Markov Chain Monte Carlo [1]

Lesson No. 12

Gustavo de J. Merli - 262948

1 Introduction

The basic idea behind MCMC is to construct a Markov chain on the state space X whose stationary distribution is the target density $p^*(x)$ of interest (this may be a prior or a posterior). That is, we perform a random walk on the state space, in such a way that the fraction of time we spend in each state x is proportional to $p^*(x)$. By drawing samples x_0, x_1, x_2,..., from the chain, we can perform Monte Carlo integration wrt p^*. We give the details below.

2 Gibbs sampling

In this section, we present one of the most popular MCMC algorithms, known as Gibbs sampling. This is the MCMC analog of coordinate descent.

2.1 Basic Idea

The idea behind Gibbs sampling is that we sample each variable in turn, conditioned on the values of all the other variables in the distribution. That is, given a joint sample x^s of all the variables, we generate a new sample x^{s+1} by sampling each component in turn, based on the most recent values of the other variables. For example, if we have $D = 3$ variables, we use

- $x_1^{s+1} \sim p(x_1|x_2^s, x_3^s)$
- $x_2^{s+1} \sim p(x_2|x_1^{s+1}, x_3^s)$
- $x_3^{s+1} \sim p(x_2|x_1^{s+1}, x_2^{s+1})$

This readily generalizes to D variables. If x_i is a visible variable, we do not sample it, since its value is already known.

The expression $p(x_i|x_{-i})$ is called the **full conditional** for variable i. In general, x_i may only depend on some of the other variables.

It is necessary to discard some of the initial samples until the Markov chain has **burned in**, or entered its stationary distribution.

2.2 Collapsed Gibbs sampling

In some cases, we can analytically integrate out some of the unknown quantities, and just sample the rest. This is called a collapsed Gibbs sampler, and it tends to be much more efficient, since it is sampling in a lower dimensional space.

More precisely, suppose we sample z and integrate out θ. Thus the θ parameters do not participate in the Markov chain; consequently we can draw conditionally independent samples $\theta^s \sim p(\theta|z^s, D)$, which will have much lower variance than samples drawn from the joint state space. This process is called **Rao-Blackwellisation** that gaurantees that the variance of the estimate created by analytically integrating out θ will always be lower (or rather, will never be higher) than the variance of a direct MC estimate.

2.3 Gibbs sampling for hierarchical GLMs

Often we have data from multiple related sources. If some sources are more reliable and/or data-rich than others, it makes sense to model all the data simultaneously, so as to enable the borrowing of statistical strength. One of the most natural way to solve such problems is to use hierarchical Bayesian modeling, also called multi-level modeling.

2.4 Blocking Gibbs sampling

Gibbs sampling can be quite slow, since it only updates one variable at a time (so-called single site updating). If the variables are highly correlated, it will take a long time to move away from the current state.

In some cases we can efficiently sample groups of variables at a time. This is called blocking Gibbs sampling or blocked Gibbs sampling, and can make much bigger moves through the state space.

3 Metropolis Hastings algorithm

Although Gibbs sampling is simple, it is somewhat restricted in the set of models to which it can be applied. Fortunately, there is a more general algorithm that can be used, known as the **Metropolis Hastings** or **MH** algorithm, which we describe below.

3.1 Basic Idea

The basic idea in MH is that at each step, we propose to move from the current state x to a new state x' with probability $q(x'|x)$, where q is called the **proposal distribution** (also called the kernel). The user is free to use any kind of proposal they want, subject to some conditions which we explain below. This makes MH quite a flexible method. A commonly used proposal is a symmetric Gaussian distribution centered on the current state, $q(x'|x) = \mathcal{N}(x'|x,\Sigma)$; this is called a **random walk Metropolis algorithm**. If we use a proposal of the form $q(x'|x) = q(x')$, where the new state is independent of the old state, we get a method known as the **independence sampler**.

Having proposed a move to x', we then decide whether to accept this proposal or not according to some formula. If the proposal is accepted, the new state is x', otherwise the new state is the same as the current state, x (i.e., we repeat the sample).

If the proposal is symmetric, so $q(x'|x) = q(x|x')$, the acceptance probability is given by the following formula:

$$r = min(1, \frac{p^*(x')}{p^*(x)}) \tag{1}$$

After computing r, a random variable u is sampled from a uniform distribution. If $u < r$, x is updated to x', otherwise the state will be the same as the current state.

