Math 5040
Markov chain Monte Carlo methods

S. Ethier

References:
Let $h$ be a positive function on a large finite set $S$. How do we simulate an $S$-valued random variable $X$ with distribution

$$\pi(x) = P(X = x) = \frac{h(x)}{\sum_{y \in S} h(y)}.$$  \hfill (1)

Typically, we do not know the value of the sum in the denominator. For example, $h$ could be identically 1, but we might not know or be able to count the number of elements that are in $S$.

The idea of MCMC is to find a Markov chain in $S$ with stationary distribution $\pi$. Simulate the Markov chain for sufficiently many steps for it to be approximately in equilibrium. The proportion of time spent in state $i$ will be approximately $\pi(i)$. 
Example

Let $S$ be the set of all $25 \times 25$ matrices of 0s and 1s with no adjacent 1s in the same row or column. Let $M$ be uniformly distributed over $S$. We want to simulate the expected proportions of 1s in $M$. The first problem is to construct the Markov chain.

Consider the following one-step transition. Starting at $M \in S$, choose an entry $(i,j)$ at random from $\{1, 2, \ldots, 25\}^2$. If $M(i,j) = 1$, then change the entry at $(i,j)$ to 0. If $M(i,j) = 0$, then change the entry at $(i,j)$ to 1 if the resulting matrix belongs to $S$; otherwise leave it unchanged. Then $P(M_1, M_2) = 1/(25)^2$ if $M_1$ and $M_2$ are elements of $S$ that differ in exactly one entry, and $P(M_1, M_1) = j/(25)^2$, where $j$ is the number of zero entries of $M_1$ that cannot be changed to a 1.

This MC is irreducible and aperiodic with $P$ symmetric, hence doubly stochastic. So the uniform distribution is stationary.
In[89]:=  
K = 25;
M = ConstantArray[0, {K, K}];
steps = 1000000;
ones = 0;
sum = 0;
For[n = 1, n ≤ steps, n++,
    i = Floor[K * RandomReal[]] + 1;
    j = Floor[K * RandomReal[]] + 1;
    If[M[[i, j]] == 1, M[[i, j]] = 0; ones--,
        If[(i = 1 || M[[Max[i - 1, 1], j]] == 0) &&
            (i = K || M[[Min[i + 1, K], j]] == 0) &&
            (j = 1 || M[[i, Max[j - 1, 1]]] == 0) &&
            (j = K || M[[i, Min[j + 1, K]]] == 0),
            M[[i, j]] = 1; ones++]];
    sum += ones];
Print[N[sum / (steps * K * K)]];
Remark. Instead of sampling the Markov chain at every step, we might sample it every 2500 steps, say, and then the observations would be closer to independent.

How would we estimate the size of the set of 25 × 25 matrices of 0s and 1s with no adjacent 1s in the same row or column? This is a tricky problem, but see Lawler and Coyle’s Lectures on Contemporary Probability (1999) for a suggested solution. It is known that the number $|S_N|$ of $N \times N$ matrices of 0s and 1s with no adjacent 1s in the same row or column satisfies

$$\beta = \lim_{N \to \infty} |S_N|^{1/N^2},$$

where $1.50304 \leq \beta \leq 1.50351$, or roughly $|S_N| \approx \beta^{N^2}$. 
Reversibility

The method works well when the function $h$ is identically 1. But in the general case we need the concept of *reversibility*. Consider a M.C. $X_0, X_1, X_2, \ldots$ with transition matrix $P$ and stationary initial distribution $\pi$, hence it is a stationary process. Now reverse the process: $X_n, X_{n-1}, X_{n-2}, \ldots$. It is Markovian with transition probabilities

$$Q(i, j) = P(X_n = j \mid X_{n+1} = i) = \frac{P(X_n = j)P(X_{n+1} = i \mid X_n = j)}{P(X_{n+1} = i)}$$

$$= \frac{\pi(j)P(j, i)}{\pi(i)}.$$ 

If $Q(i, j) = P(i, j)$ for all $i, j$, or equivalently,

$$\pi(i)P(i, j) = \pi(j)P(j, i) \quad \text{for all } i, j. \quad \text{(2)}$$

then the M.C is said to be *reversible*. 
The equations

\[ \pi(i)P(i, j) = \pi(j)P(j, i) \quad \text{for all } i, j. \]

