Comparing Standard Error Estimators Using Laplace Approximated Maximum Likelihood in Multidimensional Item Response Theory Models

Munyaka Mutua

Centre for Educational Measurement, University of Oslo

Master of Science in Assessment , Measurement and Evaluation

MAE4090: Master Thesis

Associate Professor Björn Andersson

May 12, 2023

Popular Abstract

Accurate estimation of standard errors is key to making correct inferences and comparisons. For correct and precise estimates, smaller values of standard errors are desired otherwise this could lead to unjustified certainty. When Laplace approximations are used it is unclear which standard error procedures should be used to get robust and precise standard errors. This study compares the different standard error estimators suitable with Laplace approximations through a Monte-Carlo simulation. The study sought to evaluate the standard error procedures in terms of accuracy, precision and computations efficiency as well as investigate the average coverage rate of the 95% confidence interval. The results showed that of all the five procedures investigated, using the first order Laplace produced biased estimates whereas second order Laplace produced more precise and accurate estimates. The findings add knowledge to the literature on standard error estimators using Laplace approximations.

Acknowledgements

I would like to thank my esteemed supervisor Professor Björn Andersson for his invaluable supervision, support, and tutelage throughout the master thesis project. My appreciation also goes out to my family and friends for their encouragement and support all through my studies.

Abstract

This research was conducted to address the paucity of comparative studies on different methods for estimating standard errors with Laplace approximated maximum likelihood in multidimensional item response theory models. We considered standard errors associated with the observed information matrix, a fast version of the observed information matrix and the empirical cross product matrix, along with the Sandwich estimators derived from the observed information matrix and the fast observed information matrix. This study compares the accuracy, precision, computational efficiency and the average coverage rate of the 95% confidence interval of the different standard error methods. A Monte-Carlo simulation was conducted to investigate the effect of samples size, test length, number of categories and model complexity. The simulation was based on a fully crossed design with two test lengths (4/8), three sample sizes (250/1,000/4,000), two model types (independent cluster and cross-loadings), and 2/5 number of categories (binary response and polytomous) resulting in 24 data generating conditions which all used a three-dimensional latent variable vector. The standard error estimators were evaluated in terms of accuracy, precision and computation efficiency using the coverage rates of the 95% confidence intervals, average root mean squared and the average absolute bias. In terms of average absolute bias, and average root mean squared error no method was found unacceptable, they all had close to zero values. The empirical cross product matrix was found to be more computationally efficient compared to other methods. In relation to 95% confidence interval, the average coverage rate for methods using first order Laplace were lower than the nominal level hence biased and imprecise estimates across all conditions. Standard error methods estimated using second order Laplace produced precise and accurate estimates with the correct coverage rates. This study adds knowledge to the literature on standard error estimators with Laplace approximations.

Key words: Standard Errors, Laplace approximations, Multidimensional Item Response Theory,

Confidence Intervals.

Comparing Standard Error Estimators Using Laplace Approximated Maximum Likelihood in Multidimensional Item Response Theory Models

The standard error (SE) of a statistic is the standard deviation of the sampling distribution of the estimator and is used as a measure of the degree of precision of the statistic (Vogt, 2022). SE of zero means that the statistic has no random error, whereas a large SE implies an unreliable statistic. There is not a statistical model that is a perfect reflection of the data it summarizes, but a simplification with beneficial characteristics (Wainer & Thissen, 1987). In statistical modelling SEs help capture this imperfection and can in addition be used to construct confidence intervals and carry out statistical significance tests. In the context of item response theory (IRT), SEs are an indication of the degree of precision with which item parameters are estimated (Thissen & Wainer, 1982). One method used to construct a 95% confidence interval (CI) for a parameter is to compute symmetric interval based standard normal quantiles and the estimated standard error for the parameter (Hays, 1988). Then we can say that a 95% C.I. $(a; b)$ for a parameter θ means that the parameter θ is covered by such an interval 95% of the time if the sampling would be repeated infinitely (Hoekstra et al., 2014). Therefore, smaller SE values are preferred because the smaller the SEs the greater the precision in our estimated parameters (Vogt, 2022).

Item response theory or generalized linear latent variable models usually use marginal maximum likelihood estimation to estimate latent variables (Andersson et al., 2023). The challenge with most IRT models is that the integrals are not tractable analytically and must be approximated when using maximum likelihood. When dealing with models with one or two latent variables, Gauss-Hermite quadrature approximations (Bock & Aitkin, 1981), are highly effective, however their efficiency rapidly declines when working with models with more latent variables(Andersson et al., 2023). Among other suggested solutions in IRT literature, are two

estimation approaches used in this study i.e., the first order Laplace and the second order Laplace. We focused on these two estimation approaches since many of the SE methods work only with Laplace approximations and have not been implemented in other programming software's except the Lamle package. Andersson et al. (2023) developed an estimation procedure that uses the second order Laplace approximation to the marginal loglikelihood function estimations of generalized linear latent variable models using binary or and polytomous data. An important feature of the developed algorithm especially for the current study is that it supports several multidimensional IRT structures.

Considering the crucial role of SEs in IRT model parameters and the paucity of studies examining SE procedures in IRT there is a need for more comprehensive studies to investigate the performance of SEs. Hence, one may be interested in carefully investigating the assessment of the various procedures of generating SEs in IRT. Some previous studies have been conducted, however some of the earlier studies focused on unidimensional IRT models using dichotomous data with few studies on multidimensional IRT models using polytomous data (Monroe, 2019; Paek & Cai, 2014; Yuan et al., 2014).

The current study differs from earlier studies by considering multi-dimensional IRT with dichotomous and polytomous data. The study centers on the direct comparisons of SEs from five different approaches using maximum likelihood estimates (MLEs) with Laplace approximation in higher dimensional IRT. It complements a recent study by (Andersson et al., 2023) by providing a comprehensive evaluation on the differences and similarities in performance of SEs and confidence intervals in higher order IRT models using Laplace approximations.

In IRT literature, there are several SE approaches available, and all have different computational demands associated with them. For instance, the expected Fisher information

matrix is considered the gold standard in estimating the error covariance matrix. However, it has a major drawback in that it imposes a heavy computational burden for many realistic test lengths (Paek & Cai, 2014). This emanates from the fact that its calculation requires the expectation be taken over all possible response patterns. Since the differentiation methods for approximating the expected Fisher information matrix increases exponentially as the number of items increases this renders the expected Fisher information matrix practically infeasible in the case of large number of items and many categories (Paek & Cai, 2014). On the other hand, the number of items and the differentiation methods of estimating the observed information matrix have a linear relationship suggesting that the observed information matrix is a more practical approach for item analysis in education and psychological settings. For instance, the empirical cross product information matrix is an example of an observed information matrix that only contains the response patterns to the data and is much simpler to calculate (Lin, 2018).

For this study, we considered SEs associated with the Observed information matrix(M1), a fast version of the observed information matrix (M2) and the Empirical cross product matrix (M3), along with the Sandwich estimators derived from M1 and M2 herein called the Sandwich estimator from the observed information matrix (M4) and the Sandwich estimator from the fast observed information matrix (M5). These were considered because they are suitable with Laplace approximations and are implemented in available software for the models considered in this study. M1 and M2 are obtained through numerical differentiation of the observed gradient with respect to the unknown parameters. In the study by Andersson and Xin (2021), it was found that using standard errors based on M1 yielded accurate results for independent-cluster models as long as they were specified correctly. In additional research, Andersson et al. (2023), showed that this approach is also suitable for cross-loading models. To obtain this approximation, an

objective function is defined using the unknown parameters as a vector-valued input argument. This function then computes and returns the exact observed gradient of the approximated log-likelihood by updating the mode for each response pattern based on the input parameters before computing the gradient. This approach provides an estimation of the second derivatives of the approximated log-likelihood. The numerical differentiation of M1 includes updating the mode, whereas M2 does not. This makes M2 computationally efficient compared to M1. Since this study is about the comparison of the different SE procedures, we omit the mathematical details for the different methods and instead refer to the existing literature for the mathematical details (Andersson et al., 2023).

The importance of this study is twofold. First, it adds to knowledge to the literature on the performance of SEs estimated from different information matrices. Second, knowledge on the performance of SEs or error covariance matrices is important in informing the selection of Wald test statistics for differential item functioning, model selection at the item level, overall goodness-of-fit statistics, test scoring accounting for uncertainty in item parameter calibration and the calculation of approximate confidence intervals for item parameters. (Cai & Hansen, 2013; Li & Wang, 2015; Liu et al., 2019, 2019; Ma et al., 2016; Maydeu-Olivares & Joe, 2005). Specifically, the study sought to answer the following research questions.

1. When using Laplace approximations, how do the above SE methods compare across different varying conditions (i.e., sample sizes, test length, number of categories and model complexities) in terms of computational efficiency, accuracy, and precision?
2. In practice, which estimator of SEs should be used for estimators based on Laplace approximations?

3. What is the empirical coverage rate of 95% confidence intervals when using the different methods for computing the standard errors?

The rest of this article is structured as follows. In the next subsections, there will be a brief background provided on multidimensional models for dichotomous and polytomous data, previous studies on SEs in IRT, the importance of SEs in IRT, and methods of estimating SEs in IRT. Following this, the methodology section will provide an account of the proposed research methods and feasibility, as well as the evaluation criterion for standard errors and confidence intervals. Lastly, the results will be reported and discussed.

Key Concepts Associated with SEs in IRT

Multidimensional Models for Dichotomous and Polytomous Data

Multidimensional IRT (MIRT) is an extension of the unidimensional IRT models that were developed to portray an individual's likelihood of a correct response based on item parameters and multiple latent traits (Reckase & Reckase, 2009). MIRT models are classified into two categories called compensatory and non-compensatory models (Bonifay, 2020). To illustrate the difference between the two types, assume a test of the two dimensions algebra and arithmetic proficiency. If in solving a mathematical problem the higher level trait say of algebra proficiency compensates for the low level of arithmetic proficiency, then we have a compensatory model. On the other hand, non-compensatory models restrict an examinee's standing across the multidimensional space such that an examinee's proficiency on one latent trait does not compensate for the lack in another latent trait needed for correctly endorsing an item. The current study focuses on compensatory models since they are more common within IRT literature (Immekus et al., 2019).

MIRT models can be applied to binary data as well as polytomous data. MIRT models for binary data include the multidimensional two parameter logistic model (2PL) and the

multidimensional three parameter logistic model (3PL). Polytomous MIRT models include the multidimensional graded response model (GRM), the multidimensional generalized partial credit model (GPCM) and many more. As mentioned earlier, a distinguishing feature in this study is the use of dichotomous and polytomous data. This had a direct implication on the choice of the IRT models. The 2PL model was chosen because it is suitable for estimating IRT models with binary data, whereas the GRM model was chosen because it is suitable for estimating IRT models with polytomous data (Zanon et al., 2016).

Previous Studies on Standard Errors

There have been various studies on SEs in the past, motivated by the recognition of the importance of SEs of IRT model parameters. Tsutakawa (1984) investigated the EM algorithm used to derive maximum likelihood (ML) and provided details on obtaining SEs using the observed information matrix. Yuan et al. (2014) studied information matrices and SEs for ML estimates of IRT model parameters and showed that SEs from the observed information matrix are robust, but not under all conditions. For instance, if the model is not correctly specified, only the sandwich estimator gives consistent SEs. Paek & Cai (2014) conducted a study on the comparison of SEs for IRT models based on three covariance matrices i.e., Fisher information, empirical cross-product, and supplemental expectation maximization. The results of their study show that all three methods give similar results in relation to the bias in the SEs. Andersson & Xin (2021) in their study, used the second-order Laplace with maximum likelihood and showed that the cross-product matrix and the observed information matrix produce SEs that are approximately equally accurate.

