If the goal is to generate many random variables with distribution $\{p_j\}$ so as to estimate

$$(5.1.4) \quad E(h(X)) = \sum_{j=1}^{N} h(j) p_j$$

then we can estimate $E(h(X))$ using

$$(5.1.5) \quad \frac{1}{n} \sum_{i=1}^{n} h(X_i)$$

Since the early states can be strongly affected by the initial condition, we may disregard the first k states and use

$$(5.1.6) \quad \hat{\theta} = \frac{1}{n-k} \sum_{i=k+1}^{n} h(X_i)$$

for k sufficiently large.

$\boxed{\#29 \quad 8/14}$

§5.2 The Hastings-Metropolis Algorithm

Let $\{b(j)\}_{j=1}^{m}$ be positive numbers and $B = \sum_{j=1}^{m} b(j)$. We assume m is large and B is difficult to compute. We want to simulate a random variable (or sequence of r.v.) with p.m.f.

$$\pi(j) = \frac{b(j)}{B} \quad , j = 1, 2 \ldots, m.$$

We find a Markov chain that is easy to simulate and whose limiting distribution is $\{\pi_j\}$.

The Hastings-Metropolis algorithm constructs a time reversible Markov chain with π as the limiting distribution.

Let Q be the transition probability matrix for an irreducible Markov chain with state space $S = \{1, 2 \ldots, m\}$, with entries $q(i,j)$.

We define a Markov chain $\{X_n, n \geq 0\}$ as follows.

When $X_n = i$, a random variable X such that $P(X = j) = q(i,j)$, $j = 1, \ldots, m$ is generated. If $X = j$, then X_{n+1} is set equal to j with probability $\alpha(i,j)$ and set equal to i with probability $1 - \alpha(i,j)$.

It is straightforward to show that the sequence of states constitutes a Markov chain with transition probabilities

$$P_{ij} = q(i,j)\alpha(i,j) \quad , \quad j \neq i$$

$$P_{ii} = q(i,i) + \sum_{k \neq i} q(i,k)\,(1-\alpha(i,k))$$

This Markov chain is reversible and has stationary probabilities that satisfy

$$\pi_i P_{ij} = \pi_j P_{ji} \quad j \neq i.$$

This is

$$\pi_i q(i,j)\alpha(i,j) = \pi_j\, q(j,i)\alpha(j,i).$$

These equations are satisfied if

$$(5.2.1) \quad \alpha(i,j) = \min\left(\frac{\pi_j q(j,i)}{\pi_i q(i,j)}, 1\right) = \min\left(\frac{b_j\, q(j,i)}{b_i\, q(i,j)}, 1\right)$$

Note: if $\alpha(i,j) = \pi_j\, q(j,i)/(\pi_i\, q(i,j))$, then $\alpha(j,i) = 1$ and vice versa).

Note: we do not use B to define the Markov chain.

Also note that $\{\pi_j\}$ is generally the limiting distribution as well as the stationary distribution, e.g. if $P_{ii} > 0$ for some i.

Hasting-Metropolis Algorithm

Generate a time-reversible Markov chain with limiting distribution

$$\pi_j = \frac{b(j)}{B}, \; j=1,..,m, \quad B = \sum_{j=1}^{m} b(j)$$

1. Choose an irreducible Markov chain on $\{1,..,m\}$ with transition probability matrix

$$Q = (q(i,j))$$

Choose $k \in \{1,2,..,m\}$

2. Let $n=0$ and $X_0 = k$

3. Generate a r.v. X with $P(X=j) = q(X_n, j)$ and a uniform random number U in $[0,1]$.

4) If $U < \dfrac{b(X)\,q(X,X_n)}{b(X_n)\,q(X_n,X)}$, then $NS = X$, else $NS = X_n$

5) $n = n+1$, $X_n = NS$

6) Go to (3).

Example 5.2.1

Suppose we want to generate a random element from a set $\mathcal{S}$ of all permutations $(x_1, \ldots, x_n)$ of numbers $\{1, 2, \ldots, n\}$ for which

$$\sum_{j=1}^{n} j x_j > a,$$

for a fixed a.

We define two such permutations to be neighbors if one results from the other

by an interchange of two of the positions of the other. So $(1,2,3,4)$ and $(1,2,4,3)$ are neighbors, but $(1,2,3,4)$ and $(1,3,4,2)$ are not.

We define the q transition probability function as follows. With $N(s)$ defined as the set of neighbors of s and $|N(s)|$ equal to the number of elements in $N(s)$, set

$$q(s,t) = \begin{cases} \frac{1}{|N(s)|}, & t \in N(s), \\ 0, & \text{otherwise} \end{cases}$$

Thus, the target next state from s is equally likely to be any of its neighbors.

