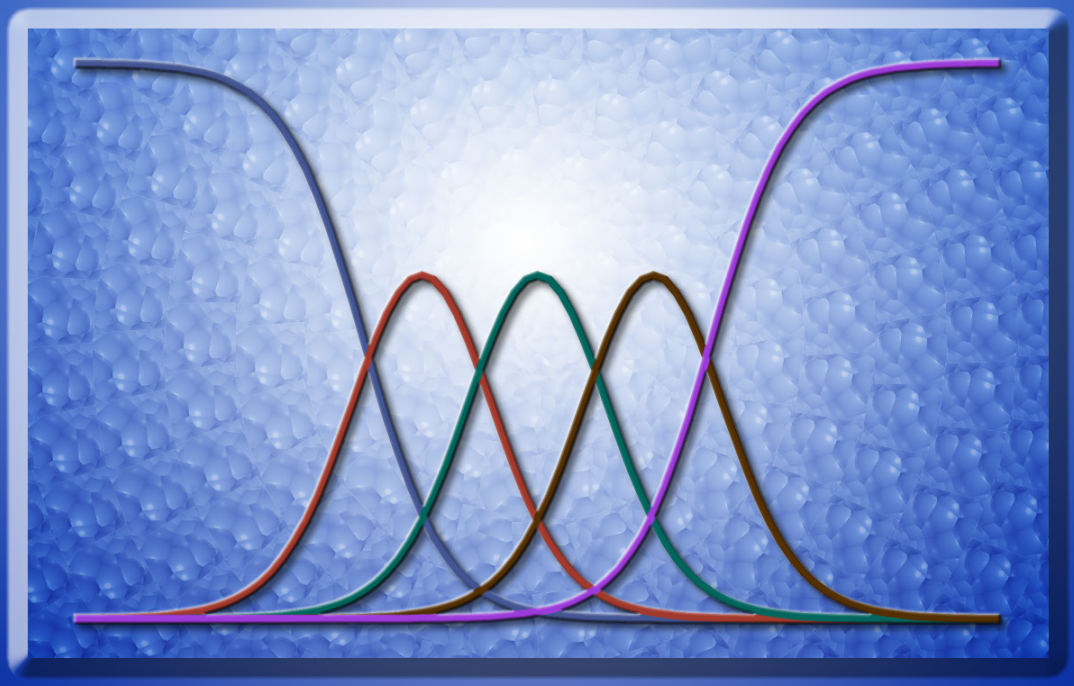
Bayesian Analysis of Item Response Theory Models Using SAS®



Clement A. Stone and Xiaowen Zhu

Contents

Chapter 5: Bayesian Estimation of Unidimensional IRT Models for Polytomously Scored Items

Introduction

The item response theory (IRT) models discussed in Chapter 4 apply to dichotomously scored items. When items are polytomously scored, such as constructed-response items in educational assessments and rating or Likert-type response items in health, social, and behavioural measures, polytomous IRT models are required. These models similarly represent the relationship between a person's level on the latent trait being measured and the probability of a particular response. But now there are more than two response categories. This chapter focuses on polytomous IRT models where items are also assumed to measure one underlying latent trait.

There are various unidimensional polytomous IRT models. The most commonly used models include (1) the graded response (GR) model (Samejima, 1969; 1996); (2) a modified GR model (Muraki, 1990), also called Muraki's rating scale (RS) model; (3) the partial credit (PC) model (Masters, 1982); (4) the generalized partial credit (GPC) model (Muraki, 1992); and (5) the nominal response (NR) model (Bock, 1972).

This chapter illustrates how to estimate a variety of IRT models for polytomous responses using PROC MCMC. Although the models are briefly described in each section, the reader is referred to Chapter 1 for more detail. The data used for the illustrations are a subset of cases and items from the DASH survey data (Stone & Irrgang, 2004) and comprise responses of 1000 examinees to 10 five-category items. The DASH survey was designed to measure physical functioning with items asking about the degree of difficulty, from 1 ("*No Difficulty*") to 5 ("*Unable*"), in performing different tasks such as opening a jar, writing, making the bed, and others. One latent trait was found to underlie the responses to this survey. In order for you to execute the code provided in this chapter, it is assumed that that the SAS data set of item responses (DASH_DATA) has been read into the SAS Work library. (See the SAS Press authors' web page for this SAS data set.)

The Graded Response Model

Samejima's (1969; 1996) graded response (GR) model is used to model the cumulative probability of responding in a particular response category or higher for graded or ordered response scales. Specifically, for an item j with m_j response categories (1, 2,… m_j), the cumulative probability that a person receives a category score k (k = 2, … m_j) or higher ($P^*_{j(k-1)}(\theta)$) is modelled by the logistic deviate $z_{j(k-1)} = Da_j(\theta - b_{j(k-1)})$, where D is the scaling constant (1.7 or 1), θ references the trait being measured for a person, a_j is the discrimination (slope) parameter for item j, and $b_{j(k-1)}$ is a between category threshold parameter (k–1 and k) for item j. Given the cumulative response probabilities, the probability of receiving a category score k on item j is defined as the difference between the cumulative probabilities for two adjacent categories, with two constraints $P^*_{j1}(\theta) = 1$ and $P^*_{j(m_j+1)}(\theta) = 0$. Note that within each item, the between category threshold parameters b_{jk} are necessarily ordered in the GR model. Also, the response function is for an individual person, but the subscript for the person is excluded for convenience in this equation and subsequent equations.

Program Template for the GR Model

A preliminary template of PROC MCMC code for estimating a GR model using the DASH data is provided in Program 5.1. For each 5-category item, one slope parameter (a_j) and four threshold parameters (b_{j1}, b_{j2} , b_{j3} , b_{j4}) are estimated. Therefore, for the DASH data and GR model, a total of 10 slope parameters (a_1~a_{10}) and 40 threshold parameters ($b_{1,1}$~$b_{1,4}$, ..., $b_{10,1}$~$b_{10,4}$) are estimated. As for the IRT model specifications for dichotomously scored items in Chapter 4, the model is parameterized in a slope intercept form where the logisitic $z_{jk} = a_j\theta - d_{jk}$, D = 1 and $b_{jk} = d_{jk}/a_j$. These b_{jk} parameters are saved rather than the d_{jk} parameters using the MONITOR option for the PROC MCMC command because the b_{jk} parameters are more directly interpretable.

The code in the template also presents the set of commands necessary to define the likelihood model for IRT models given polytomously scored items. As discussed in Chapter 3, rather than program the response probability model and use a MODEL statement to specify the conditional distribution of the data given the parameters, you can program the log-likelihood function explicitly using SAS programming statements and use the GENERAL option with the MODEL statement to specify the log of the joint distribution for the observed item responses. If you have SAS/STAT 13.1 or earlier, this approach is necessary for IRT models with polytomously scored items because there is no general categorical distribution function available in the MODEL statement beyond the binary distribution specification. As will be shown in a later section, a general categorical distribution function is available, starting in SAS/STAT 13.2, that can be used with the MODEL statement for polytomously scored items. However, this function is less efficient than the code presented in Program 5.1.

Notably, the program for estimating a GR model assumes that the response categories for each item in the data are ordered sequentially from 1 to m_j, where m_j equals the number of response categories for each item j. If the item data do not conform to this assumption, the item responses will need to be transformed prior to your using any of the programs in this chapter.

Program 5.1: Preliminary PROC MCMC Template for Estimating the GR Model for the DASH data

```
proc mcmc data=dash_data outpost=dash_post seed=23 nbi=5000 nmc=20000
nthreads=8 monitor=(a b1_ b2_ b3_ b4_) diagnostics=all plots=(trace
autocorr);

   array ix[10]; array a[10]; ❶
   array b1_[10]; array b2_[10]; array b3_[10]; array b4_[10];
   array d1_[10]; array d2_[10]; array d3_[10]; array d4_[10];
   array p_star[10,4]; array p[10,5];

/* Declare model parameters and initial values */ ❷
   parms a1 1  d1_1 -1  d2_1 -0.5  d3_1 0  d4_1 1;
   parms a2 1  d1_2 -1  d2_2 -0.5  d3_2 0  d4_2 1;
   parms a3 1  d1_3 -1  d2_3 -0.5  d3_3 0  d4_3 1;
   parms a4 1  d1_4 -1  d2_4 -0.5  d3_4 0  d4_4 1;
   parms a5 1  d1_5 -1  d2_5 -0.5  d3_5 0  d4_5 1;
   parms a6 1  d1_6 -1  d2_6 -0.5  d3_6 0  d4_6 1;
   parms a7 1  d1_7 -1  d2_7 -0.5  d3_7 0  d4_7 1;
   parms a8 1  d1_8 -1  d2_8 -0.5  d3_8 0  d4_8 1;
   parms a9 1  d1_9 -1  d2_9 -0.5  d3_9 0  d4_9 1;
   parms a10 1 d1_10 -1  d2_10 -0.5  d3_10 0  d4_10 1;

 /* Prior distributions for slope and intercept parameters */
   prior a: ~ lognormal(0, var=16); ❸

   prior d1_1 ~ normal(<distribution specifications>); ❹
   prior d2_1 ~ normal(<distribution specifications>);
   prior d3_1 ~ normal(<distribution specifications>);
   prior d4_1 ~ normal(<distribution specifications>);

   [repeat set of prior commands for other item threshold parameter sets]


 /* Prior distribution for person ability parameters */
   random theta ~ normal(0, var=1)  subject=_obs_;

 /*Specify the log-likelihood function based on the GR model */
   llike=0;
   do j=1 to 10; ❺
     p_star[j,1]=logistic(a[j]*theta-d1_[j]);
     p_star[j,2]=logistic(a[j]*theta-d2_[j]);
     p_star[j,3]=logistic(a[j]*theta-d3_[j]);
     p_star[j,4]=logistic(a[j]*theta-d4_[j]);

     b1_[j]=d1_[j]/a[j];
     b2_[j]=d2_[j]/a[j];
     b3_[j]=d3_[j]/a[j];
     b4_[j]=d4_[j]/a[j];

     p[j,1]=1-p_star[j,1] ;
     do k=2 to 4;
        p[j,k]=p_star[j,(k-1)] - p_star[j,k];
     end;
     p[j,5]=p_star[j,4] ;

     llike=llike+log(p[j,ix[j]]);
  end;

  model general(llike); ❻

run;
```

❶ Numerous arrays are designed to facilitate reference to different parameters and variables in the model. For example, ARRAY *b1_* [10] represents the parameters *b1_*1 to *b1_*10 or the set of first threshold parameters for the 10 items. Similarly, *b2_* [10], *b3_* [10], and *b4_* [10] represent the second, third, and fourth sets of threshold parameters for the 10 items. Thus, the item parameters for Item 1 are defined by *a*1, *b1_*1, *b2_*1, *b3_*1, and *b4_*1 in the program. The arrays P_STAR and P define two dimensional arrays with rows indexing the items and columns indexing cumulative response probabilities (P_STAR) and category response probabilities (P).

❷ In order to reduce autocorrelations in sampled parameter values, multiple PARMS statements are used to block sets of parameters for each item (see Chapter 3, the section "Parameter Blocking"). In this program template, initial values of 1 are given for slope parameters (a_j), and the initial values of −1, −0.5, 0, and 1 are given for the first, second, third, and fourth intercepts (d_{j1}, d_{j2} , d_{j3} , d_{j4}). It should be noted that the initial values of intercepts for this example should also be ordered to reflect the order constraint for the thresholds in the GR model: $b_{j1} < b_{j2} < b_{j3} < b_{j4}$.

❸ The PRIOR statements provide the prior distributions for parameters specified in PARMS statements. As for the models in Chapter 4, the prior distributions for the slope parameters (a_j) for each DASH item are assumed to follow a lognormal distribution with the mean of 0 and the variance of 16. In contrast to the priors in Chapter 4, no lower and upper bounds were necessary to define a reasonable set of bounds on sampled values. Since all item slopes follow the same distribution, one PRIOR statement is used with A: indicating the 10 slopes.

❹ The prior distribution specifications for the threshold (b_{jk}), or intercept parameters (d_{jk}) which are functions of the threshold parameters, must conform to the order constraint in the GR model. That is, $b_{j1} < b_{j2} < \ldots < b_{jk}$ or $d_{j1} < d_{j2} < \ldots < d_{jk}$. Though the priors for these parameters also typically follow normal distributions, one PRIOR statement for all intercepts PRIOR D: ~ NORMAL (<distribution specifications>) cannot be used as for the dichotomous IRT models (see Chapter 4). Instead, for each item, you have to specify four threshold or intercept parameters in four separate PRIOR statements to reflect the order constraint. Thus, there are 40 prior statements for the threshold or intercept parameters in total. Three different options for specifying the prior distributions are discussed in the next section.

❺ The DO loop is used to compute cumulative response category probabilities for each item (P_STAR[J,1] − P_STAR[J,4]), transform the intercept parameters (d_{jk}) to compute the threshold parameters (b_{jk}), and use differences between cumulative response probabilities to compute the probability of responding in each of the *j* response categories (P[J,1] − P[J,5]). Note that the response probability for response Category 1 (P[J,1]) is the difference between the cumulative probabilities for response Category 1 (1) and for response Category 2 (P_STAR[J,1]). The response probability in response Category 2 (P[J,2]) is the difference between the cumulative probabilities for response Category 2 (P_STAR [J, 1]) and for response Category 3 (P_STAR [J, 2]). The probability associated with the last response category (P[J,5]) is equal to cumulative probability (P_STAR [J, 4]). Lastly, if threshold parameters are included in the response probability model, rather than intercept parameters, the code calculating the threshold parameters as a function of the slope and intercept parameters would be deleted.

❻ The MODEL statement specifies the log likelihood distribution. There is no need to specify the dependent variable in the MODEL statement since the log likelihood function includes the dependent variable (IX[J] array).

Options for Specifying Prior Distributions for the Threshold and Intercept Parameters

The literature contains different approaches to specifying the prior distributions given the order constraint for the threshold or intercept parameters for each item. Three different methods are discussed that reflect an ordered constraint between four parameters, analogous to the four threshold parameters for each item in the GR model. The code below is presented without reference to the item subscript *j* to simplify the presentation.

Method 1(L)—Use the sampled values for b_k or d_k as lower bounds on the distributions for b_{k+1} or d_{k+1}:

```
parms d1 -1 d2 -.5 d3 .5 d4 1;
prior d1 ~ normal(0, var=16);
prior d2 ~ normal(0, var=16, lower= d1);
prior d3 ~ normal(0, var=16, lower= d2);
prior d4 ~ normal(0, var=16, lower= d3);
```

Method 2 (L/U)—Use sampled values for b_k or d_k as both lower and/or upper bounds on the distributions for b_{k+1} or d_{k+1}:

```
parms d1 -1 d2 -.5 d3 .5 d4 1;
prior d1 ~ normal(0, var=16, upper= d2);
prior d2 ~ normal(0, var=16, lower= d1, upper= d3);
prior d3 ~ normal(0, var=16, lower= d2, upper= d4);
prior d4 ~ normal(0, var=16, lower= d3);
```

Method 3 (Sort)—Sort the set of thresholds and use the sorted parameter values in the response probability model (Curtis, 2010):

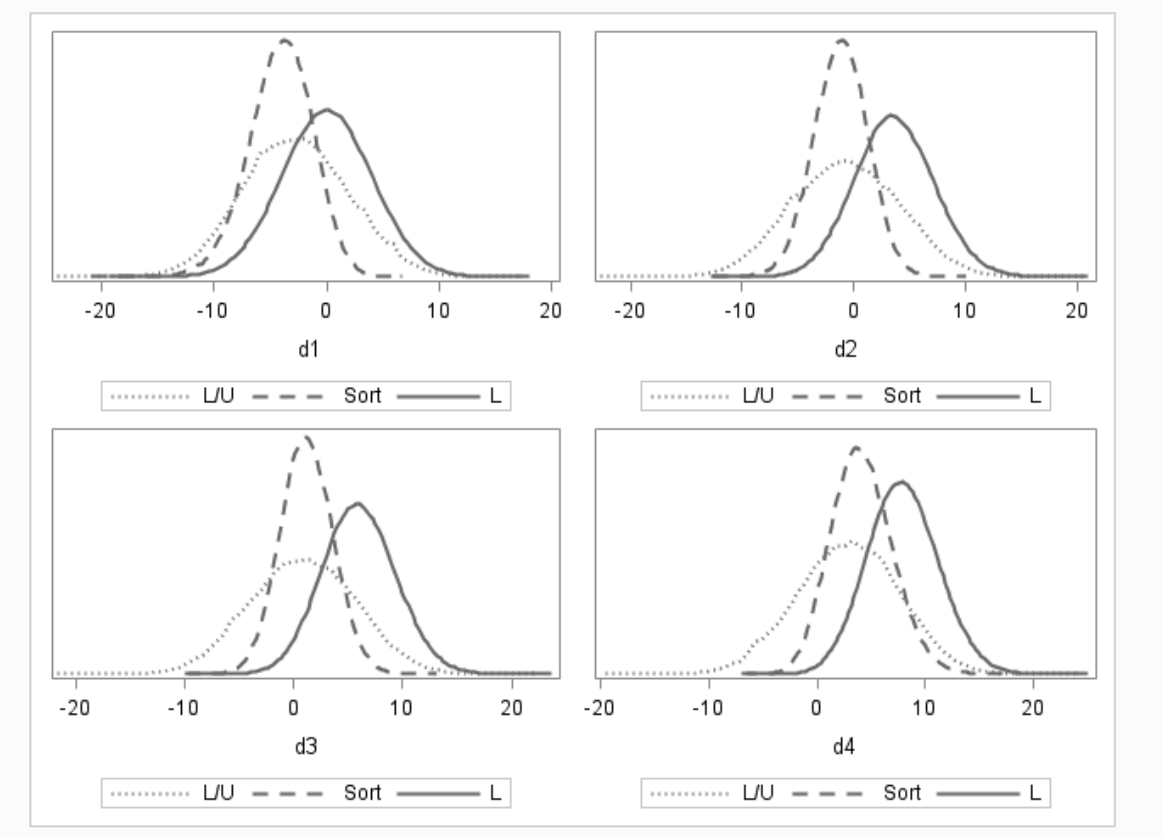
```
parms q1 -1 q2 -.5 q3 .5 q4 1;
prior q: ~ n(0, sd=4);
d1=q1;
d2=q2;
d3=q3;
d4=q4;
call sortn(d1, d2, d3, d4);
```

The use of Method 3 in PROC MCMC is complicated by the fact that no transformations can be performed directly on model parameters specified in PARMS statements. Thus, a copy of the variables in the PARMS statements may be obtained, and these copied variables may be sorted. These variables are then used in the response probability model for the GR model to satisfy the order constraint. It should also be noted that a direct sampling method cannot be used with the code as presented above for Method 2. Rather, the above presentation is used to more easily describe the parameter constraints on the normal distributions. Instead, a joint density function must be specified. This is discussed in the next section.

Output 5.1 presents the prior densities for the four threshold parameters ($d1$–$d4$) from the above three methods where Method 1 is labeled *L* for lower bounds, Method 2 is labeled *L/U* for lower or upper bounds, and Method 3 is labeled *Sort*. These graphs can be produced by modifying the code presented in Chapter 3 (Program 3.1), stacking the data sets of sampled values from the distributions, and overlaying graphs using DENSITYPLOT graphics statements.

As you can see, all three prior densities are different despite the same mean and variance specifications for the normal distributions. While Methods 2 (*L/U*) and 3 (*Sort*) have similar locations, the spreads in the densities are quite different. In contrast, you can see that the locations of the densities from Method 1 (*L*) are very different from the locations of the densities for Methods 2 (*L/U*) and 3 (*Sort*). Given the differences between the prior densities for the three methods, it may be useful to evaluate the use of all three methods with the GR model. Use of all three prior distributions and the impact on the posterior densities for the threshold parameters is explored further in a later section. Finally, the observation that Methods 1 and 2 are different runs counter to the claim by Curtis (2010) that these approaches are essentially equivalent.