As mentioned earlier, intractable integrals pose a challenge to marginal maximum likelihood estimation and some suggested solutions in the literature include Gauss-Hermite quadrature (Bock & Aitkin, 1981) and adaptive Gauss-Hermite quadrature (Cagnone & Monari,

2013; Schilling & Bock, 2005); however, these are only efficient with a few latent variables and quickly decrease in efficiency with higher dimensions. Other suggested solutions include simulation-based approaches like a Monte-Carlo and the Metropolis-Hastings Robbins-Monro method (Cai, 2010), but these are slow to converge in the case of small sample sizes (Andersson et al., 2023).

Another solution is to use the first-order and the second-order Laplace approximations (Andersson & Xin, 2021; Huber et al., 2004; Joe, 2008). In IRT, the Laplace approximation is a method for estimating the item parameters (such as item difficulty and discrimination) to obtain approximate MLEs. The first-order Laplace (Lap1) approximation uses an asymptotic expansion to approximate the required integrals and is equivalent to the adaptive quadrature with only one quadrature point per dimension. Since the computational demand of Lap1 increases linearly with increasing dimensionality it is considered an efficient method in estimating high dimensional models (Andersson & Xin, 2021). However, inaccuracy of approximation is a problem, especially when there are few observed variables per dimension and for dichotomous observed variables (Joe, 2008). Higher order Laplace approximations such as the second order Laplace (Lap2) have been proposed to improve computational accuracy (Shun, 1997). Both Lap1 and Lap2 approximations can be implemented using software packages such as R or Mplus. However, it should be noted that the accuracy of the estimates may depend on the sample size, the number of response options, and the distribution of the latent trait.

Some types of IRT models (Rasch-like models for binary and ordinal data) fit within what is called generalized linear mixed models/generalized linear random effects models. Two such approaches have used 2nd-order Laplace approximations. For instance, Noh and Lee (2007) proposed a statistically and computationally efficient restricted maximum likelihood (REML)

procedure for the analysis of dichotomous data and showed how the REML can be modified to be applied over a wide class of models and design structures. Similarly, Raudenbush et al. (2000), in recognizing the challenge in computing integrals without an explicit solution for the marginal maximum likelihood, proposed a solution that applies to generalized linear models with nested random effects. Their strategy involved the approximation of the log of the integrand via its fully multivariate Taylor expansion of higher order and integration approach using Laplace method. Their results showed that the higher-order Laplace approach is remarkably accurate and computationally fast (Raudenbush et al., 2000). However, these two papers do not specifically discuss how to estimate standard errors.

Importance of Standard Errors In IRT

The advancement of IRT has made models popular and applications of IRT are common in practical testing. Some areas where IRT models are applied include test equating, test design, and evaluation of measurement invariance (De Ayala, 1995). Standard errors are measures of precision for an estimate and are inversely related to the sample size, with smaller samples having larger standard errors and larger samples having smaller standard errors (Thissen & Wainer, 1982). This has a direct implication on the test design, for instance it means a well-designed test will make use of a large sample to reduce standard errors. This underlines the important relationship between standard errors and test designs for correct and precise estimates.

Test equating is a statistical procedure used to adjust test scores on different test forms so that the scores from different test forms are comparable (Kolen & Brennan, 2004). Irrespective of the methods of test equating to be used, the standard errors of equating should be reported to gauge the precision of the converted scores. It then follows that proper estimation of standard errors is essential in test equating.

Lastly, SEs are key in the evaluation of differential item functioning (DIF), item parameter drift (IPD) and model goodness of fit evaluation (Woods et al., 2013). For example, when measurement invariance is not achieved, standard errors are likely to be higher, indicating greater uncertainty in the ability estimates. In sum, accurate estimation of SEs is integral to making correct inferences and comparisons, therefore, consistent SEs are important elements of any statistical methods (Yuan et al., 2014) and when estimated incorrectly, they can lead to unjustified certainty.

Methods

To assess the performance of the five methods for calculating SEs we conducted a simulation study. The statistical software R (R. C. Team, 2022) and the lamle package (Andersson B., and Jin S., 2022) was used in the analysis of this study. Four factors were varied in this simulation i.e., test length, sample size, model type, number of latent variables and number of categories. The simulation was based on a fully crossed design with two test lengths (4/8), three sample sizes (250/1,000/4,000), two model types (independent cluster and cross-loadings), t factors, and 2/5 number of categories (binary response and polytomous) resulting in 2x3x2x2=24 data generating conditions which all used a three-dimensional latent variable vector. Under each condition 1,000 replications were conducted. Figure 1 gives an illustration of the independent cluster models with 12 observed variables used in this study while Figure 2 shows the cross-loadings models with 12 observed variables. The latent variables are represented by the ellipses whereas rectangles are used to denote the observed variables. The covariances between the latent variables are shown by solid lines connected with arrows on both sides. Main factor loadings are represented by solid lines whereas dotted lines are used to show cross-loadings.

Simulation Conditions

Test Length

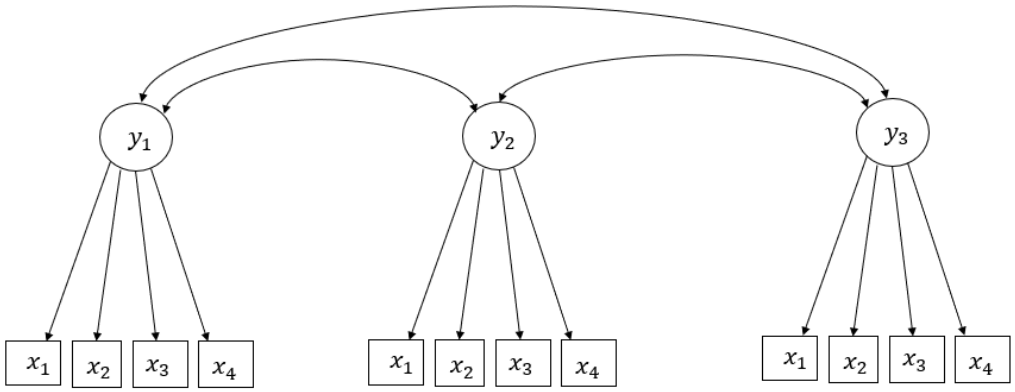
Previous research has demonstrated that test length has an effect on the accuracy and precision of an examinee's ability estimates (Sahin & Anil, 2017). A test should not be too long else it leads to fatigue on the examinees and should not be composed of few items either (too short) else it will not be able to provide a reliable measure of the examinee's position in the latent space. Test length is important because it has direct implications on the magnitude of the SEs with poorly designed tests e.g., short, and biased tests tend to produce large SEs which are indications of inaccurate and imprecise estimates. On the other hand, well designed tests, and appropriate sampling techniques will tend to produce smaller SEs hence more accurate and precise estimates. The dimensionality of the error covariance matrix is also dependent on the number of estimated item parameters i.e., the computational burden of the methods for calculating SEs increases as the dimensionality of the error covariance matrix increases (Lin, 2018). In this study we considered four and eight items per dimension for a total of either 12 or 24 items for the three-dimensional models.

Sample Size

Sample size is inversely proportional to the SE, hence the larger the sample size the smaller the SE. Thissen and Wainer (1982) shows that for samples greater than 500 the SEs will be less than or equal to 0.1. However, this is conditional on the model fitting the data well and that the item discriminations are homogenous and sufficiently large. If the item discriminations are not homogeneous and sufficiently large one would need larger sample sizes to achieve the same levels of precision. In this study we varied the samples between 250, 1000, and 4,000.

Figure 1

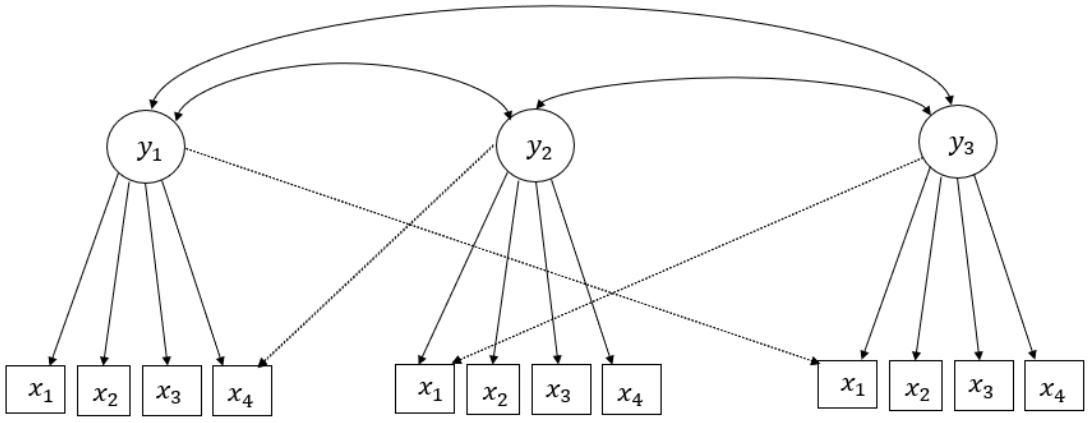
Illustration of Independent Cluster Models with 4 Items per Dimension



Note: y_1 is latent variable one, y_2 =latent variable two, y_3= latent variable three, whereas x_1=observable variable one or item number one, x_2 = observable variable two, x_3= observable variable three, x_4 = observable variable four.

Figure 2

Illustration of Cross-Loadings Models with Four Items per Dimension



Note: y_1 is latent variable one, y_2 =latent variable two, y_3= latent variable three, whereas x_1=observable variable one or item number one, x_2 = observable variable two, x_3= observable variable three, x_4 = observable variable four.

The discrimination parameters for the simulations will be selected from the U(1, 2)-distribution and the difficulty parameters from the N(0, 1)-distribution. These distributions were considered to mimic realistic item parameters used in standardized testing.

Model Complexity

Here we considered the type of the multidimensional model i.e., simple structure (between-item) and complex structure (within-item). For both the simple structure and the complex structure we considered three-dimensional tests. Four and eight observed variables per cluster were considered, for a total of 12/24 indicators respectively. In the complex structure one indicator in each cluster was allowed to cross load to another cluster or dimension to reflect the assumption that some indicators measure more than one latent variable. For instance, in the case of 3 latent variables with 12 indicators we had a total of 3 cross-loadings.

We assume that ability parameters are drawn from a multivariate normal distribution, $\boldsymbol{\theta} \sim MVN(0, \boldsymbol{\Sigma})$ where is $\boldsymbol{\Sigma}$ the variance covariance matrix of the abilities. The data were simulated from a two-parameter logistic model and a graded response model, estimating the parameters with the 2-PL model and the GRM model, respectively. For all estimated models the mean, and the variance of the first latent variable is fixed to 0 and 1, whereas all the other parameters are freely estimated.

Data Generation

The data was generated using R programming software version 4.2.2. (R. C. Team, 2022). GRM and 2-PL was used to simulate polytomous and binary item responses for estimating item parameters with slope and intercept parameters.