are called the *detailed balance equations.*

Given a transition matrix \( P \) with a unique stationary distribution \( \pi \), if we can find \( x(i) \) such that

\[ x(i)P(i, j) = x(j)P(j, i) \quad \text{for all } i, j, \]

and \( \sum_i x(i) = 1 \), then summing over \( i \) we get

\[ x(j) = \sum_i x(i)P(i, j), \]

so we get \( x(i) = \pi(i) \) by uniqueness of stationary distributions.
Example

Recall the random walk on a connected finite graph. There are weights \( w_{ij} \) associated with the edge from \( i \) to \( j \). If we define

\[
P(i, j) = \frac{w_{ij}}{\sum_k w_{ik}},
\]

where the sum extends over all neighbors of vertex \( i \), we have an irreducible finite M.C., so there is a unique stationary distribution. What is it? Check detailed balance equations. Use \( w_{ij} = w_{ji} \) to derive

\[
\pi(i) = \frac{\sum_j w_{ij}}{\sum_i \sum_j w_{ij}}.
\]
How do we find a Markov chain for general $h$? Here is the Hastings–Metropolis algorithm. Let $Q$ be any irreducible transition matrix on $S$ and define $P$ to be the transition matrix

$$
P(i, j) = \begin{cases} 
Q(i, j)\alpha(i, j) & \text{if } i \neq j, \\
Q(i, i) + \sum_{k:k\neq i} Q(i, k)(1 - \alpha(i, k)) & \text{if } i = j, 
\end{cases}$$

(3)

where

$$\alpha(i, j) = \min \left( \frac{h(j)Q(j, i)}{h(i)Q(i, j)}, 1 \right).$$

(4)

Interpretation: Instead of making a transition from $i$ to $j$ with probability $Q(i, j)$, we “censor” that transition with probability $1 - \alpha(i, j)$. That is, with probability $\alpha(i, j)$ the proposed transition from $i$ to $j$ is accepted, and with probability $1 - \alpha(i, j)$ the proposed transition from $i$ to $j$ is rejected and the chain stays at $i$. 
If \( \alpha(i, j) = h(j)Q(j, i)/[h(i)Q(i, j)] \), then \( \alpha(j, i) = 1 \) and

\[
h(i)P(i, j) = h(i)Q(i, j)\alpha(i, j) = h(i)Q(i, j)h(j)Q(j, i)/[h(i)Q(i, j)] \\
= h(j)Q(j, i)\alpha(j, i) = h(j)P(j, i) \tag{5}
\]

Similarly if \( \alpha(i, j) = 1 \). So detailed balance eqs. hold for \( P \). In particular, the unique stationary distribution is proportional to the function \( h \).
Example

Let $S$ be the set of all $K \times K$ matrices of 0s and 1s. (Physicists often take matrices of $-1$s and 1s instead to indicate spin orientation.) Let $g$ be a symmetric function on $\{0, 1\}^2$ (i.e., $g(0, 1) = g(1, 0)$). Let $\beta$ be a positive parameter. Define the energy of the matrix $M$ to be

$$E = \sum_{(i,j) \sim (i',j')} g(M(i,j), M(i',j')),$$

where $(i,j) \sim (i',j')$ means that $(i,j)$ and $(i',j')$ are nearest neighbors, that is, $|i - i'| + |j - j'| = 1$. We assume that the probability distribution of interest on $S$ gives equal weight to all matrices with the same energy, so we let

$$h(M) = e^{-\beta E}.$$
The previous example is the limiting case with $g(0, 0) = g(0, 1) = g(1, 0) = 0$, $g(1, 1) = 1$, and $\beta = \infty$. Another important example is the Ising model, with $g(0, 0) = g(1, 1) = 0$ and $g(0, 1) = 1$. (Configurations—matrices—with many pairs of neighboring spins pointing in opposite directions have high energy.)

