RESEARCH
REPORT

March 2003
RR-03-07

Assessing Convergence of the Markov Chain Monte Carlo Algorithms: A Review

Sandip Sinharay

Research &
Development Division
Princeton, NJ 08541

Assessing Convergence of the Markov Chain Monte Carlo Algorithms: A Review

Sandip Sinharay

Educational Testing Service, Princeton, NJ

March 2003

Abstract

Markov chain Monte Carlo (MCMC) algorithms are in wide use for fitting complicated statistical models in psychometrics in situations where the traditional estimation techniques are very difficult to apply. One of the stumbling blocks in using an MCMC algorithm is determining the convergence of the algorithm. Because the convergence is not that of a scalar quantity to a point, but that of a distribution to another distribution, the issue remains an enigma to many users of MCMC, especially to those without a sound knowledge of mathematical statistics. This article is an attempt to provide psychometricians using the MCMC algorithms a better understanding of the concept of convergence of the algorithms and an improved knowledge about the diagnostics tools to assess convergence of the MCMC algorithms.

Key words: Bayesian analysis, convergence diagnostics, Gibbs sampling, MCMC

Acknowledgements

The author thanks Matthew Johnson profusely for his invaluable advice resulting in significant improvement of the paper. The author gratefully acknowledges the inspiration provided by Russell Almond, the help of Kim Fryer and William Monaghan with proofreading, and the valuable advice of Hariharan Swaminathan, Robert Mislevy, Daniel Eignor, Paul Holland, Howard Wainer, and Shelby Haberman.

The author also wishes to thank the National Center for Education Statistics for allowing access to the data set for Section 4 and Robert Smith for providing the data set for Section 5.

1. Introduction

Markov chain Monte Carlo (MCMC) algorithms (Gelman, Carlin, Stern, & Rubin 1995; Gilks, Richardson, & Spiegelhalter, 1996) have recently emerged as a very popular tool for fitting Bayesian statistical models in psychometrics. The main reason of the popularity of the algorithms is their ability to fit quite complex models where the standard techniques, like maximum likelihood estimation (MLE), are very difficult to apply. Basically, an MCMC algorithm fits a statistical model to a data set by generating a random sample from the probability distribution (also called the *target distribution*) implied by the model. The generated sample serves as a discrete approximation to the probability distribution for further inference about the problem at hand. The first known application of the MCMC algorithm in psychometrics is Albert (1992). After Patz and Junker (1999a; 1999b) show how to use the algorithms to solve difficult problems in psychometrics, there has been a surge in the use of the algorithms in this field. Bradlow, Wainer, and Wang (1999), Beguin and Glas (2001), Scott and Ip (2002), Hartz, Roussos, and Stout (2002), Patz, Junker, and Johnson (in press), Johnson (2002), and Sinharay, Johnson, and Williamson (2003) are recent examples of application of the MCMC algorithms to fit quite complicated models in psychometrics.

Like any other statistical method, the MCMC methods have their own disadvantages. One key disadvantage is the difficulty of determining the *convergence* of the algorithms, i.e., determining when we can stop an MCMC algorithm and be certain that its output may be used safely for inference about the problem considered. As discussed later, the convergence of interest here is *convergence in distribution* (not *convergence to a point*), making the concept difficult to grasp. Further, both the distribution generating the sample (referred to as the *kernel*) and the target distribution are very difficult to deal with in most real applications of the MCMC algorithm. To make matters worse, an investigator has to make a judgment about convergence based on only a sample generated by the algorithm.

Even with the increasing popularity of the MCMC algorithms in psychometrics, researchers in this field often pay little attention to the issue of assessing convergence of the algorithms. For example, the treatment of the convergence issue is often inadequate

in Wang, Bradlow, and Wainer (2001) and Hartz, Roussos, and Stout (2002), the recently developed software packages applying the MCMC algorithms in psychometrics. The possible reasons are two-fold—the lack of understanding of the difficult concept of convergence and the lack of knowledge about tools for assessing convergence. This work is an attempt to help the psychometricians get over both of the hurdles.

While explaining the concept of the convergence of the MCMC algorithms and the importance of assessing convergence, this document also discusses a number of popular diagnostics in some detail. There is an additional focus given to the computation issue with each diagnostic tool discussed here. To make the contents of the work accessible to a wide audience in psychometrics, the discussion centers on diagnostics that are conceptually easier to understand and that provide little difficulty computationally (or are implemented in publicly available software). For readers with further interest, this report makes reference to a number of more complicated methods not discussed here.

Section 2 introduces the MCMC algorithms and briefly discusses the most popular MCMC algorithms, the Gibbs sampler and the Metropolis-Hastings algorithms. Then it discusses the issue of convergence of the algorithms, the difficulty in detecting the convergence, and the importance of applying a number of diagnostic tools for assessing the convergence. Section 3 discusses a number of popular MCMC convergence diagnostics with different theoretical bases. The section also provides guidance as to where one can find software that implements these methods. Section 4 applies the diagnostics to a real data set example from the National Assessment of Educational Progress (NAEP) Math Online (MOL) special study (Sandene, Bennett, Braswell, & Oranje, in press). The main goal of this section is to explain how one should approach the problem of assessing convergence of an MCMC algorithm using a variety of diagnostic tools. Another objective is to compare the performance of the different diagnostic tools. Section 5 applies the convergence diagnostics to output from SCORIGHT (Wang, Bradlow, & Wainer, 2001) software to study the convergence of the MCMC algorithm used in the software. Section 6 summarizes the work and provides recommendations on how to assess the convergence of an MCMC algorithm.

2. The MCMC Algorithms

What Is an MCMC Algorithm?

The MCMC algorithms are frequently used to obtain a random sample from a posterior distribution of interest in a Bayesian analysis. The sample serves as a discrete approximation to the posterior distribution for further inference.

A Markov chain is a sequence of random variables, $X_t, t = 1, 2, \ldots$, with the property that the distribution of X_t conditional on $X_1, X_2, \ldots, X_{t-2}, X_{t-1}$ depends only on X_{t-1}. The key idea behind an MCMC algorithm is to create a *Markov chain*, whose stationary distribution is the posterior distribution of interest, in the parameter space. Then the simulation process is run long enough so that the distribution of the process beyond some point of time is close enough to the stationary distribution. The posterior expectations of relevant functions of the parameters are then approximated using *Monte Carlo* integration. A number of books (e.g., Gelman, Carlin, Stern, & Rubin 1995; Gilks, Richardson, & Spiegelhalter, 1996) provide a more detailed discussion of the MCMC algorithms. Gibbs sampling (Geman & Geman, 1984) and Metropolis-Hastings algorithm (Metropolis & Ulam, 1949; Hastings, 1970) are two of the most popular MCMC algorithms.

Gibbs Sampling

The Gibbs sampling (or *Gibbs sampler*) algorithm is defined in terms of the components of the parameter vector. Suppose that a researcher is interested in generating a sample from a posterior distribution. Suppose also that it is possible to partition the parameter vector into a number of components or sub-vectors. Each iteration in a Gibbs sampler goes through the components of the parameter vector, drawing each component conditional on the values of all the other components and the data.

The algorithm begins from an initial value of the parameter vector, chosen in the same way as one would to find a maximum likelihood estimate with an expectation-maximization (EM) algorithm. Each iteration consists of p steps, where p is the number of components of the parameter vector. In the j-th step of any iteration, the algorithm samples the j-th component of the parameter vector from the full conditional distribution

of the same component (conditioned on the most recently generated values of the other components and the data $\boldsymbol{y}$). This description assumes that the Gibbs sampling method samples components in the same order for each iteration. In practice, it is possible to vary the order of the components.

Let us look at an example to understand how the Gibbs sampler works. Suppose we use the algorithm to generate a sample from the posterior distribution

$$\left(\begin{array}{c} \omega_1 \\ \omega_2 \end{array} \right) \Bigg| \boldsymbol{y} \sim N_2 \left(\left(\begin{array}{c} y_1 \\ y_2 \end{array} \right), \left(\begin{array}{cc} 1 & \rho \\ \rho & 1 \end{array} \right) \right),$$

where $\boldsymbol{\omega} = (\omega_1, \omega_2)'$ is the parameter vector, $\boldsymbol{y} = (y_1, y_2)'$ is the only data point, and ρ is a constant. The conditional distributions of the components of $\boldsymbol{\omega}$ are

$$\omega_1 | \omega_2, \boldsymbol{y} \sim N\left(y_1 + \rho(\omega_2 - y_2), 1 - \rho^2\right), \tag{1}$$

$$\omega_2 | \omega_1, \boldsymbol{y} \sim N\left(y_2 + \rho(\omega_1 - y_1), 1 - \rho^2\right). \tag{2}$$

The Gibbs sampler starts from an initial value of $\boldsymbol{\omega}$. Then the algorithm proceeds by alternatively sampling from the distributions (1) and (2), using the most recently sampled value of ω_2 in (1) and the most recently sampled value of ω_1 in (2).

The algorithm should be run for a large number of iterations. Determining the number such that the algorithm reaches approximate convergence to the stationary distribution is not straightforward (we will discuss the problem in detail in later sections). Note also that simulations from a Gibbs sampler, and Markov chain in general, are not independent, and the researcher must address this issue to estimate the accuracy of further inferences (Geyer, 1992).

Metropolis Algorithm

The Metropolis algorithm (Metropolis & Ulam, 1949) is useful when there is difficulty in directly sampling from one or more of the component conditional distributions of a posterior distribution.

Let $\boldsymbol{\omega}$ denote the parameter vector in the problem concerned. The t-th iteration of the Metropolis algorithm consists of the following steps:

- Sample a candidate point $\boldsymbol{\omega}^*$ from a jumping distribution $J_t(\boldsymbol{\omega}^*|\boldsymbol{\omega}^{t-1})$, which must be symmetric, i.e., $J_t(\boldsymbol{\omega}_a|\boldsymbol{\omega}_b) = J_t(\boldsymbol{\omega}_b|\boldsymbol{\omega}_a)$ for all $\boldsymbol{\omega}_a, \boldsymbol{\omega}_b$, and t.

- Compute the acceptance probability

$$r = min\left(\frac{p(\boldsymbol{\omega}^*|\boldsymbol{y})}{p(\boldsymbol{\omega}^{t-1}|\boldsymbol{y})}, 1\right). \tag{3}$$

- Set

$$\boldsymbol{\omega}^t = \begin{cases} \boldsymbol{\omega}^* & \text{with probability} \quad r \\ \boldsymbol{\omega}^{t-1} & \text{with probability} \quad (1-r). \end{cases}$$

Note that the algorithm will always accept a candidate point with higher posterior density than the current value. Candidate points with posterior density lower than the current value may or may not be accepted.

A common choice of the jumping distribution $J_t(\boldsymbol{\omega}^*|\boldsymbol{\omega}^{t-1})$ is a normal distribution with mean equal to $\boldsymbol{\omega}^{t-1}$ and a variance matrix chosen so that the algorithm has an acceptance rate around 44% for univariate parameters and around 23% for vector-valued parameters. Gelman, Roberts, and Gilks (1995) suggest these acceptance rates based on theory of multivariate normal distribution to optimize the efficiency of the algorithm.

Just like the Gibbs sampler, the Metropolis algorithm should be run for a larger number of iterations and must be monitored for approximate convergence to the stationary distribution.

Note that the computation of the acceptance probability r in (3) requires the computation of the ratio of the posterior density at two values. Therefore, it is enough to know the posterior distribution up to a normalizing constant to sample from it using Metropolis algorithm. This characteristic makes the Metropolis algorithm very useful in Bayesian analysis for which the posterior density is known only up to a normalizing constant.

The Metropolis-Hastings algorithm (Hastings, 1970) is a generalization of the Metropolis algorithm that allows one to use a jumping distribution $J_t(\boldsymbol{\omega}_a|\boldsymbol{\omega}_b)$ that is not symmetric. To allow for the asymmetry of the jumping rule, this algorithm replaces the

ratio r in (2.2) by

$$r = min\left(\frac{p(\boldsymbol{\omega}^*|\boldsymbol{y})}{p(\boldsymbol{\omega}^{t-1}|\boldsymbol{y})}\Big/\frac{J(\boldsymbol{\omega}^*|\boldsymbol{\omega}^{t-1})}{J(\boldsymbol{\omega}^{t-1}|\boldsymbol{\omega}^*)}, 1\right). \tag{4}$$

Allowing asymmetric jumping rules can increase the speed of convergence of the Markov chain.

It is possible to apply a combination Gibbs sampler and the Metropolis algorithm to the same problem. For example, Patz and Junker (1999a; 1999b) apply the Metropolis algorithm to sample a few components of the parameters in a giant Gibbs sampler and call it a *Metropolis-within-Gibbs* algorithm.

Convergence of the MCMC Algorithms

The basic idea of an MCMC algorithm is to create a Markov process that has a stationary distribution the same as a posterior distribution of interest. We can safely assume convergence of an MCMC algorithm when we are certain that the sample generated from the algorithm is indeed from the posterior distribution of interest. Technically, convergence occurs when the generated Markov chain converges in distribution to the posterior distribution of interest. Because the convergence is in distribution (and not to a point) and the generated values will vary even after convergence, the convergence of MCMC is not a well-understood concept and not easy to assess, either.

An ideal solution would be to analytically compute the convergence rate of the MCMC algorithms and then take a sufficient number of iterations to satisfy any prescribed accuracy criteria. However, this is not possible in general (Tierney, 1994). There have been attempts to provide theoretical results to determine the number of iterations required to ensure convergence for specific type of models, e.g., Brooks (1998), Polson (1996), and Rosenthal (1995). However, these results are highly problem-specific, involve laborious calculations, and usually provide huge estimates of little practical value.

In an attempt to perform at least some kind of statistical analysis to assess convergence of the MCMC algorithms in general, a number of convergence diagnostics have been suggested. A number of these apply theory of Markov chains to the sampled values to

detect if the sampled distribution has reached stationarity. A number of other diagnostics compare the sampled distributions obtained from the MCMC for different runs (one run may be the subset of another); convergence is concluded when the difference between some aspects of the empirical distributions (e.g., mean, quantile, cumulative density function, etc.) over the different runs is negligible in some sense. To many theoreticians, both these approaches are fundamentally flawed because one should measure the distance between the sampled distribution and the target posterior distribution rather than examining one or two approximations of the latter. Of course, this is impractical—if the target distribution is easy to handle, we would not use MCMC in the first place. The diagnostics are particularly unreliable for a slowly moving MCMC algorithm because the diagnostics are then often based on only a small region of the sample space, unless the algorithm runs for a very long time.

Still, statisticians rely heavily on diagnostics, if for no other reason than "a weak diagnostic is better than no diagnostic at all" (Cowles & Carlin, 1996). These diagnostics test for conditions that are only necessary, but not sufficient, for convergence. No single method exists that can guarantee the convergence of the MCMC algorithms, which forces Cowles and Carlin (1996) to conclude that "clearly our recommendations imply that automated convergence monitoring (as by a machine) is unsafe and should be avoided." They recommend a two-stage process, wherein the model specification and sampling are separated from convergence diagnosis and subsequent output analysis. Also, one should use several diagnostics of different natures to increase the chance of correctly assessing convergence.

One important question related to applying the convergence diagnostics to a practical problem is "how many parameters to monitor?" Gelman and Rubin (1992b) recommend simultaneously monitoring the convergence of all the parameters in a model. Some practitioners monitor convergence of only the parameters of interest, especially for problems with high-dimensional parameters, but this may lead to the mistake of diagnosing convergence too early. Carlin and Louis (1996, pp. 203, 356, and 361-362) have an example where they fit the model

$$y_i \sim N\left(\eta_i = \theta_i + \phi_i, 1\right), i = 1, 2, \ldots n,$$

using a proper noninformative prior distribution on the parameters θ_is and ϕ_is. Monitoring only η_is (and not θ_is and ϕ_is separately) in this situation leads to diagnosis of convergence too early. So the safe option is to monitor all the parameters. If that is really prohibitive, the researcher should monitor a carefully chosen subset of parameters (preferably those with anticipated poor convergence rates) before the others. The parameters that are nearly nonidentifiable in the model usually have poor convergence rates associated with them.

