Lecture 7.5: Advanced Markov Chain Monte Carlo

Professor Alexander Franks

2020-11-25

Challenges in MCMC

- Modern models often have *many* parameters. Large models pose a challenge for MCMC.
- When there are thousands or more parameters
  - MCMC may take a long time to conververge to the stationary distribution
  - In Metropolis-Hastings we have many tuning parameters for the proposal distribution
  - Gibbs sampling has no tuning parameters, but does not work well for highly correlated posterior distributions
- In general, MCMC is very slow relative to optimization methods

Modern MCMC

- Gibbs and Metropolis samplers have a "random walk" behavior
    - Induces autocorrelation
    - Makes it difficult to explore the posterior space
- Hamiltonian Monte Carlo (HMC) borrows an idea from physics to reduce this problem

HMC

- Imagine a marble on a frictionless surface. The location of the marble is the current value of θ_t
- The negative posterior density is the "height" of the surface
- Each iteration we flick the marble with some velocity in a random direction
- Regions of high posterior density are like "wells"

HMC

- For Metropolis-Hastings we only need to be able to evaluate the posterior at each location
- For HMC we need the gradient (derivative) of the posterior as well
    - Determines where the marble rolls
- In physics the Hamiltonian is the sum of the kinetic energies, plus the potential energy of the particles
    - As our proposal, we randomly sample a momentum for the marble and update its position accordingly
    - Can think of HMC as the MH algorithm with a very clever jumping/proposal rule

HMC

Try out HMC at:

https://chi-feng.github.io/mcmc-demo/app.html

- Choose "HamiltonianMC" algorithm
- Experiment by sampling from different target distributions
- Compare to the Random Walk Metropolis

Approximate Inference

- MCMC can be very slow in high dimensional problems
- Idea: find a distribution that is easy to sample from which closely approximate $p(\theta \mid y)$
- A couple of examples
    - Laplace Approximation
    - Variational Bayes

Laplace Approximation to the Posterior

- Approximate the posterior distribution using a multivariate normal distribution
- When we have a lot of i.i.d. observations, the posterior will be approximately normal
- Center the normal at the mode of the posterior
- Compute the (co)variance of the normal by computing the second derivative / hessian of the posterior at the mode

Laplace Approximation

- Approximate the posterior distribution using a normal distribution
- When we have a lot of i.i.d. observations, the posterior will be approximately normal
- Center the normal at the mode of the posterior
- Compute the (co)variance of the normal by computing the second derivative / hessian of the posterior at the mode

Laplace Approximation

- Let $\tilde{\theta}$ be the mode of the of the posterior distribution
- Use a Taylor Series approximation the log-posterior around the mode is
  - $logP(\theta \mid y) \approx logP(\tilde{\theta} \mid y) - 1/2(\theta - \tilde{\theta})H(\theta - \tilde{\theta})$
  - $H = \frac{d^2}{d\theta^2} logp(\theta \mid y)$
  - Note, linear term falls out because derivative at the mode is zero
- $p(\theta \mid y) \approx N(\tilde{\theta}, I(\theta)^{-1})$

Finding the mode of the posterior distribution

- Calculus
  - Take the log
  - Differentiate, set to zero and solve
- Computational
  - `optim` in R for one dimensional posteriors
  - `optimise` in R for multivariate p

Variational Bayes

- Find even better approximations
- Let θ be d dimensional parameter vector
- Let $\epsilon \sim MVN_d(0, \Sigma)$
- Let g_λ be a class of flexible functions parameterized by λ
    - Neural networks!
- Solve an optimization problem:

$$\arg\min_\lambda \operatorname{dist}(p(\theta \mid y)), p(g_\lambda(\epsilon)))$$

    - Minimize the "distance" between the true posterior and the approximate one.
- Sample ϵ from a multivariate normal. $g_\lambda(\epsilon)$ will be a sample from something close to $p(\theta \mid y)$