Output 5.1: Densities of the Four Parameters for the Different Prior Specifications



Estimation of the GR Model by Using Method 1

Program 5.2 provides code for estimating the GR model. The code in the program is based on the Method 1 prior specifications (lower bound specifications) because this method is commonly used with the GR model. In these programs, changes were made to the template to make the program more concise. First, two-dimensional arrays, B[4, 10] and D[4, 10], are used to represent the 40 threshold and 40 intercept parameters across the 10 items. Second, a joint prior distribution for the intercept parameters is specified, and all intercept parameters are sampled from this distribution. This is described in more detail below. Finally, a BEGINNODATA–ENDNODATA block of statements are used. Any statements included within the block are executed only at the first and the last observation of the data set. Thus a BEGINNODATA–ENDNODATA block of statements reduces unnecessary observational-level computations, and is appropriate for computations that are identical across observations, such as transformation of parameters.

Although not presented, Programs 5.3 and 5.4 contain the code for estimating the GR model using the other two methods and are included on the SAS Press authors' web page for this book. As discussed above, given the differences observed for the different methods of prior specifications, a comparison of results based on the different methods is presented in a later section.

Program 5.2: PROC MCMC Code for Estimating the GR Model (Method 1 Prior Specification)

```
proc mcmc data=dash_data outpost=dash_postGR seed=23 nbi=5000 nmc=20000
nthreads=8 monitor=(a b) diagnostics=all plots=(trace density);

   array ix[10];  array a[10];
   array b[4,10] b1_1-b1_10  b2_1-b2_10 b3_1-b3_10 b4_1-b4_10;
   array d[4,10] d1_1-d1_10  d2_1-d2_10 d3_1-d3_10 d4_1-d4_10;
   array p_star[10,4]; array p[10,5];

   beginnodata;
   lprior=0;
   do j=1 to 10;
      lprior = lprior + lpdfnorm(d[1,j],0,5); ❶
      lprior = lprior + lpdfnorm(d[2,j],0,5,d[1,j]);
      lprior = lprior + lpdfnorm(d[3,j],0,5,d[2,j]);
      lprior = lprior + lpdfnorm(d[3,j],0,5,d[3,j]);
   end;

   do j=1 to 10;
     do k=1 to 4;
        b[k,j] = d[k,j]/a[j]; ❷
     end;
   end;
   endnodata;

   parms a1 1  d1_1 -1  d2_1 -0.5  d3_1 0.5  d4_1 1;
   parms a2 1  d1_2 -1  d2_2 -0.5  d3_2 0.5  d4_2 1;
   parms a3 1  d1_3 -1  d2_3 -0.5  d3_3 0.5  d4_3 1;
   parms a4 1  d1_4 -1  d2_4 -0.5  d3_4 0.5  d4_4 1;
   parms a5 1  d1_5 -1  d2_5 -0.5  d3_5 0.5  d4_5 1;
   parms a6 1  d1_6 -1  d2_6 -0.5  d3_6 0.5  d4_6 1;
   parms a7 1  d1_7 -1  d2_7 -0.5  d3_7 0.5  d4_7 1;
   parms a8 1  d1_8 -1  d2_8 -0.5  d3_8 0.5  d4_8 1;
   parms a9 1  d1_9 -1  d2_9 -0.5  d3_9 0.5  d4_9 1;
   parms a10 1  d1_10 -1  d2_10 -0.5  d3_10 0.5  d4_10 1;

   prior a: ~lognormal(0, var=25); ❸
   prior d: ~general(lprior); ❹
   random theta ~ normal(0, var=1) subject=_obs_;

   llike=0;
   do j=1 to 10;
     do k=1 to 4;
        p_star[j,k]=logistic(a[j]*theta-d[k,j]); ❺
     end;
     p[j,1]=1-p_star[j,1];
     do k=2 to 4;
        p[j,k]=p_star[j,(k-1)]-p_star[j,k];
     end;
     p[j,5]=p_star[j,4];
     llike = llike + log(p[j,ix[j]]);
   end;
   model general(llike);
run;
```

❶ The set of prior specifications for threshold or intercept parameters for each item may be more concisely specified with use of a joint density function. In PROC MCMC a joint density function must be specified in the log scale. Here, the LPDFNORM is a log-density function for the normal distribution. PROC MCMC has a number of internally defined log-density functions for univariate and multivariate distributions. These functions have the basic form of LPDF*dist*(<*variable*>, <*distribution parameters*>), where '*dist*' is the name of the distribution, the variable argument references the random variable, and distribution arguments are used to specify distribution parameters. For a normal

distribution, the form is LPDFNORM (*variable*, *mu*, *SD*, *lower bounds*, *upper bounds*). The specification LPDFNORM(D[2,J], 0, 5, D[1,J]) in the DO loop indicates that the second intercept for each item follows a normal distribution with a mean of 0, standard deviation of 5, and a lower bound equal to the sampled value for first intercept parameter. The statement LPRIOR=LPRIOR + LPDFNORM(<*distribution specification*>) computes the sum of the log prior density values across all intercept parameters. Therefore, *lprior* defines the log of the joint prior distribution for the intercept parameters across the 10 items. Since these computations are the same across observations, the statements are embedded within a BEGINNODATA–ENDNODATA block. The equivalence between separate and joint prior specifications is illustrated in the next section.

❷ The transformation function of B[J, K]=D[J, K] / A[J] is included in the BEGINNODATA and ENDNODATA block also because the computations are the same across observations. Results are the same, assuming the same SEED value, whether the transformations are included in a BEGINNODATA and ENDNODATA block or as specified in Program 5.1.

❸ No bounds are used to restrict the parameter space for estimating the a_j parameters or the d_{jk} parameters other than bounds required to satisfy the order constraint. As you will see in the posterior summary output and trace plots, the sampled values did not include unreasonable values.

❹ To declare the prior distributions for the intercept parameters (d_{jk}), one PRIOR statement is required: PRIOR D: ~ GENERAL (LPRIOR). The GENERAL function indicates that the intercepts are sampled from a joint prior distribution with the joint distribution *lprior*. Notably, any new distributions used with the GENERAL function, either in the PRIOR statement or in the MODEL statement, have to be specified on the logarithm scale.

❺ The use of two-dimensional arrays simplifies the programming statements used to compute the cumulative probabilities. As shown in the program, one statement: P_STAR [J, K] = LOGISTIC (A[J]*THETA − D[K, J]) is used for computing all of the required cumulative probabilities.

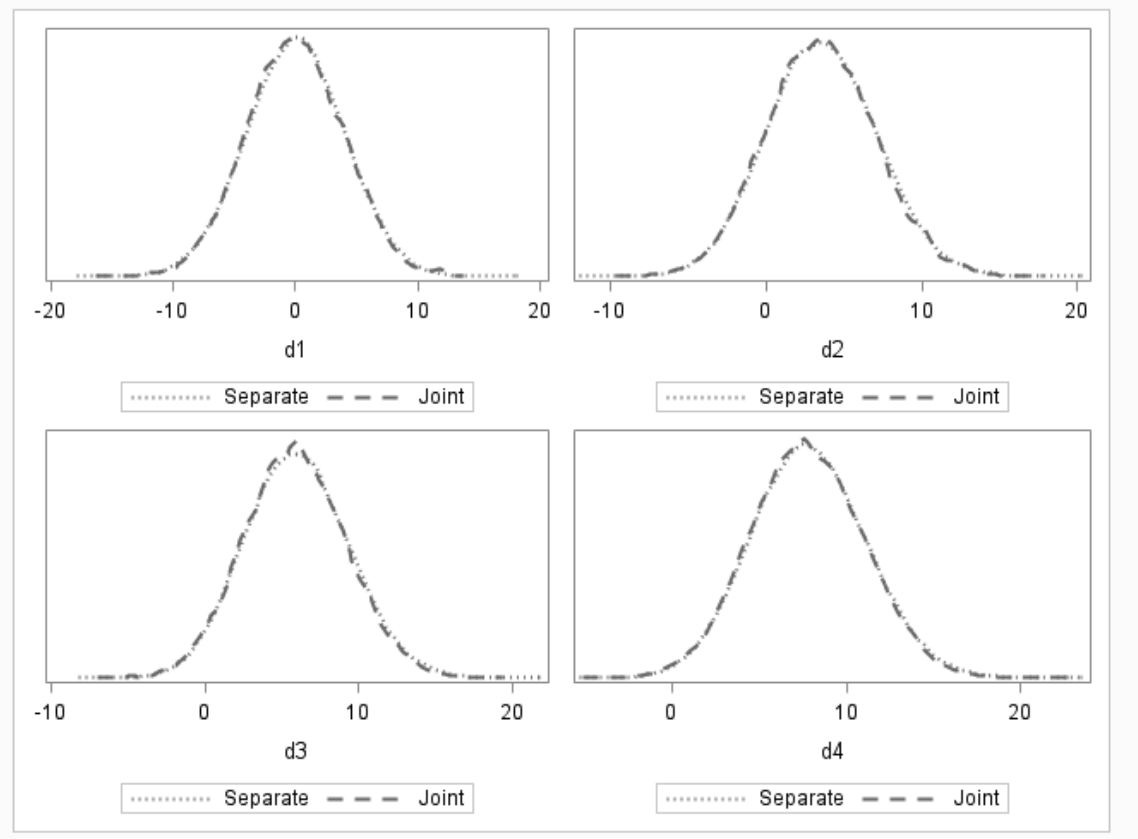
Comparison of Separate and Joint Prior Specifications

Program 5.5 presents code to test the equivalence between separate and joint prior specifications for Method 1. Output 5.2 compares the densities for the two sets of parameters. As you can see, there is essentially complete overlap between the densities for the four parameters ($d1$– $d4$), indicating equivalence between the approaches. Any trivial differences are because direct sampling is used to sample values in the separate prior specifications, whereas the metropolis sampler is used to sample values in the joint prior specifications.

Program 5.5: Comparing Separate and Joint Prior Specifications—Method 1

```
proc mcmc data=a nmc=100000 seed=1 outpost=o1 plots=(density);
   parm d1 -1 d2 -.5 d3 .5 d4 1;
   prior d1 ~ n(0, sd=5);
   prior d2 ~ n(0, sd=5, lower=d1);
   prior d3 ~ n(0, sd=5, lower=d2);
   prior d4 ~ n(0, sd=5, lower=d3);
   model general(0);
   run;

proc mcmc data=a outpost=o2 seed=23 nbi=1000 nmc=100000 plots=(density);
   parms d1 -1 d2 -.5 d3 .5 d4 1;
   beginnodata;
       lprior = lpdfnorm(d1,0,5);
       lprior = lprior + lpdfnorm(d2,0,5,d1);
      lprior = lprior + lpdfnorm(d3,0,5,d2);
      lprior = lprior + lpdfnorm(d4,0,5,d3);
   endnodata;
   prior d: ~general(lprior);
   model general(0);
run;
```

Output 5.2: Densities of the Four Parameters for Separate and Joint Prior Specifications



Output from Estimating the GR Model (Method 1 Prior Specification)

Output 5.3 presents selected posterior summary results from executing Program 5.2. These results were obtained using the autocall macro %POSTSUM to select a subset of parameters (see Chapter 3 for a discussion of the autocall macros that can be used with PROC MCMC OUTPOST data sets). As a result, the summary statistics differ from the default output produced by PROC MCMC. All slope parameter results are displayed along with a subset of threshold parameters due to the large number of threshold parameters in the model (40). These particular items (1, 7, and 10) were selected because they reflect measurement of the trait at varying locations on the θ scale. Point estimates for all parameters are available in a later section that compares Bayesian and marginal maximum likelihood (MML) parameter estimates.

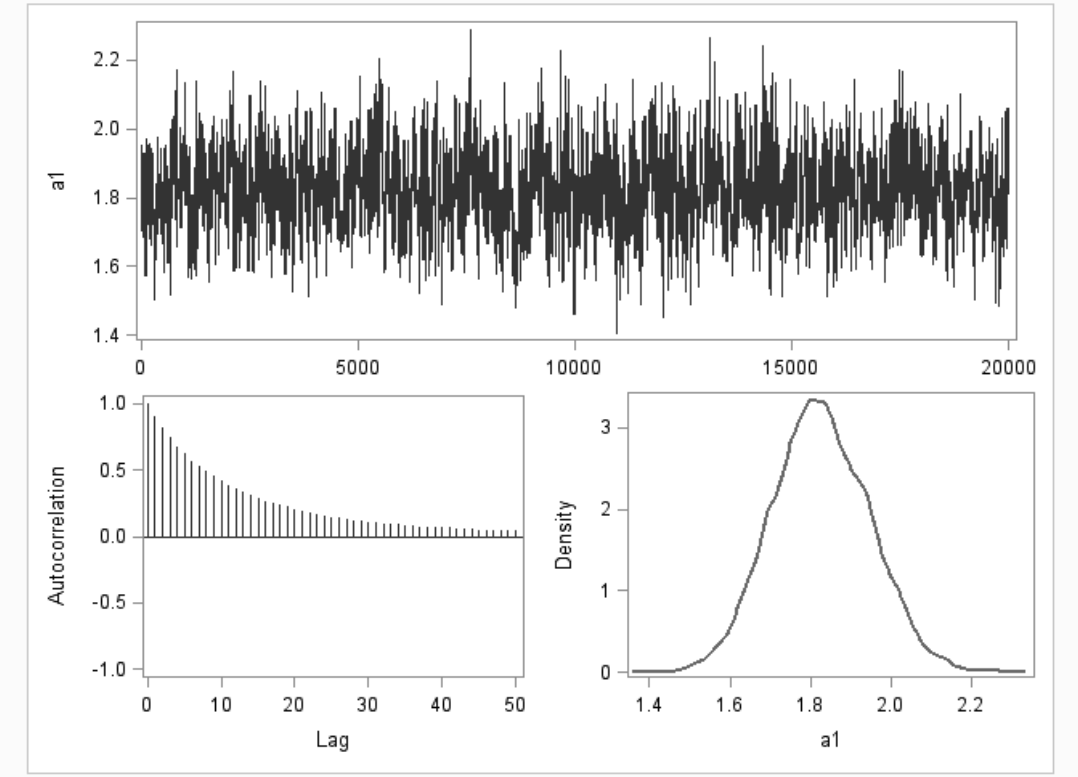
As you can see in the output, the slope point estimates (means) range from 1.83 to 3.49 indicating that the DASH items are highly discriminating. The threshold estimates for the selected items reflect the trend for all the items and do not spread out over the entire latent trait scale. A majority of the thresholds estimates are larger than 0, indicating that the subset of DASH items is measuring the trait at the upper end of the trait scale. Given the order constraint for the threshold parameters, you can see that the four thresholds within each item are indeed ordered. In addition, the standard deviations (SDs) of the posterior distributions for the parameters are quite small, indicating that all the slope and threshold parameter estimates were estimated with relatively high precision.

Output 5.3: Selected Posterior Summary Results—GR Model

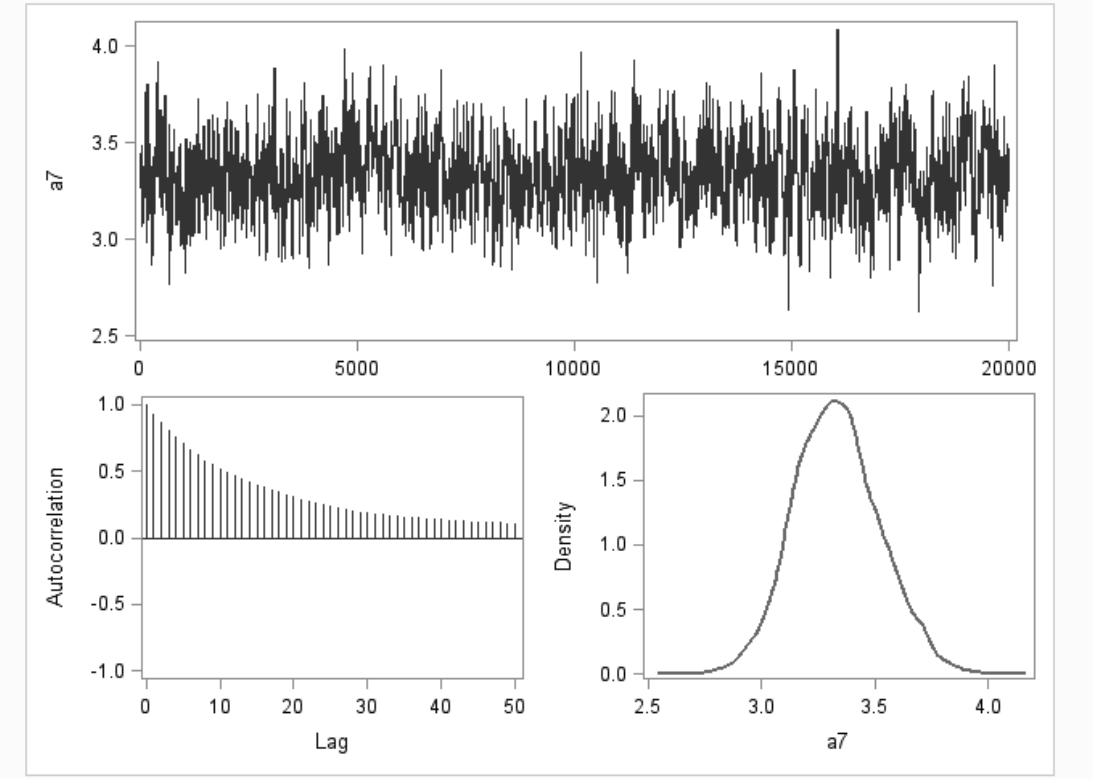
Parameter	N	Mean	StdDev	P25	P50	P75
a1	20000	1.82683	0.11983	1.74552	1.82346	1.90889
a2	20000	2.07275	0.12846	1.98427	2.06873	2.15686
a3	20000	3.49384	0.21214	3.34467	3.49237	3.62595
a4	20000	2.50094	0.13130	2.41279	2.49856	2.58600
a5	20000	2.78701	0.15396	2.68003	2.78485	2.89064
a6	20000	3.20610	0.18052	3.07562	3.19986	3.32635
a7	20000	3.33294	0.18773	3.20129	3.32623	3.45522
a8	20000	3.18409	0.17346	3.05617	3.17784	3.30121
a9	20000	2.63431	0.16026	2.52405	2.63083	2.74059
a10	20000	2.32115	0.12351	2.23545	2.31799	2.40305
b1_1	20000	0.49843	0.05959	0.45971	0.49890	0.53857
b2_1	20000	1.21552	0.08013	1.16203	1.21323	1.26861
b3_1	20000	2.17445	0.12077	2.09125	2.16983	2.25459
b4_1	20000	2.97099	0.17604	2.84972	2.96577	3.08890
b1_7	20000	-0.38875	0.04632	-0.41933	-0.38860	-0.35687
b2_7	20000	0.34375	0.04537	0.31157	0.34264	0.37403
b3_7	20000	1.18129	0.05912	1.13957	1.18079	1.22067
b4_7	20000	1.53994	0.07241	1.48955	1.53894	1.58789
b1_10	20000	-1.58753	0.08077	-1.64081	-1.58700	-1.53221
b2_10	20000	-0.73510	0.05660	-0.77245	-0.73501	-0.69712
b3_10	20000	0.05702	0.04944	0.02239	0.05758	0.09045
b4_10	20000	0.68886	0.05586	0.64969	0.68783	0.72861

Because of the volume of diagnostic plots that are available from this analysis, Outputs 5.4 to 5.12 present selected trace, autocorrelation, and density plots for item parameters in the GR model. These results were obtained using the autocall macro %TADPLOT to select the subset of parameters. You can see from the plots for the selected slope parameters (Outputs 5.4 – 5.6) that the parameters appear to be mixing well or traversing the parameter space quickly and have reasonably low autocorrelations. This is in contrast to the diagnostic plots for the selected threshold parameters (Outputs 5.7–5.12). You can see that the chains for the threshold parameters do not mix as well as the chains for the slope parameters. This results in somewhat higher autocorrelations and therefore higher dependence between sampled values. But as for other models, this higher dependence does not affect posterior summary results.

