Math 4740: The Metropolis algorithm

Before looking at this, read the first three pages of "The Markov Chain Monte Carlo Revolution" by Persi Diaconis.

Page 2 of that paper describes a Markov chain. Where did it come from? The idea is this. We are given the following ingredients:

- A very large state space, namely the space $\mathcal{X}$ of all possible correspondences f between the code space and the usual alphabet.
- A "plausibility score" for each possible correspondence, denoted $\mathrm{Pl}(f)$.
- A natural Markov chain on $\mathcal{X}$, which we call the "proposal chain." This is the chain that at each time step chooses two symbols uniformly at random and swaps the letters to which they correspond.

We define a probability distribution π on $\mathcal{X}$ by setting $\pi(f) = \mathrm{Pl}(f)/Z$ where Z is the required normalizing factor, namely $Z = \sum_{g \in \mathcal{X}} \mathrm{Pl}(g)$. Importantly, it is infeasible actually to compute Z since the state space is so large. However, for any particular f it is easy to compute $\mathrm{Pl}(f)$.

The Metropolis algorithm uses the proposal chain to create a new reversible Markov chain that has π as its stationary distribution. Before going into the details, consider what this means for the prisoners' code. We expect that the true correspondence $f_0 \in \mathcal{X}$ will have a much higher plausibility score than any of the alternatives. If we're lucky, $\pi(f_0)$ will be very close to 1 while $\sum_{f \neq f_0} \pi(f)$ is near 0.

Let (X_n) denote the Metropolis Markov chain with stationary distribution π. If we run the chain for many steps, the fraction of time spent at f_0 should be roughly $\pi(f_0)$. In practice this means that there is an initial period of time during which the chain "forgets" its starting state, and after that period is done, we observe $X_n = f_0$ over and over again.

Now, how does the Metropolis algorithm work? Let Q be the transition matrix for the proposal chain, and let P be the transition matrix for the Metropolis chain that we are trying to construct. For each distinct pair of states $x, y \in \mathcal{X}$, we will define a probability $0 \le R(x,y) \le 1$ and let

$$\begin{aligned} P(x,y) &= Q(x,y)R(x,y), \\ P(x,x) &= Q(x,x) + \sum_{y \neq x} Q(x,y)[1 - R(x,y)]. \end{aligned}$$

Starting from $X_n = x$, use Q to "propose" the next state: maybe $X_{n+1} = y$. Then flip a coin with probability $R(x,y)$ of heads. If heads, confirm the proposal and set $X_{n+1} = y$. If tails, reject the proposal and set $X_{n+1} = x$.

The detailed balance equations for P with respect to π are

$$\pi(x)Q(x,y)R(x,y) = \pi(y)Q(y,x)R(y,x)$$

for all distinct pairs $x, y \in \mathcal{X}$. Therefore we will have detailed balance as long as $R(x,y)$ and $R(y,x)$ have the right ratio, namely

$$\frac{R(x,y)}{R(y,x)} = \frac{\pi(y)Q(y,x)}{\pi(x)Q(x,y)}. \tag{1}$$

We'd like to make $R(x,y)$ and $R(y,x)$ as big as possible so that we accept as many as we can of the Q-chain's proposals, subject to the constraint that both $R(x,y)$ and $R(y,x)$ are at most 1. Let a be the right side of (1). If $a \le 1$, we set $R(y,x) = 1$ and $R(x,y) = a$. If $a \ge 1$, we set $R(x,y) = 1$ and $R(y,x) = 1/a$. This is concisely stated by the formula

$$R(x,y) = \min\left\{1, \frac{\pi(y)Q(y,x)}{\pi(x)Q(x,y)}\right\}, \tag{2}$$

which also works when we interchange x and y.

We've verified that if $R(x,y)$ is defined as in (2), then the Metropolis chain P will satisfy the detailed balance equations with respect to π, so in particular it has stationary distribution π. In addition, to compute $R(x,y)$ we only need to know the ratio $\pi(y)/\pi(x)$ rather than the values $\pi(x)$ and $\pi(y)$. This is crucial for the prisoners' code example because $\pi(x) = \mathrm{Pl}(x)/Z$ and $\pi(y) = \mathrm{Pl}(y)/Z$ are infeasible to compute, but the ratio $\pi(y)/\pi(x) = \mathrm{Pl}(y)/\mathrm{Pl}(x)$ is easy. This same phenomenon happens in many other applications.

Finally, note that the proposal chain for the prisoners' code is symmetric: if there are M symbols in the alphabet, then $Q(x,y) = Q(y,x) = 1/\binom{M}{2}$ if

y is obtained from x by swapping two symbols and $Q(x, y) = Q(y, x) = 0$ otherwise. This leads to the simpler formula

$$R(x, y) = \min\left\{1, \frac{\pi(y)}{\pi(x)}\right\} = \min\left\{1, \frac{\mathrm{Pl}(y)}{\mathrm{Pl}(x)}\right\},$$

which matches the algorithm described by Diaconis.