Comparing Gibbs sampling and EM-Algorithm method for missing data imputation*

Hao Xu T00732492

April 20, 2024

Abstract

This study presents a comparative analysis of Gibbs sampling methods and the Expectation-Maximization (EM) Algorithm for imputing missing data, specifically within the context of an IELTS scores dataset assumed to follow a multivariate normal distribution. Through implementation and simulation in R, we evaluated the efficacy of both techniques across varying levels of data incompleteness. Our findings reveal that both Gibbs sampling and the EM Algorithm demonstrate comparable performance when the proportion of missing data is minimal. However, as the extent of missing data increases to 30%, Gibbs sampling outperforms the EM Algorithm in terms of imputation accuracy and reliability. These results underscore the potential of Gibbs sampling as a robust method for handling larger proportions of missing data, providing significant implications for researchers and practitioners dealing with incomplete datasets in educational assessments and beyond.

1 Introduction

Missing data imputation is a critical process in data analysis across various fields, addressing the challenge of incomplete datasets which can potentially lead to biased or invalid conclusions if not properly handled. In the real world, missing data occur in virtually every domain including healthcare, finance, social sciences, and environmental studies. For instance, in healthcare, missing data can arise from patient dropout, non-response to surveys, or malfunctioning equipment, leading to incomplete patient records that are crucial for accurate diagnosis and treatment planning. Similarly, in finance, gaps in datasets can affect the analysis of economic trends or financial risk assessments, where every data point might represent crucial information about market behaviors or credit scores. Imputation techniques, such as the EM algorithm, enable researchers and analysts to estimate these missing values accurately, thereby maintaining the integrity of statistical analyses and ensuring that decision-making is based on robust and comprehensive data sets. This not only enhances the quality of insights derived from data but also supports more informed policy-making and strategic planning in professional practices.

*The source code is available: https://github.com/cld4h/Gibbs_Vs_EM

2 Literature Review

2.1 Gibbs sampling method

Gibbs Sampling, introduced in 1984 by Geman and Geman [1], is a form of Markov chain Monte Carlo (MCMC) methodology that facilitates the simulation from a joint probability distribution. The process involves sequentially sampling from the conditional distributions of each variable, holding the values of all other variables at their current states. This technique is particularly useful in the context of missing data imputation, where Gibbs sampling is employed to generate samples of the missing data based on their conditional distributions, given the observed data and the current estimates of parameters. This iterative approach ensures a comprehensive exploration of the distribution, making it a powerful tool for dealing with incomplete datasets in statistical analysis.

2.2 EM algorithm

The Expectation-Maximization (EM) algorithm, proposed in 1977 by Arthur P. Dempster [2], is a hybrid inferential method that straddles both Bayesian and frequentist statistical frameworks. While primarily a maximum likelihood approach, EM yields both a probability distribution over latent variables—reminiscent of Bayesian analysis—and a point estimate for the parameter θ, which could be either a maximum likelihood estimate or a posterior mode. In cases where a fully Bayesian treatment is desirable, θ can be treated as an additional latent variable, effectively integrating it within the entire probabilistic model. Under such a Bayesian extension, the traditional separation of the algorithm into Expectation (E) and Maximization (M) steps becomes less distinct, showcasing the fluidity and adaptability of the EM algorithm in handling incomplete data sets across various statistical paradigms.

2.3 Missing data imputation

In statistical analysis, understanding the nature of missing data is crucial for applying appropriate imputation methods and interpreting results accurately. Missing data can be classified into three primary types: Missing Completely at Random (MCAR), Missing at Random (MAR), and Missing Not at Random (MNAR).

Type 1: Missing Completely at Random (MCAR) Missing Completely at Random (MCAR) refers to data that is absent purely due to random factors, with no discernible pattern or structure influencing its absence. In such cases, the missing data points are independent of both the observed and unobserved data.

Type 2: Missing at Random (MAR) Missing at Random (MAR) occurs when the absence of data is related to the observable attributes of the data set. Although the missing data itself isn't random, its likelihood can be accurately predicted using other available information or variables within the same observation.

Type 3: Missing Not at Random (MNAR) Missing Not at Random (MNAR) describes a scenario where the missing data is directly influenced by the values that are missing. This type of missing data possesses inherent patterns that are closely linked to the characteristics of the missing observations themselves. Failure to accurately identify and address MNAR can lead to biased analyses and less effective solutions when applied in real-world contexts.

In this paper, we are only considering Type 1 and Type 2 case. In cases where the data are missing but not at random, we need to model the relationship between the missing data and the parameters[3], which makes comparison betweenn EM algorithms and Gibbs samplign methods complicated.

3 Objectives

The primary objectives of this study are outlined as follows:

1. **Implement the EM-Algorithms and Gibbs sampling methods in R.** Our first goal is to develop and implement robust computational procedures for both the EM Algorithm and Gibbs sampling methods using the R programming language. This implementation will involve writing custom functions and scripts to handle missing data imputation processes efficiently. By leveraging R's powerful statistical and graphical capabilities, we aim to create a flexible framework that can be applied to a wide range of datasets with missing values.

2. **Identify the assumptions of applying these two methods.** Both the EM Algorithm and Gibbs sampling rely on specific statistical assumptions which must be clearly understood and validated to ensure the appropriateness of each method for handling missing data. This objective involves a thorough examination of the underlying assumptions such as data distribution (e.g., assuming multivariate normality), independence, and the missing data mechanism (MCAR, MAR, MNAR). Recognizing these assumptions will aid in the proper application and interpretation of each technique's results.

3. **Comparative Analysis of Performance.** The final objective is to conduct a detailed comparison of the performance of the EM Algorithm and Gibbs sampling across various scenarios characterized by different percentages of missing data. Performance metrics such as imputation accuracy, computational efficiency, and the robustness of each method will be evaluated. This comparison will not only focus on scenarios with minimal missing data (e.g., 5-10%) but also explore the efficacy of both methods under substantial missing conditions (up to 30% and beyond). The insights gained from this analysis will guide practitioners in choosing the most suitable method based on the extent and nature of missing data in their specific contexts.

4 Data and Variables

In this work, we downloaded the data from 2019 IELTS Test Statistic Report[1] and extracted the "Academic mean performance by nationality" table.

The table contains five columnns, which are "Reading", "Listening", "Writing", "Speaking" and "Overall", each represent one type of score from the IELTS test. Each row of the data represent the average score of students from a specific nationality. Figure 2 (in Appendix 9.1) plots all the data poins from the table.

The univariate histogram and bivariate scatterplots of the test data is shown in Figure 1. We can see that all five variabels are normally distributed, and there are correlation between these 5 variables. So our assumption is that the data follows a multivariate normal distribution.

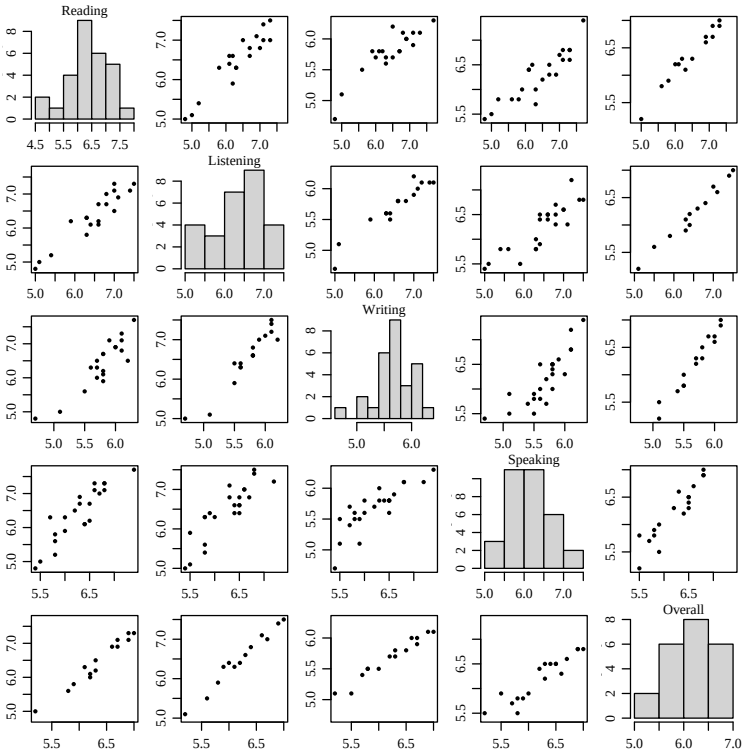


Figure 1: IELTS scores of 40 different nationalities

5 Methods

This section outlines the methodologies employed to simulate the missing data imputation preocess in the IELTS scores dataset. Specifically, we apply Gibbs Sampling and the EM Algorithm, both robust methods in the field of statistical imputation, to estimate missing values under the assumption that the data follows a multivariate normal distribution. The

[1]https://ieltscenterofficial.github.io/examcenter/for-researchers/test-statistics/test-taker-performance.html

objective is to not only impute missing values but also to compare the efficacy and computational efficiency of these two approaches under varying conditions of data incompleteness. This comparative analysis aims to identify which method provides more accurate imputations and under what circumstances.

5.1 Gibbs sampling

In Bayesian inference we treat the unknown parameters of interests as random variables. Assuming multivariate normal model, the parameters $\boldsymbol{\theta}$ and Σ are unknown, and the missing data are also an unknown. Further more, the missing data are also key compoents of our model. Treating it as such allows us to use Gibbs sampling to make inference on $\boldsymbol{\theta}$, Σ, as well as to make predictions for the missing values.[4]

Let $\mathbf{Y}$ be the $n \times p$ matrix of all IELTS scores, observed and unobserved, and let O be the $n \times p$ matrix in which $o_{i,j} = 1$ if $Y_{i,j}$ is observed and $o_{i,j} = 0$ if $Y_{i,j}$ is missing. The matrix $\mathbf{Y}$ can then be thought of as consisting of two parts:

- $\mathbf{Y}_{\text{obs}} = \{y_{i,j} : o_{i,j} = 1\}$, the data that we do observe, and
- $\mathbf{Y}_{\text{miss}} = \{y_{i,j} : o_{i,j} = 0\}$, the data that we do not observe.