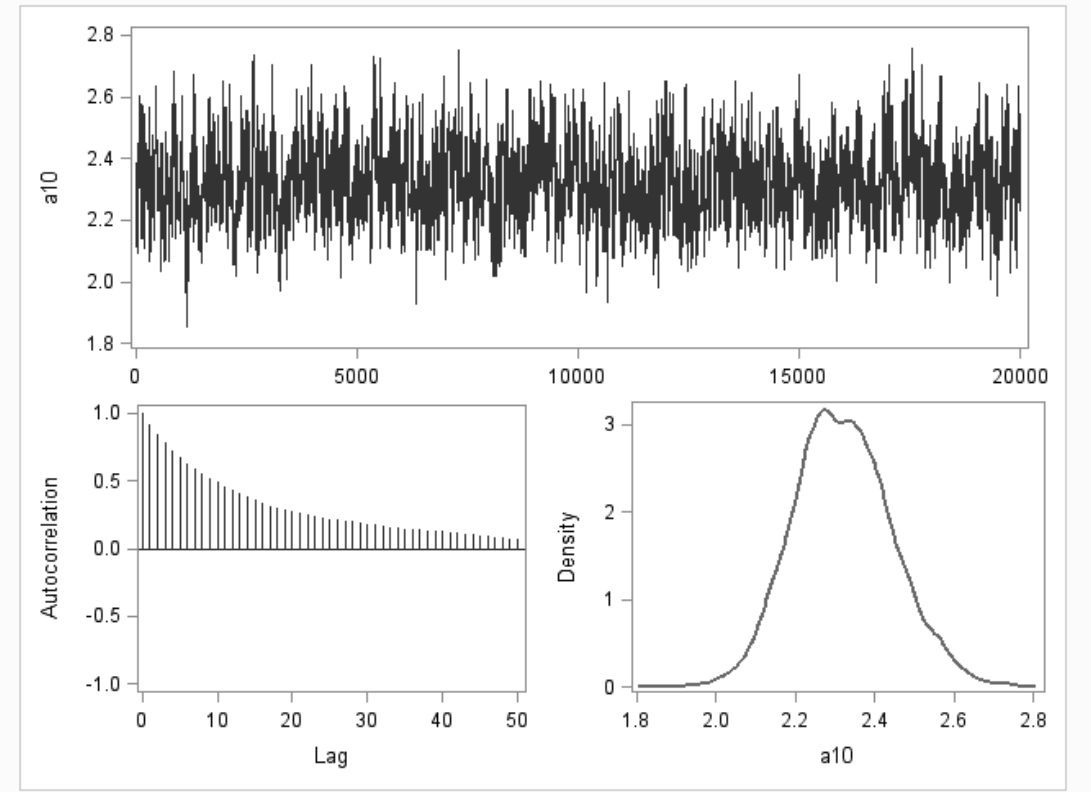
Output 5.4: Selected Diagnostics Plots for Slope Parameters—GR Model (Item 1)



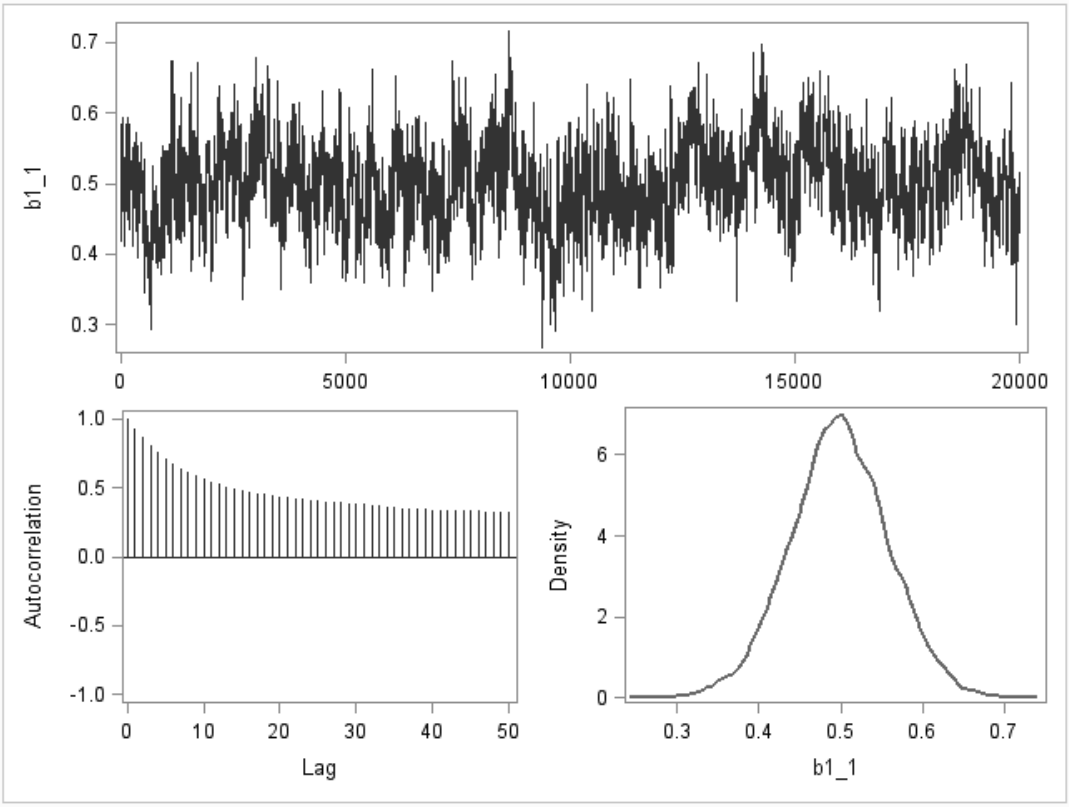
Output 5.5: Selected Diagnostics Plots for Slope Parameters—GR Model (Item 7)



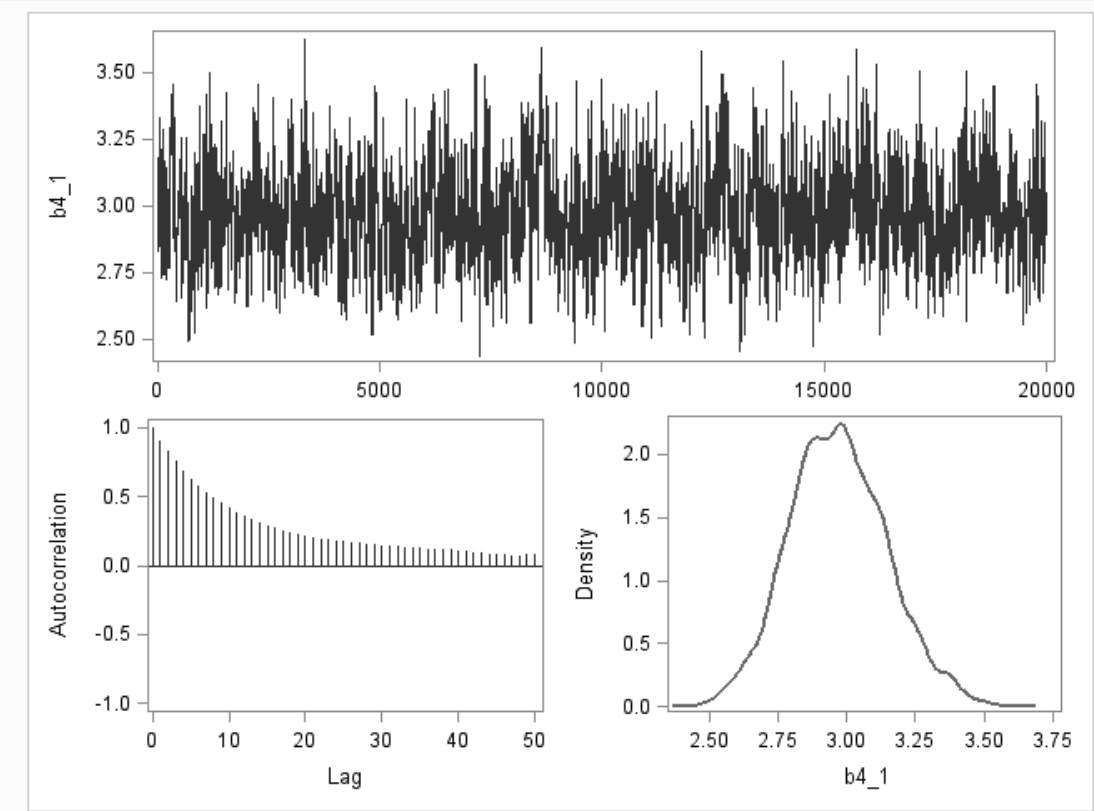
Output 5.6: Selected Diagnostics Plots for Slope Parameters—GR Model (Item 10)



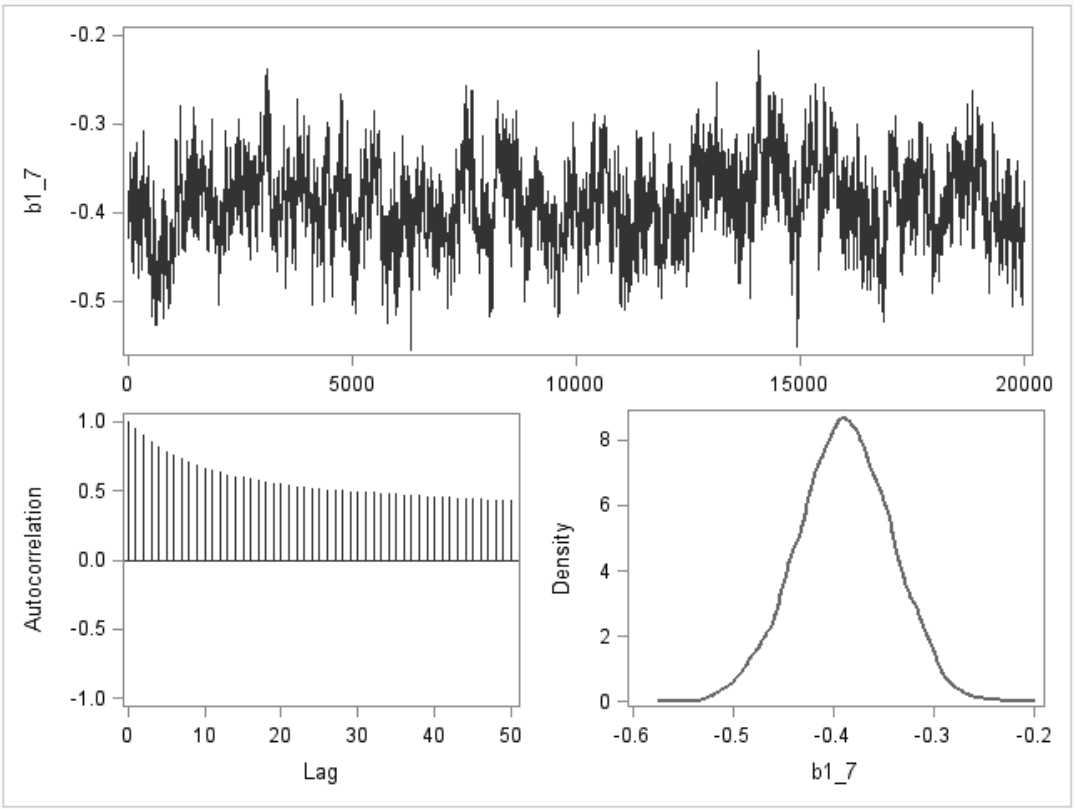
Output 5.7: Selected Diagnostics Plots for Threshold Parameters—GR Model (Item 1)



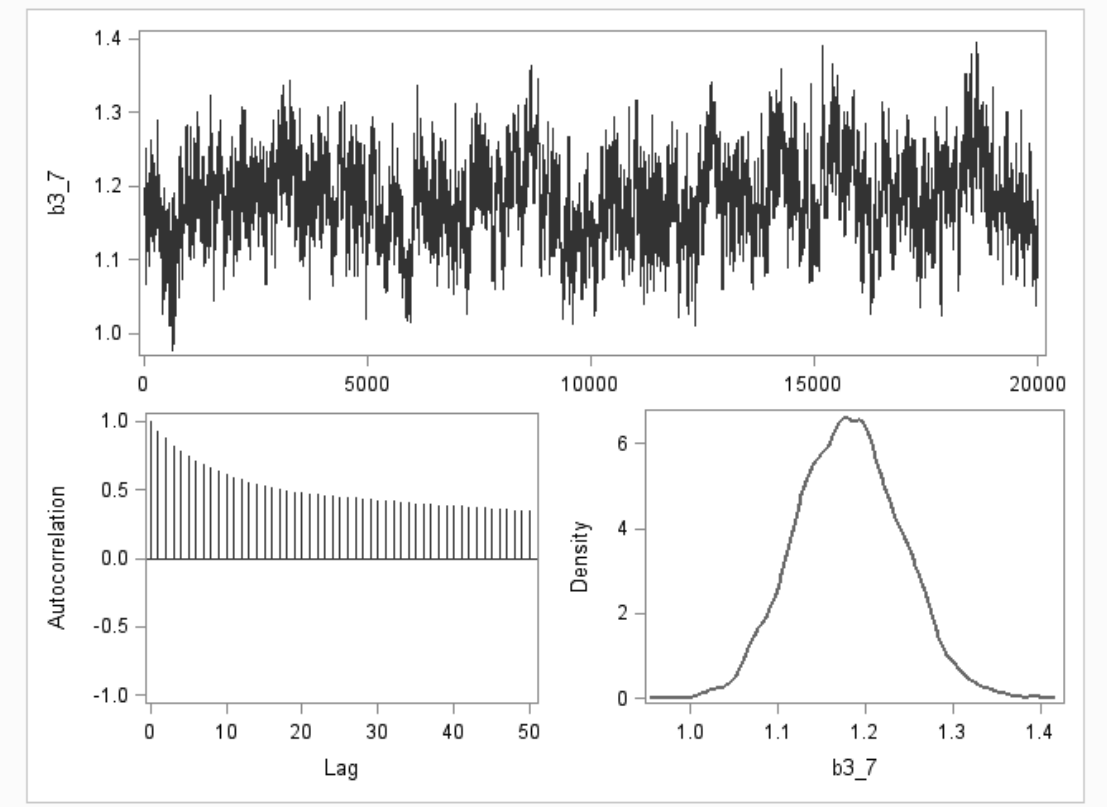
Output 5.8: Selected Diagnostics Plots for Threshold Parameters—GR Model (Item 1)



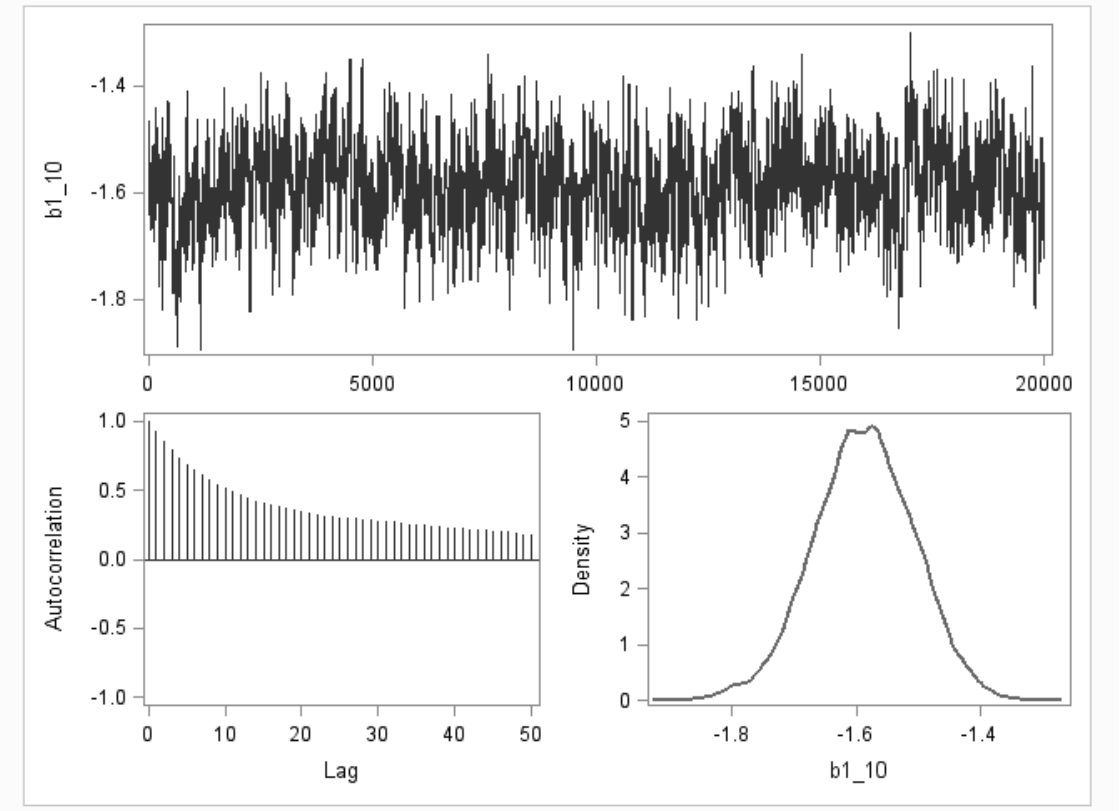
Output 5.9: Selected Diagnostics Plots for Threshold Parameters—GR Model (Item 7)



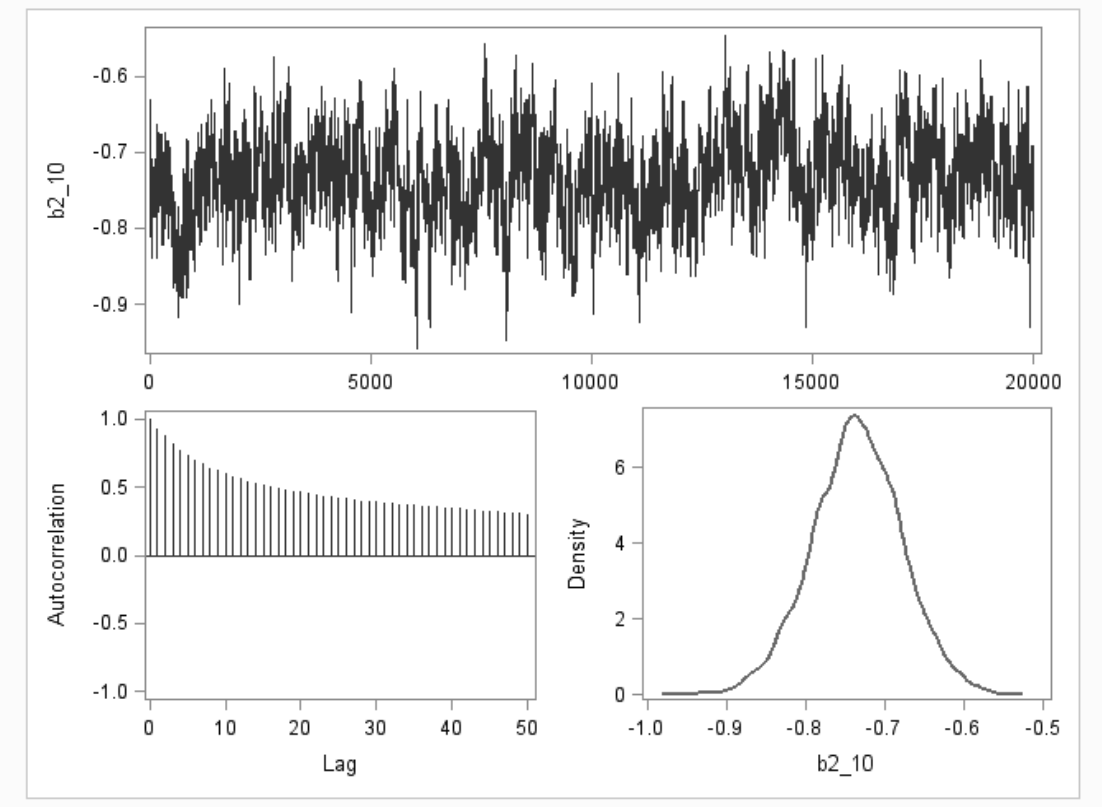
Output 5.10: Selected Diagnostics Plots for Threshold Parameters—GR Model (Item 7)



