BASICS OF MARKOV CHAIN SIMULATION

(The ratio $r$ is always defined, because a jump from $\theta^{-1}$ to $\theta^0$ can only occur if $p(\theta^{0-1}|\theta) = J(\theta^{0-1}|\theta^0)$.)

Allowing asymmetric jumping rules can be useful in increasing the speed of the random walk. Convergence to the target distribution is proved in the same way as for the Metropolis algorithm. The proof of convergence to a unique stationary distribution is identical. To prove that the stationary distribution is the target distribution, $p(\theta^0|\theta^0)$, transfer $\theta^0$ to points $\theta_0$ and $\theta_1$ with posterior densities labeled so that $p(\theta_0|\theta^0) = J(\theta_0|\theta^1)$ and $p(\theta_1|\theta^0) = J(\theta_1|\theta^0)$. If $\theta_1^{0-1}$ follows the target distribution, then it is easy to show that the unconditional probability density of a transition from $\theta_0$ to $\theta_1$ is the same as the reverse transition.

Relation between the jumping rule and efficiency of simulations

The ideal Metropolis-Hastings jumping rule is simply to sample the proposal, $\theta^*$, from the target distribution; that is, $J(\theta^0|\theta^*) = p(\theta^0|\theta^*)$ for all $\theta^0$. Then the ratio $r$ in (11.2) is always exactly 1, and the iterates $\theta^*$ are a sequence of independent draws from the target; in general, however, iterative simulation is applied to problems for which direct sampling is not possible.

A good jumping distribution has the following properties:

- For any $\theta$, it is easy to sample from $J(\theta^0|\theta)$.  
- It is easy to compute the ratio $r$.  
- Each jump goes a reasonable distance in the parameter space (otherwise the random walk moves too slowly).  
- The jumps are not rejected too frequently (otherwise the random walk wastes too much time standing still).

We return to the topic of constructing efficient simulation algorithms in the next chapter.

11.3 Using Gibbs and Metropolis as building blocks

The Gibbs sampler and the Metropolis algorithm can be used in various combinations to sample from complicated distributions. The Gibbs sampler is the simplest of the block Markov chain simulation algorithms, and it is our first choice for conditionally conjugate models where we can directly sample from each conditional posterior distribution. For example, one could use the Gibbs sampler for the normal-normal hierarchical model in Chapter 1.

The Metropolis algorithm can be used for models that are not conditionally conjugate, for example, the two-parameter logistic regression for the binary response experiment in Section 3.7. In this example, the Metropolis algorithm could be performed in vector form—iterating in the two-dimensional space of $(\alpha, \beta)$—or embedded within a Gibbs sampler framework, alternately updating $\alpha$ and $\beta$ using one-dimensional Metropolis jumps. In either case, the Metropolis algorithm will probably have to be tuned to get a good acceptance rate, as discussed in Section 12.2.

If some of the conditional posterior distributions in a model can be sampled from directly and some cannot, then the parameters can be updated one at a time, with the Gibbs sampler used where possible and one-dimensional Metropolis updating used otherwise. More generally, the parameters can be updated in blocks, where each block is allowed one Gibbs sampler or a Metropolis jump of the parameters within the block.

A general problem with conditional sampling algorithms is that they can be slow when the parameters are highly correlated in the target distribution (for example, see Figure 11.11 on page 277). This can be fixed in simple problems using reparameterization (see Section 11.4) or more generally using the more advanced algorithms mentioned in Chapter 12.

11.4 Inference and assessing convergence

Inference and assessing convergence

Inference of the Gibbs sampler as a special case of the Metropolis-Hastings algorithm

Gibbs sampling can be viewed as a special case of the Metropolis-Hastings algorithm in the following way. We first define iteration $t$ to consist of a series of $d$ steps, with step $j$ corresponding to an update of the subvector $\theta_j$ conditional on all the other subvectors of $\theta$. Then the jumping distribution, $J_{t+j}(-)$, at step $j$ of iteration $t$ only jumps in the $j$th subvector, and does so with the conditional posterior density of $\theta_j$ given $\theta_{-j}^t$:

$$J_{t+j}(-|\theta_{-j}^t) = \begin{cases} p(\theta_j|\theta_{-j}^t) & \text{if } \theta_j^t = \theta_j^{t-1} \\ 0 & \text{otherwise.} \end{cases}$$

