The Clever Machine

Topics in Computational Neuroscience & Machine Learning

MCMC: The Gibbs Sampler

NOV 5
Posted by **dustinstansbury**

In the previous post, (https://theclevermachine.wordpress.com/2012/11/04/mcmc-multivariate-distributions-block-wise-component-wise-updates/) we compared using block-wise and component-wise implementations of the Metropolis-Hastings algorithm for sampling from a multivariate probability distribution$p(\mathbf{x})$. Component-wise updates for MCMC algorithms are generally more efficient for multivariate problems than blockwise updates in that we are more likely to accept a proposed sample by drawing each component/dimension independently of the others. However, samples may still be rejected, leading to excess computation that is never used. The Gibbs sampler, another popular MCMC sampling technique, provides a means of avoiding such wasted computation. Like the component-wise implementation of the Metropolis-Hastings algorithm, the Gibbs sampler also uses component-wise updates. However, unlike in the Metropolis-Hastings algorithm, all proposed samples are accepted, so there is no wasted computation.

The Gibbs sampler is applicable for certain classes of problems, based on two main criterion. Given a target distribution $p(\mathbf{x})$, where $\mathbf{x} = (x_1, x_2, \ldots, x_D)$, The first criterion is 1) that it is necessary that we have an analytic (mathematical) expression for the conditional distribution of each variable in the joint distribution given all other variables in the joint. Formally, if the target distribution $p(\mathbf{x})$ is D -dimensional, we must have D individual expressions for

$$p(x_i|x_1, x_2, \ldots, x_{i-1}, x_{i+1}, \ldots, x_D)$$

$$= p(x_i|x_j), j \neq i.$$

Each of these expressions defines the probability of the i-th dimension given that we have values for all other ($j \neq i$) dimensions. Having the conditional distribution for each variable means that we don't need a proposal distribution or an accept/reject criterion, like in the Metropolis-Hastings algorithm. Therefore, we can simply sample from each conditional while keeping all other variables held fixed. This leads to the second criterion 2) that we must be able to sample from each conditional distribution. This caveat is obvious if we want an implementable algorithm.

The Gibbs sampler works in much the same way as the component-wise Metropolis-Hastings algorithms except that instead drawing from a proposal distribution for each dimension, then accepting or rejecting the proposed sample, we simply draw a value for that dimension according to the variable's corresponding conditional distribution. We also accept all values that are drawn.

Similar to the component-wise Metropolis-Hastings algorithm, we step through each variable sequentially, sampling it while keeping all other variables fixed. The Gibbs sampling procedure is outlined below

1. set $t = 0$
2. generate an initial state $\mathbf{x}^{(0)} \sim \pi^{(0)}$
3. repeat until $t = M$

set $t = t + 1$

for each dimension $i = 1..D$

draw x_i from $p(x_i|x_1, x_2, \ldots, x_{i-1}, x_{i+1}, \ldots, x_D)$

To get a better understanding of the Gibbs sampler at work, let's implement the Gibbs sampler to solve the same multivariate sampling problem addressed in the previous post.

Example: Sampling from a bivariate a Normal distribution

This example parallels the examples in the previous post where we sampled from a 2-D Normal distribution using block-wise and component-wise Metropolis-Hastings algorithms. Here, we show how to implement a Gibbs sampler to draw samples from the same target distribution. As a reminder, the target distribution $p(\mathbf{x})$ is a Normal form with following parameterization:

$$p(\mathbf{x}) = \mathcal{N}(\mu, \Sigma)$$

with mean

$$\mu = [\mu_1, \mu_2] = [0, 0]$$

and covariance

$$\Sigma = \begin{bmatrix} 1 & \rho_{12} \\ \rho_{21} & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0.8 \\ 0.8 & 1 \end{bmatrix}$$

In order to sample from this distribution using a Gibbs sampler, we need to have in hand the conditional distributions for variables/dimensions x_1 and x_2:

$p(x_1|x_2^{(t-1)})$ (i.e. the conditional for the first dimension, x_1)

and

$p(x_2|x_1^{(t)})$ (the conditional for the second dimension, x_2)

Where $x_2^{(t-1)}$ is the previous state of the second dimension, and $x_1^{(t)}$ is the state of the first dimension after drawing from $p(x_1|x_2^{(t-1)})$. The reason for the discrepancy between updating x_1 and x_2 using states $(t-1)$ and (t), can be is seen in step 3 of the algorithm outlined in the previous section. At iteration t we first sample a new state for variable x_1 conditioned on the most recent state of variable x_2, which is from iteration $(t-1)$. We then sample a new state for the variable x_2 conditioned on the most recent state of variable x_1, which is now from the current iteration, t.