Another question that arises, both during the formulation of the MCMC algorithm and diagnosis of convergence of the algorithm, is "how many chains to employ?" Opinions of experts vary on this issue. Geyer (1992) recommends using one very long chain because that will have the best chance of exploring the whole parameter space, especially for a slowly moving chain. On the other hand, Gelman and Rubin (1992a; 1992b) recommend running several long chains starting from over-dispersed starting values because comparing several seemingly converged chains might reveal genuine differences if the chains have not yet reached stationarity. Gelman (1995) has an example showing that two chains of simulated values for the same parameter may stabilize at different regions of the sample space. This phenomenon may happen with complicated multi-parameter target densities, especially multi-modal ones. Also, as Gelman (1995) argues, substantive flaws in the model or programming errors are often found by comparing parallel sequences. Patz and Junker (1999a) suggest that combination of the two above-mentioned approaches is useful in practice. Note that a number of popular MCMC convergence diagnostics work only for multiple chains. Because experts recommend the use of a collection of convergence diagnostics, it is wise to run multiple long chains (practitioners routinely use three to ten chains with five being the most common number).

3. Convergence Diagnostics

Cowles and Carlin (1996), Brooks and Roberts (1998), Robert (1998, chap. 2), and Mengersen, Robert, and Guihenneuc-Jouyaux (1999) are excellent reviews of MCMC convergence diagnostics. This document discusses the diagnostics that are easy to understand and compute. The diagnostics are classified under a number of broad headings.

Simple Graphical Methods

A number of simple graphical tools exist for MCMC convergence assessment. These tools examine the chain(s) of values generated for each parameter to determine if the simulation process stabilizes in some sense. These tools, although very simple and easy to implement, often provide useful feedback about the convergence of the MCMC.

Time-series Plots

Creating a time-series plot for each parameter in the model is the most popular check for convergence of an MCMC algorithm. A time-series plot in this context is a plot showing the generated values of a parameter for each iteration in a chain. Usually, a line connects the successive points of such a plot for ease of viewing the path traversed by the chain.

If different segments of a time-series plot for a parameter seem to have traversed different parts of the sample space or if there is a clear pattern (e.g., always increasing) in such a plot, the MCMC algorithm may not have converged. If an MCMC algorithm consists of multiple chains, the time-series plots are often examined after overlaying the generated values on a common graph for each parameter. Chains not traversing the sample space in the same way provides evidence of lack of convergence. For example, five chains shown in the time-series plot in Row 1 and Column 3 (marked "Slope: Item 6") of Figure 1 have hardly mixed together, providing evidence of lack of convergence of the MCMC. On the other hand, the chains in the plot in Row 2 and Column 3 (marked "Difficulty: Item 9") mix together after an initial *burn-in* (the number of iterations towards the beginning when the chain has not converged yet) period of 100 iterations and do not provide any evidence against convergence of the MCMC.

Some practitioners monitor the log of the posterior density (or, a multiple of it) at the current state of the chain as well. If the log-posterior density has an increasing trend, the chain has not reached the main mode yet. If the density is going down, the chain probably started near a mode around which there is little probability mass and is going to a more representative part of the distribution.

Running Mean Plots

A time-series plot of the running mean for each parameter in each chain is also useful. The running mean is the mean of all sampled values up to a given iteration (Smith, 2001). Usually, running means are plotted at every k-th iteration. We use $k = 50$ in this work. If the algorithm has converged, the running mean should stabilize at the posterior mean for each parameter. The running mean plots are quick summaries of the time-series plots and are easier to look at. However, these plots look only at the mean of the parameters and hence are inadequate.

Plot of Autocorrelation Functions

Looking at the plot of the autocorrelation function (ACF) for each parameter is a good practice. This tool is not strictly a convergence diagnostic tool, but helps indirectly to assess convergence of the MCMC algorithms. An MCMC algorithm generating highly autocorrelated parameter values will need a large number of iterations to be able to traverse the whole sample space of the parameters. Also, ACFs help interpret the time-series plots. Chains traversing the sample space very slowly or multiple chains traversing different regions of the sample space may be the result of high autocorrelation or multi-modality. Hence, some practitioners recommend attaching the values of the ACFs with the time-series plots discussed earlier.

Similar (not graphical) measures, though not graphical, are the posterior crosscorrelations among parameters suspected of being nearly confounded; high crosscorrelations may indicate the need for a reparameterization of the model.

Another graphical method is the CUSUM method (Yu & Mykland, 1998).

Methods Using Ratio of Dispersions

These methods consist of running multiple chains with different starting values and then comparing the dispersion between the chains against that within the chains.

Gelman-Rubin Convergence Measure and Its Improved Version

Gelman and Rubin (1992b) suggest running multiple chains with overdispersed starting values to compute (i) an estimate of the posterior distribution, and, (ii) a *reduction*

factor reflecting how much sharper the distributional estimate might become if the simulations were continued indefinitely.

The method consists of the following steps.

1. An overdispersed estimate of the target distribution is obtained. For example, the researcher can compute the modes using a maximization algorithm and then approximate the target distribution as a mixture of a number of t distributions centered at the modes.

2. Starting values for the desired number of independent chains are generated from the overdispersed estimate of the target distribution. Standard practice is to run 3 to 10 chains.

3. For each scalar quantity of interest, one computes a "potential scale reduction factor" (PSRF), which estimates the factor by which the scale of the Student t density that approximates the posterior distribution of a scalar parameter might be reduced if the simulations were continued indefinitely. Mathematically, the PSRF is given by

$$PSRF = \sqrt{\frac{n-1}{n} + \frac{1}{n}\frac{B}{W}},$$

where B is the variance between the means of the m chains, W is the average of the m within-chain variances, and n is the number of iterations of the chains (some researchers take n iterations after discarding first few iterations as burn-in). The quantity B is initially larger than W (as the initial values of the chains are overdispersed), making the PSRF considerably larger than 1 initially. However, as n becomes larger, PSRF declines to 1 if the algorithm converges; then the chains are no longer influenced by their starting values and have traversed the whole of the sample space. For most examples, values less than 1.2 or 1.1 are acceptable, but if a higher level of accuracy is required, one might want values even closer to 1.

This method is implemented in BUGS (Spiegelhalter, Thomas, Best, & Gilks, 1995), a software package used frequently by researchers to fit Bayesian statistical models with the MCMC algorithms. Also, the software packages CODA (Best, Cowles, & Vines, 1995), and

BOA (Smith, 2001), both of which are a collection of S-plus (Venables & Ripley, 2000) programs, implement this method. Further, writing a general program implementing this method is straightforward.

Univariate Measures Suggested by Brooks and Gelman

Gelman-Rubin convergence measure is basically a comparison of the between-chain and within-chain moments of the second order. Brooks and Gelman (1998) suggest comparing between-chain and within-chain moments other than those of the second order.

They also suggest an alternative to Gelman-Rubin convergence measure based on lengths of confidence intervals for each parameter. From each of the m chains, $100(1-\alpha)\%$ confidence intervals are computed, thus forming m within-chain intervals for each parameter. Then an overall $100(1-\alpha)\%$ confidence interval is computed from the entire set of sampled values (obtained by pooling the m chains) of each parameter. One may then compute, for each parameter,

$$\hat{R}_{\text{interval}} = \frac{\text{length of overall interval}}{\text{average length of the within-chain intervals}},$$

This is considerably simpler than the Gelman-Rubin measure and does not make any distributional assumption on the posterior distribution.

Brooks and Gelman (1998) also suggest a refinement of the PSRF to account for the sampling variability in the variance estimates. They suggest monitoring the "corrected scale reduction factor" (CSRF) , where

$$CSRF = \sqrt{\frac{d+3}{d+1}} PSRF,$$

and d is the the estimated degrees of freedom (obtained using the method of moments) of the Student t density that approximates the posterior distribution of the quantity of interest.

Brooks and Gelman (1998) also suggest a graphical approach to look at the CSRF as well as the corresponding pooled variance and the average within-chain variance (because at convergence, the two variances should stabilize together along with the PSRF and CSRF converging to 1) for each parameter. The software packages BUGS (Spiegelhalter, Thomas, Best, & Gilks, 1995) and BOA (Smith, 2001) implement this approach.

Brooks-Gelman MPSRF

Brooks and Gelman (1998) also suggest a multivariate extension of the PSRF to assess convergence of several parameters in a model simultaneously.

Denote the p-dimensional parameter vector as $\boldsymbol{\theta}$. Let $\boldsymbol{\theta}_i^t$ denote the generated value of the parameter vector at iteration t of chain i of the MCMC algorithm. Then the posterior variance-covariance is estimated by

$$\widehat{\boldsymbol{V}} = \frac{n-1}{n}\boldsymbol{W} + \left(1 + \frac{1}{m}\right)\frac{\boldsymbol{B}}{n},$$

where n is the number of iterations of the chains,

$$\boldsymbol{W} = \frac{1}{m(n-1)}\sum_{i=1}^{m}\sum_{t=1}^{n}\left(\boldsymbol{\theta}_i^t - \bar{\boldsymbol{\theta}}_{i.}\right)\left(\boldsymbol{\theta}_i^t - \bar{\boldsymbol{\theta}}_{i.}\right)'$$

and

$$\boldsymbol{B} = \frac{n}{m-1}\sum_{i=1}^{m}\left(\bar{\boldsymbol{\theta}}_{i.} - \bar{\boldsymbol{\theta}}_{..}\right)\left(\bar{\boldsymbol{\theta}}_{i.} - \bar{\boldsymbol{\theta}}_{..}\right)'$$

denote the estimated p-dimensional within- and between-chain covariance matrix of the parameter respectively. Some researchers compute the MPSRF from n iterations of the MCMC algorithm after discarding first few iterations as burn-in.

Brooks and Gelman (1998) show that if λ_1 is the largest eigen value of the symmetric and positive definite matrix $\boldsymbol{W}^{-1}\boldsymbol{B}/n$, then the quantity

$$\widehat{R_p} = \frac{n-1}{n} + \left(\frac{m+1}{m}\right)\lambda_1$$

tends to 1 if the chains mix together as sample size becomes large. They call the quantity $\widehat{R_p}$ as the "multivariate PSRF" or MPSRF.

The MPSRF takes values in the same scale as the PSRF so that values near 1 would indicate convergence of the chains. The big advantage of MPSRF is that it is an approximate upper bound to the maximum of the univariate PSRFs over all the variables in the MCMC. Their work shows that a graphical display of the MPSRF is a very useful convergence diagnostic simply because it may detect lack of convergence associated with the interaction of the scalar parameters, which individual PSRF plots cannot. Brooks and Gelman (1998) give an example of monitoring the PSRF with only a subset of parameters

that may lead one to the wrong conclusion. Therefore, for a high-dimensional problem, it is much wiser to examine the MPSRF (that combines convergence information of all the parameters and their interactions) than looking at a few PSRFs. This diagnostic tool is available in the BOA package (Smith, 2001).

Methods Based on Spectral Analysis

The basic idea of an MCMC algorithm is to create a stationary time-series. Naturally, time-series methods are used to assess convergence of the algorithms. We discuss two methods that use spectral analysis to obtain variance estimates of the parameters.

Geweke Convergence Diagnostic

The Geweke (1992) convergence diagnostic is appropriate for the analysis of individual chains when convergence of the mean of some function of the sampled parameters is of interest. The chain is divided into two "windows" containing a set fraction of the first and the last iterations. Geweke (1992) proposes a method to compare the mean of the sampled values in the first window to the mean of the sampled values in the last window. There should be a sufficient number of iterations between the two windows to reasonably assume that the two means are approximately independent. This method produces a Z statistic that is calculated as the difference between the two means divided by the asymptotic standard error of the difference, where the variance is determined by a spectral density estimation. As the number of iterations increases, the distribution of the Z statistic approaches the N(0,1) distribution if the chain has converged.

A look at the p-value of the Z statistic for each quantity of interest can provide evidence against convergence. Geweke suggests using 0.1 and 0.5 as the two fractions that define the windows. Geweke's method is available in the statistical packages CODA (Best, Cowles, & Vines, 1995) and BOA (Smith, 2001). The latter also produces a Geweke plot that shows the values of the Z statistic for each parameter in successively smaller segments of the chain.

Heidelberger and Welch Convergence Diagnostic

Schruben (1982) and Schruben, Singh, and Tierney (1983) suggest detecting nonstationarity in simulation output with a spectral analysis approach to estimate the variance of the sample mean. They use Brownian bridge theory and the Cramer-von Mises statistic (von Mises, 1931) to test the null hypothesis of stationarity of a Markov chain.

Heidelberger and Welch (1983) use the above test to propose a comprehensive procedure for generating a confidence interval of prespecified width for the mean of a parameter when the chain has an *initial transient* (a state when the algorithm has not reached stationarity yet). This procedure applies to a single chain.

The user prespecifies ϵ, the desired relative half-width for confidence intervals. The stationarity test of Schruben (1982) and Schruben, Singh, and Tierney (1983) is applied to the chain. If the test rejects the null hypothesis, the first 10% iterations of the chain are discarded and the stationarity test is repeated. The process continues until the resulting chain passes the stationarity test or more than 50% of the iterations have been discarded. The latter event (failure of the chain to pass the stationarity test) indicates the need to run the MCMC longer. A half-width test is performed on the portion of the chain that passes the stationarity test for each parameter. Spectral density estimation provides an estimate of the standard error of the mean. This estimate leads to an estimated half-width of the confidence interval for the mean. If the latter estimate is less than ϵ times the sample mean from the retained portion of the chain, the process stops and the sample mean and confidence interval are reported.

The Heidelberger and Welch diagnostic is implemented in BOA (Smith, 2001). The software produces a table showing for each parameter if the parameter passed or failed the stationarity test and half-width test, the number of iterations retained and discarded, the Cramer-von Mises statistic, sample mean, and the estimated half-width.

Methods Based on the Theory of Markov Chains

Raftery and Lewis Convergence Diagnostic

Raftery and Lewis (1992; 1996) suggest a method that is appropriate for individual chains and tests for convergence of the chain to the stationary distribution and estimates the run-lengths needed to accurately estimate quantiles of functions of the parameters.

The user has to specify the quantile of interest q, the desired level of accuracy r in estimating the quantile, and the probability s of attaining the indicated degree of accuracy. This method estimates the burn-in (the number of iterations that should be discarded) M and the further number of iterations needed to estimate the specified quantile to the desired accuracy N; it also estimates k, the thinning value (so that every k-th iteration of the chain should be used for inference). If the MCMC was run for fewer iterations than M, desired level of accuracy has not been reached and one should run the MCMC for longer.

This approach is based on two-state Markov chain theory, as well as standard sample size formulas involving binomial variance. Let $\theta_t, t = 1, 2, \ldots$ denote the generated values of a parameter θ from the MCMC algorithm, where the subscript t represents the iteration numbers. A binary sequence $I_t, t = 1, 2, \ldots$ is then formed, where I_t is a 0/1 indicator as to whether θ_t is less than a particular cutoff. The cutoff, which is an estimate of the q-th quantile of interest, is computed from the MCMC output. Although the binary transformations I_ts contain only partial information about the θ_ts, they form a process that can take only two values (zero and one) and are much easier to handle mathematically. The I_ts do not form a Markov chain, but k is determined so that $I_{1+(t-1)k}, t = 1, 2, \ldots$ is a Markov chain. Standard results for two-state Markov chains (for example, Cox & Miller, 1965) are then used to monitor the convergence of $I_{1+(t-1)k}, t = 1, 2, \ldots$ to a stationary distribution, yielding estimates of M and N.

