JMASM ALGORITHMS & CODE
JMASM28: Gibbs Sampling for 2PNO Multi-unidimensional Item Response Theory Models (Fortran)

Yanyan Sheng  Todd C. Headrick
Southern Illinois University Carbondale

A Fortran 77 subroutine is provided for implementing the Gibbs sampling procedure to a multi-unidimensional IRT model for binary item response data with the choice of uniform and normal prior distributions for item parameters. In addition to posterior estimates of the model parameters and their Monte Carlo standard errors, the algorithm also estimates the correlations between distinct latent traits. The subroutine requires the user to have access to the IMSL library. The source code is available at http://www.siu.edu/~epsel/sheng/Fortran/MUIRT/GSMU2.FOR. An executable file is also provided for download at http://www.siu.edu/~epsel/sheng/Fortran/MUIRT/EXAMPLE.zip to demonstrate the implementation of the algorithm on simulated data.

Key words: multi-unidimensional IRT model, two-parameter normal ogive model, MCMC, Gibbs sampling, Fortran.

Introduction
Modeling the interaction of a person’s trait and the test at the item level for binary response data, the conventional item response theory (IRT) models rely on a strong assumption of unidimensionality. That is, each test item is designed to measure some facet of a unified latent trait. However, psychological processes have consistently been found to be more complex and an increasing number of educational measurements assess a person on more than one latent trait. In the situations when a test consists of several subtests with each measuring one latent trait, the multi-unidimensional IRT models have been found to be more appropriate than the unidimensional models (Sheng & Wikle, 2007), as they allow inferences to be made about a person for each distinct trait being measured.

For the two-parameter normal ogive (2PNO) multi-unidimensional model, the probability of person \( i \) obtaining a correct response for item \( j \) in subtest \( v \), where \( i = 1, \ldots, n \), \( j = 1, \ldots, k_v \), \( v = 1, \ldots, m \), and \( K = \sum_v k_v \), is defined as

\[
P(y_{ij} = 1) = \Phi(\alpha_{vj} \theta_{ivj} - \gamma_v)
\]

(1)

\[
= \int_{-\infty}^{\alpha_{vj} \theta_{ivj} - \gamma_v} \frac{1}{\sqrt{2\pi}} e^{-\frac{t^2}{2}} dt
\]

(e.g., Lee, 1995; Sheng & Wikle, 2007), where \( \alpha_{vj} \) and \( \theta_{ivj} \) are scalar parameters representing the item discrimination and the continuous person trait in the \( v \)th latent dimension, and \( \gamma_v \) is a scalar parameter indicating the location in that dimension where the item provides maximum information. To estimate both item and person parameters simultaneously, Markov
chain Monte Carlo (MCMC; e.g., Chib &
Greenberg, 1995) techniques are used to
summarize the posterior distributions that arise
in the context of the Bayesian prior-posterior
framework (Carlin & Louis, 2000; Chib &
Greenberg, 1995; Gelfand & Smith, 1990;
Gelman, Carlin, Stern, & Rubin, 2003; Tanner &
sampling (Casella & George, 1992; Gelfand &
Smith, 1990; Geman & Geman, 1984), an
MCMC algorithm, to the 2PNO multi-
unidimensional model and illustrated the model parameterization by adopting non-informative
priors for item parameters.

Due to the reasons that informative
priors are desirable in some applications in the
Bayesian framework, and MCMC is
computational demanding (see Sheng &
Headrick, 2007, for a description of the
problems), this study focuses on using Fortran,
the fastest programming language for numerical
computing (Brainerd, 2003) to implement the
procedure. In particular, the paper provides a
Fortran subroutine that obtains the posterior
estimates and Monte Carlo standard errors of
estimates for the item and person parameters in
the 2PNO multi-unidimensional IRT model, as
well as the posterior estimates of the correlations
between the distinct latent traits. The subroutine
allows the user to specify non-informative and
informative priors for item parameters.