Starting with $M_1$, choose an entry at random from the $K^2$ possibilities and let $M_2$ be the matrix obtained by changing the chosen entry. If $h(M_2)/h(M_1) \geq 1$, then return matrix $M_2$. If $h(M_2)/h(M_1) = q < 1$, then with probability $q$ return the matrix $M_2$ and with probability $1 - q$ return the matrix $M_1$. This is precisely the Hastings–Metropolis algorithm.
Gibbs sampler

There is a widely used version of the Hastings–Metropolis algorithm called the Gibbs sampler. Here we want to simulate the distribution of an \( n \)-dimensional random vector \( \mathbf{X} = (X_1, X_2, \ldots, X_n) \). Again, its probability density \( \pi(\mathbf{x}) \) is specified only up to a constant multiple. We assume that we can simulate a random variable \( X \) with density equal to the conditional density of \( X_i \), given the values of \( X_j \) for all \( j \neq i \).

Start from \( \mathbf{x} = (x_1, \ldots, x_n) \). Choose a coordinate \( i \) at random. Then choose a coordinate \( x \) according to the conditional density of \( X_i \), given that \( X_j = x_j \) for all \( j \neq i \). Consider the state \( \mathbf{y} = (x_1, \ldots, x_{i-1}, x, x_{i+1}, \ldots, x_n) \) as a candidate for a transition. Apply Hastings–Metropolis with \ldots
\[ Q(x, y) = \frac{1}{n} P(X_i = x \mid X_j = x_j \text{ for all } j \neq i) \]

\[ = \frac{\pi(y)}{nP(X_j = x_j \text{ for all } j \neq i)} . \]

The acceptance probability is

\[ \alpha(x, y) = \min \left( \frac{\pi(y) Q(y, x)}{\pi(x) Q(x, y)}, 1 \right) \]

\[ = \min \left( \frac{\pi(y) \pi(x)}{\pi(x) \pi(y)}, 1 \right) \]

\[ = 1. \]

Hence the candidate star is always accepted.
The method also applies when the random variables are continuous.

**Example**

Suppose we want to simulate \( n \) points in the unit disk at least \( d \) apart. Take \( n = 35 \) and \( d = 1/4 \).

We can use the Gibbs sampler. Initialize as follows.
At each step, choose one of the $n$ points at random. Then simulate a random point in the unit disk (rejection method) until we get one that is a distance at least $d$ from each of the other points. Replace the $i$th point with the new one.

```math
n = 35; d = 1/4;
coords = ConstantArray[0, {n, 2}];
coords = {{-.75, 0.}, {-.5, 0.}, {-.25, 0.}, {0., 0.},
    {.25, 0.}, {.5, 0.}, {.75, 0.}, {-.75, .25}, {-.5, .25},
    {-.25, .25}, {0., .25}, {.25, .25}, {-.5, .25}, {.75, .25},
    {-.75, -.25}, {-.5, -.25}, {-.25, -.25}, {0., -.25},
    {.25, -.25}, {.5, -.25}, {.75, -.25}, {-.75, .5},
    {-.5, .5}, {-.25, .5}, {0., .5}, {.25, .5}, {.5, .5},
    {.75, .5}, {-.75, -.5}, {-.5, -.5}, {-.25, -.5},
    {0., -.5}, {.25, -.5}, {.5, -.5}, {-.75, -.5}};
Show[ContourPlot[x^2 + y^2 == 1, {x, -1, 1}, {y, -1, 1}],
  ListPlot[Table[coords, {i, 1, n}]]]
For[run = 1, run \leq 1000, run++,
  m = IntegerPart[n RandomReal[]] + 1; For[try = 1, try \leq 1000, try++,
    x = 2 RandomReal[] - 1; y = 2 RandomReal[] - 1;
    If[x^2 + y^2 < 1, flag = 1; For[i = 1, i \leq n, i++,
        If[Sqrt[(coords[[i, 1]] - x)^2 + (coords[[i, 2]] - y)^2] \leq d
          && i \neq m, flag = 0]];
    If[flag == 1, coords[[m, 1]] = x; coords[[m, 2]] = y;
      try = 1001]]];
Show[ContourPlot[x^2 + y^2 == 1, {x, -1, 1}, {y, -1, 1}],
  ListPlot[Table[coords, {i, 1, n}]]]
```
Output:
Example

Self-avoiding walks (SAWs). (From Lawler–Coyle book.) A SAW of length \( n \) in \( \mathbb{Z}^d \) is a sequence of paths 
\[ \omega = (\omega_0, \omega_1, \ldots, \omega_n) \] with \( \omega_0 = 0 \), \( \omega_i \in \mathbb{Z}^d \), \( |\omega_i - \omega_{i-1}| = 1 \), and \( \omega_i \neq \omega_j \) whenever \( i \neq j \). SAWs are models for polymer chains in chemistry.

