MSA101/MVE187 2017 Lecture 13

Petter Mostad

Chalmers University

October 10, 2017

Inference using simulation in graphical models

- Exact inference is in graphical models can easily become intractable, in particular for moderately-sized or large networks, and in particular when the graph is not sparse, or contains long loops.
- Various types of approximate methods can then be used.
- One very attractive method is simulation from the posterior using Gibbs sampling: The conditional distribution needed in the Gibbs samples can often be derived easily from the network.

Inference using Gibbs sampling in Bayesian Networks

- The conditional distribution for a variable in a Bayesian network involves its parents and its immediate decendants.
- The conditional distribution can be found with a single application of Bayes theorem.
- The computation is particularly simple when the distribution of the variable given its parents is conjugate to the likelihood defined by its decendants. In other cases, various types of simulation methods (like rejection sampling) can be employed, based on the nature of the distributions.
- Widely used programs like BUGS (WinBugs, OpenBugs), Jags (Just Another Gibbs Sampler), and Stan offer "black box" implementations of Gibbs sampling on wide classes of Bayesian Networks.

Inference using Gibbs sampling in Markov networks

- Markov networks are often specified by listing exactly the conditional distributions used in a Gibbs sampler.
- Gibbs sampling using these conditional distributions will work, but convergence speed may be a problem.
- When a Markov network is a component in a more complex model, it may be convenient to use Gibbs sampling for the whole model, including for inference for hyperparameters.

Causal networks

- Bayesian networks, like stochastic models in general, have nothing to do with causality. ("Correlation is not causation").
- However, one may *add* the following interpretation to a Bayesian network, to obtain a causal network: If one *intervenes* at a node (which is different from *observing* the value of the node) the probability distribution for the remaining nodes is given by the Bayesian network obtained from the old one by removing the conditional distribution for the intervention node.
- Example, with rain and umbrella.
- In general, one would like to infer causal networks from data: Methods may be difficult and controversial.

Some software for graphical model inference

- Exact inference: Genie/Smile: Stand-alone programs for inference. Hugin: Commercial software.
- Simulation inference: BUGS. Jags, Stan, ...
- R packages. Also for *learning the network*.

The slice sampler

- Idea: Do Gibbs sampling from "the area under the density curve".
- More formally, simmulate from the density

$$f(x, u) = I(0 < u < f_x(x))$$

- The density needs to be known only up to a constant.
- The challenge is to simulate x uniformly on $\{x : f_x(x) > u\}$.
- Example 7.10 in RC.
- Generalization: When $f(x) = \prod_{i=1}^{n} g_i(x)$ we can define the joint density

$$h(x, u_1, \ldots, u_n) = \prod_{i=1}^{n} I\left(0 < u_i < g_i(x)\right)$$

- Simulate x uniformly on $\cap_{i=1}^{n} \{x : g_i(x) > u_i\}$.

Example: Logistic regression

(Example 7.11 in RC, but book contains errors)

- Data $(x_1, y_1), \ldots, (x_n, y_n)$; $y_i \sim \text{Bernoulli}(p(x_i))$; $p(x_i) = \frac{\exp(a+bx_i)}{1+\exp(a+bx_i)}$
- Using a flat prior, simulate from posterior for (a, b) using slice sampling.
- $\pi(a, b \mid \text{data}) \propto \prod_{i=1}^{n} \left(\frac{\exp(a+bx_i)}{1+\exp(a+bx_i)}\right)^{y_i} \left(\frac{1}{1+\exp(a+bx_i)}\right)^{1-y_i} = \prod_{i=1}^{n} \frac{\exp(a+bx_i)^{y_i}}{1+\exp(a+bx_i)}$
- For $i = 1, \ldots, n$, simulate $u_i \sim \text{Uniform}\left[0, \frac{\exp(a+bx_i)^{y_i}}{1+\exp(a+bx_i)}\right]$.
- Simulate (a, b) uniformly on set satisfying, for all i, $\frac{\exp(a+bx_i)^{y_i}}{1+\exp(a+bx_i)} > u_i$.
- Corresponds to $a + bx_i > \log(u_i/(1-u_i))$ for i with $y_i = 1$, and $a + bx_i < \log((1-u_i)/u_i)$ for i with $y_i = 0$.
- Extend the Gibbs sampling, simulating for a

$$a \sim \text{Uniform}\left[\max_{y_i=1}\left(\log\frac{u_i}{1-u_i} - bx_i\right), min_{y_i=0}\left(\log\frac{1-u_i}{u_i} - bx_i\right)\right]$$

Logistic regression, cont.