If the proposal is asymmetric, so $q(x'|x) \neq q(x|x')$, we need the **Hastings correction**, given by the following:

$$r = min(1, \alpha) \tag{2}$$

$$\alpha = \frac{p^*(x')q(x|x')}{p^*(x)q(x'|x)} \tag{3}$$

3.2 Proposal distributions

For a given target distribution p^*, a proposal distribution q is valid or admissible if it gives a non-zero probability of moving to the states that have non-zero probability in the target. For example, a Gaussian random walk proposal has non-zero probability density on the entire state space, and hence is a valid proposal for any continuous state space.

When using a random walk proposal, $q(x'|x) = N(x'|x, v)$ It is very important to set the variance of the proposal v correctly: If the variance is too low, the chain will only explore one of the modes, but if the variance is too large, most of the moves will be rejected, and the chain will be very sticky, i.e., it will stay in the same state for a long time. If we set the proposal's variance just right, the samples explore the support of the target distribution. Often one experiments with different parameters until the acceptance rate is between 25% and 40%, which theory suggests is optimal, at least for Gaussian target distributions.

3.2.1 Gaussian proposals

If we have a continuous state space, the Hessian H at a local mode $\hat{w}$ can be used to define the covariance of a Gaussian proposal distribution. This approach has the advantage that the Hessian models the local curvature and length scales of each dimension; this approach therefore avoids some of the slow mixing behavior of Gibbs sampling.

There are two obvious approaches: (1) an independence proposal, $q(w'|w) = \mathcal{N}(w'|\hat{w}, H^{-1})$ or (2), a random walk proposal, $q(w'|w) = \mathcal{N}(w'|w, s^2 H^{-1})$, where s^2 is a scale factor chosen to facilitate rapid mixing. If the posterior is Gaussian, the asymptotically optimal value is to use $s^2 = 2.382/D$, where D is the dimensionality of w; this results in an acceptance rate of 0.234.

3.2.2 Mixture proposals

If one doesn't know what kind of proposal to use, one can try a mixture proposal, which is a convex combination of base proposals:

$$q(x'|x) = \sum_{k=1}^{K} w_k q_k(x'|x) \tag{4}$$

where w_k are the mixing weights. As long as each q_k is individually valid, the overall proposal will also be valid.

3.2.3 Data-driven MCMC

The most efficient proposals depend not just on the previous hidden state, but also the visible data, i.e., they have the form $q(x'|x, D)$. This is called data-driven MCMC. To create such proposals, one can sample (x, D) pairs from the forwards model and then train a discriminative classifier to predict $p(x|f(D))$, where $f(D)$ are some features extracted from the visible data.

3.3 Adaptive MCMC

One can change the parameters of the proposal as the algorithm is running to increase efficiency. This is called adaptive MCMC. This allows one to start with a broad covariance (say), allowing large moves through the space until a mode is found, followed by a narrowing of the covariance to ensure careful exploration of the region around the mode. However, one must be careful not to violate the Markov property; thus the parameters of the proposal should not depend on the entire history of the chain.

3.4 Initialization and mode hopping

It is necessary to start MCMC in an initial state that has non-zero probability. If the model has deterministic constraints, finding such a legal configuration may be a hard problem in itself. It is therefore common to initialize MCMC methods at a local mode, found using an optimizer.

4 Speed and accuracy of MCMC

In this section, we discuss a number of important theoretical and practical issues to do with MCMC.

4.1 The burn-in phase

We start MCMC from an arbitrary initial state. Only when the chain has "forgotten" where it started from will the samples be coming from the chain's stationary distribution. Samples collected before the chain has reached its stationary distribution do not come from p^*, and are usually thrown away. The initial period, whose samples will be ignored, is called the **burn-in phase**.

4.2 Mixing rates of Markov chains

The amount of time it takes for a Markov chain to converge to the stationary distribution, and forget its initial state, is called the **mixing time**.

4.3 Practical convergence diagnostics

Computing the mixing time of a chain is in general quite difficult, since the transition matrix is usually very hard to compute.

One of the simplest approaches to assessing when the method has converged is to run multiple chains from very different **overdispersed** starting points, and to plot the samples of some variables of interest. This is called a **trace plot**. If the chain has mixed, it should have "forgotten" where it started from, so the trace plots should converge to the same distribution, and thus overlap with each other.

