6.882: Bayesian Modeling and Inference **Lecture Date: March 1, 2018**

Markov Chain Monte Carlo

Lecturers: Christine Konicki, Lawrence Li, Sanjeev Murty — Scribes: Lorenzo Masoero, Cristina Mata, Andrew Xia

1 Administrativia

Here are some due dates to keep in mind as you work on your project.

- March 16 Pre-proposal (Consists of a title and abstract.)
- March 23 Proposal
- April 13 Progress Report
- May 11 Final Report

Note that there is **no class** on April 5 and May 17.

2 Markov Chain Monte Carlo

Assume we are interested in some complicated, possibly high dimensional **target distribution** $P(\cdot)$, known only up to a normalization constant. In Bayesian inference, this role is played by the posterior distribution over the parameters conditionally on the observed data[1]. We are often interested in computing functionals with respect to such posterior distribution which is, however, typically intractable and not available in closed form due to the normalization constant. Monte Carlo methods allow us to approximate such quantities by means of sample averages. Hence, we are interested in the following two objectives:

1. Obtain samples $\{x^{(t)}\}_t$ from our target distribution $P(\cdot)$
2. Use the samples to approximate the integral $\mathbf{E}_{P(.)}[\phi(x)] \approx \frac{1}{R}\sum\limits_{k=1}^{R} \phi(x^{(t)})$

During our last class we talked about two methods to obtain samples from an arbitrary target distribution $P(\cdot)$: importance sampling and rejection sampling. We also discussed some of their limitations:

- For rejection sampling, we need to find a scalar c such that $P(\cdot) \leq cQ(\cdot)$. The rejection rate (i.e. the fraction of points sampled from the proposal which are discarded and not included in the actual sample) may grow exponentially with the dimension of the problem.

[1] By Bayes' rule, the posterior distribution over the parameters x given data y is $p(x|y) = \frac{p(y|x)p(x)}{\int p(y|x)p(x)dx} \propto p(y|x)p(x)$

- In importance sampling, the "importance weights" we calculate are skewed towards areas where $Q(\cdot)$ differs greatly from $P(\cdot)$. Moreover, if the proposal $Q(\cdot)$ puts little mass over regions in which $P(\cdot)$ puts consistent mass, it might take arbitrarily long to sample points from such regions, causing biases in the estimates we produce.

In this lecture, we introduce Markov Chain Monte Carlo [MCMC] methods. MCMC is a rich class of algorithms which produce samples from a target distribution $P(\cdot)$ by means of a Markov Chain that has $P(\cdot)$ as its stationary distribution.

2.1 Markov Chains, some definitions

We introduce a number of elementary definitions and properties of Markov Chains, which will be needed to state the main result of this section.

Definition 1 (Markov Chain)**.** A sequence of random variables $\{X_t\}_{t\geq 1}$ forms a discrete time Markov Chain if it satisfies the Markov property,

$$Pr(X_{n+1} = x | X_1 = x_1, \ldots, X_n = x_n) = Pr(X_{n+1} = x | X_n = x_n)$$

We call the state space the set of values which the random variables $\{X_t\}$ can attain, and denote it via $\mathcal{X}$.

Definition 2 (Finite chain)**.** A Markov Chain $\{X_t\}_t$ is **finite** if the state space is finite. For a finite Markov chain, we can express transition probabilities of moving from a state to another by means of a row-stochastic transition matrix $T \in [0,1]^{s\times s}$, where $s = |\mathcal{X}|$ is the cardinality of the state space.

Definition 3 (Irreducible chain)**.** A Markov chain X is irreducible if $\forall\ i, j \in \mathcal{X}$, there exists some path $i \to j$ which has positive probability and takes places in a finite number of transition steps

Definition 4 (Aperiodic)**.** A state i has period k if returning to state i from the current state i must occur in multiples of k. Formally, $k = gcd\{n >=: Pr(X_n = i | X_0 = i) > 0\}$. If $k = 1$, then we say state i is aperiodic. If all states are aperiodic, then the Markov chain is aperiodic.

