# Appendix: Supplemental Materials

# Code Examples

Libraries needed for this example:

library(dplyr)

library(lavaan)

library(visualizemi)

library(dmacs)

library(ggplot2)

library(papaja) # mostly for markdown formatting

## Simulating from Models

Here’s an example of how to simulate directly from a lavaan model:

# first build your model  
# this example is separate for each group  
model.invariant.g1 <- "  
# loadings  
lv =~ .8\*q1 + .4\*q2 + .6\*q3 + .3\*q4 + .6\*q5   
# set the residual for invariance on q4  
q4 ~~ 1\*q4  
# set the intercept for invariance on q4  
q4 ~ 0\*1  
# set the intercept to zero for df purposes  
q1 ~ 0\*1  
# allow the latent mean to be estimated   
lv ~ 1"  
model.invariant.g2 <- "lv =~ .77\*q1 + .43\*q2 + .58\*q3 + .3\*q4 + .61\*q5  
q4 ~~ 1\*q4  
q4 ~ 0\*1  
q1 ~ 0\*1  
lv ~ 1"  
  
# simulate data invariant separately for each group  
df.invariant <- bind\_rows(  
 # lavaan function   
 simulateData(  
 # model with estimates   
 model = model.invariant.g1,   
 # how many data points  
 sample.nobs = 250,   
 # mean structure for mgcfa models   
 meanstructure = T,   
 # model type  
 model.type = "cfa",  
 # set seed for reproducibility   
 seed = 1234) %>%   
 # add a group label to the data   
 mutate(group = "Group 1"),   
 simulateData(  
 model = model.invariant.g2,   
 sample.nobs = 250,   
 meanstructure = T,   
 model.type = "cfa",  
 seed = 1234) %>%   
 mutate(group = "Group 2")   
)

## Simulating from Matrices

Here’s an example of how to simulate using MASS and covariance or correlation matrices.

library(MASS)  
  
# covariance matrix  
university.cov <- lav\_matrix\_lower2full(  
 c(169.00,   
 73.710, 182.2500,  
 73.229, 88.4250, 171.6100,  
 63.375, 72.5625, 127.7250, 156.2500,  
 42.120, 67.4325, 122.0265, 123.1875, 182.2500,  
 57.226, 63.2610, 117.1926, 154.4250, 138.0240, 201.6400,  
 30.875, 32.0625, 60.9805, 62.9375, 76.9500, 79.5910, 90.2500,  
 36.075, 38.9610, 61.0722, 58.2750, 65.9340, 70.9290, 81.1965, 123.2100,  
 18.096, 21.1410, 26.2131, 39.1500, 44.6310, 46.9452, 48.7635, 56.0106, 75.6900))  
  
# give it names  
rownames(university.cov) <-  
 colnames(university.cov) <-  
 c("class", "social", "learn", "chronic", "physical", "sex",   
 "depression", "anxiety", "stress")  
  
# means - you need standard deviation if you only have a correlation matrix   
university.means <- c(3.4, 4.3, 3.7, 3.2, 4.5, 1.2, 4.0, 3.5, 4.2)  
  
# use mass function  
DF <- mvrnorm(n = 200, mu = university.means, Sigma = university.cov)  
  
head(DF)

## class social learn chronic physical sex  
## [1,] 9.332202 6.7442501 8.001600 6.771713 -10.833804 -3.837477  
## [2,] 4.873877 -0.1368433 11.690145 14.572699 13.356779 22.699918  
## [3,] 13.650244 -5.5841743 -2.295967 -3.914620 -6.552379 -11.711882  
## [4,] 7.644520 -1.6201790 -15.075033 4.010138 4.741793 11.704179  
## [5,] -10.491240 14.6367273 2.951522 10.934949 1.153787 3.637487  
## [6,] 3.188521 -0.2078648 -11.655781 -8.085560 -12.482893 -18.914448  
## depression anxiety stress  
## [1,] -9.2071866 -9.0853468 -10.016408  
## [2,] -5.7266089 -5.1086786 -3.757830  
## [3,] -12.3350189 -9.5855529 -10.855687  
## [4,] -0.8850695 -8.8631134 -1.046865  
## [5,] -3.4707411 -0.1493184 -2.300569  
## [6,] -9.1500394 0.3167367 5.590050

## MGCFA: mgcfa() Function

In this example, we make our example model using *lavaan* syntax. The lv latent variable predicts the five measured variables, which are present as columns in our df.invariant data set.

*lavaan* automatically sets the mean (i.e., the intercept) for latent variables to zero. If we wish to visualize the impact of the changes in parameter estimates across groups on the latent means, we need to allow the latent mean estimation with lv ~ 1. However, adding this estimation into our model will create a non-identified model. To solve this problem, you can set one of the intercepts of another variable to a value to scale the model. Here we will set the scale of the model by using q1 ~ 0\*1, thus, scaling the expected means to zero. With simulation, this step is easy to know which variable to pick - we set the intercept on the variable we know did not show differences. In real data, you may wish to run the model steps *without* setting this option, examine the results of a configural or separate models, and then add the option for the values most similar. Additionally, you could complete partial invariance steps to determine which value appears most consistent to fix the estimate.

# create lavaan model  
model.overall <- "  
# overall one-factor model  
lv =~ q1 + q2 + q3 + q4 + q5  
# set the intercept (mean) of q1 to zero  
q1 ~ 0\*1  
# allow the lv intercept to be freely estimated  
lv ~ 1"  
# look at the data  
head(df.invariant)

## q1 q2 q3 q4 q5 group  
## 1 -0.8903542 -0.81707530 0.06137292 -1.3236407 -1.7916418 Group 1  
## 2 1.1054521 -0.03540948 -0.81299606 1.0028340 -0.1909127 Group 1  
## 3 1.4555852 1.54083484 1.59084213 -0.3345967 -0.6865496 Group 1  
## 4 -1.8745187 -1.27880245 -2.53565792 -1.0024193 -1.6253249 Group 1  
## 5 -0.4449517 -0.17782974 1.05507079 -1.2615705 1.7536428 Group 1  
## 6 0.2278813 0.71348845 1.63251893 0.6449847 -1.0055700 Group 1

The mgcfa() function is designed to flexibly allow you to leverage *lavaan*’s package functions to calculate multiple measurement steps at once. You would include:

1. the model syntax in the model argument.
2. the dataframe in the data argument of our function.
3. the name of the grouping variable in quotes for group.
4. and the equality constraints you would like to impose in order in group.equal.
5. ... any other *lavaan* arguments you would like to use such as meanstructure or estimator.

Note: you can also use sample.cov, sample.mean, sample.nobs in this step for estimation of multigroup models, but simulated dataframes are needed for bootstrapping replication estimates.

# run our mgcfa function to run all models  
results.invariant <-   
 # name of the saved model syntax  
 mgcfa(model = model.overall,   
 # name of the dataframe  
 data = df.invariant,  
 # name of the grouping variable  
 group = "group",  
 # equality constraints to impose in order  
 group.equal = c("loadings", "intercepts", "residuals"),  
 # other options to send to lavaan cfa function  
 meanstructure = T)  
  
# what is saved for you  
names(results.invariant)

## [1] "model\_coef" "model\_fit" "model\_overall"   
## [4] "group\_models" "model\_configural" "invariance\_models"

1. model\_coef: The parameter estimates for each model with the model step included in a *model* column. This set of coefficients can be used for other functions. This dataframe is created with *broom*’s tidy() function if you wish to recreate this table without running the mgcfa() function (Robinson et al., 2023).

results.invariant$model\_coef[1:10 , ]

## # A tibble: 10 × 12  
## term op estimate std.error statistic p.value std.lv std.all model  
## <chr> <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <chr>  
## 1 "lv =~ q1" =~ 1 0 NA NA 0.803 0.616 Over…  
## 2 "lv =~ q2" =~ 0.655 0.0880 7.44 9.77e-14 0.526 0.493 Over…  
## 3 "lv =~ q3" =~ 0.640 0.0895 7.15 8.83e-13 0.514 0.463 Over…  
## 4 "lv =~ q4" =~ 0.277 0.0749 3.69 2.24e- 4 0.222 0.209 Over…  
## 5 "lv =~ q5" =~ 0.955 0.117 8.13 4.44e-16 0.766 0.656 Over…  
## 6 "q1 ~1 " ~1 0 0 NA NA 0 0 Over…  
## 7 "lv ~1 " ~1 -0.0305 0.0582 -0.524 6.00e- 1 -0.0380 -0.0380 Over…  
## 8 "q1 ~~ q1" ~~ 1.05 0.0995 10.6 0 1.05 0.620 Over…  
## 9 "q2 ~~ q2" ~~ 0.860 0.0653 13.2 0 0.860 0.757 Over…  
## 10 "q3 ~~ q3" ~~ 0.966 0.0711 13.6 0 0.966 0.785 Over…  
## # ℹ 3 more variables: block <int>, group <int>, label <chr>

1. model\_fit: The model fit indices from fitmeasures() to review for overall model fit and invariance judgments. The name of the model is included in a *model* column.

head(results.invariant$model\_fit)

## # A tibble: 6 × 18  
## agfi AIC BIC cfi chisq npar rmsea rmsea.conf.high srmr tli  
## <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 0.998 7516. 7580. 1 0.650 15 0 0 0.00616 1.04   
## 2 0.948 3766. 3819. 0.976 7.79 15 0.0473 0.108 0.0312 0.953  
## 3 0.974 3768. 3820. 1 4.48 15 0 0.0831 0.0210 1.01   
## 4 0.961 7533. 7660. 0.991 12.3 30 0.0301 0.0785 0.0261 0.982  
## 5 0.965 7528. 7638. 0.994 15.4 26 0.0200 0.0660 0.0330 0.992  
## 6 0.969 7522. 7615. 1 17.3 22 0 0.0542 0.0352 1.00   
## # ℹ 8 more variables: converged <lgl>, estimator <chr>, ngroups <int>,  
## # missing\_method <chr>, nobs <int>, norig <int>, nexcluded <int>, model <chr>

1. model\_overall: A saved *lavaan* fitted model of all groups together without any equality constraints or grouping variables. These objects can be used with any function that normally takes a saved model: parameterEstimates(), modificationIndices(), semPlot::semPaths(), and so on (Epskamp, 2022).

class(results.invariant$model\_overall)

## [1] "lavaan"  
## attr(,"package")  
## [1] "lavaan"

1. group\_models: A list of saved fitted models for each group separately.

names(results.invariant$group\_models)

## [1] "model.Group 1" "model.Group 2"

1. model\_configural: A saved fitted model for the configural model that nests together each group into one model with no other constraints.

class(results.invariant$model\_configural)

## [1] "lavaan"  
## attr(,"package")  
## [1] "lavaan"

1. invariance\_models: A list of saved fitted models that consecutively adds group.equal constraints.

