Markov Chain Monte Carlo
(and Bayesian Mixture Models)

David M. Blei
Columbia University

October 14, 2014

¶ We have discussed probabilistic modeling, and have seen how the posterior distribution is the critical quantity for understanding data through a model.

¶ The goal of probabilistic modeling is use domain and data-knowledge to build structured joint distributions, and then to reason about the domain (and exploit our knowledge) through the posterior and posterior predictive distributions.

¶ We have discussed tree propagation, a method for computing posterior marginals of any variables in a tree-shaped graphical model.

¶ In theory, if our graphical model was a tree, we could shade the observations and do useful inferences about the posterior.

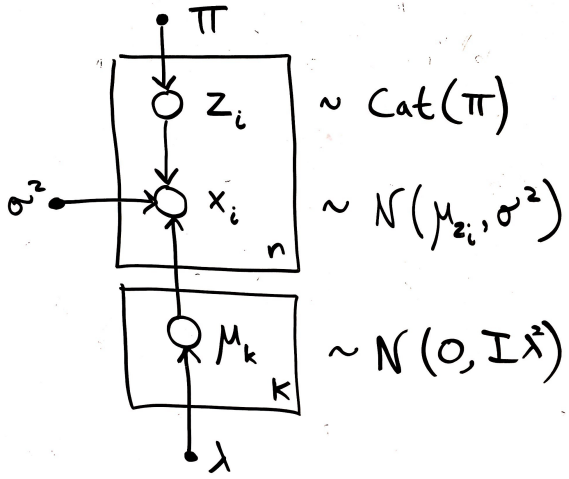
¶ For many interesting models, however, the posterior is not tractable to compute. Either the model is not a tree or the messages are not tractable to compute (because of the form of the potentials). Most modern applications of probabilistic modeling rely on *approximate posterior inference* algorithms.

¶ Thus, before we talk about the building blocks of models, we will talk about an important and general method for approximate posterior inference.

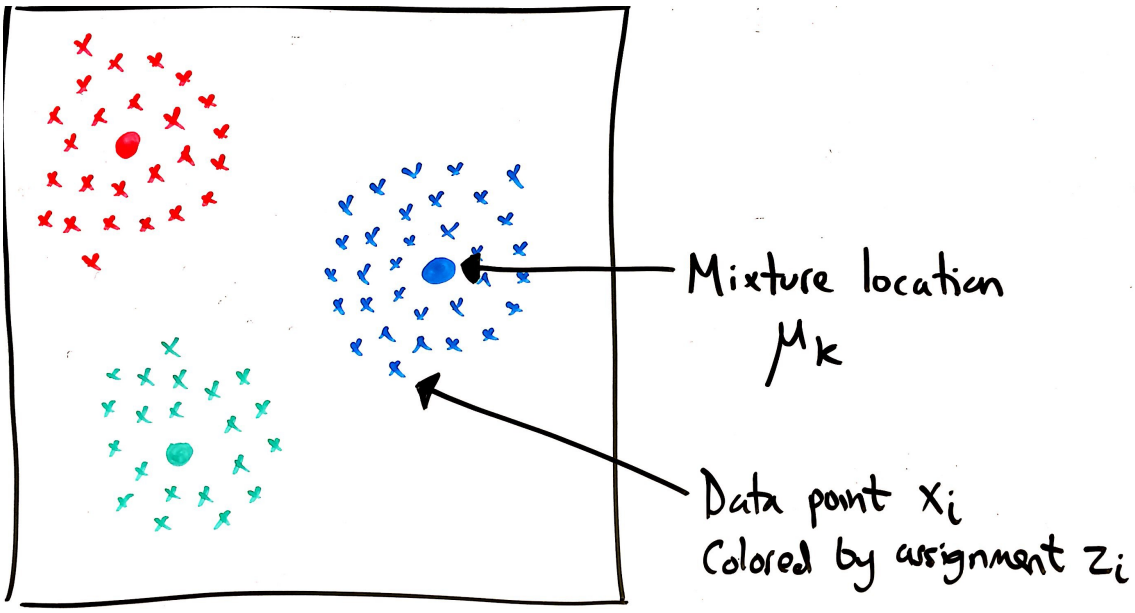
Bayesian mixture of Gaussians

¶ To lock ideas, and to give you a flavor of the simplest interesting probabilistic model, we will first discuss Bayesian mixture models.

¶ Here is the Bayesian mixture of Gaussians model, its graphical model and the generative process.



