Basis Expansion Monte Carlo

Eric Kernfeld *

University of Washington, Seattle, WA, USA

November 17, 2014

Abstract

Most Monte Carlo inference methods are left to run until the markov chain reaches its steady state. This leaves the user with a large chain of samples from the distribution of interest. We introduce Basis Expansion Monte Carlo, which runs the sampler piecewise starting at different places in the parameter space in order extract information more quickly. To make inference about the steady state, we gradually update an approximation of the hidden linear operator that underlies any Metropolis-Hastings or Gibbs sampler. We use the steady-state of the approximate operator in place of the true steady-state. Results show ...

1 Introduction

In many statistical models, it is impossible to find a closed form for the distribution of interest (we will call this π). One work-around, originating in computational physics, relies on the fact that for points x_1 and x_2 in the parameter space, $\pi(x_1)/\pi(x_2)$ may still be calculable, though $\pi(x_1)$ and $\pi(x_2)$ are not. This fact is exploited to produce a Markov chain whose steady-state distribution is guaranteed to be π.

More and references about history, background, and/or tutorials on monte carlo methods

The Metropolis-Hastings scheme consists of the following procedure.

Algorithm 1: Metropolis-Hastings algorithm

Set $x_0 = 0, i = 0$
Repeat ad nauseum:
  Increment i
  Draw x from a proposal distribution $q(x|x_{i-1})$
  Set $\alpha(x|x_{i-1}) = 1 - min(1, \frac{\pi(x)q(x_{i-1}|x)}{\pi(x_{i-1})q(x|x_{i-1})})$
  Draw u from a uniform density on $[0, 1]$. Set $x_i = x$ with probability $1 - \alpha$, i.e. if $u > \alpha$, and $x_i = x_{i-1}$ otherwise.

Suppose this MCMC algorithm produces a chain $x_1, x_2, x_3, ...$ of samples. Because the algorithm is stochastic, these samples can be viewed as realizations of random variables $X_1, X_2, X_3, ...$ with marginal density functions f_1, f_2, f_3, etc. If you initialize deterministically, then X_1 is just a constant. Because X_i is independent of past draws given X_{i-1}, we can write $f_i(x_i) = \int f_{i|i-1}(x_i, x_{i-1}) f_{i-1}(x_{i-1}) dx_{i-1}$ using the conditional density of X_i given X_{i-1}. Noting that $f_{i|i-1}$ doesn't depend on i, we can replace it with a function $f_{2|1}$ so that $f_i(x_i) = \int f_{2|1}(x_i, x_{i-1}) f_{i-1}(x_{i-1}) dx_{i-1}$. This means there is a fixed linear operator L so that $f_i = L f_{i-1}$, analogous to the transition probability matrices of discrete-space Markov chain theory. As L cannot be observed directly, we refer to it as the hidden action of an M-H algorithm.

In BEMC, we approximate L in stage one, then compute π from the approximation in stage two.

*Electronic address: ekernf01@u.washington.edu; Corresponding author

1.1 Stage one: approximating the hidden action

Let us look at this operator in more detail. Going back to the algorithm, the quantity $\alpha(x|x_{i-1})$ is the probability of rejecting a move from x_{i-1} to x. For convenience, let $\alpha(x_{i-1})$ denote the (overall) probability of rejecting a move from x_{i-1}. Splitting it up as an alternative between moving and staying put, we can write $f_{2|1}(x_2, x_1) = \alpha(x_1)\delta_{x_1}(x_2) + (1 - \alpha(x_1))r(x_2|x_1)$. We would interpret $r(x_2|x_1)$ as the probability density of landing at x_2 given that our move out of x_1 was not rejected. This is not the same as $q(x_2|x_1)$, since the lack of rejection informs us that we have more likely moved into a region of higher probability. To set up the last line below, define $(D_\alpha f)(x) \equiv \alpha(x)f(x)$, and with consistent notation, let $(D_{1-\alpha}f)(x)$ equal $(1-\alpha(x))f(x)$. Finally, let $(L_{acc}f)(x) \equiv \int r(x|y)(D_{1-\alpha}f)(y)dy$. Then:

$$\begin{aligned}
f_2(x_2) &= \int f_{2|1}(x_2, x_1)f_1(x_1)dx_1 \\
&= \int (\alpha(x_1)\delta_{x_1}(x_2) + (1 - \alpha(x_1))r(x_2|x_1))f_1(x_1)dx_1 \\
&= \int \alpha(x_1)\delta_{x_2}(x_1)f_1(x_1)dx_1 + \int (1 - \alpha(x_1))r(x_2|x_1)f_1(x_1)dx_1 \\
&= \alpha(x_2)f_1(x_2) + \int (1 - \alpha(x_1))r(x_2|x_1)f_1(x_1)dx_1 \\
&= (D_\alpha f_1)(x_2) + (L_{acc}f_1)(x_2)
\end{aligned}$$

We can sample from a pdf proportional to $D_{1-\alpha}f$ by sampling z from $f(x)$, then running an M-H iteration on z to get w and retaining the sample z if $w \neq z$. We can sample from a pdf proportional to $L_{acc}f$ by doing the same, but retaining w if $w \neq z$. These facts will be useful as we attempt to estimate L_{acc}.