Let \( \Omega_n \) be the set of all SAWs of length \( n \). What is \(|\Omega_n|? Notice that 
\[ d^n \leq |\Omega_n| \leq 2d(2d - 1)^{n-1}. \] (8)

In fact, since 
\[ |\Omega_{n+m}| \leq |\Omega_n| |\Omega_m|, \] (9)

it can be shown that

\[ \lim_{n \to \infty} |\Omega_n|^{1/n} = \beta, \] (10)

where \( d \leq \beta \leq 2d - 1 \).
There are a number of interesting questions that one can ask about SAWs. For example, what is the asymptotic probability that two independent $n$-step SAWs when combined end to end form a $2n$-step SAW? For another example, how large are SAWs, that is, how does $E[|\omega_n|^2]$ behave for large $n$?

These questions are too hard even for specialists to answer. But perhaps we can get some sense of what is going on by simulating a random SAW. Notice that the obvious rejection method is extremely inefficient if $n$ is large.

*Let us assume from now on that $d = 2$. We use MCMC. We want to simulate the uniform distribution on $\Omega_n$ for some specified $n$. Our approach is based on what is called the pivot algorithm.*
Let $O$ denote the set of orthogonal transformations of the plane that map $\mathbb{Z}^2$ onto $\mathbb{Z}^2$. These include the rotations by $\pi/2$, $\pi$, and $3\pi/2$, and the reflections about the coordinate axes and about the lines $y = x$ and $y = -x$. (We exclude the identity transformation from $O$, so $|O| = 7$.) Consider the following MC in $\Omega_n$: Starting with $\omega = (\omega_0, \omega_1, \ldots, \omega_n)$, choose a number at random from $\{0, 1, \ldots, n-1\}$ and call it $k$. Choose a transformation $T$ at random from $O$. Consider the walk obtained by fixing the first $k$ steps of the walk but performing the transformation $T$ on the remaining part of the walk, using $\omega_k$ as the origin for the transformation. This gives us a new path which may or may not be a SAW. If it is, return it as the new path; if it isn’t, return the original path. This gives us a MC on $\Omega_n$. It is irreducible and aperiodic with a symmetric $p$. So the limiting distribution is uniform.
L = 100;
saw = ConstantArray[0, (L + 1, 2)];
saw1 = ConstantArray[0, (L + 1, 2)];
saw2 = ConstantArray[0, (L + 1, 2)];
saw3 = ConstantArray[0, (L + 1, 2)];

For[n = 0, n ≤ L, n++,
    saw[[n + 1, 1]] = n; saw[[n + 1, 2]] = 0;
(* initial saw is a straight line to the right *)

