10-708: Probabilistic Graphical Models, Spring 2020

10: Sampling 2

Lecturer: Eric P. Xing Scribe: Joshua Uyheng, Geyang Zhang, Mike Chen, Wenyu Huang, Niles Christensen

1 Challenges with Markov Chain Monte Carlo

There are several challenges to using Markov Chain Monte Carlo (MCMC). First of all, performing random walks in MCMC may have poor acceptance rates, which means that we need to increase the total number of samples drawn. Moreover, the effective number of samples is small because samples can have high correlation between each other given the method that is used to draw samples in MCMC. Finally, it can be unclear from first principles when to stop burn-in.

1.1 Poor Acceptance Rates

There are tradeoffs in choosing $Q(x'|x)$ that have to do with whether the distribution has low or high variance. Proposals have to be selected to balance sufficient variance (ability to explore distribution $P(x)$) with high enough acceptance rates for speeding up sampler.

- **Low-variance** (narrow) proposals might have **high acceptance rates** but do not explore $P(x)$ efficiently.
- **High-variance** (wide) proposals explore $P(x)$ well but have **low acceptance rates**.

1.2 Autocorrelation

Due to the design of the sampling process, MCMC chains will show **autocorrelation**, which means temporally close samples feature high correlation. We want **low autocorrelation** for a better **effective sample size**.

These concerns can be monitored using the autocorrelation function $R_x(k)$ (Equation 1) and the sample size inflation factor s_x (Equation 2). The effective sample size is computed as $\frac{n}{s_x}$, which is increased with low autocorrelation but decreased with high autocorrelation.

$$R_x(k) = \frac{\sum_{t=1}^{n-k} (x_t - \bar{x})(x_{t+k} - \bar{x})}{\sum_{t=1}^{n-k} (x_t - \bar{x})^2} \tag{1}$$

$$s_x = \frac{1 + R_x(1)}{1 - R_x(1)} \tag{2}$$

1.3 Stopping Burn-In

Though there are no hard rules for stopping burn-in, we can make use of meaningful heuristics to decide that sampling chains have converged.

1. **Monitor samples directly**: If chains look well-mixed, it is likely that convergence has occurred. See Figure 1.

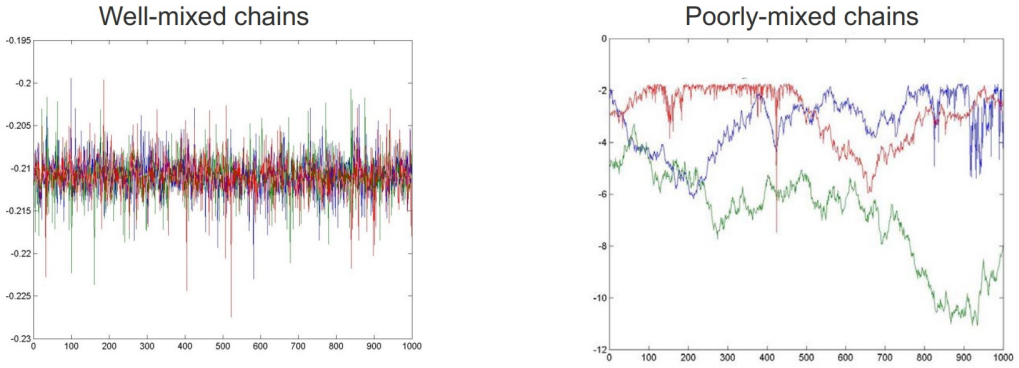

Figure 1: **Left:** Well-mixed chains indicate convergence has occurred. **Right:** Poorly mixed chains indicate that convergence has yet to occur. Figures from lecture slides.

2. **Monitor log-likelihood**: If data are high-dimensional (as is often the case), the complete log-likelihood can be visualized instead. Convergence is often suggested by the log-likelihood first climbing, then eventually plateauing. See Figure 2.

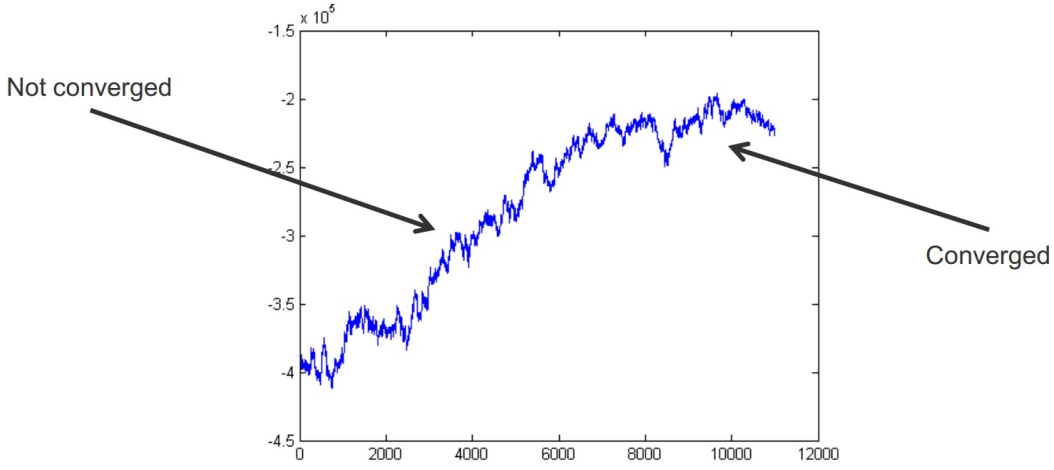

Figure 2: Visualization of log-likelihood often useful for high-dimensional samples. Plateauing behavior suggests convergence. Figure from lecture slides.