Our estimator is parametric, using a fixed set of functions $\{h_i\}_{i=1}^B$ from Ω to $\mathbb{R}$. We will choose them to be orthogonal with respect to an L_2 inner product, i.e. $\int_\Omega h_i(x)h_j(x)dx = 0$ when $i \neq j$. For $\Omega = \mathbb{R}^n$, we use Gaussian-weighted Hermite polynomials. Consider also a function $\hat{\alpha} \in \text{span}(\{h_i\}_{i=1}^B)$ and a matrix M in $\mathbb{R}^{B\times B}$. We will attempt to set things up so that $\hat{\alpha} \approx \alpha$ and $L_{acc} \approx \hat{L}_{acc}$, where $(\hat{L}_{rej}f)(x) = \hat{\alpha}(x)f(x)$ and $(\hat{L}_{acc}f)(x) = \sum_{i,j=1}^B h_i(x)M_{ij}\int h_j(x)f(x)dx$.

At this point in the narration, $\hat{\alpha}$ and M are unknown–strategies to estimate them follow. Even if they were chosen optimally, L may not take the same form as $\hat{L}_\alpha + \hat{L}_M$, so the estimate π may not be correct. Later on, we will return to these concerns.

For the moment, we need to estimate M and $\hat{\alpha}$. Fortunately, it is easy to tell when the sampler rejects and when it doesn't. Suppose for a moment that we start the sampler at a point z and it takes a single step to w. If $w \neq z$, then the sampler has shown less of a tendency to reject starting from z, and we label z with a 0. If $w = z$, we label z with a 1. Once the sample space is covered in zeroes and ones, there are many probabilistic classifier methods that could give an estimate of $\hat{L}_{rej}$, which at any given point is just the probability of labeling with a one.

Meanwhile, whenever the sampler moves, we gain information about L_{acc}. To make use of it, notice that the orthogonality of the basis functions implies $\int h_i(x)(\hat{L}_{acc}h_j)(x)dx = M_{ij}$. This can be written as an expectation $M_{ij} = E_{L_{acc}h_j}[h_i]$, which motivates us to sample from $L_{acc}h_j$ and approximate M_{ij} as a sum. All we need to do is follow the procedure from earlier: sample z from h_j, run an M-H iteration on z to get w, and retain w if $w \neq z$.

How do we sample from h, a basis function that sometimes takes negative values? How do we formally take an expectation? The important property to preserve is the law of large numbers: sample averages of some functions should still converge to their expectation. We use a classic tactic from analysis. Let h_+ be defined as $c_+^{-1}max(h, 0)$ and let h_- be defined as $-c_-^{-1}min(h, 0)$, with c_+ and c_- chosen so h_+ and h_- each integrate to one. Then define $E_h[f]$ as $c_+E_{h_+}[f] - c_-E_{h_-}[f]$. We can approximate this expectation by sampling z_{n+} from $h_+, n = 1...N_+$ and $z_{n-}, n = 1...N_-$ from h_-. We would then compute $E_h[f] \approx \frac{c_+}{N_+}\sum f(z_{n+}) - \frac{c_-}{N_-}\sum f(z_{n-})$. The optimal allocation of samples between h_+ and h_- minimizes the overall variance, $\frac{c_+^2}{N_+}Var_{h_+}[f] + \frac{c_-^2}{N_-}Var_{h_-}[f]$. To sample from h_+ and h_-, which may not have closed-form inverse CDF's, we employ rejection sampling.

To take care of one last detail, suppose ϕ is $L_{acc}[(1-\alpha)h]$ for some h, and we can only sample from ϕ by running an M-H iteration on samples from h and discarding the rejections. We can sample from ϕ_+ by sampling from h_+, then applying an M-H iteration. That is because neither L_{acc} nor multiplication by $(1-\alpha)$ will change the sign of a nonnegative function at any point: $(1-\alpha)$ is a probability, bounded below by zero, and the kernel function $r(\cdot|\cdot)$ of L_{acc} is nonnegative as well.

Algorithm 2: BEMC algorithm–stage one

```
Set M to 0.
Set T = {}. T will be the training set for α.
For b_in = 1 : B
  For b_out = 1 : B
    For n = 1 : N
      Draw a sample z_n from h_{b_in}.
      Run the sampler for one round on z_n. Call the result w_n.
      If z_n = w_n:
        Add (z_n, 1) to T.
      Otherwise:
        Add (z_n, 0) to T.
        Increment M_{b_out, b_in} by h_{b_in(w_n)}/N.
  Train α on T.
```

1.2 Stage two: computing the steady state

Given $\hat{M}$, $\hat{\alpha}$ and an initial state f_0, we want to compute $[\hat{L}_{rej} + \hat{L}_{acc}(1-\hat{\alpha})]^P(f_0)$ for some moderately high exponent P.

1.3 Gibbs sampling variant

This approximation can also be adapted to Gibbs sampling, a ubiquitous MCMC variant.