**10-708: Probabilistic Graphical Models 10-708, Spring 2014**

# 17 : Approximate Inference: Markov Chain Monte Carlo (MCMC)

*Lecturer: Eric P. Xing* *Scribes: Dan Schwartz, Karanhaar Singh*

## Recap

Recall that Monte Carlo methods are used to solve the following two tasks:

1. Generate samples from some given target probability distribution $P(x)$
2. Estimate expectations of various functions under this distribution

$$\mathbb{E}_{P(x)}(\phi(x)) = \int P(x)\phi(x)\,dx$$

These two problems are useful for approximating $P(x)$ itself and for revealing interesting information about $P(x)$ (such as various moments) respectively. Unfortunately, these are difficult problems for a variety of reasons:

- When drawing samples, we typically only have access to an unnormalized distribution

  $$P^*(x) = \frac{P(x)}{Z}$$

  Computation of the normalization factor $Z$ can be a difficult task since we have to visit every point in the space of $x$.

- Drawing samples from $P(x)$ can be very difficult in high dimensional spaces. It is not obvious to see how we can sample from $P(x)$ without enumerating all possible states, which exponentially grows with the number of dimensions. This quickly makes such a task computationally intractable.

In the previous lecture we already discussed three basic Monte Carol methods: uniform sampling, rejection sampling, and importance sampling. While these methods are useful for certain problems, they come with limitations:

- Uniform Sampling: This is one of the most basic samplers where we can compute an approximation to the expectation of some function by uniformly sampling the state space. The problem with this sampler is that it could take a very long time to hit the typical set (where the majority of the distribution lies), especially for higher dimensional problems.

- Importance Sampling: This sampler draws samples, $x$, from an auxiliary proposal density $Q(x)$ and uses weights to adjust the "importance" of the sampled point to compensate for the fact that we sample from a different distribution. This sampler requires $Q(x)$ to be a very close approximation to $P(x)$ for acceptable performance, especially in higher dimensions.

- Rejection Sampling: This sampler also draws samples, $x$, from an auxiliary proposal density $Q(x)$ but adjusts the importance of the sampled points through rejection. Like importance sampling, this sampler requires $Q(x)$ to be a very close approximation to $P(x)$ for acceptable performance, especially in higher dimensions.

All of these methods also suffer from the problem that the variance of the sample can be small even if the sample is not a good representation of the true distribution.

If we use an adaptive proposal $Q\left(x'|x^{(t)}\right)$ as opposed to a fixed $Q(x')$, we can relax the requirement imposed by importance and rejection sampling that $Q(x')$ must be very similar to $P(x')$. This intuition motivates the use of Markov Chain Monte Carol (MCMC) algorithms. Intuitively, we want to move a little away from the space we have already sampled according to a probability distribution that takes into account both the true probability and the sampling distribution.

**Metropolis-Hastings algorithm**

One such MCMC method is the Metropolis-Hastings algorithm:

1. Initialize the starting state $x^{(t)}$ at $t = 0$
2. Draw a sample $x'$ from the proposal $Q\left(x'|x^{(t)}\right)$. Note that this proposal is now a function of the previously drawn sample $x^{(t)}$ (at time step $t$)
3. Decide whether we should accept this new state by first computing the following ratio of importance sampling weights of $x'$ and $x^{(t)}$:

$$\alpha = \frac{W_{x'}}{W_{x^{(t)}}} = \frac{P^*(x')\,Q\left(x^{(t)}|x'\right)}{P^*\left(x^{(t)}\right)Q\left(x'|x^{(t)}\right)}$$

   Where

$$W_{x'} = \frac{P^*(x')}{Q\left(x'|x^{(t)}\right)}, W_{x^{(t)}} = \frac{P^*\left(x^{(t)}\right)}{Q\left(x^{(t)}|x'\right)}$$

   We accept the new state with a probability of

$$A\left(x'|x^{(t)}\right) = \min(1, \alpha)$$

   If the new state is accepted, we define the new state to be the value we had just drawn: $x^{(t+1)} \leftarrow x'$, otherwise, the new state is simply set to be the previous state: $x^{(t+1)} \leftarrow x^{(t)}$. Note that because $P$ is in both the numerator and denominator, we can use the unnormalized $P^*$ and there is no need to find the partition function $Z$. Compare this update to the rejection sampling algorithm where rejected samples are thrown away. In this case, a rejection results in the same sample being written to the list of collected samples.