The only possible jumps are to parameter vectors $\theta^*$ that match $\theta^{t-1}$ on all components other than the $j$th. Under this jumping distribution, the ratio (11.2) at the $j$th step of iteration $t$ is

$$r = \frac{p(\theta^0|\theta_{-j}^{t-1}) J_{t+j}(\theta^0|\theta_{-j}^t)}{p(\theta^{t-1}|\theta_{-j}^t) J_{t+j}(\theta^{t-1}|\theta_{-j}^t)} = \begin{cases} \frac{p(\theta_j^0|\theta_{-j}^{t-1})}{p(\theta_j^t|\theta_{-j}^{t-1})} & \text{if } \theta_j^t = \theta_j^{t-1} \\ \frac{p(\theta_j^t|\theta_{-j}^{t-1})}{p(\theta_j^0|\theta_{-j}^{t-1})} & \text{otherwise.} \end{cases}$$

If the ratio $r$ is accepted, the second line above follows from the first, because the jumping rule, $\theta^*$ differs from $\theta^{t-1}$ only in the $j$th component. This line differs from the second by applying the rules of conditional probability to $\theta = (\theta_j, \theta_{-j})$ and taking $\theta_j = \theta_j^{t-1}$.

Formally, one iteration of the Gibbs sampler is defined as we do, to include all $d$ steps corresponding to the $d$ components of $\theta$, thereby updating all of $\theta$ at each iteration. It is possible, however, to define Gibbs sampling without the restriction that each component is updated in each iteration, as long as each component is updated periodically.
Difficulties of inference from iterative simulation

Iterative simulation adds two challenges to simulation inference. First, if the iterations have not proceeded long enough, as in Figure 11.1a, the simulations may be grossly unrepresentative of the target distribution. Even when simulations have reached approximate convergence, early iterations still reflect the starting approximation rather than the target distribution; for example, consider the early iterations of Figure 11.1b and 11.2b.

The second problem with iterative simulation draws is their within-sequence correlation: aside from any convergence issues, simulation inference from correlated draws is generally less precise than from the same number of independent draws. Serial correlation in the simulations is not necessarily a problem because, at convergence, the draws are identically distributed as \( p(y) \), and so when performing inferences, we ignore the order of the simulation draws in any case. But such correlation can cause inefficiencies in simulations. Consider Figure 11.1c, which displays 900 successive iterations from each of five simulated sequences of the Metropolis algorithm: the pairwise appearance of the scatterplot would not be likely to appear from 2500 independent draws from the normal distribution but is rather a result of the slow movement of the simulation algorithm. In some sense, the "effective" number of simulation draws here is far fewer than 2500. We calculate effective sample size using formulas (11.8) on page 287.

We handle the special problems of iterative simulation in three ways. First, we attempt to design the simulation runs to allow effective monitoring of convergence, in particular by simulating multiple sequences with starting points dispersed throughout parameter space, as in Figure 11.1a. Second, we monitor the convergence of all quantities of interest by comparing variation between and within simulated sequences until "within" variation roughly equals "between" variation, as in Figure 11.1b. Only when the distribution of each simulated sequence is close to the distribution of all the sequences mixed together can they all be approximating the target distribution. Third, if the simulation efficiency is unacceptably low (in the sense of requiring too much real time on the computer to obtain approximate convergence of posterior inferences for quantities of interest), the algorithm can be altered, as we discuss in Sections 12.1 and 12.2.