Output 5.11: Selected Diagnostics Plots for Threshold Parameters—GR Model (Item 10)



Output 5.12: Selected Diagnostics Plots for Threshold Parameters—GR Model (Item 10)

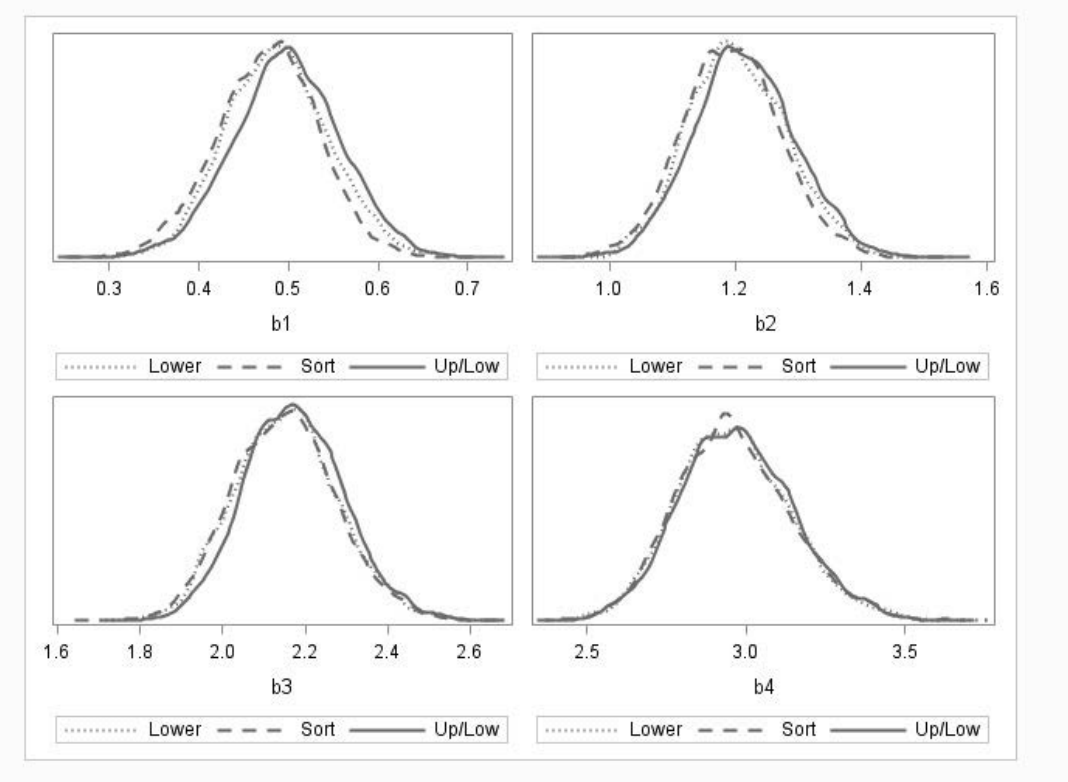


Comparison of the Posterior Densities for the Three Prior Specifications

Given the differences observed in the prior densities for the threshold parameters (see Output 5.1), you can examine any subsequent impact on the posterior densities for the parameters. Output 5.13 presents the posterior densities for the threshold parameters values for Item 1 under the three different prior density specifications. You can see that very small differences are observed between the methods using lower bounds (*Lower*) and the method using sorted values (*Sort*). However, there are some modest differences for several of the threshold parameters under the method using lower and upper bounds on the distributions (*Up/Low*). In particular, for the thresholds b_1, b_2, and b_3, the densities are shifted slightly higher than those based on the other two methods. In terms of comparing the first two moments of the distributions, the moments only differed in the hundredths place. So essentially, the point estimates for the parameters and parameter standard errors were essentially identical across the three methods.

Thus, estimation of the GR model for the DASH data set was not sensitive to the method used to specify the prior distributions for the threshold parameters. This is likely due to the relatively large amount of data in the DASH data set which "overwhelmed" the priors. Of course, since real data is analysed, it is not clear which density is closer to true values. Therefore more exploration of these different prior specifications may be needed.

Output 5.13: Comparison of Posterior Densities for the Four Threshold Parameters (Item 1)



Computation of Transformations by Post-Processing Posterior Results

Any transformations of parameter estimates may be obtained by post-processing the data set of posterior results saved using the OUTPOST option on the PROC MCMC command. For example, rather than compute the threshold parameters (b_{jk}) using the intercept parameter (d_{jk}) and slope parameters (a_{jk}) in the PROC MCMC program (see program 5.2), the OUTPOST data set can be processed. Assuming the transformation is excluded from Program 5.2, the following DATA step and PROC MEANS command may be used to compute the values of b_{jk} for each iteration, and compute point estimates for the b_{jk} parameters from means of the transformed posterior samples for the b_{jk} parameters. The ease in post-processing of the posterior samples illustrates the flexibility of using PROC MCMC for Bayesian analyses.

```
data dash_transformed;
   set dash_postGR;
   array a[10] a1-a10;
   array d[4,10] d1_1-d1_10 d2_1-d2_10 d3_1-d3_10 d4_1-d4_10;
   array b[4,10] b1_1-b1_10 b2_1-b2_10 b3_1-b3_10 b4_1-b4_10;
   do j=1 to 10;
      do k=1 to 4;
         b[k,j]=d[k,j]/a[j];
      end;
   end;
run;

proc means data=dash_transformed;
     var a1-a10  d1_1-d1_10 d2_1-d2_10 d3_1-d3_10 d4_1-d4_10 b1_1-b1_10
     b2_1-b2_10 b3_1-b3_10 b4_1-b4_10 ;
run;
```

Specification of the Likelihood Model with Use of the Table Function

Beginning with SAS/STAT 13.2, the likelihood model for items with more than two response categories can be specified using the TABLE function. The TABLE function supports the specification of a categorical distribution with an array of probabilities for the discrete values. As an example, the code for using this function with a 5-category item (Item 1 from the DASH data set) is presented below.

```
array p_star[4];
array p[5];
do k=1 to 4;
   p_star[k]=logistic(a[&j]*theta-d[k,1]);
end;
p[1]=1-p_star[1];
do k=2 to 4;
   p[k]=p_star[(k-1)]-p_star[k];
end;
p[5]=p_star[4];
model ix1 ~ table(p);
```

The TABLE function requires a vector of probabilities. A two-dimensional array of probabilities, indexed by item and response option categories, cannot be specified. This makes the use of the TABLE function cumbersome when estimating IRT models because the code must be repeated for each item separately rather than including the code within a DO LOOP across the items. Thus, you may find it easier to program the log-likelihood function explicitly as in Program 5.2. However, the macro presented below can be used to replicate the above code for each of the items in your data set.

```
array p_star[4];
array p[5];
%macro prob_gr;
   %do j=1 %to 10;
     do k=1 to 4;
        p_star[k]=logistic(a[&j]*theta-d[k,&j]);
     end;
     p[1]=1-p_star[1];
     do k=2 to 4;
        p[k]=p_star[(k-1)]-p_star[k];
     end;
     p[5]=p_star[4];
     model ix&j ~ table(p);
   %end;
%mend;
%prob_gr;
```

Estimation of the One-Parameter Graded Response Model

For the GR model, one slope parameter and four threshold parameters are estimated for each item. Under the assumption that all items have a common slope parameter, the GR model reduces to a one-parameter GR model with the logistic deviate $z_{jk} = Da(\theta - b_{jk})$ *or* $z_{jk} = D(a\theta - d_{jk})$. Therefore, the one-parameter GR model can be treated as a restricted case of the GR model. These two models can be compared using Bayesian model comparison methods (see Chapter 7) to determine which model is preferred in a particular testing application.

The PROC MCMC commands used to estimate the one-parameter GR model are provided in Program 5.6. You can see that the commands are similar to the commands used to estimate a GR model in Program 5.2, except that the slope parameter (a) is a single variable reflecting a common slope parameter across items. Thus, a single variable is specified in a PARMS statement for the parameter, PARMS A 1, in the prior distribution for the slope parameter, PRIOR A ~ LOGNORMAL(0, VAR=25), and in the response probability model, P_STAR[J,K]=LOGISTIC(A*THETA-D[K,J]).

No output is provided for this model. But it can be estimated and compared with the GR model to determine whether slope parameters for each item should be estimated. As it turns out, for the DASH data set, the point estimate (mean of posterior values) for the slope parameter a is 2.63, which is similar to the

mean value of the slope estimates from the GR model. However, the a_{jk} parameter estimates from the GR model varied a good deal (see Output 5.3) with a range from 1.82 to 3.51. This would suggest that a GR model would be more appropriate for the DASH item responses. The model comparison methods discussed in Chapter 7 would, however, be more useful in determining the appropriateness of the one-parameter GR model to the DASH data. As for the threshold b_{jk} parameters, most of the values were similar to the values from the GR model.

Program 5.6: PROC MCMC Code for Estimating the One-Parameter GR Model

```
proc mcmc data=dash_data outpost=dash_post1PGR seed=23 nbi=5000 nmc=20000
nthreads=8 monitor=(a b) diagnostics=all plots=(trace autocorr);

   array ix[10];
   array b[4,10] b1_1-b1_10  b2_1-b2_10 b3_1-b3_10 b4_1-b4_10;
   array d[4,10] d1_1-d1_10  d2_1-d2_10 d3_1-d3_10 d4_1-d4_10;
   array p_star[10,4]; array p[10,5];

   beginnodata;
   lprior=0;
   do j=1 to 10;
     lprior = lprior + lpdfnorm(d[1,j],0,5);
     do k=2 to 4;
        lprior = lprior + lpdfnorm(d[k,j],0,5,d[k-1,j]);
     end;
   end;

   do j=1 to 10;
     do k=1 to 4;
        b[k,j] = d[k,j]/a;
     end;
   end;
   endnodata;

   parms  a  1;
   parms  d1_1 -1  d2_1 -0.5  d3_1 0  d4_1 1;
   parms  d1_2 -1  d2_2 -0.5  d3_2 0  d4_2 1;
   parms  d1_3 -1  d2_3 -0.5  d3_3 0  d4_3 1;
   parms  d1_4 -1  d2_4 -0.5  d3_4 0  d4_4 1;
   parms  d1_5 -1  d2_5 -0.5  d3_5 0  d4_5 1;
   parms  d1_6 -1  d2_6 -0.5  d3_6 0  d4_6 1;
   parms  d1_7 -1  d2_7 -0.5  d3_7 0  d4_7 1;
   parms  d1_8 -1  d2_8 -0.5  d3_8 0  d4_8 1;
   parms  d1_9 -1  d2_9 -0.5  d3_9 0  d4_9 1;
   parms  d1_10 -1  d2_10 -0.5  d3_10 0  d4_10 1;

   prior a ~lognormal(0, var=25);
   prior d: ~general(lprior);
   random theta ~ normal(0, var=1) subject=_obs_;

   llike=0;
   do j=1 to 10;
     do k=1 to 4;
        p_star[j,k]=logistic(a*theta-d[k,j]);
     end;
     p[j,1]=1-p_star[j,1];
     do k=2 to 4;
        p[j,k]=p_star[j,(k-1)]-p_star[j,k];
     end;
     p[j,5]=p_star[j,4];
     llike = llike + log(p[j,ix[j]]);
   end;
   model general(llike);
  run;
```

Muraki's Rating Scale Model

Another restricted version of the GR model is Muraki's (1990) rating scale graded (RS-GR) model. In this model, the logistic deviate is $z_{jk} = Da_j(\theta - (b_j + c_k))$. The threshold parameters (b_{jk}) in the GR model are decomposed into two terms: (1) a location parameter (b_j) for each item, and (2) one set of category threshold parameters (c_k) that applies to all items. To identify the model, $\sum c_k$ across the k response options for each item are constrained to be 0. Also note that the thresholds c_k are ordered similarly to the b_{jk} parameters in the GR model. One advantage of this parameterization is that the b_j parameters can be used to directly compare where different items are measuring on the latent θ continuum. Another advantage is that the number of parameters in the model is reduced significantly as compared to the GR model.

The RS model is a restricted version of the GR model because the category threshold parameters, c_k, are assumed to be equal across all items in the model, whereas they are free to vary across items in the GR model. Thus, the RS-GR model can be compared with the GR model to determine whether one set of category threshold parameters adequately describes the item responses. This model comparison is illustrated with the DASH data set in Chapter 7.

Estimating the RS-GR Model

Program 5.7 includes the PROC MCMC code for estimating the RS-GR model with the DASH data. One slope parameter (a_j) and one location parameter (b_j) are estimated for each DASH item, and there is one set of four category threshold parameters (c_1, c_2, c_3, c_4) for the five response categories: 1 ("*No Difficulty*") to 5 ("*Unable*"). The category thresholds parameters (c_j) are declared in a separate PARMS statement ❶. Notably, only three thresholds $c_2, c_3,$ and c_4 are defined in the PARMS statement. The fourth threshold (c_1) is obtained using $c_1 = -(c_2 + c_3 + c_4)$. This computation is embedded in a BEGINNODATA – ENDNODATA block and is used to satisfy the model identification constraint $\sum c_k = 0$ ❷. The lower value of c_3 is specified as c_2, and the lower value of c_4 is specified as c_3 to also satisfy the order constraint ❸. The code used to compute the log-likelihood function is the same as for the GR model except for the response model ❹.

Note that the same strategy of decomposing the threshold parameter into location (b_j) and category threshold parameters ($c_{j1}, c_{j2}, c_{j3}, c_{j4}$) can be done for each item in the GR model. This is illustrated in Program 5.8, which can be found on the SAS Press authors' web page.

Program 5.7: PROC MCMC Code for Estimating the RS-GR Model

```
proc mcmc data=dash_data outpost=dash_postRSGR seed=23 nbi=5000 nmc=20000
nthreads=8 monitor=(a b c) diagnostics=all plots=(trace autocorr);

   array ix[10];
   array a[10]; array b[10]; array c[4];
   array p_star[10,4]; array p[10,5];

   parms a1 1  b1 0;
   parms a2 1  b2 0;
   parms a3 1  b3 0;
   parms a4 1  b4 0;
   parms a5 1  b5 0;
   parms a6 1  b6 0;
   parms a7 1  b7 0;
   parms a8 1  b8 0;
   parms a9 1  b9 0;
   parms a10 1  b10 0;
   parms c2 -2  c3 0  c4 6; ❶

   beginnodata;
     c1=-(c2+c3+c4); ❷
   endnodata;

   prior a: ~ lognormal(0, var=25);
   prior b: ~ normal(0, var=25);
   prior c2 ~ normal(0, var=25);
   prior c3 ~ normal(0, var=25, lower=c2); ❸
   prior c4 ~ normal(0, var=25, lower=c3);

   random theta ~ normal(0, var=1) subject=_obs_;

   llike=0;
   do j=1 to 10;
      do k=1 to 4;
         p_star[j,k]=logistic(a[j]*(theta-(b[j]+c[k]))); ❹
      end;
      p[j,1]=1-p_star[j,1];
      do k=2 to 4;
         p[j,k]=p_star[j,(k-1)]-p_star[j,k];
      end;
      p[j,5]=p_star[j,4];
      llike = llike + log(p[j,ix[j]]);
   end;
   model general(llike);
 run;
```

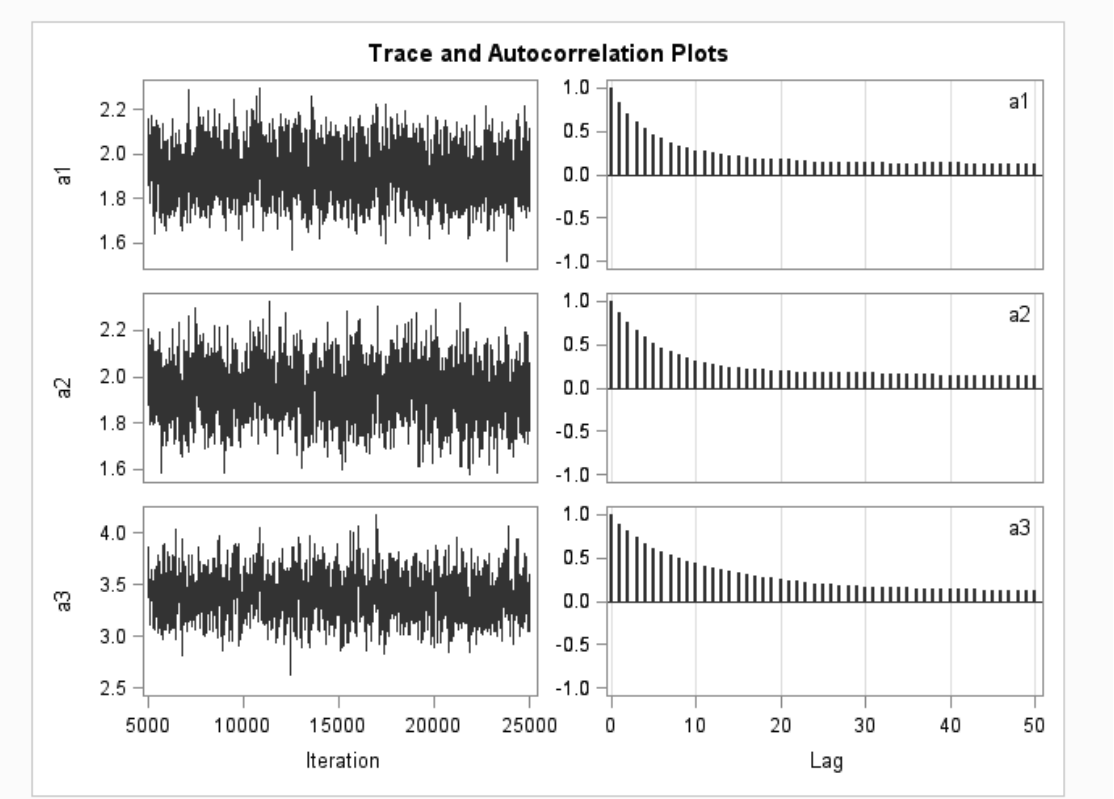
Output from Estimating the RS-GR Model

Output 5.14 shows the summary statistics for item parameters. Compared to the results for the GR model (see Output 5.3), the item slope estimates based on two models are similar. For each item, the item location parameter in the RS model approximates the average of the four thresholds in the GR model. The four category threshold parameter estimates for all items are ordered as expected. Outputs 5.15 to 5.18 provide the trace and autocorrelation plots for selected item parameters. You can see that the chains are mixing very well for the slope parameters with low autocorrelations. The location and category threshold parameters, on the other hand, exhibit reasonable mixing but a moderate degree of autocorrelation. To reduce autocorrelations in other models that have been discussed, the models where $z_{jk} = Da_j(\theta - b_{jk})$ were reparameterized into $z_{jk} = Da_j(\theta - d_{jk})$ where $b_{jk} = d_{jk}/a_j$. However, the RS-GR model is more difficult to reparameterize into a slope intercept model since the c_k parameters are not a function of each item.