For[step = 1, step ≤ 2000, step++,
    k = Floor[L RandomReal[]]; m = Floor[7 RandomReal[]] + 1;
    For[n = 0, n ≤ k, n++,
        saw1[[n + 1, 1]] = saw[[n + 1, 1]]; saw1[[n + 1, 2]] = saw[[n + 1, 2]];
        (* first k steps of saw are unchanged in saw1 *)
        For[n = k, n ≤ L, n++,
            saw2[[n - k + 1, 1]] = saw[[n + 1, 1]] - saw[[k + 1, 1]]; saw2[[n - k + 1, 2]] = saw[[n + 1, 2]] - saw[[k + 1, 2]]; saw3[[j + 1, 1]] = -saw2[[j + 1, 1]]; saw3[[j + 1, 2]] = saw2[[j + 1, 1]]]
        (* remaining steps of saw are saved, after shifting origin, in saw2 *)
        (* Now we transform saw2 *)
        If[m = 1, For[j = 0, j ≤ L - k, j++,
            saw3[[j + 1, 1]] = -saw2[[j + 1, 2]]; saw3[[j + 1, 2]] = saw2[[j + 1, 1]]];
        (* rotation of π/2 *)
        If[m = 2, For[j = 0, j ≤ L - k, j++,
            saw3[[j + 1, 1]] = -saw2[[j + 1, 2]]; saw3[[j + 1, 2]] = -saw2[[j + 1, 2]]];
        (* rotation of π *)
        If[m = 3, For[j = 0, j ≤ L - k, j++,
            saw3[[j + 1, 1]] = saw2[[j + 1, 2]]; saw3[[j + 1, 2]] = -saw2[[j + 1, 1]]];
        (* rotation of 3π/2 *)
        If[m = 4, For[j = 0, j ≤ L - k, j++,
            saw3[[j + 1, 1]] = saw2[[j + 1, 1]]; saw3[[j + 1, 2]] = -saw2[[j + 1, 2]]];
        (* reflection about x-axis *)
        If[m = 5, For[j = 0, j ≤ L - k, j++,
            saw3[[j + 1, 1]] = -saw2[[j + 1, 1]]; saw3[[j + 1, 2]] = saw2[[j + 1, 2]]];
        (* reflection about y-axis *)
        If[m = 6, For[j = 0, j ≤ L - k, j++,
            saw3[[j + 1, 1]] = saw2[[j + 1, 2]]; saw3[[j + 1, 2]] = saw2[[j + 1, 1]]];
        (* reflection about y=x *)
        If[m = 7, For[j = 0, j ≤ L - k, j++,
            saw3[[j + 1, 1]] = -saw2[[j + 1, 2]]; saw3[[j + 1, 2]] = saw2[[j + 1, 1]]];
        (* reflection about y=-x *)
        For[n = k, n ≤ L, n++,
            saw1[[n + 1, 1]] = saw2[[k + 1, 1]] + saw3[[n - k + 1, 1]]; saw1[[n + 1, 2]] = saw2[[k + 1, 2]] + saw3[[n - k + 1, 2]]];
        (* insert transformed saw segment *)
        flag = 1; For[nn = n + 1, nn ≤ L, nn++, For[nn = n + 1, nn ≤ L, nn++,
            If[saw1[[n + 1, 1]] = saw1[[nn + 1, 1]] && saw1[[n + 1, 2]] = saw1[[nn + 1, 2]],
                flag = 0]]; (* check whether saw1 is a saw *)
        If[flag = 1, For[n = 0, n ≤ L, n++,
            saw[[n + 1, 1]] = saw1[[n + 1, 1]]; saw[[n + 1, 2]] = saw1[[n + 1, 2]]];
        (* if it is, return saw1, otherwise do nothing *)
    
    Print[saw]
    ListPlot[saw, Joined → True, AspectRatio → 1]
A substitution code uses a map $f: \{\text{code space}\} \mapsto \{\text{usual alphabet, space, comma, period, digits, etc.}\}$, which is unknown. First step is to download a standard e-text and count the frequencies of the various one-step transitions, thereby getting an estimate of the one-step transition matrix $M$ between consecutive letters. (This is not the MC that we will simulate.) Then we define the plausibility of a particular $f$ by

$$h(f) = \prod_i M(f(s_i), f(s_{i+1})).$$

where $s_1, s_2, \ldots$ is the coded message. Ideally, we would like to find the $f$ that maximizes this function. Instead we will apply MCMC to simulate from the distribution $\pi(f) := h(f)/\sum_g h(g)$. 

Now suppose our symbol space has \( m \) elements and our alphabet space has \( n \geq m \) elements. Then let \( S \) be the set of one-to-one functions \( f \), so \( |S| = (n)_m \), which is large if \( m = n = 40 \), say. (We ignore distinction between upper case and lower case letters.)

What will our MC in \( S \) be? Let \( P(f, f^*) \) be determined by requiring that a transition from \( f \) correspond to a random switch of two symbols. The \( P \) is symmetric, and we apply the Hastings–Metropolis algorithm to get \( Q \).