In a practical setting, several quantiles may be of interest for a number of parameters. Then this procedure should be applied to all of those combinations and the maximum values of M and N should be taken as an estimate of the burn-in and the required number of iterations respectively.

This method is available in BOA (Smith, 2001). Also, *gibbsit*, a Fortran program

written by Raftery and Lewis implementing this diagnostic, may be obtained free by sending an e-mail message with subject line "send gibbsit from general" to *statlib@stat.cmu.edu*.

Garren and Smith (1995) suggest another method based on Markov chain theory. However, the method is more complicated, both conceptually and computationally.

Other Methods

There are convergence diagnostics of other kinds as well. Some of them are:

- Empirical methods based on the *transition kernel* (the distribution from which a draw is made in the MCMC algorithm). Roberts (1992), Ritter and Tanner (1992), Liu, Liu, and Rubin (1992), etc. provide examples of this type of convergence dignostics.
- Regeneration and coupling methods. Johnson (1996), Propp and Wilson (1996), Mykland, Tierney, and Yu (1995), and Robert (1996) suggest methods of this type.
- A method based on score statistic by Fan, Brooks, and Gelman (2002).

Summary

Table 1 summarizes the diagnostic checks discussed so far providing their:

- nature (quantitative/graphical)
- requirement of the number of chains (single/multiple)
- popularity/importance to the practitioners of MCMC (in the author's opinion) on a 4-point scale, where "1" means "extremely popular" and "4" means "used sometimes."

4. Example 1: Application to NAEP Data

To examine how the different diagnostics compare with each other, this section studies the performance of the different MCMC convergence diagnostics on a real data set. The data are from the National Assessment of Educational Progress (NAEP) Math OnLine

Table 1: ***Summary of the MCMC Convergence Diagnostics***

Name of the method	Quantitative or graphical	Single/multiple chains	Popularity/ importance
Time-series plots	Graphical	Single/multiple	1
Running mean plots	Graphical	Single/multiple	3
ACFs	Graphical	Single	1
PSRF and CSRF	Quantitative	Multiple	2
Brooks & Gelman MPSRF	Quantitative	Multiple	4
Geweke	Quantitative	Single	2
Heidelberger & Welch	Quantitative	Single	4
Raftery & Lewis	Quantitative	Single	3

(MOL) special study (Sandene, Bennett, Braswell, & Oranje, in press). We consider a subsample that consists of the responses of 974 8th grade examinees to 16 multiple choice items in one test form. We fit the three-parameter logistic model (3PL) (Lord, 1980) to the data set. The model expresses the response function for the i-th person to the j-th item as

$$P_j(\theta_i) = c_j + \frac{1 - c_j}{1 + \exp\{a_j(b_j - \theta_i)\}},$$

where θ_i is the ability of individual i, and a_j, b_j, and c_j are the discrimination (slope), difficulty, and guessing parameters for the item j. Let us assume that the interest is in calibrating the items, i.e., in obtaining item parameter estimates only. Therefore, the ability parameters θ_is are treated as nuisance parameters. There are then 48 parameters to be estimated for the model fitted and those only are monitored for convergence. However, one may argue that the ability parameters θ_is should be monitored as well, but there are as many as 974 of them. Further, these parameters are well-behaved in the chains, rarely providing any problem with convergence.

Table 2 shows the marginal maximum likelihood estimates of the item parameters, obtained by the PARSCALE program of Muraki and Bock (1991). PARSCALE integrates out θ_is using a $N(0, 1)$ prior distribution. To complete the (Bayesian) model specification, assume independent $N(0, 1)$ prior distributions on θ_is and b_js, independent $N(0, 1)$ prior distributions on $\log_e(a_j)$s, and independent $N(-1.39, 0.5)$ prior distributions on $\log\left(\frac{c_j}{1-c_j}\right)$s. Use an MCMC algorithm to fit the Bayesian model. More specifically, use a Metropolis-within-Gibbs algorithm, where, in each iteration of the algorithm, we have

Table 2: ***PARSCALE Estimates of the Item Parameters for the MOL Data***

Item No.	Slope	Difficulty	Guessing
1	0.97	0.50	0.20
2	0.80	-0.65	0.15
3	0.56	-0.42	0.17
4	1.01	1.13	0.16
5	0.93	1.26	0.14
6	0.50	4.81	0.23
7	0.99	-1.30	0.16
8	0.91	-0.83	0.14
9	1.22	0.29	0.19
10	0.85	-1.29	0.16
11	0.96	-0.94	0.17
12	0.93	0.08	0.14
13	0.90	0.59	0.17
14	1.06	0.97	0.14
15	1.08	1.20	0.14
16	0.78	1.15	0.24

a Gibbs step each for the θ_is, a_js, b_js, and c_js. Within each of these Gibbs steps, draw the individual parameters using Metropolis steps. Run five chains with widely dispersed starting values for 500 iterations. Finally, use the BOA software (Smith, 2001) to apply the diagnostics on the output of the MCMC.

Results Using the Diagnostics for 500 Iterations

Figure 1 shows the time-series plots for five of each type of item parameters. The five with the worst convergence property are shown. Figure 2 shows the corresponding running mean plots. The plots show that although the five chains do not differ much after an initial burn-in of one or two hundred for some (e.g., all the difficulty parameters) of these 15 parameters, they differ significantly for many of them, e.g., for most of the c_js. The difference indicates that the algorithm has not converged yet, and the program needs to run for a higher number of iterations.

Plot of ACFs for two of each type of parameters (denoted a1, a10, b3, b13, c5, and c15) from one chain of 500 iterations are provided in Figure 3. This figure shows part of the reason why the chains are nonmixing if the MCMC program runs for only 500

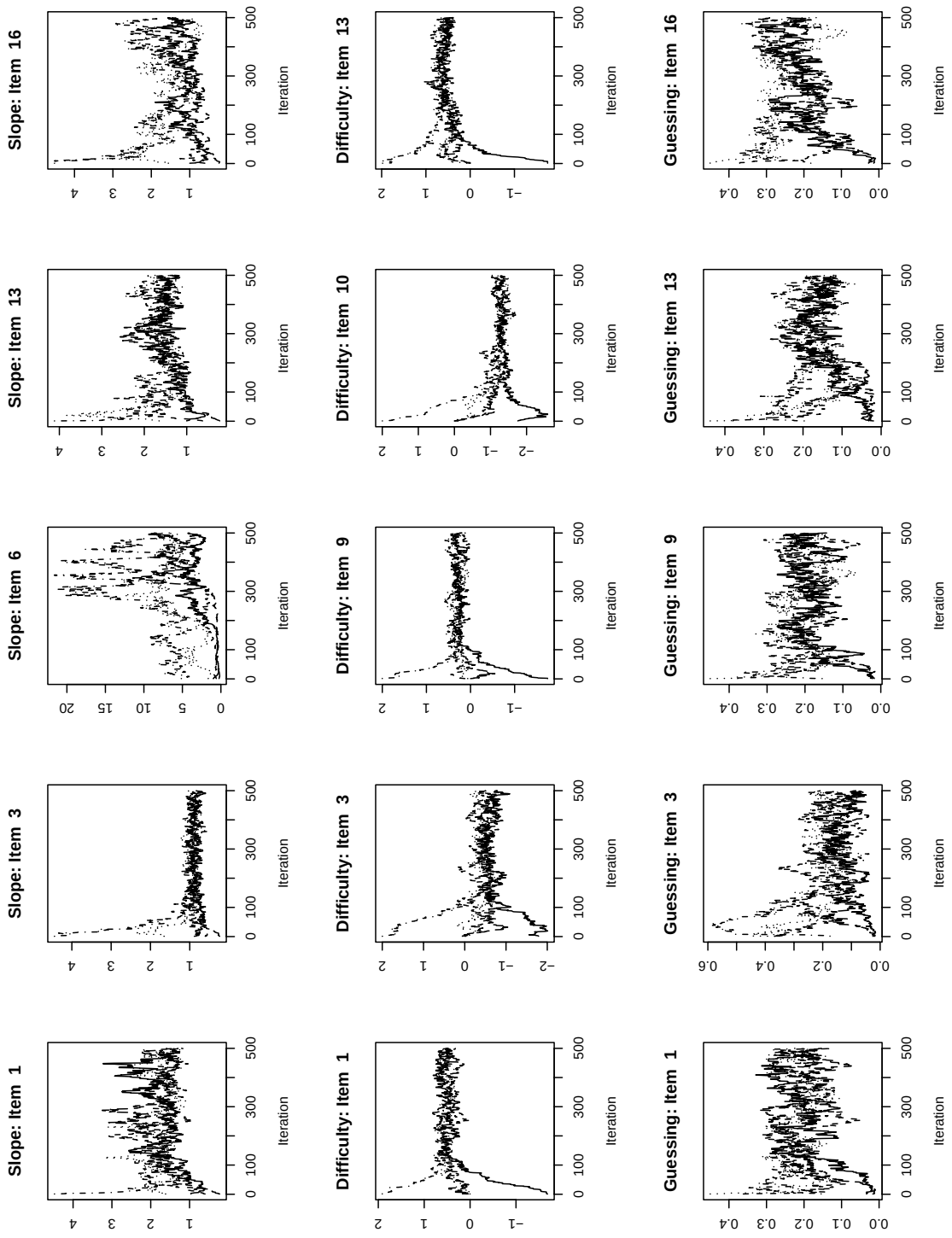

***Figure* 1: Time-series plots for five a_js, five b_js, and five c_js for five chains of 500 iterations each for the NAEP example**

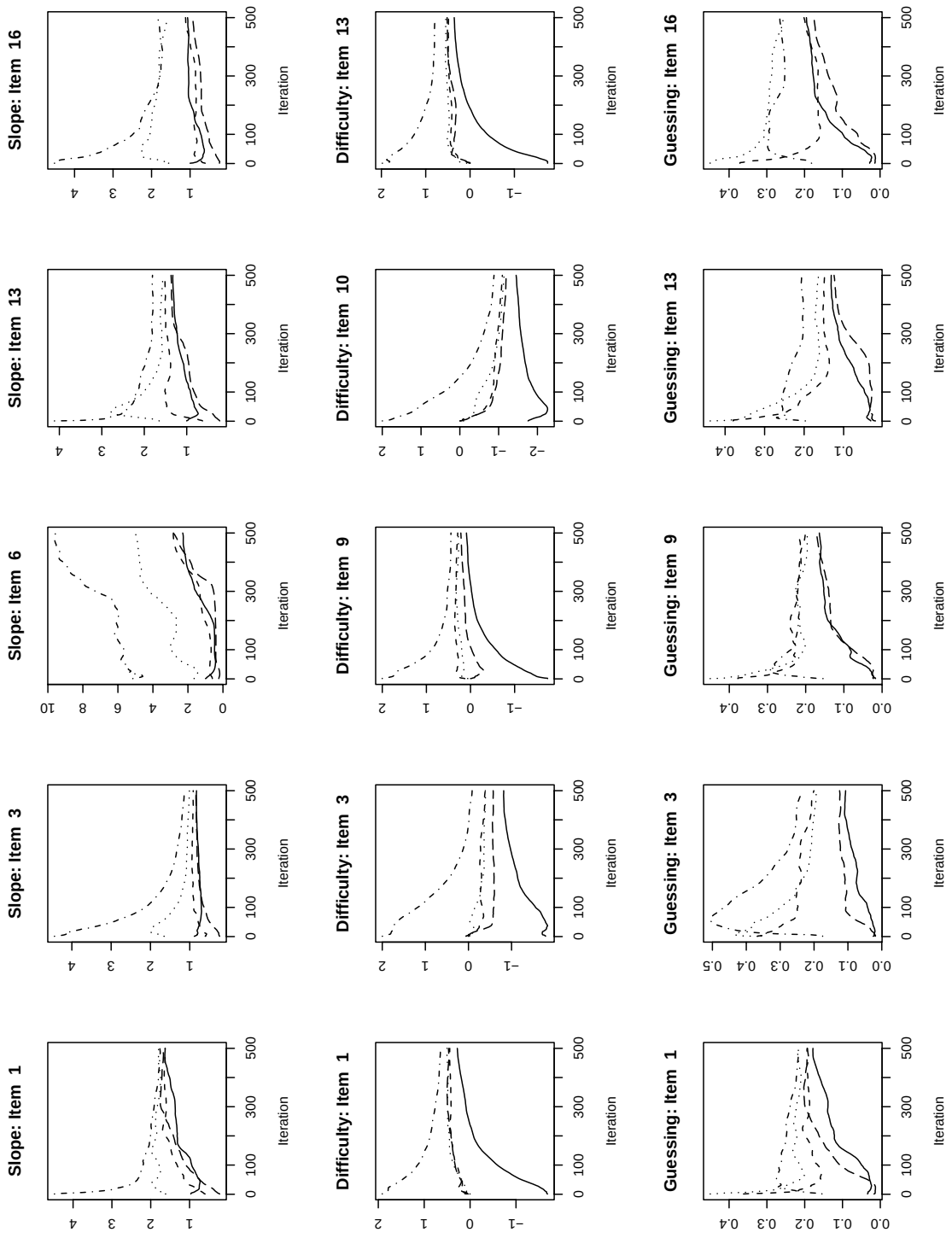

***Figure* 2: Running mean plots for five a_js, five b_js, and five c_js for five chains of 500 iterations each for the NAEP example**

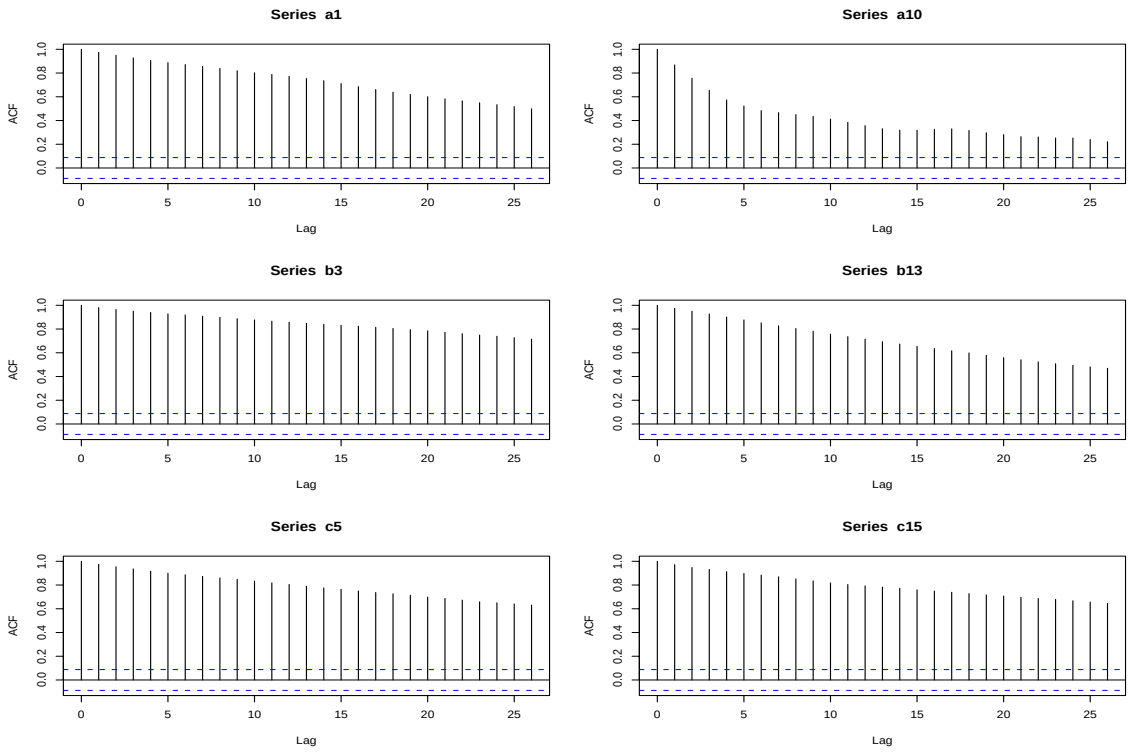

***Figure* 3: ACF plots for six parameters for one chain of 500 iterations**

iterations—the autocorrelations are quite high even at lag 25 and hence the chains are moving very slowly. The program has to run for a larger number of iterations to give the chains a chance to traverse the whole sample space.