- For b, we need to be more careful, simulating b uniformly in the interval of numbers
  - Greater than $\left(\log \frac{u_i}{1-u_i} - a\right)/x_i$ for i with $y_i = 1$ and $x_i > 0$.
  - Smaller than $\left(\log \frac{u_i}{1-u_i} - a\right)/x_i$ for i with $y_i = 1$ and $x_i < 0$.
  - Smaller than $\left(\log \frac{1-u_i}{u_i} - a\right)/x_i$ for i with $y_i = 0$ and $x_i > 0$.
  - Greater than $\left(\log \frac{1-u_i}{u_i} - a\right)/x_i$ for i with $y_i = 0$ and $x_i < 0$.
- See code `mychallenge.R` on course home page for implementation and example.
- NOTE: a and b are highly correlated! Convergence improved by centering data!
- Errors in RC:
  - Confusion beween (a, b) and (α, β)
  - Second and fourth formulas on page 220 are wrong.
  - No need to use a prior for a and b to get this to work; use centering instead.

Reparametrizations

- Because the Gibbs sampler changes some parameters at the time, its properties can be very sensitive to a reparametrizatioin.
- Generally, re-parametrizations that diminish correlation between variables will benefit the convergence speed!
- A way to improve convergence speed may be to simply make sure observed data values average to zero (and have similar variance).

Convergence in practice for MCMC sampling

- How many values do we need to simulate?
  - Convergence in distribution: A sample where every value is approximately sampled from the target distribution.
  - Convergence of averages, i.e., expectations. Monte Carlo Integration.
  - An apprroximate i.i.d. sample.
- Except for very special circumstances it is very difficult to obtain precise and useful statements about the degree of convergence.
- Some general advice:
  - Remove the first part of the simulated values (the "burn-in") before making inference.
  - You may remove all but every k'th simulated value ("thinning"). Only useful if you need an approximate i.i.d. sample. Check the autocorrelation!

Graphical and numerical monitoring of the chain

- Monitoring chain values, and cumulative averages.
- Non-parametric tests of stationarity.
- Effective sample size.
- Use several parallell chains!
  - Gives direct and intuitive ways to check if the chains have "mixed".
  - Specialized tests for convergence have been developed, comparing the variance between and the variance within chains. See RC.
  - However, your starting values need to be spread out so that all parts of the posterior density are visited. May be difficult in high dimensions.
- Consider your specific model to see if there are reasons to suspect non-convergence.

The coda R package

- Provides a convenient implementation of many proposed convergence monitoring methods
- Output from your own MCMC implementation can be converted to appropriate objects with the `mcmc()` and the `mcmc.list()` functions.
- Standard functions like `plot` and `summary` now give output relevant to the MCMC setting.
- A large number of specialized monitoring tools are also implemented.

Using improper priors

- It is quite useful to use improper priors: Completely OK as long as the posterior becomes proper.
- Proving that the posterior is proper may be difficult and may unfortunately be forgotten about.
- The output of a Metropolis-Hastings or Gibbs algorithm applied to an improper distribution will often look like some kind of random walk. HOWEVER; it may not be direcly obvious to spot the problem from the output!
- Examples 7.18, 7.19 in RC

Hybrid Gibbs Metropolis-Hastings methods

- The Metropolis-Hastings / Gibbs framework is very flexible: Often you can mix and match together many different alternative steps that the algorithm can switch between. As long as you can prove
  1. The target distribution is stationary for each (combination of) step(s).
  2. The Markov chain defined by the whole algorithm has a unique stationary distribution.

  you are OK.
- The objective of using hybrid methods is generally to speed up convergence.
- A common strategy may be to intersperse Gibbs sampling steps with Metropolis-Hastings specialized steps that change many variables simultaneously, to "jump" from one area with high likelihood to another.
- Another strategy may be to let the computer select randomly at each step between using a step from one of k possible Metropolis-Hastings algorithm for the target distribution. May be faster than figuring out which one has good convergence properties various situations.