Evaluation Criterion for SEs and CIs

Following the example of Andersson et al. (2023), we used average convergence rates, average absolute bias, average root mean square error (RMSE), average estimation time, and average coverage rate of a 95% confidence intervals estimated with SEs from each of the SE methods (M1 to M5) as the evaluation criterion. Only simulation methods whose convergence rates were greater than 50% are reported. To correctly assess the computational efficiency of the SE procedures we used the time information in the estimation procedures. The absolute bias and root mean squared error for a parameter θ with estimate $\widehat{\theta_i}$in replication i are defined as $|\text{bias}|_\theta = \frac{1}{R}\left|\sum_{i=1}^{R}(\widehat{\theta_i} - \theta)\right|$ and $\text{RMSE}_\theta = \sqrt{\sum_{i=1}^{R}(\widehat{\theta_i} - \theta)^2}\,/R$ respectively (Andersson et al., 2023). Since the true standard errors are unknown, we used the Monte-Carlo standard errors from the standard deviation of the parameter estimates in the simulation as proxies for the true values. For higher accuracy values closer to zero are preferred. A 95% CI for a parameter is constructed by calculating $x(par) \pm y(se) * z(0.975)$, where *x(par)* denotes model parameter estimates and *y(se)* denotes SEs from one of the five SE methods whereas *z(0.975)* is the 0.975 quantile of the standard normal distribution. By letting X_r denote the outcome (0,1) of the estimated CI covering the true value in a replication *r* we obtain the estimate of the coverage rate as $\hat{p} = \sum_{r=1}^{R}\frac{X_r}{R}$. Generally, the evaluation criterion involved computing overall measures of recovery of model parameters i.e., slopes, intercepts, variances and covariances in terms of above-mentioned criteria with SEs from each of the five methods of estimating SEs.

Results

In this subsection we present the results of the simulations. The outcome measures are summarized in Table 1 to Table 6 whereas the covariance matrix, and the item parameters used

in the simulations are presented in the appendix as Table A1 to Table A5. The condition for sample size 250 and 12 items had over 50% nonconvergence so we decided to remove this condition in the analysis.

Accuracy, Precision and Computational Efficiency of Standard Errors

The average absolute bias and the average root mean squared error (RMSE) for all the parameters for the 2-PL independent cluster models is summarized in Table 1. In general, small sample sizes showed higher bias values compared to larger sample sizes i.e., average absolute bias exhibited by the SE methods decreased as sample size increased. The results similarly suggest that varying the test length had a similar effect, i.e., as test length increased the bias also decreased. All SE methods produced close to zero average absolute bias except for M3 in the 250 sample size and the sample size 1,000 with 12 items which had slightly higher biases. On average M3 showed slightly higher average absolute bias compared to other SE methods when Lap1 was used. On the other hand, with Lap2 estimation method, it was M5 that on average depicted slightly higher average absolute bias compared to other SE methods. Compared with M1, M2, and M4 had almost similar biases i.e., close to zero. Though the differences are small, on average M4 showed the lowest biases, followed by M1 then M2.

Regarding the average RMSE which takes the bias and variability of an estimate into consideration when evaluating an estimator, M3 tended to show higher values than other SE methods when Lap1 was used as the estimation method. Similarly, using Lap2 as the estimation method, RMSE values are highest with the sandwich estimator M5. Increasing sample size and the number of items tended to decrease the RMSE across all SE methods and conditions. However, the methods differed in the amount of decrease in the RMSE, e.g., increasing the number of items from 12 to 24 for sample size 1,000 we observe that M3 has the highest

decrease in RMSE i.e., from 0.207 to 0.116 (91 points) whereas M4 has the lowest decrease in RMSE i.e., from 0.0135 to 0.0099 (36 points). The lowest values in RMSE were observed when sample size was largest with many items i.e., 4,000 and 24. M1 and M4 showed the lowest values.

The average absolute bias and RMSE for all parameters for the GRM independent cluster models is summarized in Table 2. The results suggest that absolute bias exhibited by the SE methods decreased as sample size increased. The highest bias (0.0794) was exhibited by M3 Lap1 when test length was 24 and sample size 250. On average, with more test items, the results showed less bias. The smallest bias (0.002) was when the test length equaled 24 and sample size 4000. M3 showed slightly higher average absolute bias and RMSE compared to other SE methods with Lap1 or Lap2 estimation method. Across all the five methods we observe close to zero biases. Nevertheless, slight differences exist between the methods. Compared with M1, the Sandwich estimator M5 showed better performance than in the 2-PL case. M4 had the lowest bias, followed by M1, M2, M5 and lastly M3. The RMSE was consistently lowest for the M1 with both Lap1 and Lap2 estimation method.

Table 1

Outcome Measures for the 2-PL Independent Cluster Models with the Lap1 and Lap2 Estimation Methods .

Outcome Measure	Sample Size	Items	Lap1					Lap2				
			M1	M2	M3	M4	M5	M1	M2	M3	M4	M5
Average Absolute Bias	250	24	.0072	.0077	.0241	.0060	.0094	.0068	.0083	.0196	.0048	.0101
	1000	12	.0087	.0090	.0155	.0057	.0111	.0061	.0109	.0066	.0060	.0148
		24	.0039	.0043	.0067	.0036	.0062	.0038	.0050	.0042	.0036	.0063
	4000	12	.0077	.0073	.0109	.0053	.0077	.0051	.0071	.0054	.0052	.0089
		24	.0036	.0038	.0043	.0036	.0047	.0036	.0041	.0036	.0035	.0048
Average Root Mean Squared Error	1000	24	.0384	.0396	.0475	.0418	.0453	.0372	.0386	.0473	.0371	.0417
		12	.0152	.0156	.0207	.0135	.0174	.0178	.0195	.0191	.0172	.0213
		24	.0099	.0105	.0116	.0099	.0118	.0104	.0112	.0110	.0104	.0121
	4000	12	.0087	.0085	.0117	.0069	.0088	.0079	.0088	.0081	.0079	.0101
		24	.0048	.0050	.0054	.0048	.0057	.0049	.0052	.0049	.0049	.0058

Notes. Lap1 = first-order Laplace, Lap2 = second-order Laplace, M1 to M5 represent standard error method of estimation i.e., M1= Observed information matrix, M2= Fast observed information matrix, M3= Empirical cross-product matrix, M4= Sandwich estimator from observed information matrix and M5= Sandwich estimator from fast observed information matrix.

The average absolute bias and RMSE for all parameters for the 2-PL cross-loadings models is summarized in Table 3. SE estimation methods with Lap2 estimation methods showed less bias compared to Lap 1 methods. The results suggest that average absolute bias exhibited by the standard error methods decreased as sample size increased. The highest bias (0.0716) was exhibited by M3 Lap1 when test length was 24 and sample size 250.

On average the M3 showed slightly higher average absolute bias and RMSE compared to other SE methods when Lap1 was used but when Lap2 estimation method was used, it was the Sandwich estimator M5 that on average depicted slightly higher average absolute bias and RMSE. Compared with M1, M2, and the M4 had almost similar biases i.e., close to zero. However, on average M1 showed the lowest biases, when Lap1 estimation method was used while M2 showed lowest bias with Lap2 estimation method.

Table 2

Outcome Measures for the GRM Independent Cluster Models with the Lap1 and Lap2 Estimation Methods.

Outcome Measure	Sample Size	Items	Lap1					Lap2				
			M1	M2	M3	M4	M5	M1	M2	M3	M4	M5
Average Absolute Bias	1000	24	.0052	.0052	.0794	.0052	.0052	.0055	.0055	.0780	.0053	.0053
		12	.0039	.0056	.0086	.0029	.0057	.0031	.0041	.0044	.0029	.0073
	4000	24	.0029	.0032	.0085	.0028	.0035	.0028	.0030	.0078	.0028	.0035
		12	.0035	.0041	.0046	.0045	.0040	.0029	.0039	.0030	.0029	.0045
		24	.0020	.0021	.0024	.0020	.0022	.0020	.0020	.0022	.0020	.0022
Average Root Mean Squared Error	1000	24	.0176	.0176	.0844	.0182	.0182	.0183	.0183	.0833	.0192	.0192
		12	.0070	.0085	.0109	.0062	.0086	.0077	.0092	.0087	.0081	.0122
	4000	24	.0054	.0058	.0098	.0055	.0061	.0055	.0057	.0094	.0057	.0063
		12	.0040	.0046	.0051	.0044	. 0045	.0038	.0043	.0039	.0039	.0055
		24	.0025	.0026	.0029	.0025	.0028	.0025	.0026	.0027	.0026	.0028

Notes. Lap1 = first-order Laplace, Lap2 = second-order Laplace, M1 to M5 represent standard error method of estimation i.e., M1= Observed information matrix, M2= Fast observed information matrix, M3= Empirical cross-product matrix, M4= Sandwich estimator from observed information matrix and M5= Sandwich estimator from fast observed information matrix.

Table 3

Outcome Measures for the 2-PL Cross-Loadings Models with Lap1 and Lap2 Estimation Methods .

Outcome Measure	Sample Size	Items	Lap1					Lap2				
			M1	M2	M3	M4	M5	M1	M2	M3	M4	M5
Average Absolute Bias	250	24	.0613	.0622	.0716	.0557	.0588	.0076	.0093	.0243	.0053	.0088
	1000	12	.0043	.0069	.0110	.0025	.0076	.0038	.0059	.0058	.0057	.0172
		24	.0025	.0031	.0047	.0022	.0053	.0039	.0037	.0025	.0049	.0054
	4000	12	.0032	.0025	.0063	.0011	.0043	.0045	.0027	.0033	.0064	.0052
		24	.0012	.0014	.0018	.0011	.0023	.0019	.0015	.0014	.0024	.0024
Average Root Mean Squared	250	24	.1043	.1052	.1204	.1170	.1189	.0430	.0440	.0576	.0420	.0441
	1000	12	.0086	.0108	.0142	.0078	.0121	.0085	.0085	.0106	.0097	.0818
		24	.0096	.0101	.0107	.0104	.0120	.0103	.0103	.0097	.0116	.0113
	4000	12	.0049	.0046	.0074	.0039	.0059	.0061	.0046	.0049	.0080	.0063
		24	.0026	.0028	.0030	.0026	.0036	.0030	.0030	.0027	.0035	.0036

Notes. Lap1 = first-order Laplace, Lap2 = second-order Laplace, M1 to M5 represent standard error method of estimation i.e., M1= Observed information matrix, M2= Fast observed information matrix, M3= Empirical cross-product matrix, M4= Sandwich estimator from observed information matrix and M5= Sandwich estimator from fast observed information matrix.

The average absolute bias and RMSE for all parameters for the GRM cross-loadings models is summarized in Table 4. SE estimation methods with Lap2 estimation showed less bias compared to Lap 1. The results suggest that average absolute bias decreased with increase in sample size. The highest bias (0.0838) was exhibited by M3 Lap1 when test length was 24 and sample size 250. The empirical cross product matrix (M3) consistently showed slightly higher average absolute bias and RMSE compared to other SE methods with either Lap1 or Lap2 estimation method. M1 had the lowest bias when Lap2 estimations method was used, whereas M4 had the lowest biases when estimation method was Lap1. However, the biases were close to zero. In the cross-loading models the root mean squared error was lowest for the M1 followed closely by the M4, M2, M5 and M3 came last.

Table 4

Outcome Measures for the GRM Cross Models with Lap1 and Lap2 Estimation Methods.