names(results.invariant$invariance\_models)

## [1] "model.loadings" "model.intercepts" "model.residuals"

## Partial Invariance: partial\_mi() Function

The partial\_mi() function aids in the calculation of partial invariance for a specific step of the MGCFA process. The function includes the following arguments:

1. saved\_model: The saved *lavaan* model with the equality constraints at the level of measurement invariance you would like to examine for partial invariance.
2. data: The dataframe where the model was estimated.
3. model: The model syntax for the overall model.
4. group: The grouping variable column in the dataframe.
5. group.equal: The equality constraints including in your original multigroup tests.
6. partial\_step: The level of partial invariance you wish to test.

partial.invariant <-  
 partial\_mi(  
 # saved model output with constraints  
 saved\_model = results.invariant$invariance\_models$model.residuals,  
 # dataframe from model   
 data = df.invariant,  
 # model syntax   
 model = model.overall,  
 # group column name   
 group = "group",  
 # group equality constraints from your mgcfa  
 group.equal = c("loadings", "intercepts", "residuals"),  
 # which step you want to examine for partial invariance  
 partial\_step = "residuals"  
 )  
  
names(partial.invariant)

## [1] "models" "fit\_table"

In this function, each parameter with the appropriate *lavaan* syntax is relaxed individually (i.e., ~1 for intercepts, ~~ for residuals, etc.). The fitted models are saved in the models output, and the fit\_table output includes all fit indices for each model to investigate potential areas of partial invariance based on the researcher’s desired criterion.

names(partial.invariant$models)

## [1] "q1 ~~ q1" "q2 ~~ q2" "q3 ~~ q3" "q4 ~~ q4" "q5 ~~ q5" "lv ~~ lv"

head(partial.invariant$fit\_table %>%   
 dplyr::select(free.parameter, cfi, rmsea))

## # A tibble: 6 × 3  
## free.parameter cfi rmsea   
## <chr> <lvn.vctr> <lvn.vctr>  
## 1 q1 ~~ q1 0.9902679 0.02108648  
## 2 q2 ~~ q2 0.9868905 0.02447336  
## 3 q3 ~~ q3 0.9958241 0.01381266  
## 4 q4 ~~ q4 1.0000000 0.00000000  
## 5 q5 ~~ q5 0.9868088 0.02454944  
## 6 lv ~~ lv 0.9906154 0.02025143

Note: the partial\_step function is used to determine which types of op or operators to freely estimate between groups. If one chooses residuals, you will also freely estimate the residual for the latent variable or any other residuals found in the model. These items may be ignored if they were not meant to be included.

## Visualization of Invariance: plot\_mi() Functions

Once we know which items are non-invariant, the model\_coef output from the mgcfa() can be used directly in plot\_mi(). The plot outputs will be described below. First, here are the arguments for the function:

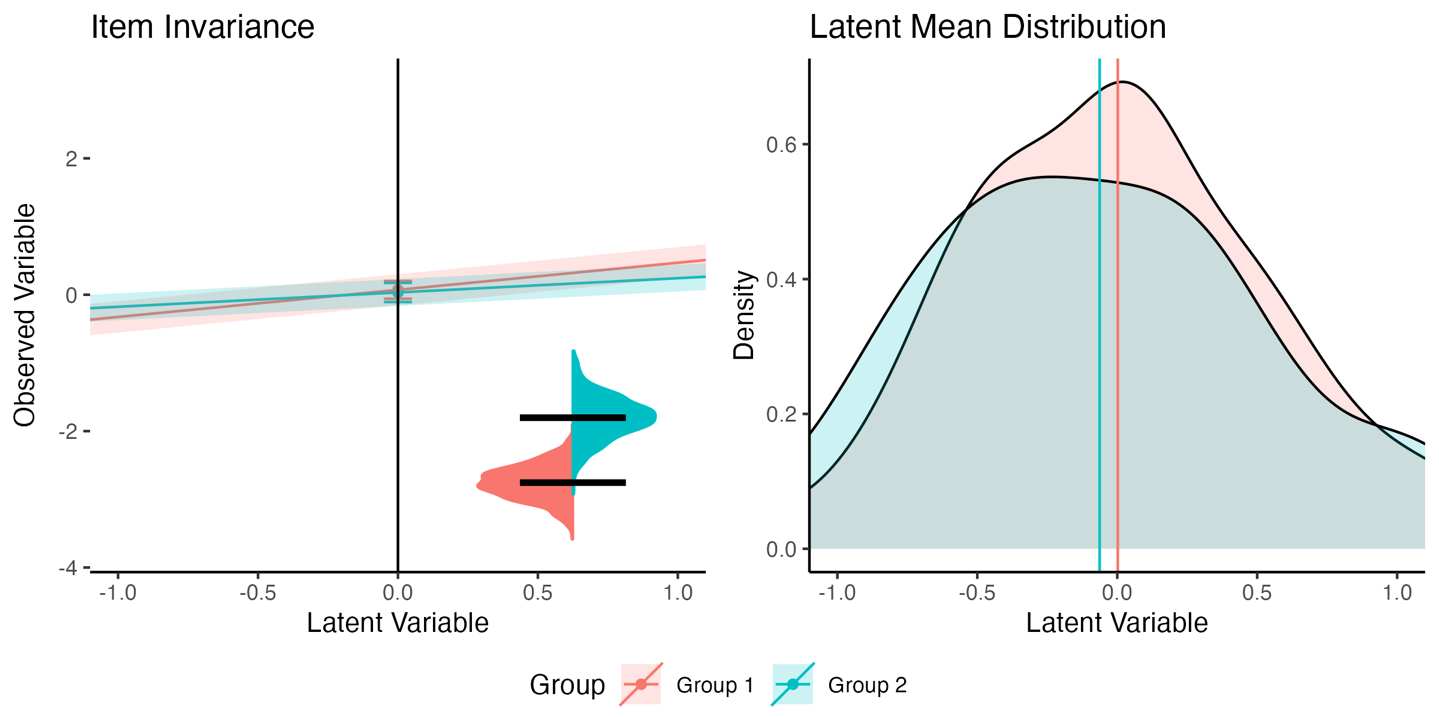
1. data\_coef: A tidy dataframe of the parameter estimates from the models. This function assumes you have used broom::tidy() on the saved model from *lavaan* and added a column called “model” with the name of the model step (Robinson et al., 2023). This function will only run for models that have used the grouping function (i.e., configural, metric, scalar, and strict or other combinations/steps you wish to examine).
2. model\_step: Which model do you want to plot? You should match this name to the one you want to extract from your model column in the data\_coef.
3. item\_name: Which observed variable from your model syntax do you want to plot? Please list this variable name exactly how it appears in the model.
4. x\_limits: What do you want the x-axis limits to be for your invariance plot? The default option is to assume the latent variable is standardized, and therefore, -1 to 1 is recommended. Use only two numbers, a lower and upper limit. This value also constrains the latent mean diagram to help zoom in on group differences because the scale of latent means is usually centered over zero. You can use this parameter to zoom out to a more traditional histogram using c(-2, 2).
5. y\_limits: What do you want the y-axis limits to be for your invariance plot? Given that the latent variable is used to predict the observed values in the data, you could use the minimum and maximum values found in the data. If that range is large, consider reducing this value to be able to visualize the results (i.e., otherwise it may be too zoomed out to judge group differences). Use only two numbers, a lower and upper limit.
6. conf.level: What confidence limit do you want to plot? Use 1 - .
7. model\_results: In this argument, include the saved *lavaan* output for the model listed in the model\_step argument.
8. lv\_name: Include the name of the latent variable, exactly how it is listed in your *lavaan* syntax. You should plot the latent variable that the item\_name is linked to. If you have items that load onto multiple latent variables, you will need to make multiple plots.
9. plot\_groups: If you include more than two groups in a multigroup model, the automatic assumption is that you want the first two groups for this visualization. If not, include the names of the groups here to plot.

invariant.plot <-   
 plot\_mi(  
 # output from model\_coef  
 data\_coef = results.invariant$model\_coef,   
 # which model do you want to plot  
 model\_step = "Configural",   
 # name of observed item  
 item\_name = "q4",   
 # latent variable limits to graph  
 x\_limits = c(-1,1),   
 # Y min and max in data   
 y\_limits = c(min(df.invariant$q4), max(df.invariant$q4)),  
 # what ci do you want  
 conf.level = .95,   
 # what model results do you want   
 model\_results = results.invariant$model\_configural,  
 # which latent variable do you want   
 lv\_name = "lv"   
)  
  
names(invariant.plot)

## [1] "complete" "intercept" "mean" "variance"

The outputs from this function are several *ggplot2* objects that can be edited or saved directly using *ggplot2* functionality (Wickham, 2016).

1. complete: The output from this model can be found in Figure 11. On the left-hand side, the item invariance is plotted, and on the right-hand side, the latent mean distributions for the two groups are plotted. In the item invariance sub-plot, the visualization includes all three components traditionally seen in MGCFA testing steps: loadings, intercepts, and residuals. Each visualization element was designed to match the traditional visualization for that type of output. All parameter estimates are plotted on the unstandardized estimates and their confidence interval based on the standard error of the estimate. All plots are made with *ggplot2* and *cowplot* (Wilke, 2020).

*Figure 11*. Invariant Model Visualization

1. intercept: Only the left-hand side of the complete plot designed to represent intercepts and factor loadings. Factor loadings represent the slope of the regression equation for the latent variable predicting the scores on the observed variable (). The y-axis indicates the observed variable scores, and here, the plot includes the entire range of the scale of the data for item four. The coefficient () for group 1 was 0.40, while the coefficient for group 2 was 0.21. The ribbon bands around the plotted slopes indicate the confidence interval for that estimate. In this plot, while the coefficients for each group are not literally equal, the overlapping and parallel slope bands indicate they are not different practically.

The item intercepts () are plotted on the middle line where they would cross the y-axis at a latent variable score of zero. These are represented by a dot with a set of confidence error bars around the point. The intercept for group 1 was 0.07, while the coefficient for group 2 was 0.03. In this invariant depiction, the overlap in the intercepts is clear, indicating they are not different. You can use y\_limits to zoom in on the graph if these are too small to be distinguishable.

1. mean: The right-hand side of the complete plot graphing the latent variable means and density from the data. The latent variable is shown on the x-axis using standardized values (i.e., *z*-scores) where -1 indicates one standard deviation below the mean for the latent variable, 0 indicates the mean for the latent variable and so on. The lines indicate the means of the latent variables from the simulated dataset. Group labels are represented in the figure caption on the bottom. Group 1 is usually the group that is alphabetically first in the data set or whichever group is the first that appears when using the levels() command.
2. variance: A split geom violin plot indicating the variance distribution of the plotted item. Residuals are trickier to plot, as they are the left over error when predicting the observed variables . It is tempting to plot this value as the confidence band around the slope, however, that defeats the purpose of understanding that the slopes are estimated separately from the residuals, and both have an associated variability around their parameter estimate. Therefore, residuals are represented in the inset picture at the bottom right of the item invariance plot. The black bars represent the estimated residual for each group (group 1: 0.91, group 2: 1.22). The distributions are plotted to represent the normal spread of values using the standard error of the residuals. The violin plot allows for direct comparison of those residuals and their potential distributions. Note that the placement has nothing to do with the x or y-axis and is designed to always show in the same location, regardless of size/value. The plots are included separately so they can be arranged in a different fashion if desired.