**Discarding early iterations of the simulation runs**

To diminish the influence of the starting values, we generally discard the first half of each sequence and focus attention on the second half. Our inference will be based on the assumption that the distributions of the simulated values \( \tilde{y} \), for large enough \( t \), are close to the target distribution, \( p(y) \). We refer to the practice of discarding early iterations of the Markov chain simulation as warm-up, depending on the context, different warm-up fractions can be appropriate. For example, in the Gibbs sampler displayed in Figure 11.2, it would be necessary to discard only a few initial iterations.\(^1\) We adopt the general practice of discarding the first half of a sequence as a conservative choice. For example, we might run 200 iterations and discard the first half. If approximate convergence has not yet been reached, we might then run another 200 iterations, now discarding all of the initial 200 iterations.

**Dependence of the iterations in each sequence**

Another issue that sometimes arises, once approximate convergence has been reached, is whether to thin the sequences by keeping every 10th simulation draw from each sequence.

---

1. In the simulation literature (including earlier editions of this book), the warm-up period is called burn-in, a term we now avoid because we feel it draws a misleading analogy to industrial processes in which products are discarded, in order to normal defects. We prefer the term warm-up to describe the early part of the simulations in which the sequences get closer to the tails of the distribution.

---

**Figure 11.3** Examples of two challenges to assessing convergence of iterative simulations. (a) In the left plot, either sequence alone looks stable, but the juxtaposition makes it clear that they have not converged to a common distribution. (b) In the right plot, the two sequences happen to cover a common distribution but neither sequence appears stationary. These graphs demonstrate the need to assess-sequence and also within-sequence information when assessing convergence.

---

and discarding the rest. In our applications, we have found it useful to skip iterations in problems with large numbers of parameters where computer storage is a problem, perhaps setting \( k \) so that the total number of iterations saved is no more than 1000.

Whether or not the sequences are thinned, if the sequences have reached approximate convergence, they can be directly used for inferences about the parameters \( \theta \) and any other quantities of interest.

**Multiple sequences with underdispersed starting points**

Our recommended approach to assessing convergence of iterative simulation is based on comparing different simulated sequences, as illustrated in Figure 11.1 on page 276, which shows three parallel simulations before and after approximate convergence. In Figure 11.1a, the multiple sequences clearly have not converged; the variance within each sequence is much less than the variance between sequences. Later, in Figure 11.1b, the sequences have mixed, and the two variance components are essentially equal.

To use such disparities, we clearly need more than one independent sequence. Thus our plan is to simulate independently at least two sequences, with starting points drawn from an underdispersed distribution (either from a crude estimate such as discussed in Section 10.2 or a more elaborate approximation as discussed in the next chapter).

---

**Monitoring scalar estimates**

We monitor each scalar estimate or other scalar quantities of interest separately. Estimates include all the parameters of interest in the model and any other quantities of interest (for example, the ratio of two parameters or the value of a predicted future observation). It is often useful also to monitor the value of the logarithm of the posterior density, which has probably already been computed if we are using a version of the Metropolis algorithm.

---

**Challenges of monitoring convergence: mixing and stationarity**

Figure 11.2 illustrates two of the challenges of monitoring convergence of iterative simulations. The first graph shows two sequences, each of which looks fine on its own (and,
BASICS OF MARKOV CHAIN SIMULATION

indeed, when looked at separately would satisfy any reasonable convergence criterion, but when looked at together reveal a clear lack of convergence. Figure 11.3a illustrates that, to achieve convergence, the sequences must together have mixed.

The second graph in Figure 11.3 shows two chains that have mixed, in the sense that they have traced out a common distribution, but they do not appear to have converged. Figure 11.3b illustrates that, to achieve convergence, each individual sequence must have stationary.

Splitting each saved sequence into two parts

We diagnose convergence (as noted above, separately for each scalar quantity of interest) by checking mixing and stationarity. There are various ways to do this: we apply a fairly simple approach in which we split each chain in half and check that all the resulting half-sequences have mixed. This simultaneously tests mixing (if all the chains have mixed, the separate parts of the different chains should also mix) and stationarity (as stationarity, the first and second half of each sequence should be tracing the same distribution).

We start with some number of simulated sequences in which the warm-up period (which, by default, we set to the first half of the simulations) has already been discarded. We then take each of these chains and split into the first and second half (this is all after discarding the warm-up iterations). Let m be the number of chains (after splitting) and n be the length of each chain. We always simulate at least two sequences so we can observe mixing; see Figure 11.3a; thus m is at least 4.