Prior selection: we take the mean of the observed data as the prior mean $\boldsymbol{\mu}_0$. For standard deviation, we want to use a weak prior. From the IELTS test statistics report[2], we know that the more than 95% people have a test score between 4 and 8.5. Therefore, we know that given the mean $\boldsymbol{\mu}_0$, we have the following property:

$$\Phi(\frac{\mathbf{8.5} - \boldsymbol{\mu}_0}{\boldsymbol{\sigma}_0}) - \Phi(\frac{\mathbf{4} - \boldsymbol{\mu}_0}{\boldsymbol{\sigma}_0}) = (95\%, 95\%, 95\%, 95\%, 95\%)^T$$

In practice, we do not have to solve the above equation precisely. Instead, we take all standard deviation elemennts in $\boldsymbol{\sigma}_0 = \frac{8-4.5}{2*\text{qnorm}(0.95)} = 1.1538$. Also we know that there are some correlations between the four sections and overall scores of IELTS test, so we take the prior variance-covariance matrix as follows:

$$\Lambda_0 = \Sigma^{(0)} = \begin{pmatrix} 1.33 & 0.67 & 0.67 & 0.67 & 0.67 \\ 0.67 & 1.33 & 0.67 & 0.67 & 0.67 \\ 0.67 & 0.67 & 1.33 & 0.67 & 0.67 \\ 0.67 & 0.67 & 0.67 & 1.33 & 0.67 \\ 0.67 & 0.67 & 0.67 & 0.67 & 1.33 \end{pmatrix}.$$

Given the starting values $\{\Sigma^{(0)}, \mathbf{Y}^{(0)}_{\text{miss}}\}$, we iteratively generate $\{\boldsymbol{\theta}^{(s+1)}, \Sigma^{(s+1)}, \mathbf{Y}^{(s+1)}_{\text{miss}}\}$ from $\{\boldsymbol{\theta}^{(s)}, \Sigma^{(s)}, \mathbf{Y}^{(s)}_{\text{miss}}\}$ by the following steps:

1. sampling $\boldsymbol{\theta}^{(s+1)}$ from multivariate normal$(\boldsymbol{\mu}_n, \Lambda_n)$, where

$$\boldsymbol{\mu}_n = (\Lambda_0^{-1} + n{\Sigma^{(s)}}^{-1})^{-1}(\Lambda_0^{-1}\boldsymbol{\mu}_0 + n{\Sigma^{(s)}}^{-1}\overline{\boldsymbol{Y}^{(s)}}) \text{ and}$$
$$\Lambda_n = (\Lambda_0^{-1} + n{\Sigma^{(s)}}^{-1})^{-1}$$

[2]https://ielts.org/researchers/our-research/test-statistics

2. sampling $\Sigma^{(s+1)}$ from invert-Wishart $(\nu_0 + 5, \mathbf{S}_n^{-1})$, where

$$\begin{aligned}
\mathbf{S}_n &= \mathbf{S}_0 + \mathbf{S}_\theta \\
\mathbf{S}_0 &= \Lambda_0 \\
\mathbf{S}_\theta &= \sum_{i=1}^{n} (\mathbf{Y}_i^{(s)} - \boldsymbol{\theta}^{(s+1)})(\mathbf{Y}_i^{(s)} - \boldsymbol{\theta}^{(s+1)})^T \text{ and} \\
\nu_0 &= 2 \text{ for a weak prior}
\end{aligned}$$

3. sampling $\mathbf{Y}_{\text{miss}}^{(s+1)}$ from multivariate normal $(\boldsymbol{\theta}_{b|a}^{(s+1)}, \Sigma_{b|a}^{(s+1)})$, where

$$\begin{aligned}
\boldsymbol{\theta}_{b|a}^{(s+1)} &= \boldsymbol{\theta}_{[b]}^{(s+1)} + \Sigma_{[b,a]}^{(s+1)} (\Sigma_{[a,a]}^{(s+1)})^{-1} (\mathbf{Y}_{[a]} - \boldsymbol{\theta}_{[a]}^{(s+1)}) \\
\Sigma_{b|a}^{(s+1)} &= \Sigma_{[b,b]}^{(s+1)} + \Sigma_{[b,a]}^{(s+1)} (\Sigma_{[a,a]}^{(s+1)})^{-1} \Sigma_{[a,b]}^{(s+1)}.
\end{aligned}$$

   Here, $\boldsymbol{\theta}_{[b]}^{(s+1)}$ refers to the elements of $\boldsymbol{\theta}^{(s+1)}$ corresponding to the indices in $\mathbf{b}$. $\mathbf{b}$ is a subset of variable indices $\{1, 2, 3, 4, 5\}$ whose corresponding variable Y is missing and $\mathbf{a}$ is a complement of $\mathbf{b}$. $\Sigma_{[a,b]}$ refers to the matrix made up of the elemennts that are in rows $\mathbf{a}$ and column $\mathbf{b}$ of Σ.

5.2 EM algorithm

Under the same setup of a multivariate normal model for the IELTS Scores data $\mathbf{Y}$ as described in section 5.1, we derive the steps for EM algorithm in this section.

The general idea to perform EM algorithm is to iterate through the following steps until apparent convergence[5]:

1. Replace missing values by estimated values
2. Estimate parameters
3. Re-estimate the missing values assuming the new parameter estimates are correct
4. Re-estimate parameters

In EM algorithm, we do not treat $\boldsymbol{\theta}$ and Σ as a random variable as in section 5.1. Instead, when estimating those parameters, we are trying to maximize the loglikelihood function:

$$\mathcal{L}(\boldsymbol{\theta}, \Sigma | \mathbf{Y}_{\text{obs}}) = \text{Const} - \frac{1}{2} \sum_{i=1}^{n} \ln \mid \Sigma_{[a,a]} \mid - \frac{1}{2} \sum_{i=1}^{n} (\mathbf{Y}_{i,[a]} - \boldsymbol{\theta}_{[a]})^T \Sigma_{[a,a]} (\mathbf{Y}_{i,[a]} - \boldsymbol{\theta}_{[a]}).$$

Here we are continue using the notation defined in section 5.1 step 3, where $\boldsymbol{\theta}_{[a]}$ and $\Sigma_{[a,a]}$ represent the mean and variance-covariance matrix of the observed values respectively.

The hypothetical distribution of the complete data $\mathbf{Y}$ is a multivariate normal which belongs to the regular exponential family, therefore it has the following sufficient statistics:

$$S = \left(\sum_{i=1}^{n} y_{i,j};\ \text{and}\ \sum_{i=1}^{n} y_{i,j} y_{i,k}, \quad j, k = 1, 2, 3, 4, 5 \right).$$

The EM algorithm is to iteratively run through the following steps until convergence: (Using $\boldsymbol{\theta}^{(t)}$ and $\Sigma^{(t)}$ to represent the estimates of the parameters at the t^{th} iteration of EM)

1. The E step of the EM algorithm for iteration $t+1$ is to calculate
   (a) $E\left(\sum_{i=1}^{n} y_{i,j}|\mathbf{Y}_{\text{obs}}, \boldsymbol{\theta}^{(t)}\right) = \sum_{i=1}^{n} y_{i,j}^{(t+1)}, j = 1,2,3,4,5$ and
   (b) $E\left(\sum_{i=1}^{n} y_{i,j}y_{i,k}|\mathbf{Y}_{\text{obs}}, \boldsymbol{\theta}^{(t)}\right) = \sum_{i=1}^{n}\left(y_{i,j}^{(t+1)}y_{i,k}^{(t+1)} + c_{j,k,i}^{(t+1)}\right), j,k = 1,2,3,4,5$
   (c) sampling $\mathbf{Y}_{\text{miss}}$ from multivariate normal $\left(\boldsymbol{\theta}_{b|a}^{(t)}, \Sigma_{b|a}^{(t)}\right)$ (same calculation as step 3 in the Gibbs sampling method)

   Here in the above steps,

$$y_{i,j}^{(t+1)} = \begin{cases} y_{i,j} & \text{, if } o_{i,j} = 1, \\ E\left(y_{i,j}|\mathbf{Y}_{i,[a]}, \boldsymbol{\theta}^{(t)}\right) & \text{, if } o_{i,j} = 0 \end{cases}$$

   and

$$c_{j,k,i}^{(t+1)} = \begin{cases} 0 & \text{, if } o_{i,j} + o_{i,k} \geq 1, \\ \text{Cov}(y_{i,j}, y_{j,k}|\mathbf{Y}_{i,[a]}, \boldsymbol{\theta}^{(t)}) & \text{, if } o_{i,j} + o_{i,k} = 0, \end{cases}$$

2. The M step is to update $\boldsymbol{\theta}^{(t+1)}$ and $\Sigma^{(t+1)}$ based on the result from the E step.
   (a) The j^{th} element at $\boldsymbol{\theta}^{(t+1)}$ is: $\theta_j^{(t+1)} = n^{-1}E\left(\sum_{i=1}^{n} y_{i,j}|\mathbf{Y}_{\text{obs}}, \boldsymbol{\theta}^{(t)}\right), j = 1,2,3,4,5$
   (b) The element at j^{th} row k^{th} column of $\Sigma^{(t+1)}$ is denoted by $\sigma_{j,k}^{(t+1)}$

$$\sigma_{j,k}^{(t+1)} = n^{-1}E\left(\sum_{i=1}^{n} y_{i,j}y_{i,k}|\mathbf{Y}_{\text{obs}}, \boldsymbol{\theta}^{(t)}\right) - \theta_j^{(t+1)}\theta_k^{(t+1)}, \qquad j,k = 1,2,3,4,5$$

3. Calculate $\|\boldsymbol{\theta}^{(t+1)} - \boldsymbol{\theta}^{(t)}\|$ and $\|\Sigma^{(t+1)} - \Sigma^{(t)}\|$ to check for convergence.

6 Simulations

To simulate missing data within our dataset, we employ the `rbinom(n*p,1,.9)` function in R, which randomly assigns `NA` (i.e., missing data status) to selected data entries. The probability parameter pp is varied to create different scenarios with missing data ranging from 10% to 30%. This approach allows us to maintain a control group by keeping a record of the true values for subsequent performance evaluation.

In the simulation of the EM algorithm, we configure the maximum number of iterations to 100 and set the convergence threshold to 10^{-6}. The outcomes of these simulations include the estimation of the final mean and variance-covariance matrix, as well as the imputation of missing data.

For the Gibbs sampling process, we choose a longer iteration count, set at 1000 iterations. This approach aims to generate robust estimates of the posterior mean, variance-covariance

matrix, and the imputed missing data. The extended number of iterations in Gibbs sampling is expected to provide more accurate and stable results, especially in scenarios with higher percentages of missing data.

The effectiveness of each method is assessed by computing the mean square error (MSE) of the predictions across the various scenarios of missing data percentages, thus providing a quantitative measure of performance.

7 Results

In this section, we present the findings from our simulations to evaluate the performance of Gibbs Sampling and the EM Algorithm in handling missing data within an IELTS scores dataset. To facilitate a clear and comprehensive understanding of how well each method predicted the missing values, we have included a detailed plot that illustrates the correlation between the predicted values and the true values. This plot can be found in the Appendix 9.2.

Furthermore, to quantitatively assess the accuracy of each imputation method across different scenarios, we calculated the mean squared errors (MSE) for varying levels of missing data. These MSE values are crucial indicators of prediction accuracy, providing a straightforward metric for comparing the performance of the two methods under conditions ranging from 10% to 30% missing data.

The mean squared errors for each scenario are listed in Table 1.