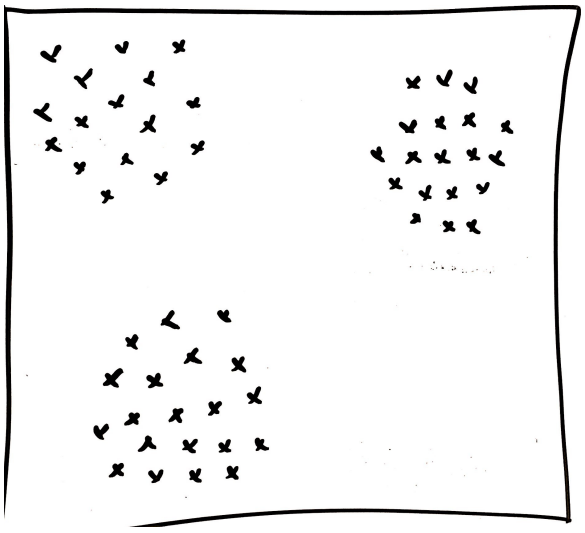
¶ To get a feel for what this model is about, let us generate data from it.



The posterior and posterior predictive distributions

¶ The posterior distribution is a distribution over the latent variables, the cluster locations and the cluster assignments, $p(z_{1:n}, \mu_{1:K} \mid x_{1:n})$.

¶ This gives an understanding of the data (at least, a grouping into K groups).



¶ What is this posterior? The mixture model assumes that each data point came from one of K distributions. However, it is unknown what those distributions are and how the data were assigned to them. The posterior is a conditional distribution over these quantities.

¶ As usual, the posterior also gives a posterior predictive distribution,

$$p(x_{n+1} \mid x_{1:n}) = \mathbb{E}\left[p(x_{n+1} \mid \mu_{1:K})\right]. \tag{1}$$

¶ The expectation is taken over the posterior cluster locations $\mu_{1:K}$. Why? Conditional independence. Inside the expectation we condition on all the observations and latent variables. From Bayes ball we have that

$$x_{n+1} \perp\!\!\!\perp z_i \mid \mu_{1:K}$$

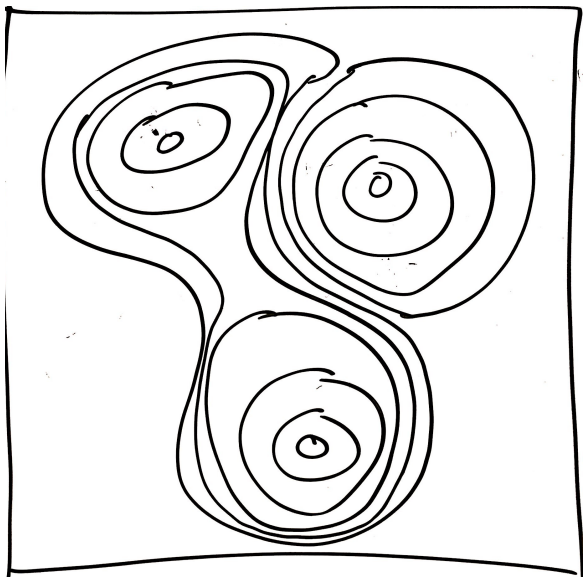
for all $i \in 1, \ldots, n$.

¶ To make notation simpler, let's denote the collection $\mu \triangleq \mu_{1:K}$. The posterior predictive distribution is

$$\begin{aligned} p(x_{n+1} \mid x_{1:n}) &= \int_{\mu} p(x_{n+1} \mid \mu) p(\mu \mid x_{1:n}) d\mu & (2) \\ &= \int_{\mu} \left(\sum_{k=1}^{K} p(z_{n+1} = k) p(x_{n+1} \mid \mu_k) \right) d\mu & (3) \\ &= \sum_{k=1}^{K} p(z_{n+1} = k) \left(\int_{\mu_k} p(x_{n+1} \mid \mu_k) p(\mu_k \mid x_{1:n}) d\mu_k \right) & (4) \end{aligned}$$

¶ What is this? We consider x_{n+1} as coming from each of the possible mixture locations (one through K) and then take a weighted average of its posterior density at each.

¶ This is a multi-modal distribution over the next data point. Here is a picture:



This predictive distribution involves the posterior through $p(\mu_k \mid x_{1:n})$, the posterior distribution of the kth component given the data.

¶ Contrast this with the predictive distribution we might obtain if we used a single Gaussian to model the data. In that case, the mean is at a location where there is very little data.

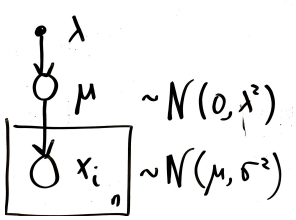
¶ Through the posterior, a mixture model tells us about a grouping of our data, and captures complex predictive distributions of future data.

The posterior is intractable to compute

¶ We cannot compute the posterior exactly. Let's see why.

¶ First an aside: the Gaussian is conjugate to the Gaussian.