For example, suppose we simulate 5 chains, each of length 1000, and then discard the first half of each as warm-up. We see the left with 5 chains, each of length 500, and we split each into two parts: iterations 1–250 (originally iterations 501–750) and iterations 251–500 (originally iterations 751–1000). We now have m = 10 chains, each of length n = 250.

Assessing mixing using between- and within-sequence variances

For each scalar estimator \( \psi \), we label the simulations as \( \psi_{ij} (i = 1, \ldots, m; j = 1, \ldots, n) \), and we compute B and W, the between- and within-sequence variances:

\[
B = \frac{1}{mn} \sum_{i=1}^{m} \sum_{j=1}^{n} (\psi_{ij} - \overline{\psi})^2
\]

\[
W = \frac{1}{m} \sum_{i=1}^{m} (\overline{\psi}_i - \overline{\psi})^2
\]

The between-sequence variance, B, contains a factor of \( n \) because it is based on the variance of the within-sequence means, \( \overline{\psi}_i \), each of which is an average of \( n \) values \( \psi_{ij} \).

We can estimate \( \text{var}(\psi) \), the marginal posterior variance of the estimated, by a weighted average of B and W, namely

\[
\text{var}(\psi) = \frac{n-1}{n} W + \frac{1}{n} B
\]

This quantity overestimates the marginal posterior variance assuming the starting distribution is appropriately dispersed, but is unbiased under stationarity (that is, the starting distribution equals the target distribution), or in the limit \( n \to \infty \) (see Exercise 11.5). This is analogous to the classical variance estimate with cluster sampling.

Meanwhile, for any finite \( n \), the 'within' variance \( W \) should be underestimated of \( \text{var}(\psi) \) because the individual sequences have not had time to range over all of the target.

Table 11.1: 95% central intervals and estimated potential scale reduction factors for three scalar summaries of the bivariate normal distribution simulated using a Metropolis algorithm. (For demonstration purposes, the running scale of the Metropolis algorithm was purposely set to be inefficient; see Figure 11.1.) Displayed are inferences from the second halves of five parallel sequences, stopping after 50, 500, 5000, and 50000 iterations. The intervals for \( \rho \) are taken from the t-distribution and \( \chi^2 \) marginal distributions for these summaries in the target distribution.

<table>
<thead>
<tr>
<th>Number of iterations</th>
<th>( \rho_1 )</th>
<th>( \rho_2 )</th>
<th>( \log(\rho_1, \rho_2) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>-2.41 3.17</td>
<td>1.36</td>
<td>501</td>
</tr>
</tbody>
</table>

The result of the potential scale reduction is high, which we have reason to believe that proceeding with further simulations may improve our inference about the target distribution of the associated scalar estimator.

Example. Bivariate unit normal density with bivariate normal jumping kernel (continued)

We illustrate the multiple sequence method using the Metropolis simulations of the bivariate normal distribution illustrated in Figure 11.3. Table 11.1 displays posterior inference for the two parameters of the distribution as well as the log posterior density (relative to the density at the mode). After 50 iterations, the variance between the five sequences is much greater than the variance within, for all three univariate summaries considered. However, the five simulated sequences have converged adequately after 2000 or certainly 5000 iterations for the quantities of interest. The comparison with the true target distribution shows how some variability remains in the posterior inferences even after the Markov chains have converged. (This must be so, considering that even the simulation draws were independent, so that the Markov chains would converge in a single iteration, it would still require hundreds or thousands of draws to obtain precise estimates of extreme posterior quantities.)

The method of monitoring convergence presented here has the key advantage of not requiring the user to examine time series graphs of simulated sequences. Inspection of multivariate plots is a notoriously unreliable method of assessing convergence and in addition is prohibitively when monitoring a large number of quantities of interest, such as can arise in complicated hierarchical models. Because it is based on means and variances, the simple
11.5 Effective number of simulation draws

Once the simulated sequences have mixed, we can compute an approximate "effective number of independent simulation draws" for any estimate of interest $\hat{\phi}$. We start with the observation that if the $n$ simulation draws within each sequence were truly independent, then the between-sequence variance $B$ would be an unbiased estimate of the posterior variance $\text{var}(\hat{\phi})$, and we would have a total of $mn$ independent simulations from the $m$ sequences. In general, however, the simulations of $\hat{\phi}$ within each sequence will be autocorrelated, and $\hat{B}$ will be larger than $\text{var}(\hat{\phi})$, in expectation.