Methodology
The Gibbs Sampling Procedure
To implement Gibbs sampling to the
2PNO multi-unidimensional IRT model defined
in (1), a latent continuous random variable Z is
introduced so that $Z_{it} \sim N(\theta_{it} - \gamma_{it}, 1)$
(Albert, 1992; Lee, 1995; Tanner & Wong,
1987). Next, denote each person’s latent traits
for all items as $\Theta_i = (\theta_{i1}, ..., \theta_{im})'$, which is
assumed to have a multivariate normal (MVN)
distribution, $\Theta_i \sim N_m(0, \Sigma)$, where $\Sigma$ is a
correlation matrix, and $\rho_{st}$ is the correlation
between $\theta_{st}$ and $\theta_{st}, s \neq t$, on the off diagonals. It
may be noted that the unidimensional IRT model is a special case of the multi-unidimensional
model where $\rho_{st} = 1$ for all $s, t$. Then, an
unconstrained covariance matrix $\Sigma'$ is introduced
(Lee, 1995), where $\Sigma' = \left[ \frac{\sigma_{ij}}{1} \right]_{m \times m}$, so that the
correlation matrix $\Sigma$ can be easily transformed
from $\Sigma'$ using $\rho_{st} = \frac{\sigma_{st}}{\sigma_s \sigma_t}$ ($s \neq t$). A non-
informative prior is assumed for $\Sigma'$ so that
$p(\Sigma') \propto \frac{1}{2}^{n+1}$. Hence, with prior
distributions assumed for $\xi_{ij}$, where
$\xi_{ij} = (\alpha_{ij}, \gamma_{ij})'$, the joint posterior distribution
for $(\theta, \xi, Z, \Sigma')$ is

$$p(\theta, \xi, Z, \Sigma') \propto f(y | Z)p(Z | \theta, \xi)p(\xi)p(\theta | \Sigma)p(\Sigma').$$

where $f(y | Z)$ is the likelihood function.

With non-informative priors for $\alpha_{ij}$
and $\gamma_{ij}$ so that $\alpha_{ij} > 0$ and $p(\gamma_{ij}) \propto 1$, the full
conditional distributions of $Z_{ij}$, $\Theta_i$, $\xi_{ij}$, and $\Sigma'$
can be derived in closed forms as follows:

\begin{align*}
Z_{ij} \mid \cdot & \sim \begin{cases} 
N_{(0, \sigma)}(\alpha_{ij} \theta_{ij} - \gamma_{ij}, 1), & \text{if } y_{ij} = 1 \\
N_{(- \sigma, \sigma)}(\alpha_{ij} \theta_{ij} - \gamma_{ij}, 1), & \text{if } y_{ij} = 0
\end{cases}; \\
\Theta_i \mid \cdot & \sim N_m((A'A + \Sigma)^{-1}A'B, (A'A + \Sigma)^{-1}),
\end{align*}

where $A_{(K \times m)} = \begin{pmatrix} 0 & \alpha_2 & \cdots & 0 \\ 0 & \alpha_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \alpha_m \end{pmatrix}$ and

\begin{align*}
B_{(k \times l)} = \begin{pmatrix} Z_{i1} + \gamma_1 \\ Z_{i2} + \gamma_2 \\ \vdots \\ Z_{im} + \gamma_m \end{pmatrix},
\end{align*}

\begin{align*}
\xi_{ij} \mid \cdot & \sim N_2((x_i'x_i)^{-1}x_i'Z_{ij}x_i(x_i'x_i)^{-1})I(\alpha_{ij} > 0),
\end{align*}
where \( x_v = [\theta, -1] \);
\[
\Sigma^* \| \bullet \sim W^{-1}(S^{-1}, n) \quad (6)
\]
(an inverse Wishart distribution), where
\[
S = \sum_{i=1}^n (c \theta_i)(c \theta_i)',
\]
and
\[
c = \begin{pmatrix}
\prod_j \alpha_j^{\frac{1}{\nu}} & 0 & \cdots & 0 \\
0 & \prod_j \alpha_j^{\frac{1}{\nu}} & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \cdots & \prod_j \alpha_j^{\frac{1}{\nu}}
\end{pmatrix}
\]
(see Lee, 1995 for a detailed derivation).

Alternatively, conjugate priors can be assumed for \( \alpha_j \) and \( \gamma_j \) so that
\[
\alpha_j \sim N_{(\nu, \nu)}(\mu_{\alpha}, \sigma_{\alpha}^2), \quad \gamma_j \sim N(\mu_{\gamma}, \sigma_{\gamma}^2).
\]
In this case, the full conditional distribution of \( \xi_{\nu} \) is derived to be
\[
\xi_{\nu} \| \bullet \sim N\left( (x', x + \Sigma_{k}^{-1})(x', Z_{\nu} + \Sigma_{k}^{-1}\mu_{k}), (x' x + \Sigma_{k}^{-1}) \right), \quad (\alpha_j > 0) \quad (7)
\]
where
\[
\Sigma_{k} = \begin{pmatrix}
\sigma_{\alpha}^2 & 0 \\
0 & \sigma_{\gamma}^2
\end{pmatrix}.
\]

