The No U-Turn Sampler

Bayesian Data Analysis

Steve Buyske

A confession

- We talked about the Metropolis algorithm as our only example of a Markov Chain Monte Carlo algorithm.
- It turns out it has shortcomings in practice, especially in high dimensions.
  - It can be improved with asymmetric acceptance probabilities (the Metropolis Hastings algorithm) and
  - adaptive jump sizes (the adaptive Metropolis Hastings algorithm), but those only improve things so much.
  - Even so, in high dimensions there will be many rejected jumps, or it will take a *very* long time to explore all of the distribution.

- Another algorithm you may come across is the Gibbs Sampling algorithm:
  - go through each dimension one at a time, and use conjugate distributions whenever possible
- The state-of-the-art general purpose MCMC algorithm for Bayesian analysis is known as the *No U Turn Sampler*, or NUTS.

Hamiltonian Monte Carlo

- The NUTS algorithm is based on the Hamiltonian Monte Carlo algorithm, so let’s start with that.
- One key idea is to take use lots of computation in picking the next proposed draw in order to get draws that both
  - are mostly accepted and that
  - move around the distribution.
- The other key idea is to steal ideas from physics to do so.

Hamiltonian Monte Carlo

- First, for computational ease the HMC uses the -log of the posterior density
  - If you have two parameters, then think of this as an upside bowl of some sort (using the log keeps it from beeing too pointed).
- Imagine a marble at the current draw, and imagine that you are going to flick in across the surface of the bowl.
  - Both the direction and the force you use will be random.
  - After a fixed amount of time, stop the marble—that's your proposal for the next draw.
- Calculating the path takes some work, but it is done as a series of steps, each step based on a calculation of the posterior at that step.
- The result is a proposal that both goes a long way and has a high probability of acceptance.

The No U Turn Sampler

- The No U Turn Sampler uses the Hamiltonian Monte Carlo algorithm, but does each flick in two directions, the original and its opposite, to keep from having the equivalent of going around all the way around the rim of the bowl.
- Stan, and therefor the `rstanarm` and `brms` packages, use NUTS, which requires almost no intervention on our parts.

“There were 15 divergent transitions after warmup”

- A warning like “There were 15 divergent transitions after warmup” indicates that the path that the NUTS algorithm diverged from what it expected 15 times.
- That’s not good—it means that you might be missing parts of the posterior.

- One approach is to use the `adapt_delta` argument, which is the average proportion of accepted proposals.
  - The default is 0.8.
  - I usually first try `adapt_delta = 0.99`, then `adapt_delta = 0.999`, and so on
  - Larger values mean a smaller step size and a slower run time.
- If that doesn't work, then I try different priors, especially something like `prior_covariance = decov(shape = 2)` or `set_prior("lkj(2)", class = "cor")`.
  - The warning often comes using hierarchical models, and these changes to prior make more extreme correlations among the group-level parameters less likely.