After some math (which which I will skip for some brevity, but see the following

(http://fourier.eng.hmc.edu/e161/lectures/gaussianprocess/node7.html) for some details), we find that the two conditional distributions for the target Normal distribution are:

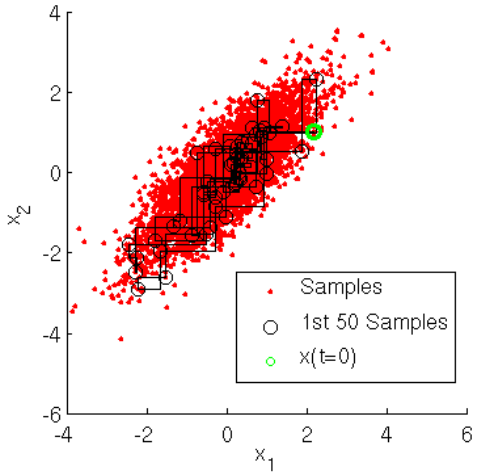
$$p(x_1|x_2^{(t-1)}) = \mathcal{N}(\mu_1 + \rho_{21}(x_2^{(t-1)} - \mu_2), \sqrt{1-\rho_{21}^2})$$

and

$$p(x_2|x_1^{(t)}) = \mathcal{N}(\mu_2 + \rho_{12}(x_1^{(t)} - \mu_1), \sqrt{1-\rho_{12}^2}),$$

which are both univariate Normal distributions, each with a mean that is dependent on the value of the most recent state of the conditioning variable, and a variance that is dependent on the target covariances between the two variables.

Using the above expressions for the conditional probabilities of variables x_1 and x_2, we implement the Gibbs sampler using MATLAB below. The output of the sampler is shown here:



(https://theclevermachine.files.wordpress.com/2012/11/gibbssampler-2dnormal1.png)
Gibbs sampler Markov chain and samples for bivariate Normal target distribution

Inspecting the figure above, note how at each iteration the Markov chain for the Gibbs sampler first takes a step only along the x_1 direction, then only along the x_2 direction. This shows how the Gibbs sampler sequentially samples the value of each variable separately, in a component-wise fashion.


```
% EXAMPLE: GIBBS SAMPLER FOR BIVARIATE NORMAL
rand('seed' ,12345);
nSamples = 5000;

mu = [0 0]; % TARGET MEAN
rho(1) = 0.8; % rho_21
rho(2) = 0.8; % rho_12

% INITIALIZE THE GIBBS SAMPLER
propSigma = 1; % PROPOSAL VARIANCE
minn = [-3 -3];
maxx = [3 3];
```

Wrapping Up

The Gibbs sampler is a popular MCMC method for sampling from complex, multivariate probability distributions. However, the Gibbs sampler cannot be used for general sampling problems. For many target distributions, it may difficult or impossible to obtain a closed-form expression for all the needed conditional distributions. In other scenarios, analytic expressions may exist for all conditionals but it may be difficult to sample from any or all of the conditional distributions (in these scenarios it is common to use univariate sampling methods such as rejection sampling and (surprise!) Metropolis-type MCMC techniques to approximate samples from each conditional). Gibbs samplers are very popular for Bayesian methods where models are often devised in such a way that conditional expressions for all model variables are easily obtained and take well-known forms that can be sampled from efficiently.

Gibbs sampling, like many MCMC techniques suffer from what is often called "slow mixing." Slow mixing occurs when the underlying Markov chain takes a long time to sufficiently explore the values of $\mathbf{x}$ in order to give a good characterization of $p(\mathbf{x})$. Slow mixing is due to a number of factors including the "random walk" nature of the Markov chain, as well as the tendency of the Markov chain to get "stuck," only sampling a single region of $\mathbf{x}$ having high-probability under $p(\mathbf{x})$. Such behaviors are bad for sampling distributions with multiple modes or heavy tails. More advanced techniques, such as Hybrid Monte Carlo have been developed to incorporate additional dynamics that increase the efficiency of the Markov chain path. We will discuss Hybrid Monte Carlo in a future post.

About dustinstansbury

I recently received my PhD from UC Berkeley where I studied computational neuroscience and machine learning.

View all posts by dustinstansbury »

Posted on November 5, 2012, in Algorithms, Density Estimation, Sampling, Sampling Methods, Statistics and tagged Conditional Distribution, Gibbs Sampler, MCMC, Multivariate Normal. Bookmark the permalink. 4 Comments.

Leave a comment

Trackbacks 2

Comments 2

markovmagic | August 28, 2013 at 6:21 pm
Reblogged this on Machine Learning Magic and commented:
This man is a genius.

chjko0206 | February 9, 2015 at 12:11 am
Reblogged this on michaelhjc.

1. Pingback: **A Gentle Introduction to Markov Chain Monte Carlo (MCMC) « The Clever Machine**
2. Pingback: **MCMC: The Gibbs Sampler, simple example w/ Matlab code | Victor Fang's Computing Space**