The crosscorrelations for the generated values of the parameters are quite high as well. Patz and Junker (1999a; 1999b) observe high crosscorrelations among the parameter estimates for the 3PL (Lord, 1980) model. In this example, there exist low to moderately high crosscorrelation within a_js, within c_js, and between generated b_js and c_js; moderate to high correlation between generated a_js and b_js; and very high correlation within generated b_js. These crosscorrelations result in the high autocorrelation in the chains because in any iteration of the MCMC, a parameter-value is drawn given the current values of the other parameters. Suppose one starts the iteration K in the MCMC by drawing a_1 given the values of the other parameters in iteration $(K-1)$. The generated value of the a_1 in iteration K then depends on the values of the other parameters in iteration $(K-1)$, which in turn are strongly related with the value of a_1 in iteration $(K-1)$. As an end-result, the

values of a_1 in iterations $(K-1)$ and K are strongly correlated.

The Geweke's convergence diagnostic produces high Z-scores and hence very low p-values for almost all the parameters in all the chains. Indicating a significant difference between the first part and the last part of the chains, the result shows that the algorithm is far from convergence.

The Heidelberger and Welch diagnostic, when applied to the five chains separately, results in 30 to 35 of the 48 parameters failing the stationarity test or half-width test, clearly indicating the lack of convergence.

The Gelman-Rubin convergence diagnostic produces only one PSRF more than 1.2 (a_6 has PSRF 1.35). Therefore, blindly applying the rough cut-off of 1.2 on the PSRFs may lead to the false impression that the MCMC algorithm has converged. However, there are 13 PSRFs between 1.1 and 1.2. The corrected PSRFs appear to be more powerful, yielding four values more than 1.2 (including a value of 1.37 for a_6) and 26 values between 1.1 and 1.2. The Brooks-Gelman MPSRF is 3.0, indicating that the algorithm is yet to converge. This result shows that the PSRFs may not always be powerful in detecting lack of convergence. Also, the MPSRF seems to be more powerful than the PSRF or its corrected version.

The Raftery and Lewis measure, when run with the BOA (Smith, 2001) default values of quantile = 0.025, accuracy = 0.005, and probability = 0.95, yielded the following warning:

> Available chain length is 500. Rerun simulation for at least 3,746 iterations OR reduce the quantile, accuracy, or probability to be estimated.

Changing the accuracy to 0.01 results in the warning that the simulation should be run for at least 937 iterations before the method can provide any estimate of the number of iterations required for convergence. Changing the quantile to 0.5 with a desired accuracy of 0.01 results in the warning that the simulation should be run for at least 9,604 iterations.

Looking at all the above, we decide to run the MCMC for five chains of 10,000 iterations each, mainly because of the high ACFs, high value of Brooks-Gelman MPSRF, high values of Gewekes Z-scores, and the suggestion by Raftery and Lewis method to run

for 9,604 iterations for quantile = 0.5 and accuracy = 0.01.

Results Using the Diagnostics for 10,000 Iterations

Figure 4 shows the time-series plots for the same subset of the item parameters plotted for the analysis with 500 iterations. The corresponding running mean plots are shown in Figure 5.

The plots show that the five chains do not differ much for the parameters after an initial burn-in period of one to two thousand iterations. The same is true for the other parameters in the model as well. This gives some evidence for the fact that the algorithm has probably converged.

The ACF plots look very similar to that with 500 iterations, showing high autocorrelation in the chains, and are not repeated. Also, just like with 500 iterations, there are significant crosscorrelations.

The Geweke's convergence diagnostic provides significant Z-scores (at 5% level) for more than one-third of the 48 parameters in all the chains of 10,000 iterations. Results are the same even when we ignore the first 2,000 iterations of the chains as burn-in and apply the diagnostic on the last 8,000. Further, changing the proportions in the first and last windows does not affect the large number of significant Z-scores. All of the above facts point to the lack of convergence.

The Heidelberger and Welch diagnostic, when applied to the five chains separately, results in 8 to 12 of the 48 parameters failing the stationarity test or half-width test, indicating that the chain has not converged yet.

The Gelman-Rubin convergence diagnostic applied with no burn-ins provides quite low PSRFs and corrected PSRFs for all the parameters, the highest value being 1.01 for the former and 1.05 for the latter. Moreover, the Brooks-Gelman MPSRF is 1.07, indicating that the five chains do not differ much and that the MCMC algorithm has converged.

Figure 6 shows a plot of the PSRF for successively larger segments of the chains for six c_js (denoted in the plot as g_4, g_5, $\ldots g_9$). The solid line shows the median of the estimated PSRF while the dashed line shows the corresponding 0.975 quantile. The plot for the other parameters look very similar. The plot suggests that the chains differ initially,

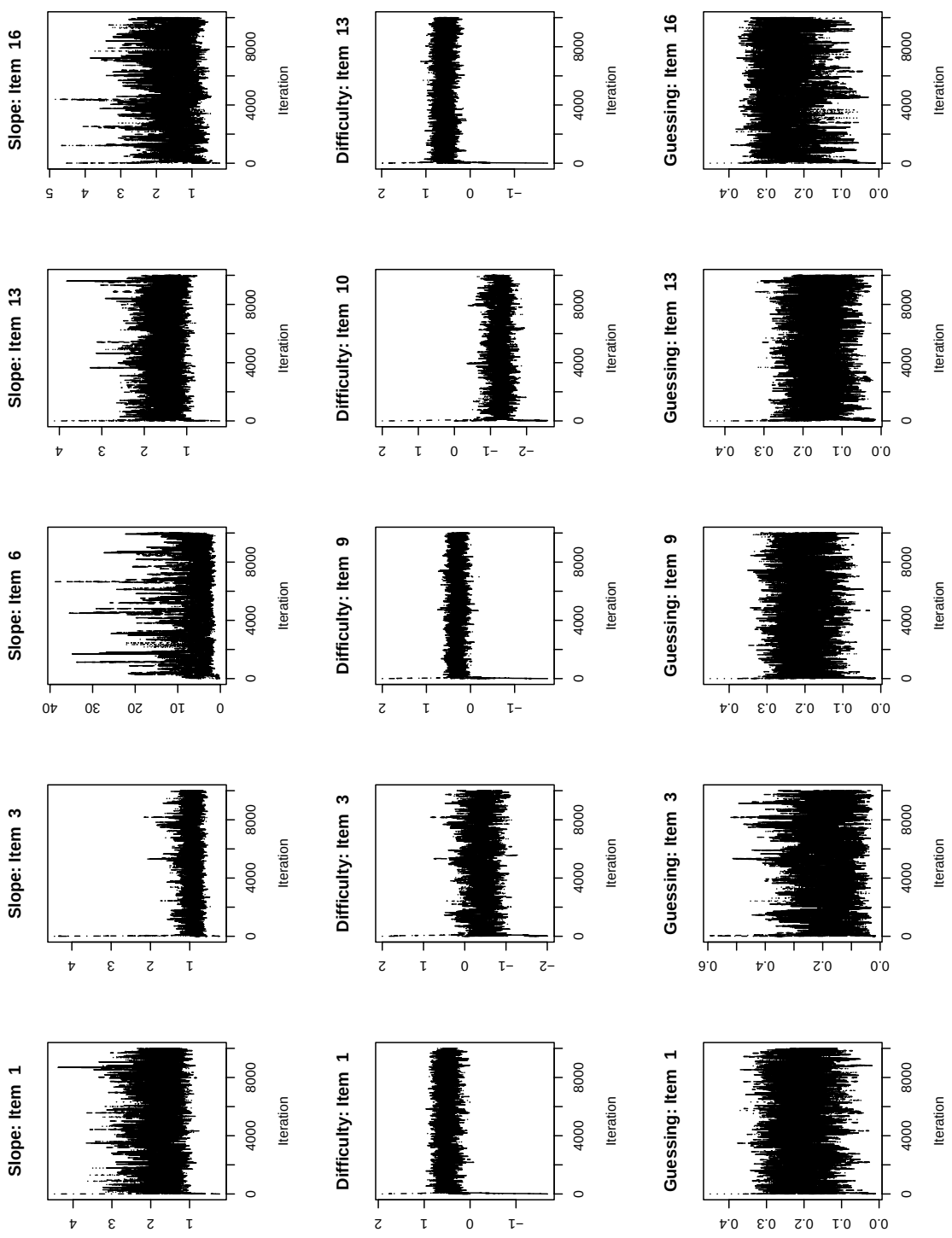

***Figure* 4: Time-series plots for five a_js, five b_js, and five c_js for five chains of 10,000 iterations each for the NAEP example**

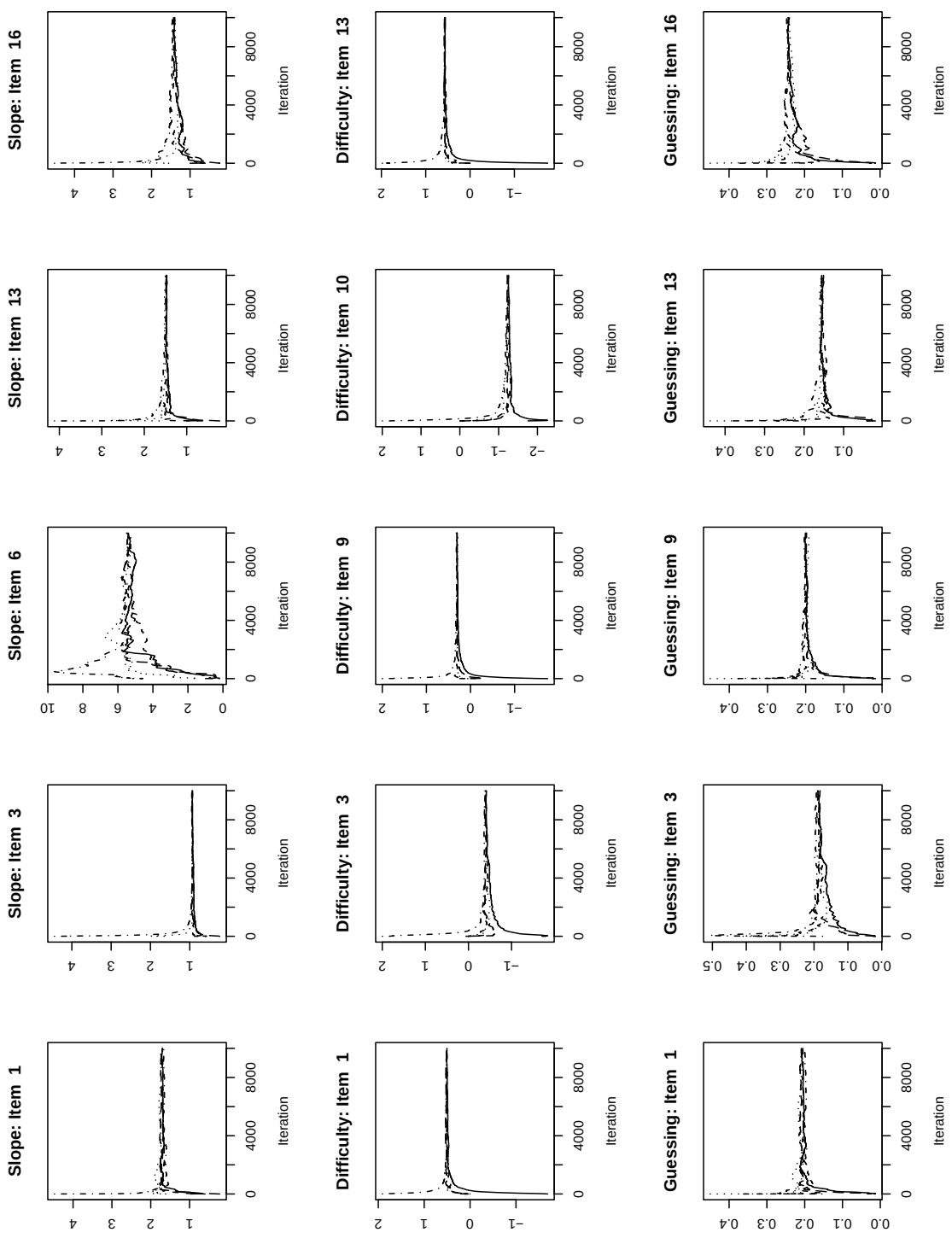

***Figure* 5: Running mean plots for five a_js, five b_js, and five c_js for five chains of 10,000 iterations each for the NAEP example**

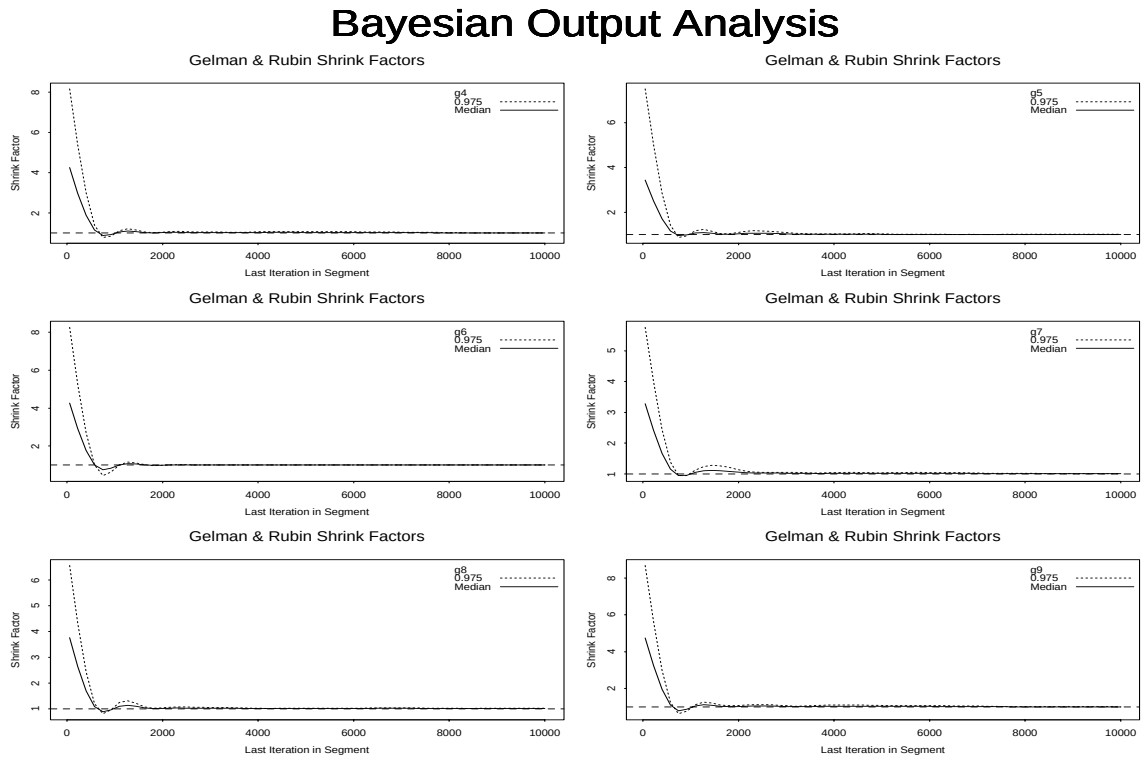

***Figure* 6: The plots for the Gelman-Rubin PSRFs for six c_js for five chains of 10,000 iterations**

but after 1,000 to 2,000 iterations, they mix together and traverse the same regions in the sample space.