## Model Replication and Effect Sizes: bootstrap\_model() Function

The bootstrap\_model function in *visualizemi* was designed to estimate the likely replication of overall model invariance with the assumption that the data used for the estimation represents the larger population. The following arguments are used:

1. saved\_configural: a saved fitted model at the configural level with no equality constraints. This model should include all other lavaan settings you would like to use, such as estimator or ordered.
2. data: The dataframe where the model was estimated.
3. model: The model syntax for the overall model.
4. group: The grouping variable column in the dataframe.
5. nboot: The number of bootstraps to run.
6. invariance\_index: The fit index you would like to use to determine invariance. Please use options and labeling from *lavaan* - see fitmeasures() for options.
7. invariance\_rule: The invariance difference score you would like to use as your rule.
8. group.equal: The equality constraints including in your original multigroup tests.

boot.model.invariant <-   
 bootstrap\_model(  
 # saved configural model   
 saved\_configural = results.invariant$model\_configural,  
 # dataframe  
 data = df.invariant,  
 # model syntax  
 model = model.overall,   
 # group variable column in dataframe  
 group = "group",  
 # number of bootstraps  
 nboot = 1000,   
 # which fit index you would like to use  
 invariance\_index = "cfi",  
 # what is your criterion for that fit index  
 invariance\_rule = .01,  
 # what equality constraints are you testing   
 group.equal = c("loadings", "intercepts", "resduals")  
 )

The data included in this function will be sampled, with replacement, at the same size as the current dataset, and the included invariance equality constraints are estimated. Each step will be compared to the previous step using the invariance index and comparison rule entered. The output is a dataframe of the proportion of non-invariant bootstraps from the real data and the same bootstrapped dataset with the group labels randomly assigned. The effect size comparison of proportions, , for non-invariant comparisons:

The alternative, , for effect size of measurement invariance replication would simply be the inverse sign of and is also included in the table. Two additional columns and represent the values divided by the upper bound of (i.e., ), to help with interpretation of the effect size (thus, bounding to -1 to 1).

## Parameter Replication and Effect Sizes: bootstrap\_partial() Function

After examining the overall model potential replication effect size, the individual parameters within a model can be bootstrapped for partial invariance to with that parameter relaxed (overall partial model statistics) and the difference in group parameter estimates (parameter effect size). This function uses arguments seen in other functions, so they will not be repeated here. The general setup consists of using the model you think could be partially invariant in the saved\_model argument and the fit index for comparison for the model with less constraints in invariance\_compare. This example examines the loadings in the invariant model, so saved\_model uses the mgcfa output for equality constraints present on the loadings and compares that model to the configural model with no equality constraints on the loadings. The partial\_step argument will be used to determine which operation syntax (i.e. =~ for loadings) to relax for modeling.

boot.partial.invariant <-   
 bootstrap\_partial(  
 # saved model you want to examine the partial loadings for   
 saved\_model = results.invariant$invariance\_models$model.loadings,  
 # the dataset   
 data = df.invariant,   
 # the model   
 model = model.overall,  
 # the group variable in the dataset  
 group = "group",   
 # number of bootstraps  
 nboot = 1000,  
 # which fit index you would like to use to determine partial invariance  
 invariance\_index = "cfi",   
 # what is the invariance rule   
 invariance\_rule = .01,   
 # what are we comparing the saved model fit index to   
 invariance\_compare = fitmeasures(results.invariant$model\_configural, "cfi"),   
 # what step are we using for invariance  
 partial\_step = "loadings",   
 # what equality constraints should be imposed   
 group.equal = c("loadings")  
 )

names(boot.partial.invariant)

## [1] "invariance\_plot" "effect\_invariance\_plot" "density\_plot"   
## [4] "boot\_DF" "boot\_summary" "boot\_effects"

The saved output includes several dataframes and plots. The first is the boot\_DF which the summary of each run in a dataframe for plotting or summarization. This dataframe includes the estimate for each parameter (term) separated by group and type (boot\_1, boot\_2 are the bootstrapped estimates for group 1 and group 2, while the same random columns indicate the randomly assigned groups). The fit index used to determine invariance is included for bootstrapped and random estimates, and then the differences between groups and if they were “invariant” or not given the researcher supplied rule.

head(boot.partial.invariant$boot\_DF)

## term boot\_1 boot\_2 random\_1 random\_2 boot\_fit random\_fit  
## 1 lv =~ q1 0.4548783 0.49928877 0.4627486 0.4651391 0.9296990 1.0000000  
## 2 lv =~ q2 0.3599017 0.56241016 0.4100874 0.4980215 0.9441125 1.0000000  
## 3 lv =~ q3 0.4254283 0.33640233 0.4274329 0.3422124 0.9377130 1.0000000  
## 4 lv =~ q4 0.3930716 0.03320619 0.1380833 0.2628802 0.9750274 1.0000000  
## 5 lv =~ q5 0.7306414 0.73512673 0.7093891 0.7532471 0.9266587 1.0000000  
## 6 lv =~ q1 0.5537083 0.57086815 0.5732166 0.5475714 0.8958929 0.9814658  
## boot\_difference random\_difference boot\_index\_difference  
## 1 -0.044410454 -0.002390463 FALSE  
## 2 -0.202508484 -0.087934027 FALSE  
## 3 0.089025927 0.085220565 FALSE  
## 4 0.359865463 -0.124796846 FALSE  
## 5 -0.004485377 -0.043857947 FALSE  
## 6 -0.017159815 0.025645271 FALSE  
## random\_index\_difference  
## 1 TRUE  
## 2 TRUE  
## 3 TRUE  
## 4 TRUE  
## 5 TRUE  
## 6 TRUE

Next, the boot\_summary includes a summarized form of the bootstrapped results from separated by bootstrapping versus random and invariant/non-invariant. The for between groups Cohen’s is shown below, and the non-central confidence interval is included. Effect sizes are only calculated when the number of bootstrapped estimates is at least 10% of the data - therefore, you would not receive effect sizes with almost no bootstrapped runs. This dataframe should be used to determine which parameter may be different and at what size between groups in a replication of the study.

boot.partial.invariant$boot\_summary %>%   
 dplyr::select(term, d\_boot, d\_random)

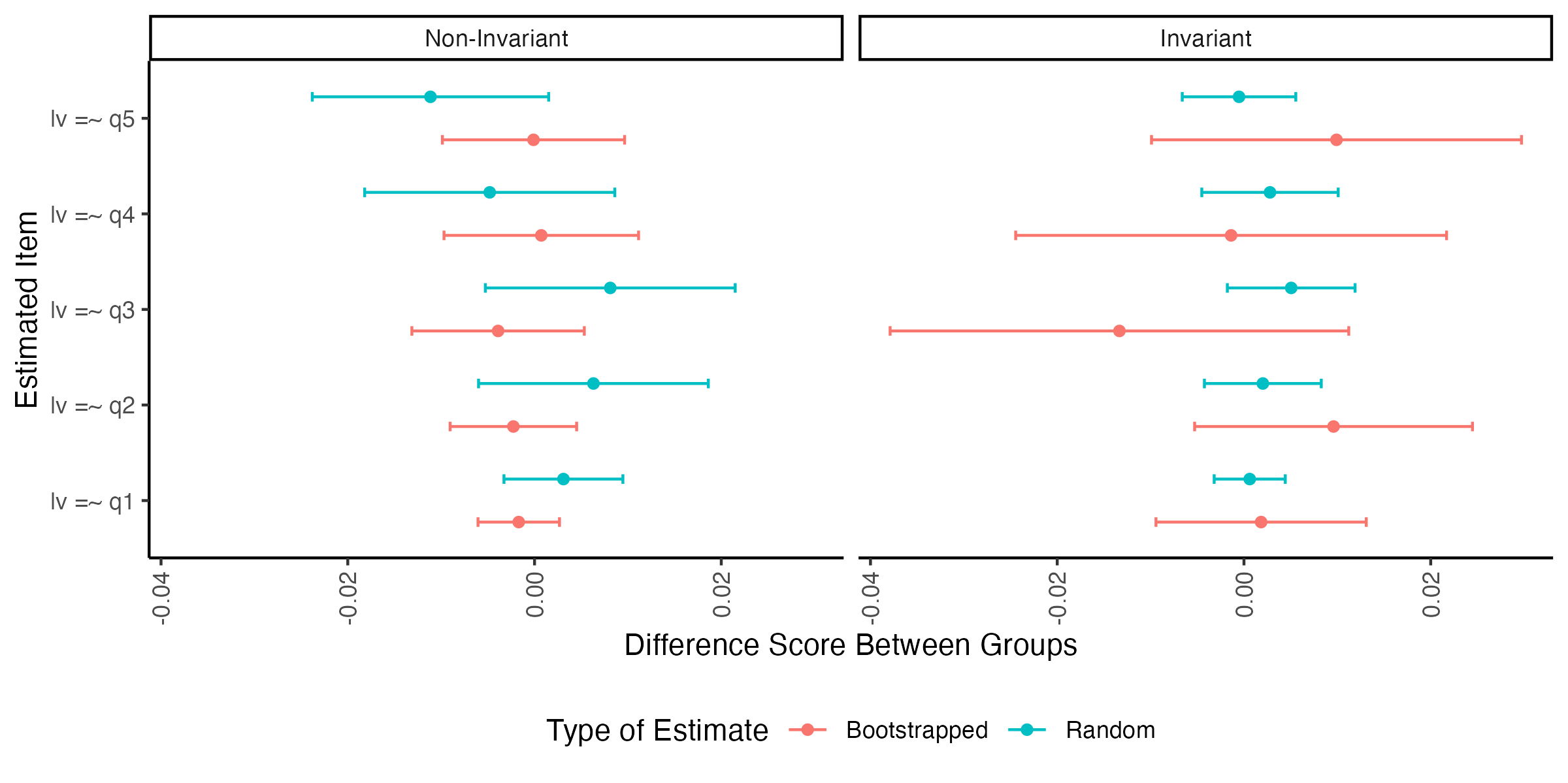
## term d\_boot d\_random  
## 1 lv =~ q1 -0.029853316 0.058271662  
## 2 lv =~ q1 0.033742666 0.011640524  
## 3 lv =~ q2 -0.032613505 0.093288563  
## 4 lv =~ q2 0.146200211 0.030925365  
## 5 lv =~ q3 -0.046329761 0.112823117  
## 6 lv =~ q3 -0.148330246 0.074265298  
## 7 lv =~ q4 0.007851687 -0.066844445  
## 8 lv =~ q4 -0.015702705 0.038884725  
## 9 lv =~ q5 -0.001285809 -0.169307690  
## 10 lv =~ q5 0.122405579 -0.008526218

