of some of them.

Looking at our unnormalized (joint) posterior, it is clear that

- considered as a function of μ only, it is quadratic, hence the conditional of μ given λ is normal, and
- considered as a function of λ only, it is of the form $c_1 \log \lambda - c_2 \lambda$, where c_1 and c_2 are not functions of λ, hence the conditional of λ given μ is gamma.

Since the remaining details (figuring out exactly which normal and which gamma) are an exercise in master's level theory that allows the reader to participate in what Clifford calls "the mystique of the statisticians" (in the quotation on p. 18), we don't wish to spoil the reader's pleasure and stop our analysis here.

We make only a few comments about the general nature of the Gibbs update.

(i) The priors used here are *not* the conjugate prior studied in master's level theory that gives a closed form solution for the posterior in which the joint prior and joint posterior are factored as marginal times conditional one way are gamma and normal and factored the other way are Student t and gamma.

(ii) Thus the class of "Gibbs friendly" priors is larger than the class of "pencil-and-paper friendly" priors usually called *conjugate*. The "Gibbs friendly" property is analogous to conjugacy, except one considers one variable at a time holding the others fixed.

(iii) But the class of "Gibbs friendly" priors is not much larger. The Metropolis update allows any prior at all. Gibbs doesn't. As I once noted in a slightly different context (Geyer, 1995)

> Computational convenience is a poor substitute for philosophy.

If you are a subjective Bayesian and a "Gibbs friendly" prior truly represents the relevant expert opinion, then use Gibbs (although you might have a difficult time convincing anyone the prior was not chosen for reasons of computational convenience). If not, then don't. And similarly with objective substituted for subjective and whatever criterion one thinks constitutes Bayesian objectivity (a Jeffreys prior, for example) substituted for expert opinion.

11 The Swendsen-Wang Algorithm

11.1 The Ising Model

11.1.1 The Basic Model

The Ising model is what is now called a *spatial lattice process* or a *graphical model* in statistical terminology. Originally, it was a model from physics, an attempt to model magnetism.

Although the Ising model can be defined on an infinite lattice $\mathbb{Z}^2 = \mathbb{Z} \times \mathbb{Z}$, where $\mathbb{Z}$ denotes the set of all integers and $\times$ indicates a Cartesian product, thus justifying calling it a stochastic process, the only models that can be simulated by MCMC or by any other simulation method must have a finite number of variables, hence a finite graph.

Thus we define the Ising model on a finite square lattice $\mathbb{D}^2$, where $\mathbb{D} = \{1, \ldots, d\}$. As in all graphical models we associate a random variable with each node of the graph. Let X_i denote the random variable at node i.

In order to have a graphical model, we must have *edges* in the graph. Let G denote the vertex set and E denote the edge set of the graph. Here $G = \mathbb{D}^2$ and E is a subset of $G^2 = G \times G$ that we have yet to define. We now do so. Writing $i = (i_1, i_2)$ and $j = (j_1, j_2)$ we set

$$E = \{\, (i, j) \in G^2 : |i_1 - j_1| + |i_2 + j_2| = 1 \,\}. \tag{11}$$

This is the so-called *nearest neighbor* graph. We have an edge if and only if either the first coordinates differ by one or the second coordinates differ by one, but not both. Each node in the interior of the graph has four neighbors (has four edges linking it to other nodes). Nodes on the sides of the graph have three neighbors. Nodes in the corners of the graph have two neighbors.

In set theoretic terminology, a subset E of $G \times G$ is a *relation* on G. A relation E is *symmetric* if $(i, j) \in E$ if and only if $(j, i) \in E$. In the terminology of graph theory, a graph is *undirected* if its edge set is a symmetric relation. All of the graphical models we consider in this section will be undirected.

In physics the X_i take values in the two-point set $\mathbb{S} = \{-1, 1\}$ and are called *spins*. Each X_i models one atom in a crystal lattice. Of course, real crystals are three-dimensional, but theory exists only for two-dimensional lattice processes. Moreover, statistical applications of Ising models, as prior distributions in Bayesian image reconstruction, are also two-dimensional. In a ferromagnetic material (like iron), spins tend to align together. In the Ising model, we model only the interaction of neighbors defined by the graph. The unnormalized density (with respect to counting measure on $\mathbb{S}^G$) of the thermodynamic equilibrium distribution of the vector of spins is

$$h(x) = \exp\left(\sum_{i \in G} \alpha_i x_i + \frac{1}{2} \sum_{(i,j) \in E} \beta_{ij} x_i x_j \right) \tag{12}$$

where the α_i and β_{ij} are parameters of the distribution. The 1/2 in (12) corrects for double counting; each edge occurs twice in E, once as (i, j) and once as (j, i). The term *Ising model* refers to the situation where we have symmetry: all of the α_i are equal and all of the β_{ij} are equal, but we have written the more general form here for future use.

11.1.2 Other Boundary Conditions

The model described in the preceding section is called the Ising model with *free boundary conditions*. It is somewhat ugly in its behavior at the boundary of the graph.

Toroidal Graph We can eliminate the boundary and restore the complete symmetry of the model by adopting a *toroidal* graph. Let $\mathbb{T}$ denote the discrete circle with d elements. We can identify $\mathbb{T}$ with the numbers 0, ..., $d - 1$ with addition and subtraction done modulo d. Now we define $G = \mathbb{T} \times \mathbb{T}$ and E by (11) as before. Now because of the modular addition and subtraction, there are no boundary points of the graph. The points 0 and $d - 1$ are adjacent in $\mathbb{T}$. This gives every node in G four neighbors. We again define the unnormalized density of the model by (12). The only thing that has changed is the definition of the edges of the graph.

The reason the graph is called "toroidal" is because if C is a circle (a true circle, not discrete), then $C \times C$ is a topological *torus*, topologically isomorphic to a geometric torus (donut shaped object). The physicists usually call this model the Ising model with *periodic boundary conditions*, the idea of that name being that a function on a circle is equivalent to a periodic function on the real line.

Conditioning on the Boundary An alternative way to deal with the boundary of the graph is to fix variables on the boundary. Let $\deg(i)$ denote the *degree* of the node i in the graph (G, E), which is the number of edges involving i, the cardinality of the set $\{ (i, j) : j \in G \text{ and } (i, j) \in E \}$.

Now we use the same graph and model as in Section 11.1.1 with the exception that all variables X_i with $\deg(i) < 4$ are held fixed. So (12) changes meaning. Not all variables x_i in (12) are random, some are held fixed. Those that are random, those with $\deg(i) = 4$ have a joint unnormalized density, now a conditional density given the other variables, that is given by (12).

There is almost no difference between the specification of the conditional and unconditional models. Clearly, this idea of fixing some set of variables to give a conditional model (conditioning on those fixed values) is more general. Any set of variables can be fixed, and the unnormalized conditional density is just the same as the original unnormalized joint density.

11.1.3 Phase Transitions

A ferromagnet undergoes phase transitions. By "phase transition" physicists mean an abrupt change in behavior as some variable changes continuously. The most familiar example is changes of state from solid to liquid to gas and vice versa. As temperature changes continuously there is an abrupt change in many properties of the material as the temperature goes past the melting point or the boiling point. Similarly, there is an abrupt change in the behavior of a magnet as the temperature goes past the *Curie temperature*. Below the Curie temperature, the material is magnetized. Above the Curie temperature, it is demagnetized.

In the Ising model, the parameter α plays the role of the external magnetic field, which for this discussion we take to be zero, and the parameter β plays the role of inverse temperature (as β goes to infinity the temperature goes to absolute zero, -273.15° C, as β goes to zero the temperature goes to infinity).

In the Ising model, $\alpha = 0$ and $\beta = 0$, gives a distribution proportional to counting measure on $\mathbb{S}^G$. The spins are then independent mean-zero random variables. And the net magnetization, which is proportional to

$$m_n(x) = \sum_{i \in G} x_i$$

has an asymptotic normal distribution by the central limit theorem (CLT). The standardized variable $m(X)/\sqrt{n}$ where $n = d^2$ is the number of nodes in the graph is approximately standard normal for large n. The net magnetization per node which, is $m_n(x)/n$, is approximately Normal$(0, 1/n)$, hence nearly zero.

In the Ising model, $\alpha = 0$ and $\beta \to \infty$, gives a distribution concentrated on two points, the two states in which all of the spins are aligned so

$$\sum_{(i,j) \in E} x_i x_j$$

has its maximal value ($4n$ with toroidal boundary conditions, $4n - 4d$ with free boundary conditions). And in either of these two (equally probable) states, $m_n(x)$ is maximal in absolute value, either $-n$ or $+n$, and $m_n(x)/n$ is either -1 or $+1$.

So much is trivial. What is not at all obvious, and what took 30 years after the invention of the model to prove rigorously, is that this behavior holds for other temperatures as well. There is a value

$$\beta_c = \frac{1}{2} \sinh^{-1}(1) = \frac{1}{2} \log(1 + \sqrt{2}) = 0.4406868 \tag{13}$$

such that for $\beta < \beta_c$ the behavior is essentially the same as at $\beta = 0$ for large n and for $\beta > \beta_c$ the behavior is essentially the same as at $\beta \to \infty$ for large n. If $\beta < \beta_c$, then $m_n(x)/n$ converges in probability to zero as n goes to infinity. If $\beta > \beta_c$, then $m_n(x)/n$ converges in probability to the random variable concentrated on the two-point set $\{-1, +1\}$ as n goes to infinity.

Ising invented this model in 1925 and proved that the one-dimensional version does not have a phase transition. Onsager proved that the two-dimensional model does have the behavior described here in 1944. Kindermann and Snell (1980) give the history and much more detail.

The behavior for $\beta = \beta_c$ is strange and beautiful. Realizations from the model exhibit fractal-like behavior with features at all scales.

11.2 The Potts Model

Potts (1952) invented a generalization of the Ising model in which the variables X_i at the nodes take values in an arbitrary finite set, which we continue to denote $\mathbb{S}$. To have a concrete name for the elements of $\mathbb{S}$, we will call them *colors*, as might be the case when a Potts model is used a prior distribution in Bayesian image reconstruction. Now it makes no sense to add or multiply the X_i, since $\mathbb{S}$ has no arithmetic structure. The only thing we can do with two values x_i and x_j is test for equality. Define the indicator function

$$e(x, y) = \begin{cases} 1, & x = y \\ 0, & x \neq y \end{cases}$$

Then we define the Potts model unnormalized density by

$$h(x) = \exp\left(\sum_{i \in G} \alpha_i(x_i) + \frac{1}{2} \sum_{(i,j) \in E} \beta_{ij} e(x_i, x_j)\right) \tag{14}$$

where $\alpha_i : \mathbb{S} \to \mathbb{R}$ is an arbitrary function and, as before, the β_{ij} are arbitrary constants. The $1/2$ in (14) corrects for double counting, just like the $1/2$ in (12).

