Exact MCMC with Firefly Monte Carlo: A Review

Benjamin Draves

October 26, 2019

1 Introduction & Motivation

Firefly Monte Carlo Markov Chain (MCMC) is a MCMC subsampling method that attempts to gain computational speed in sampling from a posterior distribution of interest [1]. In most Bayesian computation problems, repeated computation of the the likelihood function is central to each iteration of the sampling algorithm. For example, at each iteration of the celebrated Metropolis - Hastings algorithm it is necessary to calculate the acceptance probability of a newly proposed state

$$\alpha(\theta^{(t)}, \theta^*) = \min\left\{1, \frac{p(x|\theta^*)p(\theta^*)}{p(x|\theta^{(t)})p(\theta^{(t)})}\right\}.$$

In certain cases (e.g. the exponential family), the likelihood computation $p(x|\theta)$ scales as the size of the parameter space and the overall sampling complexity will only incur a $O(d)$ slowdown. For a general likelihood function however, computation of the likelihood will require scanning each data value. In this case, the likelihood computation $p(x|\theta)$ scales with the size of the data and the overall complexity of the algorithm will incur a linear time slow down $O(n)$. For large datasets, this slowdown can make many popular sampling techniques infeasible.

There are two popular ways to address this slowdown. The first, and most simple, is analyzing the likelihood directly. For example, in the exponential family under the assumption of conditional independence

$$p(x|\theta) \propto \exp\left\{\eta(\theta)' \sum_{n=1}^{N} T(x_n) - nA(\theta)\right\}.$$

Therefore, by precomputing the sufficient statistics of $\sum_{n=1}^{N} T(x_n)$, the likelihood can be computed in $O(d)$ time where $\eta(\theta) \in \mathbb{R}^d$. The clear drawback of this method is that not all likelihoods will have such a simple form. The second and more flexible option to to approximate the likelihood with subsets of data [1]. By defining a subset $S \subset x$ with $|S| << |x|$ these techniques instead compute

$$\breve{p}(x|\theta) \propto \prod_{n:x_n \in S} p_n(x_n|\theta)$$

and use $\breve{p}(x|\theta)$ as a proxy for $p(x|\theta)$. In this setting, focus is spent on constructing *representative* subsets S in some appropriate sense. Regardless of how well this approximation is constructed however, by replacing

$p(x|\theta)$ with $\breve{p}(x|\theta)$ will result in sampling from an *approximate* posterior $\breve{\pi}(\theta|x)$ instead of the true posterior $\pi(\theta|x)$.

In their paper, Dougal Macclaurin and Ryan P. Adams attempt to combine these two approaches while drawing from the *exact* posterior distribution. By introducing a set of latent variables that turn the data "on and off" they introduce this subsampling notion. In addition, they assume each contribution to the likelihood $p_n(x_n|\theta)$ has a lower bound that is "collapsible" in the same sense of exponential family. In this way, these pieces of the likelihood can be computed in a time independent of the size of data. Assuming these lower bounds are available, this approach offers significant computational improvements while providing successive independent draws from the true posterior distribution. In this report, we introduce Firefly MCMC, investigate the implementation details, and discuss its limitations and advantages.

2 FireFly Monte Carlo

Firefly Monte Carlo is a method that looks to improve computational time of MCMC sampling techniques by improving the evaluation of the likelihood function. For concreteness suppose have the following model

$$\begin{aligned} x_1, x_2, \ldots, x_N|\theta &\overset{iid}{\sim} p(x|\theta) \\ \theta &\sim p(\theta) \end{aligned}$$

where $\theta \in \Theta \subseteq \mathbb{R}^d$ for $d < N$. Under a Bayesian paradigm, we seek the posterior distribution $p(\theta|\{x_n\}_{n=1}^N)$ to complete inferential tasks. Moreover, using the conditional independence assumption, we look to find

$$p(\theta|\{x_n\}_{n=1}^N) \propto p(\theta) \prod_{n=1}^N p_n(x_n|\theta). \tag{1}$$

For notational convenience, define $L_n(\theta) = p_n(x_n|\theta)$ to the be contribution of the n-th data point to the full likelihood. Recall, as we assume that computing $p(x|\theta) = \prod_{n=1}^N L_n(\theta)$ will be costly, we look to avoid computing $L_n(\theta)$ whenever possible to improve efficiency.

