CSCI-6971 Lecture Notes: Markov Chain Monte Carlo methods*

Kristopher R. Beevers
Department of Computer Science
Rensselaer Polytechnic Institute
beevek@cs.rpi.edu

April 5, 2006

Suppose we need to draw samples from some complicated joint PDF $p(x_1, x_2, \ldots, x_n) = p(\mathbf{x})$. We have described importance sampling and sequential importance sampling techniques for drawing such samples. The basic idea of these approaches is to instead sample from some other "proposal" distribution $g(\mathbf{x})$ that approximates $p(\mathbf{x})$ and weight the samples according to $p(\mathbf{x})/g(\mathbf{x})$. However, this only works if $g(\mathbf{x})$ is a good approximation of $p(\mathbf{x})$. Often it is hard to come up with such a proposal density, particularly for high dimensional problems.

Markov Chain Monte Carlo (MCMC) methods use an alternative approach to instead generate samples from $p(\mathbf{x})$ iteratively based on Markov chains. The basic high-level MCMC algorithm for generating samples $\mathbf{x}_1, \ldots, \mathbf{x}_N$ is:

Algorithm 1 Basic MCMC

1: Draw initial state $\mathbf{x}_0$ from some initial distribution
2: **for** $t = 0$ to N **do**
3: Modify $\mathbf{x}_t$ according to some *proposal distribution* to obtain a proposed sample $\mathbf{x}'_{t+1}$:

$$\mathbf{x}'_{t+1} \sim q(\mathbf{x}_t, \mathbf{x}') \tag{1}$$

4: With some probability $A(\mathbf{x}_t, \mathbf{x}'_{t+1})$, *accept* $\mathbf{x}'_{t+1}$:

$$\mathbf{x}_{t+1} = \begin{cases} \mathbf{x}'_{t+1} & \text{with probability } A(\mathbf{x}_t, \mathbf{x}'_{t+1}) \\ \mathbf{x}_t & \text{otherwise} \end{cases} \tag{2}$$

There are a number of variants of the basic MCMC algorithm. They differ mainly in the form of the proposal distribution, the decomposition of the state $\mathbf{x}$ into components, and the form of the acceptance probability.

It is important to note that the proposal distribution in MCMC need not be similar or even related to $p(\mathbf{x})$, unlike the proposal distributions of importance sampling techniques.

*The primary sources for most of this material are: "Monte Carlo Strategies in Scientific Computing," J.S. Liu, Springer, 2001; "Probabilistic inference using Markov Chain Monte Carlo methods," R. Neal, TR CRG-TR-93-1, U. Toronto, 1993; "Sequential Monte Carlo methods for rigorous Bayesian modeling of Autonomous Compliant Motion," K. Gadeyne, PhD thesis, K.U. Leuven, 2005; and the author's own notes.

1 Metropolis-Hastings

The "core" MCMC algorithm, in terms of which most variants can be described, is the *Metropolis-Hastings algorithm*:

Algorithm 2 Metropolis-Hastings

1: Draw initial state $\mathbf{x}_0$ from some initial distribution
2: **for** $t = 0$ to N **do**
3: Sample $\mathbf{x}'_{t+1} \sim q(\mathbf{x}_t, \mathbf{x}')$, where the proposal distribution q need not (but may) depend on $\mathbf{x}_t$
4: Compute:

$$a(\mathbf{x}_t, \mathbf{x}'_{t+1}) = \frac{p(\mathbf{x}'_{t+1})}{p(\mathbf{x}_t)} \frac{q(\mathbf{x}'_{t+1}, \mathbf{x}_t)}{q(\mathbf{x}_t, \mathbf{x}'_{t+1})} \tag{3}$$

5: Draw $u \sim \text{UNIFORM}[0, 1]$
6: **if** $u \leq \min\{1, a(\mathbf{x}_t, \mathbf{x}'_{t+1})\}$ **then**
7: $\mathbf{x}_{t+1} = \mathbf{x}'_{t+1}$
8: **else**
9: $\mathbf{x}_{t+1} = \mathbf{x}_t$

Metropolis-Hastings is applicable as long as for any two states $\mathbf{x}, \mathbf{x}'$ it is possible to compute $p(\mathbf{x}')/p(\mathbf{x})$ and $q(\mathbf{x}', \mathbf{x})/q(\mathbf{x}, \mathbf{x}')$. (Note that the latter is simply 1 for symmetric proposal distributions.)

