Chapter 5

Bayesian computation

- The problem of high dimensional integration
- Conjugate priors
- Importance sampling
- Markov Chain Monte Carlo
  - Metropolis-Hastings algorithm
  - Gibbs sampler
  - MCMC diagnostics

High dimensional integration

Recall the formula for the posterior distribution:

$$\pi(\theta \mid y) = \frac{f(y,\theta)}{f(y)} = \frac{f(y \mid \theta)\pi(\theta)}{f(y)}$$

In general θ is a vector of unobserved quantities ("parameters") and in some models its dimension could be in the 100's or 1000's.

Several aspects of Bayesian inference require integration with respect to $\pi(\theta \mid y)$ and θ.

High dimensional integration

Doing model comparison requires calculation of model likelihoods, *f(y)* .

$$f(y) = \int f(y,\theta)d\theta = \int f(y \mid \theta)\pi(\theta)d\theta$$

A bit of calculus shows that *f(y)* can be expressed as an integral with respect to the posterior density,

$$f(y) = \left(\int \frac{1}{f(y \mid \theta)} \pi(\theta \mid y)d\theta \right)^{-1}$$

High dimensional integration

Point estimation and interval calculation require the (marginal) density of individual elements of θ, which again requires integration with respect to the posterior density.

For example, the Bayes estimator of θ_i (the ith element of θ) is

$$E(\theta_i \mid y) = \int \theta_i \pi(\theta \mid y)d\theta \equiv \int \theta_i \pi(\theta \mid y)d\theta_1 ... d\theta_p$$
$$= \int \theta_i \pi(\theta_i \mid y)d\theta_i$$

where

$$\pi(\theta_i \mid y) = \int \pi(\theta \mid y)d\theta_1 d\theta_2 ... d\theta_{i-1} d\theta_{i+1} ... d\theta_p$$

Conjugate priors

In simple models the integration problem can be avoided by choosing a particular type of prior.

- the prior density of θ, $\pi(\theta)$, is conjugate if $\pi(\theta \mid y)$ will belong to the same "statistical family".
  - This was the case with the IID Normal example where both prior and posterior were normally distributed.

Sampling the posterior

Instead of trying to tackle the integration problem – simulate a whopping great sample (e.g., 10,000's) from the (joint) posterior $\pi(\theta \mid y)$.

- The Bayes estimator of a parameter can then be approximated (to arbitrary precision) from its average over the samples from the posterior.
- Intervals can be obtained from drawing histograms of the sample values.
- Model likelihoods can be approximated (to arbitrary precision) by the inverse of the average of $1/f(y \mid \theta)$ over the samples from the posterior.

Sampling the posterior

Two widely used approaches for sampling from $\pi(\theta \mid y)$:

Importance sampling:

- Simulate values from a simple density that is similar to $\pi(\theta \mid y)$, and do an adjustment (re-weighting).

Markov chain Monte Carlo (MCMC)

- Simulate from a Markov chain which has $\pi(\theta \mid y)$ as its equilibrium distribution.

Importance sampling

Example: The Bayes estimator of θ_i can be written

$$E(\theta_i \mid y) = \int \theta_i \pi(\theta \mid y) d\theta = \int \frac{\theta_i \pi(\theta \mid y)}{p(\theta)} p(\theta) d\theta$$

where $p(\theta)$ is any density that is easy to sample from.

The statistical interpretation of the above formula is that the Bayes estimator can be obtained by sampling (tens of thousands of) of θ values from the distribution with density function $p(\theta)$, and calculating the average of

$$\frac{\theta_i \pi(\theta \mid y)}{p(\theta)}$$

Markov chain Monte Carlo

A Markov chain refers to a random process where the value at time (or iteration) *t* depends on the value at time *t-1* but not on any earlier values.

Methods such as the Metropolis-Hastings algorithm implement a Markov chain to generate a (very long) sequence of random θ values. The algorithm is constructed such that the generated values come from the desired "target" distribution, which in the Bayesian context is $\pi(\theta \mid y)$.

Metropolis-Hastings algorithm

If we have just generated value $\theta^{(k)}$, the Metropolis-Hastings algorithm proceeds by using a simple "proposal density" $p(\theta, \theta^{(k)})$ to suggest that the next value be θ^*. This suggested value is accepted with probability

$$\Pr(set\ \theta^{(k+1)} = \theta^*) = \min\left(1, \frac{p(\theta^{(k)}, \theta^*)\pi(\theta^* \mid y)}{p(\theta^*, \theta^{(k)})\pi(\theta^{(k)} \mid y)}\right)$$

If θ^* is not accepted then we "stay put", and set $\theta^{(k+1)} = \theta^{(k)}$.

Gibbs sampler

The Gibbs sampler is a special case of the Metropolis-Hastings algorithm in which (the vector) $\theta^{(k+1)}$ is obtained from $\theta^{(k)}$ by updating the vector elements one at a time.

WinBUGS is the Windows based version of the **B**ayesian analysis **U**sing **G**ibbs **S**ampler software.

MCMC software

The BUGS software (including WinBUGS) undoubtedly provides the easiest and most widely used implementation of MCMC. It is reasonably general, but there are some classes of models that it can not cope with.

Other software includes routines written in a variety of languages (e.g., C++). The Automatic Differentiation Modeler Software (ADMB) is more powerful and general than BUGS, but is non-free and is far harder to use. (ADMB can also do pure maximum likelihood analysis, and hence permits both Frequentist and Bayesian modeling.)

MCMC properties

- The MCMC algorithm needs to "burn in" – to become independent of the starting value.
  - Usually enough to discard the first few thousand values.
- Successive values may be highly correlated. That is, the sequence of values from the chain are not generally independent.
- Badly behaved posterior densities (e.g., bi-modal) may result in mixing problems whereby some parts of $\pi(\theta \mid y)$ are not properly sampled.

MCMC diagnostics

Several software packages are available for checking the MCMC output.

- WinBUGS provides exploratory diagnostic tools and windows for seeing trace plots and histograms of the sampled values.
- CODA (Convergence Diagnostics Analysis) and BOA (Bayesian Output Analysis) packages are available for R and Splus on a variety of platforms. These perform more sophisticated checks.
  - WinBUGS provides the option to save the MCMC output in the form required by CODA.