FireFly MC assumes there exists lower bounds $0 < B_n(\theta) \leq L_n(\theta)$ for $n \in [N]$. With these lower bounds, the authors introduce latent variables $\{z_n\}_{n=1}^N$ on $\{0, 1\}$ with distribution

$$p(z_n|\theta, x_n) = \left(\frac{L_n(\theta) - B_n(\theta)}{L_n(\theta)}\right)^{z_n} \left(\frac{B_n(\theta)}{L_n(\theta)}\right)^{1-z_n}. \tag{2}$$

More compactly, the distribution of these latent variables are $z_n|\theta, x_n \sim \text{Bern}\left(\frac{L_n(\theta)-B_n(\theta)}{L_n(\theta)}\right)$. Augmenting

the likelihood with these latent variables, we see that

$$\begin{aligned}
p(\theta, \{z_n\}_{n=1}^N | \{x_n\}_{n=1}^N) &\propto p(\theta) \prod_{n=1}^N p(x_n|\theta) p(z_n|x_n, \theta) \\
&= p(\theta) \prod_{n=1}^N L_n(\theta) \left(\frac{L_n(\theta) - B_n(\theta)}{L_n(\theta)} \right)^{z_n} \left(\frac{B_n(\theta)}{L_n(\theta)} \right)^{1-z_n} \\
&= p(\theta) \prod_{n:z_n=0} B_n(\theta) \prod_{n:z_n=1} (L_n(\theta) - B_n(\theta)) \\
&= p(\theta) \prod_{n=1}^N B_n(\theta) \prod_{n:z_n=1} \frac{L_n(\theta) - B_n(\theta)}{B_n(\theta)}.
\end{aligned}$$

To see that indeed the introduction of these latent variables does not perturb the full posterior distribution notice

$$\begin{aligned}
\sum_{i=1}^N \sum_{z_i \in \{0,1\}} p(\theta) \prod_{n=1}^N p(x_n|\theta) p(z_n|x_n, \theta) &= p(\theta) \prod_{n=1}^N p(x_n|\theta) \sum_{z_n \in \{0,1\}} p(z_n|x_n, \theta) \\
&= p(\theta) \prod_{n=1}^N p(x_n|\theta)
\end{aligned}$$

This introduction of latent variables is said to turn the likelihood term $L_N(\theta)$ "on and off". The analogy for the method is that the data are flashing like fireflies as they are included or excluded in this likelihood calculation. For a skematic describing this methodology, consider Figure 1. The introduction of latent variables $\{z_n\}_{n=1}^N$ inspires the a Gibbs type sampling scheme update $\theta|z_n$ using an appropriate MCMC algorithm and then update $z_n|\theta$. We provide this full sampling procedure and a number of ways to sampling $z_n|\theta$ in Section 3.

We now analyze the lower bounds $B_n(\theta)$ and highlight two features that will maximize their utility to computational efficiency. First notice, as we look to avoid computation of $L_n(\theta)$, we hope that $z_n = 0$ for the majority of the data. Again, this it to avoid this linear slow down $O(n)$ inherent in the calculation of the full likelihood $p(x|\theta)$. Notice this corresponds with $\{L_n(\theta) - B_n(\theta)\}$ being small for all $n \in [N]$ and values of θ. Indeed, given the data x, for some $n \in [N]$

$$\mathbb{E}[z_n] = \mathbb{E}[\mathbb{E}[z_n|\theta]] = \mathbb{E}\left[\frac{L_n(\theta) - B_n(\theta)}{L_n(\theta)} \right] = \int \frac{L_n(\theta) - B_n(\theta)}{L_n(\theta)} p(\theta|x) d\theta \tag{3}$$

Therefore, to minimize the number of points evaluated in the likelihood, we seek lower bounds $B_n(\theta)$ that are tight to $L_n(\theta)$ where the posterior places higher mass. However, at this point, it appears that the problem has just been shifted to the efficient calculation of $\prod_{n=1}^N B_n(\theta)$. This leads to the second desirable properties of $B_n(\theta)$; we hope that the calculation of $\prod_{n=1}^N B_n(\theta)$ can be completed in sub-linear time and ideally is independent of N. Therefore, we can list the desirable properties of $B_n(\theta)$ as follows.

