

Monte Carlo methods

Thomas Mailund

September 18, 2007

We are often in a situation where we have some parameters of interest, α , and some other parameters of no interest, β , where we have a model giving us the joint probability, $p(\alpha, \beta)$. Since we are only interested in α we wish to look at the marginal probability $p(\alpha) = \int p(\alpha, \beta) d\beta$.

In many cases, this can be expressed as a matter of calculating an expectation, and since averages tends to expectations in the limit, calculating such expressions can be done through sampling of stochastic variables, when explicit or analytical calculations cannot be done.

For example: we know how to calculate the likelihood of a set of sequences, S , given their phylogeny (tree topology, T , plus branch lengths, B) and a mutation rate, α :

$$\text{lh}(\alpha, T, B | S) = p(S | T, B, \alpha) \quad (1)$$

Since we have only observed the sequences, S , and (in this situation) are only interested in α , the parameters T and B are so-called *nuisance* parameters and we wish to calculate simply

$$\text{lh}(\alpha | S) = p(S | \alpha) = \sum_T \int p(S, T, B | \alpha) dB = \sum_T \int p(S | T, B, \alpha) p(T, B) dB \quad (2)$$

where the last equation assumes that the topology and branch lengths are independent of the mutation rate. The sum and integral can be seen as the expectation of $p(S | T, B, \alpha)$ under the (joint) distribution on T and B :

$$\text{lh}(\alpha | S) = \mathbf{E}_{T,B} [p(S | T, B)] \quad (3)$$

In some cases, we have analytical expressions for calculating expectations. In some cases we have efficient algorithms (e.g. dynamic programming algorithms for computing complicated sums). Then, in some cases, we have neither, but we have a process for *simulating* from the distribution. For trees and branch lengths, in a population genetics context, we have algorithms for generating so called *coalescent trees* (the exact procedure is the topic of another note). If we can sample N trees and branch lengths, T_n, B_n , from $p(T, B)$ we can approximate the expectation:

$$\mathbf{E}_{T,B} [p(S | T, B)] \approx \frac{1}{N} \sum_{n=1}^N p(S | T_n, B_n) \quad (4)$$

Methods where we do such calculations based on sampling are known as *Monte Carlo* methods. In the coalescent tree example we have a procedure to sample $T, B \sim p(T, B)$ but this isn't always the case. If, for instance, we wanted

to sample trees and branch lengths from the so-called *posterior* distribution of these, $p(T, B | D)$, we wouldn't have a similar procedure for sampling T and B .

The posterior distribution is given by

$$p(T, B | D) = \frac{p(D | T, B)p(T, B)}{p(D)} \propto p(D | T, B)p(T, B) \quad (5)$$

where \propto means *proportional to*, in this case that $p(T, B | D)$, as a function of T and B , is proportional to $p(D | T, B)p(T, B)$ with proportionality constant $p(D)$ (that does not depend on T and B). For coalescent trees, we have an algorithm for sampling $T, B \sim p(T, B)$ and we have an analytical expression for computing $p(D | T, B)$, but we do not have a simple expression for calculating, nor a simple algorithm for sampling from, $p(T, B | D)$.

When we do not have an obvious sampling method (or better still, an analytical expression) we might be able to construct one. In the following, I describe a few general techniques. We'll return to the problem above, from time to time.

Rejection sampling

Assume we wish to draw samples from a distribution $p(x)$ where we have no simple strategy for sampling $x \sim p(x)$. We assume, however, that we can calculate $p(x)$ or at least calculate it up to a normalising constant, i.e. we can calculate some $\tilde{p}(x)$ where $p(x) = K\tilde{p}(x)$ but we do not necessarily know K . A common case is where x takes values from a very large set, so we cannot simply enumerate all possible values and their probabilities and sample that way, nor compute K that way.

Assume then that we have another distribution, $q(x)$ that is easy to sample from, and that satisfy $c \cdot q(x) \geq p(x)$ for all x and for some constant c (or $c \cdot q(x) \geq \tilde{p}(x)$ if we do not know the normalising constant for $p(x)$). We call $q(x)$ the *proposal distribution* and the property $c \cdot q(x) \geq p(x)$ the *envelope property*. The trick now is to sample $x \sim q(x)$ and $u \sim U[0, 1]$ (uniform in $[0, 1]$) and to "accept" x if $u \cdot c \cdot q(x) \leq p(x)$ and "reject" it otherwise.