Percentage Missing	**EM Algorithm MSE**	**Gibbs Sampling MSE**
10%	0.036	0.015
15%	0.026	0.024
20%	0.017	0.022
25%	0.033	0.027
30%	0.069	0.039

Table 1: Comparison of Mean Squared Errors for EM Algorithm and Gibbs Sampling

We also recoreded the computation time it takes to run each algorithms, listed in Table 2

EM Algorithm	**Gibbs Sampling**
0.6542037	11.12866
1.000516	10.44828
1.118946	10.67758
1.15855	10.25005
1.262844	10.27943

Table 2: Comparison of computation time (in seconds) for EM Algorithm and Gibbs Sampling

8 Summary and Discussions

From the simulation results, we observe that both the Gibbs sampling method and the EM algorithm perform competently in the task of missing data imputation. When the amount of missing data is relatively low (up to 20%), the performance of both methods is comparable. Notably, the EM algorithm demonstrates a significantly shorter computation time, requiring approximately one-tenth the computation duration needed for Gibbs sampling.

However, the scenario changes as the missing data proportion increases to 30%. In such cases, Gibbs sampling notably surpasses the EM algorithm in terms of prediction accuracy. This superior performance of Gibbs sampling in high-missing-data contexts suggests that Bayesian methods provide a robust framework for dealing with complex uncertainty and integrating prior knowledge effectively. These methods are particularly adept at handling larger gaps in data by utilizing probability distributions to estimate missing values, rather than relying solely on observed data patterns. This approach can be particularly beneficial in scenarios where traditional methods might fail to provide reliable estimates due to the extent of incompleteness. Thus, Bayesian techniques like Gibbs sampling demonstrate considerable potential in improving the quality of data imputation in statistical analyses, especially when the missing data proportion is substantial.

These findings suggest that while the EM algorithm might be preferred for quicker imputation tasks with less missing data, Gibbs sampling emerges as the more robust method in scenarios where the missing data proportion is substantial and precision is paramount. Future research could explore the scalability of these methods in even larger datasets and the potential adjustments in their configurations to optimize performance across varying conditions.

References

[1] Stuart Geman and Donald Geman. "Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-6.6 (1984), pp. 721–741. DOI: 10.1109/TPAMI.1984.4767596 (cit. on p. 2).

[2] Arthur P Dempster, Nan M Laird, and Donald B Rubin. "Maximum likelihood from incomplete data via the EM algorithm". In: *Journal of the royal statistical society: series B (methodological)* 39.1 (1977), pp. 1–22 (cit. on p. 2).

[3] Andrew Gelman et al. *Bayesian Data Analysis*. 3rd ed. Chapman and Hall/CRC, 2013. DOI: 10.1201/b16018. URL: https://doi.org/10.1201/b16018 (cit. on p. 3).

[4] Peter D. Hoff. *A First Course in Bayesian Statistical Methods*. 1st. Springer Publishing Company, Incorporated, 2009. ISBN: 0387922997 (cit. on p. 5).

[5] Roderick J.A. Little and Donald B.Rubin. *Statistical Analysis with Missing Data*. John Wiley & Sons, Inc., 111 River Street, Hoboken, NJ 07030, USA: Wiley, 2020 (cit. on p. 6).

9 Appendix

9.1 IELTS Scores of each Nationality

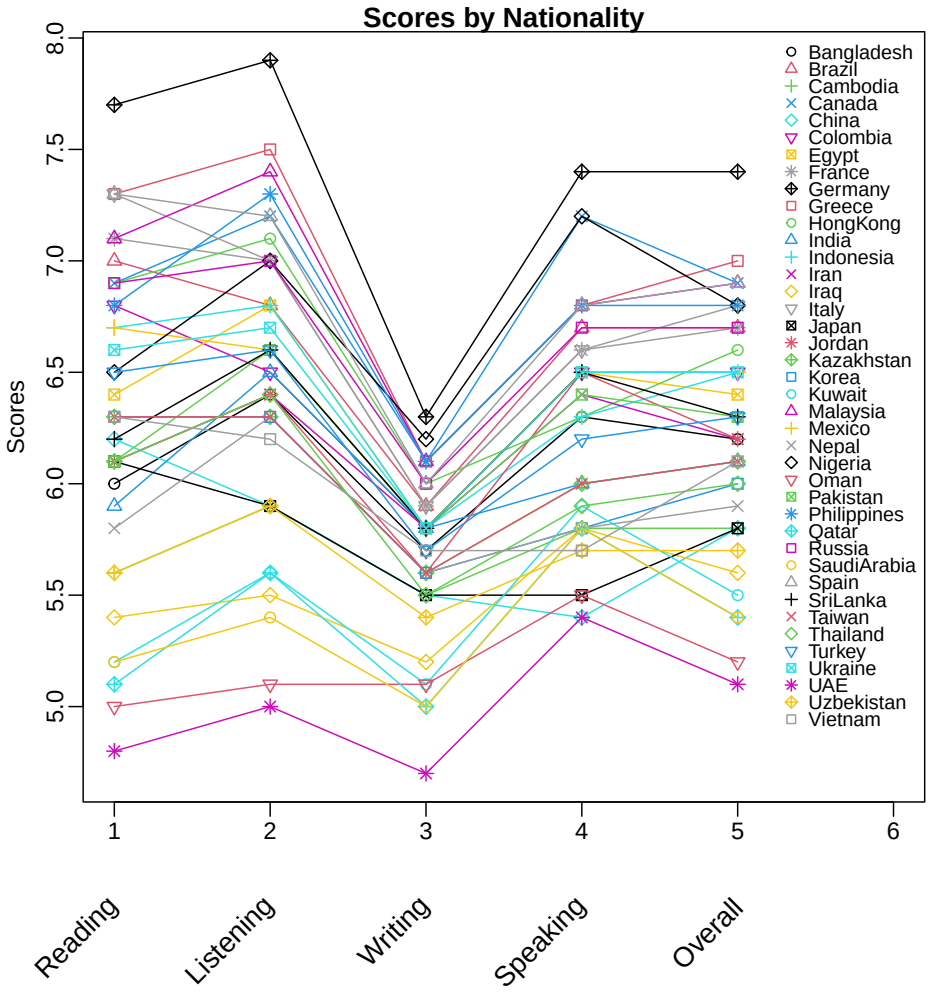
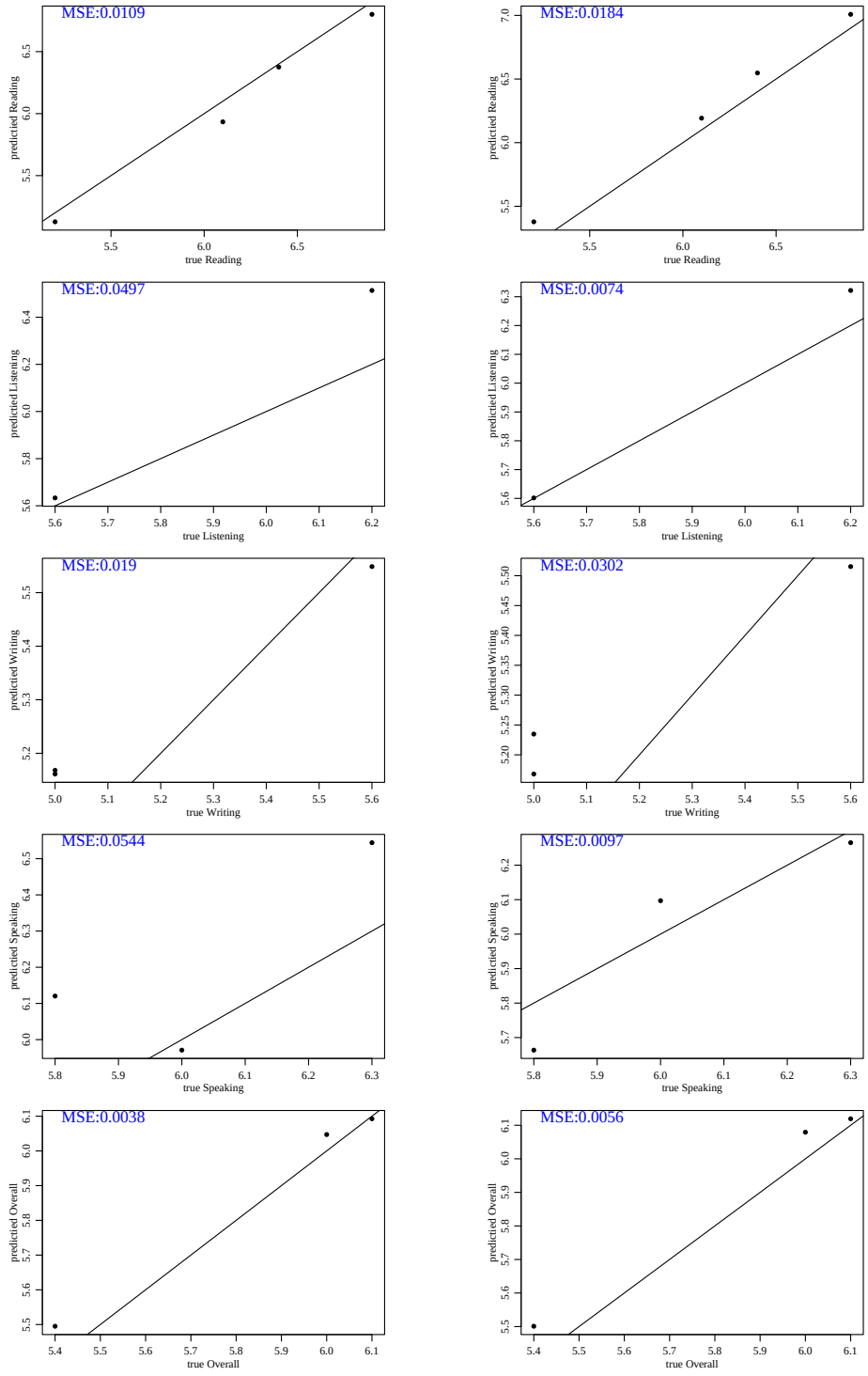


