# Basis Expansion Monte Carlo

Eric Kernfeld [*]

*University of Washington, Seattle, WA, USA*

December 22, 2014

**Abstract**

We introduce Basis Expansion Monte Carlo, which studies a Gibbs or Metropolis-Hastings sampler to infer the underlying transition kernel. To make inference about the steady state, we compute the steady-state of the approximate kernel. 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 $\pi$). 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 $\pi$.

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

One popular method, 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 $K$ so that $f_i(x_i) = \int K(x_i, x_{i-1})f_{i-1}(x_{i-1})dx_{i-1}$. This function $K$, called the Markov kernel, is analogous to the transition probability matrices of discrete-space Markov chain theory. We refer to the linear operator of integrating against $K$ as $L$, so that $f_i = Lf_{i-1}$. The object of interest is the steady state of this operator, an eigenfunction $\pi$ that has eigenvalue 1 so that for any $x$, $\pi(x) = \int K(x, t)\pi(t)dt$. In MCMC methods, chains are usually left to run until the Markov chain reaches its steady state. In BEMC, we approximate $L$, then compute $\pi$ from the approximation.

---

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

## 1.1 Stage one: approximating the kernel

Our estimator is parametric, using a fixed set of functions $\{h_i\}_{i=1}^B$ from $\Omega$ 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 Hermite functions, which are exponentially-weighted orthogonal polynomials. We will attempt to estimate a matrix $M$ in $\mathbb{R}^{B \times B}$ such that $L \approx \hat{L}$, where $(\hat{L}f)(x) = \sum_{i,j=1}^B h_i(x)M_{ij}\int h_j(x)f(x)dx$. Equivalently, we approximate $K$ as $\hat{K}(x,y) = \sum_{i,j=1}^B M_{ij}h_i(x)h_j(y)$.

This approximation can imitate continuous kernels, i.e. situations where $\int K(x_i,x_{i-1})f_{i-1}(x_{i-1})dx_{i-1}$ can be done with respect to the Lebesgue measure. This presents an obstacle, because with positive probability, the Metropolis-Hastings algorithm will reject a proposed sample and stay in place. As a workaround, we approximate the kernel not of a single M-H iteration but of $\ell$ iterations for $\ell$ around 10 or 20. The probability of $\ell$ consecutive rejections is much smaller, forcing the true kernel closer to the subspace in which we approximate it. In section 1.3, we discuss a variant that explicitly models rejection events.

How will we choose $M$? Notice that the orthogonality of the basis functions implies $\int h_i(x)(\hat{L}h_j)(x)dx = M_{ij}$. This can be written as an expectation $M_{ij} = E_{Lh_j}[h_i]$, which motivates us to sample from $Lh_j$ and approximate $M_{ij}$ as a sum. All we need to do is sample $z$ from $h_j$, run an M-H iteration on $z$ to get $w$, and retain $w$ as our sample.

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 should still converge to their expectation. We use a classic tactic from analysis. Let $h_+(x)$ be defined as $\frac{1}{c_+}\max(h(x),0)$ and let $h_-(x)$ be defined as $\frac{-1}{c_-}\min(h(x),0)$, 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 $E_h[f] \approx \frac{c_+}{N_+}\sum f(z_{n+}) - \frac{c_-}{N_-}\sum f(z_{n-})$.

To take care of one last detail, suppose $\phi$ is $Lh$ for some $h$, and we can only sample from $\phi$ by running an M-H iteration on samples from $h$. We need to know we can sample from $\phi_+$ by sampling from $h_+$ and applying an M-H iteration. In fact, we can, because $L$ will not change the sign of a function.

---

**Algorithm 2:** BEMC algorithm–stage one

---
Set $M$ to a matrix of all zeroes.
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 M-H sampler for $\ell$ rounds on $z_n$. Call the result $w_n$.
       Increment $M_{b_{out},b_{in}}$ by $h_{b_{in}(w_n)}/N$.