Hence, with starting values \( \theta^{(0)}, \xi^{(0)}, \) and \( \Sigma^{(0)} \), observations \( (Z^{(l)}, \theta^{(l)}, \xi^{(l)}, \Sigma^{(l)}) \) can be simulated using the Gibbs sampler by iteratively drawing from their respective full conditional distributions specified in (3), (4), (5), and (6) (or (3), (4), (7), and (6)). In particular, to go from \( (Z^{(l-1)}, \theta^{(l-1)}, \xi^{(l-1)}, \Sigma^{(l-1)}) \) to \( (Z^{(l)}, \theta^{(l)}, \xi^{(l)}, \Sigma^{(l)}) \), it takes four transition steps:

1. Draw \( Z^{(l)} \sim \mathcal{N}(Z | \theta^{(l-1)}, \xi^{(l-1)}); \)
2. Draw \( \theta^{(l)} \sim \mathcal{N}(\theta | Z^{(l)}, \xi^{(l)}, \Sigma^{(l-1)}); \)
3. Draw \( \xi^{(l)} \sim \mathcal{N}(\xi | Z^{(l)}, \theta^{(l)}); \)
4. Draw \( \Sigma^{(l)} \sim \mathcal{N}(\Sigma | \theta^{(l)}, \xi^{(l)}), \) and transform \( \Sigma^{(l)} \) to \( \Sigma^{(l)} \).

This iterative procedure produces a sequence of samples for the model parameters \( (\theta^{(l)}, \xi^{(l)}) \) and the hyperparameter \( \Sigma^{(l)}, l = 0, \ldots, L \). To reduce the effect of the starting values, early iterations in the Markov chain are set as burn-ins to be discarded. Samples from the remaining iterations are then used to summarize the posterior density of item parameters \( \xi \), distinct person trait parameters \( \theta \), and the correlation matrix \( \Sigma \). As with standard Monte Carlo, the posterior means of all the samples collected after burn-in are considered as estimates of the true parameters \( \xi, \theta, \) and \( \Sigma \).

However, the Monte Carlo standard errors cannot be calculated using the sample standard deviations because subsequent samples in each Markov chain are autocorrelated (e.g., Patz & Junker, 1999). One approach to calculating them is through batching (Ripley, 1987). That is, with a long chain of samples being separated into contiguous batches of equal length, the Monte Carlo standard error for each parameter is then estimated to be the standard deviation of these batch means. The Monte Carlo standard error of estimate is hence a ratio of the Monte Carlo standard error and the square root of the number of batches.

The Fortran Subroutine

The subroutine initially sets the starting values for the model parameters, \( \theta, \xi, \) and the hyperparameter \( \Sigma \), so that \( \theta_{\nu}^{(0)} = 0, \quad \alpha_{\nu}^{(0)} = 2, \quad \gamma_{\nu}^{(0)} = -\Phi^{-1}(\sum_{j} \gamma_{\nu j} / n) \sqrt{\bar{s}} \) (Albert, 1992), and \( \Sigma^{(0)} = I \), with \( I \) being the identity matrix. It then iteratively draws random samples for \( Z, \theta \) and \( \Sigma \) from their respective full conditional distributions specified in (3), (4) and (6). Samples for \( \xi_{\nu} \) are simulated either from (5), where uniform priors are assumed for \( \xi_{\nu} \), or from (7), where normal priors are adopted with
The algorithm continues until all the \( L \) samples are simulated. It then discards the early burn-in samples, and computes the posterior estimates and Monte Carlo standard errors of estimates for the model parameters, \( \theta \) and \( \xi \), as well as the hyperparameter \( \Sigma \), using batching.

For example, consider binary responses of 2,000 persons to a total of 16 test items, which are further divided into two subtests so that the first half measures one latent trait and the second half measures another (i.e., \( n = 2,000 \), \( m = 2 \), \( k_1 = 8 \), \( k_2 = 8 \), and \( K = 16 \)). Three dichotomous (0-1) data matrices were simulated from the item parameters shown in the first column of Tables 1 and 2, so the actual correlation \( (\rho) \) between the two distinct latent traits \( (\theta_1, \theta_2) \) was set to be 0.2, 0.5 and 0.8, respectively. The Gibbs sampler was implemented to each data set so that 10,000 samples were simulated with the first 5,000 taken to be burn-in. The remaining 5,000 samples were separated into 5 batches, each with 1,000 samples.