¶ Consider a simple model, where we draw a Gaussian mean μ from a Gaussian prior $\mathcal{N}(0, \lambda)$ and then generate n data points from a Gaussian $\mathcal{N}(\mu, \sigma^2)$. (We fix the variance σ^2.) Here is the graphical model



¶ You have seen the beta-Bernoulli; this is another example of a conjugate pair. Given $x_{1:n}$ the posterior distribution of μ is $\mathcal{N}(\hat{\mu}, \hat{\lambda})$, where

$$\hat{\mu} = \left(\frac{n/\sigma^2}{n/\sigma^2 + 1/\lambda^2}\right)\bar{x} \tag{5}$$

$$\hat{\lambda} = \left(n/\sigma^2 + 1/\lambda^2\right)^{-1}, \tag{6}$$

where $\bar{x}$ is the sample mean.

As for the beta-Bernoulli, as n increases the posterior mean approaches the sample mean and the posterior variance approaches zero. (Note: this is the posterior mean and variance of the unknown *mean*. The data variance σ^2 is held fixed in this analysis.)

¶ But now suppose we are working with a mixture of Gaussians. In that case, $p(\mu_1, \ldots, \mu_K \mid x_1, \ldots, x_n)$ is not easy. Suppose the prior proportions π are fixed and $K = 3$,

$$p(\mu_1', \mu_2', \mu_3' \mid x_{1:n}) = \frac{p(\mu_1', \mu_2', \mu_3', x_{1:n})}{\int_{\mu_1}\int_{\mu_2}\int_{\mu_3} p(\mu_1, \mu_2, \mu_3, x_{1:n})}. \tag{7}$$

¶ The numerator is easy,

$$\text{numerator} = p(\mu_1')p(\mu_2')p(\mu_3')\prod_{i=1}^{n} p(x_i \mid \mu_1', \mu_2', \mu_3') \tag{8}$$

where each likelihood term marginalizes out the z_i variable,

$$p(x_i \mid \mu_1', \mu_2', \mu_3') = \sum_{k=1}^{K} \pi_k \, p(x_i \mid \mu_k'). \tag{9}$$

¶ But consider the denominator, which is the marginal probability of the data,

$$p(x_{1:n}) = \int_{\mu_1}\int_{\mu_2}\int_{\mu_3} p(\mu_1)p(\mu_2)p(\mu_3)\prod_{i=1}^{n}\sum_{k=1}^{K}\pi_k \, p(x_i \mid \mu_k). \tag{10}$$

¶ One way to see this is to simply believe me. Another way is to bring the summation to the outside of the integral

$$p(x_{1:n}) = \sum_{z_{1:n}} \int p(\mu_1)p(\mu_2)p(\mu_3) \prod_{i=1}^{n} p(x_i \mid \mu_{z_i}). \tag{11}$$

This can be decomposed by partitioning the data according to $z_{1:n}$,

$$p(x_{1:n}) = \sum_{z_{1:n}} \prod_{k=1}^{3} \left(\int_{\mu_k} p(\mu_k) \prod_{\{i:z_i=k\}} p(x_i \mid \mu_k) \right). \tag{12}$$

Each term in the product is an integral under the conjugate prior, which is an expression we can compute. But there are 3^n different assignments of the data to consider.

¶ To work with Bayesian mixtures of Gaussians (and many other models), we need approximate inference.

¶ Show a mixture model fit to real data, e.g., the image mixture model.

The Gibbs sampler

¶ The main idea behind Gibbs sampling (and all of MCMC) is to approximate a distribution with a set of samples. For example, in the mixture model,

$$p(\mu, z \mid x) \approx \frac{1}{B} \sum_{b=1}^{B} \delta_{(\mu^{(b)}, z^{(b)})}(\mu, z), \tag{13}$$

where we shorthand $\mu = \mu_{1:K}$ and $z = z_{1:n}$.

¶ Let's first discuss Gibbs sampling for mixtures of Gaussians. Then we will see how it generalizes and why it works.

¶ In the Gibbs sampler, we maintain a value for each latent variable. In each iteration, sample from each latent variable conditional on the other latent variables and the observations. I like to call this distribution a *complete conditional*.

¶ **Gibbs sampling for Gaussian mixtures**

Maintain mixture locations $\mu_{1:K}$ and mixture assignments $z_{1:n}$.

Repeat:

1. For each $k \in \{1, \ldots, K\}$:
   Sample $\mu_k \mid \{\mu_{-k}, z_{1:n}, x_{1:n}\}$ from Equation 18
2. For each $i \in \{1, \ldots, n\}$:
   Sample $z_i \mid \{\mu_{1:K}, z_{-i}, x_{1:n}\}$ from Equation 16

¶ Note that within an iteration, when we sample one variable its value changes in what we subsequently condition on. E.g., when we sample μ_k for $k = 1$, this changes what μ_k for the subsequent samples.

¶ The theory around Gibbs sampling says that if we do this many times, the resulting sample will be a sample from the true posterior.

¶ Preview of the theory: The reason is that we have defined a Markov chain whose state space are the latent variables and whose *stationary distribution* is the posterior we care about.

¶ After a long time, a sample of $\mu_{1:K}$ and $z_{1:n}$ is a sample from the posterior. After waiting many long times, we can obtain B samples from the posterior.

