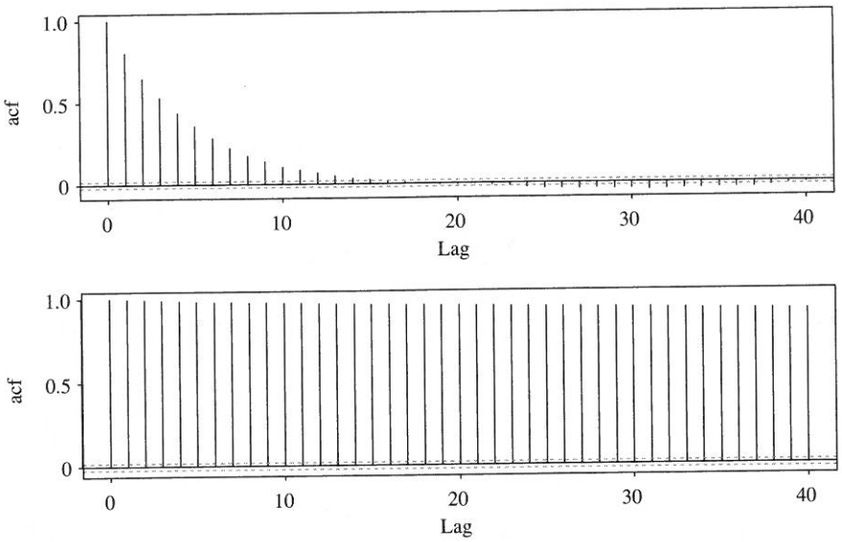

FIGURE 7.8 Autocorrelation function plots for independence chain of Example 7.2 with proposal densities Beta(1,1) (top) and Beta(2,10) (bottom).

and cusum plot should be based only on the values to be used in the final estimator. Yu and Mykland [678] suggest that cusum plots that are very wiggly and have smaller excursions from 0 indicate that the chain is mixing well. Plots that have large excursions from 0 and are smoother suggest slower mixing speeds. The cusum plot shares one drawback with many other convergence diagnostics: For a multimodal distribution where the chain is stuck in one of the modes, the cusum plot may appear to indicate good performance when, in fact, the chain is not performing well.

An autocorrelation plot summarizes the correlation in the sequence of $\mathbf{X}^{(t)}$ at different iteration lags. The autocorrelation at lag i is the correlation between iterates that are i iterations apart [212]. A chain that has poor mixing properties will exhibit slow decay of the autocorrelation as the lag between iterations increases. For problems with more than one parameter it may also be of use to consider cross-correlations between parameters that might be related, since high cross-correlations may also indicate poor mixing of the chain.

Example 7.8 (Mixture Distribution, Continued) Figure 7.8 shows autocorrelation function (acf) plots for the independence chain described in Example 7.2. In the top panel, the more appropriate proposal distribution yields a chain for which the autocorrelations decrease rather quickly. In the lower panel, the bad proposal distribution yields a chain for which autocorrelations are very high, with a correlation of 0.92 for observations that are 40 iterations apart. This panel clearly indicates poor mixing. □

7.3.1.2 Burn-in and Run Length

Key considerations in the diagnosis of convergence are the burn-in period and run length. Recall that it is only in the limit that an MCMC algorithm yields $X^{(t)} \sim f$. For any implementation, the iterates will not have exactly the correct marginal distribution, and the dependence on the initial point (or distribution) from which the chain was started may remain strong. To reduce the severity of this problem, the first D values from the chain are typically discarded as a *burn-in period.*

A commonly used approach for the determination of an appropriate burn-in period and run length is that of Gelman and Rubin [221, 224]. This method is based on a statistic motivated by an analysis of variance (ANOVA): The burn-in period or MCMC run-length should be increased if a between-chain variance is considerably larger than the within-chain variance. The variances are estimated based on the results of J runs of the MCMC algorithm to create separate, equal-length chains ($J \geq 2$) with starting values dispersed over the support of the target density.

Let L denote the length of each chain after discarding D burn-in iterates. Suppose that the variable (e.g., parameter) of interest is X, and its value at the tth iteration of the jth chain is $x_j^{(t)}$. Thus, for the jth chain, the D values $x_j^{(0)}, \ldots, x_j^{(D-1)}$ are discarded and the L values $x_j^{(D)}, \ldots, x_j^{(D+L-1)}$ are retained. Let

$$\bar{x}_j = \frac{1}{L} \sum_{t=D}^{D+L-1} x_j^{(t)} \quad \text{and} \quad \bar{x}_\cdot = \frac{1}{J} \sum_{j=1}^{J} \bar{x}_j, \tag{7.19}$$

and define the between-chain variance as

$$B = \frac{L}{J-1} \sum_{j=1}^{J} (\bar{x}_j - \bar{x}_\cdot)^2. \tag{7.20}$$

Next define

$$s_j^2 = \frac{1}{L-1} \sum_{t=D}^{D+L-1} \left(x_j^{(t)} - \bar{x}_j\right)^2$$

to be the within-chain variance for the jth chain. Then let

$$W = \frac{1}{J} \sum_{j=1}^{J} s_j^2 \tag{7.21}$$

represent the mean of the J within-chain estimated variances. Finally, let

$$R = \frac{[(L-1)/L]W + (1/L)B}{W}. \tag{7.22}$$

If all the chains are stationary, then both the numerator and the denominator should estimate the marginal variance of X. If, however, there are notable differences between the chains, then the numerator will exceed the denominator.