With the uniform or the normal prior distributions described previously, two sets of the posterior means for \( \alpha_v \) and \( \gamma_v \) as well as their Monte Carlo standard errors were obtained for each simulated data and are displayed in the rest of the tables. Note that in all the three simulated situations, item parameters were estimated with enough accuracy and the two sets of posterior estimates differed only in the third decimal place, signifying that the results are not sensitive to the choice of prior distributions for \( \xi_{\nu} \), and \( \rho \) as well as their Monte Carlo standard errors were obtained for each simulated data and are displayed in the rest of the tables. Note that in all the three simulated situations, item parameters were estimated with enough accuracy and the two sets of posterior estimates differed only in the third decimal place, signifying that the results are not sensitive to the choice of prior distributions for \( \xi_{\nu} \). In addition, the small values of the Monte Carlo standard errors of estimates suggested that the Markov chains with a run length of 10,000 and a burn-in period of 5,000 reached the stationary distribution. Further, note that the procedure recovered the latent structure accurately as well, as the posterior estimates of the correlation between the two distinct latent traits, displayed in the last row of Table 2, was close to the actual correlation in all the three situations. For this example where 2,000-by-16 data matrices were considered, each implementation took less than 25 minutes. The length of the chains may be increased to be as long as 50,000, which takes about 90-120 minutes for each execution.

Conclusion

This Fortran subroutine allows the user to choose between uniform and normal priors for the item parameters, \( \alpha_v \) and \( \gamma_v \). In addition, the user can modify the source code by assigning other values to \( \mu_\alpha, \sigma_\alpha^2, \mu_\gamma, \sigma_\gamma^2 \) to reflect different prior beliefs on their distributions. Convergence can be assessed by inspecting Monte Carlo standard errors, as well as by comparing the marginal posterior mean and standard deviation of each parameter computed for every 1,000 samples after the burn-ins. For the latter, identical values provide a rough indication of similar marginal posterior densities, which further indicates possible convergence of the Markov chain (Gelfand, Hills, Racine-Poon & Smith, 1990; Hoijtink & Molenaar, 1997).

Note that the algorithm adopts a correlation matrix in the prior distribution, \( \Theta \sim N_m(0, \Sigma) \), to solve the problem of model nonidentifiability (see e.g., Lee, 1995, for a description of the problem). Bafumi, Gelman, Park, and Kaplan (2005) provides an alternative solution to the problem.

References


### Table 1: Posterior Estimates and Monte Carlo Standard Errors of Estimates (MCSEs) for $\alpha$, with Uniform and Normal Priors

<table>
<thead>
<tr>
<th>Parameters</th>
<th>$\rho = .2$</th>
<th>$\rho = .5$</th>
<th>$\rho = .8$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Uniform</td>
<td>Normal</td>
<td>Uniform</td>
</tr>
<tr>
<td>$\alpha_1$</td>
<td>0.0966</td>
<td>0.0838 (.0013)</td>
<td>0.0828 (.0011)</td>
</tr>
<tr>
<td></td>
<td>0.0971</td>
<td>0.0675 (.0010)</td>
<td>0.0660 (.0013)</td>
</tr>
<tr>
<td></td>
<td>0.4589</td>
<td>0.4698 (.0035)</td>
<td>0.4704 (.0026)</td>
</tr>
<tr>
<td></td>
<td>0.9532</td>
<td>0.8556 (.0039)</td>
<td>0.8531 (.0069)</td>
</tr>
<tr>
<td></td>
<td>0.0771</td>
<td>0.0510 (.0009)</td>
<td>0.0502 (.0005)</td>
</tr>
<tr>
<td></td>
<td>0.4891</td>
<td>0.4900 (.0020)</td>
<td>0.4895 (.0024)</td>
</tr>
<tr>
<td></td>
<td>0.8599</td>
<td>1.0401 (.0185)</td>
<td>1.0348 (.0114)</td>
</tr>
<tr>
<td></td>
<td>0.9427</td>
<td>0.9381 (.0075)</td>
<td>0.9327 (.0024)</td>
</tr>
<tr>
<td>$\alpha_2$</td>
<td>0.2727</td>
<td>0.3013 (.0010)</td>
<td>0.2973 (.0026)</td>
</tr>
<tr>
<td></td>
<td>0.6532</td>
<td>0.7279 (.0051)</td>
<td>0.7251 (.0061)</td>
</tr>
<tr>
<td></td>
<td>0.1002</td>
<td>0.1231 (.0010)</td>
<td>0.1226 (.0014)</td>
</tr>
<tr>
<td></td>
<td>0.2339</td>
<td>0.0945 (.0014)</td>
<td>0.0965 (.0026)</td>
</tr>
<tr>
<td></td>
<td>0.9291</td>
<td>0.8554 (.155)</td>
<td>0.8552 (.131)</td>
</tr>
<tr>
<td></td>
<td>0.8618</td>
<td>0.8730 (.128)</td>
<td>0.8575 (.095)</td>
</tr>
<tr>
<td></td>
<td>0.0908</td>
<td>0.0543 (.0006)</td>
<td>0.0518 (.0016)</td>
</tr>
<tr>
<td></td>
<td>0.2083</td>
<td>0.2003 (.0006)</td>
<td>0.1967 (.0021)</td>
</tr>
</tbody>
</table>
### Table 2: Posterior Estimates and Monte Carlo Standard Errors of Estimates (MCSEs) for $\gamma$ and $\rho$ with Uniform and Normal Priors