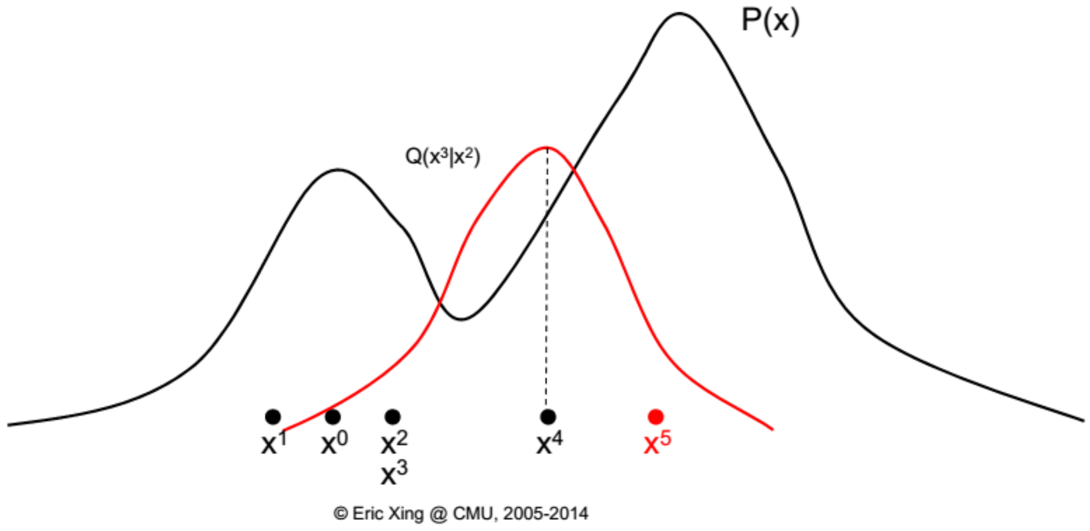
4. Repeat steps 2 and 3 until the samples "converge". This process is called the burn-in period and the definition of convergence will be covered later in this writeup.

This ratio, $A\left(x'|x^{(t)}\right)$, is the ratio of the probability of going from $x'$ to $x^{(t)}$ versus going from $x^{(t)}$ to $x'$. When the ratio is higher, $x'$ is a good place to move in the sample space. When the ratio is low, the algorithm mostly will stay at $x^{(t)}$, but the kernel $Q$ still spreads the area we may move to around the sample space.

The Metropolis-Hastings algorithm will seize events that are rare under $Q$ but have high probability under $P$, causing a dramatic shift in the proposal distribution $Q$. One issue with the algorithm is that it can be hard to move from one high probability space to another across a low probability space. Although Metropolis-Hastings will converge to the true distribution, with certain exceptions, there are no guarantees to when. In practice, 5000-10000 iterations are typically used as the burn-in period.

### Metropolis-Hastings example

Consider the task of sampling from a bimodal distribution $P(x)$. We can define our adaptive proposal distribution, $Q\left(x'|x^{(t)}\right)$, to be a Gaussian distribution whose mean is the previous state $x^{(t)}$ and whose standard deviation is fixed.



In this figure, $x^{(0)}$ is the initial state serving as the mean of the adaptive proposal distribution in this state. We sample $x'$, accept this sample based on the criteria detailed above, set the next state $x^{(1)}$ to be this accepted state, update the mean of the adaptive proposal distribution, and repeat. Note that $x^{(3)}$ marks a rejected sample since this state is set to be the previous state, $x^{(2)}$.