Details about the complete conditionals

¶ Let's work out each step of the algorithm, beginning with the complete conditional of z_i. We first look at the graphical model and observe a conditional independence,

$$p(z_i \mid \mu_{1:K}, z_{-i}, x_{1:n}) = p(z_i \mid \mu_{1:K}, x_i). \tag{14}$$

Now we calculate the distribution,

$$\begin{aligned} p(z_i \mid \mu_{1:K}, x_i) &\propto p(z_i)p(x_i \mid \mu_{z_i}) && (15)\\ &= \pi_{z_i}\phi(x_i; \mu_{z_i}, \sigma^2). && (16) \end{aligned}$$

¶ What is this? To keep things simple, assume $\pi_k = 1/K$. Then this is a categorical distribution where the probability of the the kth is proportional to the likelihood of the ith data point under the kth cluster.

¶ Notes: (a) Categorical distributions are easy to sample from. (b) This distribution requires that we know $\mu_{1:K}$.

¶ Now let's derive the complete conditional of μ_k. Again, we observe a conditional independence from the graphical model,

$$p(\mu_k \mid \mu_{-k}, z_{1:n}, x_{1:n}) = p(\mu_k \mid z_{1:n}, x_{1:n}) \tag{17}$$

¶ Here let's calculate the distribution intuitively. If we know the cluster assignments, what is the conditional distribution of μ_k? It is simple a posterior Gaussian, conditional on the data that were assigned to the kth cluster.

¶ Technically: Let z_i be an indicator vector, a K-vector with a single one. Then,

$$\mu_k \mid z_{1:n}, x_{1:n} \sim \mathcal{N}(\hat{\mu}_k, \hat{\lambda}_k) \tag{18}$$

where

$$\hat{\mu}_k = \left(\frac{n_k/\sigma^2}{n_k/\sigma^2 + 1/\lambda^2}\right) \bar{x}_k \tag{19}$$

$$\hat{\lambda} = \left(n_k/\sigma^2 + 1/\lambda^2\right)^{-1}, \tag{20}$$

and

$$n_k = \sum_{i=1}^{n} z_i^k \tag{21}$$

$$\bar{x}_k = \frac{\sum_{i=1}^{n} z_i^k x_i}{n_k}. \tag{22}$$

¶ Important: Conjugacy is helping us, even in a model for which we cannot compute the posterior.

¶ This is an approximate inference algorithm for mixtures of Gaussians. At each iteration, we first sample each mixture assignment from Equation 16 and then sample each mixture location from Equation 18.

¶ Discussion:

- The result of this sampler is one sample from the posterior. To get B samples, we run several times. In practice, we begin from an *initial state* and run for a fixed number of *burn in* iterations. We then continue to run the algorithm, collecting samples a specified *lag*.

  Initialization, burn-in, and lag are important practical issues. There are no good principled solutions, but many ad-hoc ones that work well.

- Notice that conditional independencies in the complete conditionals give us opportunities to parallelize. What can be parallelized here?

- Notice the close relationship to the expectation-maximization (EM) algorithm for mixtures.

  In the EM algorithm we iterate between the E-step and M-step. In the E-step we compute the conditional distribution of each assignment given the locations. This is precisely Equation 16).

  In the M-step we update the locations at maximum likelihood estimates under expected sufficient statistics. As we know, the MLE relates to the Bayesian posterior in Equation 19.

- The theory implies that we need infinite lag time and infinite burn-in. Practical decisions around Gibbs sampling can be difficult to make. (But, happily, in practice it's easy to come up with sensible unjustified choices.) One quantity to monitor is $\log p(\mu_{1:K}^{(t)}, z_{1:n}^{(t)}, x_{1:n})$, i.e., the log joint of the assignments of the latent variables and observations.

  This further relates to EM, which optimizes the conditional expectation (over the mixture assignments z) of this quantity.

The collapsed Gibbs sampler

¶ Sometimes we can integrate out hidden random variables from a complete conditional. This is called *collapsing*.

¶ In the mixture of Gaussians, consider collapsing the mixture locations,

$$p(z_i = k \mid z_{-i}, x_{1:n}) \propto p(z_i = k)\, p(x_i \mid z_{-i}, x_{-i}, z_i = k). \tag{23}$$

¶ The second term is simply a posterior predictive distribution

$$p(x_i \mid z_{-i}, x_{-i}, z_i) = \int_{\mu_k} p(x_i \mid \mu_k)\, p(\mu_k \mid z_{-i}, x_{-i}). \tag{24}$$

¶ **Collapsed Gibbs sampling for Gaussian mixtures**

Maintain mixture assignments $z_{1:n}$ and two derived quantities:

$$\begin{aligned} n_k &\triangleq \sum_{i=1}^n z_i^k \quad \text{(number of items per cluster)} \\ s_k &\triangleq \sum_{i=1}^n z_i^k x_i \quad \text{(cluster sum)} \end{aligned}$$

Repeatedly cycle through each data point $i \in \{1, \ldots, n\}$:

1. "Knock out" x_i from its currently assigned cluster z_i. Update n_k and s_k for its assigned cluster.
2. Sample z_i from Equation 23. The posterior Gaussian $p(\mu_k \mid z_{-i}, x_{-i})$ can be computed from the n_k and s_k.