One way to define effective sample size for correlated simulation draws is to consider the statistical efficiency of the average of the simulations $\bar{\hat{\phi}}$, as an estimate of the posterior mean $E(\hat{\phi})$. This can be a reasonable baseline even though it is not the only possible summary and might be inappropriate, for example, if there is particular interest in mean or median values rather than mean expected value in the tail of the distribution.

Continuing with this definition, it is usual to compute effective sample size using the following asymptotic formula for the variance of the average of the correlated sequence:

$$\lim_{n \to \infty} \text{var}(\bar{\hat{\phi}}) = \left(1 + 2 \sum_{i=0}^{\infty} b_i \right) \text{var}(\hat{\phi}),$$

where $b_i$ is the autocorrelation of the sequence $\hat{\phi}$ at lag $i$. If the $n$ simulation draws from each of the $n$ chains were independent, then $\text{var}(\hat{\phi})$ would simply be $\frac{1}{n} \text{var}(\hat{\phi})$ and the sample size would be $mn$. In the presence of correlation, we then define the effective sample size as

$$n_{\text{eff}} = \frac{\text{var}(\bar{\hat{\phi}})}{\text{var}(\hat{\phi})} = 1 + 2 \sum_{i=0}^{\infty} b_i.$$  

(11.5)

The asymptotic nature of (11.5)-(11.6) might seem disturbing given that in reality we will only have a finite simulation, but this should not be a problem given that we already want to run the simulations long enough for approximate convergence to the (asymptotic) target distribution.

To compute the effective sample size we need an estimate of the sum of the correlations $b_i$ for which we use information within and between sequences. We start by computing the total variance using the estimate $\text{var}(\hat{\phi})$ from (11.3); we then estimate the correlations by first computing the variance $V_i$ of each $\hat{\phi}_i$ at each lag $i$.

$$V_i = \frac{1}{n(n-1)} \sum_{j=1}^{n} (\hat{\phi}_i - \hat{\phi}_j)^2.$$  

We then estimate the correlations by inverting the formula, $E(\hat{\phi}_i - \hat{\phi}_j)^2 = 2(1 - b_i) \text{var}(\hat{\phi})$.

$$b_i = 1 - \frac{V_i}{2\text{var}(\hat{\phi})}.$$  

(11.7)

Unfortunately, we cannot simply sum all of these to estimate $n_{\text{eff}}$ in (11.6); the difficulty is that for large values of $i$ the sample correlation is too noisy. Instead we compute a partial sum, starting from lag 0 and continuing until the sum of autocorrelation estimates for the first 0.5 effective lags $b_{i+1} + b_{i+2}$ is negative. We use this positive partial sum as our estimate of $\sum_{i=0}^{\infty} b_i$ in (11.6). Putting this all together yields the estimate,

$$n_{\text{eff}} = \frac{\text{var}(\bar{\hat{\phi}})}{1 + \sum_{i=0}^{\infty} b_i}.$$  

(11.8)

where the estimated autocorrelations $\hat{b}_i$ are computed from formula (11.7) and $T$ is the first odd positive integer for which $\hat{b}_{i+1} + \hat{b}_{i+2}$ is negative.

All these calculations should be performed using only the saved iterations, after discarding the burn-in period. For example, suppose we simulate 1 chain, each of length 1000, and then discard the first half of each as warm-up. Then $m = n = 200$, and we compute $\text{var}(\bar{\hat{\phi}})$ and correlations only for the saved iterations (thus, up to a maximum lag of 299, although in practice the stopping point $T$ in (11.8) will be much lower).