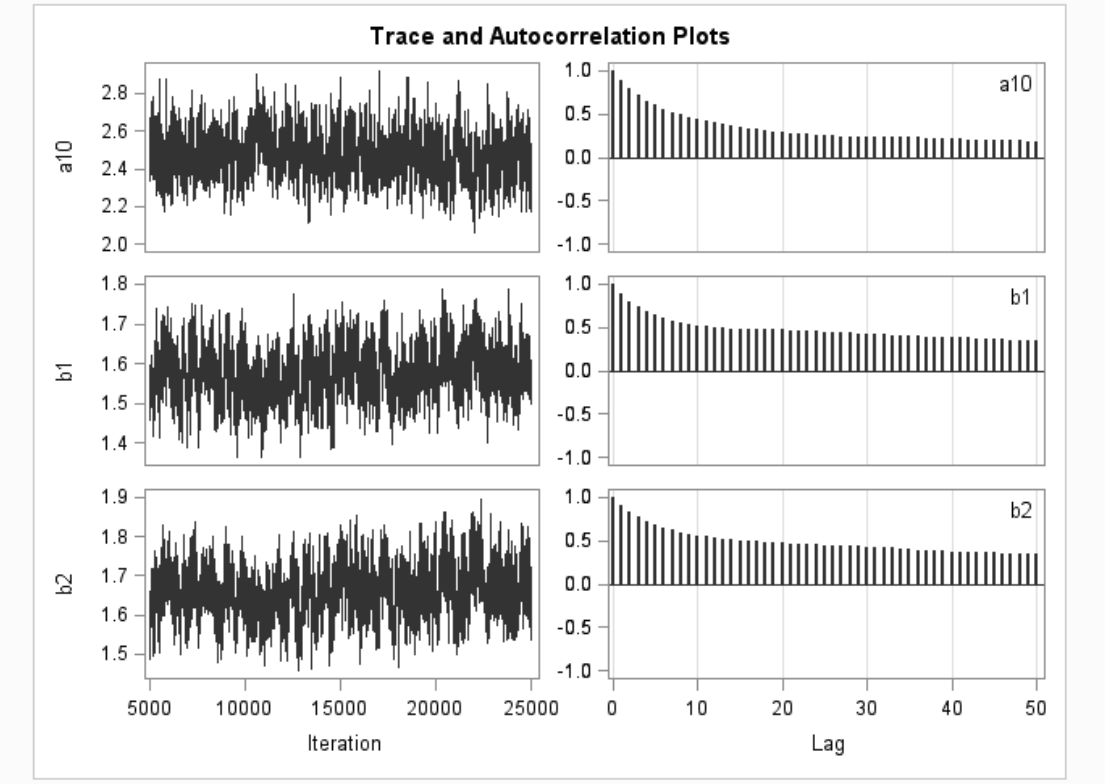
Output 5.14: Posterior Summary Statistics for the Item Parameters—RS-GR Model

Posterior Summaries and Intervals					
Parameter	N	Mean	Standard Deviation	95% HPD Interval	
a1	20000	1.9125	0.0997	1.7181	2.1088
a2	20000	1.9321	0.1085	1.7228	2.1458
a3	20000	3.3934	0.1872	3.0388	3.7657
a4	20000	2.7590	0.1324	2.4981	3.0142
a5	20000	2.5888	0.1244	2.3430	2.8309
a6	20000	3.0479	0.1638	2.7327	3.3770
a7	20000	3.2611	0.1727	2.9167	3.5796
a8	20000	3.0019	0.1581	2.6899	3.2983
a9	20000	2.1936	0.1251	1.9588	2.4377
a10	20000	2.4750	0.1169	2.2644	2.7194
b1	20000	1.5741	0.0614	1.4560	1.6982
b2	20000	1.6585	0.0665	1.5316	1.7855
b3	20000	1.1828	0.0455	1.1012	1.2731
b4	20000	0.7572	0.0425	0.6780	0.8437
b5	20000	0.1317	0.0394	0.0572	0.2122
b6	20000	0.9550	0.0430	0.8762	1.0437
b7	20000	0.6771	0.0391	0.5961	0.7520
b8	20000	0.2206	0.0381	0.1486	0.2970
b9	20000	1.5470	0.0611	1.4261	1.6629
b10	20000	-0.3847	0.0430	-0.4657	-0.2989
c1	20000	-1.0665	0.0292	-1.1208	-1.0108
c2	20000	-0.3545	0.0133	-0.3787	-0.3282
c3	20000	0.4217	0.0144	0.3954	0.4495
c4	20000	0.9993	0.0293	0.9446	1.0544

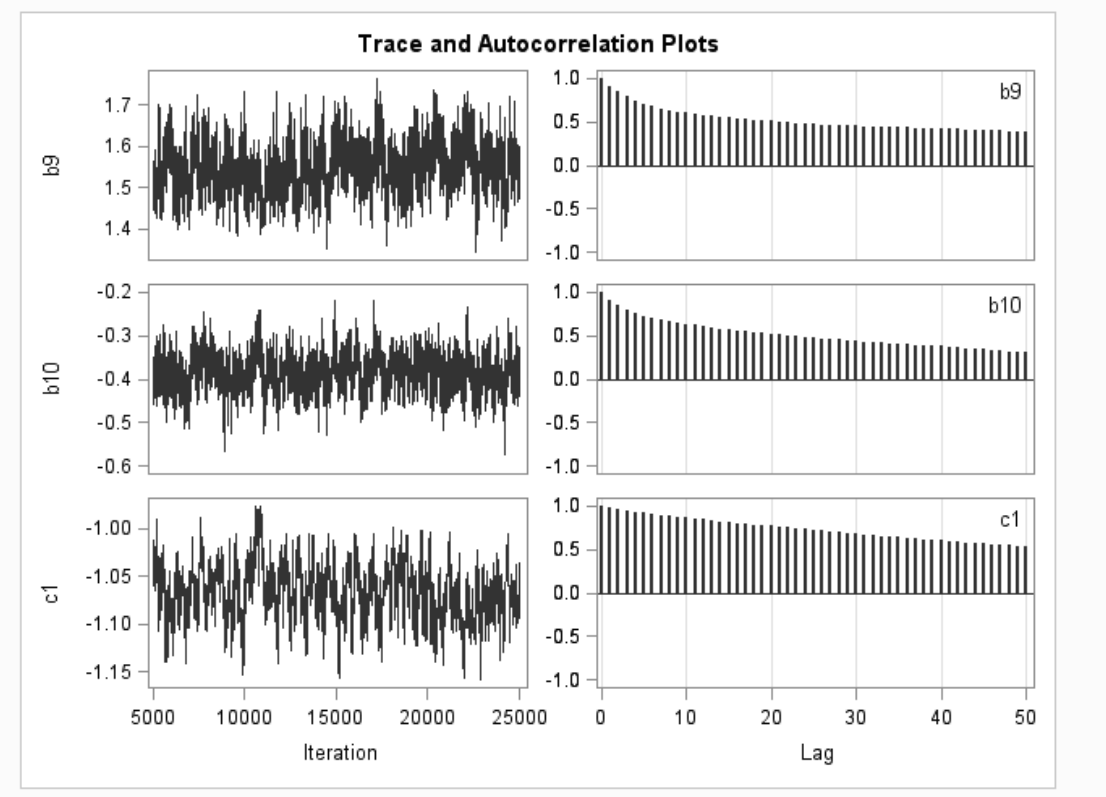
Output 5.15: Trace and Autocorrelation Plots for Item Parameters (a1, a2, a3)—RS-GR Model



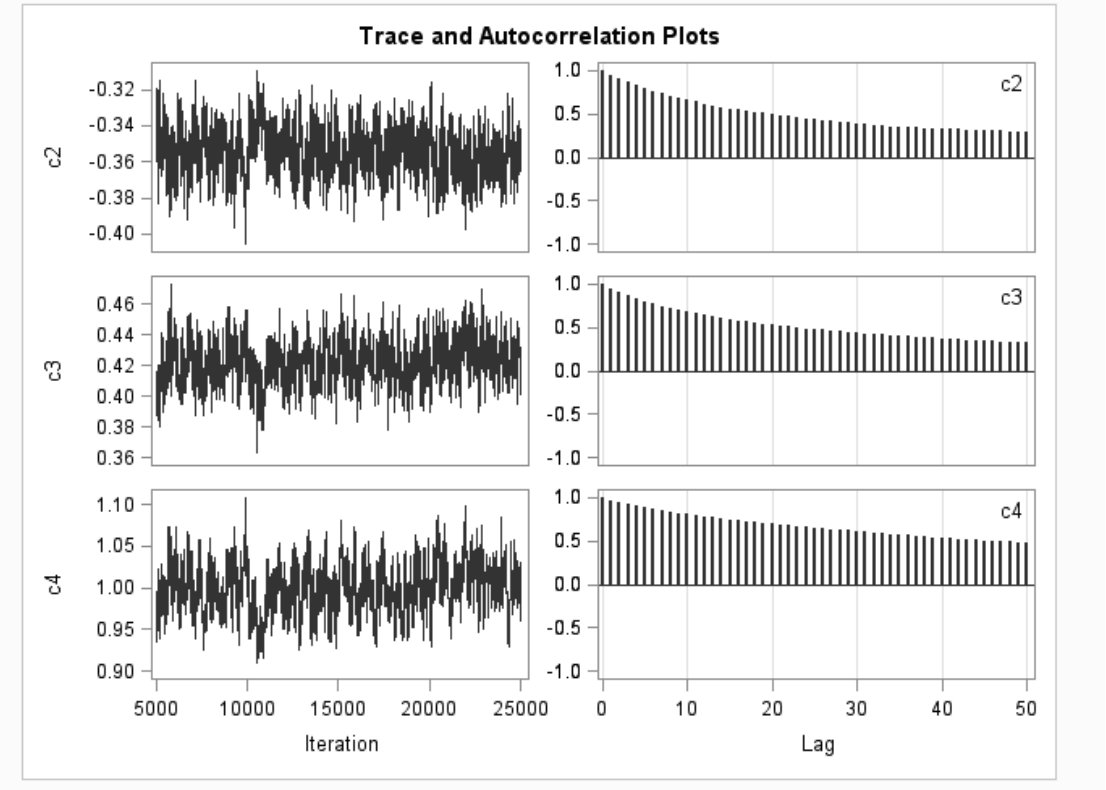
Output 5.16: Trace and Autocorrelation Plots for Item Parameters (a10, b1, b2)—RS-GR Model



Output 5.17: Trace and Autocorrelation Plots for Item Parameters (b9, b10, c1)—RS-GR Model



Output 5.18: Trace and Autocorrelation Plots for Item Parameters (c2, c3, c4)—RS-GR Model



The Nominal Response Model

Bock's (1972) nominal response (NR) model is the most general model for polytomously scored items and is used to model item responses when response categories are not necessarily ordered along the trait (θ) scale. In the NR model, the probability of an examinee with trait level θ responding in the k^{th} response option, for item *j* with m_j response options (or categories), is given by the following:

$$P_{jk}(\theta) = \frac{e^{z_{jk}}}{\sum_{k=1}^{m_j} e^{z_{jk}}}$$

for

$$k = 1, \ldots., m_j,$$

where $z_{jk} = a_{jk}\theta + c_{jk}$ is a multivariate logit for the k^{th} option in item *j*, a_{jk} is the option discrimination parameter for item *j* and category k, and c_{jk} is called the option extremity parameter for item *j* and category k. As discussed in Chapter 1, to identify the model, the constraint $\sum a_{jk} = \sum c_{jk} = 0$ across *k* or the constraint $a_{j1} = c_{j1} = 0$ for each item is typically specified.

Estimating the NR Model

Program 5.9 includes the PROC MCMC code for estimating the NR model with the DASH data set. For each DASH item, there are five slope parameters $(a_{j1}, a_{j2}, a_{j3}, a_{j4}, a_{j5})$ and five intercept parameters $(c_{j1}, c_{j2}, c_{j3}, c_{j4}, c_{j5})$ corresponding to five response categories. However, for model identification purposes, the first slope and intercept for each item are constrained to be 0 in a BEGINNODATA–ENDNODATA block. Two-dimensional arrays are used for the item parameters where the first index corresponds to the item number and the second index corresponds to the category response number. The multivariate logits and response probabilities for the 5 options are captured in the Z[5] and P[5] arrays. The item parameters within each item are treated as a block, and their initial values are set to be 0. The priors for slopes and intercepts are declared as unbounded normal distributions with mean of 0 and variance of 25 ❶. To obtain the likelihood of the data, the multivariate logits for the second to fifth response options (z_2 to z_5) are calculated first ❷. Note that the logit for the first option (z_1) is 1 since its slope and intercept are both 0. In the next step, the response probabilities for 5 options are computed ❸.

Program 5.9: PROC MCMC Code for Estimating the NR Model

```
proc mcmc data=dash_data outpost=dash_postNR seed=23 nbi=5000 nmc=20000
monitor=(a c) diagnostics=all plots=all;

  array a[10,5] a1_1-a1_5 a2_1-a2_5 a3_1-a3_5 a4_1-a4_5 a5_1-a5_5
                a6_1-a6_5 a7_1-a7_5 a8_1-a8_5 a9_1-a9_5 a10_1-a10_5;
  array c[10,5] c1_1-c1_5 c2_1-c2_5 c3_1-c3_5 c4_1-c4_5 c5_1-c5_5
                c6_1-c6_5 c7_1-c7_5 c8_1-c8_5 c9_1-c9_5 c10_1-c10_5;
  array ix[10]; array z[5]; array p[5];
  parms a1_2-a1_5 c1_2-c1_5 0;
  parms a2_2-a2_5 c2_2-c2_5 0;
  parms a3_2-a3_5 c3_2-c3_5 0;
  parms a4_2-a4_5 c4_2-c4_5 0;
  parms a5_2-a5_5 c5_2-c5_5 0;
  parms a6_2-a6_5 c6_2-c6_5 0;
  parms a7_2-a7_5 c7_2-c7_5 0;
  parms a8_2-a8_5 c8_2-c8_5 0;
  parms a9_2-a9_5 c9_2-c9_5 0;
  parms a10_2-a10_5 c10_2-c10_5 0;

  beginnodata;
  a1_1=0;a2_1=0;a3_1=0;a4_1=0;a5_1=0;a6_1=0;a7_1=0;a8_1=0;a9_1=0;a10_1=0;
  c1_1=0;c2_1=0;c3_1=0;c4_1=0;c5_1=0;c6_1=0;c7_1=0;c8_1=0;c9_1=0;c10_1=0;
  endnodata;

  prior c1_2-c1_5 c2_2-c2_5 c3_2-c3_5 c4_2-c4_5 c5_2-c5_5
        c6_2-c6_5 c7_2-c7_5 c8_2-c8_5 c9_2-c9_5 c10_2-c10_5
        ~ normal(0, var=25); ❶
  prior a1_2-a1_5 a2_2-a2_5 a3_2-a3_5 a4_2-a4_5 a5_2-a5_5
        a6_2-a6_5 a7_2-a7_5 a8_2-a8_5 a9_2-a9_5 a10_2-a10_5
        ~ normal(0, var=25);
  random theta ~ normal(0, var=1) subject=_obs_;
  llike=0;
  do j=1 to 10;
     do k=2 to 5; ❷
        z[k]=exp(a[j,k]*theta + c[j,k]);
     end;
     do k=2 to 5; ❸
        p[k]=z[k]/(1 + sum(of z2-z5));
     end;
     p[1]=1 - sum(of p2-p5); * calc prob for category 1;
     llike = llike + log(p[ix[j]]);
  end;
  model general(llike);
  run;
```

Output from Estimating the NR Model

Output 5.19 shows the summary statistics for selected item parameters. Selected results for Items 1, 7, and 10 are again reported as these items reflect items with varying item properties. The autocall macro for regenerating results, %POSTSUM, was used, and, as in previous tables, the results from this macro differ from the default summary statistics produced by PROC MCMC. Note that all item parameter estimates are available in the section comparing Bayesian and MML results.

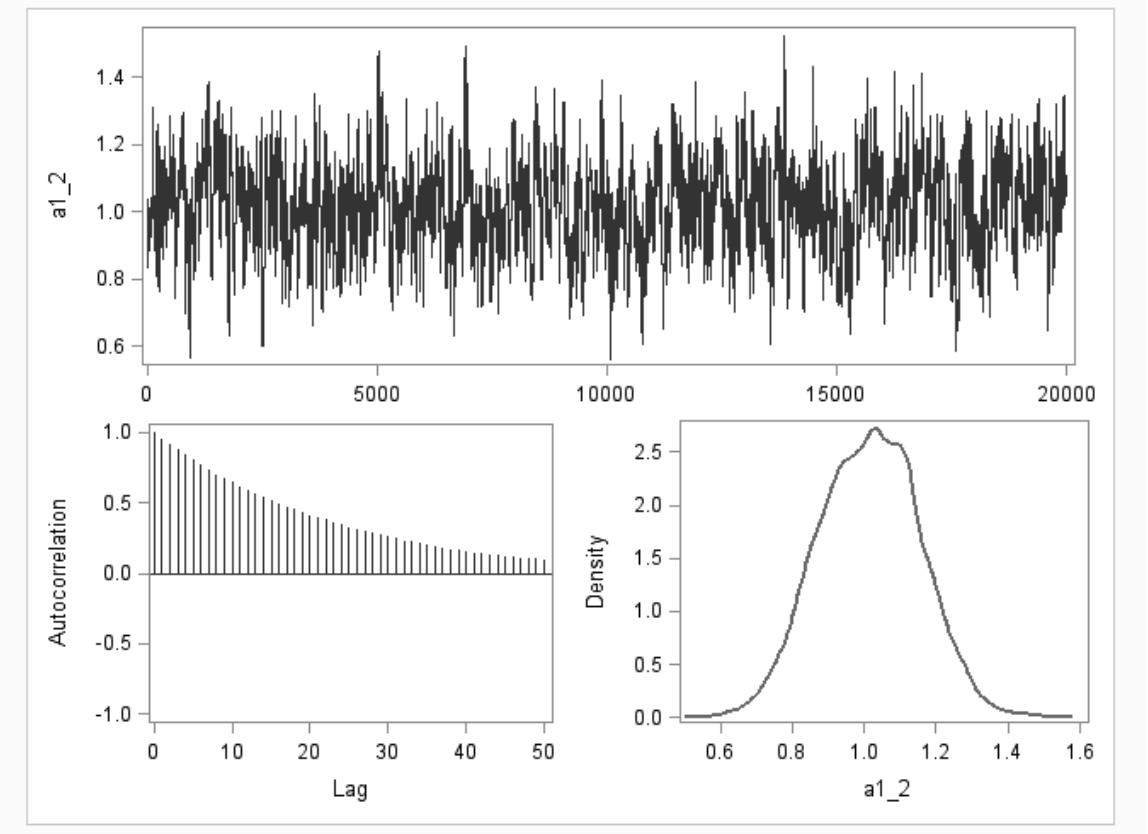
As you can see in the table, the a_{jk} parameters across response categories for an item are ordered. Given that the a_{jk} parameters reflect the slope of the linear regression of the response process (z) on θ for each response option k, increasing values of the a_{jk} parameters across response categories reflect an increasing probability of selecting option k+1 over option k. This in turn indicates that the scale of response options is ordered. Therefore, an ordered scale is indicated for the DASH items as would be expected.