¶ Collapsed Gibbs sampling can be more expensive at each iteration, but converges faster. Typically, if you can collapse then it is worth it.

Gibbs sampling in general

¶ The ease of implementing a Gibbs sampler depends on how easy it is to compute and sample from the various complete conditionals.

¶ In a graphical model, the complete conditional depends on the *Markov blanket* of the node.

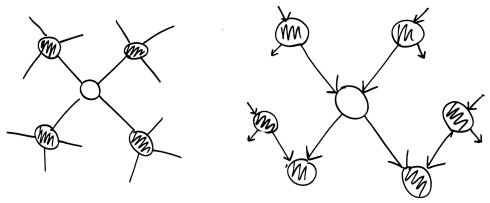
¶ Suppose the nodes are $x_1, \ldots, x_k$ (observed and unobserved).

In an undirected graphical model, the complete conditional only depends on a node's neighbors,

$$p(x_i \mid x_{-i}) = p(x_i \mid x_{\mathcal{N}(i)}). \tag{25}$$

In a directed model, the complete conditional depends on a node's parents, children, and other parents of its children,

¶ We can see these facts from the graphical model and separation / d-separation



¶ Theme: Difficult global computation is made easier in terms of many local computations. Notice how Gibbs sampling is a form of "message passing".

Markov chain Monte Carlo

¶ Markov chain Monte Carlo (MCMC) is a general and powerful methodology for collecting samples from a wide class of distributions.

¶ We will discuss some of the main theory. But note that MCMC is now a subfield of statistics. Entire dissertations and careers are built on developing this class of algorithms.

¶ In contrast to other kinds of sampling—like importance sampling and rejection sampling—MCMC scales with the dimensionality of the hidden variables. MCMC algorithms work in high dimensions, which is important for modern problems.

The Metropolis algorithm (1953)

¶ In this discussion, we will assume a target distribution $p(x)$. We are changing notation slightly and not worrying about observed or unobserved variables. (Recall we made the same jump when discussing exact inference with belief propagation.)

¶ In Metropolis, we only need to compute $p(x)$ up to a constant,

$$p(x) = \tilde{p}(x)/Z,$$

where $\tilde{p}(x)$ is an unnormalized distribution.

¶ Recall that the posterior is easy to compute up to a normalizing constant; it is the normalizing constant that gives us difficulty.

¶ Our plan is to draw a sequence of *states* $x^{(t)}$ such that the final state is a draw from the target distribution. The main idea is to

- Draw a sample x^* from a *proposal distribution* $q(x \mid x^{(t)})$.
- *Accept* the state according to a criterion, which can be random. If accepted, $x^{(t+1)} = x^*$ otherwise $x^{(t+1)} = x^{(t)}$.

¶ Specifically, we assume that $q(x_1 \mid x_2) = q(x_2 \mid x_1)$. We accept the sample x^* with the following probability

$$p\left(\text{accept } x^* \mid x^{(t)}\right) = \min\left(1, \frac{p(x^*)}{p(x^{(t)})}\right) \tag{26}$$

Note that we can compute this acceptance probability without needing to know the normalizing constant.

¶ If we move to a state with higher probability we will always accept it. Sometimes we will move to a state with lower probability.

¶ This has the flavor of a kind of stochastic search. The proposal distribution q governs our policy for taking steps, but we are careful about taking steps that lower our probability.

¶ Let $q(x_1 \mid x_2) > 0$ for all x_1, x_2. Then the marginal distribution of the tth state (under this algorithm) converges to the target distribution,

$$p_{\text{Met}}\left(x^{(t)}\right) \to p(x). \tag{27}$$

Markov chains

¶ The Metropolis algorithm defines a Markov chain on x whose *stationary distribution* is $p(x)$.

¶ To obtain independent samples from $p(x)$ we (a) run the MC for a long time (b) collect samples at some lag.

¶ We have already seen this in the Gibbs sampler. Let's now discuss some of the theory about why Gibbs sampling and Metropolis work.

¶ A first order Markov chain is defined by the conditional independence,

$$p\left(x_{t+1} \mid x_1, \ldots, x_t\right) = p\left(x_{t+1} \mid x_t\right) \tag{28}$$

(Show the graphical model, which looks like a chain. For simplicity let's assume that x is discrete.

¶ A MC is specified by the initial state distribution $p_0(x_0)$ and the transition probabilities from going to each state given the previous one.

¶ A MC is *homogenous* if the transition probabilities are the same across time points, i.e., there is one distribution $p(x_{t+1} \mid x_t)$.