As before, in physics we usually take $\alpha_i = 0$ for all i and the β_{ij} all the same, but we allow more generality for statistical applications.

The Potts model also undergoes phase transitions. The critical parameter value is

$$\beta_c = \log(1 + \sqrt{r})$$

where r is the cardinality of the set $\mathbb{S}$. Note that this agrees for the formula for the Ising model when $r = 2$ except for the extra factor $1/2$ in (13). This

factor arises from $e(x_i, x_j)$ taking values in $\{0, 1\}$ whereas in the Ising model $x_i x_j$ takes values in $\{-1, 1\}$. The Ising model is the special case of the Potts model when $\mathbb{S}$ has two elements. The fact that the two states are called "spins" and given the values -1 and $+1$ is inessential.

The behavior of the Potts model for large lattice sizes is similar to that of the Ising model. For $\beta < \beta_c$, the vector $m_n(x)$ that counts the number of x_i taking each of the r possible values has $m_n(x)/n$ converging in probability to zero. If $\beta > \beta_c$, then $m_n(x)/n$ converges in probability to the vector uniformly distributed on the r vertices of the unit simplex, $(1, 0, 0, \ldots)$, $(0, 1, 0, \ldots)$, and so forth.

11.3 Naive Metropolis and Gibbs

It is easy to do Gibbs or Metropolis using the *spatial Markov property* of these models. The conditional distribution of one X_i given the rest depends only on its neighbors.

Let x be the current state, and y be the same state as x except that x_i has a different value. Then the odds ratio for this Metropolis proposal is

$$\frac{h(y)}{h(x)} = \exp\left(\alpha_i(y_i - x_i) + \sum_{\substack{j \in G \\ (i,j) \in E}} \beta_{ij}(y_i - x_i)x_j\right)$$

simple to calculate because it only involves at most four neighboring nodes. The Metropolis update then does Metropolis rejection.

The Gibbs update chooses the two possible states (updating x_i only) which are x and y, the same states that Metropolis considers, moving to y with its conditional probability given the rest, which is

$$\frac{h(y)}{h(x) + h(y)}$$

in the notation we used for the Metropolis update. And it stays at x with probability

$$1 - \frac{h(y)}{h(x) + h(y)} = \frac{h(x)}{h(x) + h(y)}$$

The Potts model is not harder. It's naive Metropolis proposal moves from x to a y that changes only x_i and chooses among the r possible values uniformly among the $r - 1$ possible values that are different from the current

value (this is easily seen to be a symmetric proposal). Then

$$\frac{h(y)}{h(x)} = \exp\left(\alpha_i(y_i) - \alpha(x_i) + \sum_{\substack{j \in G \\ (i,j) \in E}} \beta_{ij} \bigl[e(y_i, x_j) - e(x_i, x_j) \bigr] \right)$$

almost the same, merely notational differences.

Now the Gibbs update can have any of r possible outcomes that differ from x only in the i-th coordinate. The conditional distribution of the Gibbs update is

$$p_i(y \mid x) = \frac{\exp\left(\alpha_i(y_i) + \sum_{\substack{j \in G \\ (i,j) \in E}} \beta_{ij} e(y_i, x_j)\right)}{\sum_{y^* \in \mathbb{S}} \exp\left(\alpha_i(y^*) + \sum_{\substack{j \in G \\ (i,j) \in E}} \beta_{ij} e(y^*, x_j)\right)}$$

(the denominator is the sum of the r possible numerators).

Gibbs and Metropolis are simple, but neither works well for large lattices except at very "hot" temperatures (very small β_{ij}, very weak dependence).

11.4 Swenden-Wang

From 1953 when the Metropolis algorithm was invented until 1987 when the Swendsen-Wang algorithm was invented, the Ising model was considered the archetypical *hard problem* for MCMC. The Swendsen-Wang algorithm made it *easy*.

It starts out on a way that seems at first counterintuitive, even bizarre. It complicates the problem by introducing new variables. The Swendsen-Wang algorithm (Swendsen and Wang, 1987) works for any Potts model on any graph (G, E) such that all of the coupling parameters β_{ij} are positive. The new variables are Bernoulli random variables Y_{ij} for $(i, j) \in E$. They are often called *bonds*, and can be thought of as inducing a subgraph of (G, E) in which only the edges with $Y_{ij} = 1$ are retained. This subgraph is also undirected, which requires that the matrix Y be symmetric: $Y_{ij} = Y_{ji}$ with probability one.

Having increased the number of variables and hence the state space of the Markov chain, we now need to specify the desired equilibrium distribution

on this new state space. We keep the same distribution (given by the Potts model) for the old variables (the X_i). Of course, this is now the marginal (for the vector X). We now specify a conditional for Y given X (where Y is the vector of all the bond variables), and that completes the specification of the joint distribution of X and Y. The Y_{ij} are conditionally independent (subject to symmetry) given X and

$$P(y_{ij} = 1|x) = \begin{cases} \gamma_{ij}, & x_i = x_j \\ 0, & \text{otherwise} \end{cases}$$

where the γ_{ij} are constants to be named later (we choose them after we see what makes the Swendsen-Wang algorithm simple).

The Swendsen-Wang algorithm is a block Gibbs algorithm. It samples from bonds given spins (Y given X), then from spins given bonds (X given Y). We have just seen one of these conditionals. Clearly it is simple to sample from because of the conditional independence of the Y_{ij}.

The conditional distribution of X given Y is a bit more complicated. First note that any two nodes connected by a bond must be the same color (must have the same value of their X_i). This is clearly also the case for any two nodes connected by a chain of bonds. Consider the graph (G, E_y) where

$$E_y = \{\, (i, j) \in E : Y_{ij} = 1 \,\}$$

the subgraph of (G, E) mentioned above where we let the bonds be the edges in the graph.

Let $\mathcal{A}_y$ be the partition of G induced by E_y. The elements of $\mathcal{A}_y$ are called the *maximal connected components* of (G, E_y) and there are highly efficient algorithms for constructing them (Dijkstra, 1976, Chapter 23). Formally, E_y is a relation having domain G. The *transitive reflexive closure* of this relation is the smallest equivalence relation (transitive, reflexive, and symmetric relation) that contains E_y. Every equivalence relation induces a partition and vice versa (elements are equivalent if and only if they are in the same partition). Thus the computer algorithm can also be considered an algorithm for finding equivalence classes (which is what Dijkstra, 1976 calls it).

We already know that the nodes of an element of $\mathcal{A}_y$ must have the same color (given $Y = y$). Thus we consider the probability distribution (given Y) of the colors of equivalence classes (elements of $\mathcal{A}_y$) which we now start calling *patches* to have a shorter name.

The unnormalized joint density of pixels and bonds is

$$h(x, y) = \prod_{i \in G} e^{\alpha_i(x_i)} \prod_{(i,j) \in E} \left[e^{\beta_{ij}} \gamma_{ij}^{y_{ij}} (1 - \gamma_{ij})^{1 - y_{ij}} \right]^{e(x_i, x_j)/2} (1 - y_{ij})^{1 - e(x_i, x_j)} \tag{15}$$

(where in the last term we are using the convention $0^0 = 1$, it is one unless $y_{ij} = 1$ and $e(x_i, y_j) = 0$). The division by 2 in the exponent in (15) corrects for double counting, just like the 1/2 in (14).

Our dictum that there is no difference between an unnormalized joint density and an unnormalized conditional density (they differ on in their normalizing constants) means (15) is also the unnormalized conditional of x given y considered as a function of x for fixed y.

Let $\mathcal{A}_y$ be as above. We already know that x has probability zero given y unless it is constant on elements of $\mathcal{A}_y$, that is, for all $A \in \mathcal{A}_y$, the value of x_i is the same for all $i \in A$. Call x that are constant on elements of $\mathcal{A}_y$, *patch respecting*.

Conversely, for a patch respecting x we have $e(x_i, x_j) = 0$ implies x_i and x_j are in different patches and hence $y_{ij} = 0$. Hence the last term in (15) is always equal to one (is never 0^1) for all patch respecting x. Since the other terms in (15) are never zero if we choose the γ_{ij} to be neither zero or one, we have proved that the conditional probability of x given y is nonzero if and only if x is patch respecting (with patches defined according to $\mathcal{A}_y$) subject to this restriction on the choice of the γ_{ij}.

Hence we now restrict attention to patch respecting x. Let $\sim$ denote the equivalence relation induced by $\mathcal{A}_y$ (we have $i \sim j$ if and only if there exists an $A \in \mathcal{A}_y$ that contains both i and j). Then we can write

$$\begin{aligned} h(x, y) = \exp\Bigg(& \sum_{i \in G} \alpha_i(x_i) \\ & + \frac{1}{2} \sum_{\substack{(i,j) \in E \\ i \sim j}} \Big[\beta_{ij} + y_{ij} \log(\gamma_{ij}) + (1 - y_{ij}) \log(1 - \gamma_{ij}) \Big] \\ & + \frac{1}{2} \sum_{\substack{(i,j) \in E \\ i \not\sim j}} \Big[\beta_{ij} + \log(1 - \gamma_{ij}) \Big] e(x_i, x_j) \Bigg) \end{aligned}$$

The simplification of the $\sim$ terms comes from $i \sim j$ implies $e(x_i, x_j) = 1$, and the simplification of the $\not\sim$ terms comes from $i \not\sim j$ implies $y_{ij} = 0$.

We now choose γ_{ij} to make the $\not\sim$ terms go away

$$1 - \gamma_{ij} = e^{-\beta_{ij}}, \qquad (i, j) \in E. \tag{16}$$

Then we note that the $\sim$ terms do not contain x and hence can be considered part of the normalizing constant for an unnormalized conditional density of x given y (that is, they can be dropped). Hence we can write (with this choice of γ_{ij})

$$\begin{aligned} h(x \mid y) &= \exp\left(\sum_{i \in G} \alpha_i(x_i)\right) \\ &= \prod_{A \in \mathcal{A}_y} \exp\left(\sum_{i \in A} \alpha_i(x_A)\right) \end{aligned} \tag{17}$$

where in the last expression we have written x_A for the common value of x on the patch A (we are still insisting that x be patch respecting). The conditional of x given y described by (17) is remarkably simple. Note that because of the product over patches, the the patch colors are conditionally independent given the bonds. It gets even simpler in the special case of interest to the physicists where $\alpha_i = 0$ for all i. Then all patch colors are equally likely.

We summarize.