Outcome Measure	Sample Size	Items	Lap1					Lap2				
			M1	M2	M3	M4	M5	M1	M2	M3	M4	M5
Average Absolute Bias	250	24	.0039	.0039	.0838	.0037	.0037	.0039	.0039	.0826	.0037	.0037
	1000	12	.0030	.0037	.0074	.0024	.0053	.0034	.0035	.0038	.0047	.0060
		24	.0021	.0025	.0077	.0021	.0029	.0022	.0024	.0068	.0025	.0029
	4000	12	.0012	.0018	.0021	.0009	.0020	.0019	.0013	.0011	.0025	.0023
		24	.0009	.0010	.0013	.0009	.0012	.0009	.0009	.0011	.0009	.0012
Average Root Mean Squared Error	250	24	.0175	.0175	.0888	.0182	.0182	.0182	.0182	.0878	.0191	.0191
	1000	12	.0063	.0078	.0100	.0059	.0085	.0067	.0081	.0076	.0076	.0104
		24	.0047	.0050	.0092	.0048	.0056	.0049	.0052	.0085	.0051	.0058
	4000	12	.0019	.0025	.0028	.0017	.0027	.0024	.0022	.0020	.0030	.0031
		24	.0014	.0015	.0018	.0015	.0017	.0014	.0015	.0017	.0015	.0017

Notes. Lap1 = first-order Laplace, Lap2 = second-order Laplace, M1 to M5 represent standard error method of estimation i.e., M1= Observed information matrix, M2= Fast observed information matrix, M3= Empirical cross-product matrix, M4= Sandwich estimator from observed information matrix and M5= Sandwich estimator from fast observed information matrix.

The average coverage rate of the 95% confidence interval is summarized in Table 5. The average coverage rates of the 95% confidence intervals are lower than the nominal level SE estimation methods using Lap1. Varying the settings did not significantly improve the coverage rates for Lap1 estimation. Across all settings investigated, all SE methods had coverage late below the nominal level. The results suggest that all SE methods produced biased and imprecise estimates with Lap1 estimates. On the other hand, the average coverage rate of the 95% CI for the SE methods using Lap2 is not statistically different from 95% for most of the conditions investigated. However, for the M2 and the M5, in all sample sizes investigated with 12 items the average coverage rates for the 95% CI was still slightly lower than the nominal rate even with Lap2 estimation. Therefore, the results seem to suggest with Lap2 estimation methods, having more test items increases the coverage rate, for instance with 24 items the sample size 1,000 has the correct coverage.

Table 6 summarizes the average estimation times for the SE estimation methods. Since the sandwich estimators have the same timings as the non-sandwich versions their timings are not reported in the table. The results suggest that the average estimation time increased as sample size and the number of items increased. For instance, the sample size 250 with 12 items took on average 1.3 seconds to compute the SE using M2 whereas sample size 4,000 with 24 items took 48.81 seconds. The M2 was faster in computing the standard errors compared to the M1 irrespective of which estimation method was used. Generally, independent cluster models took lesser time to compute the SEs compared to cross-loading models, but the same pattern is observed, i.e., the M2 took lesser time than the M1whether Lap1 or Lap2 was used as the estimation method. The M3 was the most computationally efficient method in terms of average estimation time for computing standard errors since it took the least time in all settings investigated.

Table 5

Average Coverage Rates for 95% CI

SEM	12items		24items	
	Lap1	Lap2	Lap1	Lap2
		sample size 1000		
M1	90.06	95.18	94.37	95.19
M2	88.95	94.39	94.28	95.13
M3	91.19	95.33	95.02	95.66
M4	89.20	95.27	94.36	95.26
M5	87.23	93.64	94.15	95.11
		sample size 4000		
M1	78.93	94.94	92.57	95.07
M2	77.73	94.33	92.41	95.01
M3	81.03	94.87	92.93	95.18
M4	77.51	95.02	92.42	95.10
M5	75.45	93.63	92.07	94.99
		sample size 250		
M1			92.15	94.91
M2			91.37	95.28
M3			93.11	94.83
M4			91.76	94.56
M5			90.13	94.81

Notes. Lap1 = first-order Laplace, Lap2 = second-order Laplace, M1 to M5 represent standard error method of estimation i.e., M1= Observed information matrix, M2= Fast observed information matrix, M3= Empirical cross-product matrix, M4= Sandwich estimator from observed information matrix and M5= Sandwich estimator from fast observed information matrix.

Table 6

Average Estimation Times in seconds for the 2-PL and GRM Models.

		2-PL						GRM					
		M1		M2		M3		M1		M2		M3	
Sample Size	Items	Lap1	Lap2	Lap1	Lap2	Lap1	Lap2	Lap1	Lap2	Lap1	Lap2	Lap1	Lap2
						Independent Cluster Models							
250	24	5.47	5.87	4.49	4.85	0.10	0.12	26.67	29.10	29.06	29.16	0.40	0.42
1000	12	5.70	6.04	3.92	4.37	0.22	0.23	14.09	15.33	9.88	11.23	0.24	0.26
	24	17.22	18.54	13.25	14.74	0.36	0.38	47.48	52.24	38.99	44.04	0.44	0.47
4000	12	20.92	22.37	14.17	15.69	0.80	0.85	50.61	55.37	34.94	39.8	0.83	0.90
	24	64.01	69.21	48.81	54.23	1.30	1.39	164.70	182.02	128.08	145.89	1.41	1.55
						Cross-Loadings Models							
250	24	10.75	13.45	9.53	12.31	0.20	0.25	40.49	55.19	40.90	56.12	0.70	0.92
1000	12	12.05	15.62	8.61	12.20	0.39	0.50	28.66	42.54	20.88	34.80	0.45	0.65
	24	35.92	46.64	28.55	39.24	0.67	0.86	95.06	142.66	76.69	126.25	0.86	0.86
4000	12	48.67	63.15	35.15	49.60	1.57	2.02	112.85	168.69	82.41	138.09	1.73	2.55
	24	145.21	188.11	115.97	158.69	2.68	3.44	367.43	558.46	294.51	486.70	3.01	4.50

Note. Lap1 = first-order Laplace, Lap2 = second-order Laplace, M1 to M5 represent standard error method of estimation i.e., M1= Observed information matrix, M2= Fast observed information matrix. GRM=Graded Response Model, 2-PL=two parameter logistic. The Sandwich estimator from the observed information matrix (M4) and the Sandwich estimator from the fast observed information matrix have the same timing as the non-Sandwich versions.

Discussion and Conclusion

Summary of Recommendations

In the context of item response theory (IRT), SEs are an indication of the degree of precision with which item parameters are estimated (Thissen & Wainer, 1982). In the literature of IRT thus far, it is unclear which estimator of SEs should be used when drawing inference with Laplace approximations. Motivated by the importance of SEs to practitioners and the paucity of comprehensive studies on performance of different SE methods in IRT literature, we conducted a Monte-Carlo Simulation study to compare the performance of five SE methods across varying conditions. Specifically, the following simulation settings were investigated, Tests with 4/8 items per dimension, sample sizes of 250, 1,000, and 4000, two model types (independent cluster and cross-loadings), number of dimensions (3 dimensions), and number of categories (2 category and 5 category). This study complements a recent study by (Andersson et al., 2023) by directly evaluating SEs from the observed information matrix (M1), fast observed information matrix (M2), the empirical cross product matrix (M3), the sandwich estimator from the observed information matrix (M4) and the sandwich estimator from the fast observed information matrix (M5).

Research question one

How do the five methods of calculating SEs in IRT compare across different varying conditions (i.e., sample sizes, test length, number of categories and model complexities) in terms of computational efficiency, accuracy, and precision?

In general, the results in this paper show that no method of estimating standard errors was found unacceptable in terms of average absolute bias and average root mean squared error. The SE methods are very close regarding to the accuracy and precision between the different

estimation methods except for the Empirical cross product matrix that has slightly higher average RMSE and average absolute bias especially for the small sample size i.e., 250. Comparing SEs from the observed information matrix (M1) to those from the empirical cross product matrix (M3), the Empirical cross-product tended to overestimate the standard errors especially for the sample size 250 and the sample size 1,000 with 12 items. Of all the simulation conditions, sample size showed the greatest impact on performance of the SE methods in relation to accuracy and precision. Large sample sizes showed more precise estimates compared to small sample sizes in all investigated conditions. It can thus be argued that larger sample sizes provide more precise estimates of the latent trait being measured. This effect of sample size on standard errors is similar to the results found by (Paek & Cai, 2014; Thissen & Wainer, 1982).

However, the average coverage rates for 95% CI in all SE methods using Lap1 produced biased and imprecise estimates. With Lap2 the standard errors are approximately equally accurate and precise for either of the five estimation methods. Results from the simulation study showed that the estimated confidence intervals for the SE methods have good coverage properties with sample sizes of at least 1,000 and 24 items for 2-PL and the GRM models when using Lap2. This is consistent with the results of (Andersson & Xin, 2018).

Research question Two

In practice, which estimator of SEs should be used for estimators based on Laplace approximations?

To be able to make an evidence based decision on their estimator of choice a researcher need to consider a combination of various factors like the sample size, test length, model complexity and available resources. Within the limits of the factors investigated in this study for

a researcher faced with time constraint and with large sample size the empirical cross product is a suitable choice.

Research question Three

What is the empirical coverage rate of 95% confidence intervals when using the different methods for computing the standard errors? In all conditions studied the standard error estimators with Lap1 the average coverage rate of the 95% CI was lower than the nominal level. On the other hand, using Lap2 estimations method conditions with 24 items had an average coverage rate of 95% CI that is not statistically significantly different to the nominal level for any sample size.

Significance and Contributions

A comparison of the different SE procedures with Laplace approximations reveal that one using Lap1 estimation would likely lead to biased and imprecise estimates for three dimensional IRT models, whereas Lap2 estimation would give accurate and precise estimates of the SEs.

Limitations and Future Research

A limitation of the current study was not investigating the performance of SE methods when the model is mis-specified, and the distribution is non normal. Therefore, future investigations are needed, and one should be cautious in generalizing the results beyond the current design. A possible future study is to evaluate the properties of the standard error estimation methods based on Laplace approximations and adaptive Gauss-Hermite quadrature approximations when the underlying latent variable distribution is non-normal.

In conclusion, no method was found unacceptable in terms of average root mean squared error and average absolute bias. The average coverage rates of the 95% CI for SE estimators using Lap1 were less than the nominal level, whereas SE estimators with Lap2 produced good

coverage rates. As for estimation times, the cross empirical product matrix took the least time to compute the standard errors.

References

Andersson, B., Jin, S., & Zhang, M. (2023). Fast estimation of multiple group generalized linear latent variable models for categorical observed variables. *Computational Statistics & Data Analysis, 182*, 107710. https://doi.org/10.1016/j.csda.2023.107710

Andersson, B., & Xin, T. (2018). Large sample confidence intervals for item response theory reliability coefficients. *Educational and Psychological Measurement*, *78*(1), 32–45. https://doi.org/doi: 10.1177/0013164417713570.

Andersson, B., & Xin, T. (2021). Estimation of latent regression item response theory models using a second-order Laplace approximation. *Journal of Educational and Behavioral Statistics*, *46*(2), 244–265.

Bock, R. D., & Aitkin, M. (1981). Marginal maximum likelihood estimation of item parameters: Application of an EM algorithm. *Psychometrika*, *46*(4), 443–459.

Bonifay, W. (2020). *Multidimensional Item Response Theory* (By pages 47-54). SAGE Publications, Inc. https://doi.org/10.4135/9781506384276

Cagnone, S., & Monari, P. (2013). Latent variable models for ordinal data by using the adaptive quadrature approximation. *Computational Statistics*, *28*(2), 597–619. https://doi.org/10.1007/s00180-012-0319-z

Cai, L. (2010). Metropolis-Hastings Robbins-Monro algorithm for confirmatory item factor analysis. *Journal of Educational and Behavioral Statistics*, *35*(3), 307–335.