Output 5.19: Selected Posterior Summary Statistics for the Item Parameters—NR Model

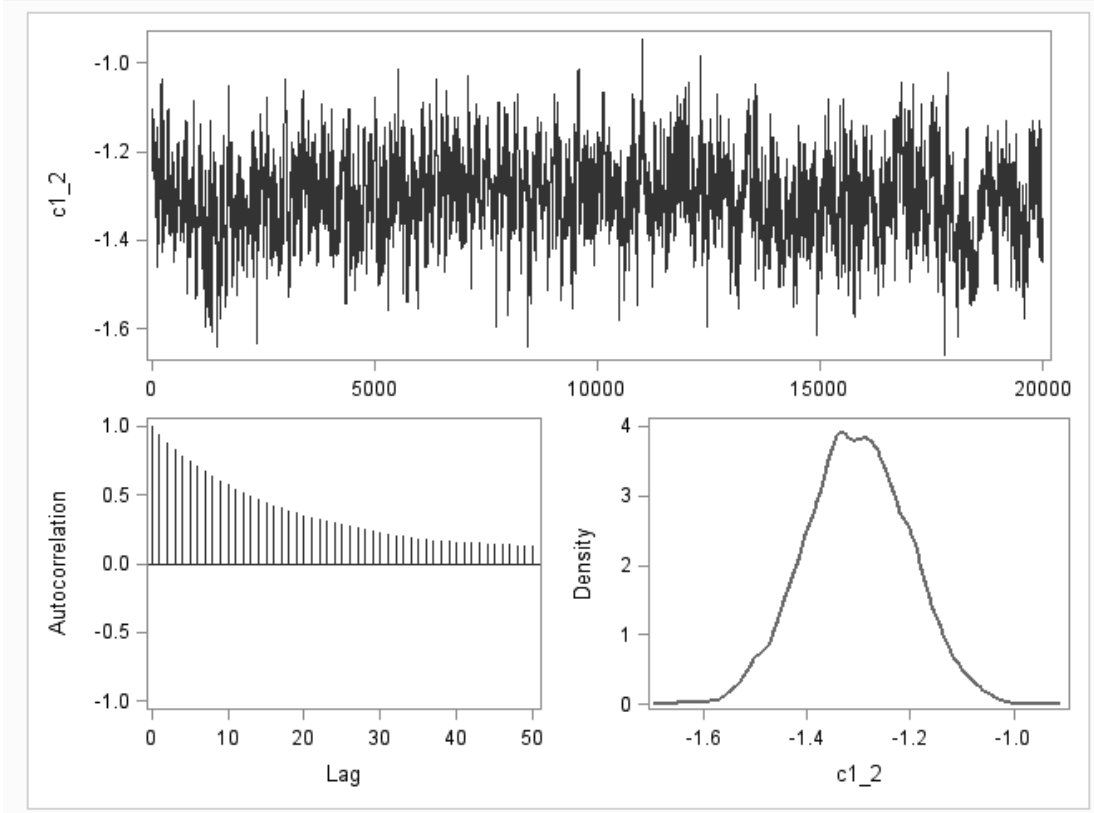
Parameter	N	Mean	StdDev	P25	P50	P75
a1_1	20000	0.00000	0.00000	0.00000	0.00000	0.0000
a1_2	20000	1.01527	0.13843	0.91723	1.01973	1.1127
a1_3	20000	2.30510	0.22294	2.14748	2.30099	2.4474
a1_4	20000	3.14763	0.33597	2.90296	3.13946	3.3768
a1_5	20000	4.63892	0.50150	4.29931	4.61398	4.9531
c1_1	20000	0.00000	0.00000	0.00000	0.00000	0.0000
c1_2	20000	-1.30335	0.09891	-1.36959	-1.30269	-1.2352
c1_3	20000	-2.35789	0.18737	-2.48225	-2.34597	-2.2245
c1_4	20000	-4.34708	0.38754	-4.59857	-4.33517	-4.0776
c1_5	20000	-7.29904	0.81826	-7.80153	-7.24497	-6.7228
a7_1	20000	0.00000	0.00000	0.00000	0.00000	0.0000
a7_2	20000	2.10357	0.22702	1.94789	2.09515	2.2438
a7_3	20000	4.59285	0.39538	4.31577	4.57409	4.8519
a7_4	20000	6.26462	0.59113	5.86207	6.22562	6.6447
a7_5	20000	9.54340	0.79714	8.99711	9.47940	10.0433
c7_1	20000	0.00000	0.00000	0.00000	0.00000	0.0000
c7_2	20000	0.55047	0.14342	0.45178	0.54482	0.6433
c7_3	20000	-0.24438	0.22397	-0.39357	-0.24335	-0.0929
c7_4	20000	-2.90596	0.45421	-3.19710	-2.87851	-2.5829
c7_5	20000	-6.46075	0.86821	-7.01838	-6.41505	-5.8953
a10_1	20000	0.00000	0.00000	0.00000	0.00000	0.0000
a10_2	20000	0.77367	0.22876	0.62049	0.77129	0.9195
a10_3	20000	2.01487	0.25893	1.83715	2.00525	2.1802
a10_4	20000	3.62973	0.31860	3.40961	3.61963	3.8378
a10_5	20000	5.38492	0.39372	5.10536	5.38092	5.6348
c10_1	20000	0.00000	0.00000	0.00000	0.00000	0.0000
c10_2	20000	1.39007	0.27718	1.20574	1.38767	1.5630
c10_3	20000	2.35598	0.26757	2.17409	2.34422	2.5274
c10_4	20000	2.12500	0.29196	1.92018	2.12039	2.3217
c10_5	20000	1.53693	0.33262	1.30885	1.53642	1.7576

Outputs 5.20 to 5.23 provide diagnostic plots for selected item parameters generated using the autocall macro %TADPLOT. These plots are typical for all the item parameters. As you can see, the chains for slopes and intercepts reflect chains that were mixing well and relatively low autocorrelations. The parameter densities reflect approximately normal distributions, although some of these distributions reflect rather large standard deviations.

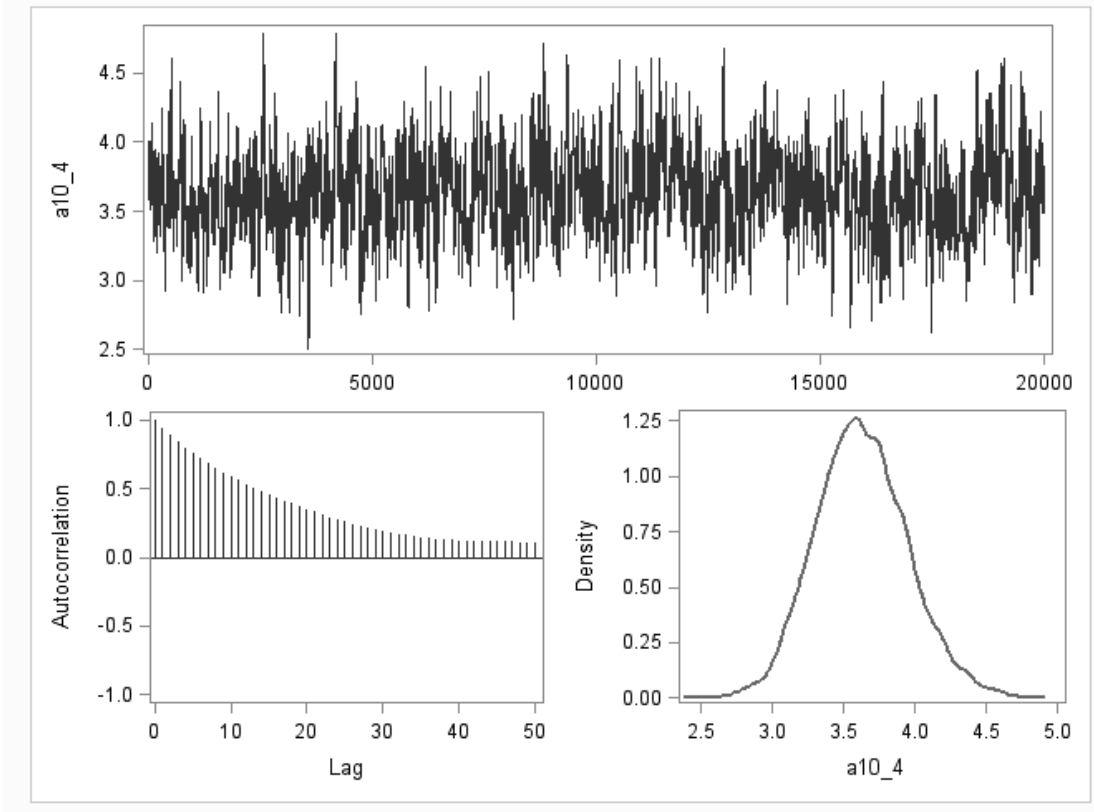
Output 5.20: Diagnostic Plots for Selected Item Parameters—NR Model



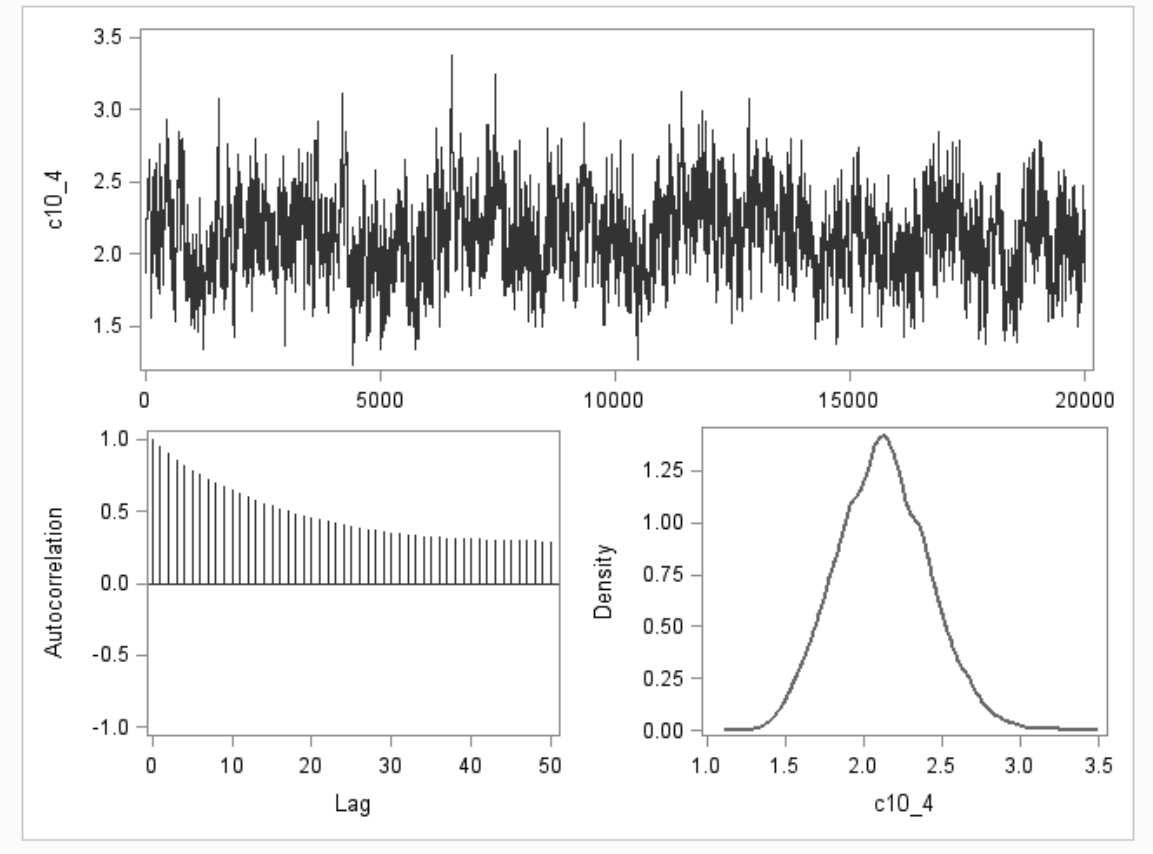
Output 5.21: Diagnostic Plots for Selected Item Parameters—NR Model



Output 5.22: Diagnostic Plots for Selected Item Parameters—NR Model



Output 5.23: Diagnostic Plots for Selected Item Parameters—NR Model



The parameters can be tranformed to conform to Bock's constraint, $\sum a_{jk} = \sum c_{jk} = 0$ across *k*, by centering the samples values. The program code that follows illustrates centering the sampled values that result from executing Program 5.9, and Output 5.24 presents output from executing the program code. In the program, centering of the values is obtained at each iteration in the posterior results by computing the sum of the sampled values for the a_{jk} and b_{jk} paramters for each item. The mean based on this sum (sum / 5) is then used to center the values for a_{jk} and b_{jk} by taking the sampled value minus the mean. PROC MEANS can then be used to obtain point estimates for the centered values. As you can see in the output (Output 5.24), the $\sum a_{jk} = \sum c_{jk} = 0$ across *k* for each item.

```
data dash_centered;
   set dash_postNR;
   array a[10,5] a1_1-a1_5 a2_1-a2_5 a3_1-a3_5 a4_1-a4_5 a5_1-a5_5
                 a6_1-a6_5 a7_1-a7_5 a8_1-a8_5 a9_1-a9_5 a10_1-a10_5;
   array c[10,5] c1_1-c1_5 c2_1-c2_5 c3_1-c3_5 c4_1-c4_5 c5_1-c5_5
                 c6_1-c6_5 c7_1-c7_5 c8_1-c8_5 c9_1-c9_5 c10_1-c10_5;
   array a_c[10,5] a_c1_1-a_c1_5 a_c2_1-a_c2_5 a_c3_1-a_c3_5
              a_c4_1-a_c4_5 a_c5_1-a_c5_5 a_c6_1-a_c6_5 a_c7_1-a_c7_5
              a_c8_1-a_c8_5 a_c9_1-a_c9_5 a_c10_1-a_c10_5;
   array c_c[10,5] c_c1_1-c_c1_5 c_c2_1-c_c2_5 c_c3_1-c_c3_5
              c_c4_1-c_c4_5 c_c5_1-c_c5_5 c_c6_1-c_c6_5 c_c7_1-c_c7_5
              c_c8_1-c_c8_5 c_c9_1-c_c9_5 c_c10_1-c_c10_5;

   keep a_c1_1--a_c10_5 c_c1_1--c_c10_5;
   do j=1 to 10;
      m_a=sum(of a[j,2], a[j,3], a[j,4], a[j,5]);
      m_c=sum(of c[j,2], c[j,3], c[j,4], c[j,5]);
      do k=1 to 5;
         if k=1 then do;
            a_c[j,k]=-m_a/5;
            c_c[j,k]=-m_c/5;
         end;
         else do;
            a_c[j,k]=a[j,k]-m_a/5;
            c_c[j,k]=c[j,k]-m_c/5;
         end;
      end;
   end;
run;
```

Output 5.24: Centered Item Parameters for Selected Items—NR Model

Variable	N	Mean	Std Dev	Minimum	Maximum
a_c1_1	20000	-2.2213829	0.1839222	-2.8998524	-1.6233387
a_c1_2	20000	-1.2061114	0.1720332	-1.8305153	-0.6019391
a_c1_3	20000	0.0837140	0.1542292	-0.5603044	0.5914316
a_c1_4	20000	0.9262465	0.2211556	0.0609770	1.6250210
a_c1_5	20000	2.4175338	0.3577571	1.1831710	4.2287058
c_c1_1	20000	3.0614710	0.2118123	2.2621511	3.9129648
c_c1_2	20000	1.7581182	0.2121112	1.0577052	2.6543831
c_c1_3	20000	0.7035850	0.2152275	0.0084019	1.5969653
c_c1_4	20000	-1.2856041	0.3128187	-2.4426934	-0.1655162
c_c1_5	20000	-4.2375700	0.6373911	-7.3446801	-2.3346525
a_c7_1	20000	-4.5008870	0.3300675	-5.8066812	-3.4121271
a_c7_2	20000	-2.3973197	0.2694364	-3.6418188	-1.5669759
a_c7_3	20000	0.0919655	0.2295359	-0.8840669	0.8685535
a_c7_4	20000	1.7637313	0.3467711	0.6502194	3.0082410
a_c7_5	20000	5.0425099	0.5417846	3.5401815	7.4974608
c_c7_1	20000	1.8121244	0.2662668	0.9255347	2.8846004
c_c7_2	20000	2.3625983	0.2236709	1.6297653	3.2565215
c_c7_3	20000	1.5677398	0.1999409	0.9317883	2.4267439
c_c7_4	20000	-1.0938396	0.3148387	-2.2610049	0.0170069
c_c7_5	20000	-4.6486230	0.6452243	-7.3412386	-2.5950851
a_c10_1	20000	-2.3606362	0.2078090	-3.2121818	-1.5771634
a_c10_2	20000	-1.5869702	0.1591477	-2.1925282	-1.1220746
a_c10_3	20000	-0.3457688	0.1287935	-0.7408829	0.1919973
a_c10_4	20000	1.2690952	0.1488751	0.7657871	1.8031600
a_c10_5	20000	3.0242800	0.2295841	2.3291078	4.1533546
c_c10_1	20000	-1.4815957	0.2164866	-2.4511990	-0.8337867
c_c10_2	20000	-0.0915296	0.1360116	-0.6482589	0.3097568
c_c10_3	20000	0.8743888	0.0889914	0.5553321	1.2188629
c_c10_4	20000	0.6434046	0.1049224	0.3102231	1.0632845
c_c10_5	20000	0.0553319	0.1602373	-0.5220587	0.5349513

The Generalized Partial Credit Model

The generalized partial credit (GPC) model (Muraki, 1992) is also widely used to model polytomous item responses. In this model, for item j with m_j response options (or categories), the probability of an examinee with trait level θ responding in a particular score category x is given by the following:

$$P_{jx}(\theta) = \frac{e^{\sum_{k=0}^{x} z_{jk}}}{\sum_{h=1}^{m_j} e^{\sum_{k=0}^{h} z_{jk}}}$$

for

$$k = 0, 1, 2, x, \ldots \, \mathrm{m_j}$$

where $z_{jk} = Da_j(\theta - \delta_{jk})$, a_j is the slope parameter, and $\delta_{jk}$is called the step difficulty parameter for category k. Different from the GR model, the response probabilities are computed directly in one step, and the step difficult parameters are not necessarily ordered. The GPC model reduces to the partial credit (PC)

model under the assumption that all items have equal slopes. Similar to the RS-GR model, some parameterizations of the GPC model decompose the parameter δ_{jk} into a location parameter (b_j) for each item and a set of category thresholds parameters (τ_{jk}). Using this parameterization, the logistic deviate is $z_{jk} = Da_j(\theta - b_j + \tau_{jk})$. To identify and estimate the parameters of the GPC model, the threshold parameter of the last category for each item is set to equal the negative of the sum of the category threshold parameters for the previous categories, that is,

$$\tau_{jm_j} = -\sum_{k=1}^{m_j-1} \tau_{jk}$$

This constrains the set of category thresholds parameters to equal 0 for each item.

Estimating the GPC Model

Program 5.10 includes the PROC MCMC code for estimating the GPC model, where $z_{jk} = Da_j(\theta - b_j + \tau_{jk})$, using the DASH data. For each DASH item, one slope parameter (a_j), one location parameter (b_j), and four category threshold parameters $(\tau_{j1}, \tau_{j2}, \tau_{j3}, \tau_{j4})$ are defined corresponding to four steps (1→2, 2→3, 3→4, 4→5).

In the code, the TAU[4,10] matrix represents the 40 threshold parameters (4 category thresholds for each item), and the Z[5] array denotes the five logistic deviate z values for each item, $z_{jk} = Da_j(\theta - \delta_{jk} + \tau_{jk})$ with D=1.0. SUM_Z[K] is an array reflecting the sum of the logistic deviates for response categories. That is, the values $\sum_{k=0}^{x} z_{jk}$ for $x = 1, 2, 3, 4, 5$. NUMER[K] are an array that reflects the numerator for response categories in the GPC model, and P[5] is an array of response probabilities for five score categories.

As for other models, the item parameters within each item are treated as a single block. Note that for the model identification, only the category threshold parameters for the first three steps are declared as parameters. The last threshold parameter within each item equals to the negative sum of the first three steps, and is computed in a BEGINNODATA–ENDNODATA block ❶. The priors for the slope parameters (a_j) are log-normal distributions, and the priors for both location and thresholds parameters are normal distributions. To calculate the likelihood of the data, the Z values are calculated first ❷ followed by the SUM_Z values. Then the numerator ❸ and denominator ❹ of the GPC model are computed and used to obtain the response probabilities ❺.