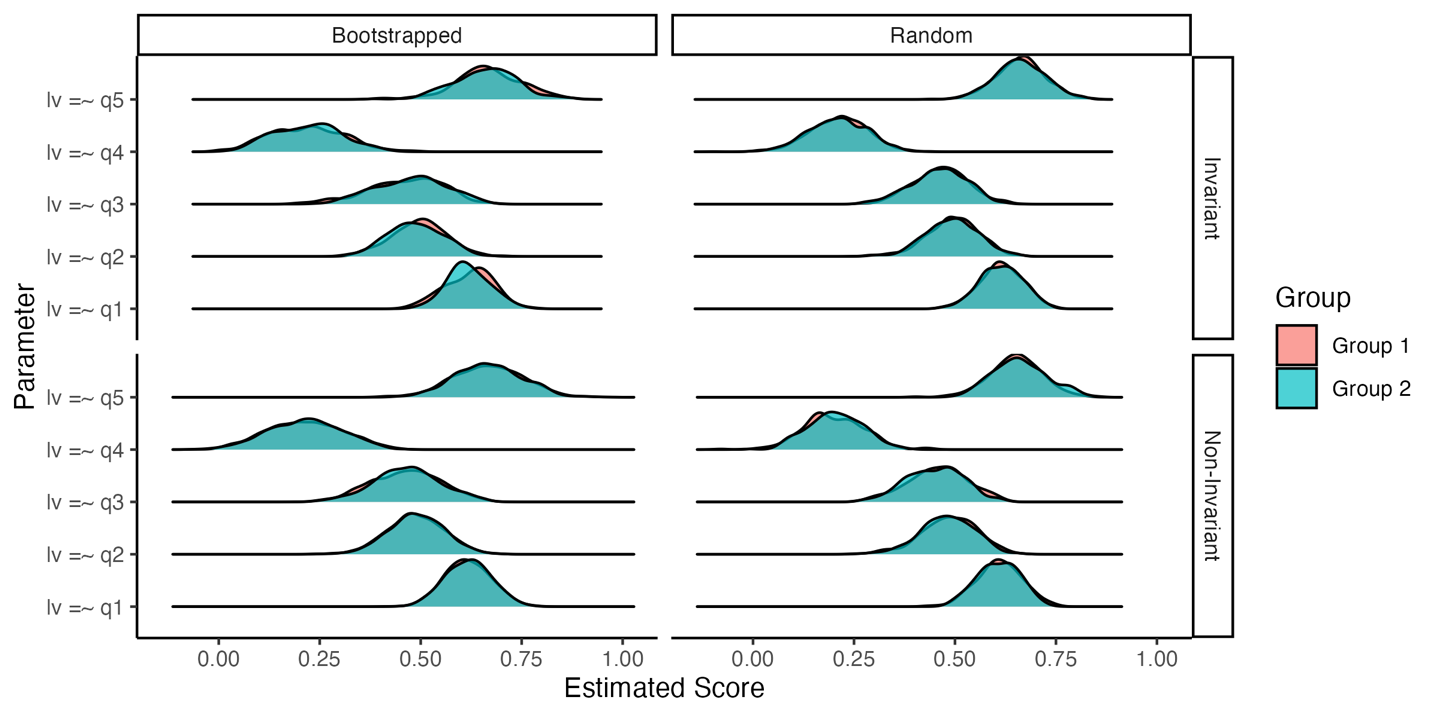
The boot\_effects table creates a summary similar to the overall model replication table based on the proportion of runs that were considered invariant versus not for each parameter. Note that the effects match the overall results, such that simulated invariant data appears to still show the likelihood that loadings may not replicate in a similar dataset.

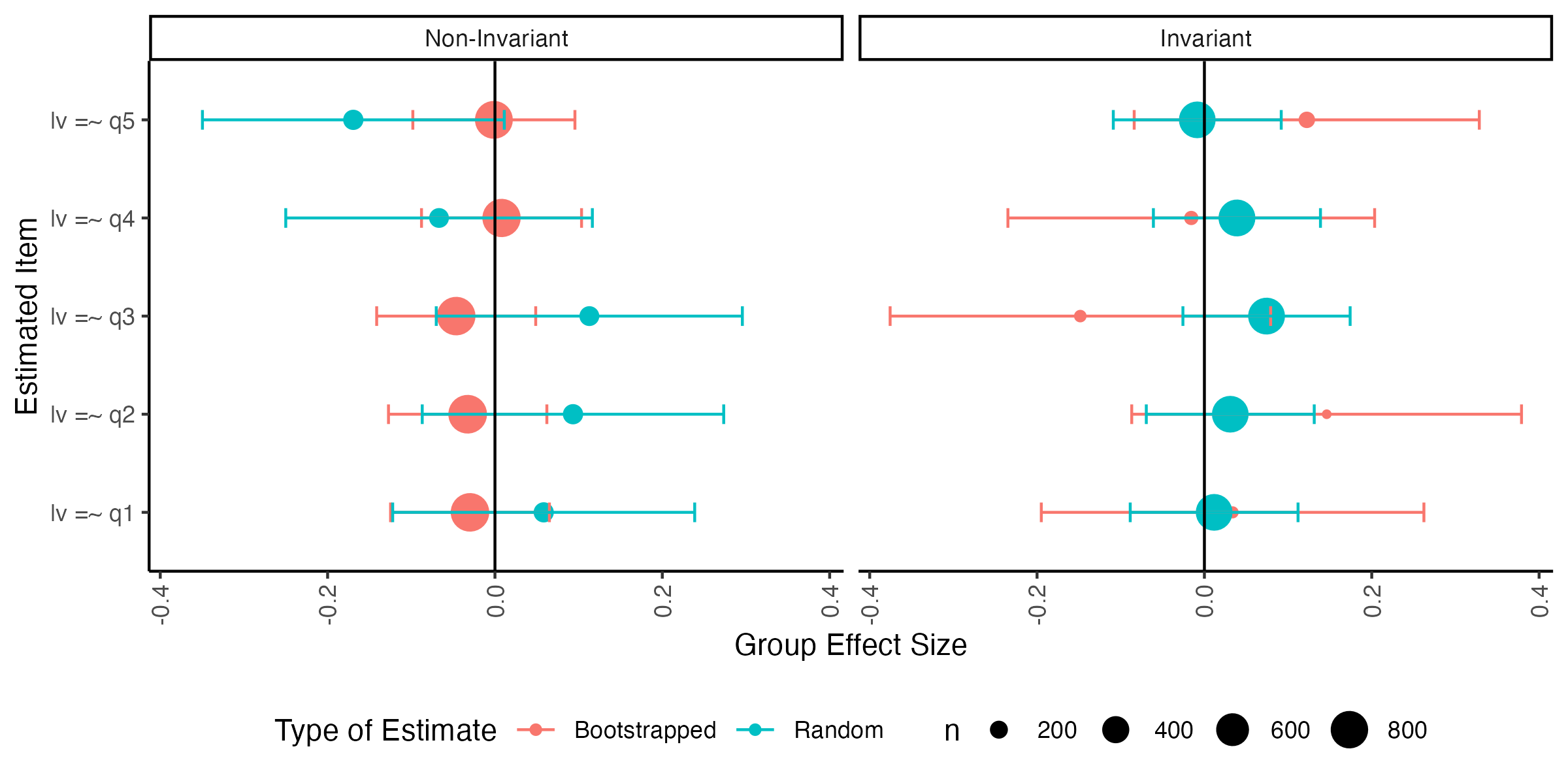
boot.partial.invariant$boot\_effects

## term non\_invariant random\_non\_invariant h\_nmi h\_mi h\_nmi\_p  
## 1 lv =~ q1 0.853 0.236 1.340078 -1.340078 0.4265601  
## 2 lv =~ q2 0.858 0.237 1.351946 -1.351946 0.4303378  
## 3 lv =~ q3 0.851 0.230 1.348639 -1.348639 0.4292851  
## 4 lv =~ q4 0.840 0.229 1.320578 -1.320578 0.4203530  
## 5 lv =~ q5 0.819 0.237 1.245789 -1.245789 0.3965468  
## h\_mi\_p  
## 1 -0.4265601  
## 2 -0.4303378  
## 3 -0.4292851  
## 4 -0.4203530  
## 5 -0.3965468

Plots of the results from dataframes can be found within the bootstrap\_partial() function. Figure 12 shows the difference between parameters for groups in the bootstrapped and randomly assigned group runs. Figure 13 shows the density plot of the estimates for each group organized by bootstrapped and randomly assigned groups and the invariance decision for each bootstrapped run. Last, Figure 14 indicates the value between groups with an indication of the number of data points in each estimate (i.e., dot size). These visualizations should allow a researcher to understand the likelihood of replication for each parameter, as well as the potential size of the differences. Therefore, one could indicate a specific smallest effect size of interest, rather than a invariance cut-off rule of thumb when planning a replication or registered report.

*Figure 12.* Visualization of the difference score between groups by parameter for invariant and non-invariant bootstrapped and randomly assigned group data.

*Figure 13*. Visualization of the number of estimates for each group by bootstrapped and randomly assigned group runs by their invariance decision.

*Figure 14*. Visualization of effect size between groups by parameter for invariant and non-invariant bootstrapped and randomly assigned group data. The size of the dots indicate the number of data points for that estimate.

**References**

Epskamp, S. (2022). *semPlot: Path diagrams and visual analysis of various SEM packages’ output*. <https://CRAN.R-project.org/package=semPlot>

Robinson, D., Hayes, A., Couch [aut, S., cre, Software, P., PBC, Patil, I., Chiu, D., Gomez, M., Demeshev, B., Menne, D., Nutter, B., Johnston, L., Bolker, B., Briatte, F., Arnold, J., Gabry, J., Selzer, L., Simpson, G., … Reinhart, A. (2023). *Broom: Convert statistical objects into tidy tibbles*. <https://CRAN.R-project.org/package=broom>

Wickham, H. (2016). *ggplot2: Elegant graphics for data analysis*. Springer-Verlag New York. <https://ggplot2.tidyverse.org>

Wilke, C. O. (2020). *Cowplot: Streamlined plot theme and plot annotations for ’ggplot2’*. <https://CRAN.R-project.org/package=cowplot>

# Simulation Study

The code for building and running simulations can be found in *simulate\_boot\_rr.Rmd*, *simulate\_boot\_partial.Rmd*, and *simulate\_combine.Rmd*.