- [Update y given x.] The bonds are conditionally independent given the pixels.
  - If $(i, j) \in E$ and $x_i = x_j$, then $Y_{ij} = 0$ with probability (16).
  - If $(i, j) \in E$ and $x_i \neq x_j$, then $Y_{ij} = 0$ with probability one.
- [Calculate patches.] Run an algorithm to determine the equivalence classes $\mathcal{A}_y$.
- [Update x given y.] The patch colors are conditionally independent given the patches (which are determined by the bonds). The probability of color c for patch A is proportional to

$$\exp\left(\sum_{i \in A} \alpha_i(c)\right)$$

What is amazing about the Swendsen-Wang algorithm is that it equilibrates very quickly, in about 50 iterations for a million pixel lattice. Naive Metropolis and Gibbs would take longer than the age of the universe on such a large lattice.

11.5 Lessons Learned

Since Swendsen and Wang (1987) appeared, other similar algorithms have been invented, all using the same clever trick with "bonds" but only for very similar models. The Swendsen-Wang idea has had little (perhaps no) application outside of spatial lattice processes. But the general idea

> Often the efficient way to do an MCMC problem is to *not* use the state space and equilibrium distribution that are given.
>
> The Swendsen-Wang algorithm uses a completely different state space (adding "bond" variables), which must then have a different equilibrium distribution. The new and old equilibrium distributions have a connection, for Swendsen-Wang, one is a marginal of the other.

Besag and Green (1993) coined the name *auxillary variables* algorithm for methods of the Swendsen-Wang type in which new variables are added to the state space. Parallel and serial tempering are other algorithms of this type, although they differ the relation between the new equilibrium distribution an the old. For tempering the old is a conditional of the new, not a marginal as in Swendsen-Wang.

Many other MCMC users have thought up "auxillary variables" algorithms. It's easy. Just add variables and stir. The only issues are the following.

1. One must be absolutely clear what is the state space of the new Markov chain (with added variables). Exactly what variables are added.

2. One must be absolutely clear what is the equilibrium distribution of the new Markov chain.

3. One must be absolutely clear that the proposed sampler is an instance of the Metropolis-Hastings-Green algorithm, or if not, that one has some proof that it preserves the equilibrium distribution given in 2.

4. One must be absolutely clear about the relation between the new and old distributions and how sampling the new distribution solves the problem as given.

Many people have gotten confused about one of these steps when thinking about auxiliary variable problems, often beginning with item 1. Variables are used but one forgets that they are thereby part of the state and must be part of the distribution in item 2 (and so forth).

12 Annealing and Tempering

12.1 Random Search Optimization

Random search optimization and MCMC are often confused, with some famous papers mixing the two without drawing clear distinctions. Let us draw the distinction clearly here ourselves.

> Random search optimization that is similar to MCMC is generally *much easier* than MCMC. One doesn't have to preserve any particular equilibrium distribution. One only needs to evaluate the objective function at a fairly dense set of points.

12.2 Simple Random Search Optimization

By *simple* random search optimization, we mean the points X_1, X_2, ... at which the objective function g is evaluated are an IID sample from some distribution f.

Suppose g is twice continuously differentiable in a neighborhood of the of the optimal value x^* and that f is also continuous on that neighborhood. Suppose that $\nabla^2 g(x^*)$ is positive definite so that x^* is a strong local minimum. Level sets of g for levels near $g(x^*)$ are then approximately hyper-ellipsoids

$$x^* + rA, \tag{18}$$

where A is a "unit" hyper-ellipsoid, and have hyper-volume $r^d V$ where V is the hyper-volume of A and d is the dimension of the space.

The probability of a point landing in rA is for small r nearly $r^d V f(x^*)$ by continuity of f. The probablity that at least one point in the first n lands in rA is approximately

$$1 - \bigl(1 - r^d V f(x^*)\bigr)^n. \tag{19}$$

Let $\hat{r}_n$ denote the smallest r such that (18) contains an X_i. Then (19) is also the approximate cumulative distribution function (CDF) of $\hat{r}_n$.

If we divide r^d by n in (19) and let n go to infinity, we get

$$1 - \exp\bigl(-r^d V f(x^*)\bigr) \tag{20}$$

and this says that

$$n V \hat{r}_n^d \xrightarrow{\mathcal{D}} \operatorname{Exp}\bigl(f(x^*)\bigr).$$

The actual details of the asymptotic argument here are not important, because we rarely do simple random search. But the qualitative results are important.

> Optimization is *different* from averaging. The rate is n not $n^{1/2}$. The asymptotic distribution is not normal.

That the rate is fast (n not root n) is an indication that optimization is easier.

Of course, the fast rate depends on the assumption of smoothness of the objective function. Nothing can be said about an arbitrarily obnoxious objective function.

12.3 Adaptive Random Search Optimization

By *adaptive* random search optimization, we mean the points X_1, X_2, ... at which the objective function g is evaluated are *not* IID but rather the distribution of X_n depends on $g(X_1)$, ..., $g(X_n)$. The idea is too look harder in the neighborhood of previously seen good X_i values (so one can get closer to the optimal value faster), but not to entirely stop looking in new regions of the state space (so one does not fail to find the optimum eventually, even if one's adaptive strategy is bad).

Nothing is easier than inventing adaptive random search algorithms. There are no requirements other than the one just mentioned that some proportion of the effort must be expended on exploring new regions so that one eventually finds the optimum. The most famous adaptive random search algorithms are those with catchy names, *simulated annealing* and *genetic algorithms*, and stories to go with the names.

People like stories and they also like catchy names, but what people like has little to do with performance of optimization.

> Vague analogies with metallurgy and genetics do noting to improve adaptive random search.

Comparison of simulated annealing and genetic algorithms with other simpler (and earlier invented) adaptive random search algorithms shows no performance advantages. One can usually invent something better than either simulated annealing or genetic algorithms for any particular problem. That being the case, we will say nothing about genetic algorithms, which are unrelated to MCMC and only enough about simulated annealing to explain how it has influenced MCMC.

12.4 Simulated Annealing

The most famous method of adaptive random search optimization, famous for its catchy name rather than any performance advantage it has over

other adaptive random search algorithms, is *simulated annealing*, which was proposed by Kirkpatrick et al. (1983). We are interested in it primarily because of its connections with MCMC. We are given an *objective function* f to minimize. Simulated annealing runs a Metropolis algorithm with equilibrium density

$$h_\tau(x) = e^{-f(x)/\tau}$$

where τ is the "temperature" of the distribution.

When τ is very large, then h_τ is nearly uniform (assuming that h_τ is integrable so a corresponding normalized density exists). When τ is nearly zero, then h_τ is nearly concentrated on the set of minimum values (the "argmin") of the function f. The idea of simulated annealing is to "cool" the process slowly, changing τ as the process proceeds. Let τ_k denote the sequence of taus, τ_k being used for the k-th iteration.

Since the distribution preserved by the Metropolis update (having unnormalized density h_{τ_k}) keeps changing, we have a Markov chain $x_1, x_2, \ldots$ with *nonstationary transition probabilities*, which cannot be analyzed according to Markov chain theory for chains with stationary transition probabilities (which is all we have discussed so far).

Theory for simulated annealing interesting mathematics but provides little guide for practical problems. We take as a reference Locatelli (2000), which not only provides a new theorem with somewhat different conditions from earlier work, but also provides a good survey of earlier work. This theory only applies to continuous objective functions defined on compact subsets of $\mathbb{R}^d$, but analogous theory holds for discrete domains (also discussed in Locatelli, 2000).

Let y_k be the "empirical argmin" any x_j such that

$$f(y_k) = \min_{i=1,\ldots,k} f(x_i)$$

Let A be the "theoretical argmin"

$$A = \{\, x : f(x) = \inf f(x) \,\}$$

we assume A is nonempty.

Let $d(y_k, A)$ denote the distance from y_k to A. Then Bélisle (1992) shows that the sequence $d(y_k, A)$ converges in probability to zero provided the proposal distribution $q_k(x, \cdot\,)$ for Metropolis updates, a density w. r. t. Lebesgue measure, implies the existence of $\rho > 0$ such that

$$q_k(x, y) \geq \rho, \qquad \textit{for all } x \textit{ and } y \textit{ in the domain of } f \textit{ and for all } k \tag{21}$$

This condition essentially makes simulated annealing behave like simple random search, because there is positive probability in going from anywhere to anywhere in one step.

Locatelli (2000, p. 126) complains this condition is not what is wanted

> The negative consequence [of Bélisle's condition (21)] is that, [a repeat of (21)]. Therefore, at any iteration, there is a probability bounded away from zero of sampling points in regions far from the global optimum region. Instead, we would like to be able to perform steps which are only local in order to explore more deeply the most promising parts of the feasible region.

Locatelli (2000) introduces technical conditions reflecting this informal expression that guarantee not only that $d(y_k, A)$ converges in probability to zero, but also that $d(x_k, A)$ converges in probability to zero, and also $x_{k+1} - x_k$ converges to zero. But in order to obtain this, cooling must be at a very slow rate

$$\tau_k = \frac{C}{\log k} \tag{22}$$

where C is a positive constant that depends on a complicated fashion on the particular problem structure (more precisely, see Locatelli, 2000, Assumption 3.3). The logarithmic cooling rate is seen in other simulated annealing theorems (discussed in Locatelli, 2000), including problem in which f has discrete domain.

The trouble with this body of theory (and we have no wish to pick on Locatelli, 2000, other theory is similar) is that the rate (22) is so slow that no one would ever use it in practice. Hence the theory is absolutely useless, an interesting theoretical exercise, but one that tells nothing useful about simulated annealing.

The quotation above from Locatelli (2000) is interesting because it shows exactly how and where this body of theory goes wrong. His point "we would like to be able to perform steps which are only local in order to explore more deeply the most promising parts of the feasible region" is completely wrong headed. This refusal to *ever* explore beyond a "local" region, once one has cooled sufficiently is the whole reason why such slow cooling is necessary. If "cooling" imposes irreversible limitations on the search, then it had better be very, very slow. If no such irreversible limitations are imposed, then the problem becomes much easier. A search for `reheated "simulated annealing"` in Google turns up many hits but apparently no authoritative reference. The notion that monotonic cooling is a bad idea

is apparently widespread, but I do not know who originated it. A form of "reheating" is the key to serial tempering (Section 12.6 below).

12.5 Parallel Tempering

What is now called *parallel tempering* was invented under the name *Metropolis-coupled MCMC*, abbreviated $(MC)^3$, and first appeared in Geyer (1991). Despite its origin in a lowly conference proceeding, it has become very popular under its new name, a back formation by analogy with simulated tempering, which is described in the following section. A Google search for `"parallel tempering"` turns up 18,400 hits and a Wikipedia entry.