<table>
<thead>
<tr>
<th>Parameters</th>
<th>$\rho = .2$</th>
<th>$\rho = .5$</th>
<th>$\rho = .8$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Uniform (MCSE)</td>
<td>Normal (MCSE)</td>
<td>Uniform (MCSE)</td>
</tr>
<tr>
<td>$\gamma_1$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.3629</td>
<td>0.3457 (.0007)</td>
<td>0.3447 (.0003)</td>
<td>0.3467 (.0010)</td>
</tr>
<tr>
<td>-0.9010</td>
<td>-0.8881 (.0003)</td>
<td>-0.8875 (.0002)</td>
<td>-0.8891 (.0006)</td>
</tr>
<tr>
<td>-0.9339</td>
<td>-0.9288 (.0017)</td>
<td>-0.9277 (.0017)</td>
<td>-0.9288 (.0018)</td>
</tr>
<tr>
<td>-0.3978</td>
<td>-0.3976 (.0023)</td>
<td>-0.3983 (.0017)</td>
<td>-0.4035 (.0018)</td>
</tr>
<tr>
<td>0.3987</td>
<td>0.4077 (.0003)</td>
<td>0.4076 (.0008)</td>
<td>0.4085 (.0006)</td>
</tr>
<tr>
<td>0.1654</td>
<td>0.1679 (.0003)</td>
<td>0.1681 (.0005)</td>
<td>0.1675 (.0009)</td>
</tr>
<tr>
<td>-0.8108</td>
<td>-0.8302 (.0082)</td>
<td>-0.8294 (.0062)</td>
<td>-0.8232 (.0032)</td>
</tr>
<tr>
<td>-0.8012</td>
<td>-0.7091 (.0025)</td>
<td>-0.7064 (.0019)</td>
<td>-0.7145 (.0043)</td>
</tr>
<tr>
<td>$\gamma_2$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.2452</td>
<td>0.2902 (.0008)</td>
<td>0.2896 (.0007)</td>
<td>0.3122 (.0005)</td>
</tr>
<tr>
<td>0.9792</td>
<td>1.0954 (.0031)</td>
<td>1.0941 (.0032)</td>
<td>1.0476 (.0015)</td>
</tr>
<tr>
<td>-0.0190</td>
<td>-0.0216 (.0006)</td>
<td>-0.0212 (.0005)</td>
<td>-0.0058 (.0006)</td>
</tr>
<tr>
<td>0.8749</td>
<td>0.9549 (.0005)</td>
<td>0.9536 (.0006)</td>
<td>0.9624 (.0008)</td>
</tr>
<tr>
<td>-0.3119</td>
<td>-0.2139 (.0026)</td>
<td>-0.2143 (.0013)</td>
<td>-0.2049 (.0019)</td>
</tr>
<tr>
<td>0.2005</td>
<td>0.2902 (.0025)</td>
<td>0.2888 (.0024)</td>
<td>0.2781 (.0021)</td>
</tr>
<tr>
<td>0.4626</td>
<td>0.4658 (.0011)</td>
<td>0.4638 (.0004)</td>
<td>0.4514 (.0004)</td>
</tr>
<tr>
<td>0.7184</td>
<td>0.7528 (.0008)</td>
<td>0.7510 (.0007)</td>
<td>0.7485 (.0007)</td>
</tr>
<tr>
<td>$\rho$</td>
<td>0.1850 (.0022)</td>
<td>0.1853 (.0018)</td>
<td>0.5209 (.0031)</td>
</tr>
</tbody>
</table>
GIBBS SAMPLING FOR 2PNO MULTI-UNIDIMENSIONAL ITEM RESPONSE MODELS