Asymptotically, Metropolis-Hastings samples are samples from $p(\mathbf{x})$ (which we will prove later). However, the samples are *not independent* because they are correlated through the proposal distribution $q(\mathbf{x}, \mathbf{x}')$. This correlation can have effects on the running time of Metropolis-Hastings required to allow the Markov chain to explore the entire state space.

It is also important to note that if a candidate state is rejected, the current state *becomes the new state* and should be counted again in any time averages or similar computations. This is in contrast to importance sampling where if a sample is discarded (e.g. during resampling) it does not contribute to the computation at hand.

1.1 Decomposing the state

Because it can be hard to define a fully joint proposal distribution, Metropolis-Hastings is often performed "component-wise," i.e., we modify only a single component or block of components x_k of $\mathbf{x}$ at a time. Components may be selected for modification according to some pre-specified random distribution (e.g. uniform over the components), but more frequently they are modified in sequence:

$$\begin{aligned} x'_1 &\sim q_1(\mathbf{x}, x_1) \\ x'_2 &\sim q_2(\mathbf{x}, x_2) \\ &\cdots \\ x'_n &\sim q_n(\mathbf{x}, x_n) \end{aligned}$$

Often, each component will be of the same "type" and it is appropriate to use the same proposal distribution for all components. In deciding how to decompose the state into components, there are several factors for consideration:

- How does the decomposition affect the choice of proposal distribution (cf. above)?

- It is desireable to use a decomposition for which the ratio $p(\mathbf{x}')/p(\mathbf{x})$ can be computed more efficiently if $\mathbf{x}$ and $\mathbf{x}'$ differ in only a single known component, as opposed to differing arbitrarily.
- It is also desireable to choose a decomposition such that the components of the state are nearly independent, which may speed exploration of the state space.

1.2 Choosing a proposal distribution

There are a number of obvious choices for the proposal distribution $q(\mathbf{x}, \mathbf{x}')$. For discrete components, it is common to choose the uniform distribution over the state space. An alternative is to use a distribution that is uniform over all values except the current one.

For continuous components, common choices are the Gaussian distribution (or multivariate Gaussian for compound components) centered on the current value; or alternatively the Cauchy distribution (or multivariate Cauchy), which has heavier tails that allow for occasional large jumps in the Markov chain. Both of these distributions are symmetric, so computing the acceptance probability is easy. They also both lead to ergodic Markov chains since there is a nonzero probability of moving to any state (see the discussion of convergence in Section 1.6 for more details). Yet another alternative which can be shown to maintain these properties is a uniform distribution over an interval centered at the current value. All of these distributions require width parameters that can be set either using *a priori* knowledge or via trial and error.

1.3 Burn-in period

Samples generated using Metropolis-Hastings are only *asymptotically* from $p(\mathbf{x})$, in much the same way as a Markov chain approaches the steady state or invariant distribution asymptotically (again, see Section 1.6). This implies that some samples from the beginning of the MCMC process should be discarded. The beginning of an MCMC process is thus known as the *burn-in period*. The length m of the burn-in period depends on the time required to explore the state space (often referred to as the *mixing time* in MCMC literature). Samples $\mathbf{x}_1, \ldots, \mathbf{x}_m$ generated by Metropolis-Hastings are discarded. This leads to the following general estimator:

$$\mathbf{E}\left[f(\mathbf{x})\right] \approx \frac{1}{T-m} \sum_{i=m+1}^{T} f(\mathbf{x}^i) \tag{4}$$

where m is the length of the burn-in period and T is the stopping time, i.e., the total number of samples to generate.

1.4 Stopping time

As we have noted, because of the correlation between samples, the Markov chain must be run long enough to fully explore the state space. There are several heuristics for determining when to stop sampling:

- The stopping time strongly depends on the "convergence ratio":

$$\frac{\text{typical step size of Markov chain}}{\text{representative length of state space}} \tag{5}$$

- The step size of the Markov chain depends upon the choice of proposal density, e.g., the step size is be proportional to var $(q(\mathbf{x}, \mathbf{x}'))$. Too large of a step size leads to many rejections; too small of a step size leads to slow mixing.