(a) $1 - B_n(\theta)/L_n(\theta)$ is small for regions of high probability of the posterior $p(\theta|x)$

(b) $\prod_{n=1}^N B_n(\theta)$ can be computed in time independent of N

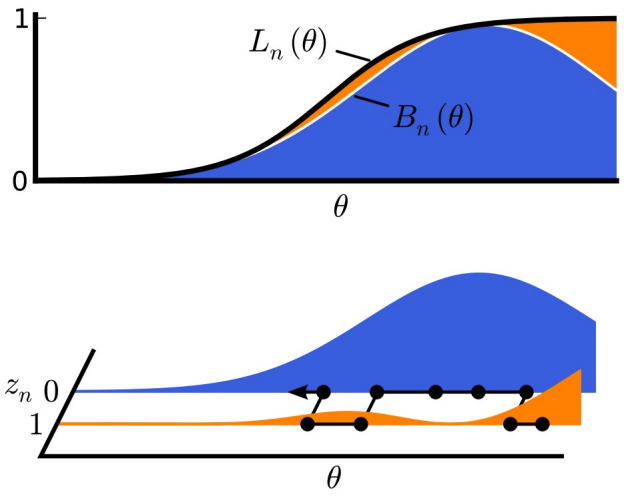

Figure 1: Figure from [1]. Expansion of the parameter space: by expanding into the latent space, we can directly exploit the structure of $B_n(\theta)$ for faster computations.

There is a natural tradeoff inherent in the preceding features. Feature (a) requires that the $B_n(\theta)$ be tight. Of course, we could choose $B_n(\theta) = L_n(\theta)$ but this second feature would not be satisfied. On the other hand, if we find bounds that can be computed in a time independent of N, they may not be tight enough to effectively "turn off" enough datapoints in the likelihood to yield computational gains.

In their paper, Maclaurin and Adams admit "...useful lower bounds can be difficult to obtain for many problems." However, they do site a very tight bound available for Logistic Regression problems and are hopeful that the growing research on variational lower bounds will yield others. Even if tight bounds exist, however, certain bounds will need to be adjusted to the posterior distribution. Therefore, a full MCMC method or variational method may need to be used *before* the use of Firefly Monte Carlo. This is a clear limitation of the methodology. We consider two bounds in our simulation studies; the one presented in [1] and a bound developed through direct expansion of the likelihood function. In the proceeding sections, we give the Firefly Monte Carlo algorithm, discuss sampling techniques for z_n, and consider the logistic regression likelihood as a concrete example.

3 Implementation Considerations

Having given an introduction to the Firefly Monte Carlo method and an analysis of the lower bounds $B_n(\theta)$, we now highlight some of the implementation details. For concreteness, we consider a Random-Walk Metropolis-Hastings Algorithm example. However, we stress that Firefly MC is applicable in any MCMC approach where repeated evaluation of the likelihood is necessary. This includes accept-reject methods with different proposals, variational methods, Metropolis-adjusted Langevin algorithm (MALA) to name but a few. For the moment, assume that we know how to update the latent variables $\{z_n\}_{n=1}^N$. Then the Firefly Monte Carlo sampling scheme is given in algorithm 1.

Algorithm 1 Firefly MC: A Random-Walk Metropolis-Hastings Algorithm Example

Input: Initial θ_0 vector, proposal variance σ^2

Output: Samples for the exact posterior $p(\theta|\{x_n\}_{n=1}^N)$

for $i = t, \ldots,$ Iters **do**
  With $(\theta^{(t-1)}, x)$, update latent variables $\{z_n\}_{n=1}^N$
  Make proposal $\theta' = \theta^{(i-1)} + \eta$ where $\eta \sim N(0, \sigma^2 I)$
  Sample $u \sim U(0,1)$
  if $\frac{\text{JointPost}(\theta', \{z_n\}_{n=1}^N)}{\text{JointPost}(\theta^{(t-1)}, \{z_n\}_{n=1}^N)} > u$ **then**
    $\theta^{(t)} = \theta'$
  else
    $\theta^{(t)} = \theta^{(t-1)}$
  end if
end for

While this algorithm simply lays out the structure of the general Metropolis-Hastings algorithm, the computational gains occur in Algorithm 2. *JointPost* is the method that calculates the carefully constructed augmented posterior given in Section 2. Notice that under certain families of lower bounds, the first line of this algorithm can be completed in a time independent of N. Moreover, we only require a loop over $z_n = 1$, which for good choices of B_n will be significantly less than N. Therefore, the calculation of the acceptance probability in this algorithm can be much faster as we do not require a linear scan of the data.