Figure 2: IELTS Scores of each Nationality

9.2 Simulation result:

9.2.1 10% of missing data



(a) EM algorithm (b) Gibbs sampling method

Figure 3: Prediction vs True value

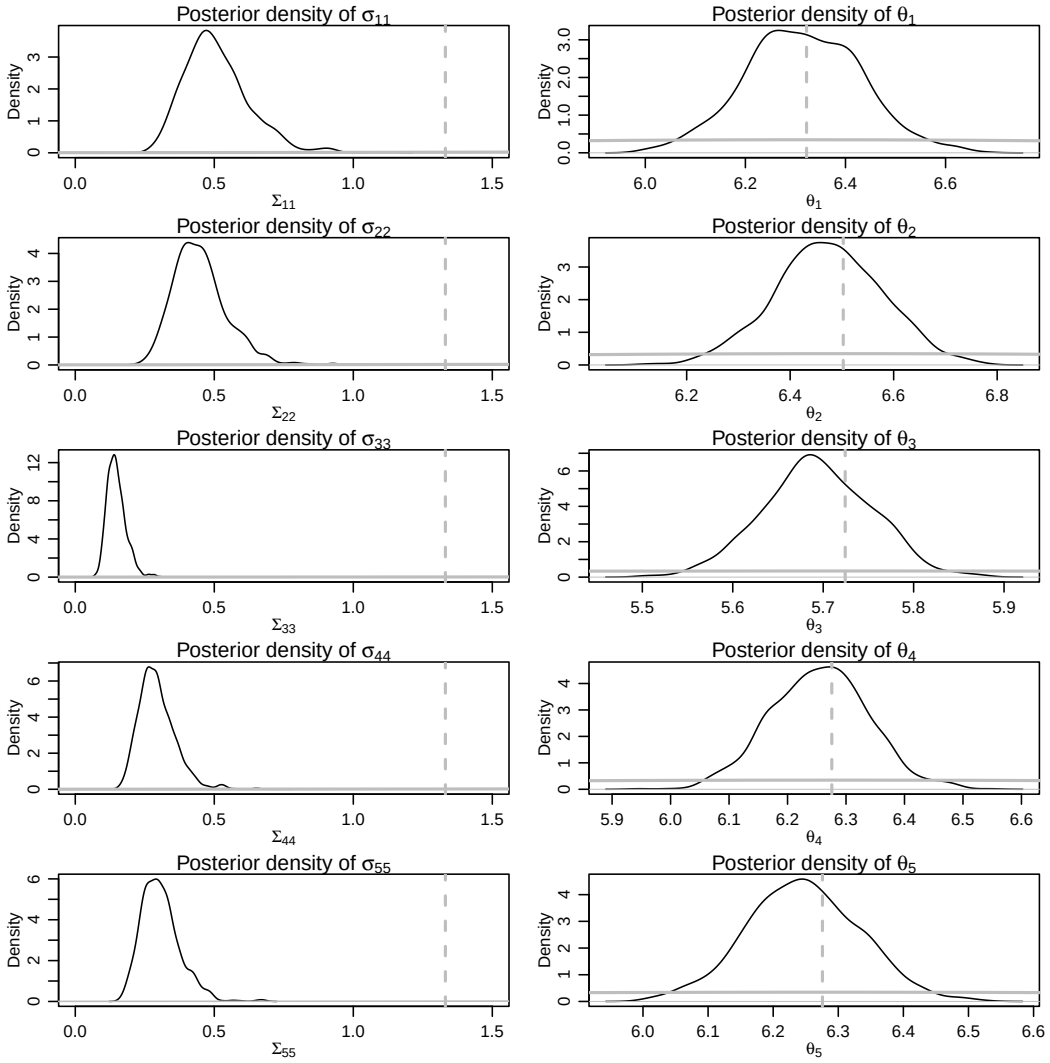
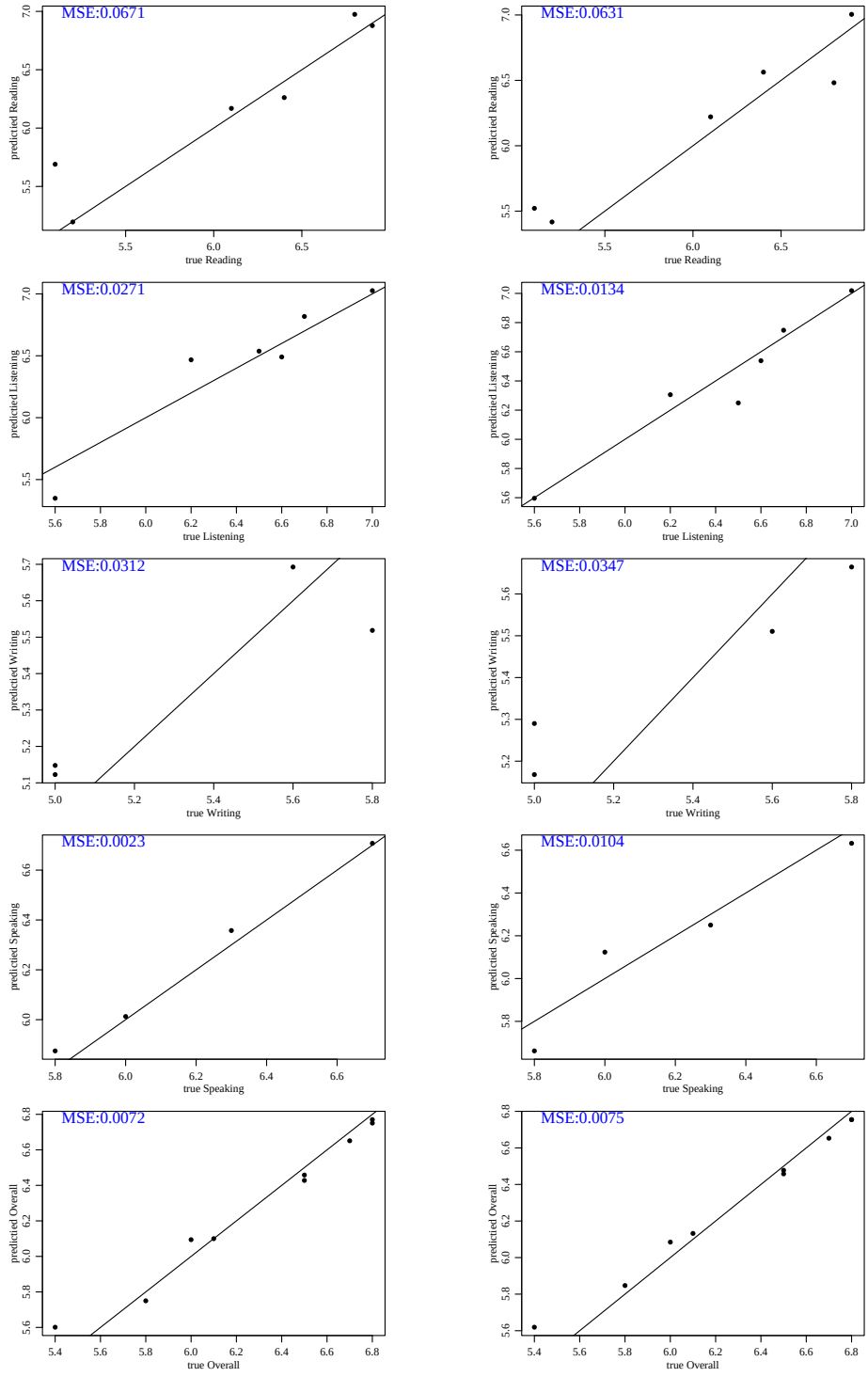


Figure 4: The posterior (and prior) distribution of $\sigma_{j,j}$ (left) and θ_j (right). Posterior distribution is ploted in black, and prior distribution is ploted in gray. The vertical dashed line marks the initial value for prior.

9.2.2 15% of missing data



(a) EM algorithm (b) Gibbs sampling method

Figure 5: Prediction vs True value

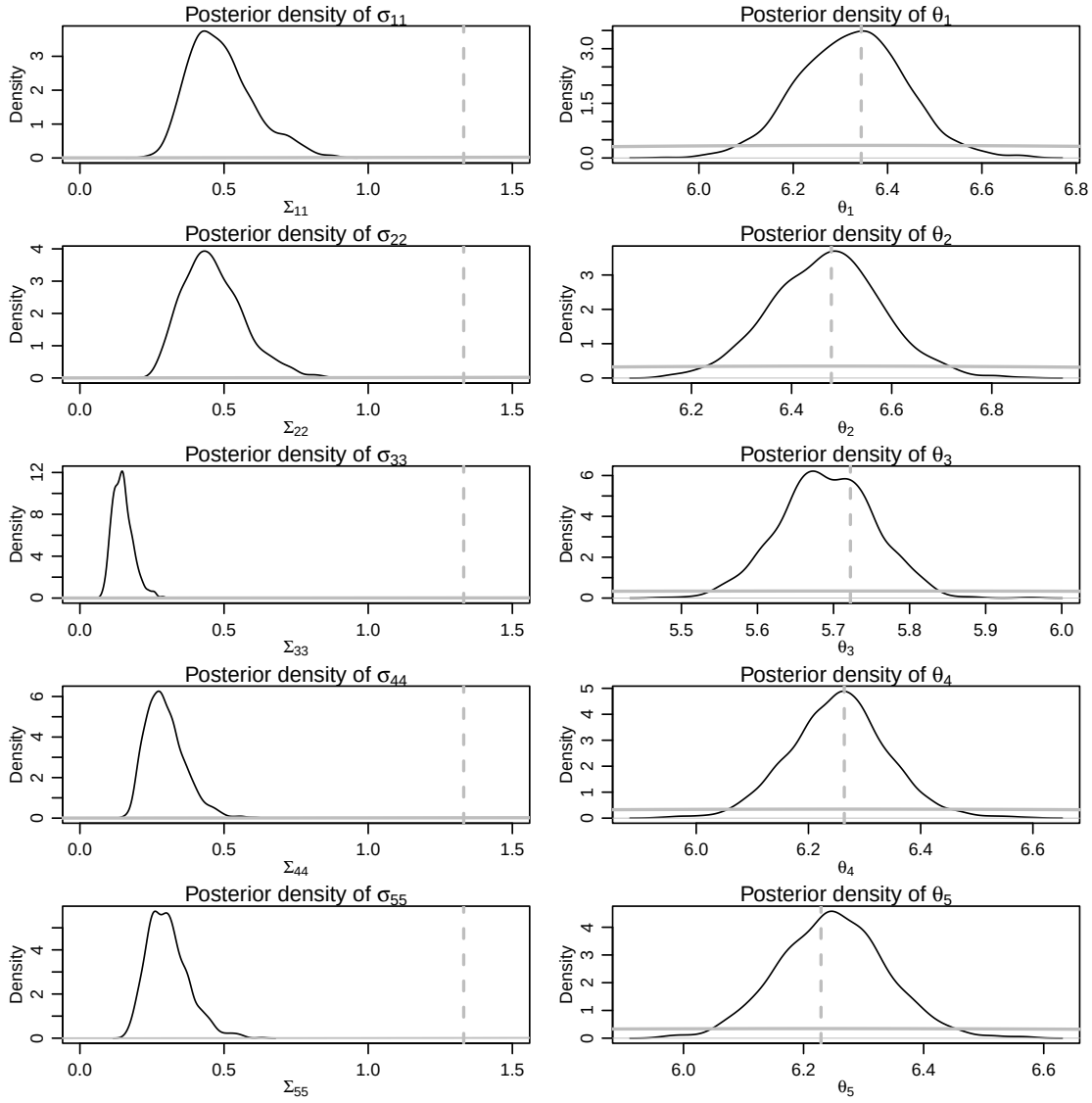
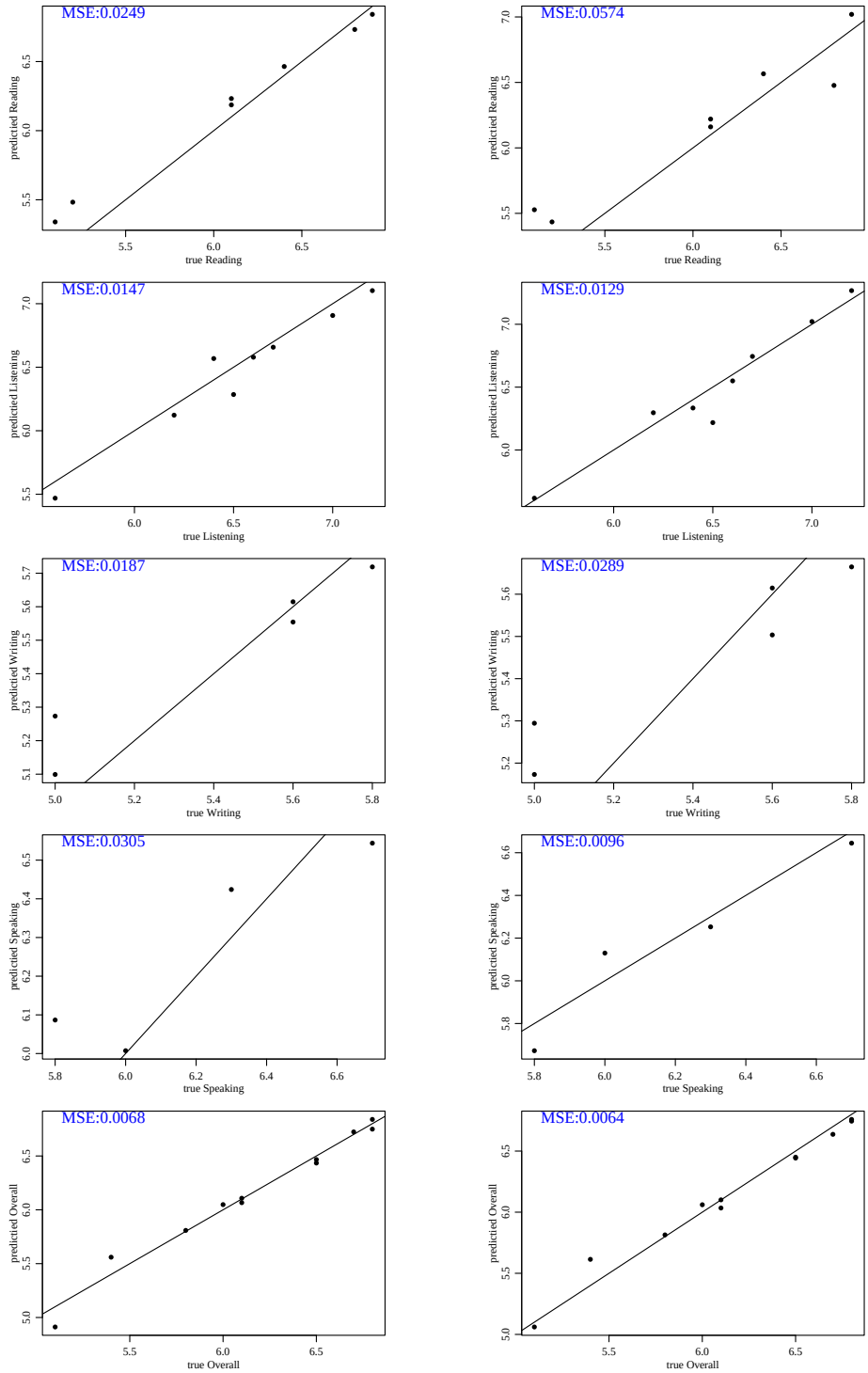