Start with a preliminary guess, \( f \). Change to \( f^* \) by a random transposition of the values \( f \) assigns to two symbols. If \( h(f^*)/h(f) \geq 1 \), accept \( f^* \). If \( q := h(f^*)/h(f) < 1 \), accept \( f^* \) with probability \( q \). Otherwise stay with \( f \).
where $s_i$ runs over consecutive symbols in the coded message. Functions $f$ which have high values of $P_l(f)$ are good candidates for decryption. Maximizing $f$'s were searched for by running the following Markov chain Monte Carlo algorithm:

1. Start with a preliminary guess, say $f$.
2. Compute $P_l(f)$.
3. Change to $f^\ast$ by making a random transposition of the values $f$ assigns to two symbols.
4. Compute $P_l(f^\ast)$; if this is larger than $P_l(f)$, accept $f^\ast$.
5. If not, flip a $P_l(f^\ast) / P_l(f)$ coin; if it comes up heads, accept $f^\ast$.
6. If the coin toss comes up tails, stay at $f$.

The algorithm continues, trying to improve the current $f$ by making random transpositions. The coin tosses allow it to go to less plausible $f$'s, and keep it from getting stuck in local maxima.

Of course, the space of $f$'s is huge (40! or so). Why should this Metropolis random walk succeed? To investigate this, Marc tried the algorithm out on a problem to which he knew the answer. Figure 2 shows a well-known section of Shakespeare's *Hamlet*.

The text was scrambled at random and the Monte Carlo algorithm was run. Figure 3 shows sample output.

After 100 steps, the message is a mess. After two thousand steps, the decrypted message makes sense. It stays essentially the same as further steps are tried. I find it remarkable that a few thousand steps of this simple optimization procedure work so well. Over the past few years, friends in math and computer science...
where $s_i$ runs over consecutive symbols in the coded message. Functions $f$ which have high values of $\text{Pl}(f)$ are good candidates for decryption. Maximizing $f$'s were searched for by running the following Markov chain Monte Carlo algorithm:

- Start with a preliminary guess, say $f$.
- Compute $\text{Pl}(f)$.
- Change to $f^\star$ by making a random transposition of the values $f$ assigns to two symbols.
- Compute $\text{Pl}(f^\star)$; if this is larger than $\text{Pl}(f)$, accept $f^\star$.
- If not, flip a $\text{Pl}(f^\star)/\text{Pl}(f)$ coin; if it comes up heads, accept $f^\star$.
- If the coin toss comes up tails, stay at $f$.

The algorithm continues, trying to improve the current $f$ by making random transpositions. The coin tosses allow it to go to less plausible $f$'s, and keep it from getting stuck in local maxima.

Of course, the space of $f$'s is huge (40! or so). Why should this Metropolis random walk succeed? To investigate this, Marc tried the algorithm out on a problem to which he knew the answer. Figure 2 shows a well-known section of Shakespeare's Hamlet.

The text was scrambled at random and the Monte Carlo algorithm was run. Figure 3 shows sample output.

After 100 steps, the message is a mess. After two thousand steps, the decrypted message makes sense. It stays essentially the same as further steps are tried. I find it remarkable that a few thousand steps of this simple optimization procedure work so well. Over the past few years, friends in math and computer science...
Example 1 (Cryptography). Stanford’s Statistics Department has a drop-in consulting service. One day, a psychologist from the state prison system showed up with a collection of coded messages. Figure 1 shows part of a typical example.
Example 1 (Cryptography). Stanford’s Statistics Department has a drop-in consulting service. One day, a psychologist from the state prison system showed up with a collection of coded messages. Figure 1 shows part of a typical example.

---

to bat-rb. con todo mi respeto. i was sitting down playing chess with danny de emf and boxer de el centro was sitting next to us. boxer was making loud and loud voices so i tell him por favor can you kick back homie cause im playing chess a minute later the vato starts back up again so this time i tell him con respecto homie can you kick back. the vato stop for a minute and he starts up again so i tell him check this out shut the f**k up cause im tired of your voice and if you got a problem with it we can go to celda and handle it. i really felt disrespected thats why i told him. anyways after i tell him that the next thing I know that vato slashes me and leaves. dy the time i figure im hit i try to get away but the c.o. is walking in my direction and he gets me right dy a celda. so i go to the hole. when im in the hole my home boys hit doxer so now "b" is also in the hole. while im in the hole im getting schoold wrong and