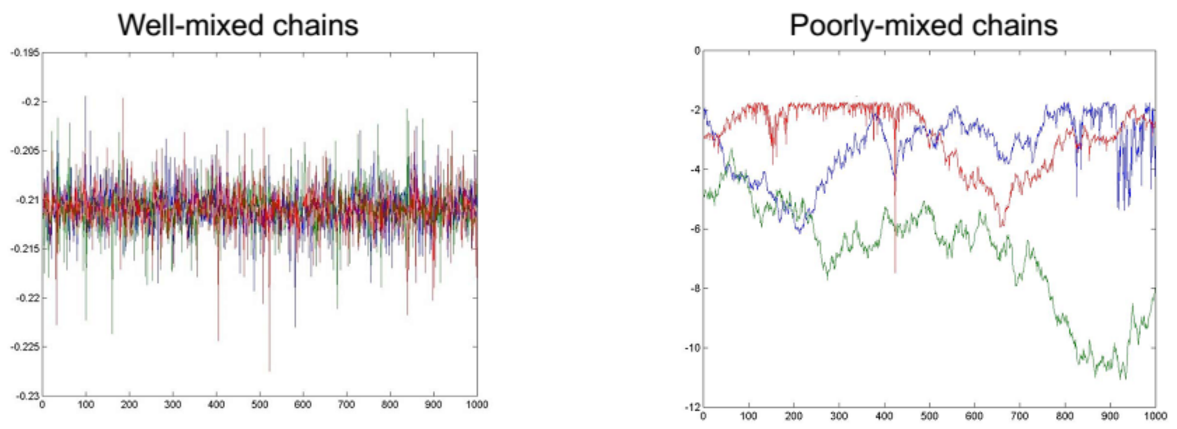
### Metropolis-Hastings convergence

Recall that in rejection sampling all accepted points are independently drawn samples from the target distribution. This desirable property comes at the cost of having a low acceptance rate, especially when the proposal distribution is not a good approximation of the target distribution. The Metropolis-Hastings algorithm, on the other hand, will generate samples based on a Markov process. Since a given sample is dependent on the state of the previous sample, the samples generated are clearly not independent from each other. In order to obtain samples that are effectively independent, we may need to run the algorithm for a considerable amount of time. A good rule of thumb is that if the largest length scale of the state space of the target distribution is $L$ and the smallest length scale of the state space of the proposal distribution is $\epsilon$, we can define a lower bound on the number of iterations of the Metropolis-Hastings method:

$$T \approx \left(\frac{L}{\epsilon}\right)^2$$

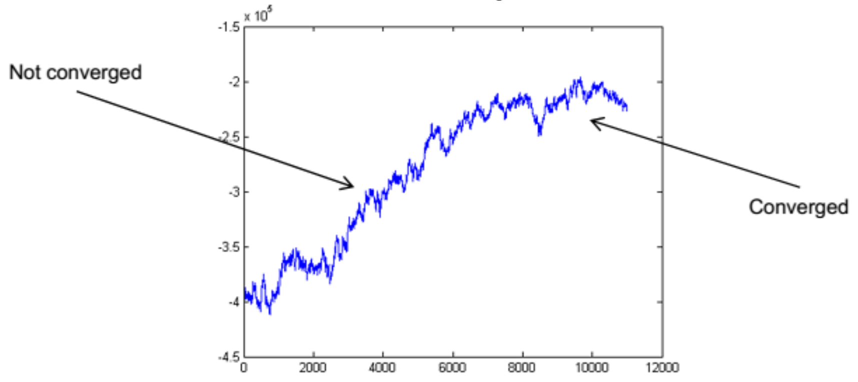
Adjusting the length scale of the proposal distribution results in a trade-off between the acceptance rate and the level of dependence between consecutive samples. Recall that in high dimensional spaces, most of the mass tends to be concentrated in small portions of the state space. Thus if the length scale $\epsilon$ is too large, the sampler will tend to pick points in low probability parts of the true distribution and the rejection rate will be high. However, if the length scale is too small, the sampler will take longer to explore the state space and therefore take longer to converge. Intuitively, we can set $\epsilon$ to be a number smaller than, but on a similar order of magnitude, to $L$, but we do have to be careful as selection of the optimal $\epsilon$ is highly dependent on the properties of the target distribution.