¶ We calculate the marginal probability of a particular variable in the chain. This is done recursively,

$$p_{t+1}\left(x_{t+1}\right) = \sum_{x_t} p\left(x_{t+1} \mid x_t\right) p_t(x_t) \tag{29}$$

This is expressed with the transition probability and the marginal probability of the previous variable. Thus, given the initial distribution $p_0(x_0)$, we can compute any marginal.

¶ Some distributions can be *invariant* or *stationary* with respect to a Markov chain. These distributions leave the corresponding marginal distribution invariant.

¶ Define notation $T(x' \to x)$ to be the probability of moving from x' to x,

$$T(x' \to x) \triangleq p(x \mid x').$$

A distribution $p^*(x)$ is stationary if

$$p^*(x) = \sum_{x'} p^*(x') T(x' \to x) \tag{30}$$

In other words, suppose the marginal of the previous time point is $p^*(\cdot)$. Then running the chain one step yields the same marginal for the current time step.

¶ In general, an MCMC can have zero or more stationary distributions. (Aside: The algorithm that put Google on the map computes the stationary distribution of a random walker on a graph of web pages.)

¶ A sufficient (but not necessary) condition for $p(x)$ being an invariant distribution is that it satisfies *detailed balance*,

$$p^*(x) T(x \to x') = p^*(x') T(x' \to x). \tag{31}$$

This considers the two joint distributions: starting at x and moving to x', starting at x' and moving to x. If these are equal under the MC's transition matrix T then p^* is a stationary distribution of T.

¶ We can see that detail balance is sufficient

$$\begin{aligned}\sum_{x'} p^*(x')T(x' \to x) &= \sum_{x'} p^*(x)T(x \to x') && (32)\\ &= p^*(x)\sum_{x'} T(x \to x') && (33)\\ &= p^*(x) && (34)\end{aligned}$$

We used detailed balance in the first line. This shows that $p^*(x)$ satisfies Equation 30.

¶ One more concept: *ergodicity*. A MC is ergodic if $p_t(x) \to p^*(x)$ regardless of $p_0(x)$. An ergodic Markov chain has only one stationary distribution. In this case, it is called the *equilibrium distribution*.

¶ The plan in designing MCMC algorithms is to create homogenous ergodic Markov chains whose stationary distribution is the target distribution. In Bayesian applications, such as in the mixture model we have discussed, we want a Markov chain whose stationary distribution is the posterior.

¶ Neal (1993) has a Fundamental Theorem. Suppose we have a homogenous MC on a finite state space with transition probabilities $T(x' \to x)$, and we have found a stationary distribution $p^*(x)$. If

$$\nu \triangleq \min_x \min_{x':p^*(x')>0} \frac{T(x \to x')}{p^*(x')} > 0 \qquad (35)$$

then the MC is ergodic.

We know something about the distance to the stationary distribution at time t,

$$|p^*(x) - p_t(x)| \leq (1-\nu)^t. \qquad (36)$$

¶ Intuition: A Markov chain is ergodic if there is a way of getting from any state to any other state (that has non-zero probability under the stationary distribution).

For example, in the Google webgraph, they needed to add an "escape probability". If there are islands of pages then the random walker is not ergodic.

¶ Again, the main goal of MCMC algorithms is to construct a homogenous MC whose stationary distribution is the target distribution. Then run the chain and collect samples.

The computational effort to obtain these samples involves

1. The amount of computation needed to simulate each transition.
2. The time for the chain to converge to its equilibrium distribution. This is called the *burn in*.
3. The number of draws needed to move from one state drawn from p^* to another independent state drawn from p^*. This is called the *lag*.

Items 2 and 3 have no good theoretical answers (though there have been efforts). These remain empirical matters.

Metropolis-Hastings

¶ We will prove that the Metropolis algorithm works and that Gibbs sampling works by describing the algorithm that generalizes both, Metropolis-Hastings.

¶ First, a nuisance. Suppose our state space of interest has K components $x = \{x_1, \ldots, x_K\}$. (For example, consider K nodes in a graphical model.)

- We will consider K transition matrices $B_k(x \to x')$, where each one holds all x_j fixed for $j \neq k$ and only samples x_k.
- To move from $x^{(t)}$ to $x^{(t+1)}$ we iteratively apply each of these transitions. So the transition probabilities for the chain are the product of the $B_k(\cdot)$'s.
- Key: If detailed balance holds for each B_k then it holds for their product.

¶ Our current state is x. At each iteration we draw x' from $B_k(x \to x')$. Note this only changes x_k. We accept the new x' with probability

$$A_k(x', x) = \min\left(1, \frac{p(x')B_k(x' \to x)}{p(x)B_k(x \to x')}\right). \tag{37}$$

Notes

- No need to compute the normalizer
- We do need to compute the transitions, not just sample from them.
- Suppose $B_k(x' \to x) = B_k(x \to x')$. This is Metropolis.