Bounded or long-tailed distributions

The above convergence diagnostics are based on means and variances, and they will not work well for parameters or scalar summaries for which the posterior distribution, $p(\hat{\phi}|y)$, is not Gaussian. (As discussed in Chapter 4, asymptotically the posterior distribution should typically be normally distributed as the data sample size approaches infinity, but we are never actually at the asymptotic limit (in fact we are often interested in learning from small samples), and (b) it is common to have only a small amount of data on individual parameters that are part of a hierarchical model.)

For summaries $\hat{\phi}$ whose distributions are constrained or otherwise far from normal, we can preprocess simulations using transformations before computing the potential scale reduction factor $R$ and the effective sample size $n_{\text{eff}}$. We can take the logarithm of all positive quantities, the logit of quantities that are constrained to fall in (0,1), and use the rank transformation for long-tailed distributions. Transforming the simulations to have behaving distributions should allow mean and variance-based convergence diagnostics to work better.

Stopping the simulations

We monitor convergence for the entire multivariate distribution, $p(\hat{\phi}|y)$, by computing the potential scale reduction factor $R$ (11.4) and the effective sample size (11.8) for each scalar summary of interest. (Recall that we are using $R$ to denote the vector of unknowns in the posterior distribution, and $\hat{\phi}$ to represent scalar summaries, considered one at a time.)

We recommend computing the potential scale reduction for all scalar estimates of interest; if $R$ is not near 1 for all of them, continue the simulation runs (perhaps altering the simulation algorithm itself to make the simulations more efficient, as described in the next section). The condition of $R$ being near 1 depends on the problem at hand, but we generally have been satisfied with setting 1.1 as a threshold.

We can use effective sample size $n_{\text{eff}}$ to give us a sense of the precision obtained from our simulation. As we have discussed in Section 10.5, for many purposes it should suffice to have 100 or even 10 independent simulation draws. (If $n_{\text{eff}} = 10$, the simulation standard error is increased by $1/\sqrt{10} = 1.05$. As a default rule, we suggest running the simulation until $n_{\text{eff}}$ at least 5m, that is, until there are the equivalent of at least 10 independent draws per sequence (recall that $m$ is twice the number of sequences, as we have split each sequence into two parts so that $R$ can assess stationarity as well as mixing). Having an effective sample size of 10 per sequence should typically correspond to stability of all the simulated sequences. For some purposes, more precision will be desired, and then a higher effective sample size threshold can be used.
Table 11.2. Computer time in seconds for blood drawn from 64 animals randomly allocated to five different diets. Different treatments had different numbers of observations because the randomization was unrestricted. From Box, Hunter, and Hunter (1978), who adjusted the data so that the means are integers, a complication we ignore in our analysis.

Once is near 1 and 49 is more than 10 per chain for all scalar estimators of interest, just collect the 45 simulations (with warm-up iterations already excluded, as noted below) and treat them as a sample from the target distribution.

Even if an iterative simulation appears to converge and has passed all tests of convergence, it may still be far from convergence if important aspects of the target distribution were not captured by the starting distribution and are not easily reachable by the simulation algorithm. When we declare approximate convergence, we are actually assuming that each individual sequence appears stationary and that the observed sequence has mixed well with each other. These checks are not hypothesis tests. There is no p-value and no statistical significance. We assess discrepancy from convergence via practical significance (or some conventional version thereof, such as is > 1).

11.6 Example: hierarchical normal model

We illustrate the simulation algorithms with a hierarchical normal model, extending the problem discussed in Section 5.4 by allowing an unknown data variance, $\sigma^2$. The example is continued in Section 11.6 to illustrate model-based computation. We demonstrate with the normal model because it is simple enough that the key computational ideas do not get lost in the details.

Data from a small experiment

We demonstrate the computations on a small experimental dataset, displayed in Table 11.2, that has been used previously in an example in the statistical literature. Our purpose here is to illustrate computational methods, not to perform a full Bayesian data analysis (which includes model construction and model checking), and so we do not discuss the applied context.

