Markov Chain Monte Carlo, with Broken Samplers

Fritz Obermeyer

2016-09-12

`http://fritzo.org/notes/2016/broken.pdf`

But Why?

- Sub-sampling big data
- Latency / network failure / worker failure
- Low-precision numerics / sketching
- Miscalibrated stochastic hardware

Monte Carlo Sampling

approximates an integral

$$I = \int f(x) d\, p(x)$$

by a finite average

$$\hat{I} = \frac{1}{T} \sum_{t=1}^{T} f(X_t) \qquad\qquad X_t \sim p(-)$$

to minimize squared error

$$\mathbb{E}\left[\left(\hat{I} - I\right)^2\right] = \underbrace{\mathbb{V}\left[\hat{I}\right]}_{\text{variance}} + \underbrace{\left(\mathbb{E}\left[\hat{I}\right] - I\right)}_{\text{bias}}^2$$

What if your sampler is broken?

Let's say $\theta \sim q(-)$ is a desired parameter distribution

$$p(x) = \int p(x;\theta) d\, q(\theta)$$

but your sampler is miscalibrated according to $\theta \sim q'(-)$

$$p'(x) = \int p(x;\theta) d\, q'(\theta)$$

Then reject based on side-observations

If $q'(\theta)/q(\theta)$ is bounded, then we can approximate

$$p(x) = \int p(x;\theta)\frac{q(\theta)}{q'(\theta)}d\,q'(\theta)$$

by rejection sampling (error detection).

Then reject based on side-observations

If $q'(\theta)/q(\theta)$ is bounded, then we can approximate

$$p(x) = \int p(x;\theta) \frac{q(\theta)}{q'(\theta)} d\, q'(\theta)$$

by rejection sampling (error detection).

If $q'(\theta)/q(\theta)$ is difficult to compute, then we can reject based on summary statistics (Approximate Bayesian Computation).

Markov Chain Monte Carlo

is "Stateful Monte Carlo" with randomized transitions P_t

$$
\begin{aligned}
X_0 &= \text{fixed} \\
X_t &\sim P_t(X_{t-1}) \qquad\qquad \text{for } t \in \{1, \ldots, T\} \\
f_t &= f(X_t)
\end{aligned}
$$

This is a Markov Chain

$$
\begin{array}{ccccccccc}
X_0 & \xrightarrow{P_1} & X_1 & \xrightarrow{P_2} & X_2 & \xrightarrow{P_3} & X_3 & \xrightarrow{P_4} & \ldots \\
 & & \downarrow f & & \downarrow f & & \downarrow f & & \\
 & & f_1 & & f_2 & & f_3 & &
\end{array}
$$

Modern MCMC Methods

Optimize the policy $[P_1, \ldots, P_T]$ (including duration T) to minimize error subject to a budget constraint

$$\sum_{t=1}^{T} \mathrm{cost}(P_t) \leq \mathrm{budget}$$

Modern MCMC Methods

Optimize the policy $[P_1, \ldots, P_T]$ (including duration T) to minimize error subject to a budget constraint

$$\sum_{t=1}^{T} \mathrm{cost}(P_t) \leq \mathrm{budget}$$

Stochastic Gradient Langevin Dynamics
(Welling, Teh 2011)

Approximate Metropolis-Hastings
(Korattikara, Chen, Welling 2014)
(Chen, Fox, Guestrin 2014)

Stochastic Gradient Langevin Dynamics

Stochastic Gradient Descent
makes approximate gradient descent steps
computed from small minibatches of data.

Stochastic Gradient Langevin Dynamics

Stochastic Gradient Descent
makes approximate gradient descent steps
computed from small minibatches of data.

Observe: X_0 is horribly biased and "burn in" work is wasted.

Idea: Gradually transition from optimization to MCMC
by starting with large noisy steps, then reducing step size.

Stochastic Gradient Langevin Dynamics

Stochastic Gradient Descent
makes approximate gradient descent steps
computed from small minibatches of data.

Observe: X_0 is horribly biased and "burn in" work is wasted.

Idea: Gradually transition from optimization to MCMC
by starting with large noisy steps, then reducing step size.

- Constant per-step cost $\mathrm{cost}(P_t)$.
- Policy balances precision (small step size)
  with effective speed (large step size).

Approximate Metropolis-Hastings

Metropolis-Hastings steps *propose* a new candidate state X_t,
then randomly accept or reject that state,
usually by evaluating likelihood of data.

Approximate Metropolis-Hastings

Metropolis-Hastings steps *propose* a new candidate state X_t, then randomly accept or reject that state, usually by evaluating likelihood of data.

Observe: It's often obvious from even a little data that a candidate state should be rejected. It's wasteful to look at the entire dataset.

Idea: Allow approximate decisions by balancing decision bias with reduced per-step cost. Cheaper steps in a fixed budget allows for more samples and hence lower variance.

Approximate Metropolis-Hastings

Metropolis-Hastings steps *propose* a new candidate state X_t, then randomly accept or reject that state, usually by evaluating likelihood of data.

Observe: It's often obvious from even a little data that a candidate state should be rejected. It's wasteful to look at the entire dataset.

Idea: Allow approximate decisions by balancing decision bias with reduced per-step cost. Cheaper steps in a fixed budget allows for more samples and hence lower variance.

- Variable per-step cost $\mathrm{cost}(P_t)$.
- Policy balances per-step cost with per-step error.