- For efficient mixing the step size should be the same order of magnitude as the smallest "length scale" of $p(\mathbf{x})$, e.g., if $p(\mathbf{x})$ is a Gaussian mixture, the standard deviation of the lowest-variance Gaussian.
- When adding an MCMC step to a sequential Monte Carlo sampler (known as the "Resample-Move" strategy or "roughening"), it is typical to use a Gaussian proposal density with the current sample as the mean and the sample variance as the variance.

1.5 Sample independence

The correlation between samples affects the quality of MCMC results and the iterations required before stopping. Reducing the correlation is thus desirable. There are several strategies for generating more independent samples; two of the simplest are:

- Start and run several independent Markov chains in parallel
- Keep only every kth sample from the Markov chain

1.6 Convergence

As we have stated earlier, Metropolis-Hastings asymptotically generates samples from $p(\mathbf{x})$. To prove this, we rely on several ideas from our earlier discussion of Markov chains.

Definition 1.1. Given some initial distribution $f^{(0)}(\mathbf{x})$ for the Markov chain and a *transition kernel* $T(\mathbf{x}, \mathbf{x}') = p(\mathbf{x}'|\mathbf{x})$, the PDF for the chain at time t is:

$$f^{(t)}(\mathbf{x}') = \int T(\mathbf{x}, \mathbf{x}') f^{(t-1)}(\mathbf{x})\, d\mathbf{x} \tag{6}$$

In the context of discrete-time Markov chains we have termed this the *t-step transition probability*.

Definition 1.2. Recall that a Markov chain is *ergodic* if all of its states are both *aperiodic* and *recurrent*. (See the notes on discrete-time Markov chains for further discussion.) An ergodic Markov chain is *time reversible* with invariant distribution $f(\mathbf{x})$ if it satisfies the *detailed balance equation*:

$$T(\mathbf{x}^a, \mathbf{x}^b) f(\mathbf{x}^b) = T(\mathbf{x}^b, \mathbf{x}^a) f(\mathbf{x}^a) \tag{7}$$

or equivalently:

$$f(\mathbf{x}^b) = \int T(\mathbf{x}^b, \mathbf{x}^a) f(\mathbf{x}^a)\, d\mathbf{x}^a \tag{8}$$

(See the earlier notes for a derivation.)

It is known that any ergodic Markov chain that satisfies the detailed balance equation eventually converges to the steady state (or invariant) distribution of the chain, $f(\mathbf{x})$, regardless of $f^{(0)}(\mathbf{x})$.

Theorem 1.3. *The Metropolis-Hastings algorithm asymptotically generates samples from $p(\mathbf{x})$.*

Proof: We prove the theorem by showing that the Markov chain described by Metropolis-Hastings is ergodic and staisfies the detailed balance equation, and so has $p(\mathbf{x})$ as its invariant distribution.

The Markov chain is ergodic as long as $T(\mathbf{x}, \mathbf{x}') \neq 0$ for all $\mathbf{x}, \mathbf{x}'$ (i.e., as long as it is possible to reach every state from any other state). From the Metropolis-Hastings algorithm, we have:

$$T(\mathbf{x}, \mathbf{x}') = q(\mathbf{x}, \mathbf{x}')a(\mathbf{x}, \mathbf{x}') + I(\mathbf{x} = \mathbf{x}') \left(1 - \int q(\mathbf{x}', \mathbf{x}'')a(\mathbf{x}', \mathbf{x}'')\, d\mathbf{x}''\right) \tag{9}$$

Here, I corresponds to the indicator function (1 if the argument is true, 0 otherwise). The transition kernel is composed of two terms. The first term represents the probability of arriving in some state $\mathbf{x}' \neq \mathbf{x}$ and is nonzero as long as $p(\mathbf{x}) > 0$ and $q(\mathbf{x}, \mathbf{x}') > 0$ for all $\mathbf{x}, \mathbf{x}'$. The second term corresponds to the probability of remaining in state $\mathbf{x}$, where $1 - \int q(\mathbf{x}'', \mathbf{x}')a(\mathbf{x}'', \mathbf{x}')\, d\mathbf{x}''$ encodes the probability of rejecting a move (by integrating over all possible moves). Since the second term is also positive, there is a nonzero probability of remaining in $\mathbf{x}$. Thus, all possible states can be reached from any state $\mathbf{x}$ so the Markov chain is ergodic, given suitable proposal and target densities.