The idea is very simple, although not completely obvious. Suppose we are interested in a finite set of distributions specified by unnormalized densities h_i, $i = 1, \ldots, d$. By analogy with simulated annealing, these might all be "heated" versions of a single basic distribution

$$h_i(x) = h(x)^{1/\tau_i}$$

where the τ_i are "temperatures" in the language of simulated annealing. But as we shall see there is no reason for using this particular form. There is no requirement other than that the h_i all have the same domain.

The state space of the parallel tempering Markov chain is the d-fold product of the common domain of all the h_i. Thus the state of the Markov chain is a d-vector $(x_1, \ldots, x_d)$ and in this section we will have subscripts indexing components of the state vector rather than iterations of the Markov chain.

The equilibrium distribution of the parallel tempering Markov chain is the product density

$$h(x) = \prod_{i=1}^{d} h_i(x_i)$$

that makes the components of the state vector independent under the equilibrium distribution (as we shall see, the components are not independent in the Markov chain).

The update of the parallel tempering Markov chain is a combined update with the following elementary updates.

- Update x_i preserving h_i.
- Choose an index pair (i, j) and propose to swap x_i and x_j. Accept the proposal with probability $\min(r, 1)$ where

$$r = \frac{h_i(x_j) h_j(x_i)}{h_i(x_i) h_j(x_j)}$$

These updates can be combined by any valid combining mechanism (composition or mixing). The first kind of update is not part of the parallel tempering idea. It is supposed that one already has ideas about simulating from the h_i that can be used here. Again, any valid method (Metropolis, Gibbs, whatever) can be used.

12.6 Serial Tempering

13 Monte Carlo Likelihood

MCMC is most commonly used (in statistics) for Bayesian inference, but that is not the only use. It is useful whenever there are probabilities and expectations that one wants to know and cannot be done by pencil an paper, by computer algebra systems (like Mathematica and Maple), by numerical integration, or by ordinary Monte Carlo.

In likelihood inference, it is often possible to calculate the likelihood. This is what makes likelihood inference so simple and powerful. If you can calculate the likelihood, then (if you use the "usual asymptotics") you can do everything.

Note the contrast with Bayesian inference. If you can compute the likelihood and the prior, then you've only gotten started. You usually need MCMC to calculate the posterior. In likelihood inference, if you can compute the likelihood, you're done. No fancy computational methods are needed. (You may need to calculate derivatives of the log likelihood by finite differences, but that's trivial.)

This there is no need for MCMC or even IIDMC when doing likelihood inference in the kinds of problems where a Bayesian needs MCMC. However, that is not the end of the story. There are harder likelihood problems.

13.1 Methods

For an area that is unknown to many statisticians, there is an amazing variety of methods.

Monte Carlo Likelihood Approximation (MCLA) Methods that directly approximate the log likelihood and indirectly its derivatives at all parameter values simultaneously using the importance sampling formula. Originated by Geyer and Thompson (1992) for the unknown normalizing constant models, by Thompson and Guo (1991) for missing data models, and by Gelfand and Carlin (1993), for the combination of the two. Theory developed in Geyer (1994).

Monte Carlo Newton-Raphson (MCNR) Methods that approximate first and second derivatives of the log likelihood and use Newton-Raphson iteration to find the maximum likelihood estimate. Originated by Penttinen (1984). Reinvented by several later authors who were unaware of Penttinen's work.

Markov Chain Stochastic Approximation (MCSA) Methods using Markov chains with *nonstationary* transition probabilities, roughly describable as attempting to simulate from h_θ when θ is changing, adjusting θ until the first derivative of the log likelihood at θ is nearly zero. Originated by Younes (1988) and by Moyeed and Baddeley (1991). IIDMC stochastic approximation is very old (Wasan, 1969), but theory for MCMC stochastic approximation was only developed in Benveniste et al. (1990). See Younes (1999) for a full development of MCSA likelihood approximation.

Monte Carlo EM (MCEM) and MM (MCMM) Methods that do a Monte Carlo version of the EM algorithm (Dempster, Laird, and Rubin, 1977) or the MM algorithm (Hunter and Lange, 2004). Originated by Wei and Tanner (1990) and Guo and Thompson (1992).

13.2 Unknown Normalizing Constant Models

Suppose

$$\mathcal{H} = \{ h_\theta : \theta \in \Theta \} \tag{23}$$

is a family of *unnormalized* densities w. r. t. some measure λ. This means the "normalizing constants"

$$c(\theta) = \int h_\theta(x) \lambda(dx) \tag{24}$$

are nonzero and finite for each $\theta \in \Theta$. Note that the "normalizing constants" depend on the parameter θ, so *normalizing function* $c : \Theta \to (0, \infty)$ would be better terminology. Then

$$f_\theta(x) = \frac{1}{c(\theta)} h_\theta(x)$$

define the normalized densities of a statistical model.

Since f_θ is entirely determined by h_θ through (24), we can consider (23) a perfectly good specification of a statistical model. Before MCMC we didn't specify models this way because we didn't know what to do with

unnormalized densities. Now we do. MCMC has no trouble sampling from h_θ. If you can sample, you can calculate probabilities and expectations by Monte Carlo. If you can calculate probabilities and expectations, then you can do statistical inference.

The log likelihood for the model (23) is given by

$$l(\theta) = \log h_\theta(x) - \log c(\theta) \tag{25}$$

where x is the observed data (when the model is correct, a realization from some f_θ). When the integral in (24) is intractable, we must do it by some means if we are to calculate (25). Sometimes it can be done by numerical integration. Since this is a course in MCMC, we consider doing it by MCMC (with IIDMC a special case).

We use not only MCMC but importance sampling. To do the integral in (24) by MCMC in any way, we must write it as an expectation. We write it as an expectation w. r. t. the distribution with unnormalized density h^* w. r. t. λ.

$$c(\theta) = \int \frac{h_\theta(x)}{h^*(x)} h^*(x) \lambda(dx) = E^* \left\{ \frac{h_\theta(X)}{h^*(X)} \right\} \tag{26}$$

In order for (26) to make sense we must *never have divide by zero* in (26). More formally, we assume

$$h^*(x) = 0 \text{ implies } h_\theta(x) = 0, \qquad \text{for all } \theta \in \Theta \text{ and for } \lambda\text{-almost-all } x$$

(that we have 0/0, which is undefined, on a set of λ measure zero does not cause any problems).

Suppose X_1, X_2, ... is a Markov chain with equilibrium distribution h^*. Then the natural estimator of $c(\theta)$ is

$$c_n(\theta) = \frac{1}{n} \sum_{i=1}^{n} \frac{h_\theta(X_i)}{h^*(X_i)} \tag{27}$$

and the natural estimator of (25) is

$$l_n(\theta) = \log h_\theta(x) - \log \left(\frac{1}{n} \sum_{i=1}^{n} \frac{h_\theta(X_i)}{h^*(X_i)} \right) \tag{28}$$

The connection with importance sampling is somewhat vague, but apparent. Clearly, (26) uses the theoretical importance sampling formula, to change an integral w. r. t. $\lambda(dx)$ into an integral w. r. t. $h^*(x)\lambda(dx)$. Moreover, (27) looks like the denominator in normalized importance weights. But

(28) does not look much like importance sampling. The connection, however, becomes more apparent when we differentiate (all derivatives are with respect to θ).

$$\begin{aligned}
\nabla l_n(\theta) &= \nabla \log h_\theta(x) - \frac{\frac{1}{n}\sum_{i=1}^n \frac{\nabla h_\theta(X_i)}{h^*(X_i)}}{\frac{1}{n}\sum_{i=1}^n \frac{h_\theta(X_i)}{h^*(X_i)}} \\
&= \nabla \log h_\theta(x) - \frac{\sum_{i=1}^n \nabla \log h_\theta(X_i) \frac{h_\theta(X_i)}{h^*(X_i)}}{\sum_{i=1}^n \frac{h_\theta(X_i)}{h^*(X_i)}} \qquad (29a) \\
&= \nabla \log h_\theta(x) - \sum_{i=1}^n \bigl[\nabla \log h_\theta(X_i)\bigr] w^*_{n,\theta}(X_i)
\end{aligned}$$

where the $w^*_{n,\theta}(X_i)$ are normalized importance weights

$$w^*_{n,\theta}(x) = \frac{\frac{h_\theta(x)}{h^*(x)}}{\sum_{i=1}^n \frac{h_\theta(X_i)}{h^*(X_i)}} \qquad (29b)$$

for shifting a sample w. r. t. h^* to calculate expectations w. r. t. h_θ. Thus we write

$$\nabla l_n(\theta) = \nabla \log h_\theta(x) - E^*_{n,\theta}\bigl\{\nabla \log h_\theta(X_i)\bigr\} \qquad (29c)$$

where we take the second term on the right hand side to be a shorthand for the second term on the right hand side in (29a). Or, more formally, we could define for any integrable function g

$$E^*_{n,\theta}\bigl\{g(X_i)\bigr\} = \sum_{i=1}^n g(X_i) w^*_{n,\theta}(X_i)$$

This is the importance sampling estimate of $E_\theta\{g(X)\}$ using normalized importance weights and a sample of size n from h^*.

If the likelihood achieves its maximum at a point in the interior of the parameter space (where the first derivative is zero), then we can get a Monte Carlo approximation to the MLE by setting the Monte Carlo approximation of the log likelihood to zero and solving for θ. From (29c) we see that this entails setting $\nabla h_\theta(x)$ equal to its expectation calculated by MCMC and solving for θ.

Before we go further down that path, let us look at the second derivative.

$$\begin{aligned}\nabla^2 l_n(\theta) = \nabla^2 \log h_\theta(x) &- \frac{\frac{1}{n}\sum_{i=1}^n \frac{\nabla^2 h_\theta(X_i)}{h^*(X_i)}}{\frac{1}{n}\sum_{i=1}^n \frac{h_\theta(X_i)}{h^*(X_i)}} \\ &+ \left(\frac{\frac{1}{n}\sum_{i=1}^n \frac{\nabla h_\theta(X_i)}{h^*(X_i)}}{\frac{1}{n}\sum_{i=1}^n \frac{h_\theta(X_i)}{h^*(X_i)}}\right)\left(\frac{\frac{1}{n}\sum_{i=1}^n \frac{\nabla h_\theta(X_i)}{h^*(X_i)}}{\frac{1}{n}\sum_{i=1}^n \frac{h_\theta(X_i)}{h^*(X_i)}}\right)^T\end{aligned} \tag{30a}$$

The term of the form (vector)(vector)^T is an outer product. If v_j are the components of the vector in questions, then the outer product is the matrix with j, k element $v_j v_k$. Readers who find the vector notation unintitive should calculate with coordinates, meaning calculate $\partial^2 l_n(\theta)/\partial\theta_j\partial\theta_k$, and satisfy themselves that the matrix whose j, k entries are these partials, which is $\nabla^2 l_n(\theta)$ does agree with (30a).