---

## 1.2 BEMC-G, a Gibbs sampling variant

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

## 1.3 BEMC-R, a variant modeling rejections

As we mention in section 1.1, our scheme is able to model continuous kernels. On the other hand, the Metropolis-Hastings algorithm sometimes rejects proposed samples, so its kernel will have a component shaped like a Dirac delta function. In this section, we introduce a variant of BEMC that explicitly models rejections by the sampler.

Let us look at the Metropolis-Hastings kernel 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 up the next draw as an alternative between moving and staying put, we can write $K(x_2,x_1) = \alpha(x_1)\delta_{x_1}(x_2) + (1 - \alpha(x_1))r(x_2|x_1)$. In this expression, $r(x_2|x_1)$ is the conditional density of $x_2$ 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 $D_\alpha$ from $\alpha$ so that $(D_\alpha f)(x) \equiv \alpha(x)f(x)$, and let $(L_{acc}f)(x) \equiv \int r(x|y)(1 - \alpha(y))f(y)dy$. Then:

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

We can sample from a pdf proportional to $D_\alpha f$ 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 nearly the same steps, but retaining $w$ if $w \neq z$. These facts will be useful as we attempt to estimate $L_{acc}$.

This time around, we will try to estimate a function $\hat{\alpha}$ and a matrix $M$ so that $\hat{\alpha} \approx \alpha$ and $L_{acc} \approx \hat{L}_{acc}$, where $(\hat{L}_{acc}f)(x) = \sum_{i,j=1}^{B} h_i(x)M_{ij}\int h_j(x)f(x)dx$. Even if the parameters were chosen optimally, $L$ may not take the same form as $\hat{L}_\alpha + \hat{L}_M$, so the estimate $\hat{\pi}$ will not be correct. EMK: Need some results answering "in what sense is your method correct?"

For this variant, we need still need to estimate $M$ with the added complication of trying to infer $\hat{\alpha}$ at the same time. Fortunately, it is easy to tell when the sampler rejects and when it doesn't, and this provides a way to tease out information about $\alpha$. 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 $\hat{\alpha}$, which at any given point is just the probability of labeling with a one. Meanwhile, whenever the sampler moves, we gain information about $L_{acc}$, and we can update $M$ as before.

This strategy still throws away useful information. To see why, recall that the Metropolis-Hastings algorithm makes a proposal, computes an rejection probability, flips a proverbial coin with that probability, and then discards the rejection probability. When drawing a chain of samples, the rejection probability serves no further purpose, so discarding it is natural. In BEMC-R, though, it provides a more efficient estimate of $\alpha$. If the rejection probability when proposing a move to $w$ from $z$ is $p$, then the better procedure is to label $z$ with $p$. Likewise, instead of updating the estimate of $M_{ij}$ with sample of weight 1 with probability $p$, we can update it with a sample of weight $p$.

---

**Algorithm 3:** BEMC-R algorithm–stage one

---
Set $M$ to 0.
Set a scalar $W$ to zero. $W$ is the effective number of samples in an estimate of an entry of $M$.
Set $T = \{\}$. $T$ will be the training set for $\hat{\alpha}$.
For $b_{in} = 1 : B$
   For $b_{out} = 1 : B$
     For $n = 1 : N$
        Draw a sample $z_n$ from $h_{b_{in}}$.
        Draw a proposal $w_n$ and compute its rejection probability $p$.
        Add $(z_n, p)$ to $T$.
        Increment $M_{b_{out},b_{in}}$ by $ph_{b_{in}(w_n)}$.
        Increment $W$ by $p$.
     Divide $M_{b_{out},b_{in}}$ by $W$.
Train $\hat{\alpha}$ on $T$.