## Design and Analysis

Data was simulated using the simulateData function in the *R* package *lavaan* (Rosseel, 2012) assuming multivariate normality using a of 0 and of 1 for the data. This function allows you to write *lavaan* syntax for your model with estimated values to generate data for observed variables (see supplemental for examples). The data included two groups of individuals (“Group 1”, “Group 2”) for a multigroup confirmatory factor analysis ( = 250, *N* = 500). The latent variables were assumed to be continuous normal (the package functions do not require this assumption). The model consisted of five observed items predicted by one latent variable (lv =~ q1 + q2 + q3 + q4 + q5); however, the demonstration in this manuscript extends to multiple latent variables and other combinations of observed variables. Each item was assumed to be related to the latent variable with loadings approximately equal to .40 to .80, except when cases of non-invariance on the loadings was simulated.

The Brown (2015) steps of testing measurement invariance are demonstrated in this manuscript for illustration purposes, but in line with Stark et al. (2006) suggestions, the visualizations show the impact of loadings and intercepts together. A convenience function mgcfa is used for these steps or other measurement invariance test orders and combinations. Fit indices for the steps for multigroup models are presented in the appendix for comparison of cutoff rules of thumb (Cheung & Rensvold, 2002) to effect sizes and visualizations presented in this manuscript. Fit indices include Akaike Information Criterion (AIC, Akaike, 1998), Bayesian Information Criterion (BIC, Schwarz, 1978), Comparative Fit Index (CFI, Bentler, 1990), Tucker Lewis Index (TLI, Tucker & Lewis, 1973), root mean squared error of approximation RMSEA (Steiger, 1990), and standardized root mean square residual (SRMR, Bentler, 1995).

The data was then simulated to represent invariance across all model steps, small, medium, and large invariance using estimated sizes from Nye et al. (2019). While is used primarily for an effect size of the (non)-invariance for intercepts and loadings together, a similar approach was taken for the estimation of small, medium, and large effects on the residuals. The effect size is presented for all models, calculated from the *dmacs* package (Dueber, 2023; Nye & Drasgow, 2011). Only one item in each model was manipulated from the invariant model to create the non-invariant models. Given the data was simulated with a *z*-score scaling, the loading values were simulated at .30 points apart (given suggestions of .2, .4, .7), the intercepts at .25 points apart, and the residuals at .25 points apart. To plan a simulation for your own study, these values can be used to simulate small, medium, and large non-invariance effects by first converting data into *z*-score.

## Visualize Parameter Differences

The value for item 4 in the invariant model was 0.16, representing a nil or unimportant difference in this manuscript. It is important to note that while Nye et al. (2019) suggests specific sizes for small, medium, and large, each researcher should determine for themselves what effects represent. Figure 15 displays the results from the small ( = 0.27) difference in loadings, while Figure 16 displays the results from the medium ( = 0.61) difference in loadings, and Figure 17 shows the large ( = 0.66) differences. When investigating the slope values, we can clearly see the change in the loading for the second group (the only manipulated variable, although random data set generation may also change intercepts and residuals slightly). At the medium effect size, we see that the confidence bands do not overlap (at the edges), and at the large effect size, we can see a clear separation of two lines. Note that the intercepts in this model are estimated as equal so the loading representation will not literally separate, but the steepness of the lines is the indicator of the difference between the slopes. You can imagine these lines are interpreted like a simple slopes analysis for interactions in regression (Cohen et al., 2003). When simple slopes for interactions are plotted, if they are parallel, there is no interaction, and if they cross, then there is an interaction. Here, we can use this same logic. If they are parallel, there is likely invariance (they are the same), and the further from parallel they become, the larger the effect size for the differences between group loadings.

The latent means in Figure 17 do appear to show differences, albeit visually small. The latent means diagram shows the impact of any group differences that aren’t constrained, and this image shows the configural model (as the metric model would force them to be equal). In the simulated model, the *only* manipulated parameter is question 4’s loading. In real models, the differences may be larger due to other variation found in the parameter estimates. Therefore, once you discover items you believe would make a model “partially” invariant, you may wish to estimate that model and graph the item again using the partially invariant model to see only the effect of the non-invariant items. Additionally, consider that we set the scaling of the model to 0. The estimate for the lv mean in the large loading model was group 1: 0.00, and group 2: -0.06, which results in 0.06 difference in group means. The practical implications of this difference will depend on the research and interpretations of the researcher.

A diagram of a normal distribution

Description automatically generated with medium confidence

*Figure 15*. Small Loadings Model Visualization

*A diagram of different sizes and colors

Description automatically generated with medium confidenceFigure 16*. Medium Loadings Model Visualization

A diagram of a normal distribution

Description automatically generated with medium confidence *Figure 17*. Large Loadings Model Visualization

For intercepts, the small (Figure 18), medium (Figure 19), and large (Figure 20) depictions represent values of 0.26, 0.47, and 0.70, respectively. Intercept differences can be clearly seen represented by the spacing out of the intercept locations (and thus, the overall line as well). While the changes in intercept do not appear to change the latent means, the caveat to this simulation is that only item four was manipulated. An example is provided below that demonstrates large changes in latent means.

A diagram of a normal distribution

Description automatically generated*Figure 18*. Small Intercepts Model Visualization

A diagram of a normal distribution

Description automatically generated *Figure 19*. Medium Intercepts Model Visualization

A diagram of a normal distribution

Description automatically generated *Figure 20*. Large Intercepts Model Visualization

Last, the effect of the residuals is plotted in small (Figure 21), medium (Figure 22), and large (Figure 23) formats. While values are not technically available for the residuals, our models showed 0.19, 0.10, and 0.16, respectively. These differences in values are variable due to the random generation of data sets for each measurement invariance manipulation. At first glance, the differences in the small chart may seem large, because the black lines are not touching, but notice that the distributions overlap, indicating a likely small difference. The medium and large differences better illustrate differences in residuals across groups. Further, the impact of the residuals on the shape of the latent mean distribution can also been seen (and unintentionally, in the first figures as well due to random variation). The impact is due to the standard error of the residuals, as smaller standard errors represent lepokurtic distributions (taller), and larger standard errors represent platykurtic distributions (flatter). The effect size difference of the residuals does not appear to change the effects in the latent means.

*A diagram of a function

Description automatically generated with medium confidenceFigure 21*. Small Residuals Model Visualization

*A diagram of a normal distribution

Description automatically generated with medium confidence**Figure 22*. Medium Residuals Model Visualization

*A diagram of a variety of groups

Description automatically generated with medium confidence Figure 23*. Large Residuals Model Visualization

## Model Replication and Effect Size

Figure 24 portrays the values by simulated non-invariance, strength of non-invariance, and type of equality constraint. This image represents 100 simulations of data by 1000 bootstrapped runs (averaged) to explore the expected pattern of results. The bars are arranged to show what a researcher might inspect when thinking about replication possibilities and their effect sizes (i.e., only three bars for each equality constraint would be calculated).

A graph of different colored squares

Description automatically generated with medium confidence*Figure 24*. Visualization of the effect size of bootstrapped replication proportions on simulated data. Each panel indicates the simulated data type, colors represent the differences in the strength of the non-invariance, and the bars on the x-axis represent the effect size for the equality constraint.

In the data that was simulated to be invariant between groups, effect sizes are still non-zero (loadings = 0.28, intercepts = = 0.06, = 0.00). This result mirrors the effects found in the literature - that often, many models fail to show invariance, and potentially not because measurement is poor but because of natural random variation in parameter estimates. This result also indicates the need to be able to identify if specific parameters are driving the differences, which is shown in the next section.

Next, Figure 24 demonstrates the patterns one might find for small, medium, and large effects at each type of invariance when data is simulated with *one* difference. For loadings, the pattern shows a larger effect for loadings with zero or negative effect sizes for other effect sizes. The intercept simulations show non-zero effect sizes in the loadings and intercepts, likely for the same reasons is interpreted as a combined effect size. When intercepts are changed, loadings may naturally shift with those means. Last, the residual results present an unexpected pattern, wherein the effect is primarily seen in the loadings, rather than the residuals step. However, when distributions of error variance are different, one may expect that those effects are pushed toward the loadings as well (as values can vary more, thus potentially weakening the relationship between observed and latent variable).

An example of interpretation on real data is given in a later section. From a research study, only one effect size for each equality constraint would be calculated. The interpretation will often be up to the researcher’s smallest effect of interest, and this simulation gives some guidance that the values should not be interpreted with traditional rules of thumb. The pattern of effects is potentially the most useful information: 1) positive effects on the loadings with negative or very close to zero effects on the other parameters may indicate a non-replication in loadings, 2) equal effects on loadings and intercepts with smaller or negative effects may indicate intercepts may be an issue, and 3) residuals may be determined by the same pattern as loadings but with a smaller ratio of loadings to residuals effect (i.e., loadings / residuals . The “size” could be determined by the ratio of effect sizes for each constraint. Of course, this represents one simulation study, and results from many studies in a meta-analysis would be fruitful for future work.

## Parameter Replication and Effect Size

Figure 25 shows the effect size differences within large loadings simulations. The results demonstrate that most of the loadings were considered non-invariant in the bootstrapped models (while holdings all others equal). This result is partially due to simulating very good data, so small changes in loadings results in a drop in fit for our chosen invariance index. However, we can use this graph to show that question four shows a possible effect size ranging from -0.07 to 0.13. The value for question four was 0.27, representing about a quarter of a possible total effect. Last, the density plot in Figure 26 shows the separation of the two different groups loadings in item four, thus, illustrating group differences in the findings for their loadings. Each of the other combination of plots can be found in the supplemental materials.

A diagram of a group

Description automatically generated with medium confidence

*A diagram of a group

Description automatically generated with medium confidenceFigure 25*. Bootstrapped and Random Group effect size differences in loadings for the Large Loading difference simulation. The size of the point reprensents the number of data points included in that calculation.

*A graph of different colored lines

Description automatically generated with medium confidenceFigure 26*. Bootstrapped and Random density plots for invariant and non-invariant bootstrapped partial effects examining only large loadings.

## MGCFA Model Fit Statistics

Model fit statistics are provided for each of the ten model combinations (invariant, three sizes for each loadings, intercepts, and residuals). These tables could be used to examine the traditional change in fit statistics cutoff rules of thumb (Cheung & Rensvold, 2002), such as CFI or RMSEA, to the visualizations presented in the manuscript.