We must also show that the detailed balance equation holds for the invariant distribution equal to the target distribution $p(\mathbf{x})$. If $\mathbf{x} = \mathbf{x}'$ this is trivial. Otherwise, we have:

$$\begin{aligned}
T(\mathbf{x}, \mathbf{x}')p(\mathbf{x}') &= T(\mathbf{x}', \mathbf{x})p(\mathbf{x}) && (10)\\
q(\mathbf{x}, \mathbf{x}')a(\mathbf{x}, \mathbf{x}')p(\mathbf{x}') &= q(\mathbf{x}', \mathbf{x})a(\mathbf{x}', \mathbf{x})p(\mathbf{x}) && (11)\\
\frac{a(\mathbf{x}, \mathbf{x}')}{a(\mathbf{x}', \mathbf{x})} &= \frac{q(\mathbf{x}', \mathbf{x})p(\mathbf{x})}{q(\mathbf{x}, \mathbf{x}')p(\mathbf{x}')} && (12)
\end{aligned}$$

and we see that the acceptance probability used by Metropolis-Hastings suffices. □

2 The Gibbs sampler

A commonly encountered version of MCMC is the *Gibbs sampler*. The Gibbs sampler is essentially a variation of Metropolis-Hastings in which each component $x_k \in \mathbf{x}$ is replaced in turn by sampling from its conditional distribution given the values of all the other components. Typically the component-wise transitions are applied in sequence as with other MCMC approaches:

$$\begin{aligned}
x_1^{(t+1)} &\sim p(x_1 | x_2^{(t)}, x_3^{(t)}, \ldots, x_n^{(t)})\\
x_2^{(t+1)} &\sim p(x_2 | x_1^{(t+1)}, x_3^{(t)}, \ldots, x_n^{(t)})\\
&\ldots\\
x_i^{(t+1)} &\sim p(x_i | x_1^{(t+1)}, \ldots, x_{i-1}^{(t+1)}, x_{i+1}^{(t)}, \ldots, x_n^{(t)})\\
&\ldots\\
x_n^{(t+1)} &\sim p(x_n | x_1^{(t+1)}, x_2^{(t+1)}, \ldots, x_{n-1}^{(t+1)})
\end{aligned}$$

Note that the new value for x_{i-1} is used *immediately* in sampling x_i. A reasonable alternative is to pick an x_i to update at random from a pre-specified distribution.

Whether Gibbs sampling can be applied depends heavily on whether it is possible to easily sample from the conditional distributions of the components. In the best case, the conditional distribution has some parametric form (e.g. Gaussian) from which we know how to sample. As with Metropolis-Hastings, it is desireable to group several state variables into a single component if a convenient way of generating from the multivariate conditional distribution is available.

3 Other MCMC variants

There are many other variations of MCMC. A few of the more popular ones are:

- **Metropolis sampling**: developed prior to the more general Metropolis-Hastings algorithm, the Metropolis sampler assumes a symmetric proposal distribution and uses the acceptance probability $a(\mathbf{x}, \mathbf{x}') = \min\{1, p(\mathbf{x}')/p(\mathbf{x})\}$.

- **Boltzman sampling**: this algorithm uses the "Boltzman" acceptance function:

$$A(\mathbf{x}, \mathbf{x}') = \frac{p(\mathbf{x}')}{p(\mathbf{x}) + p(\mathbf{x}')} \tag{13}$$

  The basic idea is to "forget" which of $\mathbf{x}$ or $\mathbf{x}'$ is the current state, and then select between them at random according to their relative probabilities. However, there are no clear advantages (or disadvantages) of this approach with respect to the Metropolis-Hastings technique.

- **Independence sampling**: the proposal distribution $q(\mathbf{x}, \mathbf{x}')$ may be independent of the current state $\mathbf{x}$. MCMC with such a proposal is known as independence sampling. This approach only works well if q is a good approximation of $p(\mathbf{x})$.