Appendix: Fortran Subroutine

```fortran
SUBROUTINE GSMU2(Y, N, K, M, MN, L, BURNIN, BN, UNIF, ITEM, PER, RPER)
C*************************************************************************
C Y = the n-by-K binary item response data
C N = the number of subjects
C K = the test length (total number of items)
C M = the number of subtests
C MN = an array with numbers of items in the m subtests
C L = the number of iterations using Gibbs sampling
C BURNIN = the early number of iterations that are to be discarded
C BN = the number of batches
C UNIF = a 0-1 indicator with 0 specifying normal priors for item
C parameters and 1 specifying uniform priors for them
C ITEM = a K-by-4 matrix of posterior estimates and MCSEs for item
C parameters
C PER = a n-by-2m matrix of posterior estimates and MCSEs for person
C traits
C RPER = a (m*(m-1)/2)-by-2 matrix of posterior estimates and MCSEs
C for the correlation(s) between person traits
C*************************************************************************
INTEGER N, K, MN(M), L, Y(N,K), IRANK, IND(M), UNIF, COUNT,
& BURNIN, BSIZE, BN
REAL A(K), G(K), TH(N,M), AA(K,M), ZLP(N,K), LP, Z(N,K), PHAT(K),
& U, PVAR(M, M), SIGMA(M,M), RSIG(M,M), PVAR1(M, M), RTH(M),
```

652
& BA(K), PMEAN1(M), PMEAN(M), X(N,2), XX(2,2), IX(2,2), RMN(M),
& ZV(N,1), XZ(2,1), AMAT(2,2), BZ(2,1), AMU, GMU, AVAR, GVAR,
& AGMU(2,1), AGVAR(2,2), AGSIG(2,2), BETA(2), C(M,M), CTH(M,N),
& D(M,M), AV(L,K), GV(L,K), RHO(M,M,L), THV(N,M,L), ITEM(K,4),
& PER(N,2*M), SUM0, SUM1, SUM2, SUM3, M0, M1, M2, M3, TOT,
& TOT1, TOT2, TOT3, SS, SS1, SS2, SS3, RPER(M*(M-1)/2,2),
& PRODA, VAR(M,M)

DOUBLE PRECISION BB, TMP

C*************************************************************************
C Connect to external libraries for normal (RNNOR), uniform (RNUN), and
C multivariate normal (RNMVN) random number generator, inverse (DNORIN)
C and CDF (ANORDF, DNORDF) for the standard normal distribution, and
C Cholesky factorization (CHFAC) routines
C*************************************************************************

EXTERNAL RNNOR, RNUN, ANORDF, CHFAC, DNORDF, DNORIN, RNMVN

C*************************************************************************
C Set initial values for item parameters a(v), g(v), person ability
C theta, and the hyperparameter sigma, so a(v)=2, g(v)=\Phi^{-1}(\sum y_i/n)\sqrt{5}
C for all k(v) items, theta(v)=0 for all n person traits, and sigma=1
C*************************************************************************

PHAT = SUM(Y, DIM = 1)
DO 10 I = 1, K
   A(I) = 2.0
   G(I) = -ANORIN(PHAT(I)/FLOAT(N))*SQRT(5.0)
10 CONTINUE

DO 15 I = 1, N
   DO 15 J = 1, M
      TH(I, J) = 0.0
15 CONTINUE

DO 20 I = 1, M
   DO 20 J = 1, M
      SIGMA(I, J) = 0.0
      SIGMA(I, I) = 1.0
20 CONTINUE

RMN = FLOAT(MN)

C*************************************************************************
C Start iteration
C*************************************************************************

COUNT = 0
DO 30 IT = 1, L
   COUNT = COUNT + 1
   DO 40 I = 1, K
      DO 40 J = 1, M
         AA(I, J) = 0.0
40 CONTINUE
   JJ = 0
   DO 50 I = 1, M
      J = 1
      DO WHILE (J .LE. MN(I))
         JJ = JJ + 1
         AA(JJ, I) = A(JJ)
         J = J + 1
      END DO
50 CONTINUE
GIBBS SAMPLING FOR 2PNO MULTI-UNIDIMENSIONAL ITEM RESPONSE MODELS