2 Hamiltonian Monte Carlo

2.1 Motivation

In Random walk in MCMC, the proposal is at risk of slipping into low probability region and therefore $Q(x_{new}|x_{old})$ may have low acceptance rate. Also, when the variance of proposal is small, samples may be highly correlated with each other.

We want to a way to control the direction of the next sample so we can avoid low acceptance rate and keep low correlation. HMC is using gradient information to control the direction.

2.2 Hamiltonians and Operators

Hamiltonians are a way of describing the state of subatomic particles. A Hamiltonian is a function of two vectors: a position vector, x, and a momentum vector, p. We say that

$$H(p, x) = K(p) + U(x)$$

where H is the Hamiltonian, K is a function that yields kinetic energy given momentum, and U is a function that yields potential energy given position.

These notions allow us to explore Hamiltonian dynamics. In general, information about the Hamiltonian at a specific time allows us to reason about future time steps. Specifically, we say that

$$\frac{dx_i}{dt} = \frac{dH}{dp_i}$$

$$\frac{dp_i}{dt} = \frac{dH}{dx_i}$$

3 Computing Updates

3.1 Euler's Method

We can use Euler's method to compute a sequence of samples. Given Euler's equation:

$$p_i(t+\epsilon) = p_i(t) + \epsilon \frac{dp_i}{dt}(t) = p_i(t) - \epsilon \frac{dU}{dq_i}(q(t)) \tag{3}$$

$$q_i(t+\epsilon) = q_i(t) + \epsilon \frac{dq_i}{dt}(t) = q_i(t) + \epsilon \frac{p_i(t)}{m_i} \tag{4}$$

We can use p_i to compute $q_t, q_{t+1}, ...q_N$.

3.1.1 Shortcomings of Euler's Method

Euler's method has one main shortcoming. This is the fact that an update can be expressed as

$$\begin{pmatrix} p(t+\epsilon) \\ q(t+\epsilon) \end{pmatrix} = \begin{pmatrix} 1 & a \\ b & 1 \end{pmatrix} \begin{pmatrix} p(t) \\ q(t) \end{pmatrix} \tag{5}$$

It can be seen that, depending on the values of a and b, the volume of the space described by $\begin{pmatrix} p(t) \\ q(t) \end{pmatrix}$ can potentially change with repeated updates. As such, Euler's method describes a divergent series. This is undesirable, as the points we sample from it can start to get arbitrarily far from our original series. Furthermore, this is not possible in the original physics interpretation of a Hamiltonian, which can interfere with intuitive understanding of the system.

3.2 Leapfrog Method

In the leapfrog method, we compute the next step slightly differently from before. Now, we compute the following

$$\begin{aligned} p_i(t+\epsilon/2) &= p_i(t) - (\epsilon/2)\frac{dU}{dq_i}(q(t)) \\ q_i(t+\epsilon) &= q_i(t) + \epsilon\frac{p_i(t+\epsilon/2)}{m_i} \\ p_i(t+\epsilon) &= p_i(t+\epsilon/2) - (\epsilon/2)\frac{dU}{dq_i}(q(t+\epsilon)) \end{aligned}$$

3.2.1 Compared to Euler's Method

The leapfrog method solves Euler's Method's problem of being a divergent series. When applying a similar process to the one used to generate equation 5, we instead get the following:

$$\begin{pmatrix} p(t+\epsilon) \\ q(t+\epsilon) \end{pmatrix} = \begin{pmatrix} 1 & a \\ 0 & 1 \end{pmatrix} \begin{pmatrix} p(t+\epsilon/2) \\ q(t+\epsilon) \end{pmatrix} \tag{6}$$

Notably, the matrix $\begin{pmatrix} 1 & a \\ 0 & 1 \end{pmatrix}$ corresponds to a shear transformation, meaning that it is volume-preserving. This transformation can also seen to be volume-preserving because its determinant is 1. As such, this series is not divergent, and is better behaved.

3.3 Langevin MCMC

Main idea is to do one step Leapfrog. Compared with Leapfrog method, Langevin MCMC algorithm can be aggragated in a more compact form.

3.3.1 Pros against Leapfrog method

Given a large datasets, stocahstic Langevin has control over computing the gradient using subsets of samples instead of whole gradient.