Figure 7 shows a plot of the Brooks-Gelman MPSRF (denoted R_p) along with the maximum PSRF (denoted R_{max}) for successively larger segments of the chains. This plot provides very valuable information. It shows that the MPSRF far exceeds the maximum PSRF for the first few thousand iterations and hence making a decision of convergence based only on the PSRFs may lead one to false sense of security. So the MPSRF seems to be a very powerful tool for detecting convergence of the MCMC. Not only does it provide a one-number summary of the PSRFs, but it also detects lack of convergence when the PSRFs (or their corrected version) may fail to do so. This plot also suggests that although the chains differ significantly for the first few thousand iterations, they mix together after that and five chains of 10,000 iterations each is probably sufficient to ensure convergence of the chains. Also, the plot suggests using a burn-in of about 2,000 each.

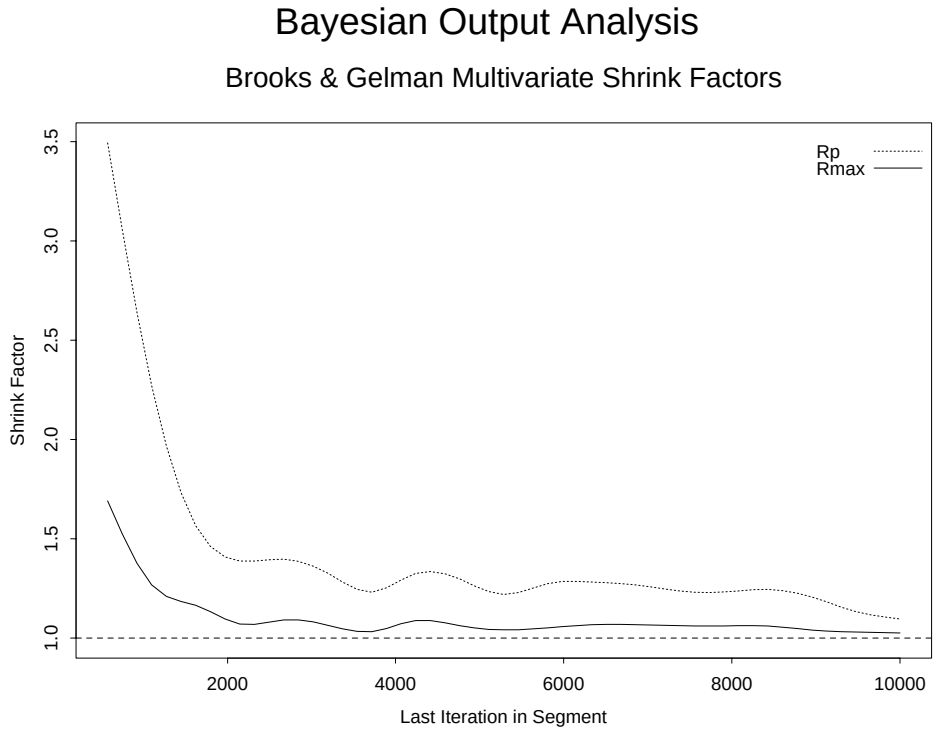

***Figure* 7: The plot of the Brooks-Gelman MPSRF for five chains of 10,000 iterations**

The Raftery and Lewis measure, applied to the chains with the values of quantile = 0.025, accuracy = 0.01, and probability = 0.95, provide the required number of iterations to achieve convergence between 7,600 to 45,540. The method suggests burn-ins between 39 and 207 and thinning the output by values between 6 to 23. Overall, the method suggests that if using one chain, it will be wise to use a chain of length at least 50,000.

Results Using the Diagnostics for 50,000 Iterations

We run one chain of length 50,000 and apply the Geweke's diagnostic on the last 40,000 iterations with the proportion in the two windows set at 0.3 and 0.5. There is only one significant Z-score, providing evidence in favor of convergence of the chain.

The Heidelberger and Welch diagnostic, when applied to the five chains separately, results in only 2 of the 48 parameters failing the stationarity test or half-width test, indicating that the chain has converged for all practical purposes.

The Raftery and Lewis measure, when applied to the chain with the values of quantile = 0.025, accuracy = 0.01, and probability = 0.95, provides the required number of iterations to achieve convergence that range between 6,180 to 46,728, making us more confident that the chain has converged.

Discussion

Looking at the above, it seems that if a researcher wishes to use only one chain, its length should be at least 50,000 with a burn-in of a few thousand to ensure the convergence. Alternatively, running five chains of 10,000 each and treating the first 1,000 to 2,000 iterations as burn-ins is also a safe option.

This study also shows that blindly applying only one convergence diagnostic may result in diagnosis of convergence too prematurely. For example, for five chains with 500 iterations each, applying the often used cut-off of 1.2 to the Gelman-Rubin diagnostic may lead one to prematurely conclude the convergence of the algorithm as there is only one PSRF above 1.2. In fact, for five chains of 1,000 iterations (results not shown here), all the PSRFs are below 1.1 (wrongly indicating that the algorithm has converged), whereas the other methods indicate a clear lack of convergence.

Also, there is considerable difference in the results produced by the diagnostics. This is because the different diagnostic tools examine distinct aspects of convergence. Hence, it is wiser to use a variety of diagnostic tools of different type rather than a single plot or statistic. The results reaffirm that automated convergence monitoring is unsafe and should be avoided—it is very difficult for a machine to look at the results from a variety of tools, judge them properly, and then take a collective decision.

5. Example 2: SCORIGHT

A *testlet* (Wainer & Kiely, 1987) is a subset of items generated from a single stimulus, e.g., a reading comprehension passage. The *testlet model* (Bradlow, Wainer, & Wang, 1999; Wang, Bradlow, & Wainer, 2000) takes into account the dependence of the responses of an examinee to the items within a testlet. SCORIGHT (Wang, Bradlow, &

Wainer, 2001) is recently developed software that uses an MCMC algorithm to fit the testlet model for analyzing item response data containing testlets.

Denote the response of the i-th person to the j-th item as y_{ij}. Suppose the item j belongs to the testlet $d(j)$. Assuming, without loss of generality, that there are only dichotomous items, the testlet model assumes

$$\begin{aligned} P(y_{ij}=1) &= c_j + (1-c_j)\frac{1}{1+\exp\{-t_{ij}\}}, \\ t_{ij} &= a_j\left(\theta_i - b_j - \gamma_{id(j)}\right). \end{aligned} \tag{5}$$

The term $\gamma_{id(j)}$ is the *testlet effect* (interaction) of testlet $d(j)$ with person i. Further, the model assumes that $\theta_i \sim N(0,1)$ and $\gamma_{id(j)} \sim N\left(0, \sigma^2_{d(j)}\right)$. A multivariate normal prior distribution is assumed on $\left(\log(a_j), b_j, \log\left(\frac{c_j}{1-c_j}\right)\right)'$ and noninformative prior distributions are assumed on the parameters of the multivariate normal prior distribution. The prior distribution on $\log\left(\sigma^2_{d(j)}\right)$ is normal and that on the parameters of this normal prior distribution is a noninformative distribution. Traditional maximum likelihood estimation to estimate the parameters of the model is very difficult to implement in this situation. The SCORIGHT software uses an MCMC algorithm to fit the model. However, SCORIGHT does not provide any check on the convergence of the MCMC, making it difficult for users of the software to decide how long to run the algorithm. Moreover, in the user's guide of the software, there is an example where an MCMC algorithm runs for 6,000 iterations (without any check of convergence), making many users of SCORIGHT believe that a few thousand iterations are good enough to ensure convergence of the MCMC algorithm used. However, a few thousand iterations may be too few to be sure about the convergence because the model is quite complex, with possibly correlated parameter estimates. This example examines the convergence of the Markov chains created by SCORIGHT by applying the different convergence diagnostics to the output of the software.

The Data Set

The data set consists of the response of 1,612 examinees to a recent achievement examination in English. The test consists of 60 items, divided among seven testlets. This

study considers only a subset of the items, 25 items from five testlets—five items each from three testlets and four and six items each from the other two testlets.

SCORIGHT fits the model (5) to the data. Run five chains with widely dispersed starting values. One can provide starting values for the item parameters a_j, b_j, and c_j in SCORIGHT. First fit a simple 3PL model (Lord, 1980) to the data using PARSCALE (Muraki & Bock, 1991) to get a rough idea of the variation of the item parameter estimates and form starting values based on that. For example, the estimates of b_js from a 3PL model fit to the data range between -2.4 to 2.3; hence use combinations of -3, 0, and 3 as starting values of b_js in the five chains. Chains mixing together even with such different starting values will be a good indication of the MCMC algorithm not depending on the starting value and hence converging.

We first run the program for 1,000 iterations and apply the diagnostics on the output. The version of SCORIGHT available to the author outputs the generated values of the item parameters a_j, b_j, and c_j; testlet variances $\sigma^2_{d(j)}$s; and examinee abilities θ_js. We monitor all of these except the examinee abilities θ_j (as in Example 1, we assume that we are calibrating and so the 1,612 examinee abilities are not monitored), which results in the monitoring of 80 parameters. It would be ideal to monitor the parameters of the prior distributions of the item parameters and the hyper-parameters, but the version of SCORIGHT used does not produce them. Therefore, the diagnostics will provide an estimate of the required number of iterations that is less than what it would be if we could monitor all the parameters.

Results Using the Diagnostics for 1,000 Iterations

Figure 8 shows the time-series plots for a few item parameters (with slowest convergence) of each type and logarithm of the testlet variances.

Figure 9 shows the corresponding running mean plots. The plots show that the five chains differ for a number of the parameters. Also, the generated values of the testlet variances are increasing and have not yet settled down. This shows that the algorithm has not converged yet and the program needs to run for many more number of iterations.

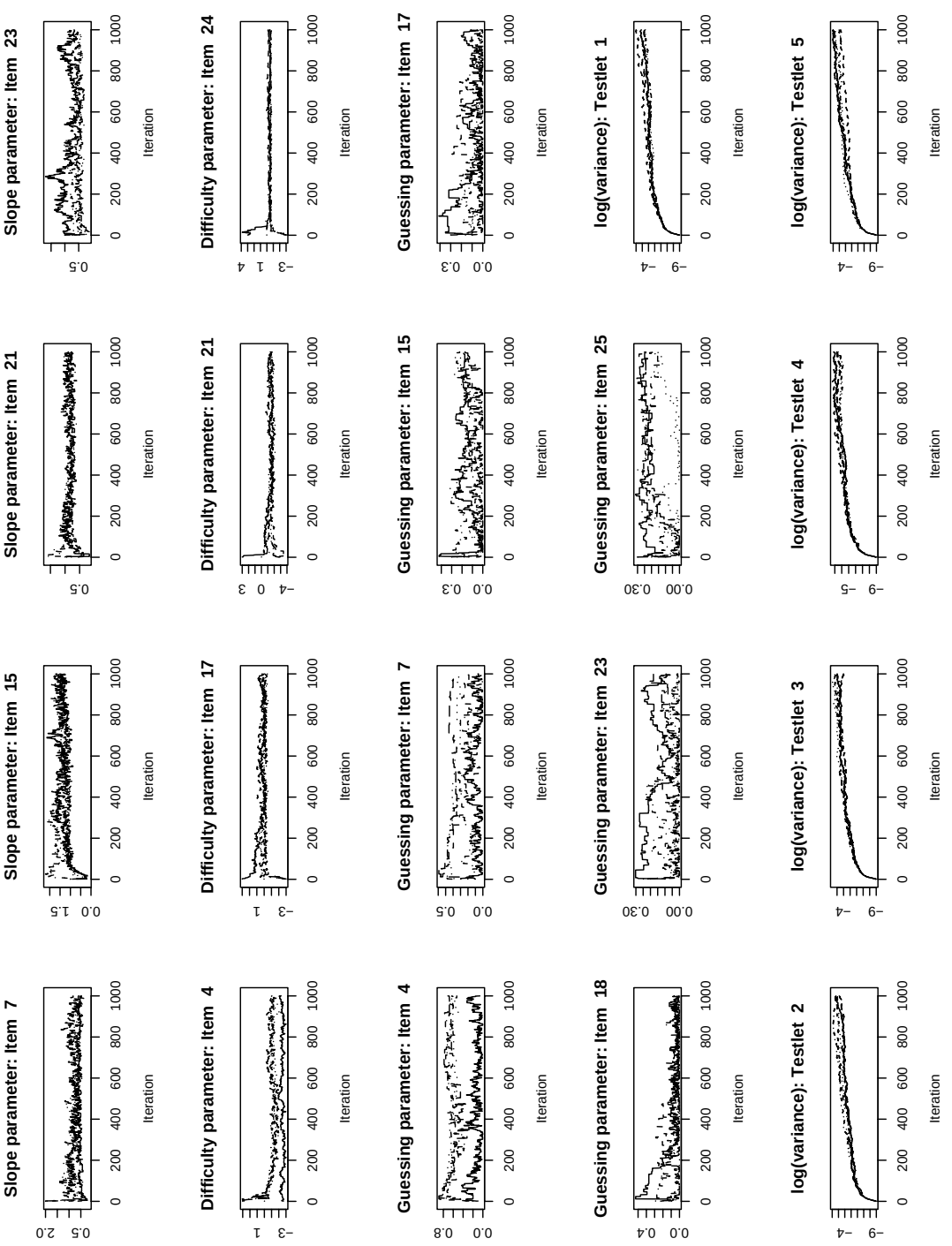

***Figure* 8: Time-series plots for four a_js, four b_js, seven c_js, and all five $\log\{\sigma^2_{d(j)}\}$s for five chains of 1,000 iterations each for the SCORIGHT example**