How to optimize an MCMC policy

(in the presence of broken samplers)

Let X_t be hidden sampler state, Y_t be observed. Approximate

$$I \approx \hat{I} = \frac{1}{T} \sum_{t=1}^{T} f(Y_t)$$

and minimize squared error $\|\hat{I} - I\|^2$ by choosing $[P_1, \ldots, P_T]$

$$\begin{array}{ccccccccc}
X_0 & \xrightarrow{P_1} & X_1 & \xrightarrow{P_2} & X_2 & \xrightarrow{P_3} & X_3 & \xrightarrow{P_4} & \ldots \\
 & & \downarrow h & & \downarrow h & & \downarrow h & & \\
 & & Y_1 & & Y_2 & & Y_3 & & \\
 & & \downarrow f & & \downarrow f & & \downarrow f & & \\
 & & f_1 & & f_2 & & f_3 & &
\end{array}$$

What is a Markov Decision Process?

Sequential decision making
with 1-step state history.

Ignores uncertainty in sampler state.

Markov Chain	Markov Decision Process

$$X_0 \xrightarrow{P_1} X_1 \xrightarrow{P_2} X_2 \xrightarrow{P_3} X_3 \xrightarrow{P_4} \ldots$$

What is a Partially Observable MDP?

Bayesian approach to sequential decision making.

Model-based approach to Reinforcement Learning.

An impossibly intractable 1st step.

Markov Chain	Markov Decision Process
Hidden Markov Model	Partially Observable Markov Decision Process

$$\begin{array}{ccccccccc} X_0 & \xrightarrow{P_1} & X_1 & \xrightarrow{P_2} & X_2 & \xrightarrow{P_3} & X_3 & \xrightarrow{P_4} & \dots \\ & & \downarrow h & & \downarrow h & & \downarrow h & & \\ & & Y_1 & & Y_2 & & Y_3 & & \end{array}$$

How to optimize a POMDP?

1-step horizon (Greedy algorithm)

Finite-horizon (Value iteration)

Decaying horizon (Bellman's equation)

1-Step Horizon (Greedy)

Let $f_t = f(Y_t)$. Let $F_t = \sum_{s=1}^{t} f_t$ be the partial sum.

Suppose you've already sampled $(X_1, Y_1), \ldots, (X_{t-1}, Y_{t-1})$. Now choose P_t to minimize $\mathbb{V}[F_t]$.

Observe that

$$\mathbb{V}[F_t] \;=\; \frac{1}{t^2}\mathbb{V}[f_t] \;+\; \frac{2(t-1)}{t^2}\mathrm{Cov}[f_t, F_{t-1}] \;+\; \text{const.}$$

so that it's much more important to be diverse than precise.

1-Step Horizon (Greedy)

Let $f_t = f(Y_t)$. Let $F_t = \sum_{s=1}^{t} f_t$ be the partial sum.

Suppose you've already sampled $(X_1, Y_1), \ldots, (X_{t-1}, Y_{t-1})$.
Now choose P_t to minimize $\mathbb{V}[F_t]$.

Observe that

$$\mathbb{V}[F_t] \;=\; \frac{1}{t^2}\mathbb{V}[f_t] \;+\; \frac{2(t-1)}{t^2}\mathrm{Cov}[f_t, F_{t-1}] \;+\; \text{const.}$$

so that it's much more important to be diverse than precise.

Problem: This completely ignores future samples.

Value Iteration

1. Solve for final step P_t^* as a function of all previous Y_t, by simply minimizing final variance.
2. Solve for penultimate iteration P_{t-1}^* by minimizing expected error, assuming optimal solution P_t^* from step 1 will be used after P_{t-1}^*.
3. etc.

... until first step is reached.

Value Iteration

1. Solve for final step P_t^* as a function of all previous Y_t, by simply minimizing final variance.
2. Solve for penultimate iteration P_{t-1}^* by minimizing expected error, assuming optimal solution P_t^* from step 1 will be used after P_{t-1}^*.
3. etc.

…until first step is reached.

Problem: Doesn't work for variable-cost steps (variable T).

Bellman's Equation for discounted loss

Consider an exponentially decaying objective function, corresponding to the estimator

$$F_t = \frac{F_{t-1} + \gamma(P_t) f(Y_t)}{1 + \gamma(P_t)}, \text{ where } \qquad \gamma(P_t) = \frac{\mathrm{cost}(P_t)}{\mathrm{budget}}$$

Assuming a bound k on history relevance, the optimal policy is a fixed point of

$$P^*(Y_{t-k}, \ldots, Y_{t-1}) = \arg\min_{P(-)} \mathbb{V}[F_{t-1} + \gamma(P) f(h(P(X_{t-1})))]$$

where the distribution of Y, X, F depends on $P(-)$.

Summary

1. Recognize sequential, stateful nature of problem.
2. Minimize squared error of integral, as in MCMC.
3. Model sampler as a POMDP with hidden noise.
4. Find a policy that balances
   Exploitation (estimating the quantity of interest) with
   Exploration (calibrating and re-calibrating the sampler).

References

Max Welling, Yee Whye Teh (2011)
“Bayesian Learning via Stochastic Gradient Langevin Dynamics” (pdf)

Anoop Korattikara, Yutian Chen, Max Welling (2014)
“Austerity in MCMC Land: Cutting the Metropolis-Hastings Budget” (pdf)

Tianqi Chen, Emily B. Fox, Carlos Guestrin (2014)
“Stochastic Gradient Hamiltonian Monte Carlo” (pdf)