Figure 6: The posterior (and prior) distribution of $\sigma_{j,j}$ (left) and θ_j (right). Posterior distribution is ploted in black, and prior distribution is ploted in gray. The vertical dashed line marks the initial value for prior.

9.2.3 20% of missing data



(a) EM algorithm (b) Gibbs sampling method

Figure 7: Prediction vs True value

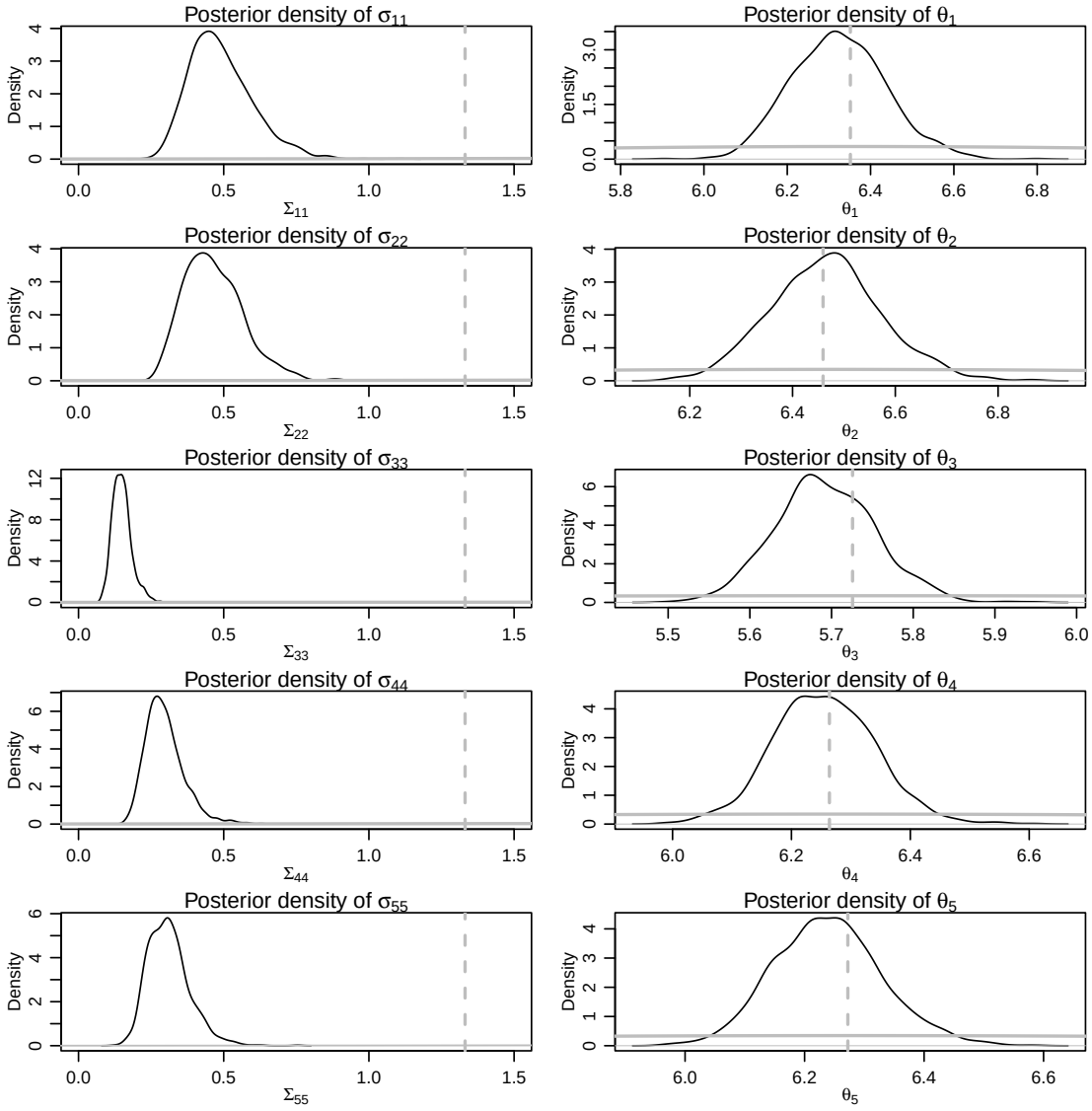
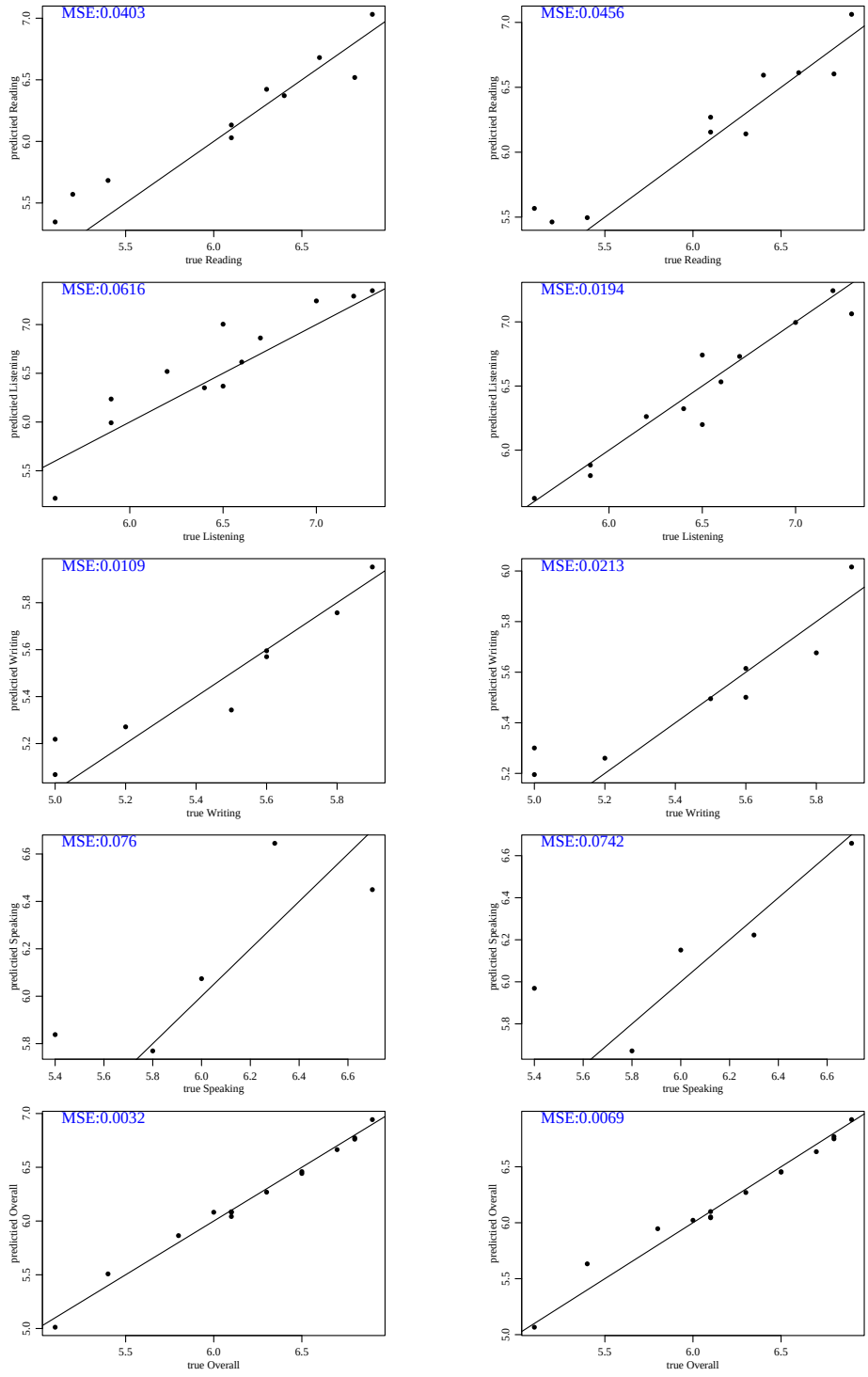


Figure 8: The posterior (and prior) distribution of $\sigma_{j,j}$ (left) and θ_j (right). Posterior distribution is ploted in black, and prior distribution is ploted in gray. The vertical dashed line marks the initial value for prior.