4.4 Accuracy of MCMC

The samples produced by MCMC are auto-correlated, and this reduces their information content relative to independent or "perfect" samples. We can quantify this as follows. Suppose we want to estimate the mean of $f(X)$, for some function f, where $X \sim p()$. Denote the true mean by

$$f^* = \mathbb{E}[f(X)] \tag{5}$$

A Monte Carlos estimate is given by

$$\bar{f} = \frac{1}{S}\sum_{s=1}^{S} f_s \tag{6}$$

where $f_s = f(x_s)$ and $x_s \sim p(x)$. An MCMC estimate of the variance of this estimate is given by

$$Var_{MCMC}[\bar{f}] = Var_{MC}(\bar{f} + \frac{1}{S^2}\sum_{s \neq t} \mathbb{E}[(f_s - f^*)(f_t - f^*)] \tag{7}$$

where the first term is the Monte Carlo estimate of the variance if the samples weren't correlated, and the second term depends on the correlation of the samples. We can measure this as follows. Define the sample-based auto-correlation at lag t of a set of samples $f_1,...,f_S$ as follows:

$$p_t = \frac{\frac{1}{S-t}\sum_{s=1}^{S-t}(f_s - \bar{f})(f_{s+t} - \bar{f})}{\frac{1}{S-1}\sum_{s=1}^{S}(f_s - \bar{f})^2} \tag{8}$$

This is called the **autocorrelation function** (ACF).

4.5 How many chains?

In practice it is common to run a medium number of chains (say 3) of medium length (say 100,000 steps), and to take samples from each after discarding the first half of the samples.

5 Auxiliary variable MCMC

Sometimes we can dramatically improve the efficiency of sampling by introducing dummy auxiliary variables, in order to reduce correlation between the original variables. If the original variables are denoted by x, and the auxiliary variables by z, we require that $\sum_z p(x,z) = p(x)$, and that $p(x,z)$ is easier to sample from than just $p(x)$. If we meet these two conditions, we can sample in the enlarged model, and then throw away the sampled z values, thereby recovering samples from $p(x)$.

5.1 Auxiliary variable sampling for logistic regression

The latent variable interpretation of probit regression had the form

$$z_i = w^T x_i + \epsilon_i \tag{9}$$

$$\epsilon_i \sim \mathcal{N}(0,1) \tag{10}$$

$$y_i = 1 = \mathbb{I}(z_i \geq 0) \tag{11}$$

It is straightforward to convert this into an auxiliary variable Gibbs sampler, since $p(w|D)$ is Gaussian and $p(z_i|x_i, y_i, w)$ is truncated Gaussian, both of which are easy to sample from.

Now let us discuss how to derive an auxiliary variable Gibbs sampler for logistic regression. Let ϵ_i follow a **logistic distribution**, with pdf

$$p_{Logistic}(\epsilon) = \frac{e^{-\epsilon}}{(1+e^{-\epsilon})^2} \tag{12}$$

with mean $\mathbb{E}[\epsilon] = 0$ and variance $var[\epsilon] = \pi^2/3$. The cdf has the form $F(\epsilon) = sigm(\epsilon)$, which is the logistic function. Since $y_i = 1$ iff $w^T x_i + \epsilon > 0$, we have, by symmetry, that

$$p(y_i = 1|x_i, w) = sigm(w^T x_i) \tag{13}$$

as required.

A simpler approach is to approximate the logistic distribution by the Student distribution. Specifically, we will make the approximation $\epsilon_i \sim T(0,1,v)$, where $v \approx 8$. We can now use the scale mixture of Gaussians representation of the Student to simplify inference. In particular, we write

$$\lambda_i \sim Ga(v/2, v/2) \tag{14}$$

$$\epsilon_i \sim \mathcal{N}(0, \lambda_i^{-1}) \tag{15}$$

$$z_i = w^T x_i + \epsilon_i \tag{16}$$

$$y_i = 1|z_i = \mathbb{I}(z_i \geq 0) \tag{17}$$

All of the full conditionals now have a simple form.

5.2 Slice sampling

Consider sampling from a univariate, but multimodal, distribution $\tilde{p}(x)$. We can sometimes improve the ability to make large moves by adding an auxiliary variable u. We define the joint distribution as follows:

$$\hat{p}(x,u) = 1/Z_p \text{ if } 0 \leq u \leq \tilde{p}(x) \text{ and } 0 \text{ otherwise} \tag{18}$$

It is possible to sample from $p(x)$ by sampling from $\hat{p}(x,u)$ and then ignoring u. The full conditionals have the form

$$p(u|x) = U_{[0,\tilde{p}(x)]}(u) \tag{19}$$

$$p(x|u) = U_A(x) \tag{20}$$

where $A = \{x : \tilde{p}(x) \geq u\}$ is the set of points on or above the chosen height u. This corresponds to a slice through the distribution, hence the term **slice sampling**.