We can monitor convergence by plotting samples over multiple runs of a Metropolis-Hastings sampling method. If the resultant chains are well mixed, then we can state that samples have probably converged, otherwise we should continue the burn in process.



© Eric Xing @ CMU, 2005-2014

One problem with visualizing the chains in this way is that visualizing all dimensions of random variables becomes hard as we increase the dimensionality of the problem. One way to mitigate this issue is to plot the complete log-likelihood function, which is dependent on all model random variables over time. Typically, we will see the log-likelihood plateau once we approach convergence.



© Eric Xing @ CMU, 2005-2014

**Markov chains**

At its core, the Metropolis-Hasting method is based on states selected by a Markov chain. A Markov chain is a sequence of random variables that satisfy the Markov property which states that each variable is conditionally independent of its ancestors given its parent. Define $p^{(t)}(x)$ to be the probability distribution of the state at time step $t$ and $\pi(x)$ as the true distribution. We define a Markov chain by an initial probability distribution $p^{(0)}(x)$ and a transitional probability $T(x'|x)$ .

For sampling, we consider only homogeneous Markov chains, where $T(x'|x)$ is independent of the time step $t$. In the case of Metropolis-Hastings, the transition probability is: $T(x'|x) = A(x'|x)Q(x'|x)$

It is often helpful to think of an MCMC sampler as moving from one distribution of states to another, rather than from one state to another. This can be formalized:

$p^{(t+1)}(x') = \int T(x'|x)p^{(t)}(x)dx$

When designing a Markov chain for sampling, we must ensure that the following properties hold:

1. The target distribution must be an invariant distribution of the chain. Formally:

$$\pi(x') = \int T(x'|x)\,\pi(x)\,dx\ \forall x'$$

2. The Markov chain must be ergodic. Intuitively, this means that we can reach the final stationary distribution no matter what initial distribution we start with. Formally:

$$\forall p^{(0)}(x)\,,\ \text{as } t \to \infty:$$
$$p^{(t)}(x) \to \pi(x)$$

   In order for a Markov chain to be ergodic, the following two properties must hold:

   (a) The chain must be irreducible. Specifically, we must be able to reach all parts of the state space in a finite number of steps. This property ensures that the sampler can populate the whole state space.

   (b) The chain must be aperiodic. Specifically, the chain will not tend towards a periodic limit-cycle where we can only reach a certain subset of states at any time step. In other words, we must be able to reach all states at any given time step. This property ensures that the sampler does not get stuck in a cycle. A simple example of a periodic limit-cycle is the random walk along the edges of a square - we cannot move to the vertex across the diagonal of the square at any given time step.

The detailed balance property is defined as follows:

$$T(x'|x)\,P(x) = T(x|x')\,P(x')\,\forall x, x'$$

Markov chains that satisfy this property are called reversible Markov chains. We can actually prove that all reversible Markov chains will converge to the target distribution, in other words, all reversible Markov chains have a stationary distribution. Detailed balance is a sufficient, but not necessary condition for having a stationary distribution:

$$T(x'|x)\,P(x) = T(x|x')\,P(x')\,\forall x, x'$$

Take the sum over the space of $x'$ on both sides

$$\sum_{x'} T(x'|x)\,P(x) = \sum_{x'} T(x|x')\,P(x')\,\forall x, x'$$

Pull out $P(x)$, which is not a function of $x'$

$$P(x)\sum_{x'} T(x'|x) = \sum_{x'} T(x|x')\, P(x')\, \forall x, x'$$

Note that summation over the entire space of any probability distribution is 1, so we can eliminate the first summation

$$P(x) = \sum_{x'} T(x|x')\, P(x')\, \forall x, x'$$

This is the definition of a stationary distribution.