Cai, L., & Hansen, M. (2013). Limited-information goodness-of-fit testing of hierarchical item factor models. *British Journal of Mathematical and Statistical Psychology*, *66*(2), 245–276.

De Ayala, R. J. (1995). *An Investigation of the Standard Errors of Expected A Posteriori Ability Estimates.*

Hays, W. L. (1988). Statistics (4th edn). *New York: Holt*.

Hoekstra, R., Morey, R. D., Rouder, J. N., & Wagenmakers, E.-J. (2014). Robust misinterpretation of confidence intervals. *Psychonomic Bulletin & Review*, *21*(5), 1157–1164. https://doi-org.ezproxy.uio.no/10.3758/s13423-013-0572-3

Huber, P., Ronchetti, E., & Victoria-Feser, M.-P. (2004). Estimation of Generalized Linear Latent Variable Models. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, *66*(4), 893–908. https://doi.org/10.1111/j.1467-9868.2004.05627.x

Immekus, J. C., Snyder, K. E., & Ralston, P. A. (2019). Multidimensional item response theory for factor structure assessment in educational psychology research. *Frontiers in Education*, *4*, 45. https://doi.org/10.3389/feduc.2019.00045

Joe, H. (2008). Accuracy of Laplace approximation for discrete response mixed models. *Computational Statistics & Data Analysis*, *52*(12), 5066–5074. https://doi.org/10.1016/j.csda.2008.05.002

Kolen, M. J., & Brennan, R. L. (2004). *Test equating, scaling, and linking*.

Li, X., & Wang, W.-C. (2015). Assessment of differential item functioning under cognitive diagnosis models: The DINA model example. *Journal of Educational Measurement*, *52*(1), 28–54.

Lin, Z. (2018). *The comparison of standard error methods in the marginal maximum likelihood estimation of the two-parameter logistic item response model when the distribution of the latent trait is nonnormal* [The Florida State University]. http://purl.flvc.org/fsu/fd/2018_Sp_Lin_fsu_0071E_14423

Liu, Y., Xin, T., Andersson, B., & Tian, W. (2019). Information matrix estimation procedures for cognitive diagnostic models. *British Journal of Mathematical and Statistical Psychology*, *72*(1), 18–37.

Ma, W., Iaconangelo, C., & de la Torre, J. (2016). Model similarity, model selection, and attribute classification. *Applied Psychological Measurement*, *40*(3), 200–217. https://doi.org/10.1177/0146621615621717

Maydeu-Olivares, A., & Joe, H. (2005). Limited-and full-information estimation and goodness-of-fit testing in 2 n contingency tables: A unified framework. *Journal of the American Statistical Association*, *100*(471), 1009–1020.

Monroe, S. (2019). Estimation of expected Fisher information for IRT models. *Journal of Educational and Behavioral Statistics*, *44*(4), 431–447. https://doi.org/10.1007/978-0-387-89976-3_4

Noh, M., & Lee, Y. (2007). REML estimation for binary data in GLMMs. *Journal of Multivariate Analysis*, *98*(5), 896–915.

Paek, I., & Cai, L. (2014). A comparison of item parameter standard error estimation procedures for unidimensional and multidimensional item response theory modeling. *Educational and Psychological Measurement*, *74*(1), 58–76.

Raudenbush, S. W., Yang, M.-L., & Yosef, M. (2000). Maximum likelihood for generalized linear models with nested random effects via high-order, multivariate Laplace approximation. *Journal of Computational and Graphical Statistics*, *9*(1), 141–157.

Reckase, M. D., & Reckase, M. D. (2009). *Multidimensional item response theory models*. Springer.

Schilling, S., & Bock, R. Darrell. (2005). High-dimensional maximum marginal likelihood item factor analysis by adaptive quadrature. *Psychometrika*, *70*(3), 533–555. https://doi.org/10.1007/s11336-003-1141-x

Shun, Z. (1997). Another Look at the Salamander Mating Data: A Modified Laplace Approximation Approach. *Journal of the American Statistical Association*, *92*(437), 341–349. https://doi.org/10.1080/01621459.1997.10473632

Thissen, D., & Wainer, H. (1982). Some standard errors in item response theory. *Psychometrika*, *47*(4), 397–412. https://doi.org/10.1007/BF02293705

Tsutakawa, R. K. (1984). Estimation of two-parameter logistic item response curves. *Journal of Educational Statistics*, *9*(4), 263–276. https://doi-org.ezproxy.uio.no/10.3102/10769986009004263

Vogt, W. (2022). *Dictionary of Statistics & Methodology* (By pages 307-307; 3rd ed., Vol. 1–0). SAGE Publications, Inc. https://doi.org/10.4135/9781412983907

Wainer, H., & Thissen, D. (1987). Estimating ability with the wrong model. *Journal of Educational Statistics*, *12*(4), 339–368. https://doi-org.ezproxy.uio.no/10.3102/10769986012004339

Woods, C. M., Cai, L., & Wang, M. (2013). The Langer-improved Wald test for DIF testing with multiple groups: Evaluation and comparison to two-group IRT. *Educational and Psychological Measurement*, *73*(3), 532–547. https://doi-org.ezproxy.uio.no/10.1177/0013164412464875

Yuan, K.-H., Cheng, Y., & Patton, J. (2014). Information matrices and standard errors for MLEs of item parameters in IRT. *Psychometrika*, *79*(2), 232–254. https://doi-org.ezproxy.uio.no/10.1007/s11336-013-9334-4

Zanon, C., Hutz, C. S., Yoo, H. (Henry), & Hambleton, R. K. (2016). An application of item response theory to psychological test development. *Psicologia: Reflexão e Crítica, 29*(1), 18. https://doi.org/10.1186/s41155-016-0040-x

Appendices

Appendix I: GDPR documents & Ethical approval

We used simulated data, so this is not needful.