Algorithm 2 JointPost: A method to calculate the posterior with augmented likelihood

Input: Parameter vector θ and latent variables $\{z_n\}_{n=1}^N$

Output: Evaluation of the Likelihood

Set $P = p(\theta) \times \prod_{n=1}^N B_n(\theta)$ ▷ Cache sufficient statistics if possible
for $n : z_n = 1$ **do**
  $P = P \times \frac{L_n(\theta) - B_n(\theta)}{B_n(\theta)}$ ▷ Cache $L_n(\theta)$, and $B_n(\theta)$ for sampling z_n
end for

All that remains is sampling the latent variables $\{z_n\}_{n=1}^N$. Recall that $z_n \sim \text{Bern}\left(\frac{L_n(\theta) - B_n(\theta)}{L_n(\theta)}\right)$. An initial approach to updating the z_n would be to complete a true Gibbs step and sample from this Bernoulli. Notice however, that under this scheme we would need to scan over all n latent variables as calculate $L_n(\theta)$. This was precisely the problem we have been trying to avoid when calculating the full likelihood. Instead the author's suggest two methods; explicit and implicit sampling.

Explicit sampling, given in Algorithm 3, is a simple approach to this problem. In essence, at each stage, the algorithm only updates a random $\alpha\%$ of the latent variables. A clear drawback of this approach is that it restricts the mixing speed by a factor of $1/\alpha$. The authors do note, however, as most the mixing happens in the θ space as compared to the z_n space, this method works quite well in practice.

Algorithm 3 Explicit Sampling: A method to update $\{z_n\}_{n=1}^N$

Input: α resample fraction, parameter vector θ, data x, and latent variables $\{z_n\}_{n=1}^N$

Output: Updated latent variables $\{z_n\}_{n=1}^N$

for $j = 1, \ldots, \lceil N \times \alpha \rceil$ **do**
    Sample $n \sim \text{Random}(1, N)$
    Update $z_n \sim \text{Bern}\left(\frac{L_n(\theta) - B_n(\theta)}{L_n(\theta)}\right)$
end for

In the case that $\sum_{n=1}^N \mathbb{E}[z_n] << N$, randomly updating the latent variables will not be the most efficient approach as several variables will remain unchanged. As an alternative, the authors suggest a more sophisticated way of updating the latent variables, implicit sampling, that is stated in Algorithm 4. In this scheme, the authors suggest a Metropolis step with fixed proposals

$$q_{d \to b} = Q(z_n' = 1 | z_n = 0)$$
$$q_{b \to d} = Q(z_n' = 0 | z_n = 1)$$

and accepting with respect to $p(z_n | x_n, \theta)$ Regarding $q_{d \to b}$ and $q_{b \to d}$ as hyperparameters, the notation $b \to d$ means "bright to dim" and $d \to b$ means "dim to bright."

Algorithm 4 Implicit Sampling: A method to update $\{z_n\}_{n=1}^N$

Input: Parameter vector θ, data x, and latent variables $\{z_n\}_{n=1}^N$

Output: Updated latent variables $\{\hat{z}_n\}_{n=1}^N$

for $n : z_n = 1$ **do**
    Sample proposal $z_n' \sim \text{Bern}(q_{b \to d})$
    Sample $u \sim U(0, 1)$
    if $u < \min\left\{1, \frac{p(z_n' | x_n, \theta)}{p(z_n | x_n, \theta)} \frac{Q(z_n | z_n')}{Q(z_n' | z_n)}\right\}$ **then**
        Accept proposal $\hat{z}_n = z_n'$
    else
        Reject proposal $\hat{z}_n = z_n$
    end if
end for
for $n : z_n = 0$ **do**
    Sample proposal $z_n' \sim \text{Bern}(q_{d \to b})$
    Sample $u \sim U(0, 1)$
    if $u < \min\left\{1, \frac{p(z_n' | x_n, \theta)}{p(z_n | x_n, \theta)} \frac{Q(z_n | z_n')}{Q(z_n' | z_n)}\right\}$ **then**
        Accept proposal $\hat{z}_n = z_n'$
    else
        Reject proposal $\hat{z}_n = z_n$
    end if
end for

With the introduction of this proposal step, there are several computational gains the author's highlight.