Definition 5 (Stationary Distribution)**.** Given a transition probability matrix T for a Markov Chain $\{X_t\}_t$, π is a stationary distribution for T if it solves the equation $\pi = T\pi$.

Remark: In Bayesian inference, the target distribution π will often be the posterior distribution of the parameters given the data, known up to a normalization constant.

2.2 Convergence in Markov Chain

The intuition behind the MCMC method is that if we can construct a Markov Chain with the property that its unique stationary distribution is the target distribution of interest, we can run such a Markov Chain, wait until it has converged to its stationary distribution, and then treat the samples produced by the chain as draws from the target distribution.

The reason for the success of MCMC is that (perhaps surprisingly) it is not particularly hard to design a Markov Chain which has the desired complicated target distribution, known up to a normalizing constant, as the desired stationary distribution.

Now, we can present the following theorem:

Theorem 6. *If a Markov Chain is finite, irreducible, and aperiodic, then there exists a unique stationary distribution π, such that the Markov Chain will converge to π.*

A sufficient condition to verify that a given distribution is a stationary distribution of a transition matrix, is detailed balance, which we introduce below.

Definition 7 (Detailed Balance)**.** Let T be the transition probability matrix for a Markov Chain, where $T_{i\to j} = Pr(X_t = j|X_{t-1} = i)$. Let π_i, π_j be the equilibrium probabilities of state i, j respectively.

The chain satisfies the *detailed balance* condition if $\forall\, i, j \in \mathcal{X}$,

$$\pi_i T_{i\to j} = \pi_j T_{j\to i}$$

where the above equation is known as the *detailed balance equation.* If a Markov Chain is in "detailed balance", it is also said to be reversible.

Intuitively, $\pi_i T_{i\to j}$ represents the amount of probability that flows from edge $i \to j$ in a single time step. If there is no net flow in probability among $i \leftrightarrow j$, then the chain is in the stationary distribution.

We emphasize that the detailed balance condition is *sufficient but not necessary* to verify that π is the stationary distribution for T. Intuitively, for stationarity, we just need the *net probability flow* out of state i to be equivalent to the net probability flow in, without the stricter requirement that every edge is balanced. A stationary distribution needs to solve the system $\pi = T\pi$. A stationary distribution can be thought of as "global balance," while detailed balance requires a stricter notion of "local balance."

3 Metropolis Hastings

The Metropolis Hastings [MH] algorithm is an important MCMC method to produce samples from a desired target distribution $P(\cdot)$. It produces a Markov Chain in which each value $X_t = x$ of the Markov Chain is obtained combining the two following steps:

1. **proposal step**: conditioned on the current state x of the Markov Chain, sample a candidate value.

   $$x' \sim Q(x'; x)$$

   where $Q(\cdot;\cdot)$ is a suitable **proposal** distribution, that takes as input the current state of the chain x and proposes a new, but not necessarily different, state x'

2. **acceptance step**: compute the acceptance probability

   $$A_{x\to x'} = \Pr(\text{accept } x'|x) = \min\left\{1, \frac{P(x')}{P(x)}\frac{Q(x;x')}{Q(x';x)}\right\}$$

   and let the new state be x' with probability $A_{x\to x'}$, otherwise let the new state be x

It is easy to show that the procedure described above produces a Markov Chain whose stationary distribution is the desired target distribution $P(\cdot)$. Moreover, under mild assumptions on $Q(\cdot;\cdot)$ and $P(\cdot)$, such stationary distribution is unique.

Theorem 8 (Correctness of the MH algorithm)**.** *The MH algorithm produces a Markov Chain whose stationary distribution is given by the target $P(\cdot)$.*

Proof. We restrict our attention to the case in which the state space is finite, i.e. we express the proposal distribution $Q(\cdot;\cdot)$ by means of a row-stochastic matrix. We show that the transition probability T^{MH}

induced by MH is in detailed balance with respect to the target distribution $P(\cdot)$, which is a sufficient condition to ensure that $P(\cdot)$ is the stationary distribution of the Markov Chain induced by MH.