In practice, it can be difficult to identify the set A. So we can use the following approach: construct an interval $x_{min} \leq x \leq x_{max}$ around the current point x^s of some width. We then test to see if each end point lies within the slice. If it does, we keep extending in that direction until it lies outside the slice. This is called stepping out.

A candidate value x' is then chosen uniformly from this region. If it lies within the slice, it is kept, so $x^{s+1} = x'$ Otherwise we shrink the region such that x' forms one end and such that the region still contains x^s. Then another sample is drawn. We continue in this way until a sample is accepted.

6 Annealing methods

Many distributions are multimodal and hence hard to sample from. However, by analogy to the way metals are heated up and then cooled down in order to make the molecules align, we can imagine using a computational temperature parameter to smooth out a distribution, gradually cooling it to recover the original "bumpy" distribution.

6.1 Simulated annealing

Simulated annealing is a stochastic algorithm that attempts to find the global optimum of a black-box function $f(x)$. It is closely related to the Metropolis Hastings algorithm for generating samples from a probability distribution. SA can be used for both discrete and continuous optimization.

The key quantity is the Boltzmann distribution, which specifies that the probability of being in any particular state x is given by

$$p(x) \propto exp(-f(x)/T) \tag{21}$$

where $f(x)$ is the "energy" of the system and T is the computational temperature. As the temperature approaches 0 (so the system is cooled), the system spends more and more time in its minimum energy (most probable) state.

We can generate an algorithm from this as follows. At each step, sample a new state according to some proposal distribution $x \sim q(|x_k)$. For real-valued parameters, this is often simply a random walk proposal, $x' = x_k + \epsilon_k$, where $\epsilon_k \sim \mathcal{N}(0, \Sigma)$. For discrete optimization, other kinds of local moves must be defined.

Having proposed a new state, we compute

$$\alpha = exp((f(x) - f(x'))/T) \tag{22}$$

We then accept the new state (i.e., set $x_{k+1} = x'$) with probability $min(1, \alpha)$, otherwise we stay in the current state (i.e., set $x_{k+1} = x_k$). This means that if the new state has lower energy (is more probable), we will definitely accept it, but it it has higher energy (is less probable), we might still accept, depending on the current temperature.

6.2 Parallel tempering

One way to combine MCMC and annealing is to run multiple chains in parallel at different temperatures, and allow one chain to sample from another chain at a neighboring temperature. In this way, the high temperature chain can make long distance moves through the state space, and have this influence lower temperature chains. This is known as parallel tempering.

7 Approximating the marginal likelihood

The marginal likelihood $p(D|M)$ is a key quantity for Bayesian model selection, and is given by

$$p(D|M) = \int p(D|\theta, M)p(\theta|M)d\theta \tag{23}$$

Unfortunately, this integral is often intractable to compute. In this section, we briefly discuss some ways to approximate this expression using Monte Carlo.

7.1 The candidate method

There is a simple method for approximating the marginal likelihood known as the **Candidate method**. This exploits the following identity

$$p(D|M) = \frac{p(D|\theta, M)p(\theta|M)}{p(\theta|D, M)} \tag{24}$$

Once we have picked some value, we can evaluate $p(D|\theta, M)$, $p(\theta|M)$ and the denominator quite easily. But the method can give very innaccurate results in practice.

7.2 Annealed importance sampling

We can use annealed importance sampling to evaluate a ratio of partition functions. Notice that $Z_0 = \int f_0(x)dx = \int f(z)dz$, and $Z_n = \int f_n(x)dx = \int g(z)dz$. Hence

$$\frac{Z_0}{Z_n} \approx \frac{1}{S}\sum_{s=1}^{S} w_s \tag{25}$$

If f_n is a prior and f_0 is the posterior, we can estimate $Z_n = p(D)$ using the above equation, provided the prior has a known normalization constant Z_0. This is generally considered the method of choice for evaluating difficult partition functions.

References

[1] K. P. Murphy, *Machine Learning : A Probabilistic Perspective*, 1st. Cambridge, Mass. [u.a.]: MIT Press, 2013.