We would further like to put (30a) in a form that involves only expectations (calculated by importance sampling) of derivatives of $\log h_\theta$. To do this we need

$$\nabla \log h = \frac{\nabla h}{h}$$

hence

$$\nabla^2 \log h = \frac{\nabla^2 h}{h} - \left(\frac{\nabla h}{h}\right)\left(\frac{\nabla h}{h}\right)^T$$

So

$$\begin{aligned}\nabla^2 l_n(\theta) = \nabla^2 \log h_\theta(x) &- E_{n,\theta}^*\bigl\{\nabla^2 \log h_\theta(X_i)\bigr\} \\ &- E_{n,\theta}^*\Bigl\{\bigl[\nabla \log h_\theta(X_i)\bigr]\bigl[\nabla \log h_\theta(X_i)\bigr]^T\Bigr\} \\ &+ E_{n,\theta}^*\bigl\{\nabla \log h_\theta(X_i)\bigr\} E_{n,\theta}^*\bigl\{\nabla \log h_\theta(X_i)\bigr\}^T\end{aligned} \tag{30b}$$

(we get two terms involving outer products of first derivatives, one with the outer product inside the expectation, one with the expectations inside the outer product). The reason for writing everything (except the importance weights) in terms of derivatives of $\log h_\theta$ is that these are generally numerically much better behaved than derivatives of h_θ itself.

13.2.1 MCLA

The acronym MCLA is retrofitted; Geyer and Thompson (1992) and Geyer (1994) did not call it that. It is often called "simulated maximum likelihood" by people who do not understand it and don't like it. As we shall

see, it unifies all the other methods, which are in various ways simplifications of it. Whenever any of the other methods perform badly, the answer to the problem lies in applying MCLA correctly.

As with every importance sampling algorithm, the performance depends on having a good importance sampling distribution h^*. It may take some preliminary trial and error to find a good h^* Geyer and Thompson (1992). The method of umbrella sampling (Section 13.5 below) can always be used to find a good h^*.

Since (28) approximates the log likelihood and (30a) minus the observed Fisher information, we have everything we need for likelihood inference, if we can apply the "usual asymptotics" for maximum likelihood. Even if the asymptotics do not apply, Geyer (1991) shows that we can easily do a parametric bootstrap.

Let X_1, X_2, ... be an MCMC sample from h^* as used above. Suppose $\hat{\theta}_n$ is the Monte Carlo MLE (MCMLE), the maximizer of (28). This is our best approximation of the true unknown parameter value. Let $\widetilde{X}_1$, $\widetilde{X}_2$, ... be an MCMC sample from $h_{\hat{\theta}_n}$. This is our best approximation of the sampling distribution of the data under the true unknown parameter value. For each $\widetilde{X}_j$, maximize (28) with x replaced by $\widetilde{X}_j$, denoting the maximizer $\tilde{\theta}_j$. Then the distribution of the $\tilde{\theta}_j$ is the parametric bootstrap distribution of the MLE and can be used for inference that does not depend on the validity of the "usual" asymptotics. (It does depend on the sampling distribution of the MLE not depending too much on θ so that our use of the distribution under $\hat{\theta}_n$ rather than under the true unknown θ does not make much difference, but that's all. It does not depend on asymptotic normality of anything.) Thus with only two Monte Carlo samples, we can find out everything. (Of course, in practice, one usually does more than two runs of the Markov chain just tuning the Metropolis algorithm before starting serious sampling.)

MCLA is also the only known method of calculating likelihood ratios for likelihood ratio tests. The other methods avoid calculating the likelihood itself. Thus MCLA is the only known method of dealing with situations where the "usual asymptotics" of maximum likelihood do not apply. As we shall see, the other methods are intolerably slow, are not automatic, and hence cannot be bootstrapped. Nothing like the bootstrap in Geyer (1991) has ever been done for the competing methods.

13.2.2 MCNR

In the special case where $h^* = h_\theta$ all normalized importance weights (29b) are equal, and the MC approximations of the log likelihood derivatives

become

$$\nabla l_n(\theta) = \nabla \log h_\theta(x) - \frac{1}{n} \sum_{i=1}^{n} \nabla \log h_\theta(X_i) \tag{31a}$$

and

$$\begin{aligned} \nabla^2 l_n(\theta) = {} & \nabla^2 \log h_\theta(x) - \frac{1}{n} \sum_{i=1}^{n} \nabla^2 \log h_\theta(X_i) \\ & - \left(\frac{1}{n} \sum_{i=1}^{n} \big[\nabla \log h_\theta(X_i)\big] \big[\nabla \log h_\theta(X_i)\big]^T \right) \\ & + \left(\frac{1}{n} \sum_{i=1}^{n} \nabla \log h_\theta(X_i) \right) \left(\frac{1}{n} \sum_{i=1}^{n} \nabla \log h_\theta(X_i) \right)^T \end{aligned} \tag{31b}$$

and still make sense, although the equation for the likelihood itself becomes meaningless (we can't vary θ and also hold it fixed so $h_\theta = h^*$).

But having the log likelihood derivatives means Newton-Raphson is possible. If we have evaluated at a point θ, then the Newton-Raphson update moves to

$$\theta + \big(-\nabla^2 l_n(\theta)\big)^{-1} \nabla l_n(\theta)$$

assuming the matrix being inverted is positive definite (if it isn't, then Newton-Raphson makes no sense as an attempt at maximization).

This problem with lack of positive definiteness is only part of the problem with MCNR. When the matrix is positive definite but nearly singular, then the Newton step will often be very bad.

A Digression on Newton's Method Many naive users have too much respect for Newton-Raphson. It can behave very badly on even the best behaved problems.

Consider maximum likelihood for a binomial model with x successes in n trials. If we use the canonical parameterization (logit of the success probability), the log likelihood is

$$l(\theta) = -x \log\left(1 + e^{-\theta}\right) - (n - x) \log\left(1 + e^{\theta}\right)$$

and if $0 < x < n$ is a strictly concave function with a unique maximum at $\hat{\theta} = \operatorname{logit}(x/n)$. But $l''(\theta)$ converges to zero as θ goes to infinity or minus infinity, and Newton-Raphson does not converge (it diverges to infinity) when not started close enough to the maximum.

This badness of Newton-Raphson is well known. Authoritative textbooks Fletcher (1987); Nocedal and Wright (1999) never recommend its use. What they do recommend is Newton-Raphson modified by *safeguarding*, meaning any of several methods, the most popular of which are line search and trust regions, of assuring the algorithm always goes uphill on the objective function (when maximizing) and makes sufficient progress in each iteration.

What is good about Newton's method is that it is the best possible method *when started near enough to the solution*, a fact sometimes called the Dennis-Moré theorem in optimization: every optimization method that has superlinear convergence is asymptotically equivalent to Newton-Raphson Fletcher (1987, Theorem 6.2.3), where here the "asymptotically" has nothing to do with the kind of asymptotics we use in statistics, but refers to the rate of convergence. An iterative method with iterates θ_1, θ_2, ... is said to be *linearly convergent* if it converges to a point θ^* and

$$\|\theta_{k+1} - \theta^*\| = O(\|\theta_k - \theta^*\|) \tag{32a}$$

superlinearly convergent if

$$\|\theta_{k+1} - \theta^*\| = o(\|\theta_k - \theta^*\|) \tag{32b}$$

and *quadratically convergent* if

$$\|\theta_{k+1} - \theta^*\| = O(\|\theta_k - \theta^*\|^2) \tag{32c}$$

Note that mere linear convergence is very slow; by itself (32a) does not even assure that the sequence of iterates converges (that's an additional assumption), although (32b) and (32c) do assure the sequence converges. Another method with iterates ψ_1, ψ_2, ... is said to be asymptotically equivalent to the first method if

$$\|\psi_{k+1} - \theta_{k+1}\| = o(\|\theta_{k+1} - \theta_k\|)$$

so when one method makes small steps, the other method makes nearly the same steps. Newton-Raphson has quadratic convergence on well behaved functions (Fletcher, 1987, Theorem 3.1.1). The Dennis-Moré theorem says that you either get that rate of convergence and you do so by being nearly the same as Newton-Raphson when close to the solution, or you have mere linear convergence (which is slow). If a linearly convergent algorithm takes 100 iterations to reduce the error from 0.1 to 0.01, then it will take another 100 iterations to reduce the error to 0.001, another 100 to reduce to 10^{-4}, and so forth. Newton-Raphson (or asymptotically equivalent algorithms) go from 0.01 to 10^{-4} in one step, then to 10^{-8} in the next step, then 10^{-16} in the next (which is the precision of computer arithmetic).

Back to Monte Carlo None of the preceding discussion had anything to do with Monte Carlo. With Monte Carlo approximation, we never get high accuracy, so it is unclear what the relevance of the Dennis-Moré theorem is. The bad behavior of Newton-Raphson when far from the solution does carry over to the Monte Carlo situation. It is just not safe to use without safeguarding. But none of the MCNR literature seems to be aware of the need for safeguarding. One suspects that anyone who recommends MCNR without any sort of safeguarding has never actually used it. They used the MCNR update, of course, but they also supplied some form of safeguarding, some form of trial and error, some form of human intervention, that strictly speaking is not part of the method.

Note that the most fundamental form of safeguarding discussed by optimization textbooks, assuring that the method always takes uphill steps, is not possible in MCNR, because it refuses to calculate the objective function, only derivatives. To apply safeguarding to MCNR, we must go back to MCLA!

On the other hand, when close to the solution, MCLA and MCNR nearly agree on the step to the maximum (because the importance weights are nearly equal and Newton-Raphson needs no safeguarding when sufficiently close to the solution). Thus an MCNR update that is the last one intends to do (the one after which one will declare the algorithm converged) does nearly the same thing as MCLA maximum likelihood. Thus MCNR can never be better than MCLA and usually is worse.

Also note that MCLA is not an iterative algorithm in the same sense that MCNR is. MCNR requires repeated MCMC sampling. We need samples from each h_{θ_k} that are the "iterates" of MCNR. As the method converges the distributions are much the same, and if we were willing to use the importance sampling formula, we could avoid the repeated sampling. But that use of importance sampling would be MCLA. Seen in this light, MCNR is simple refusal to gain the benefits of importance sampling in MCMC coupled with naivete about the dangers of Newton-Raphson when far from the solution.