We can show that x is then sampled from $p(x)$ by showing that the conditional distribution $p(x | u \cdot c \cdot q(x) \leq p(x))$ is $p(x)$:

$$p(x | u \cdot c \cdot q(x) \leq p(x)) = \frac{p(u \cdot c \cdot q(x) \leq p(x) | x) q(x)}{\int p(u \cdot c \cdot q(x) \leq p(x) | x) q(x) dx} \quad (6)$$

$$= \frac{[p(x)/cq(x)] q(x)}{\int [p(x)/cq(x)] q(x) dx} \quad (7)$$

$$= \frac{1/c \cdot [p(x)/q(x)] q(x)}{1/c \cdot \int [p(x)/q(x)] q(x) dx} \quad (8)$$

$$= p(x) \quad (9)$$

where (6) uses Bayes rule, (7) uses that the probability of accepting is exactly the fraction between $c \cdot q(x)$ and $p(x)$, (8) is just arithmetic, and (9) cancels the

cs and uses that a distribution integrates to 1. Repeating the arithmetic with $p(x) = K\tilde{p}(x)$ would just have the K s cancel, showing that we only need to know p up to a normalising constant.

The probability of acceptance is given by

$$p(\text{accept}) = \int [p(x)/cq(x)] q(x) dx = \frac{1}{c} \int p(x) dx = \frac{1}{c} \quad (10)$$

so the larger the constant needed to make $c \cdot q(x) \geq p(x)$ for all x , the larger a fraction of our samples we need to reject, wasting computations.

Returning to our tree sampling example we can see if we can approach it with a rejection sampling procedure. Our proposal distribution could be $q(x) = p(T, B)$ and our desired distribution, known up to a normalising constant, $\tilde{p}(x) = p(D | T, B)$. What about c ?

With c we have a bit of a problem. We need some c such that $c \cdot p(T, B) \geq p(D | T, B)$ for all T and B , but we do not really have any obvious choices. This is a general problem with rejection sampling: we need the constant c , but rarely do we have one (and often when we use one, we do not actually prove that it satisfy the inequality). Without it, the proof makes no guarantees about the correctness of our sampling strategy.

What happens if we just propose $T, B \sim p(T, B)$ and accept with probability $p(D | T, B)$? The probability of the accepted T, B are then:

$$p(T, B | u \leq p(D | T, B)) = \frac{p(u \leq p(D | T, B)) p(T, B)}{\sum_T \int p(u \leq p(D | T, B)) p(T, B) dB} \quad (11)$$

$$= \frac{p(D | T, B) p(T, B)}{p(D)} = p(T, B | D) \quad (12)$$

which is exactly what we want! So in this case it was much simpler than the general case, and we didn't even need the constant c .

How can it be that we didn't need c ? Is it something we can do generally? We can try to redo the proof in the general case with $p(x)$ and $q(x)$. We are sampling from $q(x)$ and accept with some function proportional $\tilde{p}(x) = \frac{1}{K}p(x)$. The probability of the accepted values are:

$$p(x | u \leq \tilde{p}(x)) = \frac{p(u \leq \tilde{p}(x)) q(x)}{\int p(u \leq \tilde{p}(x)) q(x) dx} \quad (13)$$

$$= \frac{\tilde{p}(x)q(x)}{\int \tilde{p}(x)q(x) dx} = \frac{\frac{1}{K}p(x)q(x)}{\int \frac{1}{K}p(x)q(x) dx} \quad (14)$$

$$= \frac{p(x)q(x)}{\int p(x)q(x) dx} \quad (15)$$

It was only because $p(x) \propto k\tilde{p}(x) \cdot q(x)$, so to speak, with the proportionality constant implicitly handled, that it work. This won't always be the case. When

it isn't, we cannot simply sample from $q(x)$ and accept with probability $\tilde{p}(x)$, and then we *do* need the c .

What about the acceptance probability? Here we get (of course)

$$p(\text{accept}) = \sum_T \int p(D|T, B)p(T, B) dB = p(D) \quad (16)$$

which is typically *very* small. So we easily need to sample millions of trees for each acceptance. This is not an efficient approach to this particular problem. For some problems it is, but not this particular one.

Importance sampling

Importance sampling tries to improve on the problems with high rejection rates. It won't let you sample from the distribution you are interested in, but it allows you to compute expectations there, which, as you may recall, was our original intend.

If we wish to know the expectation of some function f of x under probability distribution $p(x)$ we can change it to an expectation under another distribution, $q(x)$ as follows:

$$E_p[f(x)] = \int f(x)p(x) dx = \int f(x)p(x)\frac{q(x)}{q(x)} dx \quad (17)$$

$$= \int f(x)\frac{p(x)}{q(x)}q(x) dx = E_q\left[f(x)\frac{p(x)}{q(x)}\right] \quad (18)$$

$$\approx \frac{1}{N} \sum_{n=1}^N f(x_n)\frac{p(x_n)}{q(x_n)} \quad (19)$$

where the the sum is over x_n sampled from $q(x)$.