Since the desired limiting probabilities of the Markov chain are uniform and constant,

$$\pi(s) = \pi(t) = C,$$

for some constant C. Therefore,

$$\alpha(s,t) = \min\left(\frac{|N(s)|}{|N(t)|}, 1\right).$$

In other words, if the present state of the Markov chain is s, then one of its neighbors is randomly chosen - say t. If t is a state with fewer neighbors than s then t is the next state.

If t does not have fewer neighbors, a random number U is generated, and the next state is t if

$$U < \frac{|N(s)|}{|N(t)|}$$

and s otherwise.

The limiting probabilities are $\pi(s) = \frac{1}{|\mathcal{S}|}$.

§5.3 The Gibbs Sampler

The Gibbs Sampler is the most widely used version of the Hastings-Metropolis algorithm.

(!) important for application to Bayesian statistics

Let $\vec{X} = (X_1, .., X_n)$ be a random vector with p.m.f. $p(\vec{x})$, that need only be specified up to a multiplicative constant, and suppose we want to generate a random vector having p.m.f.

$$p(\vec{x}) = C\, g(\vec{x})$$

where g is known but C is not.

We assume that for any i and values x_j, $j \neq i$,

we can generate a random variable X having p.m.f.

(5.3.1) $$P(X = x) = P(X_i = x \mid X_j = x_j,\ j \neq i)$$

(we are assuming these conditional probabilities are easier to generate)

We use the Hastings-Metropolis algorithm on a Markov chain with states $\vec{x} = (x_1, .., x_n)$ and transition probabilities defined as follows:

whenever the present state is $\vec{x}$, a coordinate that is equally likely to be any of $1, 2, .., n$ is chosen. If i is chosen, then a random variable X whose p.m.f. is given by (5.3.1) is generated and if $X = x$, then the state

$$\vec{Y} = (x_1, x_2, \ldots, x_{i-1}, x, x_{i+1}, \ldots, x_n)$$

is considered as the candidate next state. In other words, with $\vec{X}$ and $\vec{Y}$ given, the Gibbs sampler uses the Hastings-Metropolis algorithm with

$$q(\vec{X}, \vec{Y}) = \frac{1}{n} P(X_i = x \mid X_j = x_j, j \neq i)$$

$$= \frac{P(\vec{Y})}{n P(X_j = x_j, j \neq i)}$$

Since the goal is the limiting mass function P, (5.2.1) implies that the vector $\vec{Y}$ is accepted as the new state with probability

$$\alpha(\vec{X}, \vec{Y}) = \min\left(\frac{P(\vec{Y}) q(\vec{Y}, \vec{X})}{P(\vec{X}) q(\vec{X}, \vec{Y})}, 1\right)$$

$$= \min\left(\frac{P(\vec{Y}) P(\vec{X})}{P(\vec{X}) P(\vec{Y})}, 1\right) = 1.$$

So the candidate state is always accepted.

Example 5.3.1

We want to generate n random points in the unit circle conditional on the event that no two points are within a distance d of each other, where

$$\beta = P(\text{no two points are within } d \text{ of each other}).$$

is assumed to be a small positive number.

We could generate such a set by choosing sets at random until one is accepted.

We can also use a Gibbs sampler. We start with n points, $x_1, \ldots, x_n$, in the circle such that no two are within a distance d of each other.

We generate a random number U and let

I = integer part of $(nU) + 1$. We also generate 1 random point in the unit circle. If this point is not within d of any other of the $n-1$ points, excluding X_I, then replace X_I by this generated point, otherwise generate a new point and repeat the operation. After a large number of iterations, the set of n points will have approximately the desired distribution.

Example 5.3.2

Let X_i, $i=1,\dots,n$, be independent random variables with X_i having exponential distribution with parameter λ_i, $i=1,\dots,n$.

Let $S = \sum_{i=1}^{n} X_i$ and suppose we want to generate the random vector $\vec{X} = (X_1,\dots,X_n)$ conditional on the event that $S > c$ for some large positive constant $c > 0$.

In other words, we want to generate the value of a random vector whose density function is

$$f(x_1, x_2, \dots, x_n) = \frac{1}{P(S>c)} \prod_{i=1}^{n} \lambda_i e^{-\lambda_i x_i}, \quad \sum_{i=1}^{n} x_i > c$$

We start with an initial vector $\vec{x} = (x_1,\dots,x_n)$ satisfying $x_i > 0$, $i=1,\dots,n$ and
$$\sum_{i=1}^{n} x_i > c.$$

We generate a random number U and set $I =$ integer part of $(nU+1)$. Suppose $I = i$.

Now, we want to generate and exponential random variable X with rate λ_i conditioned on the event that $X + \sum_{j \neq i} X_j > c$, i.e., generate X conditional on the event it is greater than $c - \sum_{j \neq i} X_j$.