C*************************************************************************
C Update samples for Z from its normal posterior distributions
C*************************************************************************
ZLP = MATMUL(TH, TRANSPOSE(AA))
DO 60 I = 1, N
    DO 60 J = 1, K
        LP = ZLP(I, J) - G(J)
        BB = ANORDF(0.0 - LP)
        CALL RNUN(1, U)
        TMP = BB*(1-Y(I, J)) + (1-BB)*Y(I, J))*U + BB*Y(I, J)
        Z(I, J) = DNORIN(TMP) + LP
    CONTINUE
60 CONTINUE
C*************************************************************************
C Update samples for theta from their MVN posterior distributions
C*************************************************************************
C*************************************************************************
C Call the matrix inversion routine.
C Invert matrix SIGMA with the inverse stored in RSIG
C*************************************************************************
CALL MIGS(SIGMA, M, RSIG, INDX)
PVAR1 = RSIG + MATMUL(TRANSPOSE(AA), AA)
C*************************************************************************
C Call the matrix inversion routine to invert matrix PVAR1 with the
C inverse stored in PVAR
C*************************************************************************
CALL MIGS(PVAR1, M, PVAR, INDX)
DO 70 I = 1, N
    DO 80 J = 1, K
        BA(J) = Z(I, J) + G(J)
    CONTINUE
70 CONTINUE
PMEAN1 = MATMUL(TRANSPOSE(AA), BA)
PMEAN = MATMUL(PVAR, PMEAN1)
C*************************************************************************
C Call the Cholesky factorization routine. Compute the Cholesky factorization
C of the symmetric definite matrix PVAR and store the C result in RSIG
C*************************************************************************
CALL CHFAC (M, PVAR, M, 0.00001, IRANK , RSIG, M)
C*************************************************************************
C Generate a random sample of theta(v) from MVN dist by calling RNMVN
C*************************************************************************
CALL RNMVN (1, M, RSIG, M, RTH, 1)
DO 90 J = 1, M
    TH(I, J) = RTH(J) + PMEAN(J)
    THV(I, J, COUNT) = TH(I, J)
90 CONTINUE
70 CONTINUE
C*************************************************************************
C Update samples for item parameters, a(v) and g(v) from their MVN
C posterior distributions
C*************************************************************************
JJ = 0
DO 100 J = 1, M
    DO 110 I = 1, N
        X(I, 1) = TH(I, J)
        X(I, 2) = -1
    CONTINUE
110 CONTINUE
IF (UNIF == 0) THEN
C*************************************************************************
C Specify the prior means and variances for a(v) and g(v)
C*************************************************************************
AMU = 0.0
GMU = 0.0
AVAR = 1.0
GVAR = 1.0
C*************************************************************************
C Put them in vector or matrix format
C*************************************************************************
AGMU(1, 1) = AMU
AGMU(2, 1) = GMU
AGVAR(1, 1) = AVAR
AGVAR(2, 2) = GVAR
C*************************************************************************
C Call the matrix inversion routine to invert matrix AGVAR with the
C inverse stored in AGSIG
C*************************************************************************
CALL MIGS(AGVAR, 2, AGSIG, INDX)
XX = MATMUL(TRANSPOSE(X), X) + AGSIG
ELSE IF (UNIF ~~ 1) THEN
XX = MATMUL(TRANSPOSE(X), X)
ENDIF
C*************************************************************************
C Call the matrix inversion routine to invert matrix XX with the
C inverse stored in IX
C*************************************************************************
CALL MIGS(XX, 2, IX, INDX)
C*************************************************************************
C Call the Cholesky factorization routine. Compute the Cholesky
C factorization of the symmetric definite matrix IX and store the
C result in AMAT
C*************************************************************************
CALL CHFAC(2, IX, 2, 0.00001, IRANK, AMAT, 2)
JM = 0
PRODA = 1.0
JM = JM + 1
JJ = JJ + 1
DO 120 I = 1, N
ZV(I, 1) = Z(I, JJ)
CONTINUE
IF (UNIF == 0) THEN
XZ = MATMUL(AGSIG, AGMU) + MATMUL(TRANSPOSE(X), ZV)
ELSE IF (UNIF == 1) THEN
XZ = MATMUL(TRANSPOSE(X), ZV)
ENDIF
BZ = MATMUL(IX, XZ)
A(JJ) = 0
DO WHILE (A(JJ) .LE. 0)
CALL RNVMVN(1, 2, AMAT, 2, BETA, 1)
A(JJ) = BETA(1) + BZ(1, 1)
G(JJ) = BETA(2) + BZ(2, 1)
END DO
AV(COUNT, JJ) = A(JJ)
GV(COUNT, JJ) = G(JJ)
END IF
GIBBS SAMPLING FOR 2PNO MULTI-UNIDIMENSIONAL ITEM RESPONSE MODELS

PRODA = PRODA*A(JJ)

IF (JM .LT. MN(J)) THEN
    GOTO 130
END IF

DO 135 I = 1, M
    C(I, J) = 0.0
135  CONTINUE

C(J, J) = PRODA ** (1/RMN(J))

CONTINUE

C*************************************************************************
C Update samples for the hyperparameter, SIGMA
C*************************************************************************

CTH = MATMUL(C, TRANSPOSE(TH))
D = MATMUL(CTH, TRANSPOSE(CTH))

C*************************************************************************
C Call the subroutine to generate the unconstrained covariance matrix
C VAR from the inverse Wishart distribution
C*************************************************************************

CALL INVWISHRND(D, M, N, VAR)

DO 140 I = 1, M
    DO 140 J = 1, M
        SIGMA(I, J) = VAR(I, J)/SQRT(VAR(I, RHO(I, J, COUNT)) = SIGMA(I, J)
140  CONTINUE