From the algorithm, it follows that

$$T^{MH}_{i\to j} = \begin{cases} Q_{i,j}A_{i\to j} & \text{if } j \neq i \\ 1 - \sum_{j\neq i} T^{MH}_{i\to j} & \text{if } j = i \end{cases}$$

Without loss of generality, assume that it holds $P^*(j)Q_{j,i} \geq P^*(i)Q_{i,j}$. This implies that $A_{i\to j} = 1$, $A_{j\to i} = \frac{P(i)Q_{i\to j}}{P(j)Q_{j\to i}}$. Hence,

$$P(i)T^{MH}_{i\to j} = P(i)Q_{i,j} = P(i)Q_{i,j}\frac{P(j)Q_{j,i}}{P(j)Q_{j,i}} = \left(\frac{P(i)Q_{i,j}}{P(j)Q_{j,i}}\right)Q_{j,i}P(j) = A_{j\to i}Q_{j,i} = T^{MH}_{j\to i}P(j)$$

This proves that T^{MH} and $P(\cdot)$ are in detailed balance, i.e. $P = T^{MH}P$. Hence, $P(\cdot)$is the stationary distribution of the chain induced by T^{MH}. Moreover, if the chain is finite, irreducible, and aperiodic, $P(\cdot)$ is the unique stationary distribution of the chain. □

3.1 Random walk Metropolis Hastings

Random Walk Metropolis Hastings [RWMH] algorithms are the class of MH algorithms in which the proposal distribution $Q(\cdot;\cdot)$ is *symmetric* around the current value of the Markov Chain, i.e. the proposed state at time $t+1$ conditionally on the chain being in state $x^{(t)} = x$ at step t is given by

$$x' = x + \epsilon, \qquad \epsilon \sim g$$

for some distribution g symmetric around 0. Notice that under such a proposal, the acceptance probability simplifies:

$$A_{x\to x'} = \Pr(\text{accept } x'|x) = \min\left\{1, \frac{P(x')}{P(x)}\frac{g(x'-x)}{g(x-x')}\right\} = \min\left\{1, \frac{P(x')}{P(x)}\right\}$$

In particular, notice that this implies that whenever a state x' which is more likely than the current state x is proposed, it will be accepted with probability 1.
For a general state $\mathcal{X} \subset \mathbb{R}^d$, we can summarize the RWMH for a symmetric proposal $g : \mathbb{R}^d \to \mathbb{R}_+$ symmetric around 0:

1. initialize the chain at $\mathbf{x}^{(0)} = \begin{bmatrix} x_1^{(0)} & \dots & x_d^{(0)} \end{bmatrix}$
2. for $t \geq 1$:
   (a) draw $\epsilon^{(t)} \sim g$, propose $\mathbf{x'} = \mathbf{x}^{(t-1)} + \epsilon^{(t)}$
   (b) compute $A_{\mathbf{x}^{t-1}\to\mathbf{x'}} = \min\left\{1, \frac{P(\mathbf{x'})}{P(\mathbf{x^{(t-1)}})})\right\}$
   (c) set $\mathbf{x}^{(t)} = \begin{cases} \mathbf{x'} & \text{with probability } A_{\mathbf{x}^{t-1}\to\mathbf{x'}} \\ \mathbf{x}^{(t-1)} & \text{with probability } 1 - A_{\mathbf{x}^{t-1}\to\mathbf{x'}} \end{cases}$

3.1.1 Mixing

A crucial feature of a good MCMC method is its ability to "mix", i.e. to efficiently explore the state space and converge fast to the stationary distribution of interest. What can we say about the mixing properties of RWMH?

The unidimensional case: As a simple analysis, consider the case in which the state space has dimension one, i.e. $d = 1$. After R steps, the expected distance covered on the state space is about $\sqrt{R}\epsilon$. This means that if the largest length scale of the state space is L, we need $R \simeq \frac{L}{\epsilon^2 f}$ steps before we obtain a sample that is roughly independent of the initial condition, where $f \in [0, 1]$ is the average fraction of proposed states which are accepted.