First, notice if we propose no transition, $z'_n = z_n$, we do not need to calculate $p(z_n|\theta, x_n)$. Moreover, when proposing a move from $z_n = 1$ to $z'_n = 0$, it is necessary to calculate both

$$p(z'_n = 0|\theta, x_n) = \frac{B_n(\theta)}{L_n(\theta)} \quad \text{and} \quad p(z_n = 1|\theta, x_n) = \frac{L_n(\theta) - B_n(\theta)}{L_n(\theta)}$$

However, notice that both $B_n(\theta)$ and $L_n(\theta)$ were calculated and stored in *JointPost* for $n : z_n = 1$. As this is the case, we incur no additional computational cost from proposing a transition from $z_n = 1$ to $z'_n = 0$. Therefore, as it is computationally advantageous for $z_n = 0$, the authors suggest propose this transition at each stage. That is, set $q_{b\to d} = 1$, leaving $q_{d\to b}$ as the only hyperparmeter. Therefore, all that remains is to consider transitions from bright to dark. Similarly as above, when proposing a move from $z_n = 0$ to $z'_n = 1$, it is necessary to calculate both

$$p(z'_n = 1|\theta, x_n) = \frac{L_n(\theta) - B_n(\theta)}{L_n(\theta)} \quad \text{and} \quad p(z_n = 0|\theta, x_n) = \frac{B_n(\theta)}{L_n(\theta)}$$

In this case, we need to evaluate $L_n(\theta)$ and $B_n(\theta)$ for $z_n = 0$. However, if we choose $q_{d\to b}$ sufficiently small, we can further restrict the number of evaluations of $L_n(\theta)$ required at each stage of this Metropolis Algorithm. Lastly, notice due to the fact that we chosen a fixed proposal probability, we can vectorize the proposal of z'_n by using a geometric distribution. We update Algorithm 4 with these computational improvements in Algorithm 5.

Algorithm 5 Fast Implicit Sampling: A method to update $\{z_n\}_{n=1}^N$

Input: Parameter vector θ, data x, and latent variables $\{z_n\}_{n=1}^N$

Output: Updated latent variables $\{\hat{z}_n\}_{n=1}^N$

for $n : z_n = 1$ **do**
  Sample reject threshold $u \sim U(0,1)$
  if $u > q_{b\to d}$ **then** ▷ Propose Bright to Bright
    Reject move proposal $\hat{z}_n = z_n = 1$
  else ▷ Propose Bright to Dark
    Calculate acceptance probability with cached values $\alpha = \min\left\{1, \frac{B_n(\theta) q_{d\to b}}{L_n(\theta) - B_n(\theta)}\right\}$
    Sample acceptance threshold $v \sim U(0,1)$
    if $v < \alpha$ **then**
      Accept proposal $\hat{z}_n = z'_n = 0$
    else
      Reject proposal $\hat{z}_n = z_n = 1$
    end if
  end if
end for
Draw first proposed move $g \sim \text{Geom}(q_{d\to b})$ ▷ Propose Dark to Bright
while $g \leq |\{n : z_n = 0\}|$ **do**
  Calculate acceptance probability $\beta = \min\{1, \frac{L_n(\theta) - B_n(\theta)}{q_{d\to b} L_n(\theta)}\}$
  Sample acceptance threshold $u \sim U(0,1)$
  if $u < \beta$ **then**
    Accept proposal $\hat{z}_g = z'_g = 1$
  else
    Reject proposal $\hat{z}_g = z_g = 0$
  end if
  Sample next proposed state $g = g + \text{Geom}(q_{d\to b})$ ▷ Next proposed move from Dark to Bright
end while

To this point, we have introduced the backbone of Firefly MCMC as a way to improve MCMC samplers by improved likelihood evaluation.

References

[1] D. Maclaurin and R. P. Adams. Firefly monte carlo: Exact mcmc with subsets of data, 2014.