10/23/19

Physics 8805: Learning from Data: Lecture 17

Notebooks for today: (also MCMC visualization from Cameron)
- mcmc-parallel_tempering.ipynb
- Liouville_Theorem_visualization.ipynb, Orbital_eqs_with_different_algorithms.ipynb
- PyMC3_intro.ipynb

Follow-ups:
- Summarize computational possibilities for evidence and why it is hard ⇒ (90)

- Parallel tempering recap. → (95)
  - Review (92)-(97)
  - particularly consider the evidence calculation on (99).

- Note: Changing the environment/kernel for Jupyter notebook

Continue with HMC intro.

Parallel tempering summary points:
- Simulate N replicas of a system at different $\beta = 1/T$, where the temperature dependent posterior is $\log p_\beta(\theta|D,I) = C + \beta \log p(D|\vec{\theta},I) + \log p(\theta|I)$
  - The temperatures range from β small (T large) to $\beta = 1$, which is the result we are trying to find. The user chooses the β values.
  - $\beta = 0$ samples the prior, so it is spread over accessible parameter space
- The N chains run in parallel. A swap of configurations is proposed at random intervals between adjacent chains. A Metropolis-like criterion is used to decide whether the swap is selected or not.
- The evidence can be calculated numerically by thermodynamic integration use the results at all the temperature in a numerical quadrature formula.

10/23/19

- We've seen some different strategies for sampling difficult posteriors, such as an affine-invariant sampling approach (emcee) and a thermodynamic approach (parallel tempering).

- One of the most widespread techniques in contemporary samplers is Hamiltonian Monte Carlo, or HMC.
  - We'll look at some visualizations as motivation, then consider some examples using pymc3.

- We return to the excellent set of interactive demos by Chi Feng at https://chi-feng.github.io/mcmc-demo/ and their adaptation by Richard McElreath at http://elevanth.org/blog/2017/11/28/build-a-better-markov-chain/.
These are linked on the 8805 Canvas visualization page.

- The McElreath blog piece forcefully advocates abandoning Metropolis-Hastings sampling in favor of HMC. Let's take a look.
  - First recall the Random Walk MH.
    1) Make a random proposal for new parameter values (a step in parameter space, indicated by an arrow).
    2) Accept or reject the proposal based on a Metropolis criterion.
  - This is diffusion (random walk) so not efficient in exploring the space and needs special tuning to avoid too many rejections.
  - The donut shape in the simulation is common in higher dimensions and it is difficult to explore. (Eg, consider a multidimensional uncorrelated Gaussian. In spherical coordinates the distribution $\propto r^n e^{-r^2/2\sigma^2}$, which is peaked away from $r=0$.)

10/23/14

- Now consider the "Better Living through Physics" part ⇒ an HMC simulation.
  - The idea is that we map our parameter vector $\vec{\theta}$ to a particle in an n-dimensional space. The surface is an n-dimensional (inverted) bowl with the shape given by minus-log (target distribution), where the target distribution is the posterior.
  - Treat the system as frictionless. "Flick" the particle in a random direction, so it travels across the bowl.
  - See the simulation: the little gray arrow is the flick. After the particle travels some distance, decide whether to accept. Most endpoints are within a high probability region, so a high percentage is accepted.
  - Chains can get far from the starting point easily ⇒ efficient exploration of the full shape.
  - More calculation along the path, but fewer samples ⇒ this is typically a winning trade-off.
  - Check the donut cases ⇒ works very well!

- There is a further improvement called NUTS – "no-U-turn sampler".
  - The idea is to address the problem that HMC needs to be told how many steps to take before another flick.
    - too few steps ⇒ samples are too similar
    - too many steps ⇒ also too similar

Class: What are the drawbacks of the simulation paths in "Stan's NUTS"?

- Solution is NUTS.
  - adaptively finds a good number of steps.
  - simulates in both directions to figure out when the path turns around (U-turn) and stops it.
  - There are other adaptive features – see the documentation.

- Note that NUTS still has trouble with multimodal targets
  ⇒ can explore each high probability area, but has trouble going between them.

10/23/19

References for HMC:
- "Hamiltonian Monte Carlo Explained" by Alex Rogozhnikov
- "MCMC Using Hamiltonian Dynamics" by Radford Neal.

- The basic idea is to translate a pdf for the distribution desired into a potential energy function and then add a (fictitious!) momentum variable. In the Markov chain at each iteration, one resamples the momentum (flick!), creates a proposal using Hamiltonian dynamics, and then does a Metropolis update.

- Ok, so recall Hamiltonian dynamics, now applied to a d-dimensional position vector q and a d-dimensional momentum vector p
  $\Rightarrow$ 2*d phase space for Hamiltonian $H(q,p)$.

- The Hamilton equations of motion describe time evolution:
  (remember total and partial derivatives (what is held fixed)) $\Rightarrow$ $\frac{dq_i}{dt} = \frac{\partial H}{\partial p_i}$ and $\frac{dp_i}{dt} = -\frac{\partial H}{\partial q_i}$ $i = 1, \dots, d$
  $\Rightarrow$ these map states at t to states at $t+s$.

- We take the form of H to be $H(q,p) = U(q) + K(p)$
  - potential energy $U(q)$ is minus the log probability density of the distribution for q we seek to sample.
  - $K(p)$ is the kinetic energy
    $$K(p) = (p^T M^{-1} p)/2$$
    where M is a symmetric, positive "mass matrix", typically diagonal and even $M = I_d$ (proportional to identity matrix in d-dimensions)
    - This is minus the log probability density (plus a constant) of a Gaussian with zero mean and covariance matrix M.

- What are we going to do with this? We consider a canonical distribution
  $$P(q,p) = \frac{1}{Z} e^{-H(q,p)/T} = \frac{1}{Z} e^{-U(q)/T} e^{-K(p)/T}$$
  so q and p are <u>independent</u>. We are interested in q; p is fake to make things work.
  Usually $U(q)$ is a posterior: $-\log[p(q|D)\,p(q)]$ where $q \to \theta$.

10/23/19

Two steps of the HMC algorithm:

1) New values for the momentum variables are randomly drawn from their Gaussian distribution, independent of current position values.
   - This means p_i will have mean zero and variance M_i if M is diagonal.
   - q isn't changed, p is from the correct conditional distribution given q, so the canonical joint distribution is invariant.

2) Proposal from Hamiltonian dynamics for a new state.
Simulate from (q,p) with L steps of size ε.
At the end, the momenta are flipped in sign and the new proposed state (q^*, p^*) is accepted with probability

$$\min\left[1, e^{-H(q^*,p^*)+H(q,p)}\right] = \min\left[1, e^{-U(q^*)+U(q)-K(p^*)+K(p)}\right]$$

(ΔS → with T=1)

   - The momentum flip makes the proposal symmetric, but not done in practice.
   - So the probability distribution for (q,p) jointly is (almost) unchanged because energy is conserved, but in terms of x & q we get a very different probability density.

You can show that HMC leaves the canonical distribution invariant because detailed balance holds, which is what we need. It will also be ergodic ⇒ it doesn't get stuck in a subset of state space but samples all.

Essential features:
- Reversibility needed so that desired distribution is invariant.
- Conservation of the Hamiltonian (which is the energy here)
- Volume preservation — preserves volume in (q,p) space — this is Liouville's Theorem. (If we take a cluster of points and follow their time evolution, the volume they occupy is unchanged.)
  ⇒ This is critical because a change in volume would mean we would have to make a nontrivial adjustment to the proposal (because the normalization Z would change).