Appendix II: Data Management and Analysis Code

Below is a Sample code used in to generate data, run models, and the simulations, additional codes are available upon request.

```
#######################
rm(list=ls())
#a function to generate data, and run models using generated data and
#return objects of interest to the study
#.....................................................
generate_data <- function(nitems, sample.size, ncategories, nfactors,
model.type, seed ) {
  library(lamle)
  library(mvtnorm)
  if ( ncategories== 2){
     set.seed(124)
    a<- runif(nitems,0.8, 2)
     set.seed(124)
    b<-  rnorm(nitems)
  } else if(ncategories == 5 ){
     #Item parameter generation
    set.seed(124)
    a <- runif(nitems, 0.8, 2)
```

```
    set.seed(124)
    b <- vector("list", nitems)
    for(j in 1:nitems) b[[j]] <- -c(runif(1, -3, -2),
                                    runif(1, -1.5, -0.5),
                                    runif(1, 0, 1),
                                    runif(1, 1.5, 2.5))

  }
  #set up covariance matrix for 3D model
  set.seed(1234)
  covmat=diag(rep(1, nfactors))
 covmat[lower.tri(covmat)] <- runif((nfactors*(nfactors-1)/2),0.4,0.6)
 covmat[upper.tri(covmat)] <- t(covmat)[upper.tri(covmat)]
if (nfactors == 3 && model.type == "IND"){
####  Setup three-dimensional independent-clusters model
    mydim <- matrix(NA, nrow = nitems, ncol = nfactors)
    mydim[1:(nitems / nfactors), 1] <- 1
    mydim[(nitems / nfactors + 1):(2 * nitems / nfactors), 2] <- 1
    mydim[(2 * nitems / nfactors + 1):(3 *nitems / nfactors), 3] <- 1
    #######and
    GRMa <- matrix(0, nrow = nitems, ncol = 3)
    GRMa[1:(nitems / nfactors), 1] <- a[1:(nitems / nfactors)]
    GRMa[(nitems / nfactors + 1):(2 * nitems / nfactors),
     2] <- a[(nitems / nfactors + 1):(2 * nitems / nfactors)]
    GRMa[(2 * nitems / nfactors + 1):(3 * nitems / nfactors),
```

```
3] <- a[(2 * nitems / nfactors + 1):(3 * nitems / nfactors)]


#Data generation: independent-clusters
set.seed(seed)
latmat <- rmvnorm(sample.size, c(0, 0, 0), covmat)
dataGRMIND <- matrix(NA, nrow =  sample.size, ncol = nitems)
dataGRMIND[1:sample.size, ] <- DGP(GRMa, b[1:nitems],
rep("GRM", nitems), latmat)
 colnames(dataGRMIND[1:sample.size, ]) <- paste0("item", 1:nitems)
 #estimate lamle object using generated response data
myGRMINDLap1 <- try(lamle(y = dataGRMIND[1:sample.size, ],
                          model = mydim,
                          modeltype = rep("GRM", nitems),
                          first.step = 25,
                          optimizer = "BHHH",
                          method = "lap",
                          maxit = 200,
                          accuracy = 1,
                          obsinfo = TRUE,
                          thetaupdate = TRUE))


if(myGRMINDLap1$iter >= 200) {
  myGRMINDLap1 <- try(lamle(y = dataGRMIND[1:sample.size, ],
                            model = mydim,
                            modeltype = rep("GRM", nitems),
                            first.step = 25,
```

```
                                optimizer = "BFGS",
                                method = "lap",
                                maxit = 200,
                                accuracy = 1,
                                obsinfo = TRUE,
                                thetaupdate = TRUE))
 }
myGRMINDLap2 <- try(lamle(y = dataGRMIND[,],
                          model = mydim,
                          modeltype = rep("GRM", nitems),
                          first.step = 25,
                          optimizer = "BHHH",
                          maxit = 200,
                          method = "lap",
                          accuracy = 2,
                          obsinfo = TRUE,
                          thetaupdate = TRUE))
  if(myGRMINDLap2$iter >= 200) {
   myGRMINDLap2 <- try(lamle(y = dataGRMIND[1:sample.size, ],
                                model = mydim,
                                modeltype = rep("GRM", nitems),
                                first.step = 25,
                                optimizer = "BFGS",
                                method = "lap",
                                maxit = 200,
                                accuracy = 2,
```

```
                                obsinfo = TRUE,
                                thetaupdate = TRUE))
}
myGRMINDLap1FastObs <- try(lamle(y = dataGRMIND[,],
                                     model = mydim,
                                     modeltype = rep("GRM", nitems),
                                     first.step = 25,
                                     optimizer = "BHHH",
                                     method = "lap",
                                     accuracy = 1,
                                     maxit = 200,
                                     obsinfo = TRUE,
                                     thetaupdate = FALSE))
if(myGRMINDLap1FastObs$iter >= 200) {
 myGRMINDLap1FastObs <- try(lamle(y = dataGRMIND[1:sample.size, ],
                                       model = mydim,
                                       modeltype = rep("GRM", nitems),
                                       first.step = 25,
                                       optimizer = "BFGS",
                                       method = "lap",
                                       maxit = 200,
                                       accuracy = 1,
                                       obsinfo = TRUE,
                                       thetaupdate = TRUE))
}
myGRMINDLap2FastObs <- try(lamle(y = dataGRMIND[,],
```

```
                                    model = mydim,
                                    modeltype = rep("GRM", nitems),
                                    first.step = 25,
                                    optimizer = "BHHH",
                                    method = "lap",
                                    maxit = 200,
                                    accuracy = 2,
                                    obsinfo = TRUE,
                                    thetaupdate = FALSE))

  if(myGRMINDLap2FastObs$iter >= 200) {
  myGRMINDLap2FastObs <- try(lamle(y = dataGRMIND[1:sample.size, ],
                                     model = mydim,
                                     modeltype = rep("GRM", nitems),
                                     first.step = 25,
                                     optimizer = "BFGS",
                                     method = "lap",
                                     accuracy = 2,
                                     maxit = 200,
                                     obsinfo = TRUE,
                                     thetaupdate = TRUE))

  }

} else if(nfactors == 6 && model.type == "IND" ){
```

```
####  Setup three-dimensional independent-clusters model
mydim <- matrix(NA, nrow = nitems, ncol =nfactors)
mydim[1:(nitems / nfactors), 1] <- 1
mydim[(nitems / nfactors + 1):(2 * nitems / nfactors), 2] <- 1
mydim[(2 * nitems / nfactors + 1):(3 * nitems / nfactors), 3] <- 1
mydim[(3 * nitems / nfactors + 1):(4 * nitems / nfactors), 4] <- 1
mydim[(4 * nitems / nfactors + 1):(5 * nitems / nfactors), 5] <- 1
mydim[(5 * nitems / nfactors + 1):(6 * nitems / nfactors), 6] <- 1
GRMa <- matrix(0, nrow = nitems, ncol = nfactors)
GRMa[1:(nitems / nfactors), 1] <- a[1:(nitems / nfactors)]
GRMa[(nitems / nfactors + 1):(2 * nitems / nfactors),
2] <-a[(nitems / nfactors + 1):(2 * nitems / nfactors)]
GRMa[(2 * nitems / nfactors + 1):(3 * nitems / nfactors),
3] <- a[(2 * nitems / nfactors + 1):(3 * nitems / nfactors)]
GRMa[(3 * nitems / nfactors + 1):(4 * nitems / nfactors),
4] <- a[(3 * nitems / nfactors + 1):(4 * nitems / nfactors)]
GRMa[(4 * nitems / nfactors + 1):(5 * nitems / nfactors),
5] <- a[(4 * nitems / nfactors + 1):(5 * nitems / nfactors)]
GRMa[(5 * nitems / nfactors + 1):(6 * nitems / nfactors), 6]
<- a[(5 * nitems / nfactors + 1):(6 * nitems / nfactors)]
set.seed(seed)
latmat <- rmvnorm(sample.size, c(0, 0, 0, 0, 0, 0), covmat)
dataGRMIND <- matrix(NA, nrow =  sample.size, ncol = nitems)
dataGRMIND[1:sample.size, ] <- DGP(GRMa, b[1:nitems],
```

```
rep("GRM", nitems), latmat)


colnames(dataGRMIND[1:sample.size, ]) <- paste0("item", 1:nitems)
#estimate lamle object using generated response data
myGRMINDLap1 <- try(lamle(y = dataGRMIND[1:sample.size, ],
                          model = mydim,
                          modeltype = rep("GRM", nitems),
                          first.step = 25,
                          optimizer = "BHHH",
                          method = "lap",
                          maxit = 200,
                          accuracy = 1,
                          obsinfo = TRUE,
                          thetaupdate = TRUE))
if(myGRMINDLap1$iter >= 200) {
  myGRMINDLap1 <- try(lamle(y = dataGRMIND[1:sample.size, ],
                            model = mydim,
                            modeltype = rep("GRM", nitems),
                            first.step = 25,
                            optimizer = "BFGS",
                            method = "lap",
                            accuracy = 1,
                            obsinfo = TRUE,
                            thetaupdate = TRUE))
}
myGRMINDLap2 <- try(lamle(y = dataGRMIND[,],
```

```
                              model = mydim,
                              modeltype = rep("GRM", nitems),
                              first.step = 25,
                              optimizer = "BHHH",
                              method = "lap",
                              maxit = 200,
                              accuracy = 2,
                              obsinfo = TRUE,
                              thetaupdate = TRUE))
  if(myGRMINDLap2$iter >= 200) {
  myGRMINDLap2 <- try(lamle(y = dataGRMIND[1:sample.size, ],
                              model = mydim,
                              modeltype = rep("GRM", nitems),
                              first.step = 25,
                              optimizer = "BFGS",
                              method = "lap",
                              accuracy = 2,
                              obsinfo = TRUE,
                              thetaupdate = TRUE))
}
myGRMINDLap1FastObs <- try(lamle(y = dataGRMIND[,],
                                   model = mydim,
                                   modeltype = rep("GRM", nitems),
                                   first.step = 25,
                                   optimizer = "BHHH",
                                   method = "lap",
```

```
                                    maxit = 200,
                                    accuracy = 1,
                                    obsinfo = TRUE,
                                    thetaupdate = FALSE))
if(myGRMINDLap1FastObs$iter >= 200) {
 myGRMINDLap1FastObs <- try(lamle(y = dataGRMIND[1:sample.size, ],
                                    model = mydim,
                                    modeltype = rep("GRM", nitems),
                                    first.step = 25,
                                    optimizer = "BFGS",
                                    method = "lap",
                                    accuracy = 1,
                                    obsinfo = TRUE,
                                    thetaupdate = TRUE))
}
myGRMINDLap2FastObs <- try(lamle(y = dataGRMIND[,],
                                    model = mydim,
                                    modeltype = rep("GRM", nitems),
                                    first.step = 25,
                                    optimizer = "BHHH",
                                    maxit = 200,
                                    method = "lap",
                                    accuracy = 2,
                                    obsinfo = TRUE,
                                    thetaupdate = FALSE))
if(myGRMINDLap2FastObs$iter >= 200) {
```

```
   myGRMINDLap2FastObs <- try(lamle(y = dataGRMIND[1:sample.size, ],
                                    model = mydim,
                                    modeltype = rep("GRM", nitems),
                                    first.step = 25,
                                    optimizer = "BFGS",
                                    method = "lap",
                                    accuracy = 2,
                                    obsinfo = TRUE,
                                    thetaupdate = TRUE))
  }

}
if (nfactors == 3 && model.type == "CL"){
  #Data generating model: cross-loadings
  if ( ncategories== 2){
    set.seed(124)
    a<- runif(1000,0.8, 2)
    set.seed(124)
    b<-  rnorm(nitems)
  } else if(ncategories == 5 ){
    #Item parameter generation
    set.seed(124)
    a <- runif(1000, 0.8, 2)
    set.seed(124)
    b <- vector("list", nitems)
    for(j in 1:nitems) b[[j]] <- -c(runif(1, -3, -2),
```

```
                                        runif(1, -1.5, -0.5),
                                        runif(1, 0, 1),
                                        runif(1, 1.5, 2.5))
    }
#set up covariance matrix for 3D model
set.seed(1234)
covmat=diag(rep(1, nfactors))
covmat[lower.tri(covmat)] <- runif((nfactors*(nfactors-1)/2), 0.4,0.6)
covmat[upper.tri(covmat)] <- t(covmat)[upper.tri(covmat)]
#set the model structure for CL
mydim <- matrix(NA, nrow = nitems, ncol =nfactors)
mydim[1:(nitems / nfactors), 1] <- 1
mydim[(nitems / nfactors + 1):(2 * nitems / nfactors), 2] <- 1
mydim[(2 * nitems / nfactors + 1):(3 * nitems / nfactors), 3] <- 1
mydim[1, c(2)] <- 1
mydim[(nitems / nfactors + 1):(nitems / nfactors + 1), c(3)] <- 1
mydim[(2 * nitems / nfactors + 1):(2 * nitems / nfactors +1),c(1)]<- 1
###and a mat
GRMa <- matrix(0,nrow = nitems,
ncol = nfactors)GRMa[!is.na(mydim)] <- a[1:(sum(!is.na(mydim)))]
 ###data generation
set.seed(seed)
latmat <- rmvnorm(sample.size, c(0, 0, 0), covmat)
dataGRMCL <- matrix(NA, nrow =  sample.size, ncol = nitems)
dataGRMCL[1:sample.size, ] <- DGP(GRMa, b[1:nitems],
rep("GRM", nitems), latmat)
```

```
#estimate model for 3 D CL clusters
#Estimation: cross-loadings for 6 D model cl clusters
myGRMCLLap1 <- try(lamle(y = dataGRMCL[1:sample.size, ],
                         model = mydim,
                          modeltype = rep("GRM", nitems),
                          first.step = 25,
                          optimizer = "BHHH",
                          method = "lap",
                          maxit = 200,
                          accuracy = 1,
                          obsinfo = TRUE,
                          thetaupdate = TRUE))
      if(myGRMCLLap1$iter >= 200) {
    myGRMCLLap1 <- try(lamle(y = dataGRMCL[1:sample.size, ],
                             model = mydim,
                             modeltype = rep("GRM", nitems),
                             first.step = 25,
                             optimizer = "BFGS",
                             method = "lap",
                             maxit = 200,
                             accuracy = 1,
                             obsinfo = TRUE,
                             thetaupdate = TRUE))
  }
  myGRMCLLap2 <- try(lamle(y = dataGRMCL[1:sample.size, ],
                           model = mydim,
```

```
                            modeltype = rep("GRM", nitems),
                            first.step = 25,
                            optimizer = "BHHH",
                            method = "lap",
                            maxit = 200,
                            accuracy = 2,
                            obsinfo = TRUE,
                            thetaupdate = TRUE))
if(myGRMCLLap2$iter >= 200) {
  myGRMCLLap2 <- try(lamle(y = dataGRMCL[1:sample.size, ],
                             model = mydim,
                             modeltype = rep("GRM", nitems),
                             first.step = 25,
                             optimizer = "BFGS",
                             method = "lap",
                             maxit = 200,
                             accuracy = 2,
                             obsinfo = TRUE,
                             thetaupdate = TRUE))
}
myGRMCLLap1FastObs <- try(lamle(y = dataGRMCL[1:sample.size, ],
                                  model = mydim,