Of course, for this to work, $q(x)$ shouldn't be zero in the area we integrate—dividing by zero is frowned upon—which essentially means that we cannot permit $q(x)$ to be zero anywhere $p(x)$ isn't ($p(x) > 0$ should imply $q(x) > 0$), but aside from that, we can use any $q(x)$ we can sample from.

The quality of the estimate will depend heavily on the choice of proposal distribution $q(x)$, though. The variance in estimating a mean always depends on the variance in the sample distribution—think central limit theorem—and in importance sampling, the variance in the weighting ration, $p(x)/q(x)$, will contribute to the variance in the samples.

What happens when we only know $p(x)$ up to a normalising constant? Or when we only know $q(x)$ up to a normalising constant?¹ Let $\tilde{p}(x) = K_p p(x)$ and

¹By knowing $q(x)$ up to a normalising constant I mean we can sample from $q(x)$ but we can only calculate the value $\tilde{q}(x)$ where we do not know the normalisation constant.

$\tilde{q}(x) = K_q q(x)$. We then have:

$$E_p[f(x)] = \int f(x)p(x) dx = \int f(x) \frac{\tilde{p}(x)/K_p}{\tilde{q}(x)/K_q} q(x) dx = \frac{K_q}{K_p} E_q \left[f(x) \frac{\tilde{p}(x)}{\tilde{q}(x)} \right] \quad (20)$$

$$\approx \frac{K_q}{K_p} \frac{1}{N} \sum_{n=1}^N f(x_n) \frac{\tilde{p}(x_n)}{\tilde{q}(x_n)} \quad (21)$$

where, remember, we do not actually know K_p or K_q and thus not $\frac{K_q}{K_p}$. We *do* know, however, $K_p = \int \tilde{p}(x) dx$ by definition of normalising constant. Therefore

$$\frac{K_p}{K_q} = \frac{1}{K_q} \int \tilde{p}(x) dx = \frac{1}{K_q} \int \frac{\tilde{p}(x)}{q(x)} q(x) dx = \int \frac{\tilde{p}(x)}{K_q q(x)} q(x) dx \quad (22)$$

$$= \int \frac{\tilde{p}(x)}{\tilde{q}(x)} q(x) dx \approx \frac{1}{N} \sum_{n=1}^N \frac{\tilde{p}(x_n)}{\tilde{q}(x_n)} \quad (23)$$

where the values in the last sum are sampled from $q(x)$ (which we assume we can sample from even when the normalising constant is unknown).

We need to compute all the ratios $\tilde{p}(x)/\tilde{q}(x)$ when doing the importance sampling anyway, and (23) tells us that by dividing with their average we get the right mean:

$$E_p[f(x)] \approx \left(\sum_{n=1}^N \frac{\tilde{p}(x)}{\tilde{q}(x)} \right)^{-1} \left(\sum_{n=1}^N f(x) \frac{\tilde{p}(x)}{\tilde{q}(x)} \right) \quad (24)$$

where (24) combines (21) with (23) (with $1/N$ cancelling in the two expressions). With this minor modification we can also importance sample when we only know the distributions up to normalisation.

Markov-Chain Monte Carlo

Markov-Chain Monte Carlo (MCMC) methods take a different approach than the rejection sampling and the importance sampling methods. In MCMCs the idea is to construct a Markov chain with stationary distribution $p(x)$ and then sample from this by running the chain for a while, until it has converged to $p(x)$ after which we sample simply by running the chain some more. Doing it this way, we will not get independent samples from $p(x)$ but we can still calculate the expectation by calculating averages. We just need to be careful about getting enough samples, which can potentially be far more than needed if the samples were independent.

In Ewens and Grant section 11.5 you saw the two main approaches to constructing a Markov chain with a desired stationary distribution: The Hastings-Metropolis algorithm and Gibbs sampling.

In theory, both gives you recipes for constructing Markov chains that will converge to the desired distribution so you can sample from it to calculate

the expectation. In practise, there are a few problems that must be addressed: How do we recognise that the Markov chain has converged to the stationary distribution? and how do we ensure that the Markov chain traverses the state space efficiently, so our samples are giving us an accurate picture of the distribution? (remember that we are not drawing independent, but possibly highly correlated samples).

This will be the topic of the next lecture.