Using the fact that an exponential conditioned to be greater than a positive constant is distributed as the constant plus the exponential, we should generate an exponential random variable Y with rate λ_i ($Y = -\frac{1}{\lambda_i} \log U$, $U \sim$ uniform) and set

$$X = Y + \left(c - \sum_{j \neq i} X_j\right)^+$$

where

$$b^+ = \begin{cases} b, & b > 0, \\ 0, & b \leq 0. \end{cases}$$

The value of X_i is set to X, and a new iteration begins.

§5.4 Simulated Annealing

We consider the following problem

Let A be a finite set of vectors and $V(\vec{x})$ a nonnegative function for $\vec{x} \in A$. We are interested in finding the maximal value of V over A and at least one vector where the maximal value is obtained. In other words, with

$$V^* = \max_{\vec{x} \in A} V(\vec{x})$$

and

$$\mathcal{M} = \{ \vec{x} \in A \;\; V(\vec{x}) = V^* \}$$

we want to find V^* and at least one element in $\mathcal{M}$.

We let $\lambda > 0$ and consider the pmf on vectors in A:

$$p_\lambda(\vec{x}) = \frac{e^{\lambda V(\vec{x})}}{\sum_{\vec{x} \in A} e^{\lambda V(\vec{x})}}$$

Multiplying by $e^{-\lambda V^*}/e^{-\lambda V^*}$ and using $|\mathcal{M}|$ = number of elements in $\mathcal{M}$, we find

$$P_\lambda(\vec{x}) = \frac{e^{\lambda(V(\vec{x})-V^*)}}{|\mathcal{M}| + \sum_{\vec{x}\notin\mathcal{M}} e^{\lambda(V(\vec{x})-V^*)}}$$

Since $V(\vec{x})-V^* < 0$ for $X \notin \mathcal{M}$, as $\lambda \to \infty$

$$P_\lambda(\vec{x}) \longrightarrow \frac{\delta(\vec{x},\mathcal{M})}{|\mathcal{M}|},$$

where $\delta(\vec{x},\mathcal{M}) = \begin{cases} 1, & x \in \mathcal{M}, \\ 0, & x \notin \mathcal{M}. \end{cases}$

So, if we let λ be large and generate a Markov chain whose limiting distribution is $P_\lambda(\vec{x})$, then most of the "mass" of this limiting distribution will be concentrated at points in $\mathcal{M}$.

We introduce the idea of neighboring vectors and then use a Hastings-Metropolis algorithm.

We say two vectors $\vec{x}, \vec{y} \in A$ are neighbors if they differ in only a single coordinate or if one can be obtained from the other by an interchange of two components.

We let the target next state from $\vec{x}$ to be equally likely to be any of its neighbors. If neighbor $\vec{y}$ is chosen, the next state becomes y with probability

$$\min\left\{1, \frac{e^{\lambda V(\vec{y})}/|N(\vec{y})|}{e^{\lambda V(\vec{x})}/|N(\vec{x})|}\right\}$$

or remains $\vec{x}$ otherwise, where $|N(\vec{z})|$ is the number of neighbors of $\vec{z}$.

If each vector has the same number of neighbors, which can usually be arranged by increasing the state space and setting $V=0$ on the additional states, then when the state is $\vec{x}$, one of its neighbors, $\vec{y}$, is randomly chosen; if $V(\vec{y}) \geq V(\vec{x})$, then the chain moves to $\vec{y}$ and if $V(\vec{y}) < V(\vec{x})$, the chain moves to state $\vec{y}$ with probability $\exp(\lambda(V(\vec{y}) - V(\vec{x})))$ or remains in $\vec{x}$ otherwise.

One weakness is that because λ is large, if the chain enters a state $\vec{x}$ whose V value is larger than all of its neighbors, then it might take a long time for the chain to move to a different state. We need the large value of λ for the limiting distribution to put most of its weight on points in $\mathcal{M}$, but such a value typically requires a very large number of transitions before the limiting distribution is approached.

Now since A is finite, the concept of convergence is somewhat strained. So this algorithm is usually viewed heuristically, and the λ's are allowed to vary.

Simulated annealing is a variation. If the n^{th} state of the Markov chain is $\vec{x}$, then a neighboring value is randomly selected. If it is $\vec{y}$, then the next state is either $\vec{y}$ with probability

$$\min\left\{1, \frac{\exp\{\lambda_n V(\vec{y})\}/|N(\vec{y})|}{\exp\{\lambda_n V(\vec{x})\}/|N(\vec{x})|}\right\}$$

or it remains $\vec{x}$, where λ_n, $n \geq 1$, is in a prescribed set of values that start out small (resulting in a large number of changes of state), and then grow.
For example, a useful choice is

$$\lambda_n = C\log(1+n), \quad C>0 \text{ fixed.}$$