The model

Under the hierarchical normal model (restated here, for convenience), data $y_{ij}, i = 1, \ldots, n_i, j = 1, \ldots, J$, are independently normally distributed within each group $j$ with means $\theta_j$ and common variance $\sigma^2$. The total number of observations is $n = \sum_{i=1}^{n_i}$, $n_j$. The group means are assumed to follow a normal distribution with unknown mean $\mu$ and variance $\tau^2$, and a uniform prior distribution is assumed for $(\mu, \log \sigma, \log \tau)$ with $\sigma > 0$ and $\tau > 0$ equivalently, $(\mu, \log \sigma, \log \tau) \sim \tau$. If we were to assign a uniform prior distribution to be the posterior distribution would be improper, as discussed in Chapter 5.

The joint posterior density of all the parameters is

$$p(\theta, \mu, \log \sigma, \log \tau | y) \propto \tau^{J+1} \prod_{j=1}^{J} N(\theta_j | \mu, \tau^2) \prod_{i=1}^{n_i} N(y_{ij} | \theta_j, \sigma^2).$$

Example: hierarchical normal model

Starting points

In this example, we can choose overdispersed starting points for each parameter $\theta_j$ by simply taking random points from the data $y_{ij}$ from group $j$. We obtain 10 starting points for the simulations by drawing $\theta_j$ independently in this way for each group. We also need starting points for $\mu$, which can be taken as the average of the starting $\theta_j$, $\bar{\theta}_j$. No starting values are needed for $\sigma$ or $\tau$ as they can be drawn as the first steps in the Gibbs sampler.

Section 13.6 presents a more elaborate procedure for constructing a starting distribution for the iterative simulations using the posterior mode and a normal approximation.

Gibbs sampler

The conditional distributions for this model all have simple conjugate forms:

1. Conditional posterior distribution of each $\theta_j$. The factors in the joint posterior density that involve $\theta_j$ are the $N(\mu, \tau^2)$ prior distribution and the normal likelihood from the data in the $j$th group, $y_{ij}, i = 1, \ldots, n_j$. The conditional posterior distribution of each $\theta_j$ given the other parameters in the model is

$$\theta_j | \mu, \sigma, \tau, y \sim N(\bar{\theta}_j, \sigma^2/j),$$

where the parameters of the conditional posterior distribution depend on $\mu, \sigma$, and $\tau$ as well as $y$:

$$\bar{\theta}_j = \frac{\mu + \sum_{i=1}^{n_j} y_{ij}}{\frac{1}{\sigma^2} + \frac{n_j}{\tau^2}},$$

$$\sigma_j = \left(\frac{1}{\frac{1}{\sigma^2} + \frac{n_j}{\tau^2}}\right)^{1/2}$$

These conditional distributions are independent; thus drawing the $\theta_j$'s one at a time is equivalent to drawing the vector $\theta$ at all once from its conditional posterior distribution.

2. Conditional posterior distribution of $\mu$. Conditional on $y$ and the other parameters in the model, $\mu$ has a normal distribution determined by the $\bar{\theta}_j$'s:

$$p(\mu | \tau, \sigma, y) \sim N(\mu, \tau^2/j),$$

where

$$\mu = \frac{1}{J} \sum_{j=1}^{J} \theta_j.$$  

3. Conditional posterior distribution of $\sigma^2$. The conditional posterior density for $\sigma^2$ has the form corresponding to a normal variance with known mean; there are $n$ observations $y_{ij}$ with mean $\theta_j$. The conditional posterior distribution is

$$\sigma^2 | \theta, \mu, \tau, y \sim Inv-Chisq(n, \sigma^2),$$

where

$$\sigma^2 = \frac{1}{n} \sum_{i=1}^{n_i} (y_{ij} - \theta_j)^2.$$

4. Conditional posterior distribution of $\tau^2$. Conditional on the data and the other parameters in the model, $\tau^2$ has a scaled inverse-$\chi^2$ distribution, with parameters depending only on $\mu$ and $\bar{\theta}_j$ (as can be seen by examining the joint posterior density):

$$\tau^2 | \theta, \mu, \sigma, y \sim Inv-Chisq(J - 1, \tau^2).$$