Lecture 13, Tuesday w7, 2014-10-28

Markov chain Monte Carlo (MCMC)

We finished last time noticing that picking good step-sizes is important for Metropolis methods.

MCMC methods can apply a series of different transition operators in sequence, or at random. For example, Gibbs sampling uses a sequence (or random choice) of transition operators, each of which only updates one particular variable. We can also use a random choice of step size.

Another way to make us more robust to a fixed choice of step size is slice sampling. See one of the short readings below for a detailed explanation.

Running MCMC methods:

It can be a good idea to do a few multiple runs. 1) if chains get trapped in different local modes of the posterior, we might notice that multiple runs disagree. 2) If we have independent estimates from different runs, it's easy to estimate a standard error on the mean of their estimates. (For more sophisticated estimates, see the R-CODA R package.)

Look at trace plots of parameters or predictions over time, to spot any problems. Some practitioners will discard any transient "burn-in" period they see, before the Markov chain reached equilibrium. Strictly speaking this procedure isn't necessary: any estimates using the whole run are still consistent.

Knowing whether our Markov chain has adequately explored the space of plausible parameters in an inference problem is usually hard. A lot of hope is involved in many applications. As a minimum, we can run our code on synthetic data, to see if it works correctly when we know the true underlying explanation of the data.

Examples (pp1–19 of "Deterministic approximations" slides)

I showed you an example of MCMC (slice sampling) working in practice. This example is non-examinable.

Take home message: In complicated posteriors, it may be hard to summarize the distribution in any simpler way than a set of posterior samples. For simple posteriors, a maximum likelihood or MAP estimate of the parameters may be pretty good, and an optimizer would return that much quicker than MCMC.

We'd like an error bar on any estimates we make though: we should represent at least the width of our posterior. Many posteriors are approximately Gaussian (but not all! I showed a couple of plots of exceptions). An optimizer can find the mode of that approximate Gaussian. Now we also want the width of the Gaussian. We can estimate the width much more cheaply than MCMC (next lecture!).

Recommended reading

For slice sampling, read one of:

My PhD thesis, pp39–40, attaches more words to the same diagrams as seen in lectures. But has little other detail. http://homepages.inf.ed.ac.uk/imurray2/pub/07thesis/

MacKay's textbook: pp365–377 (down to 'Computer-friendly slice sampling')

Murphy's textbook: Section 24.5.2 pp864–866.

Optional extras

Previous years have covered "Hamiltonian Monte Carlo" (HMC), (see e.g., MacKay p387). This method uses gradients of the log posterior to construct a dynamical system that will make larger updates than typically possible with the methods discussed in lectures. I thought it was a bit much to include, so it's non-examinable. If you wanted to do MCMC on the parameter posterior of a neural network of moderate size, this method is what you'd need though. The method isn't hard to implement, but making some of the choices required to run it can need some experience.

Software that makes HMC work well on Bayesian neural networks is: http://www.cs.toronto.edu/∼radford/fbm.software.html