Table 3. *Model Fit for Invariant Model*

| Model | AIC | BIC | CFI | TLI | RMSEA | SRMR |
| --- | --- | --- | --- | --- | --- | --- |
| Overall | 7,516.454 | 7,579.673 | 1.000 | 1.036 | 0.000 | 0.006 |
| Group Group 1 | 3,765.749 | 3,818.571 | 0.976 | 0.953 | 0.047 | 0.031 |
| Group Group 2 | 3,767.599 | 3,820.421 | 1.000 | 1.008 | 0.000 | 0.021 |
| Configural | 7,533.348 | 7,659.786 | 0.991 | 0.982 | 0.030 | 0.026 |
| Loadings | 7,528.476 | 7,638.056 | 0.994 | 0.992 | 0.020 | 0.033 |
| Intercepts | 7,522.397 | 7,615.118 | 1.000 | 1.003 | 0.000 | 0.035 |
| Residuals | 7,520.435 | 7,592.083 | 0.991 | 0.992 | 0.020 | 0.046 |

Table 4. *Model Fit for Small Differences in Loadings*

| Model | AIC | BIC | CFI | TLI | RMSEA | SRMR |
| --- | --- | --- | --- | --- | --- | --- |
| Overall | 7,530.321 | 7,593.540 | 0.977 | 0.955 | 0.049 | 0.025 |
| Group Group 1 | 3,765.749 | 3,818.571 | 0.976 | 0.953 | 0.047 | 0.031 |
| Group Group 2 | 3,785.242 | 3,838.064 | 0.979 | 0.958 | 0.050 | 0.029 |
| Configural | 7,550.991 | 7,677.430 | 0.978 | 0.956 | 0.048 | 0.030 |
| Loadings | 7,550.133 | 7,659.713 | 0.966 | 0.952 | 0.051 | 0.047 |
| Intercepts | 7,542.675 | 7,635.397 | 0.979 | 0.977 | 0.035 | 0.047 |
| Residuals | 7,534.091 | 7,605.739 | 0.993 | 0.994 | 0.019 | 0.054 |

Table 5. *Model Fit for Medium Differences in Loadings*

| Model | AIC | BIC | CFI | TLI | RMSEA | SRMR |
| --- | --- | --- | --- | --- | --- | --- |
| Overall | 7,595.822 | 7,659.041 | 1.000 | 1.017 | 0.000 | 0.012 |
| Group Group 1 | 3,765.749 | 3,818.571 | 0.976 | 0.953 | 0.047 | 0.031 |
| Group Group 2 | 3,815.314 | 3,868.136 | 0.996 | 0.991 | 0.025 | 0.023 |
| Configural | 7,581.063 | 7,707.501 | 0.988 | 0.976 | 0.038 | 0.027 |
| Loadings | 7,609.348 | 7,718.928 | 0.878 | 0.826 | 0.101 | 0.079 |
| Intercepts | 7,601.550 | 7,694.271 | 0.891 | 0.879 | 0.084 | 0.079 |
| Residuals | 7,596.811 | 7,668.459 | 0.890 | 0.905 | 0.075 | 0.096 |

Table 6. *Model Fit for Large Differences in Loadings*

| Model | AIC | BIC | CFI | TLI | RMSEA | SRMR |
| --- | --- | --- | --- | --- | --- | --- |
| Overall | 7,665.760 | 7,728.979 | 1.000 | 1.022 | 0.000 | 0.010 |
| Group Group 1 | 3,765.749 | 3,818.571 | 0.976 | 0.953 | 0.047 | 0.031 |
| Group Group 2 | 3,855.953 | 3,908.775 | 0.997 | 0.995 | 0.020 | 0.023 |
| Configural | 7,621.702 | 7,748.140 | 0.989 | 0.978 | 0.036 | 0.027 |
| Loadings | 7,659.690 | 7,769.270 | 0.852 | 0.789 | 0.114 | 0.088 |
| Intercepts | 7,652.360 | 7,745.081 | 0.863 | 0.848 | 0.097 | 0.088 |
| Residuals | 7,664.853 | 7,736.502 | 0.806 | 0.832 | 0.102 | 0.135 |

Table 7. *Model Fit for Small Differences in Intercepts*

| Model | AIC | BIC | CFI | TLI | RMSEA | SRMR |
| --- | --- | --- | --- | --- | --- | --- |
| Overall | 7,520.471 | 7,583.690 | 1.000 | 1.035 | 0.000 | 0.007 |
| Group Group 1 | 3,765.749 | 3,818.571 | 0.976 | 0.953 | 0.047 | 0.031 |
| Group Group 2 | 3,767.599 | 3,820.421 | 1.000 | 1.008 | 0.000 | 0.021 |
| Configural | 7,533.348 | 7,659.786 | 0.991 | 0.982 | 0.030 | 0.026 |
| Loadings | 7,528.476 | 7,638.056 | 0.994 | 0.992 | 0.020 | 0.033 |
| Intercepts | 7,526.312 | 7,619.034 | 0.987 | 0.986 | 0.027 | 0.040 |
| Residuals | 7,524.356 | 7,596.005 | 0.975 | 0.978 | 0.033 | 0.050 |

Table 8. *Model Fit for Medium Differences in Intercepts*

| Model | AIC | BIC | CFI | TLI | RMSEA | SRMR |
| --- | --- | --- | --- | --- | --- | --- |
| Overall | 7,538.375 | 7,601.594 | 1.000 | 1.033 | 0.000 | 0.008 |
| Group Group 1 | 3,765.749 | 3,818.571 | 0.976 | 0.953 | 0.047 | 0.031 |
| Group Group 2 | 3,767.599 | 3,820.421 | 1.000 | 1.008 | 0.000 | 0.021 |
| Configural | 7,533.348 | 7,659.786 | 0.991 | 0.982 | 0.030 | 0.026 |
| Loadings | 7,528.476 | 7,638.056 | 0.994 | 0.992 | 0.020 | 0.033 |
| Intercepts | 7,544.002 | 7,636.724 | 0.917 | 0.907 | 0.068 | 0.059 |
| Residuals | 7,542.064 | 7,613.712 | 0.905 | 0.917 | 0.065 | 0.067 |

Table 9. *Model Fit for Large Differences in Intercepts*

| Model | AIC | BIC | CFI | TLI | RMSEA | SRMR |
| --- | --- | --- | --- | --- | --- | --- |
| Overall | 7,568.748 | 7,631.967 | 1.000 | 1.032 | 0.000 | 0.008 |
| Group Group 1 | 3,765.749 | 3,818.571 | 0.976 | 0.953 | 0.047 | 0.031 |
| Group Group 2 | 3,767.599 | 3,820.421 | 1.000 | 1.008 | 0.000 | 0.021 |
| Configural | 7,533.348 | 7,659.786 | 0.991 | 0.982 | 0.030 | 0.026 |
| Loadings | 7,528.476 | 7,638.056 | 0.994 | 0.992 | 0.020 | 0.033 |
| Intercepts | 7,574.054 | 7,666.776 | 0.797 | 0.775 | 0.106 | 0.084 |
| Residuals | 7,572.174 | 7,643.823 | 0.785 | 0.813 | 0.097 | 0.090 |

Table 10. *Model Fit for Small Differences in Residuals*

| Model | AIC | BIC | CFI | TLI | RMSEA | SRMR |
| --- | --- | --- | --- | --- | --- | --- |
| Overall | 7,462.007 | 7,525.226 | 1.000 | 1.020 | 0.000 | 0.013 |
| Group Group 1 | 3,765.749 | 3,818.571 | 0.976 | 0.953 | 0.047 | 0.031 |
| Group Group 2 | 3,703.797 | 3,756.619 | 0.962 | 0.924 | 0.061 | 0.037 |
| Configural | 7,469.546 | 7,595.984 | 0.969 | 0.938 | 0.054 | 0.034 |
| Loadings | 7,471.637 | 7,581.217 | 0.944 | 0.920 | 0.062 | 0.049 |
| Intercepts | 7,465.722 | 7,558.443 | 0.952 | 0.946 | 0.051 | 0.051 |
| Residuals | 7,465.986 | 7,537.635 | 0.930 | 0.939 | 0.054 | 0.065 |

Table 11. *Model Fit for Medium Differences in Residuals*

| Model | AIC | BIC | CFI | TLI | RMSEA | SRMR |
| --- | --- | --- | --- | --- | --- | --- |
| Overall | 7,382.876 | 7,446.095 | 0.992 | 0.985 | 0.028 | 0.020 |
| Group Group 1 | 3,765.749 | 3,818.571 | 0.976 | 0.953 | 0.047 | 0.031 |
| Group Group 2 | 3,601.920 | 3,654.742 | 1.000 | 1.023 | 0.000 | 0.018 |
| Configural | 7,367.669 | 7,494.108 | 0.996 | 0.992 | 0.020 | 0.025 |
| Loadings | 7,371.027 | 7,480.607 | 0.969 | 0.956 | 0.049 | 0.046 |
| Intercepts | 7,364.724 | 7,457.446 | 0.977 | 0.975 | 0.037 | 0.047 |
| Residuals | 7,386.854 | 7,458.502 | 0.877 | 0.893 | 0.076 | 0.076 |

Table 12. *Model Fit for Large Differences in Residuals*

| Model | AIC | BIC | CFI | TLI | RMSEA | SRMR |
| --- | --- | --- | --- | --- | --- | --- |
| Overall | 7,301.897 | 7,365.116 | 0.993 | 0.986 | 0.026 | 0.019 |
| Group Group 1 | 3,765.749 | 3,818.571 | 0.976 | 0.953 | 0.047 | 0.031 |
| Group Group 2 | 3,454.309 | 3,507.131 | 0.945 | 0.889 | 0.075 | 0.036 |
| Configural | 7,220.058 | 7,346.496 | 0.960 | 0.920 | 0.063 | 0.034 |
| Loadings | 7,218.878 | 7,328.458 | 0.948 | 0.926 | 0.060 | 0.046 |
| Intercepts | 7,213.011 | 7,305.732 | 0.956 | 0.951 | 0.049 | 0.048 |
| Residuals | 7,305.660 | 7,377.309 | 0.559 | 0.617 | 0.137 | 0.189 |

# 

**References**

Akaike, H. (1998). *Information theory and an extension of the maximum likelihood principle* (E. Parzen, K. Tanabe, & G. Kitagawa, Eds.; pp. 199–213). Springer New York. <http://link.springer.com/10.1007/978-1-4612-1694-0_15>

Bentler, P. M. (1990). Comparative fit indexes in structural models. *Psychological Bulletin*, *107*(2), 238–246. <https://doi.org/10.1037/0033-2909.107.2.238>

Bentler, P. M. (1995). *EQS structural equations program manual*.

Brown, T. A. (2015). *Confirmatory factor analysis for applied research* (Second edition). The Guilford Press.

Cheung, G. W., & Rensvold, R. B. (2002). Evaluating Goodness-of-Fit Indexes for Testing Measurement Invariance. *Structural Equation Modeling: A Multidisciplinary Journal*, *9*(2), 233–255. <https://doi.org/10.1207/s15328007sem0902_5>

Cohen, J., Cohen, P., West, S. G., & Aiken, L. (2003). *Applied multiple regression / correlation analysis for the behavioral sciences* (3rd ed.). Lawrence Erlbaum Associates.

