MSA101/MVE187 2018 Lecture 8

Petter Mostad

Chalmers University

September 27, 2018

Checking convergence

- We know the results from MCMC will be correct in the limit when the sample size $\to \infty$.
- Only in very special cases (e.g. using "coupling") do we know how big the sample size needs to be to get a certain accuracy.
- In practice "checking convergence" means checking for signs of non-convergence or slow convergence (slow "mixing"):
  - Monitor variable values and cumulative averages.
  - Check autocorrelations for variables.
  - Check acceptance rates (but higher is not always better, unless you are using independent proposals!)
  - Use multiple starting points for the MCMC chain!
  - Use multiple parallell chains, and compare variace within chains with variance between chains! (Special tests have been developed).
- An important ingredient is to *understand* your model and your posterior, so that you can guess what might cause convergence problems, and check for such problems.

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 the generated sample

- Remove the "first part" of the chain (the "burn-in").
- To obtain a sample that is approximately i.i.d., one may use "thinning": Keeping only each k'th simulated value in the chain.
- Not necessary unless you need the i.i.d. property! ("Effective sample size")

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.
- Replacing a variable x with $\log(x)$ may make posterior densities more symmetric, improving convergence.
- A simple way to improve convergence speed may be to make sure observed data values average to zero and have similar variance.

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

Missing data

- Idea: Simulate the missing data given the parameters, and then simulate the parameters given the missing data: Gibbs sampling idea!
- Example: Censored data, for example in survival analysis: We want to learn about density $f(\cdot \mid \theta)$ from sample where $x_1, \ldots, x_k$ are observed values and $c_1, \ldots, c_n$ are observations that the corresponding x_i is greater than some a_i. The likelihood becomes

$$\pi(x_1, \ldots, x_k, c_1, \ldots, c_n \mid \theta) = \prod_{i=1}^{k} f(x_i \mid \theta) \prod_{i=1}^{n} (1 - F(a_i \mid \theta))$$

  where $F(\cdot \mid \theta)$ is the cumulative density.
- Simulating alternatively the missing data and distribution for the parameters given *all* data may be easier than dealing with the likelihood above.
- Example 7.6 in RC: A Normal$(\theta, 1)$ model with data truncated at a.

Importance sampling

- MC integration computes

$$\int h(x)f(x)\,dx$$

where $f(x)$ is a probability density function, by simulating $x_1, \ldots, x_m$ according to f and taking the averages of $h(x_1), \ldots, h(x_m)$. The result has accuracy $\sqrt{Var(h(X))/m}$.

- Instead, we may re-write the integral as

$$\int \left[\frac{h(x)f(x)}{g(x)}\right] g(x)\,dx$$

and simulate x_i according to g and taking the averages of $h(x_1)f(x_1)/g(x_1), \ldots, h(x_m)f(x_m)/g(x_m)$.

- A good idea if $Var(h(X)f(X)/g(X))$ is much smaller than $Var(h(X))$.