9.2.4 25% of missing data



(a) EM algorithm (b) Gibbs sampling method

Figure 9: Prediction vs True value

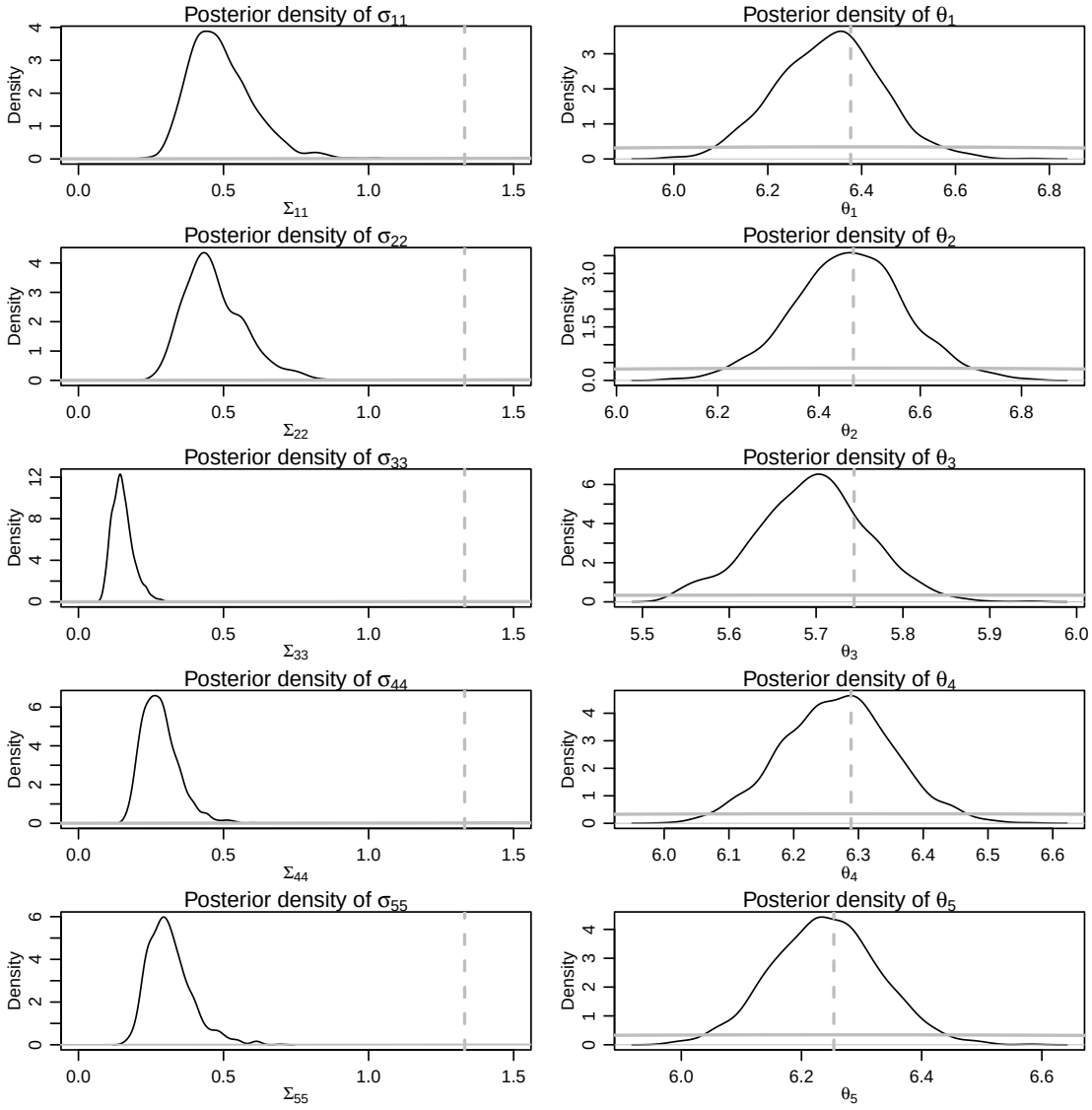
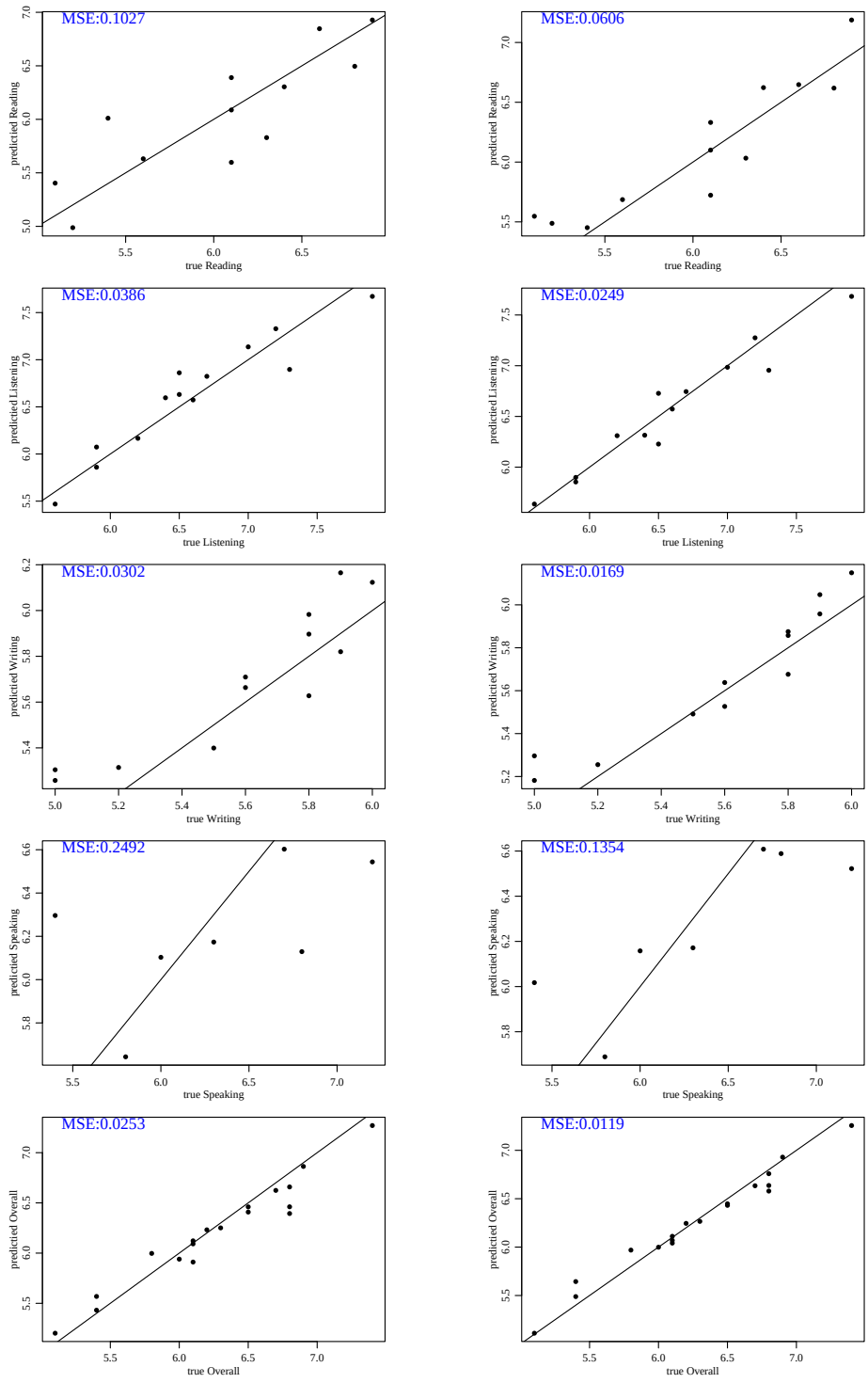


Figure 10: The posterior (and prior) distribution of $\sigma_{j,j}$ (left) and θ_j (right). Posterior distribution is ploted in black, and prior distribution is ploted in gray. The vertical dashed line marks the initial value for prior.