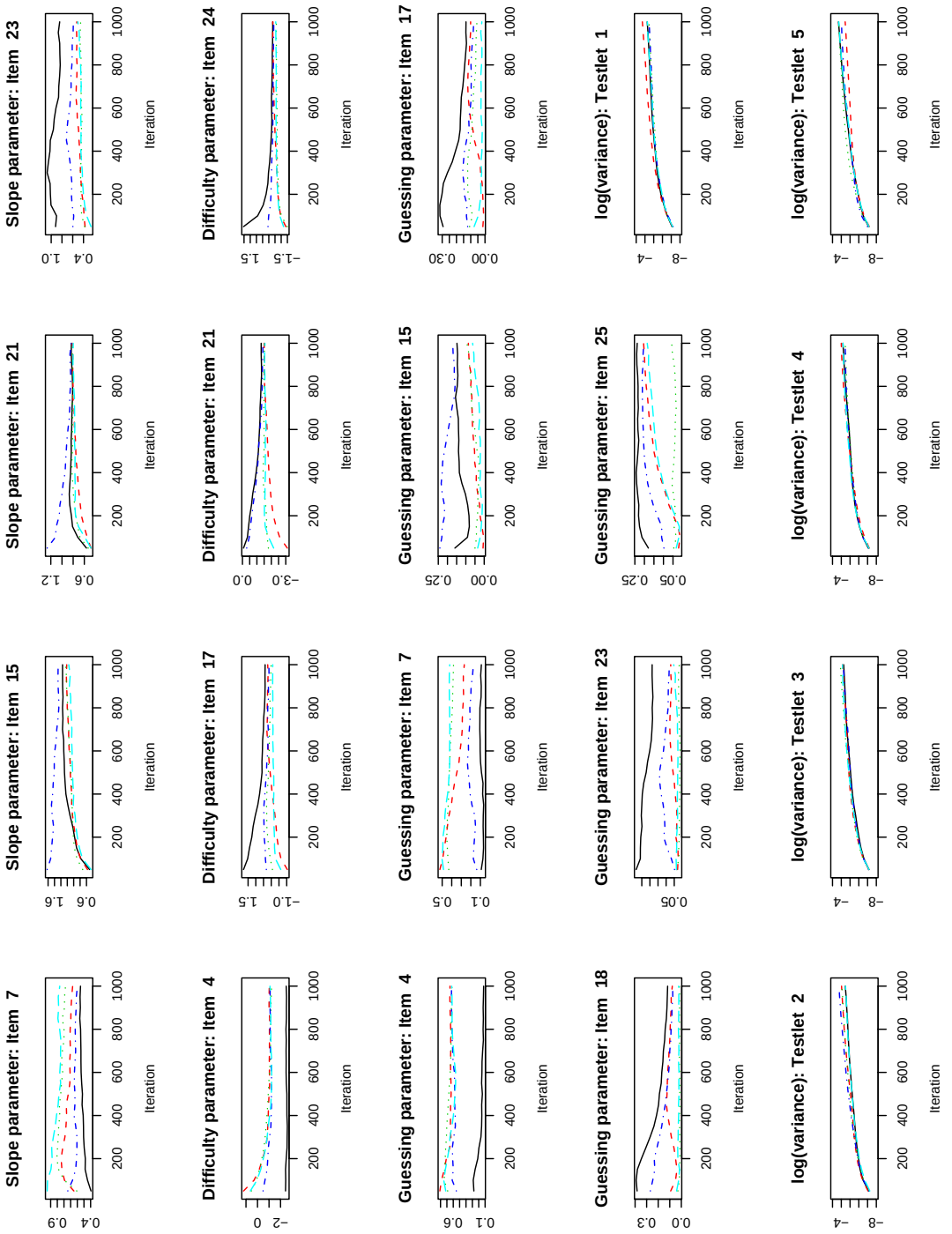

***Figure* 9: Running mean plots for four a_js, four b_js, seven c_js, and all five $\sigma^2_{d(j)}$s for five chains of 1,000 iterations each for the SCORIGHT example**

Figure 10 shows the plot of the ACFs for three c_js (denoted c7, c8, c9), and three $\sigma^2_{d(j)}$s (denoted v1, v2, v3) from one chain of 1,000 iterations. The autocorrelations are very

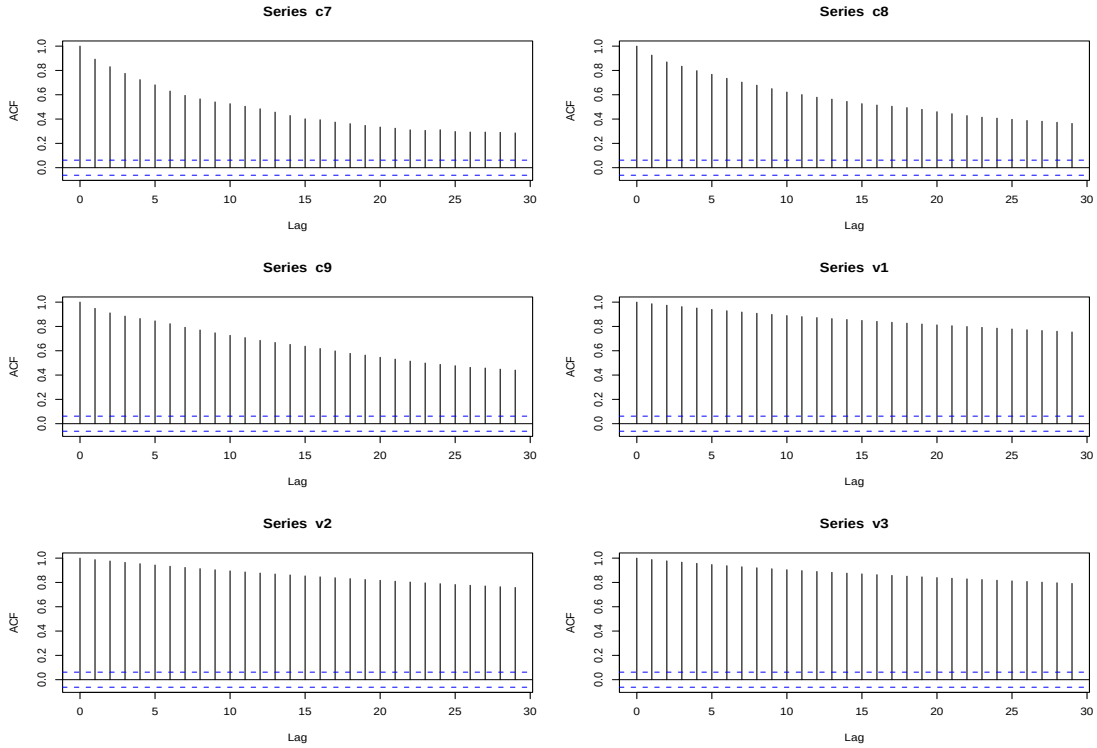

***Figure* 10: ACF plots for three c_js and three $\sigma^2_{d(j)}$s for one chain of 1,000 iterations**

close to 1 even at lag 30, indicating that the chains are moving very slowly. Therefore, the program has to be run for more iterations to give the chains a chance to traverse the whole sample space. The crosscorrelations of the parameters are quite high as in the previous example, resulting in high autocorrelation in the chains.

The Geweke's convergence diagnostic yields high Z-scores and hence very low p-values for almost all the parameters in all the chains, indicating that there is a significant difference between the first part and the last part of the chains and hence we are nowhere near convergence.

The Heidelberger and Welch diagnostic, when applied, results in 55 to 65 of the 80 monitored parameters failing the stationarity test or half-width test, clearly indicating that the chain is yet to converge.

When we apply the Gelman-Rubin convergence diagnostic, we get 14 PSRFs more than 1.2 (c_4 has the highest PSRF of 2.55). Hence, this measure also indicates to the lack of convergence of the MCMC algorithm. The corrected PSRFs are found to be more powerful, yielding 21 values more than 1.2 (including a value of 2.9 for c_4). The Brooks-Gelman MPSRF is 22.7, indicating that the algorithm is far from reaching convergence.

The Raftery and Lewis measure, when run with the BOA (Smith, 2001) default values of quantile = 0.025, accuracy = 0.005, and probability = 0.95 yields the warning that the simulation should be run for at least 3,746 iterations.

Looking at the above, we run SCORIGHT for five chains of 10,000 iterations each and apply the convergence diagnostics to the output.

Results Using the Diagnostics for 10,000 Iterations

The time-series plots for a few item parameters and logarithm of the testlet variances, the same ones plotted during the analysis for 1,000 iterations, are shown in Figure 11. The running mean plots for the same set of parameters are shown in Figure 12.

The plots show that the five chains still differ for a number of the parameters, e.g., for c_4 and c_{23}, indicating that we need to run the MCMC algorithm for more iterations. Figure 11 suggests that the starting values of $\sigma^2_{d(j)}$s (which could not be input by the user in the version of the software used) are poorly chosen in SCORIGHT and better starting values could have hastened the convergence of the MCMC algorithm.

The Geweke's convergence diagnostic yields high Z-scores and hence very low p-values for 65 to 70 parameters out of the 80 parameters in all the chains. The Heidelberger and Welch diagnostic obtains the same result and has 60 to 65 of the 80 monitored parameters failing the stationarity test or half-width test, clearly indicating that the chain has yet to converge.

When we apply the Gelman-Rubin convergence diagnostic, we find that all PSRFs are less than 1.05 (a_{23} has the highest PSRF of 1.04). CSRFs give similar values as well. Even after applying the diagnostic to the last half of the 10,000 iterations (as done by many practitioners), we get all PSRFs and CSRFs less than 1.1. These values imply the convergence of the MCMC algorithm.

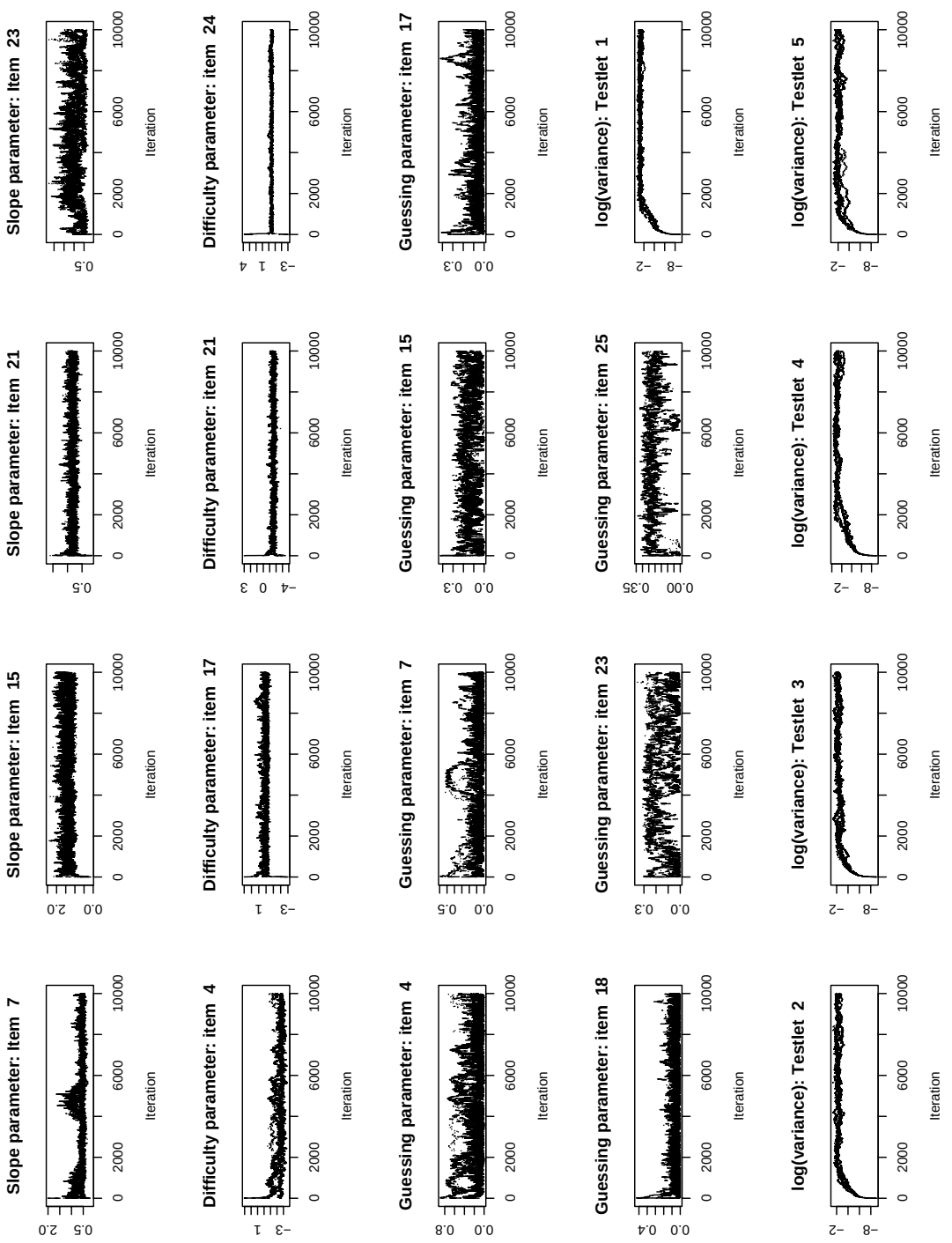

***Figure* 11: Time-series plots for four a_js, four b_js, seven c_js, and all five $\log\{\sigma^2_{d(j)}\}$s for five chains of 10,000 iterations each for the SCORIGHT example**

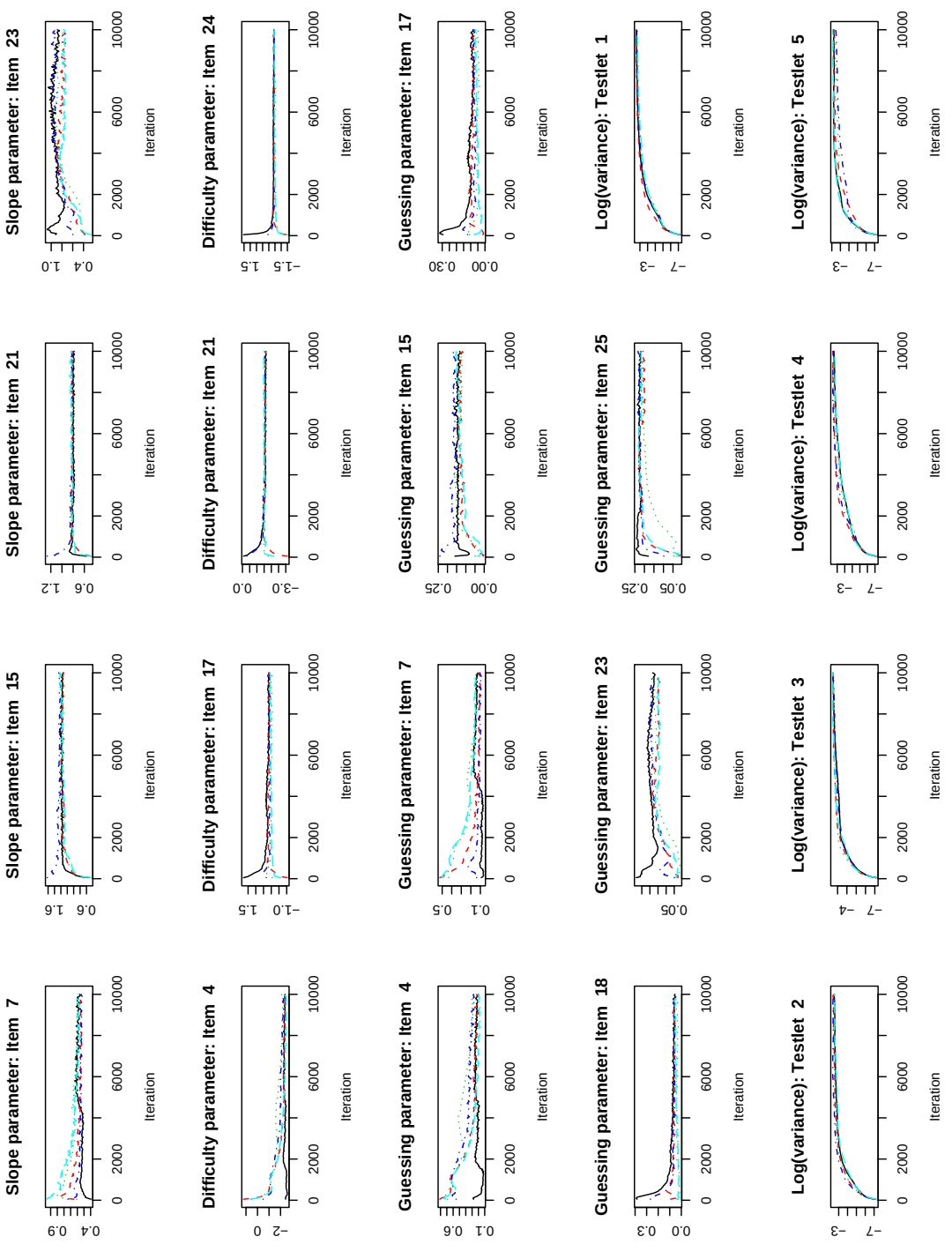

***Figure* 12: Running mean plots for four a_js, four b_js, seven c_js, and all five $\sigma^2_{d(j)}$s for five chains of 10,000 iterations each for the SCORIGHT example**