4 More Ideas on Approximate Inference

Previously, we have seen a lot of methods on computing $Q(\cdot)$, the proposed target distribution. For example, it can be defined as a Gaussian, or a moving Gaussian depends on your previous sample. To make it fancier, we can draw a sample from the Gaussian and then compute the gradient to find our next sample.

To compute the building block of a good proposal $Q(\cdot)$, we have to investigate the original target distribution $P(\cdot)$. In MC, it is used to calculate the weight r, but not used to compute $Q(\cdot)$; in MCMC, it is used to compute A; in HMC, we use it to compute the gradient (write as $\frac{\partial U}{\partial q}$ in equation, where U is original distribution and q is our target x).

Now, we try to make a better proposal of $Q(x_{new}|x_{old})$ directly from the knowledge of P.

4.1 Variational MCMC

Recall Jensen's inequality:

$$log(p(x|\theta)) \geq E_{q(z)}[log(p(x|\theta))] - E_{q(z)}[log(q(z)]$$

Where we are interested in posterior of parameters $p(\theta|x)$ but transform to find the expectation of hidden variable $q(z|\lambda)$, where λ is the variational parameter, and use it as a lower bound to approach P. However, the calculation of Q could still be hard. So we ask: if we can introduce another variational parameter ξ to replace $p(x|\theta)$ as $p(x|\theta, \xi)$, so to make the computing of Q easier? The idea here is to find a middle place between $q(z|\lambda)$ and $p(x|\theta, \xi)$ as $P^{est}(\theta|x, \lambda, \xi)$, and let this estimation be as close as to $P(\theta|x)$.

$$Q(x_{new}|x_{old}) = P^{est}(\theta|x, \lambda, \xi) \leq P(\theta|x)$$

In this process, finding $P^{est}(\theta|x, \lambda, \xi)$ can be taken as a optimization problem, and approaching P with P^{est} can be done with MCMC. Generally, this method helps increasing mixing rate, and acceptance rate in lower dimension; it also decrease the convergence time and get uncorrelated information. Also, compare to Hamiltonian MCMC which also combined optimization and variatioanl inference, this method is more general.

4.2 Sequential MC with Particle Filters

Recall weighted resampling:
1. Draw N samples from proposal Q;
2. Get samples weight $w^m = \frac{P(x^m)|Q(x^m)}{\sum_l P(x^l)|Q(x^l)} = \frac{r^m}{\sum_m r^m}$;
3. Sub-sample those N samples w.r.t weights.

In this way, you can have the samples' distribution as close as to the original distribution.

Similarly, this idea can be adapted by sequential sampling. Suppose we have a HMM but with very complex transition probability, if we have $P(X_t|\mathbf{Y}_{1:t})$, how can we show $P(X_t + 1|\mathbf{Y}_{1:t+1})$? Observe the expression of $P(X_t|\mathbf{Y}_{1:t})$:

$$P(X_t|\mathbf{Y}_{1:t}) = P(X_t|\mathbf{Y}_{1:t-1}, Y_t) = \frac{P(X_t|\mathbf{Y}_{1:t-1})P(Y_t|X_t)}{\int P(X_t|\mathbf{Y}_{1:t-1})P(Y_t|X_t)dX_t}$$

where $P(X_t|\mathbf{Y}_{1:t-1})$ is the sampling step and $\frac{P(Y_t|X_t)}{\int P(X_t|\mathbf{Y}_{1:t-1})P(Y_t|X_t)dX_t}$ is the weight of current sampling. So $P(X_t|\mathbf{Y}_{1:t})$ can be sampled by:

$$X_t^m \sim P(X_t|\mathbf{Y}_{1:t-1}), w_t^m = \frac{P(Y_t|X_t^m)}{\sum_m P(Y_t|X_t^m)}$$

Thus, we can iteratively do:

$$P(X_{t+1}|\mathbf{Y}_{1:t}) = \int P(X_t|\mathbf{Y}_{1:t})P(X_{t+1}|X_t)dX_t = \sum_m w_t^m P(X_{t+1}|X_t^m)$$

$$P(X_t + 1|\mathbf{Y}_{1:t+1}) = \frac{P(X_{t+1}|\mathbf{Y}_{1:t})P(Y_{t+1}|X_{t+1})}{\int P(X_{t+1}|\mathbf{Y}_{1:t})P(Y_{t+1}|X_{t+1})dX_{t+1}}$$

$$\Rightarrow X_{t+1}^m \sim P(X_{t+1}|\mathbf{Y}_{1:t}), w_{t+1}^m = \frac{P(Y_{t+1}|X_{t+1}^m)}{\sum_m P(Y_{t+1}|X_{t+1}^m)}$$

e.g switching SSM
Each hidden state $S_t = \{X_t^1, X_t^2\}$. Knowing $S_t^m \sim P(S_t|S_{t-1}^m)$, approximate $P(X_t|Y_{1:t}, S_{1:t}^m)$.