From this fact, we can derive a useful rule of thumb that provides a lower bound on the number of iterations needed to obtain an independent sample from the original state when RWMH is used: if the largest length scale of probable states is L, we must run RWMH for at least $R \simeq (L/\epsilon)^2$ to obtain an independent sample.

The value of L is often unknown, hence understanding how many iterations are needed to obtain samples from the stationary distribution is a tricky problem. A common strategy used in practice, is to *tune* the step size ϵ to target a desired acceptance rate f.

The multidimensional case: How does the problem of mixing change when we move to higher dimensions? Again, we consider a simple case to build intuition on the general behavior of the algorithm. Assume that the target distribution is separable along the axes, with standard deviation σ_l along each axis $l = 1, \dots, d$, and the proposal is a spherical Gaussian with standard deviation ϵ along each direction. Denote by $\sigma_{\max} \geq \sigma_{\min} > 0$ the largest and the smallest standard deviations, and assume that we have picked a value ϵ such that the acceptance probability α is close to 1.

Under the assumptions stated above, we can consider the state in each dimension as evolving independently from the others, i.e. a d-dimensional random walk of step size ϵ. Hence, the largest lengthscale $\sigma_{\max}$ determines the number of iterations needed in order to obtain an independent sample from the original state, i.e we need $R \simeq \sigma_{\max}/\epsilon^2$ iterations in order to obtain a sample independent from the original state.

We notice two things:

+ the lower bound we have found on R does not directly depend on the dimensionality d. This is in contrast with rejection and importance sampling, in which we had exponential dependence on the dimensionality d.

- if we want to minimize the number of iterations $R = R(\epsilon)$ needed in order to obtain an independent sample as a function of the stepsize, we would like to make ϵ as big as possible. However, we also have to preserve the property that the acceptance rate is close to one, and we cannot hope to do so unless $\epsilon \approx \sigma_{\min}$[2]. In practice, we cannot hope for a better time than $R \simeq (\sigma_{\max}/\sigma_{\min})^2$.

See MacKay (1998) for extensive discussion on these problems.

3.2 Gibbs Sampling

Gibbs Sampling is a special case of MH. If our target is high dimensional, it might be complicated to come up with a suitable proposal distribution Q. The idea behind Gibbs sampling is that sometimes we can make our life easier by breaking the sampling process into a smaller chunks, sampling one variable at a time. Specifically, Gibbs sampling uses conditional distributions as proposals: let $P(\cdot) : \mathbb{R}^n \to \mathbb{R}$ denote our target distribution, from which we want to get samples $\{x_1^{(t)}, x_2^{(t)}, ..., x_n^{(t)}\}_{t \geq 1}$. Assume we have a set of current samples $\{x_1^{(t)}, x_2^{(t)}, ..., x_n^{(t)}\}$. To obtain the next set of samples, at timestep $t+1$, we do the following: for a given index $i \in [n] := \{1, \dots, n\}$, we fix $x_{-i}^{(t+1)}$ to be the set of most recent samples produced by the algorithm, excluding coordinate i. For example, if the index i was chosen sequentially, we would have

[2] In special cases where the second smallest lengthscale $\sigma_{n-1} \gg \sigma_n$, the optimal ϵ might be closer to σ_{n-1} rather than σ_n.

$x_{-i}^{(t+1)} := \{\underbrace{x_1^{(t+1)}, x_2^{(t+1)}, \ldots, x_{i-1}^{(t+1)}}_{\text{“fresh” samples from } t+1}, \underbrace{x_{i+1}^{(t)}, \ldots, x_n^{(t)}}_{\text{“old” samples from } t}\}$. Then we sample from the complete conditional:

$$x_i^{t+1} \sim Q(x_i; x_{-i}^{(t+1)}) = P(\cdot|x_1^{(t+1)}, x_2^{(t+1)}, ..., x_{i-1}^{(t+1)}, x_{i+1}^{(t)}, x_{i+1}^{(t)}..., x_n^{(t)})$$