However, The Brooks-Gelman MPSRF is 1.71 when applied to the 10,000 iterations and 2.17 when applied to the last 5,000 iterations. Figure 13 shows a plot of the Brooks-Gelman MPSRF (denoted R_p) along with the maximum PSRF (denoted R_{max}) for successively larger segments of the chains. The plot indicates that the MPSRF is steadily

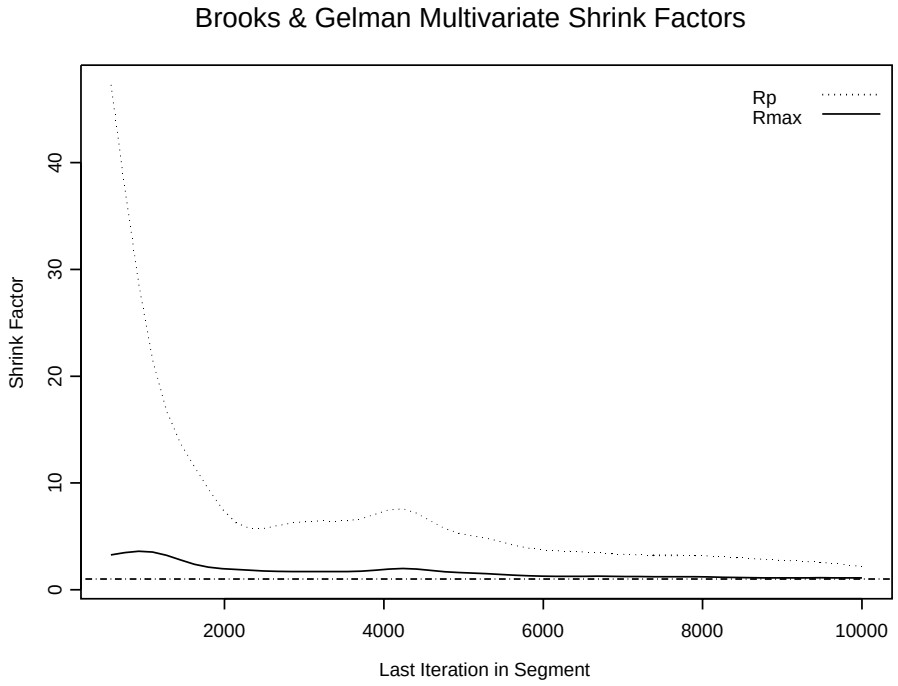

***Figure* 13: The plot of the Brooks-Gelman MPSRF for five chains of 10,000 iterations each for the SCORIGHT example**

decreasing even after 10,000 iterations, and hence the MCMC algorithm is yet to converge. This is another classic example of the MPSRF detecting lack of convergence where the PSRF and CSRF fail to do so.

The Raftery and Lewis measure, when run with quantile = 0.025, accuracy = 0.01, and probability = 0.95, results in values of required number of iterations that range from 9,000 to 55,000.

We then decide to run the MCMC for five chains of 50,000 iterations each.

Results Using the Diagnostics for 50,000 Iterations

Figure 14 shows the time-series plots for a few item parameters and logarithm of the testlet variances, the same ones plotted during the analysis with 10,000 iterations. Figure 15 shows the corresponding running mean plots.

The five chains seem to mix together for all the parameters except the variance parameters. Figure 15 shows that there is still some visible difference between the five chains for $\sigma^2_{d(j)}$s. Whether this difference is significant or not remains to be seen.

The Gelman-Rubin convergence diagnostic results in small PSRFs (all less than 1.021), implying that the MCMC algorithm has converged. CSRFs give similar values as well. The values are low even when applying the measure to the last half of the 50,000 iterations. The value of the Brooks-Gelman MPSRF is 1.19. Figure 16 shows a plot of the Brooks-Gelman MPSRF (denoted R_p) along with the maximum PSRF (denoted R_{max}) for successively larger segments of the chains. The plot shows that the MPSRF has settled down close to 1.0 after 30,000 to 40,000 iterations, implying that the chains have mixed together. Therefore, it is safe to assume convergence and treat the five chains of 50,000 (with the first 20,000 to 25,000 iterations, when the MPSRF remains considerably higher than 1, being discarded as burn-in) as the posterior sample for any further statistical inference. The difference between the chains in Figure 15 is not significant. In fact, the same plot, redrawn using the last 30,000 iterations from the chains, shows little difference between the chains.

However, The Geweke's convergence diagnostic provides high Z-scores and hence very low p-values for about half of the parameters in all the chains. The Heidelberger and Welch diagnostic produces similar results, with 30 to 40 of the 80 monitored parameters failing the stationarity test or half-width test, clearly indicating that the chain has not converged yet. Significantly, all the testlet variance parameters fail both the tests, indicating that to estimate them accurately, the algorithm has to run longer.

The Raftery and Lewis measure, when run with quantile = 0.025, accuracy = 0.01, and probability = 0.95, gives values of required number of iterations that range from 10,000 to 117,000.

***Figure* 14: Time-series plots for four a_js, four b_js, seven c_js, and all five $\log\{\sigma^2_{d(j)}\}$s for five chains of 50,000 iterations each for the SCORIGHT example**

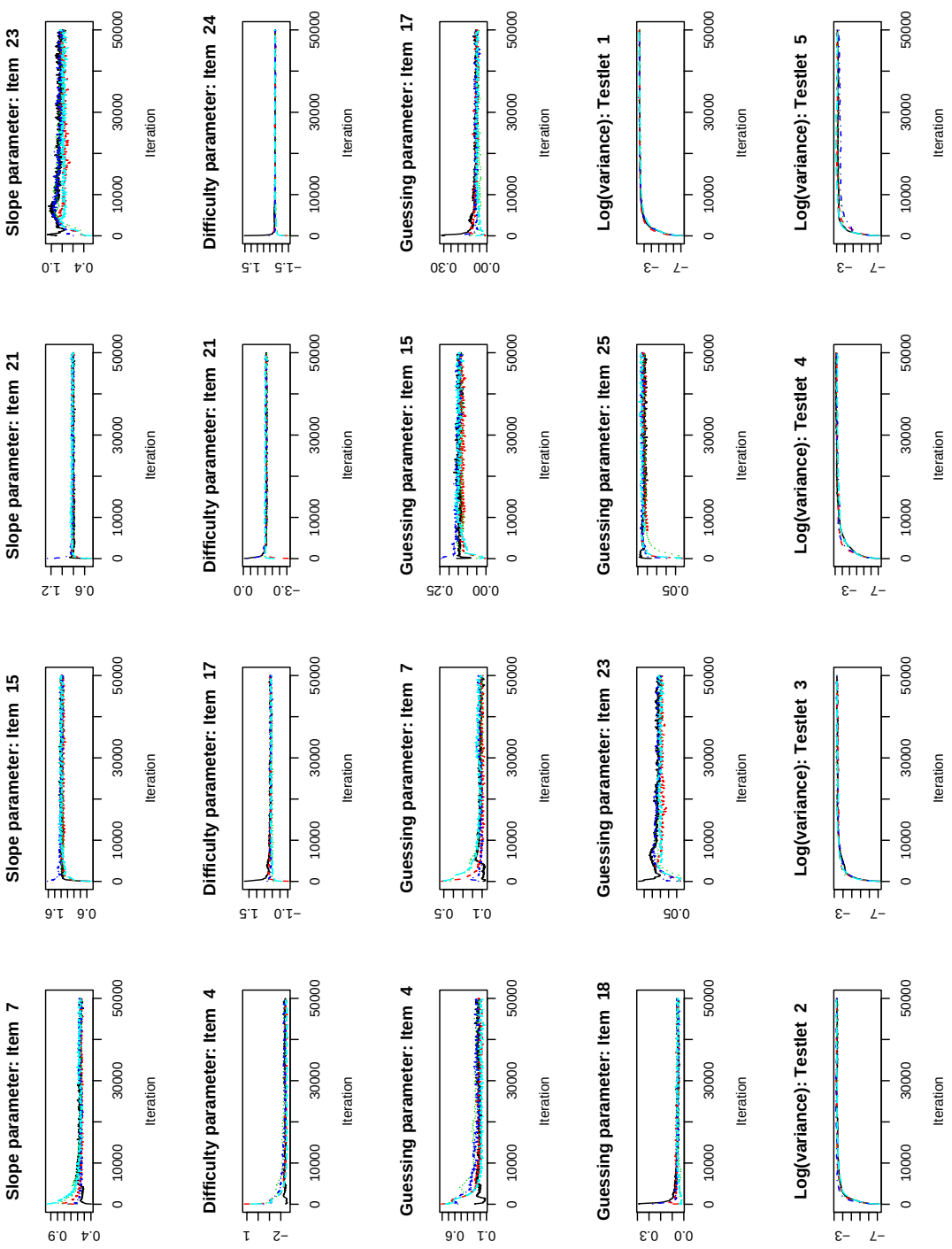

***Figure* 15: Running mean plots for four a_js, four b_js, seven c_js, and all five $\sigma^2_{d(j)}$s for five chains of 50,000 iterations each for the SCORIGHT example**

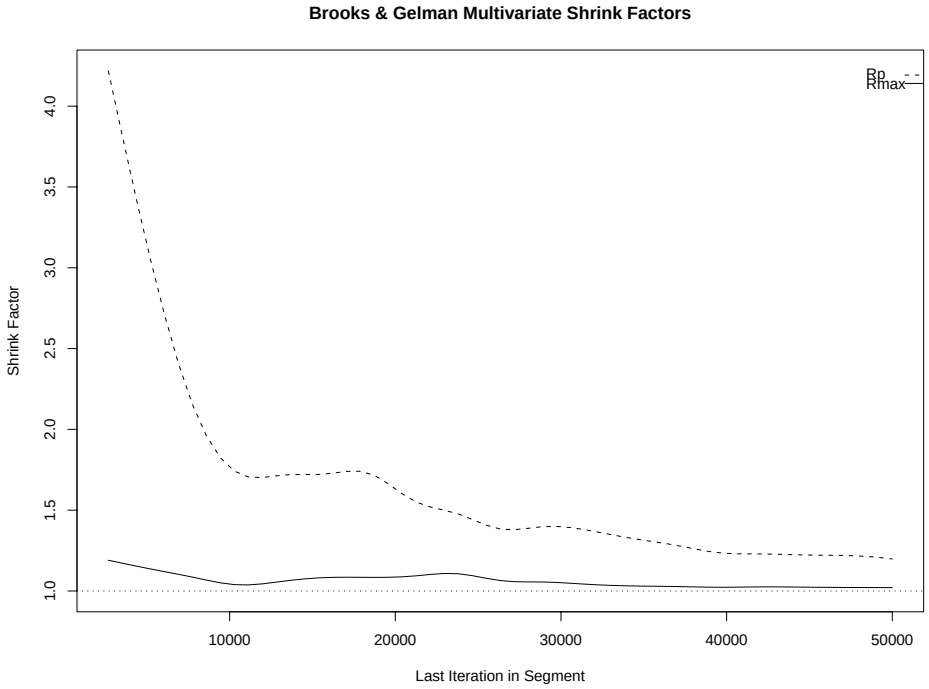

***Figure* 16: The plot of the Brooks-Gelman MPSRF for five chains of 50,000 iterations each for the SCORIGHT example**

These results indicate that if a researcher uses one chain only, even 50,000 iterations are not adequate to ensure convergence. We then run one chain of 100,000 iterations.

Results Using the Diagnostics for 100,000 Iterations

Figure 17 shows a time-series plot of all the variance parameters and four c_js whose time-series plots look most jagged. Geweke's convergence diagnostic, when applied to the last 90,000 iterations of this long chain, provides only four extreme p-values, indicating that there is not a significant difference between the first and last part of the chain after a burn-in of 10,000 iterations.

The Heidelberger and Welch diagnostic results in only 3 of the 80 monitored parameters failing the stationarity test or half-width test, indicating that the chain has converged.

The Raftery and Lewis measure, when run with quantile = 0.025, accuracy = 0.01,

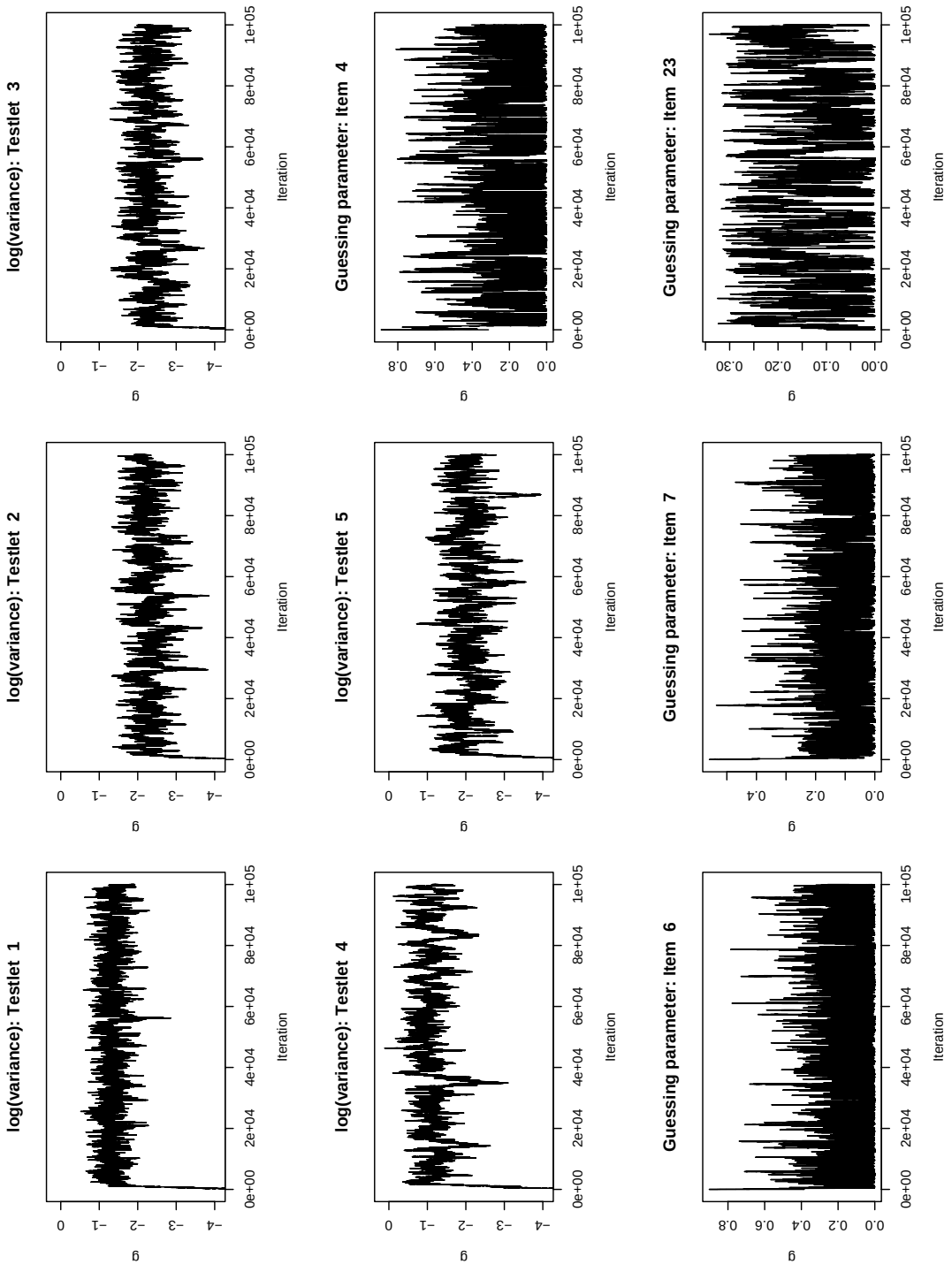

***Figure* 17: Time-series plots for all $\log\{\sigma^2_{d(j)}\}$s and four c_js for one chain of 100,000 iteration**

and probability = 0.95, gives values of required number of iterations that range from 12,000 to 107,000, with only one value above 100,000.