---

## 1.4 Computing the steady state in BEMC-R

Given $\hat{M}$, $\hat{\alpha}$, and an initial state $f_0$, we want to compute $[D_{\hat{\alpha}} + \hat{L}_{acc}]^P(f_0)$ for some moderately high exponent $P$. To simplify the problem, suppose we set $f_0$ to $h_1$, one of the initial $B$ basis functions. Also, suppose that we restrict $\hat{\alpha}$ to a form where for any of our basis functions $h_i$, we can expand $\hat{\alpha} h_i$ as a sum $\sum_{i=1}^{B} c_i h_i$. Because of the orthogonality, computing $\hat{L}_{acc}(f_0)$ is simple: $\hat{L}_{acc}(f_0) = \sum_{i=1}^{B} \hat{L}_{acc}(c_i h_i) = \sum_{i=1}^{B} [\hat{M}c]_i h_i$. The difficulty lies in finding a representation of $D_{\hat{\alpha}}(f_0)$ in this basis, i.e. evaluating or quickly approximating integrals of the form $\int_{\Omega} h_i(x) h_j(x) \hat{\alpha}(x) dx$. EMK: Maybe we'll choose a crafty form for $\hat{\alpha}$ and do this analytically.

## 2 Implementation details

At this point, we will introduce a family of basis functions and begin to work in specifics. We will at first discuss the scenario where the sample space $\Omega$ is $\mathbf{R}$. Our basis of choice, the Hermite functions, are generated by setting $\phi_{n+1} = \sqrt{\frac{2}{n+1}} \left[ x\phi_n - \sqrt{\frac{n}{2}} \phi_{n-1} \right]$, with $\phi_0 = \pi^{-\frac{1}{4}} e^{\frac{-x^2}{2}}$ and $\phi_1 = \sqrt{2}\pi^{-\frac{1}{4}} x e^{\frac{-x^2}{2}}$. It has the property that $\int_{\mathbf{R}} \phi_i \phi_j = 1$ if $i = j$ and 0 otherwise.

In order to fit the BEMC mold, we need to separate the positive and negative parts for each function, find their normalizing constants $c_+$ and $c_-$, and sample from densities proportional to the positive and negative parts. Luckily, the sign changes are few in number and easy to find: they're just the roots of the Hermite polynomials. EMK: Only true if you make sure Hermite polynomials have no irreducible quadratic factors. Is there a closed form for the Hermite polynomial roots? We need to integrate terms of the form $cx^k \exp(\frac{-x^2}{2})$, a task we can rewrite using gamma densities. Below, $Y$ is a Gamma random variable with shape $a = (k+1)/2$ and rate $b = -1/2$; $F_Y$ is its cumulative distribution function. The variables $x_0$ and $x_1$ are roots of whichever Hermite polynomial we are working with.

$$
\int_{x_0}^{x_1} cx^k \exp(\frac{-x^2}{2}) dx = \int_{x_0^2}^{x_1^2} \frac{c}{2} y^{(\frac{k-1}{2})} \exp(\frac{-y}{2}) dy
$$
$$
= \frac{c}{2} \frac{\Gamma(a)}{b^a} \int_{x_0^2}^{x_1^2} \frac{b^a}{\Gamma(a)} y^{(a-1)} \exp(-by) dy
$$
$$
= \frac{c}{2} \frac{\Gamma(a)}{b^a} (F_Y(x_1^2) - F_Y(x_1^2))
$$

Strategy: find a gamma density that dominates, then do that funny rejection sampling variant compute $c_+$ and $c_-$. Strategy: find the zeroes of the Hermite polynomial, then numerically integrate–probably could just use `pgamma()` in `R`.

## 3 Proof of Correctness

In stage 1, we will suffer some error while estimating $M$ and $\alpha$ from. The true transition kernel cannot always be represented in the finite-dimensional form we impose, which is a second source of error. In stage two of BEMC-R, we incur a third source, namely that $D_{\hat{\alpha}}(f_0)$ does not necessarily take the form $\sum_{i=1}^{B} c_i h_i$, especially if $\hat{\alpha}$ is also represented in the same basis. For example, if the basis were Gaussian-weighted polynomials up to degree 15, the degree of $D_{\hat{\alpha}}(f_0)$ would quickly exceed 15, and the variance of the Gaussian would change.