In theory, $\sqrt{R} \to 1$ as $L \to \infty$. In practice, the numerator in (7.22) is slightly too large and the denominator is slightly too small. An adjusted estimator is given by

$$\hat{R} = \frac{J+1}{J} R - \frac{L-1}{JL}.$$

Some authors suggest that $\sqrt{\hat{R}} < 1.1$ indicates that the burn-in and chain length are sufficient [544]. Another useful convergence diagnostic is a plot of the values of $\hat{R}$ versus the number of iterations. When $\hat{R}$ has not stabilized near 1, this suggests lack of convergence. If the chosen burn-in period did not yield an acceptable result, then D should be increased, L should be increased, or preferably both. A conservative choice is to use one-half of the iterations for burn-in. The performance of this diagnostic is improved if the iterates $x_j^{(t)}$ are transformed so that their distribution is approximately normal. Alternatively, a reparameterization of the model could be undertaken and the chain rerun.

There are several potential difficulties with this approach. Selecting suitable starting values in cases of multimodal f may be difficult, and the procedure will not work if all of the chains become stuck in the same subregion or mode. Due to its unidimensionality, the method may also give a misleading impression of convergence for multidimensional target distributions. Enhancements of the Gelman–Rubin statistic are described in [71, 224], including an improved estimate of R in (7.22) that accounts for variability in unknown parameters. In practice, these improvements lead to very similar results. An extension for multidimensional target distributions is given in [71].

Raftery and Lewis [526] proposed a very different quantitative strategy for estimating run length and burn-in period. Some researchers advocate no burn-in [231].

7.3.1.3 *Choice of Proposal*

As illustrated in Example 7.2, mixing is strongly affected by features of the proposal distribution, especially its spread. Further, advice on desirable features of a proposal distribution depends on the type of MCMC algorithm employed.

For a general Metropolis–Hastings chain such as an independence chain, it seems intuitively clear that we wish the proposal distribution g to approximate the target distribution f very well, which in turn suggests that a very high rate of accepting proposals is desirable. Although we would like g to resemble f, the tail behavior of g is more important than its resemblance to f in regions of high density. In particular, if f/g is bounded, the convergence of the Markov chain to its stationary distribution is faster overall [543]. Thus, it is wiser to aim for a proposal distribution that is somewhat more diffuse than f.

In practice, the variance of the proposal distribution can be selected through an informal iterative process. Start a chain, and monitor the proportion of proposals that have been accepted; then adjust the spread of the proposal distribution accordingly. After some predetermined acceptance rate is achieved, restart the chain using the appropriately scaled proposal distribution. For a Metropolis algorithm with normal target and proposal distributions, it has been suggested that an acceptance rate of between 25 and 50% should be preferred, with the best choice being about 44% for one-dimensional problems and decreasing to about 23.4% for higher-dimensional problems [545, 549]. To apply such rules, care are must be taken to ensure that the target and proposal distributions are roughly normally distributed or at least simple, unimodal distributions. If, for example, the target distribution is multimodal, the chain may get stuck in one mode without adequate exploration of the other portions of the parameter space. In this case the acceptance rate may very high, but the probability of jumping from one mode to another may be low. This suggests one difficult issue with most MCMC methods; it is useful to have as much knowledge as possible about the target distribution, even though that distribution is typically unknown.

Methods for adaptive Markov chain Monte Carlo (Section 8.1) tune the proposal distribution in the Metropolis algorithm during the MCMC algorithm. These methods have the advantage that they are automatic and, in some implementations, do not require the user to stop, tune, and restart the algorithm on multiple occasions.

7.3.1.4 *Reparameterization*

Model reparameterization can provide substantial improvements in the mixing behavior of MCMC algorithms. For a Gibbs sampler, performance is enhanced when components of $\mathbf{X}$ are as independent as possible. Reparameterization is the primary strategy for reducing dependence. For example, if f is a bivariate normal distribution with very strong positive correlation, both univariate conditionals will allow only small steps away from $\mathbf{X}^{(t)} = \mathbf{x}^{(t)}$ along one axis. Therefore, the Gibbs sampler will explore f very slowly. However, suppose $\mathbf{Y} = (\mathbf{X_1} + \mathbf{X_2}, \mathbf{X_1} - \mathbf{X_2})$. This transformation yields one univariate conditional on the axis of maximal variation in $\mathbf{X}$ and the second on an orthogonal axis. If we view the support of f as cigar shaped, then the univariate conditionals for $\mathbf{Y}$ allow one step along the length of the cigar, followed by one across its width. Therefore, the parameterization inherent in $\mathbf{Y}$ makes it far easier to move from one point supported by the target distribution to any other point in a single move (or a few moves).

Different models require different reparameterization strategies. For example, if there are continuous covariates in a linear model, it is useful to center and scale the covariates to reduce correlations between the parameters in the model. For Bayesian treatment of linear models with random effects, hierarchical centering can be used to accelerate MCMC convergence [218, 219]. The term *hierarchical centering* comes from the idea that the parameters are centered as opposed to centering the covariates. Hierarchical centering involves reexpressing a linear model into another form that produces different conditional distributions for the Gibbs sampler.

Example 7.9 (Hierarchical Centered Random Effects Model) For example, consider a study of pollutant levels where it is known that tests performed at different laboratories have different levels of measurement error. Let y_{ij} be the pollutant level of the jth sample that was tested at the ith laboratory. We might consider a simple random effects model

$$y_{ij} = \mu + \alpha_i + \epsilon_{ij} \tag{7.23}$$

where $i = 1, \ldots, I$ and $j = 1, \ldots, n_i$. In the Bayesian paradigm, we might assume $\mu \sim N(\mu_0, \sigma_\mu^2)$, $\alpha_i \sim N(0, \sigma_\alpha^2)$, and $\epsilon_{ij} \sim N(0, \sigma_\epsilon^2)$. The hierarchical centered form of (7.23) is a simple reparameterization of the model with $y_{ij} = \gamma_i + \epsilon_{ij}$ where