These results suggest that if using only one chain, the MCMC algorithm needs to run for at least 100,000 iterations to ensure the convergence of the algorithm. Note that the required number of iterations suggested by the diagnostics is way above the few thousand found in the example in the *User's Guide for SCORIGHT* (Wang, Bradlow, & Wainer, 2001).

6. Summary and Recommendations

Even though the MCMC algorithms are in wide use in psychometrics nowadays, the researchers in the field often ignore the important issue of convergence of the algorithms. This report first explains the concept of convergence and then proceeds to discuss a number of popular MCMC convergence diagnostics. Included among them are methods with different theoretical bases, testing for different aspects of convergence. Even though none of these methods provide a guarantee of convergence of the MCMC, they test necessary conditions of convergence and provide at least some check of the phenomenon.

This work also applies the diagnostics to two real data examples in psychometrics. The examples demonstrate how to apply a step-by-step approach to assess the convergence of the MCMC algorithms in real applications. First, the researcher should run a few iterations (depending on the complexity of the problem) of an MCMC algorithm. The next step is to apply the convergence diagnostics to the output of the program and make a decision as to whether the algorithm has converged. If the diagnostics indicate a lack of convergence, the investigator should try to determine the additional number of iterations from the results provided by the diagnostics and run the algorithm again. This process should be iterated until no lack of convergence is found.

Because none of the convergence diagnostics is guaranteed to work (and each check only a necessary condition of convergence), it is wise to apply as many of them, belonging to different types, as possible. The real data examples show that the different measures suggest different number of iterations required for convergence. This is not surprising, given

the different nature of the diagnostics. A safe option is to conclude convergence only when all of a variety of diagnostics indicate convergence.

About the question of how many parameters to monitor, the safe option is to monitor all the parameters of the model simultaneously. The Brooks-Rubin MPSRF has a distinct advantage over the other diagnostic tools in this respect, because the former monitors not only all the parameters simultaneously in one plot, but their interactions as well. However, monitoring all the paremeters may be computationally prohibitive. For example, on a SUN Blade 1000 workstation equipped with 950 MHz CPU and half gigabyte of RAM, obtaining a plot of the Brooks-Rubin MPSRF using BOA (Smith, 2001) was not possible for the SCORIGHT example with five chains of 50,000 iterations. The S-plus program running the BOA subroutine was terminated because it needed more dynamic memory than what was available. One way to get around this type of problem is to thin the posterior sample by retaining every k-th iteration, $k > 1$, and running the convergence diagnostics on the thinned output.

The diagnostics indicate for both the examples that the number of iterations required to ensure convergence of the MCMC algorithm is quite large. This large number is mainly an outcome of the high autocorrelation within the chains, caused by significant crosscorrelations among the parameters. The presence of high crosscorrelations (and hence high autocorrelations) is typical with the popular models in psychometrics. For example, Patz and Junker (1999b) show that the estimate of the difficulty parameter for any item is highly correlated with the estimate of the guessing parameter of that item in a 3PL (Lord, 1980) model. Hence, fitting models using MCMC in psychometrics will typically require a large number of iterations to ensure the convergence of the algorithm. For example, Scott and Ip (2002) use an MCMC algorithm with 100,000 iterations and discard the first 50,000 iterations as burn-in to analyze the NAEP reading assessment data. Reparameterization of the models (e.g., Tsutakawa, 1992) may result in faster convergence of the MCMC algorithms. More sophisticated MCMC algorithms, e.g., data augmentation (Albert, 1992; Beguin & Glas, 2001), the resampling and adaptive switching of the transition kernel (Gelfand & Sahu, 1994) and multichain annealing or "simulated tempering" (Geyer & Thompson, 1995), promise to reduce the time to convergence.

Finally, as a caution, a researcher should use additional methods for increasing understanding of the target distribution, because of the lack of a guaranteed diagnostic tool. If possible, approximate maximum likelihood estimates of the parameters of the model should be computed and compared to the estimates obtained by the MCMC algorithm. Gelman and Rubin (1992b) suggest that the investigator should look in general for the modes and create simple modal approximations of the distribution before running an MCMC algorithm. Comparison of the output from the MCMC algorithm to that from the modal approximations is likely to reveal limitations of both approaches.

References

Albert, J. (1992). Bayesian estimation of normal ogive item response functions using Gibbs sampling. *Journal of Educational Statistics, 17,* 251-269.

Beguin, A. A., & Glas, C. A. W. (2001). MCMC estimation and some model-fit analysis of multidimensional IRT models. *Psychometrika, 66*(4),541-561.

Best, N. G., Cowles, M. K., & Vines, K. (1995). CODA: Convergence diagnosis and output analysis software for Gibbs sampling output (Version 0.3)[Computer software]. Cambridge, UK: University of Cambridge, MRC Biostatistics Unit.

Bradlow, E. T., Wainer, H., & Wang, X. (1999). A Bayesian random effects model for testlets. *Psychometrika, 64,* 153-168.

Brooks, S (1998). MCMC convergence diagnosis via multivariate bounds on log-concave densities. *The Annals of Statistics, 26*(1), 398-433.

Brooks, S., & Roberts, G. O. (1998). Convergence assessments of Markov chain Monte Carlo algorithms. *Statistics and Computing, 8,* 319-335.

Brooks, S., & Gelman, A. (1998). General methods for monitoring convergence of iterative simulations. *Journal of Computational and Graphical Statistics, 7*(4), 434-455.

Carlin, B. P., & Louis, T. A. (1996). *Bayes and empirical Bayes methods for data analysis.* London: Chapman and Hall.

Cowles, M. K., & Carlin, B. P. (1996). Markov chain Monte Carlo convergence diagnostics: A comparative review. *Journal of the American Statistical Association, 91,* 883-904.

Cox, D. R., & Miller, H. D. (1965). *The Theory of Stochastic Processes.* London: Chapman and Hall.

Fan, Y., Brooks, S., & Gelman, A. (2002). *Convergence assessment of Monte Carlo simulations via the score statistics* (Technical Report). Kent, UK: University of Kent, Institute of Mathematics and Statistics.

Garren, S., & Smith, R. L. (1995). *Estimating the second largest eigenvalue of a Markov transition matrix* (Technical Report #95-13). Cambridge, UK: University of Cambridge, Statistical Laboratory.

Gelfand, A. E., & Sahu, S. K. (1994). On Markov chain Monte Carlo acceleration. *Journal of Computational and Graphical Statistics, 3,* 261-276.

Gelman, A., Carlin, J. B., Stern, H. S., & Rubin, D. B. (1995). *Bayesian data analysis.* New York: Chapman & Hall.

Gelman, A. (1995). Inference and monitoring convergence. In W. R. Gilks, S. Richardson, & D. J. Spiegelhalter (Eds.), *Markov chain Monte Carlo in practice* (pp. 131-143). London: Chapman and Hall.

Gelman, A., Roberts, G., & Gilks, W. (1995). Efficient Metropolis jumping rules. In J. M. Bernardo, J. O. Berger, A. P. Dawid, & A. F. M. Smith (Eds.), *Bayesian Statistics 5.* New York: Oxford University Press.

Gelman, A., & Rubin, D. B. (1992a). A single series from the Gibbs sampler provides a false sense of security. In J. M. Bernardo, J. O. Berger, A. P. Dawid, & A. F. M. Smith (Eds.), *Bayesian Statistics 4* (pp. 625-631). New York: Oxford University Press.

Gelman, A., & Rubin, D. B. (1992b). Inference from iterative simulation using multiple sequences. *Statistical Science, 45*, 457-511.

Geman, S., & Geman, D. (1984). Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 6,* 721-741.

Geweke, J. (1992). Evaluating the accuracy of sampling-based approaches to calculating posterior moments. In J. M. Bernardo, J. O. Berger, A. P. Dawid, & A. F. M. Smith (Eds.), *Bayesian Statistics 4* (pp. 169-193). New York: Oxford University Press.

Geyer, C. J. (1992). Practical Markov chain Monte Carlo. *Statistical Science, 7,* 473-483.

Geyer, C. J., & Thompson, E. A. (1995). Annealing Markov chain Monte Carlo with applications to ancestral inference. *Journal of the American Statistical Association, 90,* 909-920.

Gilks, W. R., Richardson, S., & Spiegelhalter, D. J. (1996). *Markov chain Monte Carlo in practice.* London: Chapman and Hall.

Hartz, S., Roussos, L., & Stout, W. (2002). *Prime assessment skills diagnosis, theory and practice.* Princeton, NJ: Educational Testing Service.

Hastings, W. K. (1970). Monte carlo sampling methods using Markov chains and their applications. *Biometrika, 57,* 97-109.

Heidelberger, P., & Welch, P. D. (1983). Simulation run length control in the presence of an initial transient. *Operations Research, 31,* 1109-1144.

Johnson, M. S. (2002). *A Bayesian hierarchical model for multidimensional performance assessments.* Paper presented at the annual meeting of the National Council on Measurement in Education, New Orleans, LA.

Johnson, V. E. (1996). Studying convergence of Markov chain Monte Carlo algorithms using coupled sample paths. *Journal of the American Statistical Association, 91,* 154-166.

Liu, C., Liu, J., & Rubin, D. B. (1992). A variational control variable for assessing the convergence of the Gibbs sampler. *Proceedings of the American Statistical Association, Statistical Computing Section,* 74-78.

Lord, F. M. (1980). *Applications of item response theory to practical testing problems.* Hillsdale, NJ: Lawrence Erlbaum Associates.

Mengersen, K. L., Robert, C. P., & Guihenneuc-Jouyaux, C. (1999). MCMC convergence diagnostics: A reviewww (with discussion). In J. M. Bernardo, J. O. Berger, A. P. Dawid, & A. F. M. Smith (Eds.), *Bayesian Statistics 6* (pp. 415-440). Oxford: Oxford University Press.

Metropolis, N., & Ulam, S. (1949). The Monte Carlo method. *Journal of the American Statistical Association, 44,* 335-341.

Muraki, E., & Bock, R. D. (1991). *PARSCALE: parametric scaling of rating data.* Chicago: Scientific Software International.

Mykland, P., Tierney, L., & Yu, B. (1995). Regeneration in Markov chain samplers. *Journal of the American Statistical Association, 90,* 233-241.

Patz, R., & Junker, B. (1999a). A straightforward approach to Markov chain Monte Carlo methods for item response models. *Journal of Educational and Behavioral Statistics, 24,* 146-178.

Patz, R., & Junker, B. (1999b). Applications and extensions of MCMC in IRT: Multiple item types, missing data, and rated responses. *Journal of Educational and Behavioral Statistics, 24,* 342-366.

Patz, R., Junker, B., & Johnson, M. S. (in press). The hierarchical rater model for rated test items and its applications to large-scale educational assessment data. *Journal of Educational and Behavioral Statistics.*

Polson, N. G. (1996). Convergence of Markov chain Monte Carlo algorithms. In J. M. Bernardo, J. O. Berger, A. P. Dawid, & A. F. M. Smith (Eds.), *Bayesian Statistics 5* (pp. 297-322). New York: Oxford University Press.

Propp, J. G., & Wilson, D. B. (1996). Exact sampling with coupled Markov chains and applications to statistical mechanics. *Random Structures and Algorithms, 9,* 223-252.

Raftery, A. L., & Lewis, S. (1992). How many iterations in the Gibbs sampler? In J. M. Bernardo, J. O. Berger, A. P. Dawid, & A. F. M. Smith (Eds.), *Bayesian Statistics 4* (pp. 763-773). New York: Oxford University Press.

Raftery, A. L., & Lewis, S. (1996). Implementing MCMC. In W. R. Gilks, S. Richardson, & D. J. Spiegelhalter (Eds.), *Markov chain Monte Carlo in practice* (pp. 115-130). London: Chapman and Hall.

Ritter, C., & Tanner, M. A. (1992). Facilitating the Gibbs sampler: The Gibbs stopper and the Griddy-Gibbs sampler. *Journal of the American Statistical Association, 87,* 861-868.

Robert, C. P. (1996). Convergence assessments for Markov chain Monte Carlo methods. *Statistical Science, 10,* 231-253.

Robert, C. P. (Ed.). (1998). *Discretization and MCMC convergence assessment.* New York: Wiley.

Roberts, G. O. (1992). Convergence diagnostics of the Gibbs sampler. In J. M. Bernardo, J. O. Berger, A. P. Dawid, & A. F. M. Smith (Eds.), *Bayesian Statistics 4* (pp. 775-782). New York: Oxford University Press.

Rosenthal, J. S. (1995). Minorization conditions and convergence rates for Markov chain Monte Carlo. *Journal of the American Statistical Association, 90,* 558-566.

Sandene, B., Bennett, R. E., Braswell, J., & Oranje, A. (in press). *The Math Online Study: final report.* Washington, DC: National Center for Educational Statistics, Office of Educational Research and Improvement, US Department of Education.

Schruben, L. W. (1982). Detecting initialization bias in simulation output. *Operations Research, 30,* 569-590.

Schruben, L. W., Singh, H., & Tierney, L. (1983). Optimal tests for initialization bias in simulation output. *Operations Research, 31,* 1167-1178.

Scott, S. L., & Ip, E. H. (2002). Empirical Bayes and item-clustering effects in a latent variable hierarchical model: A case study from the National Assessment of Educational Progress. *Journal of the American Statistical Association, 97,* 409-419.

Sinharay, S., Johnson, M. S., & Williamson, D. M. (2003). *An application of a Bayesian hierarchical model for item family calibration* (ETS RR-03-04). Princeton, NJ: Educational Testing Service.

Smith, B. (2001). Bayesian output analysis program (BOA) (Version 1.0.0)[Computer software]. Iowa City, IA: University of Iowa, College of Public Health.

Spiegelhalter, D. J., Thomas, A., Best, N. G., & Gilks, W. R. (1995). BUGS: Bayesian inference using Gibbs sampling (Version 0.50) [Computer software]. Cambridge, UK: University of Cambridge, MRC Biostatistics Unit.

Tierney, L. (1994). Markov chains for exploring posterior distributions. *Annals of Statistics, 22,* 1701-1762.

Tsutakawa, R. K. (1992). Prior distributions for item response curves. *British Journal of Mathematical and Statistical Psychology, 45,* 51-74.

Venables, W. N., & Ripley, B. D. (2000). *S Programming.* New York: Springer-Verlag.

von Mises, R. (1931). *Wahrscheinlichkeitsrechnung.* Deuticke, Wien.

Wainer, H., & Kiely, G. (1987). Item clusters and computerized adaptive testing; A case for testlets. *Journal of Educational Measurement, 24,* 185-202.

Wang, X., Bradlow, E. T., & Wainer, H. (2001). *User's guide for SCORIGHT (Version 1.2): A computer program for scoring tests built of testlets.* (ETS RR-01-06). Princeton, NJ: Educational Testing Service.

Wang, X., Bradlow, E. T., & Wainer, H. (2000). A general Bayesian model for testlets: Theory and applications. *Applied Psychological Measurement, 26* (1), 109-128.

Yu, B., & Mykland, P. (1998). Looking at Markov samplers through CUSUM path plots: A simple diagnostic idea. *Statistics and Computing, 8,* 275-286.