9.2.5 30% of missing data



(a) EM algorithm (b) Gibbs sampling method

Figure 11: Prediction vs True value

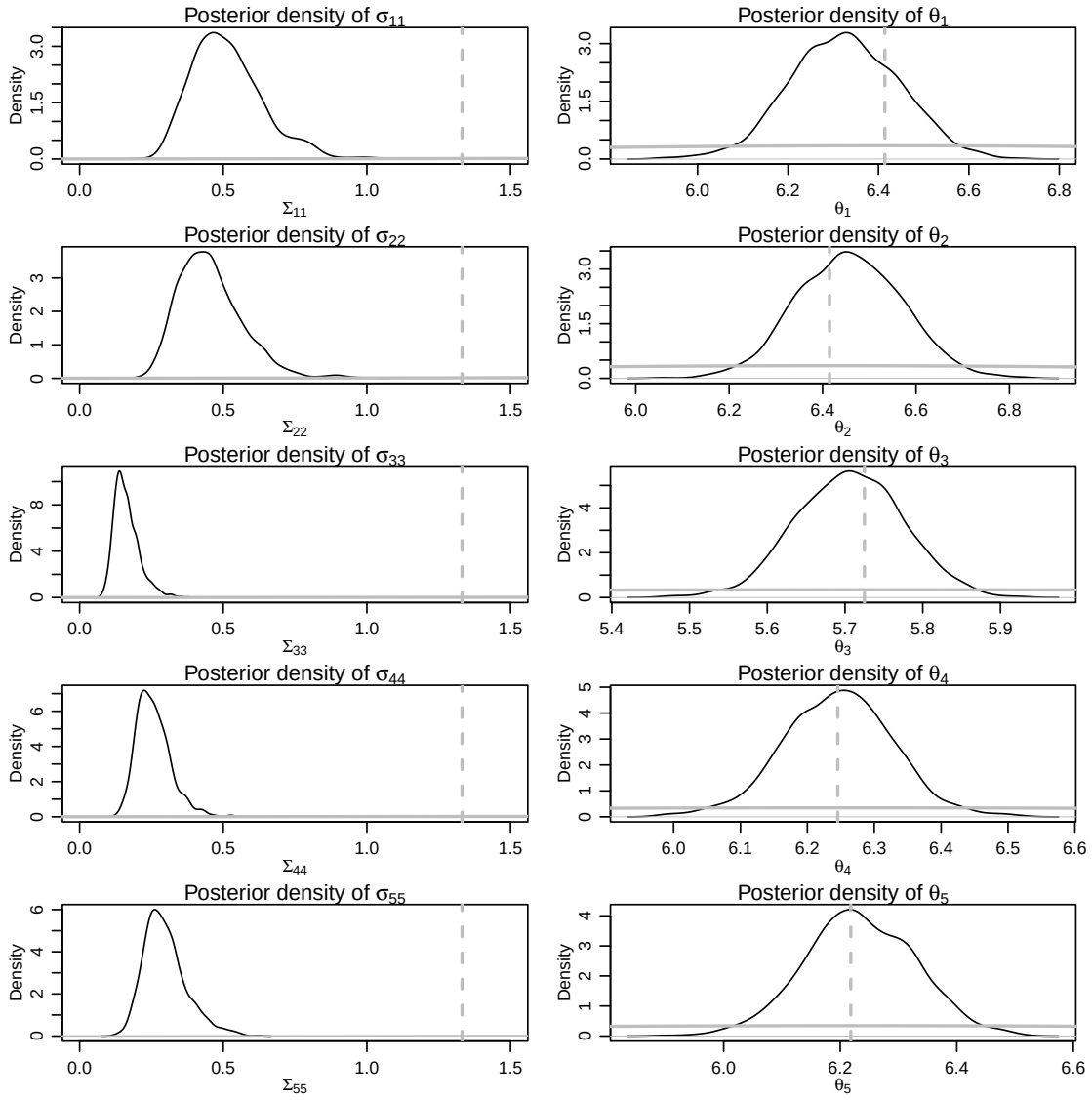


Figure 12: The posterior (and prior) distribution of $\sigma_{j,j}$ (left) and θ_j (right). Posterior distribution is ploted in black, and prior distribution is ploted in gray. The vertical dashed line marks the initial value for prior.

9.3 Source code

```
IELTS <- read.csv("IELTS.csv")
```

```
pdf("pic/IELTS.pdf")
par(mar=c(7,4.5,1,1),mgp=c(1.88,0.50,0))
plot(0, type = "n", xlim = c(1, ncol(IELTS)), ylim = range(IELTS[, -1]),
     xlab = "", ylab = "Scores", main = "Scores by Nationality")

# Add lines for each nationality
for (i in 1:nrow(IELTS)) {
```

```
  lines(1:(ncol(IELTS)-1), IELTS[i, -1], type = "o", pch = i%%10, col = i)
}

# Add legend
legend("topright", legend = IELTS[,1], col = 1:nrow(IELTS), pch = (1:nrow(IELTS))%%10,
       bty = "n", cex = 0.8, y.intersp = 0.75)

text(x = 1:5,
     y = par("usr")[3] - 0.45,
     labels = colnames(IELTS)[-1],
     xpd = NA,
     srt = 45,
     cex = 1.2,
     adj = 1)
dev.off()
```

```
## pdf
##   2
```

9.3.1 IELTS scores Missing Data Imputation By Gibbs sampler

```
#### Simulate multivariate normal vector
rmvnorm<-
function(n,mu,Sigma) {
  p<-length(mu)
  res<-matrix(0,nrow=n,ncol=p)
  if( n>0 & p>0 ) {
  E<-matrix(rnorm(n*p),n,p)
  res<-t(  t(E%*%chol(Sigma)) +c(mu))
                  }
  res
                       }
#### Simulate from the Wishart distribution
rwish<-function(n,nu0,S0)
{
  sS0 <- chol(S0)
  S<-array( dim=c( dim(S0),n ) )
  for(i in 1:n)
  {
     Z <- matrix(rnorm(nu0 * dim(S0)[1]), nu0, dim(S0)[1]) %*% sS0
     S[,,i]<- t(Z)%*%Z
  }
  S[,,1:n]
}
```

```r
Y0<-IELTS[,2:6]
Y<-Y0
n<-dim(Y)[1]
p<-dim(Y)[2]

set.seed(1)
# Create some NA values!
O<-matrix(rbinom(n*p,1,.7),n,p)
Y[O==0]<-NA

pdf("pic/stat.pdf",family="Times", height=6,width=6)
par(mar=c(1,1,.5,.5)*1.75,mfrow=c(p,p),mgp=c(1.75,.75,0))
for(j1 in 1:p) {
for(j2 in 1:p) {
 if(j1==j2){hist(Y[,j1],main="");mtext(colnames(Y)[j1],side=3,line=-.1,cex=.7)}
  if(j1!=j2) { plot(Y[,j1],Y[,j2],xlab="",ylab="",pch=16,cex=.7)}
                }}
dev.off()
```