CONTINUE

C*************************************************************************
C Calculate the batch means and se's for a(v), g(v), theta(v) and
C their correlations, and store them in ITEM, PER, and RPER
C*************************************************************************

BSIZE = (L - BURNIN)/BN

DO 150 J = 1, K
    COUNT = BURNIN
    TOT1 = 0.0
    TOT2 = 0.0
    SS1 = 0.0
    SS2 = 0.0
    DO 160 JJ = 1, BN
        SUM1 = 0.0
        SUM2 = 0.0
        DO 170 I = 1, BSIZE
            COUNT = COUNT + 1
            SUM1 = SUM1 + AV(COUNT, J)
            SUM2 = SUM2 + GV(COUNT, J)
        170 CONTINUE
        M1 = SUM1/FLOAT(BSIZE)
        M2 = SUM2/FLOAT(BSIZE)
        TOT1 = TOT1 + M1
        TOT2 = TOT2 + M2
        SS1 = SS1 + M1*M1
        SS2 = SS2 + M2*M2
    160 CONTINUE

ITEM(J, 1) = TOT1/FLOAT(BN)
ITEM(J, 2) = SQRT((SS1-(TOT1*TOT1/BN))/(BN-1))/SQRT(FLOAT(BN))
ITEM(J, 3) = TOT2/BN
ITEM(J, 4) = SQRT((SS2-(TOT2*TOT2/BN))/(BN-1))/SQRT(FLOAT(BN))
CONTINUE
JJ = 0
JK = 0
DO 180 IM = 1, M
   JJ = JK + 1
   JK = JJ + 1
DO 190 J = 1, N
   COUNT = BURNIN
   TOT3 = 0.0
   SS3 = 0.0
DO 200 IB = 1, BN
   SUM3 = 0.0
   DO 210 I = 1, BSIZE
      COUNT = COUNT + 1
      SUM3 = SUM3 + THV(J, IM, COUNT)
   210 CONTINUE
   M3 = SUM3/FLOAT(BSIZE)
   TOT3 = TOT3 + M3
   SS3 = SS3 + M3*M3
200 CONTINUE
   PER(J, JJ) = TOT3/FLOAT(BN)
   PER(J, JK) = SQRT((SS3-(TOT3*TOT3/BN))/(BN-1))/SQRT(FLOAT(BN))
190 CONTINUE
CONTINUE
JJ = 0
DO 220 J = 1, M
   DO 220 IM = J + 1, M
      JK = JK + 1
      COUNT=BURNIN
      TOT = 0.0
      SS = 0.0
   DO 230 JJ = 1, BN
      SUM0 = 0.0
      DO 240 I = 1, BSIZE
         COUNT = COUNT + 1
         SUM0 = SUM0 + RHO(J, IM, COUNT)
      240 CONTINUE
      M0 = SUM0/FLOAT(BSIZE)
      TOT = TOT + M0
      SS = SS + M0*M0
230 CONTINUE
   RPER(JK, 1) = TOT/FLOAT(BN)
   RPER(JK, 2) = SQRT((SS-(TOT*TOT/BN))/(BN-1))/SQRT(FLOAT(BN))
220 CONTINUE
RETURN
END
SUBROUTINE INVWISHRND(S, P, V, IW)
C*************************************************************************
C S = p-by-p symmetric, positive definite 'scale' matrix
C P = order of the scale matrix
C V = 'degree of freedom parameter'
C (V must be an integer for this routine)
C IW = random matrix from the inverse Wishart distribution
C Note:
C different sources use different parameterizations w.r.t. V.
C this routine uses the model that
C density (IW) is proportional to
C \[ \exp\left[-.5*\text{trace}(S*\text{inv}(IW))\right]/[\text{det}(IW)^{(V+p+1)/2}] \]
C With this density definition:
C \[ \text{mean}(IW) = S/(V-p-1) \]
C*************************************************************************
INTEGER P, V, IRANK, INDX(P)
REAL S(P, P), IS(P, P), IW(P, P), W(P, P), Z(V, P), ZZ(P, P),
& A(P, P), AZ(P, P)
DO 10 I = 1, V
DO 10 J = 1, P
CALL RNNOR (1, Z(I, J))
10 CONTINUE
ZZ = MATMUL(TRANSPOSE(Z), Z)
CALL MIGS(S, P, IS, INDX)
CALL CHFAC (P, IS, P, 0.00001, IRANK, A, P)
AZ = MATMUL(TRANSPOSE(A), ZZ)
W = MATMUL(AZ, A)
CALL MIGS(W, P, IW, INDX)
RETURN
END