Program 5.10: PROC MCMC Code for Estimating the GPC Model

```
proc mcmc data=dash_data outpost=dash_postGPC seed=23  nbi=5000 nmc=20000
nthreads=8 monitor=(a b tau)  diagnostics=all plots=(trace autocorr);

array ix[10]; array a[10]; array b[10];
array tau[4,10] tau1_1-tau1_10 tau2_1-tau2_10 tau3_1-tau3_10
      tau4_1-tau4_10;
array z[5] z1-z5;
array sum_z[5] sum_z1-sum_z5;
array numer[5] numer1-numer5;
array p[5] p1-p5;

parms a1 1  b1 0   tau1_1 0  tau2_1 0  tau3_1 0 ;
parms a2 1  b2 0   tau1_2 0  tau2_2 0  tau3_2 0 ;
parms a3 1  b3 0   tau1_3 0  tau2_3 0  tau3_3 0 ;
parms a4 1  b4 0   tau1_4 0  tau2_4 0  tau3_4 0 ;
parms a5 1  b5 0   tau1_5 0  tau2_5 0  tau3_5 0 ;
parms a6 1  b6 0   tau1_6 0  tau2_6 0  tau3_6 0 ;
parms a7 1  b7 0   tau1_7 0  tau2_7 0  tau3_7 0 ;
parms a8 1  b8 0   tau1_8 0  tau2_8 0  tau3_8 0 ;
parms a9 1  b9 0   tau1_9 0  tau2_9 0  tau3_9 0 ;
parms a10 1 b10 0  tau1_10 0 tau2_10 0 tau3_10 0 ;

beginnodata; ❶
  tau4_1=-(tau1_1+tau2_1+tau3_1);
  tau4_2=-(tau1_2+tau2_2+tau3_2);
  tau4_3=-(tau1_3+tau2_3+tau3_3);
  tau4_4=-(tau1_4+tau2_4+tau3_4);
  tau4_5=-(tau1_5+tau2_5+tau3_5);
  tau4_6=-(tau1_6+tau2_6+tau3_6);
  tau4_7=-(tau1_7+tau2_7+tau3_7);
  tau4_8=-(tau1_8+tau2_8+tau3_8);
  tau4_9=-(tau1_9+tau2_9+tau3_9);
  tau4_10=-(tau1_10+tau2_10+tau3_10);
endnodata;

prior a: ~ lognormal(0, var=25);
prior b: ~ normal(0, var=25);
prior tau1_1-tau1_10 ~ normal(0, var=25);
prior tau2_1-tau2_10 ~ normal(0, var=25);
prior tau3_1-tau3_10 ~ normal(0, var=25);
random theta ~ normal(0, var=1) subject=_obs_;

llike=0;
do j=1 to 10;
   z[1]=0;
   sum_z[1]=0;
   numer[1]=1;
   do k=2 to 5; ❷
      z[k]=a[j]*(theta - b[j] + tau[k-1,j]);
      sum_z[k] = sum_z[k-1]+z[k];
      numer[k] = exp(sum_z[k]); ❸
   end;
   denom = sum (of numer1-numer5); ❹

   do k=1 to 5;
      p[k]=numer[k]/denom; ❺
   end;

   llike=llike+log(p[ix[j]]);
end;
model general(llike);
run;
```

Output from Estimating the GPC Model

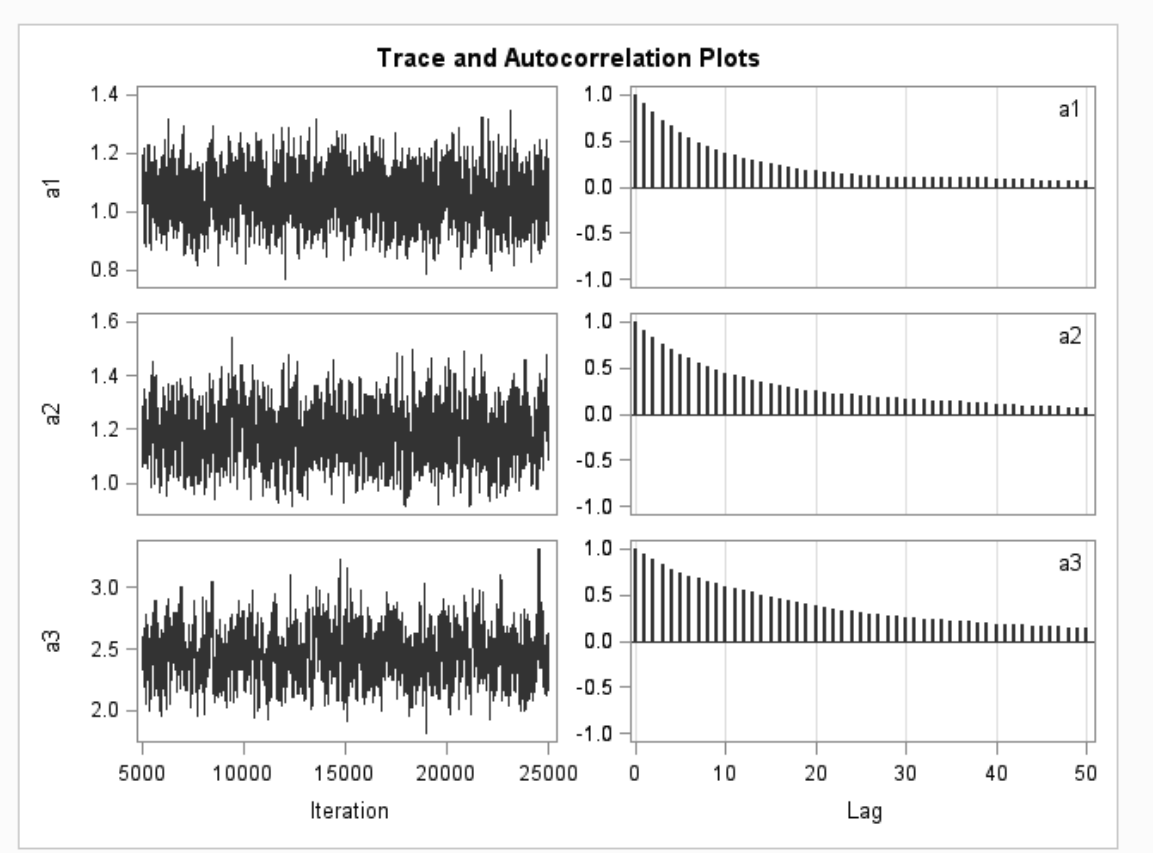
Output 5.25 shows the summary statistics for selected item parameters from the GPC model. As previously done, selected results were generated with use of the autocall macro %POSTSUM because there are a large number of item parameters. The slope estimates vary across items, which indicates that the GPC model may be more appropriate for the DASH item responses than the PC model (a common slope model). The locations are all greater than 0 except for the last item. The four threshold parameter estimates (means) within each item are not ordered as in the GR model. Note that the item parameters for all items are presented in the section comparing Bayesian and MML estimation methods.

Output 5.25: Posterior Summary Statistics for Selected Item Parameters—GPC Model

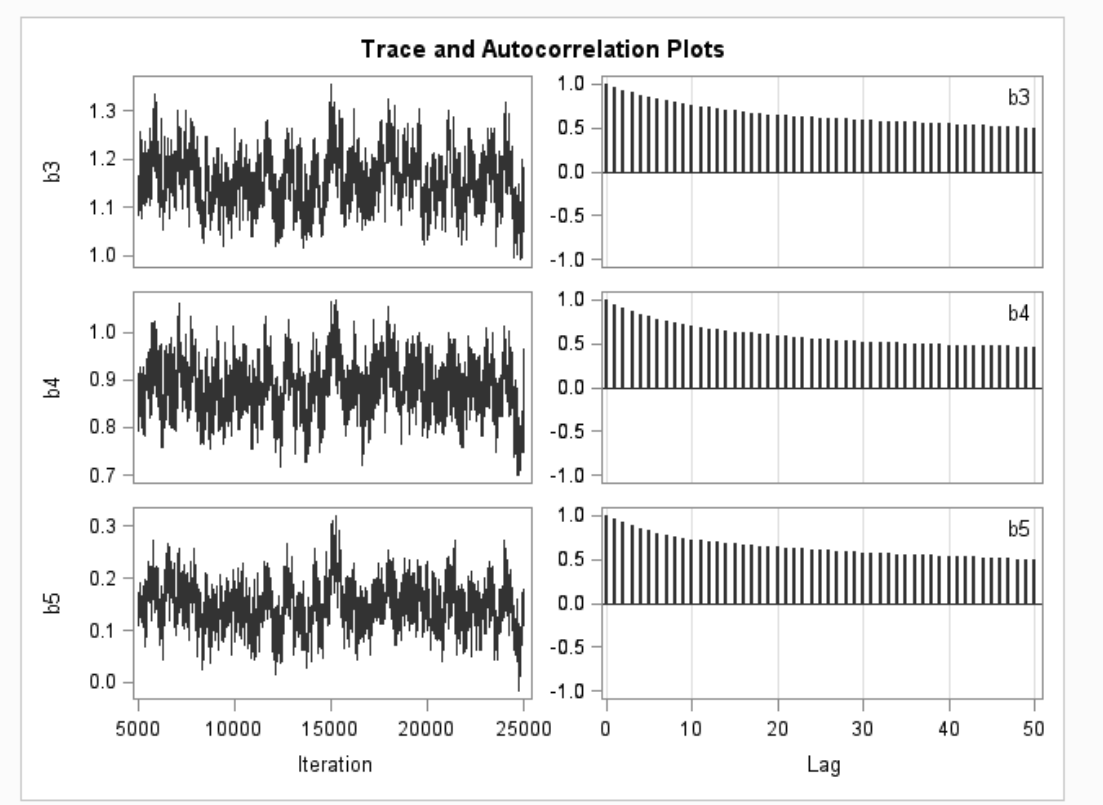
Parameter	N	Mean	StdDev	P25	P50	P75
a1	20000	1.04879	0.08073	0.99416	1.04560	1.10107
b1	20000	1.64950	0.09013	1.58597	1.64806	1.70996
tau1_1	20000	0.36402	0.11425	0.29360	0.36451	0.44214
tau2_1	20000	0.73499	0.13863	0.64057	0.73408	0.83004
tau3_1	20000	-0.52865	0.18589	-0.64987	-0.52218	-0.40281
tau4_1	20000	-0.57035	0.23076	-0.72798	-0.57477	-0.42193
a5	20000	1.60720	0.10984	1.53240	1.60257	1.68136
b5	20000	0.14750	0.04226	0.11941	0.14764	0.17413
tau1_5	20000	0.77014	0.06287	0.72784	0.76937	0.81093
tau2_5	20000	0.35864	0.06558	0.31384	0.35782	0.40238
tau3_5	20000	-0.46620	0.07002	-0.51208	-0.46568	-0.41854
tau4_5	20000	-0.66257	0.07009	-0.70916	-0.66374	-0.61597
a7	20000	2.16691	0.15452	2.06466	2.15596	2.26712
b7	20000	0.67350	0.04557	0.64123	0.67338	0.70408
tau1_7	20000	0.93778	0.04832	0.90490	0.93675	0.97123
tau2_7	20000	0.32349	0.04924	0.28968	0.32323	0.35723
tau3_7	20000	-0.81321	0.08302	-0.86689	-0.81164	-0.75563
tau4_7	20000	-0.44807	0.08457	-0.50609	-0.44927	-0.39369
a10	20000	1.33277	0.09502	1.26708	1.32758	1.39358
b10	20000	-0.43146	0.04684	-0.46212	-0.43162	-0.40106
tau1_10	20000	1.10249	0.09134	1.04054	1.09687	1.16473
tau2_10	20000	0.27979	0.07779	0.22923	0.27937	0.33226
tau3_10	20000	-0.65578	0.08274	-0.70918	-0.65135	-0.60104
tau4_10	20000	-0.72650	0.07602	-0.77804	-0.72621	-0.67262

Outputs 5.26 to 5.28 illustrate the trace and autocorrelation plots for selected item parameters. The plots that are presented represent typical plots across the set of items. You can see that the chains for the slope parameters (a_j) and the category thresholds parameters (τ_{jk}) all reflect chains mixing well with low dependence among sampled values. This is in contrast with the location parameters (b_j), which exhibit reasonably good mixing but also moderately high dependence among sampled values.

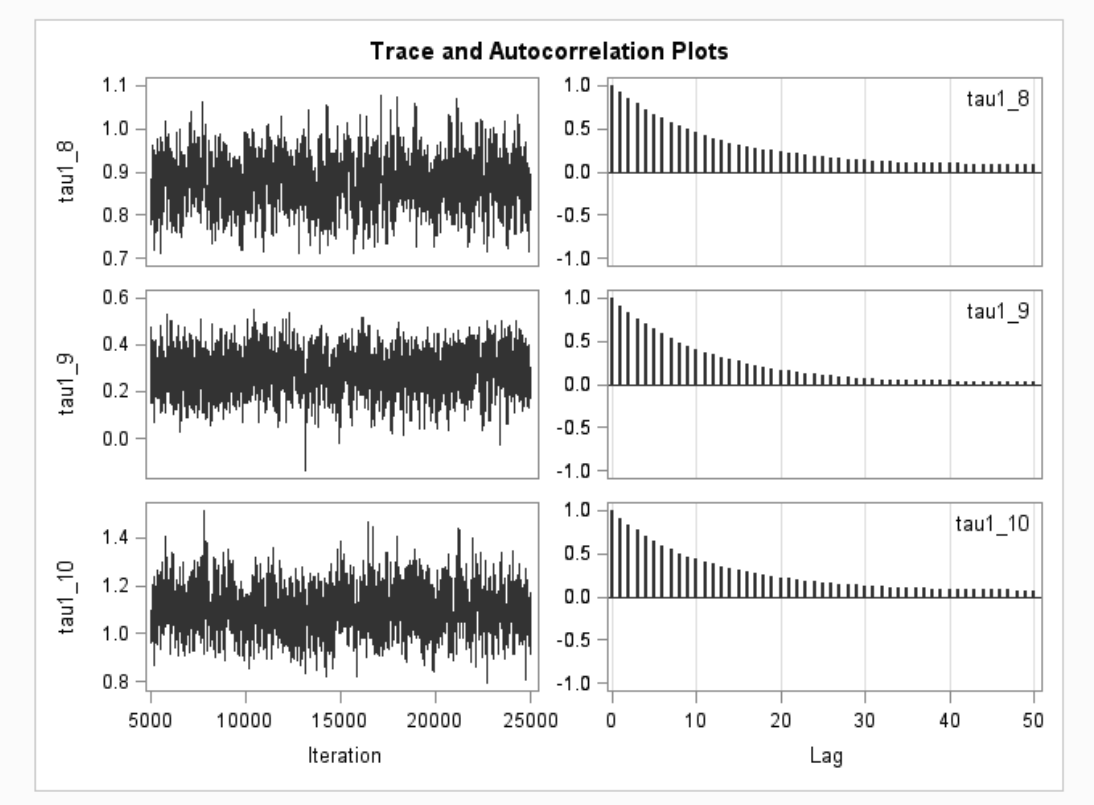
Output 5.26: Trace and Autocorrelation Plots for Selected Item Slope Parameters—GPC Model



Output 5.27: Trace and Autocorrelation Plots for Selected Item Location Parameters—GPC Model



Output 5.28: Trace and Autocorrelation Plots for Item Tau Parameters—GPC Model



Comparison of Results Based on MML Estimation

To compare the Bayesian estimation results with traditional MML estimation results, we estimated different models with the DASH data using the computer program IRTPRO (Cai, Thissen, & du Toit, 2011). Tables 5.1, 5.2, and 5.3 present the comparison of Bayesian and MML results for the GR, NR, and GPC models. Notably, the procedure PROC IRT (SAS, 2014, Chapter 53) could be used for some of these comparisons. But it could not be used universally for all comparisons in this book, so this SAS procedure was not used.

Across the tables, you can see that the Bayesian and MML estimates are very similar for the different models. The one exception may be the comparison for the NR model (see Table 5.2), where the slope parameters are more similar than the intercept parameters between two estimation methods, and not unexpectedly, the differences for more extreme values tend to be larger.

Table 5.1: Comparison of Bayesian and MML Item Parameter Estimates for the GR Model

Estimation Method	a	b1	b2	b3	b4
Item 1					
Bayesian	1.83	0.50	1.23	2.17	2.97
MML	1.82	0.49	1.20	2.16	2.95
Item 2					
Bayesian	2.07	0.61	1.23	1.89	2.65
MML	2.06	0.60	1.22	1.88	2.65
Item 3					
Bayesian	3.49	0.14	0.79	1.65	2.14
MML	3.49	0.13	0.78	1.63	2.12
Item 4					
Bayesian	2.50	−0.39	0.45	1.29	2.13
MML	2.49	−0.41	0.44	1.28	2.12
Item 5					
Bayesian	2.79	−0.84	−0.19	0.50	1.08
MML	2.79	−0.85	−0.20	0.49	1.07
Item 6					
Bayesian	3.21	−0.09	0.61	1.36	1.84
MML	3.19	−0.10	0.59	1.35	1.82
Item 7					
Bayesian	3.33	−0.39	0.34	1.18	1.54
MML	3.32	−0.40	0.33	1.17	1.53
Item 8					
Bayesian	3.18	−0.79	−0.10	0.60	1.16
MML	3.18	−0.78	−0.11	0.59	1.15
Item 9					
Bayesian	2.63	0.53	1.11	1.71	2.08
MML	2.62	0.51	1.09	1.70	2.06
Item 10					
Bayesian	2.32	−1.59	−0.74	0.06	0.69
MML	2.33	−1.59	−0.74	0.04	0.68

Table 5.2: Comparison of Bayesian and MML Item Parameter Estimates for the NR Model

Estimation	a1	a2	a3	a4	a5	c1	c2	c3	c4	c5
Item 1										
Bayesian	0	1.02	2.31	3.15	4.64	0	−1.30	−2.36	−4.35	−7.30
MML	0	1.00	2.30	3.18	4.68	0	−1.32	−2.40	−4.44	−7.46
Item 2										
Bayesian	0	1.35	2.56	3.32	5.35	0	−1.61	−2.83	−4.26	−8.02
MML	0	1.33	2.57	3.36	5.44	0	−1.62	−2.89	−4.35	−8.27
Item 3										
Bayesian	0	2.70	5.55	7.42	9.16	0	−0.73	−2.77	−6.23	−9.25
MML	0	2.73	5.74	7.80	9.61	0	−0.79	−2.99	−6.75	−9.90
Item 4										
Bayesian	0	1.64	3.27	4.59	7.80	0	0.33	−0.42	−2.22	−7.50
MML	0	1.62	3.22	4.58	8.00	0	0.29	−0.47	−2.33	−7.97
Item 5										
Bayesian	0	1.29	3.33	4.90	6.88	0	0.79	1.23	0.30	−1.21
MML	0	1.24	3.30	4.85	6.83	0	0.77	1.16	0.31	−1.34
Item 6										
Bayesian	0	2.52	4.89	6.40	8.01	0	−0.13	−1.50	−3.98	−6.21
MML	0	2.52	4.93	6.48	8.10	0	−0.18	−1.62	−4.17	−6.47
Item 7										
Bayesian	0	2.10	4.59	6.26	9.54	0	0.55	−0.24	−2.91	−6.46
MML	0	2.09	4.60	6.32	9.63	0	0.51	−0.34	−3.06	−6.76
Item 8										
Bayesian	0	1.70	3.87	5.52	8.41	0	1.07	1.31	0.15	−2.45
MML	0	1.69	3.84	5.47	8.33	0	1.04	1.24	0.04	−2.59
Item 9										
Bayesian	0	2.23	3.24	4.57	5.73	0	−1.70	−2.86	−5.25	−6.65
MML	0	2.23	3.26	4.59	5.77	0	−1.74	−2.94	−5.43	−6.80
Item 10										
Bayesian	0	0.77	2.02	3.63	5.38	0	1.39	2.37	2.13	1.54
MML	0	0.77	2.00	3.59	5.33	0	1.38	2.32	2.05	1.43

Table 5.3: Comparison of Bayesian and MML Item Parameter Estimates for the GPC Model

Estimation	a	b	τ1	τ2	τ3	τ4
Item 1						
Bayesian	1.04	1.64	0.36	0.74	−0.53	−0.57
MML	1.09	1.59	0.36	0.70	−0.50	−0.57
Item 2						
Bayesian	1.18	1.55	0.20	0.53	−0.13	−0.60
MML	1.24	1.50	0.21	0.50	−0.13	−0.58
Item 3						
Bayesian	2.46	1.15	0.84	0.40	−0.58	−0.66
MML	2.56	1.12	0.82	0.38	−0.56	−0.65
Item 4						
Bayesian	1.61	0.88	1.08	0.41	−0.48	−1.01
MML	1.67	0.86	1.06	0.39	−0.45	−0.99

Estimation	a	b	τ1	τ2	τ3	τ4
Item 5						
Bayesian	1.61	0.15	0.77	0.36	−0.47	−0.66
MML	1.66	0.14	0.75	0.35	−0.45	−0.65
Item 6						
Bayesian	2.09	0.92	0.82	0.32	−0.56	−0.59
MML	2.17	0.89	0.80	0.31	−0.54	−0.57
Item 7						
Bayesian	2.17	0.67	0.94	0.32	−0.81	−0.45
MML	2.25	0.65	0.91	0.31	−0.78	−0.44
Item 8						
Bayesian	1.94	0.24	0.88	0.35	−0.49	−0.74
MML	2.01	0.23	0.86	0.34	−0.48	−0.72
Item 9						
Bayesian	1.54	1.29	0.28	0.30	−0.53	−0.06
MML	1.60	1.24	0.29	0.30	−0.52	−0.07
Item 10						
Bayesian	1.33	−0.43	1.10	0.28	−0.66	−0.73
MML	1.38	−0.42	1.08	0.27	−0.64	−0.71

Graphs of Item Category Response Functions

Differing from the dichotomous IRT models, polytomous items have more than one response curves, one for each category. However, as illustrated in Chapter 4 to plot ICCs for binary response items, the category response curves can be drawn for each polytomous item. Program 5.11 provides the commands for drawing the item category response curves for the DASH items based on the GR model. The plots are shown in Output 5.29.