13.2.3 MCSA

MCSA uses only first derivatives without importance sampling (31a), but it continuously adjusts the θ in the distribution with unnormalized density $h^* = h_\theta$ preserved by the MCMC update, thus producing a Markov chain with nonstationary transition probabilities. Traditionally, the adjustment occurs in each step and has the form

$$\theta_{k+1} = \theta_k + \epsilon_k \bigl[\nabla \log h_{\theta_k}(x) - \nabla \log h_{\theta_k}(X_k) \bigr] \tag{33}$$

Note that the term in brackets is a very crude approximation to the score (the first derivative of the log likelihood), the special case of (31a) where $n = 1$. The reason why we use only the $n = 1$ case is that if ϵ_k is small and slowly changing, so we have θ_k also slowly changing, and if h_θ is continuous so h_{θ_k} it is also slowly changing, then we get

$$\theta_{k+m} \approx \theta_k + m\epsilon_k \left[\nabla \log h_{\theta_k}(x) - \frac{1}{m} \sum_{i=k}^{k+m} \nabla \log h_{\theta_k}(X_i) \right]$$

so it makes little difference whether one adjusts every step or every m steps.

Younes (1999) shows how very difficult it is to assure convergence of MCSA. Very strong (unverifiable in practice) regularity conditions concerning the Markov chain transition mechanism are required, and even then explicit recommendations for the sequence ϵ_n are not a sufficient guide to practice. Also Younes (1999) considers a much more general scheme than updating every m steps. Instead we update at constant parameter value θ_k for m_k steps, where m_k can depend on θ_k as well as k. The update is

$$\theta_{k+1} = \theta_k + \epsilon_k \left[\nabla \log h_{\theta_k}(x) - \frac{1}{m_k} \sum_{i=m_1+\cdots+m_{k-1}+1}^{m_1+\cdots+m_k} \nabla \log h_{\theta_k}(X_i) \right] \tag{34}$$

Then Younes (1999) considers various schemes in which ϵ_k goes to zero at various rates while m_k goes to infinity. But such complicated conditions provide little guide for practice. (This is not meant to disparage Younes's work. The asymptotics of MCSA is a very difficult area. Getting any results at all is a major theoretical accomplishment. But that doesn't help applied people very much.)

We do not consider schemes in which m_k grows, because when m_k is very large, then one would do better to switch to MCLA. Again we see that MCLA provides the unifying framework and competitor to all the other methods. Once we consider switching from MCSA to MCLA, we see that there is no point to actually running MCSA "to convergence" because we should switch to MCSA or MCNR when we get close enough so that they work.

Nevertheless, MCSA is a useful tool because it does provide a simple means of "getting close" that is easily implemented. If one has an MCMC sampler and one can modify the source code, then adding a simple MCSA update (33) of the parameter is very easy so long as one has a closed-form expression for the derivative $\nabla \log h_\theta(x)$.

If one cannot or does not want to modify the source code for the sampler, then one can still easily implement (34) with $m_k = m$ a constant sequence by putting the sampler in a loop. For example, supposing that `logh(x, theta)` is an R function that evaluates $h_\theta(x)$ and that `dellogh(x, theta)` is an R function that evaluates $\nabla h_\theta(x)$ the following loop (which has not been actually tested, since these notes aren't done using `Sweave`)

```
out <- metrop(logh, xstart, nbatch = 1, blen = m,
    outfun = dellogh, scale = scale)
for (k in 1:niter) {
    out <- metrop(out, theta = theta)
    theta <- theta + epsilon * as.numeric(out$batch)
}
```

should work when `xstart`, `m`, `theta`, `scale`, and `niter` have been previously defined. Here we don't bother to make `epsilon` a decreasing sequence, since we haven't discussed that yet. But we could also insert

```
    epsilon <- epsilon * rho
```

at the bottom of the loop, where `rho` has been previously defined and is some number just a little less than 1.

And this does bring us to schemes for adjusting ϵ_k. As Younes (1999) notes, the traditional IID stochastic approximation criteria are that

$$\sum_{k=1}^{\infty} \epsilon_k = \infty$$

so an infinite amount of adjustment over an infinite run of the chain is possible (otherwise the scheme could never get to the correct θ if started too far away) but

$$\sum_{k=1}^{\infty} \epsilon_k^\alpha < \infty$$

for some $\alpha > 1$. Simple sequences that satisfy these criteria are

$$\epsilon_k = \frac{a}{k^\beta} \tag{35}$$

where $a > 0$ and $1/\alpha < \beta \le 1$. The smaller β is, the more slowly ϵ_k goes to zero and the more chance MCSA has to converge (but also the more time it takes).

In our experience, there is little point in worrying about ϵ_k sequences if one is willing to apply some human input to the process. Just run at fixed ϵ (any fixed ϵ) and look at the sequence θ_k for guidance. Does it appear to be wandering around haphazardly, at least in the latter portion of the run? Then ϵ is too large. Decrease it. Does it appear to move almost monotonically, still heading in the direction of the solution, but still a ways away? Then ϵ is too small. Increase it.

In the current state of the art, one has two choices. One can do a huge amount of theory (like Younes, 1999) and still be unable to decide on a particular a and β to use in (35). Or one can use trial and error.

13.2.4 MCMM

The EM algorithm applies only to the missing data situation, so it does not apply here in the unnormalized density case. Hunter and Lange (2004) name the MM algorithm a generalization of the EM algorithm that does not involve missing data or conditional expectation. The EM algorithm had a long history before Dempster et al. (1977), a history they and their discussants cite. Dempster et al. (1977) only gave it a name and popularized it. The theorems in Dempster et al. (1977) that are correct were not new (several of their "theorems" are not theorems, being incorrect beyond patching). Similarly, the MM algorithm is not new. Hunter and Lange (2004) recap a long history in the optimization literature. Hunter and Lange (2004) have only given it a new name (MM patterned after EM) and hope to popularize it among statisticians.

The MM stands for minorize-maximize (or majorize-minimize, if one is minimizing the objective function). Suppose we maximize (as in maximum likelihood), the objective function is l, and there is a *minorizing function* $q(\theta \mid \theta')$ satisfying

$$q(\theta \mid \theta') \le l(\theta), \qquad \text{for all } \theta \tag{36a}$$

and

$$q(\theta' \mid \theta') = l(\theta'). \tag{36b}$$

Property (36a) is the minorization property. Together these properties assure that if θ_1, θ_2, ... is a sequence such that

$$q(\theta_{k+1} \mid \theta_k) > q(\theta_k \mid \theta_k), \qquad \text{for all } k \tag{36c}$$

then (36a), (36b), and (36c) together assure that

$$l(\theta_{k+1}) > l(\theta_k), \qquad \text{for all } k \tag{36d}$$

This is just like the proof for EM, in fact EM is a special case of MM. What is different about EM, is that EM requres that the minorization inequality (36a) must be produced by application of Jensen's inequality to certain conditional expectations. Any other way of producing a minorization isn't EM but the more general MM. Hunter and Lange (2004) are certainly right in pointing out that ther

13.2.5 The Acid Test

Not a method of finding a solution, but simply a method of *checking* a solution (which is a much simpler problem). Suppose one has a θ that one wants to *check* whether it is an MLE or not. More precisely, we check whether θ satisfies the first order necessary condition for a maximum of the log likelihood (the score is zero). To do this we simulate a Markov chain having equilibrium distribution with unnormalized density h_θ and evaluate (31a). It should be zero, at least to within MCSE.

We should *always do this check*. No matter what method one uses to find a putative solution, if the solution doesn't pass this test (which is much simpler to do than any of the proposed methods of finding a solution), then the putative solution isn't a solution.

Since all of the methods can fail, only the foolish refuse to check whether they have failed.

13.3 Missing Data Models

This section covers models that are specified as joint densities $f_\theta(x, y)$ where x is missing and y is observed. It includes an models with unobserved variables, whether these variables are potentially observable or not. When the variables x are not observable under any circumstances, they are often called *latent variables* or *random effects*, but there is no difference as far as we are concerned between the two situations. Missing is missing, whether it is accidental (data that could have been observed were not) or deliberate (description of the model involves unobservable random variables). Another application of the methodology in this section is so-called *empirical Bayes* estimation, which is not, despite the name, Bayesian

> No one is less Bayesian than an empirical Bayesian.
>
> — D. V. Lindley

Empirical Bayes is just missing data maximum likelihood with a shift in terminology. What the likelihoodist calls latent variables, a Bayesian calls

parameters, because to a Bayesian any unobservable variable is a parameter and all uncertainty is described by probability, hence they are random variables having a distribution. So far so Bayesian. What the likelihoodist calls parameters, the Bayesian calls hyperparameters. In a (fully) Bayesian analysis, the hyperparameters also get a prior distribution, and if there are parameters of the hyperprior, these hyperhyperparameters, also get a prior, an so forth indefinitely in an arbitrarily deep hierarchy. But a so-called empirical Bayes analysis just maximizes the same function the likelihoodist would call the likelihood and the empirical Bayesian calls the posterior, considered as a function of the (hyper)parameters. They are maximizing the same function and reporting the same numbers, only the terminology is different.

The likelihood is, of course, the probability of the data considered as a function of the parameter. What is special about missing data problems is that this is the probability of the *observed* data y, not the joint data (often called "complete data" in this context, x and y). Thus the likelihood is

$$L(\theta) = f_\theta(y) = \int f_\theta(x, y) \mu(dx) \tag{37}$$

assuming f_θ is a density w. r. t. a product measure $\mu \times \nu$. What makes missing data hard is that the integral in (37) is often intractable. In fact, the only widely used missing data models in which the integral is tractable are the classical normal random effects models where the joint distribution of x and y is multivariate normal.

Even in cases where the classical EM algorithm can be used to maximize (37) even though it cannot be computed explicitly, the EM algorithm does not compute likelihood ratios, Fisher information, and many other quantities needed for a full likelihood analysis. Hence these methods are useful even in those cases. Only when there is no problem computing (37) and its derivatives is there no scope for the methods of this section.

13.3.1 MCLA

This being a course on MCMC, we compute the integral in (37) by MCMC. We suspect from the preceding section that we need importance sampling, and in any event we need to convert the integral in (37) to an

expectation in order to use any form of Monte Carlo. Thus we write

$$\begin{aligned} L(\theta) &= \int \frac{f_\theta(x,y)}{f^*(x,y)} f^*(x,y) \mu(dx) \\ &= E^* \left\{ \frac{f_\theta(X,Y)}{f^*(X,Y)} \;\middle|\; Y = y \right\} \end{aligned} \tag{38}$$

again we have the requirement of no divide by zero, or more precisely

$$f^*(x,y) = 0 \text{ implies } f_\theta(x,y) = 0, \qquad \text{for all } \theta \text{ and for } \mu\text{-almost-all } x$$

where y here is the observed value of this variable. Unlike the case in the preceding section f^* and f_θ are *normalized* densities (the case where they are unnormalized is the following section).