                                  modeltype = rep("GRM", nitems),
                                  first.step = 25,
                                  optimizer = "BHHH",
                                  method = "lap",
```

```
                                    maxit = 200,
                                    accuracy = 1,
                                    obsinfo = TRUE,
                                    thetaupdate = FALSE))
if(myGRMCLLap1FastObs$iter >= 200) {
  myGRMCLLap1FastObs <- try(lamle(y = dataGRMCL[1:sample.size, ],
                                    model = mydim,
                                    modeltype = rep("GRM", nitems),
                                    first.step = 25,
                                    optimizer = "BFGS",
                                    method = "lap",
                                    maxit = 200,
                                    accuracy = 1,
                                    obsinfo = TRUE,
                                    thetaupdate = TRUE))
}
myGRMCLLap2FastObs <- try(lamle(y = dataGRMCL[1:sample.size, ],
                                    model = mydim,
                                    modeltype = rep("GRM", nitems),
                                    first.step = 25,
                                    optimizer = "BHHH",
                                    method = "lap",
                                    accuracy = 2,
                                    maxit = 200,
                                    obsinfo = TRUE,
                                    thetaupdate = FALSE))
```

```
  if(myGRMCLLap2FastObs$iter >= 200) {
    myGRMCLLap2FastObs <- try(lamle(y = dataGRMCL[1:sample.size, ],
                                    model = mydim,
                                    modeltype = rep("GRM", nitems),
                                    first.step = 25,
                                    optimizer = "BFGS",
                                    method = "lap",
                                    maxit = 200,
                                    accuracy = 2,
                                    obsinfo = TRUE,
                                    thetaupdate = TRUE))
  }
} else if(nfactors == 6 && model.type == "CL" ){
####  Setup 6D CL-clusters model
mydim <- matrix(NA, nrow = nitems, ncol =nfactors)
mydim[1:(nitems / nfactors), 1] <- 1
mydim[(nitems / nfactors + 1):(2 * nitems / nfactors), 2] <- 1
mydim[(2 * nitems / nfactors + 1):(3 * nitems / nfactors), 3] <- 1
mydim[(3 * nitems / nfactors + 1):(4 * nitems / nfactors), 4] <- 1
mydim[(4 * nitems / nfactors + 1):(5 * nitems / nfactors), 5] <- 1
mydim[(5 * nitems / nfactors + 1):(6 * nitems / nfactors), 6] <- 1
GRMa <- matrix(0, nrow = nitems, ncol = nfactors)
GRMa[1:(nitems / nfactors), 1] <- a[1:(nitems / nfactors)]
GRMa[(nitems / nfactors + 1):(2 * nitems / nfactors),
 2] <- a[(nitems / nfactors + 1):(2 * nitems / nfactors)]
GRMa[(2 * nitems / nfactors + 1):(3 * nitems / nfactors),
```

```
  3] <- a[(2 * nitems / nfactors + 1):(3 * nitems / nfactors)]
GRMa[(3 * nitems / nfactors + 1):(4 * nitems / nfactors),
 4] <- a[(3 * nitems / nfactors + 1):(4 * nitems / nfactors)]
GRMa[(4 * nitems / nfactors + 1):(5 * nitems / nfactors),
 5] <- a[(4 * nitems / nfactors + 1):(5 * nitems / nfactors)]
GRMa[(5 * nitems / nfactors + 1):(6 * nitems / nfactors),
 6] <- a[(5 * nitems / nfactors + 1):(6 * nitems / nfactors)]
set.seed(seed)
latmat <- rmvnorm(sample.size, c(0, 0, 0, 0, 0, 0), covmat)
dataGRMCL <- matrix(NA, nrow =  sample.size, ncol = nitems)
dataGRMCL[1:sample.size, ] <- DGP(GRMa, b[1:nitems], rep("GRM",
nitems), latmat)
colnames(dataGRMCL[1:sample.size, ]) <- paste0("item", 1:nitems)
#Estimation: cross-loadings for 6 D model cl clusters
    myGRMCLLap1 <- try(lamle(y = dataGRMCL[1:sample.size, ],
                             model = mydim,
                             modeltype = rep("GRM", nitems),
                             first.step = 25,
                             optimizer = "BHHH",
                             method = "lap",
                             maxit = 200,
                             accuracy = 1,
                             obsinfo = TRUE,
                             thetaupdate = TRUE))
    if(myGRMCLLap1$iter >= 200) {
      myGRMCLLap1 <- try(lamle(y = dataGRMCL[1:sample.size, ],
```

```
                              model = mydim,
                              modeltype = rep("GRM", nitems),
                              first.step = 25,
                              optimizer = "BFGS",
                              method = "lap",
                              accuracy = 1,
                              obsinfo = TRUE,
                              thetaupdate = TRUE))
}
myGRMCLLap2 <- try(lamle(y = dataGRMCL[1:sample.size, ],
                         model = mydim,
                         modeltype = rep("GRM", nitems),
                         first.step = 25,
                         optimizer = "BHHH",
                         method = "lap",
                         maxit = 200,
                         accuracy = 2,
                         obsinfo = TRUE,
                         thetaupdate = TRUE))
if(myGRMCLLap2$iter >= 200) {
  myGRMCLLap2 <- try(lamle(y = dataGRMCL[1:sample.size, ],
                              model = mydim,
                              modeltype = rep("GRM", nitems),
                              first.step = 25,
                              optimizer = "BFGS",
                              method = "lap",
```

```
                                maxit = 200,
                                accuracy = 2,
                                obsinfo = TRUE,
                                thetaupdate = TRUE))
}
myGRMCLLap1FastObs <- try(lamle(y = dataGRMCL[1:sample.size, ],
                                model = mydim,
                                modeltype = rep("GRM", nitems),
                                first.step = 25,
                                optimizer = "BHHH",
                                method = "lap",
                                maxit = 200,
                                accuracy = 1,
                                obsinfo = TRUE,
                                thetaupdate = FALSE))
if(myGRMCLLap1FastObs$iter >= 200) {
  myGRMCLLap1FastObs <- try(lamle(y = dataGRMCL[1:sample.size, ],
                                  model = mydim,
                                  modeltype = rep("GRM", nitems),
                                  first.step = 25,
                                  optimizer = "BFGS",
                                  method = "lap",
                                  maxit = 200,
                                  accuracy = 1,
                                  obsinfo = TRUE,
                                  thetaupdate = TRUE))
```

```
  }

  myGRMCLLap2FastObs <- try(lamle(y = dataGRMCL[1:sample.size, ],
                                  model = mydim,
                                  modeltype = rep("GRM", nitems),
                                  first.step = 25,
                                  optimizer = "BHHH",
                                  method = "lap",
                                  maxit = 200,
                                  accuracy = 2,
                                  obsinfo = TRUE,
                                  thetaupdate = FALSE))
      if(myGRMCLLap2FastObs$iter >= 200) {
    myGRMCLLap2FastObs <- try(lamle(y = dataGRMCL[1:sample.size, ],
                                    model = mydim,
                                    modeltype = rep("GRM", nitems),
                                    first.step = 25,
                                    optimizer = "BFGS",
                                    method = "lap",
                                    maxit = 200,
                                    accuracy = 2,
                                    obsinfo = TRUE,
                                    thetaupdate = TRUE))
        }

}
```

```
  # Combine the generated values in a list
if (model.type=="IND"){
  data <- list(a = a,
               b = b,
               seed = seed,
               dataGRMIND = dataGRMIND[1:sample.size, ],
               myGRMINDLap2FastObs=myGRMINDLap2FastObs,
               myGRMINDLap2=myGRMINDLap2,
               myGRMINDLap1FastObs=myGRMINDLap1FastObs,
               myGRMINDLap1=myGRMINDLap1)
}else if (model.type == "CL"){
  data <- list(a = a,
               b = b,
               seed = seed,
               dataGRMCL=dataGRMCL[1:sample.size, ],
               myGRMCLLap2FastObs=myGRMCLLap2FastObs,
               myGRMCLLap1FastObs=myGRMCLLap1FastObs,
               myGRMCLLap1=myGRMCLLap1,
               myGRMCLLap1=myGRMCLLap1)
}
#Combine the results in a single data set
if (model.type=="IND"){
  result <- list(sample.size=sample.size,
                 nitem=nitems,
                 model.type=model.type,
```

```
            ncategories=ncategories,
            nfactors=nfactors,
            myGRMINDLap1.par=myGRMINDLap1$par,
            myGRMINDLap1.iter=myGRMINDLap1$iter,
            myGRMINDLap1.Amat=myGRMINDLap1$Amat,
            myGRMINDLap1.timing=myGRMINDLap1$timing,
            myGRMINDLap2.iter=myGRMINDLap2$iter,
            myGRMINDLap2.par=myGRMINDLap2$par,
            myGRMINDLap2.Amat=myGRMINDLap2$Amat,
            myGRMINDLap2.timing=myGRMINDLap2$timing,
            myGRMINDLap2FastObs.iter=myGRMINDLap2FastObs$iter,
            myGRMINDLap2FastObs.par=myGRMINDLap2FastObs$par,
            myGRMINDLap2FastObs.Amat=myGRMINDLap2FastObs$Amat,
         myGRMINDLap2FastObs.timing=myGRMINDLap2FastObs$timing,
            myGRMINDLap1FastObs.iter=myGRMINDLap1FastObs$iter,
            myGRMINDLap1FastObs.par=myGRMINDLap1FastObs$par,
            myGRMINDLap1FastObs.Amat=myGRMINDLap1FastObs$Amat,
        myGRMINDLap1FastObs.timing=myGRMINDLap1FastObs$timing,
            accuracy=myGRMINDLap1$accuracy,
            a=a,
            b=b,
            GRMa=GRMa,
            covmat=covmat,
      #Observed information matrix
      SE.M1Lap1=sqrt(diag(solve(-data$myGRMINDLap1$Amat))),
    #Fast observed information matrix
```

```
          SE.M2Lap1=sqrt(diag(solve(-data$myGRMINDLap1FastObs$Amat))),
            #Sandwich estimator from observed information matrix
SE.M3Lap1=sqrt(diag(solve(data$myGRMINDLap1$Bmat))),
                  #Sandwich estimator from observed information matrix
SE.M4Lap1=sqrt(diag(solve(-myGRMINDLap1$Amat) %*% myGRMINDLap1$Bmat
%*% solve(-myGRMINDLap1$Amat))),
             #Sandwich estimator from fast observed information matrix
         SE.M5Lap1=sqrt(diag(solve(-myGRMINDLap1FastObs$Amat) %*%
myGRMINDLap1FastObs$Bmat %*% solve(-myGRMINDLap1FastObs$Amat))),
                    #LAPLACE 2 se
                    #Observed information matrix
                    SE.M1Lap2=sqrt(diag(solve(-myGRMINDLap2$Amat))),
                    #Fast observed information matrix
                SE.M2Lap2=sqrt(diag(solve(-myGRMINDLap2FastObs$Amat))),
                    #Empirical cross-product matrix
                    SE.M3Lap2=sqrt(diag(solve(myGRMINDLap2$Bmat))),
                   #Sandwich estimator from observed information matrix
                    SE.M4Lap2=sqrt(diag(solve(-myGRMINDLap2$Amat) %*%
myGRMINDLap2$Bmat %*% solve(-myGRMINDLap2$Amat))),
               #Sandwich estimator from fast observed information matrix
               SE.M5Lap2=sqrt(diag(solve(-myGRMINDLap2FastObs$Amat) %*%
myGRMINDLap2FastObs$Bmat %*% solve(-myGRMINDLap2FastObs$Amat))))
  }else if (model.type == "CL"){
    result <- list(sample.size=sample.size,
                    nitem=nitems,
                    model.type=model.type,
```

```
                    ncategories=ncategories,
                    nfactors=nfactors,
                    accurance=myGRMCLLap1$accuracy,
                    myGRMCLLap1.par=myGRMCLLap1$par,
                    myGRMCLLap1.iter=myGRMCLLap1$iter,
                    myGRMCLLap1.Amat=myGRMCLLap1$Amat,
                    myGRMCLLap1.timing=myGRMCLLap1$timing,
                    myGRMCLLap2.iter=myGRMCLLap2$iter,
                    myGRMCLLap2.par=myGRMCLLap2$par,
                    myGRMCLLap2.Amat=myGRMCLLap2$Amat,
                    myGRMCLLap2.timing=myGRMCLLap2$timing,
                    myGRMCLLap2FastObs.iter=myGRMCLLap2FastObs$iter,
                    myGRMCLLap2FastObs.par=myGRMCLLap2FastObs$par,
                    myGRMCLLap2FastObs.Amat=myGRMCLLap2FastObs$Amat,
myGRMCLLap2FastObs.timing=myGRMCLLap2FastObs$timing,
                    myGRMCLLap1FastObs.iter=myGRMCLLap1FastObs$iter,
                    myGRMCLLap1FastObs.par=myGRMCLLap1FastObs$par,
                    myGRMCLLap1FastObs.Amat=myGRMCLLap1FastObs$Amat,
myGRMCLLap1FastObs.timing=myGRMCLLap1FastObs$timing,
                    a=a,
                    b=b,
                    GRMa=GRMa,
                    covmat=covmat,
                    #LAPLACE 1 se "CROSSLOADING" MODELS
                    #Observed information matrix
```

```
                    SE.CLM1Lap1=sqrt(diag(solve(-
data$myGRMCLLap1$Amat))),
                    #Fast observed information matrix
                    SE.CLM2Lap1=sqrt(diag(solve(-
data$myGRMCLLap1FastObs$Amat))),
              #Sandwich estimator from observed information matrix
SE.CLM3Lap1=sqrt(diag(solve(data$myGRMCLLap1$Bmat))),
             #Sandwich estimator from observed information matrix
                    SE.CLM4Lap1=sqrt(diag(solve(-myGRMCLLap1$Amat) %*%
myGRMCLLap1$Bmat %*% solve(-myGRMCLLap1$Amat))),
            #Sandwich estimator from fast observed information matrix
              SE.CLM5Lap1=sqrt(diag(solve(-myGRMCLLap1FastObs$Amat) %*%
myGRMCLLap1FastObs$Bmat %*% solve(-myGRMCLLap1FastObs$Amat))),
                    #Observed information matrix
                    SE.CLM1Lap2=sqrt(diag(solve(-myGRMCLLap2$Amat))),
                    #Fast observed information matrix
                    SE.CLM2Lap2=sqrt(diag(solve(-
myGRMCLLap2FastObs$Amat))),
                    #Empirical cross-product matrix
                    SE.CLM3Lap2=sqrt(diag(solve(myGRMCLLap2$Bmat))),
                   #Sandwich estimator from observed information matrix
                    SE.CLM4Lap2=sqrt(diag(solve(-myGRMCLLap2$Amat) %*%
myGRMCLLap2$Bmat %*% solve(-myGRMCLLap1$Amat))),
              #Sandwich estimator from fast observed information matrix
```

```
                    SE.CLM5Lap2=sqrt(diag(solve(-
myGRMCLLap2FastObs$Amat) %*% myGRMCLLap2FastObs$Bmat %*% solve(-
myGRMCLLap2FastObs$Amat))))
  }
  return(result)
}
# end of generate function
#Simulations
library(doParallel)
library(mvtnorm)
library(dplyr)
library(lamle)
mydir="/home/munyakam/k/"
#mydir="C:/Users/andre/Desktop/tbR/"
# Set up all the conditions
R = 1000
seed = sample.int(1000000, 1000)
sample.size = c(250,1000,4000)
nitems = c(12,24)
model.type = c("IND", "CL")
ncategories= c(2,5)
nfactors=3
# Register four clusters
cl <- makeCluster(30)
registerDoParallel(cl)
# Run nested foreach loops
```

```
simresults <- foreach(s=model.type,
.packages = c("lamle","doParallel","mvtnorm",  "dplyr"),
.combine = rbind) %:%
  foreach(n=sample.size,
 .packages = c("lamle", "doParallel",  "mvtnorm", "dplyr"),
          .combine = rbind)  %:%
  foreach(k=ncategories,
          .packages = c("lamle", "doParallel", "mvtnorm", "dplyr"),
          .combine = rbind)  %:%
    foreach(d=nfactors,
          .packages = c("lamle", "doParallel", "mvtnorm", "dplyr"),
          .combine = rbind)  %:%
    foreach(j=nitems,
          .packages = c("lamle", "doParallel", "mvtnorm", "dplyr"),
          .combine = rbind)  %:%
    foreach(i=1:R,
          .packages = c("lamle", "mvtnorm", "doParallel", "dplyr"),
          .combine = rbind) %dopar% {
            # Generate item parameters and data
            step1 <- generate_data(nitems=j, sample.size = n,
model.type = s, nfactors = d, ncategories=k, seed=i)
            try(saveRDS(step1, file = paste0(mydir, "rep", i,"n", n,
"cat", k, "nfac", d,"nitem", j, "mod", s, ".RDS")))
          }
# Stop the clusters
stopCluster(cl)
```