If our proposed sample is x' and x is the previous sample, then our acceptance probability takes a particular form:

$$A_{x\to x'} = \min\left\{1, \frac{P(x')}{P(x)}\frac{Q(x;x')}{Q(x';x)}\right\} = \min\left\{1, \frac{P(x_i^{(t+1)}|x_{-i}^{(t+1)})P(x_{-i}^{(t+1)})}{P(x_i^{(t)}|x_{-i}^{(t+1)})P(x_{-i}^{(t+1)})}\frac{P(x_i^{(t)}|x_{-i}^{(t+1)})}{P(x_i^{(t+1)}|x_{-i}^{(t+1)})}\right\} = 1$$

since $Q(x;x') = P(x_i|x'_{-i}) = P(x_i|x_{-i}^{(t+1)})$ and $Q(x';x) = P(x'_i|x_{-i}^{(t+1)})$. When calculating the acceptance probability we need to keep the x_{-i} constant, i.e., when we generate an instance x' from the distribution $P(.|x_{-i})$ we must keep the current values of the other variables constant. Note that when $x'_{-i} = x_{-i}, A_{x'\to x} = 1$.

Why might we want an acceptance rate of 1? Recall that one of the problems with rejection sampling is that the algorithm might take a long time to produce a novel sample at a given timestep if the acceptance rate is low. A high acceptance rate implies that Gibbs sampling does not get “stuck” sampling on a certain timestep the way that rejection sampling does. The tradeoff we are making is that we might accept samples that are very close to our current state, which means we must take many samples before we get an independent sample, and we might incur in “random-walk” behaviors.

If the reader is interested in seeing a visualization of Gibbs Sampling, the following repository contains code necessary to do so: https://github.com/16lawrencel/mcmc-viz

3.2.1 Example: Latent Dirichlet Allocation

Recall that Latent Dirichlet Allocation (LDA) is a generative probabilistic model for collections of discrete data (Blei et al. (2003)).

The classical application of LDA is topic modeling: there, the data is a collection of documents, where each document is modeled as a collection of word-counts of a shared alphabet across all documents. Each document is modeled as a mixture of underlying latent topics, and the goal of LDA is to learn the set of latent topics, the word probabilities associated to each topic, and the “mixing weights” of each document, i.e. the topic-document weights. For each word $w_{d,n}$, the n^{th} word of document d, LDA seeks to sample a topic $z_{d,n} \sim \text{Multinomial}(\theta_d)$ and to sample a word $x_{d,n} \sim \text{Multinomial}(\phi_{z_{d,n}})$ where the parameters of these multinomials have Dirichlet priors (Poreus et al. (2008)). Specifically we are trying to find

$$p(\theta_d|\alpha, \beta, \theta_{-d}, z_{d,n}, w_{d,n})$$

where $\alpha, \beta, \theta_{-d}$, and $z_{d,n}$ are the latent variables, $w_{d,n}$ are the observed variables.

In order to apply Gibbs sampling to estimate θ_d, we need to use the concept of a Markov Blanket (Blei (2015), Ermon (2017)). The Markov Blanket of a node x, denoted $MB(x)$, is the set of nodes in a network that is comprised of x’s parents, children, and the children’s parents. This has a relationship to conditional independence, which we won’t cover here, but it suffices to say that in a network with some number of nodes x_i, $p(x_i|x_{-i}) = p(x_i|MB(x_i))$.

Continuing with LDA, it is straightforward to verify that $MB(\theta_d) = \{\alpha, z_{dn}\}$, which we use to get the updated conditional

$$p(\theta_d|\alpha, z_{d,n}) = \alpha p(\theta_d|\alpha) \prod_n p(z_{d,n}|\theta_d)$$

and since $\theta_d \sim \text{Dir}(\alpha)$ and $z_{d,n} \sim \text{Mult}(\theta_d)$, this equals

$$\text{Dir}(\alpha + c_1, \alpha + c_2, ..., \alpha + c_{N_d})$$

and similarly for $z_{d,n}$,

$$\begin{aligned} p(z_{d,n}|\alpha, \beta, \theta_{-d}, z_{d,n}, w_{d,n}) &= p(z_{d,n}|\theta_d, w_{d,n}, \beta) \\ &= \alpha p(z_{d,n}, \theta_d, w_{d,n}, \beta) \\ &= \alpha p(z_{d,n}|\theta_d) p(w_{d,n}|z_{d,n}) \\ &= \theta_{d,z_{d,n}} \beta_{z_{d,n},w_{d,n}} \end{aligned}$$