Now let X_1, X_2, ... be a Markov chain having equilibrium distribution with unnormalized density $f^*(\,\cdot\,, y)$. Since y is fixed, we are sampling the conditional distribution of x given y corresponding to the joint density f^*. We also see that it makes no difference whether f^* is normalized, as should have been obvious from the begininning, since it never makes a difference if we change the likelihood by a multiplicative constant that does not depend on the parameter (the normalizing constant for f^* does not depend on θ). The natural Monte Carlo estimate of (38) is

$$L_n(\theta) = \frac{1}{n} \sum_{i=1}^n \frac{f_\theta(X_i, y)}{h^*(X_i, y)} \tag{39}$$

where we have changed notation from f^* to h^* to recognize our realization that h^* may be unnormalized, though f_θ must be normalized. The corresponding log likelihood

$$l_n(\theta) = \log \left(\frac{1}{n} \sum_{i=1}^n \frac{f_\theta(X_i, y)}{h^*(X_i, y)} \right) \tag{40}$$

As in the case in the preceding section, we record its derivatives

$$\begin{aligned} \nabla l_n(\theta) &= \frac{\sum_{i=1}^n \nabla \log f_\theta(X_i, y) \frac{f_\theta(X_i,y)}{h^*(X_i,y)}}{\sum_{i=1}^n \frac{f_\theta(X_i,y)}{h^*(X_i,y)}} \\ &= \sum_{i=1}^n \nabla \log f_\theta(X_i, y) w^*_{n,\theta}(X_i, y) \\ &= E^*_{n,\theta} \big\{ \nabla \log f_\theta(X, Y) \;\big|\; Y = y \big\} \end{aligned} \tag{41}$$

the notation in the last line suggesting that the weighted average in the preceding line does indeed estimate the conditional expectation

$$\begin{aligned}\nabla \log L(\theta) &= \frac{1}{L(\theta)} E^* \left\{ \frac{\nabla f_\theta(X,Y)}{h^*(X,Y)} \;\middle|\; Y = y \right\} \\ &= \frac{E^* \left\{ \frac{\nabla f_\theta(X,Y)}{h^*(X,Y)} \;\middle|\; Y = y \right\}}{E^* \left\{ \frac{f_\theta(X,Y)}{h^*(X,Y)} \;\middle|\; Y = y \right\}}\end{aligned}$$

where in both of the above equations the normalized importance weights are

$$w^*_{n,\theta}(x,y) = \frac{\frac{f_\theta(x,y)}{h^*(x,y)}}{\sum_{i=1}^n \frac{f_\theta(X_i,y)}{h^*(X_i,y)}} \tag{42}$$

Also

$$\begin{aligned}\nabla^2 l_n(\theta) = {} & \frac{\sum_{i=1}^n \frac{\nabla^2 f_\theta(X_i,y)}{h^*(X_i,y)}}{\sum_{i=1}^n \frac{f_\theta(X_i,y)}{h^*(X_i,y)}} \\ & - \left(\frac{\sum_{i=1}^n \frac{\nabla f_\theta(X_i,y)}{h^*(X_i,y)}}{\sum_{i=1}^n \frac{f_\theta(X_i,y)}{h^*(X_i,y)}} \right) \left(\frac{\sum_{i=1}^n \frac{\nabla f_\theta(X_i,y)}{h^*(X_i,y)}}{\sum_{i=1}^n \frac{f_\theta(X_i,y)}{h^*(X_i,y)}} \right)^T \\ = {} & E^*_{n,\theta} \big\{ \nabla^2 \log f_\theta(X,Y) \;\big|\; Y = y \big\} \\ & + E^*_{n,\theta} \Big\{ \big[\nabla \log f_\theta(X,Y) \big] \big[\nabla \log f_\theta(X,Y) \big]^T \;\Big|\; Y = y \Big\} \\ & - E^*_{n,\theta} \big\{ \nabla \log f_\theta(X,Y) \;\big|\; Y = y \big\} E^*_{n,\theta} \big\{ \nabla \log f_\theta(X,Y) \;\big|\; Y = y \big\}^T\end{aligned} \tag{43}$$

As in (30b) we get two terms involving outer products of first derivatives, one with the outer product inside the expectation, one with the expectations inside the outer product.

MCLA is not much different in this context than in the unknown normalizing constant context. The formulas are slightly different. Geyer (1994) comments that this form of MCLA is somewhat less well behaved because the random term is positive rather than negative. Since the terms being averaged (the unnormalized importance weights) are bounded below by zero but unbounded above, this means that outliers, when they occur are positive for missing data MCLA but negative in unknown normalizing constant MCLA, and when maximizing positive outliers have more effect than negative outliers. This was noticed because stronger regularity conditions are required

in the consistency for missing data MCLA, but once the phenomenon has been pointed out, one sees it in real examples. Other than this slight theoretical difference, which merely requires Monte Carlo sample sizes larger in one case than the other, there is no important difference.

13.3.2 MCNR

As in the other case, the derivatives simplify when we take $h^* = f_\theta$ in which case we get

$$\nabla l_n(\theta) = \frac{1}{n} \sum_{i=1}^{n} \nabla \log f_\theta(X_i, y) \tag{44a}$$

and

$$\begin{aligned} \nabla^2 l_n(\theta) = {} & \frac{1}{n} \sum_{i=1}^{n} \nabla^2 f_\theta(X_i, y) \\ & + \frac{1}{n} \sum_{i=1}^{n} \bigl[\nabla \log f_\theta(X_i, y)\bigr] \bigl[\nabla \log f_\theta(X_i, y)\bigr]^T \\ & - \left(\frac{1}{n} \sum_{i=1}^{n} \nabla \log f_\theta(X_i, y) \right) \left(\frac{1}{n} \sum_{i=1}^{n} \nabla \log f_\theta(X_i, y) \right)^T \end{aligned} \tag{44b}$$

All of the comments that applied to MCNR in the preceding section apply here too.

13.3.3 MCSA

In this case the update of θ in MCSA, the competitor (33) in the other case, is

$$\theta_{k+1} = \theta_k + \epsilon_k \nabla \log f_{\theta_k}(X_k, y) \tag{45}$$

Otherwise the cases are very similar.

13.3.4 MCEM

Here, since we could not find an MCMM algorithm for the other case, we have something new. The q function for the EM algorithm applied to our missing data problem is

$$q(\theta \mid \psi) = E_\psi\{\log f_\theta(X, Y) \mid Y = y\}$$

For comparison with MCLA, we change this standard q function to

$$q(\theta \mid \psi) = E_\psi \left\{ \log \frac{f_\theta(X, Y)}{f_\psi(X, Y)} \;\middle|\; Y = y \right\}$$

(since this only changes the additive term by a constant that does not depend on θ this does not change the sequence of iterates produced by the algorithm). If we apply the conditional Jensen inequality to this we get

$$\begin{aligned}
q(\theta \mid \psi) &= E_\psi \left\{ \log \frac{f_\theta(X, Y)}{f_\psi(X, Y)} \;\middle|\; Y = y \right\} \\
&\le \log \left(E_\psi \left\{ \frac{f_\theta(X, Y)}{f_\psi(X, Y)} \;\middle|\; Y = y \right\} \right) \\
&\le \log \left(\int \frac{f_\theta(x, y)}{f_\psi(x, y)} f_\psi(x \mid y) \mu(dx) \right) \\
&\le \log \left(\int \frac{f_\theta(x, y)}{f_\psi(x, y)} \cdot \frac{f_\psi(x, y)}{f_\psi(y)} \mu(dx) \right) \\
&\le \log \left(\frac{1}{f_\psi(y)} \int f_\theta(x, y) \mu(dx) \right) \\
&= l(\theta) - l(\psi)
\end{aligned}$$

And from this minorization condition we get the guarantee that the EM algorithm goes uphill on the log likelihood, because it satisfies (36a) and (36b) with $l(\theta)$ replaced by $l(\theta) - l(\psi)$. Of course, this is non-Monte-Carlo EM.

The natural approximation of the EM algorithm when the q function cannot be calculated in closed form is

$$q_n(\theta \mid \psi) = \frac{1}{n} \sum_{i=1}^{n} \log f_\theta(X_i, \psi)$$

where X_1, X_2, ... form a Markov chain with equilibrium distribution having unnormalized density $f_\psi(\,\cdot\,, y)$, that is, we sample from the conditional distribution of X given Y for the parameter ψ.

The MCEM algorithm produces a sequence of iterates θ_1, θ_2, ... satisfying

$$q_n(\theta_{k+1} | \theta_k) > q_n(\theta_k | \theta_k)$$

and such a sequence is guaranteed to go uphill on what? Not on the true log likelihood because of Monte Carlo error. It is by the conditional Jensen

inequality to go uphill on the MCLA approximation to the log likelihood

$$\begin{aligned} q_n(\theta|\psi) - q_n(\psi|\psi) &= \frac{1}{n} \sum_{i=1}^{n} \log \frac{f_\theta(X_i, \psi)}{f_\psi(X_i, \psi)} \\ &\le \log \left(\frac{1}{n} \sum_{i=1}^{n} \frac{f_\theta(X_i, \psi)}{f_\psi(X_i, \psi)} \right) \\ &= l_n(\theta) \end{aligned}$$

if in the last step we use $f_\psi = h^*$ as the importance sampling distribution for MCLA, in which case $l_n(\psi) = 0$, which is why there is no term like that at the end. Thus we see that MCEM has no virtues unless MCLA gives virtues to it, and for that MCLA must possess the same virtues itself. Moreover, when we compare the expressions on the opposite sides of the inequality, we see that MCEM and MCLA involve exactly the same calculations but just in a different order. In one the log is inside the sum, in the other the log is outside.

We know from the Dennis-Moré theorem (p. 13.2.2 that the EM algorithm, not being asymptotically equivalent to Newton-Raphson is slowly converging. In fact, it is notoriously slowly converging, often taking thousands or millions of iterations to produce a few significant figures (this is non-Monte-Carlo EM). MCEM is slower still, since each of these iterations is costly, requiring and MCMC run.

The good news is that we have no intention of MCEM "converging" to high accuracy, since no Monte Carlo method can produce high accuracy. The bad news is that MCEM is still very efficient. Once it gets close to a solution, it is simply perverse to keep doing MCEM iterations instead of switching to MCLA or MCNR. But MCEM is inefficient in the role of "getting close" compared to MCSA. It is not clear that there is any role in which MCEM makes sense. It is, however, the most popular method for these problems. Such is the influence of widely popularized methods.