Dueber, D. (2023). *Dmacs*. <https://github.com/ddueber/dmacs>

Nye, C. D., Bradburn, J., Olenick, J., Bialko, C., & Drasgow, F. (2019). How Big Are My Effects? Examining the Magnitude of Effect Sizes in Studies of Measurement Equivalence. *Organizational Research Methods*, *22*(3), 678–709. <https://doi.org/10.1177/1094428118761122>

Nye, C. D., & Drasgow, F. (2011). Effect size indices for analyses of measurement equivalence: Understanding the practical importance of differences between groups. *Journal of Applied Psychology*, *96*(5), 966–980. <https://doi.org/10.1037/a0022955>

Rosseel, Y. (2012). Lavaan: An r package for structural equation modeling. *Journal of Statistical Software*, *48*(1), 1–36. <https://doi.org/10.18637/jss.v048.i02>

Schwarz, G. (1978). Estimating the dimension of a model. *The Annals of Statistics*, *6*(2), 461–464. <https://www.jstor.org/stable/2958889>

Stark, S., Chernyshenko, O. S., & Drasgow, F. (2006). Detecting differential item functioning with confirmatory factor analysis and item response theory: Toward a unified strategy. *Journal of Applied Psychology*, *91*(6), 1292–1306. <https://doi.org/10.1037/0021-9010.91.6.1292>

Steiger, J. H. (1990). Structural model evaluation and modification: An interval estimation approach. *Multivariate Behavioral Research*, *25*(2), 173–180. <https://doi.org/10.1207/s15327906mbr2502_4>

Tucker, L. R., & Lewis, C. (1973). A reliability coefficient for maximum likelihood factor analysis. *Psychometrika*, *38*(1), 1–10. <https://doi.org/10.1007/BF02291170>

# Simulated Partial Invariance Results

Table 13. *Fit Estimates for Partial Invariance Residuals on Invariant Data*

| Estimated Parameter | CFI | RSMEA |
| --- | --- | --- |
| q1 q1 | 0.990 | 0.021 |
| q2 q2 | 0.987 | 0.024 |
| q3 q3 | 0.996 | 0.014 |
| q4 q4 | 1.000 | 0.000 |
| q5 q5 | 0.987 | 0.025 |
| lv lv | 0.991 | 0.020 |

Table 14. *Fit Estimates for Partial Invariance Loadings for Small Loading Data*

| Estimated Parameter | CFI | RSMEA |
| --- | --- | --- |
| lv = q1 | 0.993 | 0.019 |
| lv = q2 | 0.989 | 0.023 |
| lv = q3 | 0.989 | 0.023 |
| lv = q4 | 1.000 | 0.000 |
| lv = q5 | 0.994 | 0.017 |

Table 15. *Fit Estimates for Partial Invariance Loadings for Medium Loading Data*

| Estimated Parameter | CFI | RSMEA |
| --- | --- | --- |
| lv = q1 | 0.890 | 0.075 |
| lv = q2 | 0.904 | 0.072 |
| lv = q3 | 0.887 | 0.078 |
| lv = q4 | 1.000 | 0.000 |
| lv = q5 | 0.914 | 0.068 |

Table 16. *Fit Estimates for Partial Invariance Loadings for Large Loading Data*

| Estimated Parameter | CFI | RSMEA |
| --- | --- | --- |
| lv = q1 | 0.806 | 0.102 |
| lv = q2 | 0.812 | 0.102 |
| lv = q3 | 0.813 | 0.102 |
| lv = q4 | 1.000 | 0.000 |
| lv = q5 | 0.861 | 0.088 |

Table 17. *Fit Estimates for Partial Invariance Loadings for Small Intercept Data*

| Estimated Parameter | CFI | RSMEA |
| --- | --- | --- |
| q1 1 | 0.975 | 0.033 |
| lv 1 | 0.975 | 0.033 |
| q2 1 | 0.972 | 0.035 |
| q3 1 | 0.972 | 0.036 |
| q4 1 | 0.988 | 0.023 |
| q5 1 | 0.971 | 0.036 |

Table 18. *Fit Estimates for Partial Invariance Loadings for Medium Intercept Data*

| Estimated Parameter | CFI | RSMEA |
| --- | --- | --- |
| q1 1 | 0.905 | 0.065 |
| lv 1 | 0.905 | 0.065 |
| q2 1 | 0.901 | 0.067 |
| q3 1 | 0.901 | 0.067 |
| q4 1 | 0.988 | 0.023 |
| q5 1 | 0.902 | 0.067 |

Table 19. *Fit Estimates for Partial Invariance Loadings for Large Intercept Data*

| Estimated Parameter | CFI | RSMEA |
| --- | --- | --- |
| q1 1 | 0.785 | 0.097 |
| lv 1 | 0.785 | 0.097 |
| q2 1 | 0.781 | 0.100 |
| q3 1 | 0.781 | 0.100 |
| q4 1 | 0.988 | 0.023 |
| q5 1 | 0.784 | 0.099 |

Table 20. *Fit Estimates for Partial Invariance Loadings for Small Residual Data*

| Estimated Parameter | CFI | RSMEA |
| --- | --- | --- |
| q1 q1 | 0.928 | 0.056 |
| q2 q2 | 0.936 | 0.053 |
| q3 q3 | 0.926 | 0.057 |
| q4 q4 | 0.955 | 0.044 |
| q5 q5 | 0.926 | 0.057 |
| lv lv | 0.930 | 0.054 |

Table 21. *Fit Estimates for Partial Invariance Loadings for Medium Residual Data*

| Estimated Parameter | CFI | RSMEA |
| --- | --- | --- |
| q1 q1 | 0.879 | 0.077 |
| q2 q2 | 0.873 | 0.079 |
| q3 q3 | 0.878 | 0.077 |
| q4 q4 | 0.980 | 0.031 |
| q5 q5 | 0.873 | 0.079 |
| lv lv | 0.877 | 0.076 |

Table 22. *Fit Estimates for Partial Invariance Loadings for Large Residual Data*

| Estimated Parameter | CFI | RSMEA |
| --- | --- | --- |
| q1 q1 | 0.555 | 0.141 |
| q2 q2 | 0.556 | 0.141 |
| q3 q3 | 0.556 | 0.141 |
| q4 q4 | 0.967 | 0.039 |
| q5 q5 | 0.557 | 0.141 |
| lv lv | 0.559 | 0.137 |

# 

# Invariance Plots Difference Scores by Condition

A graph of a number of results

Description automatically generated with medium confidenceA graph with red and blue lines

Description automatically generatedA graph with red and blue lines

Description automatically generatedA graph with red and blue dots

Description automatically generatedA graph with red and blue dots

Description automatically generatedA graph with red and blue dots

Description automatically generatedA graph with red and blue lines

Description automatically generatedA graph with red and blue dots

Description automatically generatedA graph with red and blue dots

Description automatically generatedA graph with red and blue lines

Description automatically generated

# Invariance Plots Effect Sizes by Condition

A diagram of a group effect

Description automatically generated with medium confidenceA graph with red and blue dots

Description automatically generatedA graph with different colored lines

Description automatically generated with medium confidenceA graph with red and blue dots

Description automatically generatedA graph with red and blue dots

Description automatically generatedA graph with different colored dots

Description automatically generated with medium confidenceA graph with red and blue dots

Description automatically generatedA graph with red and blue dots

Description automatically generatedA graph with red and blue dots

Description automatically generatedA graph with red and blue dots

Description automatically generated

# Density Plots by Condition

A graph of different colored lines

Description automatically generated with medium confidenceA graph of different sizes and colors

Description automatically generated with medium confidenceA graph of different sizes and colors

Description automatically generated with medium confidenceA graph of different sizes and colors

Description automatically generated with medium confidenceA graph of different sizes and colors

Description automatically generated with medium confidenceA graph of different sizes and colors

Description automatically generated with medium confidenceA graph showing a random and random

Description automatically generatedA graph of different sizes and colors

Description automatically generated with medium confidenceA graph of different sizes and colors

Description automatically generated with medium confidenceA graph of a random and random

Description automatically generated

## Replication Test

set.seed(890343)  
library(rio)  
library(dplyr)  
library(visualizemi)  
library(papaja)  
library(tidyr)  
library(dmacs)  
library(ggplot2)  
library(scales)  
library(introdataviz)  
library(lavaan)

## Data

# our data  
load("manu\_data/RS14.Rdata")  
  
DF <- DF  
  
# take only interesting columns and give a label  
DF <- DF %>%   
 # make it comparable by only testing students  
 filter(DF$sample == "Student") %>%   
 mutate(group = "Aiena") %>%   
 dplyr::select(RS1:RS14, group)   
  
# Chen et al.  
DF.pone <- import("manu\_data/data1-RS14.dta")  
# take only interesting columns and give a label  
DF.pone <- DF.pone %>%   
 dplyr::select(starts\_with("rs")) %>%   
 mutate(group = "Chen")  
# match column style  
colnames(DF.pone)[1:14] <- toupper(colnames(DF.pone)[1:14])   
  
# create one dataset  
DF.combo <- bind\_rows(DF, DF.pone)  
  
DF.combo %>%   
 group\_by(group) %>%   
 summarize(sample = n())

## # A tibble: 2 × 2  
## group sample  
## <chr> <int>  
## 1 Aiena 1765  
## 2 Chen 1010

## MGCFA

# build the one-factor model   
model.rs <- "RS =~ RS1+RS2+RS3+RS4+RS5+RS6+RS7+RS8+RS9+RS10+RS11+RS12+RS13+RS14"  
  
# run the multi-group CFA  
results.rs <- mgcfa(  
 model = model.rs,  
 data = DF.combo,   
 group = "group",   
 group.equal = c("loadings", "intercepts", "residuals")  
 )  
  
# examine the results  
results.rs$model\_fit %>%   
 dplyr::select(model, AIC, BIC, cfi, tli, rmsea, srmr)

## # A tibble: 7 × 7  
## model AIC BIC cfi tli rmsea srmr  
## <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 Overall 113442. 113608. 0.939 0.928 0.0890 0.0344  
## 2 Group Aiena 69101. 69254. 0.928 0.915 0.108 0.0353  
## 3 Group Chen 42603. 42741. 0.929 0.916 0.0766 0.0399  
## 4 Configural 111760. 112258. 0.929 0.916 0.0975 0.0347  
## 5 loadings 111789. 112210. 0.927 0.921 0.0946 0.0427  
## 6 intercepts 112785. 113129. 0.892 0.891 0.111 0.0599  
## 7 residuals 113318. 113579. 0.873 0.880 0.116 0.0661

Overall, the one-factor model fits the data well. Each group also shows adequate model fit. If we use CFI <= .01, we find that the loadings would be considered invariant across the English and Chinese samples. The intercepts were not invariant.