Utility functions

```
## pdf
##   2
```

```r
## prior parameters
p<-dim(Y)[2]
# Take the population mean as the prior
mu0<-apply(Y[,c(1:5)],MARGIN=2,FUN=mean,na.rm=TRUE)
# Standard deivation: 4.0~8.5 -> 95% sigma_0 = 1.1538
sd0<-rep(1.1538, 5)
L0<-matrix(.5,p,p) ; diag(L0)<-1 ; L0<-L0*outer(sd0,sd0)
nu0<-p+2 ; S0<-L0
###

### starting values
Sigma<-S0
Y.full<-Y
for(j in 1:p)
{
  Y.full[is.na(Y.full[,j]),j]<-mean(Y.full[,j],na.rm=TRUE)
}
###

### Gibbs sampler
```

```r
THETA<-SIGMA<-Y.MISS<-NULL
set.seed(1)
GIBBS_start_time<- Sys.time()
for(s in 1:1000)
{

  ###update theta
  ybar<-apply(Y.full,2,mean)
  Ln<-solve( solve(L0) + n*solve(Sigma) )
  mun<-Ln%*%( solve(L0)%*%mu0 + n*solve(Sigma)%*%ybar )
  theta<-rmvnorm(1,mun,Ln)
  ###

  ###update Sigma
  Sn<- S0 + ( t(Y.full)-c(theta) )%*%t( t(Y.full)-c(theta) )
#  Sigma<-rinvwish(1,nu0+n,solve(Sn))
  Sigma<-solve( rwish(1, nu0+n, solve(Sn)) )
  ###

  ###update missing data
  for(i in 1:n)
  {
    b <- ( O[i,]==0 )
    a <- ( O[i,]==1 )
    iSa<- solve(Sigma[a,a])
    beta.j <- Sigma[b,a]%*%iSa
    s2.j   <- Sigma[b,b] - Sigma[b,a]%*%iSa%*%Sigma[a,b]
    theta.j<- theta[b] + beta.j%*%(t(Y.full[i,a])-theta[a])
    Y.full[i,b] <- rmvnorm(1,theta.j,s2.j )
  }

  ### save results
  THETA<-rbind(THETA,theta) ; SIGMA<-rbind(SIGMA,c(Sigma))
  Y.MISS<-rbind(Y.MISS, Y.full[O==0] )
  ###

  #cat(s,theta,"\n")
}
GIBBS_end_time<- Sys.time()

#### Posterior mean and correlation matrix
apply(THETA,2,mean)
```

```
## [1] 6.325467 6.457498 5.705664 6.245985 6.233816
```

```
COR <- array( dim=c(p,p,1000) )
for(s in 1:1000)
{
  Sig<-matrix( SIGMA[s,] ,nrow=p,ncol=p)
  COR[,,s] <- Sig/sqrt( outer( diag(Sig),diag(Sig) ) )
}

apply(COR,c(1,2),mean)
```

```
##           [,1]      [,2]      [,3]      [,4]      [,5]
## [1,] 1.0000000 0.8834623 0.7915787 0.8035453 0.8782880
## [2,] 0.8834623 1.0000000 0.8206582 0.8060882 0.8836628
## [3,] 0.7915787 0.8206582 1.0000000 0.7376627 0.8145011
## [4,] 0.8035453 0.8060882 0.7376627 1.0000000 0.8238962
## [5,] 0.8782880 0.8836628 0.8145011 0.8238962 1.0000000
```

```
return
```

```
## .Primitive("return")
```

```
##### Plot the Predicted vs True value
pdf("pic/Gibbs_PredvsTrue.pdf",family="Times", height=14,width=4)
Y.true<-Y0
V<-matrix(1:p,nrow=n,ncol=p,byrow=TRUE)

v.miss<-V[O==0]
y.pred<-apply(Y.MISS,2,mean)
y.true<-Y.true[O==0]
par(mfrow=c(5,1),mar=c(3,3,1,1),mgp=c(1.75,.75,0))
for(j in 1:p){ plot(y.true[v.miss==j], y.pred[v.miss==j],
          xlab=paste("true", colnames(Y.true)[j]),
          ylab=paste("predictied", colnames(Y.true)[j]),pch=16 )
          abline(0,1)
mse <- round(mean((y.true[v.miss==j]- y.pred[v.miss==j])^2),4)
text(x=min(y.true[v.miss==j]),y=max(y.pred[v.miss==j]),paste("MSE:", mse, sep=""),
     col="BLUE", cex=1.5, adj=0,pos=4)
#cat(j, mean( (y.true[v.miss==j]- y.pred[v.miss==j])^2),"\n")
}
MSE_GIBBS <- mean((y.true- y.pred)^2)
dev.off()
```

```
## pdf
##   2
```

```
#### Convert SIGMA to an array of correlation parameters
COR<-array(dim=c(p,p,dim(SIGMA)[1]) )
for(s in 1:dim(SIGMA)[1]) {
```

```
Sig<-matrix( SIGMA[s,] ,nrow=p,ncol=p)
COR[,,s] <- Sig/sqrt(outer( diag(Sig),diag(Sig)))
                              }
colnames(COR)<-rownames(COR)<-colnames(Y)



#### Function for posterior quantile intervals for matrices
#### From the `sbgcop` package
plotci.sA<-function(sA, ylabs = colnames(sA[, , 1]), mgp = c(1.75, 0.75,
    0))
{
    qA <- qM.sM(sA)
    p <- dim(qA)[1]
    tmp <- c(qA)
    tmp <- tmp[tmp != 1]
    par(mgp = mgp)
    for (j in 1:p) {
        plot(0, 0, type = "n", ylim = range(c(tmp), na.rm = TRUE),
            xlim = c(1, p), ylab = ylabs[j], xaxt = "n", xlab = "")
        points((1:p)[-j], qA[j, -j, 2], pch = 16, cex = 0.6)
        segments(x0 = (1:p)[-j], y0 = qA[j, -j, 1], x1 = (1:p)[-j],
            y1 = qA[j, -j, 3])
        abline(h = 0, col = "gray")
        abline(v = j, col = "gray")
    }
    axis(side = 1, at = 1:p, labels = colnames(qA[, , 1]), las = 2)
}

sR.sC<-function(sC)
{
    p <- dim(sC)[1]
    s <- dim(sC)[3]
    sR <- array(dim = c(p, p, s))
    dimnames(sR) <- dimnames(sC)
    for (l in 1:s) {
        C <- sC[, , l]
        R <- C * NA
        for (j in 1:p) {
            R[j, -j] <- C[j, -j] %*% solve(C[-j, -j])
        }
        sR[, , l] <- R
    }
    sR
```

```
}

qM.sM<-function (sM, quantiles = c(0.025, 0.5, 0.975))
{
    p1 <- dim(sM)[1]
    p2 <- dim(sM)[2]
    s <- dim(sM)[3]
    qM <- array(dim = c(p1, p2, length(quantiles)))
    dimnames(qM) <- list(dimnames(sM)[[1]], dimnames(sM)[[2]],
        paste(quantiles * 100, rep("% quantile", length(quantiles)),
            sep = ""))
    for (l in 1:length(quantiles)) {
        qM[, , l] <- apply(sM, c(1, 2), quantile, prob = quantiles[l],
            na.rm = TRUE)
    }
    qM
}




#### Figure 7.4
pdf("pic/Gibbs_CI.pdf",height=6,width=6,family="Times")

par(mfcol=c(5,2),mar=c(3,2.75,1,1),mgp=c(1.75,.75,0),oma=c(1.5,0,0,0))
plotci.sA(COR)

REG<-sR.sC(COR)
plotci.sA(REG)
dev.off()
```

```
## pdf
##   2
```

```
CQ<-apply(COR, c(1,2), quantile,prob=c(.025,.5,.975) )

round(CQ[1,,],2)
```

```
##           Reading Listening Writing Speaking Overall
## Reading      1.00      0.78    0.63     0.65    0.77
## Listening    0.78      1.00    0.67     0.65    0.79
## Writing      0.63      0.67    1.00     0.55    0.66
## Speaking     0.65      0.65    0.55     1.00    0.67
## Overall      0.77      0.79    0.66     0.67    1.00
```

```r
round(CQ[2,,],2)
```

```
##           Reading Listening Writing Speaking Overall
## Reading      1.00      0.89    0.80     0.81    0.89
## Listening    0.89      1.00    0.83     0.82    0.89
## Writing      0.80      0.83    1.00     0.75    0.82
## Speaking     0.81      0.82    0.75     1.00    0.83
## Overall      0.89      0.89    0.82     0.83    1.00
```

```r
round(CQ[3,,],2)
```

```
##           Reading Listening Writing Speaking Overall
## Reading      1.00      0.94    0.89     0.90    0.94
## Listening    0.94      1.00    0.91     0.90    0.95
## Writing      0.89      0.91    1.00     0.86    0.91
## Speaking     0.90      0.90    0.86     1.00    0.91
## Overall      0.94      0.95    0.91     0.91    1.00
```

```r
round(apply(COR,c(1,2),mean),2)
```

```
##           Reading Listening Writing Speaking Overall
## Reading      1.00      0.88    0.79     0.80    0.88
## Listening    0.88      1.00    0.82     0.81    0.88
## Writing      0.79      0.82    1.00     0.74    0.81
## Speaking     0.80      0.81    0.74     1.00    0.82
## Overall      0.88      0.88    0.81     0.82    1.00
```

```r
pdf("pic/Posterior_vs_Prior.pdf")
par(mfrow = c(5,2), mar=c(3,3,1,1),mgp=c(1.75,.75,0))

S0_prior <- rwish(n=1000, nu0=nu0, S0)

plot(density(SIGMA[,1]),
     main=expression(paste("Posterior density of ", sigma[11],sep="")),
     xlab=expression(Sigma[11]), xlim=c(0,1.5))
lines(density(S0_prior[1,1,]),col="gray", lwd=2)
abline(v=S0[1], lty=2, col="GRAY", lwd=2)

plot(density(THETA[,1]),
     main=expression(paste("Posterior density of ", theta[1],sep="")),
     xlab=expression(theta[1]))
x_theta <- seq(4, 9, length.out=100)
lines(x_theta, dnorm(x_theta, mean=mu0[1], sd=sqrt(S0[1])),
      col="GRAY", lwd=2)
abline(v=mu0[1], lty=2, col="GRAY", lwd=2)
```

```
plot(density(SIGMA[,5*1+2]),
     main=expression(paste("Posterior density of ", sigma[22],sep="")),
     xlab=expression(Sigma[22]), xlim=c(0,1.5))
lines(density(S0_prior[2,2,]),col="gray", lwd=2)
abline(v=S0[5*1+2], lty=2, col="GRAY", lwd=2)

plot(density(THETA[,2]),
     main=expression(paste("Posterior density of ", theta[2],sep="")),
     xlab=expression(theta[2]))
x_theta <- seq(4, 9, length.out=100)
lines(x_theta, dnorm(x_theta, mean=mu0[2], sd=sqrt(S0[5*1+2])),
      col="GRAY", lwd=2)
abline(v=mu0[2], lty=2, col="GRAY", lwd=2)

plot(density(SIGMA[,5*2+3]),
     main=expression(paste("Posterior density of ", sigma[33],sep="")),
     xlab=expression(Sigma[33]), xlim=c(0,1.5))
lines(density(S0_prior[3,3,]),col="gray", lwd=2)
abline(v=S0[5*2+3], lty=2, col="GRAY", lwd=2)

plot(density(THETA[,3]),
     main=expression(paste("Posterior density of ", theta[3],sep="")),
     xlab=expression(theta[3]))
x_theta <- seq(4, 9, length.out=100)
lines(x_theta, dnorm(x_theta, mean=mu0[3], sd=sqrt(S0[5*2+3])),
      col="GRAY", lwd=2)
abline(v=mu0[3], lty=2, col="GRAY", lwd=2)

plot(density(SIGMA[,5*3+4]),
     main=expression(paste("Posterior density of ", sigma[44],sep="")),
     xlab=expression(Sigma[44]), xlim=c(0,1.5))
lines(density(S0_prior[4,4,]),col="gray", lwd=2)
abline(v=S0[5*3+4], lty=2, col="GRAY", lwd=2)

plot(density(THETA[,4]),
     main=expression(paste("Posterior density of ", theta[4],sep="")),
     xlab=expression(theta[4]))
x_theta <- seq(4, 9, length.out=100)
lines(x_theta, dnorm(x_theta, mean=mu0[4], sd=sqrt(S0[5*3+4])),
      col="GRAY", lwd=2)
abline(v=mu0[4], lty=2, col="GRAY", lwd=2)

plot(density(SIGMA[,5*4+5]),
     main=expression(paste("Posterior density of ", sigma[55],sep="")),
```

```
     xlab=expression(Sigma[55]), xlim=c(0,1.5))
lines(density(S0_prior[5,5,]),col="gray")
abline(v=S0[5*4+5], lty=2, col="GRAY", lwd=2)

plot(density(THETA[,5]),
     main=expression(paste("Posterior density of ", theta[5],sep="")),
     xlab=expression(theta[5]))
x_theta <- seq(4, 9, length.out=100)
lines(x_theta, dnorm(x_theta, mean=mu0[5], sd=sqrt(S0[5*4+5])),
      col="GRAY", lwd=2)
abline(v=mu0[5], lty=2, col="GRAY", lwd=2)
dev.off()
```

```
## pdf
##   2
```

9.3.2 IELTS scores Missing Data Imputation By Gibbs sampler

```
em_algorithm <- function(data, max_iter = 100, tol = 1e-6) {
    # Initial estimates
    means <- apply(data, 2, function(x) mean(x, na.rm = TRUE))
    sigma <- cov(data, use = "complete.obs")

    # Iteration control
    iter <- 1
    conv <- FALSE

    while (iter <= max_iter && !conv) {
        # E-step: Estimate missing values
    #print(iter)
        data_imputed <- data
        for (i in 1:nrow(data)) {
            missing <- is.na(data[i,])
            if (any(missing)) {
                obs <- !missing
                mu_obs <- means[obs]
                mu_miss <- means[missing]
                Sigma_oo <- sigma[obs, obs]
                Sigma_om <- sigma[obs, missing]
                Sigma_mm <- sigma[missing, missing]
                Sigma_mo <- t(Sigma_om)
            s2.j   <- Sigma_mm - Sigma_mo%*%solve(Sigma_oo)%*%Sigma_om

                # Conditional mean and covariance
```

```
                cond_mean <- mu_miss + (Sigma_mo %*% solve(Sigma_oo)) %*% (t(data[i, obs
                data_imputed[i, missing] <- rmvnorm(1, cond_mean, s2.j)
            }
        }

        # M-step: Re-estimate parameters
        old_means <- means
        old_sigma <- sigma

        means <- colMeans(data_imputed, na.rm = TRUE)
        sigma <- cov(data_imputed, use = "pairwise.complete.obs")

        # Check for convergence
        conv <- max(abs(old_means - means), abs(sigma - old_sigma)) < tol
        iter <- iter + 1
    }

    return(list(means = means, sigma = sigma, data_imputed = data_imputed))
}
```

```
EM_start_time<- Sys.time()
em_result <- em_algorithm(Y)
EM_end_time<- Sys.time()
```

```
##### Plot the Predicted vs True value
pdf("pic/EM_PredvsTrue.pdf",family="Times", height=14,width=4)
Y.true<-Y0
V<-matrix(1:p,nrow=n,ncol=p,byrow=TRUE)

v.miss<-V[O==0]
y.pred <- em_result$data_imputed
y.pred <- y.pred[O==0]
y.true<-Y.true[O==0]
MSE_EM<- mean((y.true- y.pred)^2)
par(mfrow=c(5,1),mar=c(3,3,1,1),mgp=c(1.75,.75,0))
for(j in 1:p){ plot(y.true[v.miss==j], y.pred[v.miss==j],
          xlab=paste("true", colnames(Y.true)[j]),
          ylab=paste("predictied", colnames(Y.true)[j]),pch=16 )
          abline(0,1)
mse <- round(mean((y.true[v.miss==j]- y.pred[v.miss==j])^2),4)
text(x=min(y.true[v.miss==j]),y=max(y.pred[v.miss==j]),paste("MSE:", mse, sep=""),
     col="BLUE", cex=1.5, adj=0,pos=4)
  cat(j, mean( (y.true[v.miss==j]- y.pred[v.miss==j])^2),"\n") }
```

```
## 1 0.1027275
```

```
## 2 0.03862703
## 3 0.03020887
## 4 0.249199
## 5 0.02529312
```

```
dev.off()
```

```
## pdf
##   2
```

```
print(MSE_GIBBS)
```

```
## [1] 0.03895665
```

```
print(GIBBS_end_time - GIBBS_start_time)
```

```
## Time difference of 10.27271 secs
```

```
print(MSE_EM)
```

```
## [1] 0.06930736
```

```
print(EM_end_time - EM_start_time)
```

```
## Time difference of 1.245525 secs
```