Metropolis-Hastings is an example of a reversible Markov chain:

$$T(x'|x) = Q(x'|x)A(x'|x)$$

$$A(x'|x) = \min\left(1, \frac{P(x')Q(x|x')}{P(x)Q(x'|x)}\right)$$

Assume without loss of generality

$$A(x'|x) \leq 1 \Rightarrow A(x'|x) = \frac{P(x')Q(x|x')}{P(x)Q(x'|x)}$$

$$\Rightarrow P(x)Q(x'|x)A(x'|x) = P(x')Q(x|x')$$

Note:

$$A(x'|x) \leq 1 \Rightarrow \frac{P(x')Q(x|x')}{P(x)Q(x'|x)} \leq 1 \Rightarrow \frac{P(x)Q(x'|x)}{P(x')Q(x|x')} \geq 1 \Rightarrow A(x|x') = 1$$

Therefore (multiplying by a special form of 1)

$$P(x)Q(x'|x)A(x'|x) = P(x')Q(x|x')A(x|x')$$

This is detailed balance

$$P(x)T(x'|x) = P(x')T(x|x')$$

One way to construct Markov chains is by mixing base transition functions, $B$, to define our transition matrix $T$ . When transitioning to a new state, we pick a base transition function at random defined by a set of probabilities associated with each base transition function. The individual base transition functions do not need to be ergodic but all must keep the invariance property satisfied.

**Gibbs Sampling**

Gibbs sampling is a special case of the Metropolis-Hastings method where the proposal distributions are tractable conditional distributions on $P(x)$. Specifically, we assume that the conditional distributions hold the following form: $P(x_i | \{x_j\}\, \forall j \neq i)\, \forall i$. This form of the proposal distribution fits very nicely with various probabilistic graphical models such as mixture models and Latent Dirichlet allocation models. The algorithm of the Gibbs sampler is very similar to the standard Metropolis-Hastings algorithm:

1. For a model that contains $K$ variables, initialize the starting states $x_1^{(t)}, \ldots, x_K^{(i)}$ at $t = 0$

2. For some ordering of the $K$ variables, draw samples one parameter at a time:

$$x_1^{(t+1)} \sim P\left(x_1'|x_2^{(t)}, ...x_K^{(t)}\right)$$

$$\vdots$$

$$x_K^{(t+1)} \sim P\left(x_k'|x_1^{(t)}, ...x_{K-1}^{(t)}\right)$$

3. Repeat until convergence.

To see that Gibbs sampling is a special case of Metropolis-Hastings:

Define $x_i$ to be the $i^{th}$ element of the feature vector $x$ and $x_{-i}$ to be all other elements. Let

$$Q(x_i', x_{-i}|x_i, x_{-i}) = P(x_i'|x_{-i})$$

then

$$A(x_i', x_{-i}|x_i, x_{-i}) = \min\left(1, \frac{P(x_i', x_{-i})Q(x_i, x_{-i}|x_i', x_{-i})}{P(x_i, x_{-i})Q(x_i', x_{-i}|x_i, x_{-i})}\right)$$

$$= \min\left(1, \frac{P(x_i', x_{-i})P(x_i|x_{-i})}{P(x_i, x_{-i})P(x_i'|x_{-i})}\right)$$

$$= \min\left(1, \frac{P(x_i'|x_{-i})P(x_{-i})P(x_i|x_{-i})}{P(x_i|x_{-i})P(x_{-i})P(x_i'|x_{-i})}\right)$$

$$= \min(1, 1) = 1$$

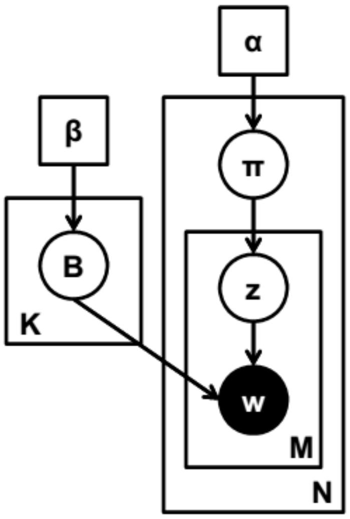
Thus, $A(x'|x)$ is always 1 in the Gibbs special case of Metropolis-Hastings, so every sample is accepted.