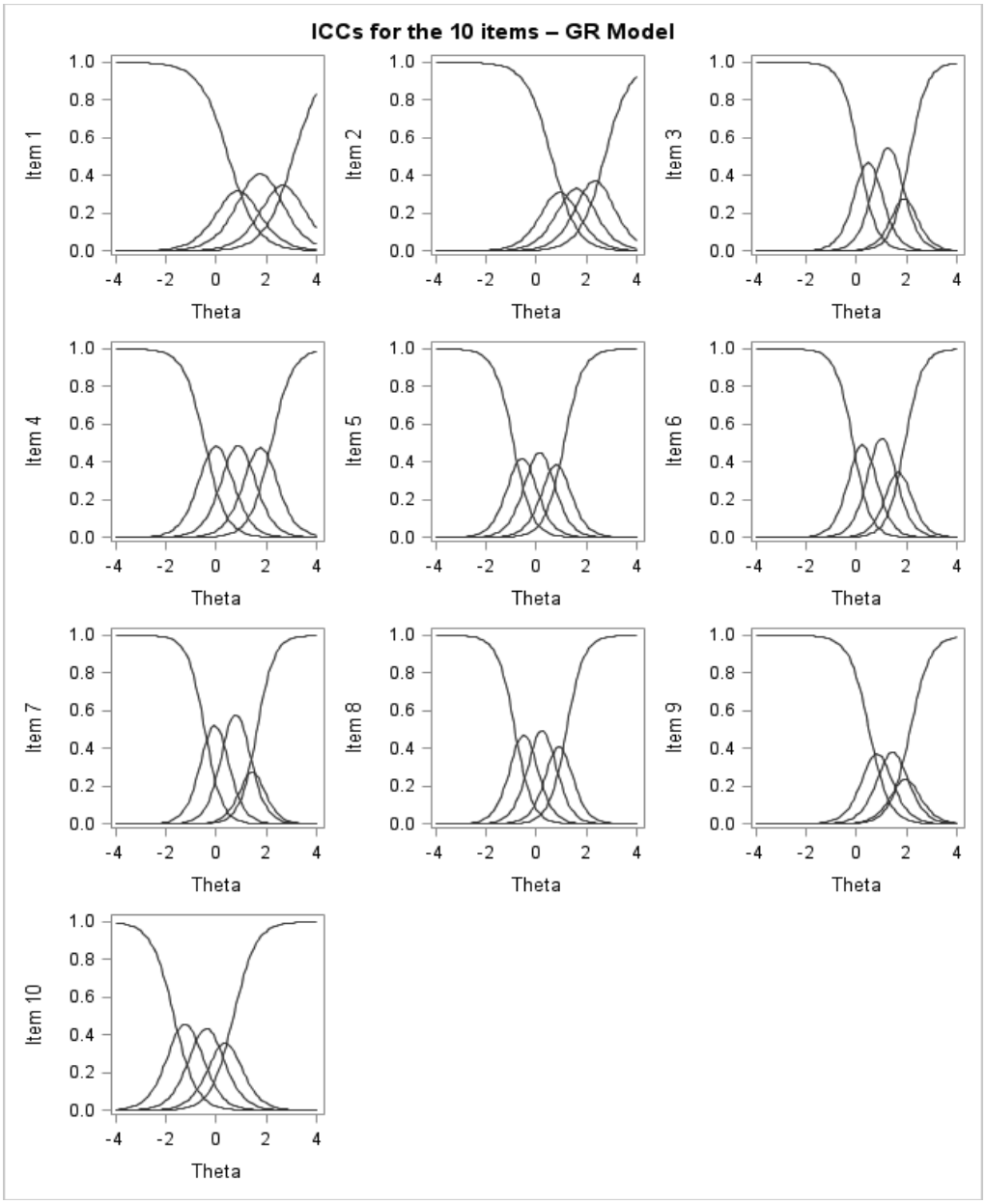
In the program, point estimates for the item parameters are first obtained using PROC MEANS ❶. These point estimates are saved in a data set and used to generate a data set of response category probabilities for each item ❷. In this data set, the rows reflect discrete values of θ from –4 to 4 in increments of .1. Given this data set, a graphics template can be defined to plot the category curves using a SERIESPLOT command for each response category and for each item ❸. A LAYOUT LATTICE command ❹ is used to graph all the category plots in a single graph with common scales for the *x*-axes and *y*-axes.

Program 5.11: Graphing Item Category Response Curves for Polytomous IRT models—GR Model

```
%let nitems=10; /* number of items */
%let ncats=5;   /* number of response categories */
%let nthres=4;  /* number of threshold parameters */

/* save point estimates for item parameters*/
proc means data=dash_postout noprint; ❶
     var a1-a&nitems b1_1-b1_&nitems b2_1-b2_&nitems b3_1-b3_&nitems
         b4_1-b4_&nitems;
     output out=means_gr  mean=;
run;

/* create dataset of expected item response probabilities across theta */
data plotdata_gr; ❷
     set means_gr;
     array a[&nitems] a1-a&nitems;
     array b[&nthres,&nitems] b1_1-b1_&nitems  b2_1-b2_&nitems
           b3_1-b3_&nitems b4_1-b4_&nitems;
```

```
     array p_star[&nitems,&nthres];
     array p[&nitems,&ncats];

     retain group(0);
     do theta=-4 to 4 by .1;
        group=group+1;
        do j=1 to &nitems;
           do k=1 to 4;
              p_star[j,k]=logistic(a[j]*(theta-b[k,j]));
           end;
           p[j,1]=1-p_star[j,1];
           do k=2 to 4;
              p[j,k]=p_star[j,(k-1)]-p_star[j,k];
           end;
           p[j,5]=p_star[j,4];
        end;
     output;
     end;
run;

/* define a graph template for category icc plots using seriesplots */
proc template;
     define statgraph iccplots;
     begingraph  / designwidth=630px designheight=840px;
     entrytitle "Category ICCs for the &nitems items - GR Model";
     layout lattice / columns=3 rows=4 rowgutter=10 columngutter=10; ❹
     layout overlay /
       xaxisopts=(label="Theta" linearopts=(viewmin=-4 viewmax=4))
       yaxisopts=(label="Item 1" linearopts=(viewmin=0  viewmax=1));
       seriesplot x=theta y=p1; ❸
       seriesplot x=theta y=p2;
       seriesplot x=theta y=p3;
       seriesplot x=theta y=p4;
       seriesplot x=theta y=p5;
       endlayout;

       *Repeat for Item2~Item9;

     layout overlay /
       xaxisopts=(label="Theta" linearopts=(viewmin=-4 viewmax=4))
       yaxisopts=(label="Item 10" linearopts=(viewmin=0  viewmax=1));
       seriesplot x=theta y=p46;
       seriesplot x=theta y=p47;
       seriesplot x=theta y=p48;
       seriesplot x=theta y=p49;
       seriesplot x=theta y=p50;
       endlayout;
    endlayout;
    endgraph;
end;
run;

/* produce the plots using the template */
proc sgrender data=plotdata_gr template=iccplots;
run;
```

Output 5.29: Category ICC Plots for the 10 DASH Items



Graphs of Test Information Functions

A program to illustrate the graphing of test information functions for the GR model is also included on the SAS Press authors' web page for this book (Program 5.12). Test information functions are computed by summing up the information about θ provided by each item. These functions are useful in the test development process for exploring the degree of precision of a test or differing sets of items across the θ scale (see, for example, Hambleton & Swamination 1985).

Index

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

R

S

T

U

V

W

Y

Z

About This Book

Purpose

Assessment has played and continues to play an integral role in society and in a culture of inquiry. Accordingly, numerous instruments have been developed over the years to measure many different characteristics of individuals in the educational, psychological, health, social, and behavioral sciences. These instruments have been used to measure the status or level of different characteristics in individuals, as well as to capture differences in these characteristics across individuals.

To support the development of instruments and measure characteristics in individuals, test theories have been used to describe how inferences, predictions, or estimates of a particular characteristic, trait, or ability of a person may be made from responses to items. Test theories such as classical test theory (CTT) and item response theory (IRT) provide models for explaining test performance in relation to variables that are assumed to influence behavior. They provide methods for selecting items, evaluating tests or scales, obtaining scores, and quantifying sources of errors in the measurement process. In the early decades of the 21st century, psychometricians have favored IRT, as opposed to CTT, in scale development and assessment applications.

IRT models consist of a family of mathematical models that predict item performance by using parameters that characterize both the items in an instrument and the respondents. Although numerous methods for estimating the parameters of IRT models exist, interest in estimating the parameters using Bayesian methods has grown tremendously. In part, this growth is due to the appeal of the Bayesian paradigm among psychometricians and statisticians, as well as to the advantages of these methods with small sample sizes, more complex or highly parameterized models (such as multidimensional IRT models), and interest in simultaneous estimation of item and person parameters. In contrast to traditional approaches for estimating model parameters, a Bayesian paradigm considers model parameters to be random variables and uses Bayes theorem to obtain distributions for the model parameters.

Recently, routines have become available in the SAS system software to implement general Bayesian analyses (PROC MCMC). Use of the SAS system for Bayesian analysis of IRT models has several significant advantages over other available programs: (1) It is commonly used by researchers across disciplines; (2) it provides a robust programming language that extends the capability of the program—in particular, the capability for model checking; and (3) it shows increased performance and efficiency through the use of parallel processing. The purpose of this book is to illustrate Bayesian estimation and evaluation of a variety of IRT models that are of interest to psychometricians, scale developers, and practitioners responsible for implementing assessment programs.

Is This Book for You?

This book is designed for psychometricians, scale developers, and individuals interested in applications of Bayesian methods and model checking of IRT models for the development and analysis of assessment data. In addition, you may find this book useful if you are interested in applications with small sample sizes, applications requiring more complex or highly parameterized models (such as multidimensional IRT models), and simultaneous estimation of item and person parameters.

Prerequisites

Although introductory material related to IRT and Bayesian analysis is included in the book, some prior knowledge about these topics is necessary for you to better understand and implement the procedures in your own assessment applications. For detail and more technical information, see the references provided.

Scope of This Book

This book is example-driven, with sections presenting code for estimating particular models and results that illustrate convergence diagnostics and inferences for parameters, as well as results that can be used specifically by scale developers—for example, plotting item response functions.

Chapters 1 and 2 provide introductory treatments of IRT and Bayesian analysis. These chapters are not meant to provide detailed information for individuals interested in learning about these topics. Rather, these chapters provide supporting documentation for the subsequent chapters that describe Bayesian estimation of different IRT models with the use of SAS PROC MCMC. Chapter 1, on IRT, motivates the use of the different models and provides the model formulations that will be used in subsequent chapters. Chapter 2, on Bayesian analysis, provides supporting documentation for terms and concepts integral to the use of PROC MCMC and the subsequent examples.

Chapter 3 presents an overview of the PROC MCMC syntax commands and a template for the estimation of IRT models, but focuses on syntax that is most relevant to the estimation of IRT models. As a result, there is some duplication of this material with other SAS documentation for Bayesian analysis and the use of PROC MCMC. However, presentation of PROC MCMC in this book provides a self-contained guide to the use of PROC MCMC and Bayesian analysis of IRT models.

Chapters 4 to 6 provide detailed presentations of the syntax used to estimate IRT models for different types of assessment applications. In these chapters, the IRT models and PROC MCMC syntax to estimate the models are discussed. Examples then illustrate the generation of convergence diagnostics, use of different prior distributions, and summary statistics from the posterior distributions for parameters. In some cases, item responses from data sets that are commonly analyzed in the literature are used. In other cases, item responses are simulated so that direct comparisons between estimated parameters and population parameters can be made. The comparison between estimated parameters and parameters used to obtain simulated item responses facilitates the evaluation of using PROC MCMC to estimate the models. In addition, comparisons of model parameter estimates derived from using PROC MCMC with other available software are also presented.

Specifically, Chapter 4 illustrates estimation of IRT models for dichotomously scored items, and Chapter 5 illustrates estimation of models for polytomously scored items. For both of these chapters, the models that are discussed assume that a single person parameter determines the performance of respondents (unidimensional IRT models).

Chapter 6 illustrates extensions of IRT models to more highly parameterized models, including multidimensional IRT models, models that account for any dependence between sets of items (testlet IRT models), multilevel IRT models, and applications to differential item functioning (DIF). Multidimensional IRT models are considered when more than one person parameter is assumed. Testlet IRT models may be used for testing applications in which individuals respond to a set of items based on a single or common stimulus (*testlet*). Multilevel or hierarchical models may be estimated in testing applications in which there exists a nested structure of item responses within clusters of persons or within clusters of items. Finally, models for evaluating differential item functioning and the use of mixture IRT models are considered to examine whether items function differently across subpopulations of examinees (for example, masters versus nonmasters or males versus females). In the case of mixture IRT models, these models assume that the population of examinees consists of qualitatively different subpopulations or latent classes that are unknown. These models provide a mechanism for clustering examinees to better understand any heterogeneity in item responses.

Although the focus is often on estimating IRT model parameters, it is equally important to evaluate the fit of a particular IRT model to the item responses. When a model doesn't fit the data, the validity of any inferences for the model parameter estimates is threatened. The SAS system and PROC MCMC affords advantages over other available software for Bayesian analysis of IRT models. For evaluating model-data-fit, in particular, SAS provides a robust programming language and built-in statistical and graphical tools. The programming language and the tools expand the capability of the program beyond estimating the model to computing many different types of statistics for comparing competing models and evaluating model fit.

The topics of model comparison and model checking in a Bayesian analysis are discussed in Chapters 7 and 8 of this book. Chapter 7 considers methods for comparing competing models so that a preferred model can be identified. Methods based on information criteria indices and the Bayes Factor are considered. Chapter 8 begins with a discussion of item and person fit in the context of IRT models and a description of a popular model checking tool in Bayesian analysis—posterior predictive model checking (PPMC). Then possible discrepancy statistics for a PPMC analysis of IRT models are discussed, followed by examples illustrating the use of SAS procedures and programming commands to perform a PPMC analysis. The use of output in the form of tables and graphs is also included to illustrate the capabilities of a PPMC analysis in SAS, as well as the different types of output that can be produced to report a PPMC analysis.

About the Examples

Software Used to Develop the Book's Content

PROC MCMC statements and examples using PROC MCMC in this book are based on SAS 9.4 and SAS/STAT 13.2.

Example Programming Code and Data

Examples are provided to estimate IRT models for dichotomously and polytomously scored items, as well as more complex IRT models, such as multidimensional and hierarchical models.

You can access the example programs and data sets for this book by linking to its author page at http://support.sas.com/publishing/authors/stone.html. Select the name of the author. Then look for the cover thumbnail of this book, and select *Example Code* and *Data* to display the SAS programs that are included in this book.

For an alphabetical listing of all books for which example code and data are available, see http://support.sas.com/bookcode. Select a title to display the book's example code.

If you cannot attain access to the code through the website, send email to saspress@sas.com.

Output and Graphics

All output was derived from programs using PROC MCMC or from other SAS programs. All graphics were obtained with the use of SAS Graphics Template Language with default SAS Output Delivery System settings.

Additional Help

Although this book provides examples of many of the IRT analyses used by psychometricians, questions specific to your aims and issues may arise. To fully support you, SAS Institute and SAS Press offer you the following help resources:

- For questions about topics covered in this book, contact the author through SAS Press:
  - Send questions by email to saspress@sas.com. Include the book title in your correspondence.
  - Submit feedback on the author's page at http://support.sas.com/author_feedback.
- For questions about topics in or beyond the scope of this book, post queries to the relevant SAS Support Communities at https://communities.sas.com/welcome.
- SAS Institute maintains a comprehensive website with up-to-date information. One page that is particularly useful to both the novice and the seasoned SAS user is its Knowledge Base. Search for relevant notes in the *Samples and SAS Notes* section of the Knowledge Base at http://support.sas.com/resources.
- Registered SAS users or their organizations can access SAS Customer Support at http://support.sas.com. Here you can pose specific questions to SAS Customer Support. Under *Support*, click *Submit a Problem*. You will need to provide an email address to which replies can be sent, identify your organization, and provide a customer site number or license information. This information can be found in your SAS logs.

Keep in Touch

We look forward to hearing from you. We invite questions, comments, and concerns. If you want to contact us about a specific book, please include the book title in your correspondence.

Contact the Author through SAS Press

- By email: saspress@sas.com
- Via the web: http://support.sas.com/author_feedback

Purchase SAS Books

For a complete list of books available through SAS, visit sas.com/store/books.

- Phone: 1-800-727-0025
- E-mail: sasbook@sas.com

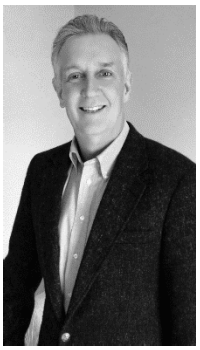
Subscribe to the SAS Training and Book Report

Receive up-to-date information about SAS training, certification, and publications via email by subscribing to the SAS Training & Book Report monthly eNewsletter. Read the archives and subscribe today at http://support.sas.com/community/newsletters/training!

Publish with SAS

SAS is recruiting authors! Are you interested in writing a book? Visit http://support.sas.com/saspress for more information.

About These Authors



Clement A. Stone is a professor in the Research Methodology program at the University of Pittsburgh School of Education. He is an expert in psychometrics, including educational and psychological instrument development and validation, item response theory (IRT) models and applications, and Bayesian analysis of IRT models. Also an expert in SAS software, he has used SAS extensively in research utilizing simulation methods, and he instructs graduate students in the use of SAS. He applies IRT in the development and validation of educational, psychological, and behavioral assessments, including a research focus on educational achievement, critical thinking, psychosocial stress, communication outcomes, risk factors for addiction, and physical disability. He has published numerous articles, and he coauthored a chapter on the development and analysis of performance assessments in Educational Measuremen. In addition to publishing in and reviewing for numerous prominent journals, he has served on the editorial boards for the Journal of Educational Measurement, Applied Measurement in Education, Educational and Psychological Measurement, and the American Educational Research Journal. Stone holds a Ph.D. in research methods, measurement, and statistics from the University of Arizona.

Xiaowen Zhu is an associate professor in the Department of Sociology and a research fellow in the Institute for Empirical Social Science Research, Xi'an Jiaotong University, China. Formerly, she was a psychometrician for the Data Recognition Corporation in the United States, working on large-scale state educational assessments. She has a strong theoretical background, operational experience, and research experience in psychometrics and applied statistics. She is especially adept at Bayesian analysis of IRT models and has published journal articles on this subject. In addition, she has used SAS throughout her research career for general statistical analysis and simulation studies. Zhu holds a Ph.D. in research methodology from the University of Pittsburgh.

Learn more about these authors by visiting their author pages, where you can download free book excerpts, access example code and data, read the latest reviews, get updates, and more:
http://support.sas.com/publishing/authors/stone.html
http://support.sas.com/publishing/authors/zhu.html