13.4 Unknown Normalizing Constant Missing Data Models

Suppose now our model is given by a family of unnormalized densities $h_\theta(x, y)$ with x missing and y observed. The normalizing constants are

$$c(\theta) = \iint h_\theta(x, y) \mu(dx) \nu(dy)$$

where the h_θ are densities w. r. t. the product measure $\mu \times \nu$. The normalized complete data densities are

$$f_\theta(x, y) = \frac{1}{c(\theta)} f_\theta(x, y)$$

The likelihood is

$$\begin{aligned} L(\theta) &= f_\theta(y) \\ &= \frac{1}{c(\theta)} \int h_\theta(x, y) \mu(dx) \end{aligned}$$

Thus the log likelihood is

$$l(\theta) = \log \left(\int h_\theta(x, y) \mu(dx) \right) - \log \left(\iint h_\theta(x, y) \mu(dx) \nu(dy) \right) \tag{46}$$

As always, in order to do the integrals by Monte Carlo, we use the importance sampling formula, writing

$$l(\theta) = \log \left(E^* \left\{ \frac{h_\theta(X, Y)}{h^*(X, Y)} \;\middle|\; Y = y \right\} \right) - \log \left(E^{**} \left\{ \frac{h_\theta(X, Y)}{h^{**}(X, Y)} \right\} \right) \tag{47}$$

where we have introduced two different importance sampling distributions, there being no need for them to be the same. As always we need no divide by zero for (47) to be valid. For once, we will not write out the condition explicitly.

If X_1^*, X_2^*, ... are a Markov chain with equibrium distribution having unnormalized density $h^*(\,\cdot\,, y)$ and If (X_i^{**}, Y_i^{**}), $i = 1, 2, \ldots$ are a Markov chain with equibrium distribution having unnormalized density h^{**}, then

$$l_{m,n}(\theta) = \log \left(\frac{1}{m} \sum_{i=1}^{m} \frac{h_\theta(X_i^*, y)}{h^*(X_i^*, y)} \right) - \log \left(\frac{1}{n} \sum_{i=1}^{n} \frac{h_\theta(X_i^{**}, Y_i^{**})}{h^{**}(X_i^{**}, Y_i^{**})} \right) \tag{48}$$

is the natural Monte Carlo approximation to (47). Note that the two samples being unrelated, there is no reason to have the same Monte Carlo sample size, so we have allowed them to be different.

We shall not laboriously repeat the details of all the issues. MCLA and MCNR are obvious. One only needs calculus to differentiate (48). We forgo the details.

13.4.1 MCSA

Since this involves two samplers not one, it is no longer clear how to do MCSA. It does seem that if one ran a simultaneous sampler with state $(X_i^*, X_i^{**}, Y_i^{**})$ having joint equilibrium unnormalized density

$$h_k(x^*, x^{**}, y^{**}) = f_{\theta_k}(x^*, y) f_{\theta_k}(x^{**}, y^{**}),$$

then

$$\theta_{k+1} = \theta_k + \epsilon_k \big[\nabla h_{\theta_k}(X_k^*, y) - \nabla h_{\theta_k}(X_k^{**}, Y_k^{**}) \big]$$

would be some sort of MCSA step, but (as far as I know) this has never been studied.

13.4.2 MCEM

To do MCEM one just moves the log inside as we did before. The MCEM q function is

$$q_{m,n}(\theta, \psi) = \left(\frac{1}{m} \sum_{i=1}^{m} \log \frac{h_\theta(X_i^*, y)}{h_\psi(X_i^*, y)} \right) - \log \left(\frac{1}{n} \sum_{i=1}^{n} \frac{h_\theta(X_i^{**}, Y_i^{**})}{h^{**}(X_i^{**}, Y_i^{**})} \right) \tag{49}$$

One can go ahead and do MCEM if one has a serious MCEM addiction. Of course, this scheme is an odd combination of MCEM and MCLA. There is no way (as far as I know) to MCEMize the second term. Hence one must use importance sampling anyway. So if one has a serious importance sampling aversion, one must simply avoid these sorts of models.

I will concede that to the extent that EM deserves its reputation as being more stable than other optimization methods and being especially well behaved when started far from the solution (there is much folklore saying this, but no theorem), one may prefer MCEM in this role of "getting close" since MCSA has not been studied. Actually (as far as I know), neither MCSA nor MCEM has ever been used for these sorts of models for the simple reason that these sorts of models have not been used much and have never been extensively investigated.

13.5 Umbrella Sampling

References

Bélisle, C. J. P. (1992). Convergence theorems for a class of simulated annealing algorithms on $\mathbb{R}^d$. *Journal of Applied Probability*, 29:885–895.

Bélisle, C. J. P., Romeijn, H. E., and Smith, R. L. (1993). Hit-and-run algorithms for generating multivariate distributions. *Mathematics of Operations Research*, 18:255–266.

Benveniste, A., Métivier, M., and Priouret, P. (1990). *Adaptive algorithms and stochastic approximations*. Berlin: Springer-Verlag.

Besag, J. (1994). Discussion of Grenander and Miller (1994). *Journal of the Royal Statistical Society, Series B*, 56:591–592.

Besag, J., Green, P., Higdon, D., and Mengersen, K. (1995). Bayesian computation and stochastic systems. *Statistical Science*, 10:3–41. With discussion.

Besag, J. and Green, P. J. (1993). Spatial statistics and Bayesian computation. *Journal of the Royal Statistical Society, Series B*, 55:25–37. With discussion.

Chen, M.-H. and Schmeiser, B. (1993). Performance of the Gibbs, hit-and-run, and Metropolis samplers. *Journal of Computational and Graphical Statistics*, 2:251–272.

Clifford, P. (1993). Discussion of Smith and Roberts (1993), Besag and Green (1993), and Gilks et al. (1993). *Journal of the Royal Statistical Society, Series B*, 55:53–54.

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. With discussion.

Dijkstra, E. W. (1976). *A Dicipline of Programming*. Englewood Cliffs, NJ: Prentice-Hall.

Fletcher, R. (1987). *Practical Methods of Optimization*. John Wiley, 2nd edition.

Gelfand, A. E. and Carlin, B. P. (1993). Maximum-likelihood estimation for constrained- or missing-data models. *Canadian Journal of Statistics*, 21:303–311.

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

Gelman, A., Roberts, G. O., and Gilks, W. R. (1996). Efficient Metropolis jumping rules. In *Bayesian Statistics 5 – Proceedings of the Fifth Valencia International Meeting*, J. M. Bernardo, J. O. Berger, A. P. Dawid, and A. F. M. Smith, editors, pages 599–607. Clarendon Press [Oxford University Press].

Geyer, C. J. (1991). Markov chain Monte Carlo maximum likelihood. In *Computing Science and Statistics: Proc. 23rd Symp. Interface*, E. Keramidas, editor, pages 156–163. Interface Foundation.

Geyer, C. J. (1992). Practical Markov chain Monte Carlo. *Statistical Science*, 7:473–511. With discussion.

Geyer, C. J. (1994). On the convergence of Monte Carlo maximum likelihood calculations. *Journal of the Royal Statistical Society, Series B*, 56:261–274.

Geyer, C. J. (1995). Comment on Besag et al. (1995). *Statistical Science*, 10:46–48.

Geyer, C. J. and Thompson, E. A. (1992). Constrained Monte Carlo maximum likelihood for dependent data. *Journal of the Royal Statistical Society, Series B*, 54:657–699. With discussion.

Geyer, C. J. and Thompson, E. A. (1995). Annealing Markov chain Monte Carlo with applications to ancestral inference. *Journal of the American Statistical Association*, 90:909–920.

Gilks, W. R., Clayton, D. G., Spiegelhalter, D. J., Best, N. G., and McNeil, A. J. (1993). Modelling complexity: Applications of Gibbs sampling in medicine. *Journal of the Royal Statistical Society, Series B*, 55:39–52. With discussion.

Grenander, U. and Miller, M. I. (1994). Representations of knowledge in complex systems. *Journal of the Royal Statistical Society, Series B*, 56:549–603. With discussion.

Guo, S. W. and Thompson, E. A. (1992). Monte Carlo estimation of mixed models for large complex pedigrees. *American Journal of Human Genetics*, 51:1111–1126.

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

Hunter, D. R. and Lange, K. (2004). A tutorial on MM algorithms. *American Statistician*, 58:30–37.

Kindermann, R. and Snell, J. L. (1980). *Markov Random Fields and Their Applications*. Providence, RI: American Mathematical Society.

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

Locatelli, M. (2000). Convergence of a simulated annealing algorithm for continuous global optimization. *Journal of Global Optimization*, 18:219–234.

MacEachern, S. N. and Berliner, L. M. (1994). Subsampling the gibbs sampler. *American Statistician*, 48:188–190.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E. (1953). Equation of state calculations by fast computing machines. *Journal of Chemical Physics*, 21:1087–1092.

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

Mira, A. and Geyer, C. J. (2000). On non-reversible Markov chains. In *Monte Carlo Methods*, N. Madras, editor, pages 93–108. Toronto, Canada: Fields Institute.

Moyeed, R. A. and Baddeley, A. J. (1991). Stochastic approximation of the MLE for a spatial point pattern. *Scandinavian Journal of Statistics*, 18:39–50.

Nocedal, J. and Wright, S. J. (1999). *Numerical Optimization*. Springer-Verlag.

Penttinen, A. (1984). *Modelling Interaction in Spatial Point Patterns: Parameter Estimation by the Maximum Likelihood Method*. Number 7 in Jyväskylä Studies in Computer Science, Economics, and Statistics. University of Jyväskylä.

Potts, R. B. (1952). Some generalized order-disorder transformations. *Proceedings of the Cambridge Philosophical Society*, 48:106–109.

Ripley, B. D. (1987). *Stochastic Simulation*. New York: Wiley.

Roberts, G. O. and Rosenthal, J. S. (1998). Optimal scaling of discrete approximations to Langevin diffusions. *Journal of the Royal Statistical Society, Series B*, 60:255–268.

Roberts, G. O. and Tweedie, R. L. (1996). Exponential convergence of Langevin distributions and their discrete approximations. *Bernoulli*, 2:341–363.

Smith, A. F. M. and Roberts, G. O. (1993). Bayesian computation via the Gibbs sampler and related Markov chain Monte Carlo methods. *Journal of the Royal Statistical Society, Series B*, 55:3–23. With discussion.

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

Thompson, E. A. and Guo, S. W. (1991). Evaluation of likelihood ratios for complex genetic models. *IMA J. Math. Appl. Med. Biol.*, 8:149–169.

Wasan, M. T. (1969). *Stochastic Approximation*. Cambridge University Press.

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

Younes, L. (1988). Estimation and annealing for Gibbsian fields. *Annales de l'Institut Henri Poincaré. Probabilités et Statistiques*, 24:269–294.

Younes, L. (1999). On the convergence of Markovian stochastic algorithms with rapidly decreasing ergodicity rates. *Stochastics and Stochastics Reports*, 65:177–228.