3.3 Visual Comparison

The following figures compare Random Walk Metropolis Hastings with Gibbs Sampling. The target distribution is a two dimensional multivariate normal with mean $\mu = (0, 0)$ and variance $\Sigma = \begin{pmatrix} 1.25 & 2.75 \\ 2.75 & 9.75 \end{pmatrix}$. Both algorithms are run for 1000 iterations. The target distribution is represented by oval rainbow lines and the x and y axes correspond to Cartesian coordinates. The samples are represented by red dots. RWMH is run with a Gaussian proposal with standard deviation $\epsilon = 0.01$. Under this proposal, one can see that RWMH produces samples that are quite close together and are mostly within a two by two square centered at 0. Gibbs does a better job, exploring the state space more effectively.

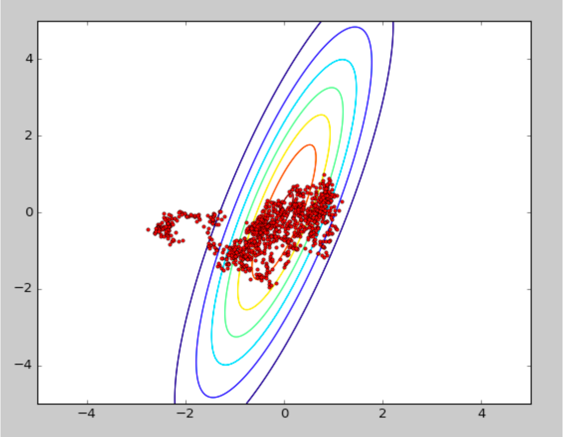

Figure 1: Random Walk Metropolis Hastings

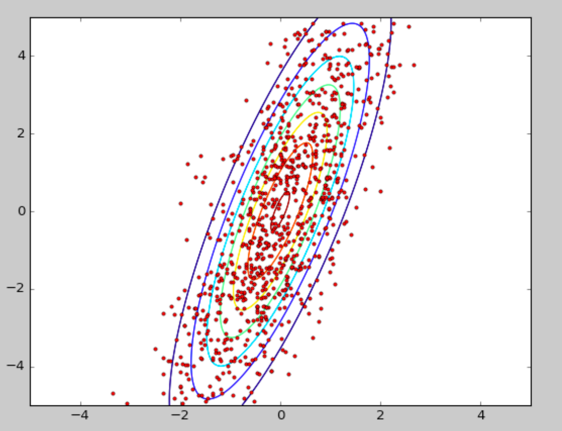

Figure 2: Gibbs Sampling

Remark: a distinctive feature of Gibbs sampling, is that - by sampling one coordinate at the time - they result in horizontal and vertical updates along the x and y axis.

References

Blei, D. (2015). Bayesian mixture models and the Gibbs sampler. http://www.cs.columbia.edu/ blei/fogm/2015F/notes/mixtures-and-gibbs.pdf.

Blei, D., Ng, A., and Jordan, M. (2003). Latent Dirichlet allocation. In *Journal of Machine Learning Research*.

Ermon, S. (2017). Gibbs sampling. https://ermongroup.github.io/cs323-notes/probabilistic/gibbs/.

MacKay, D. J. (1998). Introduction to Monte Carlo methods.

Poreus, I., Newman, D., Ihler, A., Asuncion, A., Smith, P., and Welling, M. (2008). Fast collapsed Gibbs sampling for latent Dirichlet allocation. In *Knowledge Discovery and Data Mining Conference*.