If we are performing Gibbs sampling on a distribution that can be modeled by a probabilistic graphical model, the complicated form of the conditional distribution can be simplified by conditioning only on the Markov Blanket. Since Gibbs sampling is a Metropolis-Hastings method, it will produce samples that are not independent from each other meaning that it may take a considerable amount of time before the algorithm generates an independent sample. Specifically, the lower bound on the number of iterations required to generate an independent sample is $\left(\frac{L}{\epsilon}\right)^2$.

### Rao-Blackwell/Collapsed Gibbs

Rao-Blackwell is a very general method that can be applied to many types of sampling. When parts of a distribution can be computed analytically, some random variables can be integrated out of the distributions of interest to reduce the variance of the sampling procedure. This can have a large effect on the number of samples that need to be taken before convergence.

One example of Rao-Blackwell is the collapsed Gibbs sampler inference algorithm used by Griffiths and Steyvers for finding word-topic assignments in their mixture model for finding scientific topics. The key insight to this sampler is that we can integrate out certain random variables such that we only need to sample the variable of interest, the word-topic assignments $z$.



© Eric Xing @ CMU, 2005-2014

Since we have integrated over the topics and topic vectors, only samples of $z$ need to be taken. Since $\pi$ and $B$ are drawn from Dirichlet distributions, they can be integrated out of the conditional distribution of $z_i$.

Thus, after integration, the conditional probability distribution, $P\left(z_i|z_{-i}, w\right)$, is the product of two Dirichlet-Multinomial conditional distributions (note that after integrating out $\pi$, the $\{z_i\}$ become dependent on each other).

$$P\left(z_i = j|z_{-i}, w\right) \propto \left[\frac{n_{-i,j}^{(w_i)} + \beta}{n_{-i,j}^{(\cdot)} + W\beta}\right]\left[\frac{n_{-i,j}^{(d_i)} + \alpha}{n_{-i,\cdot}^{(\cdot)} + T\alpha}\right]$$

The first term of this equation represents the word-topic term and the second represents the doc-topic term. Each $n$ term represents the number of word positions within the domain specified by the superscript for a given topic $j$ (excluding $w_i$).

**Auto-correlation**

Given an execution of a Markov chain, we can plot the auto-correlation of the resultant states. The auto-correlation function gives us a measure of how correlated nearby samples are to each other. This can help us determine when two samples are "far enough" to be considered independent draws. We define the auto-correlation function as follows:

$$R_x\left(k\right) = \frac{\sum_{t=1}^{n-k}\left(x_t - \bar{x}\right)\left(x_{t+k}\right)}{\sum_{t=1}^{n-k}\left(x_t - \bar{x}\right)}$$

We can also define the Sample Size Inflation Factor (SSIF) which is based on two adjacent draws:

$$s_x = \frac{1 + R_x\left(1\right)}{1 - R_x\left(1\right)}$$

The SSIF gives us a measure of "effective sample size". We can define the effective sample size as $\frac{n}{s}$ where a higher auto-correlation gives us a smaller effective sample size.

**Acknowledgments**

Portions of these scribe notes have been taken directly from

- Eric Xing's Probabilistic Graphical Models Lecture 17 notes from Spring 2014
- Information Theory, Inference, and Learning Algorithms by David J. C. Mackay, Chpt 29
- D. Mackay, Introduction to Monte Carlo Methods

All figures have been taken from Eric Xing's Probabilistic Graphical Models Lecture 17 notes from Spring 2014.