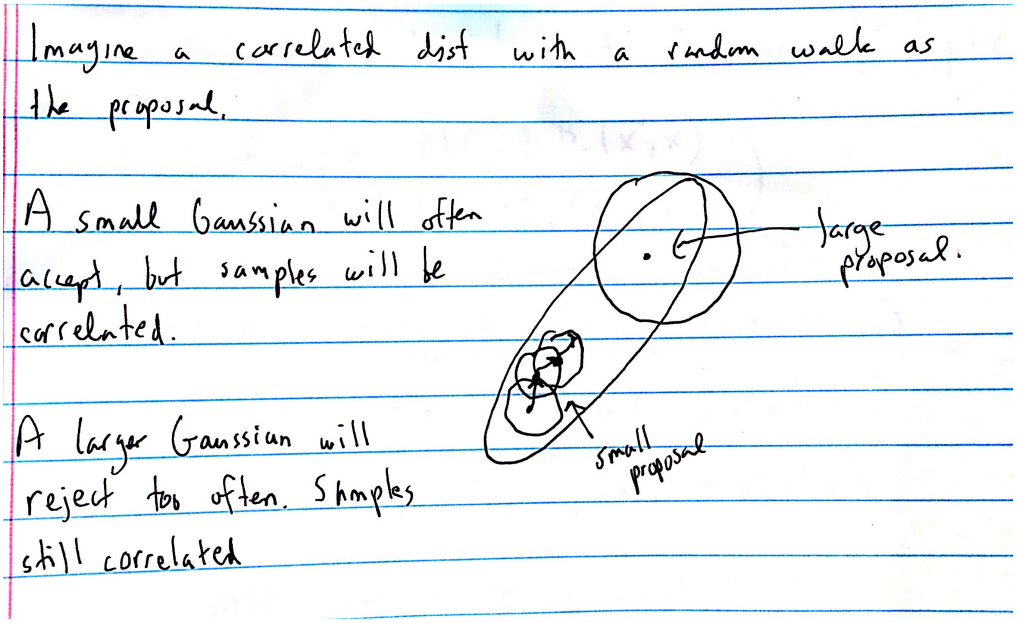
¶ We check detailed balance to confirm that this has the right stationary distribution. Note that the transition for the kth Markov chain is $B_k(x \to x')A_k(x', x)$.

$$\begin{aligned} p(x)B_k(x \to x')A_k(x', x) &= p(x)B_k(x \to x') \min\left(1, \frac{p(x')B_k(x' \to x)}{p(x)B_k(x \to x')}\right) \\ &= \min\left(p(x)B_k(x \to x'), p(x')B_k(x' \to x)\right) \\ &= \min\left(p(x')B_k(x' \to x), p(x)B_k(x \to x')\right) \\ &= p(x')B_k(x' \to x)A_k(x, x') \end{aligned}$$

This means that MH has the right stationary distribution. (It also means that the Metropolis algorithm has the right stationary distribution.)

¶ It is up to us to design the proposal distribution. In practice, the key to fast mixing is to trade off the *rejection rate* and the *amount of movement* in the proposal.

Both have to do with the amount of dependence between successive samples. They are highly correlated if we reject a lot or we don't move much.



The Gibbs sampler

¶ We are finally ready to show that the Gibbs sampler is a valid MCMC algorithm.

¶ Again, let $x = \{x_1, \ldots, x_K\}$. For each iteration we sample from $p(x_k \mid x_{-k})$. Fact: This is a Markov chain whose stationary distribution is $p(x)$.

¶ To see that this works, set B_k to change x_k according to the complete conditional. The acceptance probability is

$$A_k(x', x) = \min\left(1, \frac{p(x')\,p(x_k \mid x'_{-k})}{p(x)\,p(x'_k \mid x_{-k})}\right) \tag{38}$$

Unpack the second term

$$\frac{p(x')\,p(x'_k \mid x_{-k})}{p(x)\,p(x_k \mid x'_{-k})} = \frac{p(x'_{-k})\,p(x'_k \mid x'_{-k})\,p(x_k \mid x'_{-k})}{p(x_{-k})\,p(x_k \mid x_{-k})\,p(x'_k \mid x_{-k})} \tag{39}$$

Note that $x_{-k} = x'_{-k}$. Thus, the acceptance probability is equal to one.

¶ Gibbs sampling is a Metropolis-Hastings algorithm that always accepts.

¶ Some notes:

- Gibbs is usually a good first attempt at approximate inference. It often works well, especially if you can collapse variables.
- That said, designing specialized proposal distributions in an MH context can lead to more efficient samplers. But this requires more work and experimentation.
- In some models MH is required because we cannot compute the complete conditionals. MH inside Gibbs is also a good option, where some B_k are Gibbs steps and others are MH steps.
- An interest research area is designing better generic MCMC algorithms, those that do not require reasoning about the model. Later this semester, we will have a guest lecture from Bob Carpenter about Stan. Stan implements efficient and generic sampling, which does not even require specifying the complete conditionals.