Appendix III: Supplemental Material

Table A1

Model Parameters for the Three-Dimensional Models 2-PL with 12 Indicators.

	Independent Clusters			Cross Loadings			
Variable	a_1	a_2	a_3	a_1	a_2	a_3	b
1	0.90	0.00	0.00	0.90	1.14	0.00	-0.52
2	1.29	0.00	0.00	1.29	0.00	0.00	-0.05
3	1.42	0.00	0.00	1.42	0.00	0.00	1.86
4	1.28	0.00	0.00	1.28	0.00	0.00	0.17
5	0.00	1.07	0.00	0.00	1.73	1.86	0.80
6	0.00	1.15	0.00	0.00	1.83	0.00	-0.70
7	0.00	1.50	0.00	0.00	1.71	0.00	-0.87
8	0.00	1.39	0.00	0.00	1.82	0.00	-0.17
9	0.00	0.00	1.91	1.91	0.00	0.84	0.49
10	0.00	0.00	1.14	0.00	0.00	1.55	-1.43
11	0.00	0.00	1.73	0.00	0.00	1.52	-0.05
12	0.00	0.00	1.83	0.00	0.00	0.89	-0.55

Table A2

Model Parameters for the Three-Dimensional Models GRM with 12 Indicators.

	Independent Clusters			Cross Loadings						
Variable	a_1	a_2	a_3	a_1	a_2	a_3	b_2	b_3	b_4	b_5
1	0.90	0.00	0.00	0.90	1.14	0.00	2.70	1.28	-0.48	-1.62
2	1.29	0.00	0.00	1.29	0.00	0.00	2.03	1.21	-0.57	-1.88
3	1.42	0.00	0.00	1.42	0.00	0.00	2.21	1.38	-0.24	-2.35
4	1.28	0.00	0.00	1.28	0.00	0.00	2.81	0.73	-0.43	-2.11
5	0.00	1.07	0.00	0.00	1.73	1.86	2.71	1.34	-0.76	-2.49
6	0.00	1.15	0.00	0.00	1.83	0.00	2.54	1.36	-0.23	-2.26
7	0.00	1.50	0.00	0.00	1.71	0.00	2.10	0.65	-0.96	-1.76
8	0.00	1.39	0.00	0.00	1.82	0.00	2.88	0.75	-0.75	-2.28
9	0.00	0.00	1.91	1.91	0.00	0.84	2.72	1.47	-0.29	-1.96
10	0.00	0.00	1.14	0.00	0.00	1.55	2.18	1.11	-0.23	-2.19
11	0.00	0.00	1.73	0.00	0.00	1.52	2.74	0.70	-0.26	-2.05
12	0.00	0.00	1.83	0.00	0.00	0.89	2.37	0.68	-0.35	-2.26

Table A3

Model Parameters for the Three-Dimensional Models 2-PL with 24 Indicators.

	Independent Clusters			Cross Loadings			
Variable	a_1	a_2	a_3	a_1	a_2	a_3	b
1	0.90	0.00	0.00	0.90	1.14	0.00	-0.52
2	1.29	0.00	0.00	1.29	0.00	0.00	-0.05
3	1.42	0.00	0.00	1.42	0.00	0.00	1.86
4	1.28	0.00	0.00	1.28	0.00	0.00	0.17
5	1.07	0.00	0.00	1.07	0.00	0.00	0.80
6	1.15	0.00	0.00	1.15	0.00	0.00	-0.70
7	1.50	0.00	0.00	1.50	0.00	0.00	-0.87
8	1.39	0.00	0.00	1.39	0.00	0.00	-0.17
9	0.00	1.91	0.00	0.00	1.73	1.86	0.49
10	0.00	1.14	0.00	0.00	1.83	0.00	-1.43
11	0.00	1.73	0.00	0.00	1.71	0.00	-0.05
12	0.00	1.83	0.00	0.00	1.82	0.00	-0.55
13	0.00	1.71	0.00	0.00	1.29	0.00	0.32
14	0.00	1.82	0.00	0.00	0.87	0.00	-0.67
15	0.00	1.29	0.00	0.00	1.49	0.00	-0.17
16	0.00	0.87	0.00	0.00	1.69	0.00	0.09
17	0.00	0.00	1.49	1.91	0.00	0.84	-0.57
18	0.00	0.00	1.69	0.00	0.00	1.55	0.70
19	0.00	0.00	1.86	0.00	0.00	1.52	-0.09
20	0.00	0.00	0.84	0.00	0.00	0.89	-0.74
21	0.00	0.00	1.55	0.00	0.00	1.30	1.26
22	0.00	0.00	1.52	0.00	0.00	1.21	1.78
23	0.00	0.00	0.89	0.00	0.00	1.04	-1.20
24	0.00	0.00	1.30	0.00	0.00	1.81	0.68

Table A4

Model Parameters for the Three-Dimensional Models GRM with 24 Indicators.

	Independent Clusters			Cross Loadings						
Variable	a_1	a_2	a_3	a_1	a_2	a_3	b_2	b_3	b_4	b_5
1	0.90	0.00	0.00	0.90	1.14	0.00	2.70	1.28	-0.48	-1.62
2	1.29	0.00	0.00	1.29	0.00	0.00	2.03	1.21	-0.57	-1.88
3	1.42	0.00	0.00	1.42	0.00	0.00	2.21	1.38	-0.24	-2.35
4	1.28	0.00	0.00	1.28	0.00	0.00	2.81	0.73	-0.43	-2.11
5	1.07	0.00	0.00	1.07	0.00	0.00	2.31	0.79	-0.08	-2.11
6	1.15	0.00	0.00	1.15	0.00	0.00	2.52	0.69	-0.29	-2.03
7	1.50	0.00	0.00	1.50	0.00	0.00	2.37	0.90	-0.25	-1.50
8	1.39	0.00	0.00	1.39	0.00	0.00	2.57	1.26	-0.54	-1.95
9	0.00	1.91	0.00	0.00	1.73	1.86	2.71	1.34	-0.76	-2.49
10	0.00	1.14	0.00	0.00	1.83	0.00	2.54	1.36	-0.23	-2.26
11	0.00	1.73	0.00	0.00	1.71	0.00	2.10	0.65	-0.96	-1.76
12	0.00	1.83	0.00	0.00	1.82	0.00	2.88	0.75	-0.75	-2.28
13	0.00	1.71	0.00	0.00	1.29	0.00	2.06	0.58	-0.17	-1.90
14	0.00	1.82	0.00	0.00	0.87	0.00	2.20	1.10	-0.32	-2.14
15	0.00	1.29	0.00	0.00	1.49	0.00	2.05	1.12	-0.09	-1.91
16	0.00	0.87	0.00	0.00	1.69	0.00	2.75	0.58	-0.53	-1.51
17	0.00	0.00	1.49	1.91	0.00	0.84	2.72	1.47	-0.29	-1.96
18	0.00	0.00	1.69	0.00	0.00	1.55	2.18	1.11	-0.23	-2.19
19	0.00	0.00	1.86	0.00	0.00	1.52	2.74	0.70	-0.26	-2.05
20	0.00	0.00	0.84	0.00	0.00	0.89	2.37	0.68	-0.35	-2.26
21	0.00	0.00	1.55	0.00	0.00	1.30	2.23	1.48	-0.20	-1.68
22	0.00	0.00	1.52	0.00	0.00	1.21	2.73	0.87	-0.90	-1.78
23	0.00	0.00	0.89	0.00	0.00	1.04	2.29	1.43	-0.56	-2.16
24	0.00	0.00	1.30	0.00	0.00	1.81	2.90	0.75	-0.68	-2.18

Table A5

Covariance Matrix Used in the 3-Dimensional Simulation

LV	F1	F2	F3
F1	1.00	0.45	0.47
F2	0.45	1.00	0.51
F3	0.47	0.51	1.00

Note. LV= latent variable, F1= latent variable one, F2=latent variable two and F3 = latent variable three.