Rao-Blackwellization and the collapsed Gibbs sampler

¶ Why does collapsing help? The reason is *Rao-Blackwellization*.

¶ Suppose our random variables are z and β. Consider a function for which we want to take a posterior expectation $f(z, \beta)$. Our goal is to calculate $\mathbb{E}\left[f(Z, \beta)\right]$ under the posterior distribution $p(z, \beta \mid x)$.

¶ In all MCMC algorithms, we construct an estimator of this expectation through independent samples from the distribution,

$$\mathbb{E}\left[f(Z,\beta)\right] \approx \frac{1}{S}\sum_{s=1}^{S} f\left(z_s,\beta_s\right) \triangleq \kappa\left(x, z_{1:S}, \beta_{1:S}\right), \tag{40}$$

where $\{z_s, \beta_s\}$ are independent samples from $p(z, \beta \mid x)$.

¶ We emphasize that the expectation itself is a function of the "data", including x (non-random), $z_{1:S}$ (random), and $\beta_{1:S}$ (random). This is a statistic and so we can contemplate estimators of it with respect to random samples.

Consider the estimator as an expectation with respect to those samples,

$$\mathbb{E}\left[\kappa(x, z_{1:S}, \beta_{1:S})\right] = \mathbb{E}\left[f(Z,\beta)\right]. \tag{41}$$

This assumes the samples really came from the posterior. It says that the estimator is *unbiased*.

But the estimator also has a variance, $\mathrm{Var}\left[\kappa(x, z_{1:S}, \beta_{1:S})\right]$, which is its spread around the mean (i.e., the expected squared difference). For example, as you might guess, the variance is larger when $S = 1$ and smaller when S is large.

¶ Rao-Blackwellization considers estimators derived from the "tower property", or iterated expectation. Specifically, consider

$$\mathbb{E}\left[\mathbb{E}\left[f(Z,\beta) \mid Z\right]\right] = \mathbb{E}\left[f(Z,\beta)\right]. \tag{42}$$

On the LHS the first expectation is taken with respect to $p(z \mid x)$; the second expectation is taken with respect to $p(\beta \mid z, x)$. Suppose we can take expectations analytically with respect to the second expectation. We set up an the alternative estimator,

$$\mathbb{E}\left[f(Z,\beta)\right] \approx \frac{1}{S}\sum_{s=1}^{S} \mathbb{E}\left[f(z_s,\beta) \mid z_s\right] \triangleq \lambda(z_s, x) \tag{43}$$

where the RHS expectation is with respect to $p(\beta \mid z_s, x)$ and $z_s \sim p(z_s \mid x)$. (Note that β is integrated out.) This too is an unbiased estimator of $\mathbb{E}\left[f(Z,\beta)\right]$. But it has a smaller variance.

¶ In the mixture case above, we sample from the mixture assignments. This implicitly integrates out the mixture locations. Resulting posterior expectations (of anything, including of mixture locations) will have lower variance than in the uncollapsed Gibbs sampler.

A loose history

¶ Metropolis et al. (1953) introduces the Metropolis algorithm in the context of some integrals that come up in physics. (He also invented simulated annealing in the same paper.) This work stemmed from his work in the 40s (with others) in Los Alamos.

¶ Hastings (1970) generalizes the algorithm to Metropolis-Hastings and sets it in a more statistical context. He shows that the Metropolis algorithm (and Metropolis-Hastings) works because they sample from a Markov chain with the appropriate stationary distribution.

¶ Geman and Geman (1984) developed the Gibbs sampler for Ising models, showing that it too samples from an appropriate Markov chain. Gelfand and Smith (1990) built on this work to show how Gibbs sampling can be used in many Bayesian settings. This is also what we've shown in these notes.[1]

¶ In the 90s, statisticians like Tierney, Kass, and Gelman (in a series of papers and books) solidified the relationship between Metropolis, Metropolis-Hastings, and Gibbs sampling. In parallel, the rise of computing power transformed where and how these algorithms can be used.

¶ Today, MCMC is a vital tool for modern applications of probabilistic models.

Aside: Exponential families and conjugacy

[To be written]

Mixtures of exponential families

[To be written]

[1] At that same time, Gelfand (who was faculty at the University of Connecticut) played a monthly poker game with a group of mathematicians and statisticians including Prof. Ron Blei (relation to author: father). They called it "the probability seminar." Sometimes they played in the author's childhood home. Early in the evening, he was allowed to grab a handful of pretzels but, otherwise, was encouraged to stay out of the way.

References

Gelfand, A. and Smith, A. (1990). Sampling based approaches to calculating marginal densities. *Journal of the American Statistical Association*, 85:398–409.

Geman, S. and Geman, D. (1984). Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:721–741.

Hastings, W. (1970). Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57:97–109.

Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller, M., and Teller, E. (1953). Equations of state calculations by fast computing machines. *Journal of Chemical Physics*, 21:1087–1092.