## Partial Invariance

partial.rs <-  
 partial\_mi(  
 saved\_model = results.rs$invariance\_models$model.intercepts,  
 data = DF.combo,  
 model = model.rs,  
 group = "group",  
 # be sure to do only up to the step you are interested in  
 group.equal = c("loadings", "intercepts"),  
 partial\_step = "intercepts")  
  
partial.rs$fit\_table %>%   
 dplyr::select(free.parameter, cfi) %>%   
 arrange(-cfi)

## # A tibble: 15 × 2  
## free.parameter cfi   
## <chr> <lvn.vctr>  
## 1 "RS2 ~1 " 0.8970997   
## 2 "RS11 ~1 " 0.8963948   
## 3 "RS4 ~1 " 0.8961938   
## 4 "RS12 ~1 " 0.8960095   
## 5 "RS14 ~1 " 0.8954496   
## 6 "RS1 ~1 " 0.8954215   
## 7 "RS10 ~1 " 0.8953395   
## 8 "RS3 ~1 " 0.8948455   
## 9 "RS5 ~1 " 0.8945236   
## 10 "RS7 ~1 " 0.8941450   
## 11 "RS8 ~1 " 0.8920022   
## 12 "RS13 ~1 " 0.8919491   
## 13 "RS9 ~1 " 0.8919293   
## 14 "RS ~1 " 0.8917757   
## 15 "RS6 ~1 " 0.8917604

Examining partial invariance reveals several potential candidates for partial invariance. In this next section, we relaxed group constraints until we achieved partial invariance (i.e., CFI <= .01). We will need to find our CFI as at least 0.92. More than half the items are necessary to achieve “partial” invariance (which really implies no invariance is likely possible).

# run the partially invariant model with group.partial  
partial.rs.1 <- mgcfa(model = model.rs,   
 data = DF.combo,   
 group = "group",   
 group.equal = c("loadings", "intercepts"),  
 group.partial = c("RS2 ~1", "RS11 ~1",  
 "RS4 ~1", "RS12 ~1",  
 "RS14 ~1", "RS1 ~1",  
 "RS10 ~1", "RS3 ~ 1"),  
 meanstructure = TRUE)  
  
# examine the fit indices   
partial.rs.1$model\_fit %>%   
 filter(model == "intercepts") %>%   
 dplyr::select(AIC, BIC, cfi, tli, rmsea, srmr)

## # A tibble: 1 × 6  
## AIC BIC cfi tli rmsea srmr  
## <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 111935. 112326. 0.922 0.917 0.0966 0.0462

# examine the intercepts   
partial.rs.1$model\_coef %>%   
 filter(model == "intercepts") %>%   
 dplyr::select(term, group, estimate) %>%   
 pivot\_wider(id\_cols = c("term"),   
 names\_from = "group",   
 values\_from = "estimate") %>%   
 dplyr::rename(English = `1`,   
 Chinese = `2`) %>%   
 filter(term %in% c("RS1 ~1 ", "RS12 ~1 ",  
 "RS7 ~1 ", "RS4 ~1 ",  
 "RS5 ~1 ", "RS3 ~1 "))

## # A tibble: 6 × 3  
## term English Chinese  
## <chr> <dbl> <dbl>  
## 1 "RS1 ~1 " 5.24 4.78  
## 2 "RS3 ~1 " 5.14 5.46  
## 3 "RS4 ~1 " 5.29 5.71  
## 4 "RS5 ~1 " 5.13 5.13  
## 5 "RS7 ~1 " 5.26 5.26  
## 6 "RS12 ~1 " 5.57 5.16

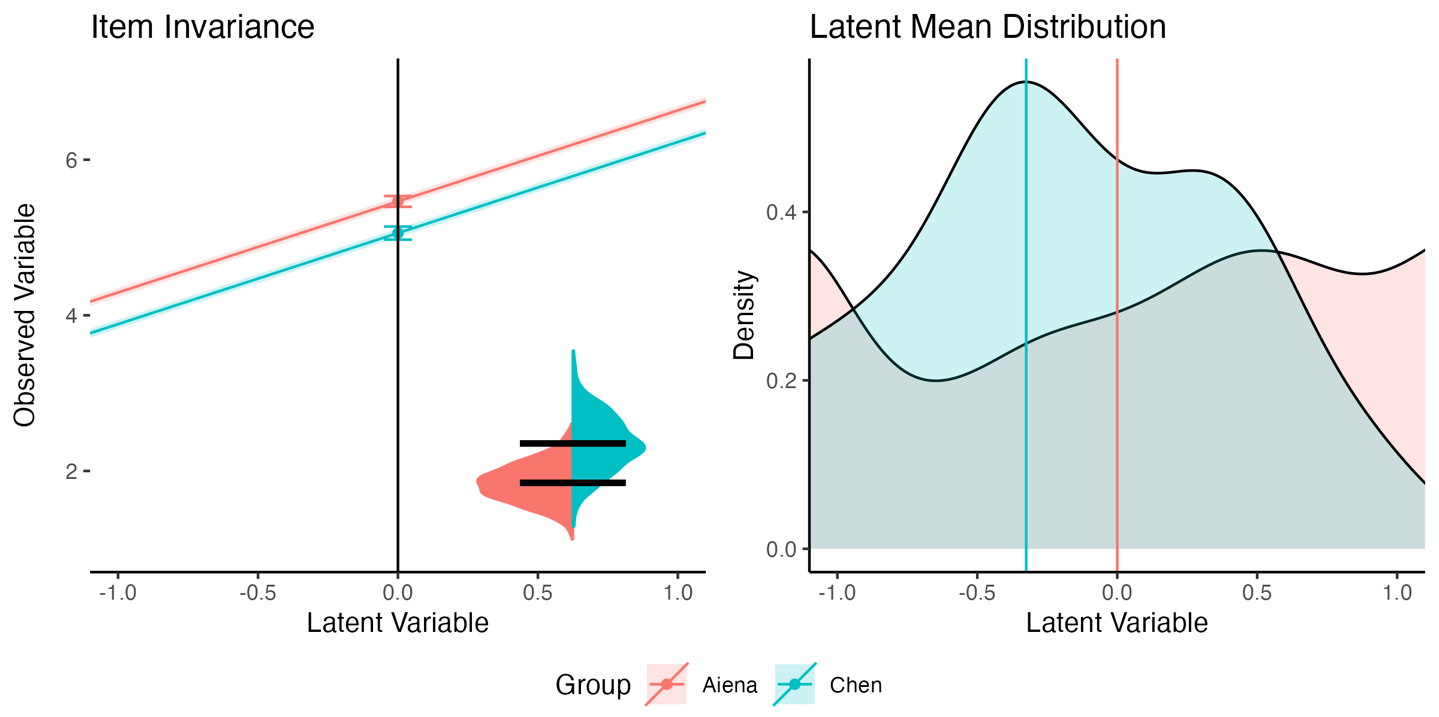
# effect size model   
round(lavaan\_dmacs(partial.rs.1$invariance\_models$model.intercepts, "Chen")$DMACS, 2)

## RS1 RS2 RS3 RS4 RS5 RS6 RS7 RS8 RS9 RS10 RS11 RS12 RS13 RS14   
## 0.32 0.29 0.23 0.30 0.00 0.00 0.00 0.00 0.00 0.29 0.27 0.29 0.00 0.26

## Visualize Invariance

for (i in c("RS2", "RS11",  
 "RS4", "RS12",  
 "RS14", "RS1",  
 "RS10", "RS3")){  
 cat(paste0("Plot for ", i, "\n"))  
 print(plot\_mi(  
 # output from model\_coef  
 data\_coef = partial.rs.1$model\_coef,   
 # which model do you want to plot  
 model\_step = "intercepts",   
 # name of observed item  
 item\_name = i,   
 # latent variable limits to graph  
 x\_limits = c(-1,1),   
 # Y min and max in data   
 y\_limits = c(min(DF.combo[ , i]), max(DF.combo[ , i])),  
 # what ci do you want  
 conf.level = .95,   
 # what model results do you want   
 model\_results = partial.rs.1$invariance\_models$model.intercepts,  
 # which latent variable do you want   
 lv\_name = "RS"   
)$complete)  
   
 ggsave(paste0("figures/rs14-chinese-", i, ".png"),   
 dpi = 300,   
 width = 8,   
 units = "in")  
   
 cat(paste0("-------\n"))  
   
}

## Plot for RS2



## -------  
## Plot for RS11

A diagram of different types of graphs

Description automatically generated with medium confidence

## -------  
## Plot for RS4

A diagram of different types of graphs

Description automatically generated with medium confidence

## -------  
## Plot for RS12

A diagram of different types of graphs

Description automatically generated with medium confidence

## -------  
## Plot for RS14

A diagram of a variety of graphs

Description automatically generated with medium confidence

## -------  
## Plot for RS1

A diagram of a variety of graphs

Description automatically generated with medium confidence

## -------  
## Plot for RS10

A diagram of different types of graphs

Description automatically generated with medium confidence

## -------  
## Plot for RS3

A diagram of different types of graphs

Description automatically generated with medium confidence

## -------

## Bootstrap Model

boot.model.rs <-   
 bootstrap\_model(  
 saved\_configural = results.rs$model\_configural,  
 data = DF.combo,  
 model = model.rs,   
 group = "group",   
 nboot = 1000,   
 invariance\_index = "cfi",  
 invariance\_rule = .01,   
 group.equal = c("loadings", "intercepts")  
)  
  
# export(boot.model.rs, "manu\_data/boot.model.rs.appendix7.csv")  
  
boot.model.rs

## model non\_invariant random\_non\_invariant h\_nmi h\_nmi\_p  
## 1 intercepts 1 0 3.141593 1

In this case, we do not see loadings print out. That implies that all models in both real data and randomized data are invariant because the function only calculates information for non-invariance. We see that the intercepts are unlikely to ever replicate across Chinese and English samples. This result is not surprising given the large number of relaxed parameters required to achieve partial invariance.

## Bootstrap Partial Invariance

boot.partial.rs <-   
 bootstrap\_partial(  
 saved\_model = results.rs$invariance\_models$model.intercepts,  
 data = DF.combo,  
 model = model.rs,   
 group = "group",   
 nboot = 1000,   
 invariance\_index = "cfi",  
 invariance\_rule = .01,   
 invariance\_compare = fitmeasures(results.rs$invariance\_models$model.loadings, "cfi"),   
 partial\_step = "intercepts",  
 group.equal = c("loadings", "intercepts")  
)  
  
saveRDS(boot.partial.rs, "manu\_data/boot.partial.rs.appendix7.Rdata")

### Each Parameter on the Overall Model Invariance.

boot.partial.rs$boot\_effects %>%   
 dplyr::select(term, non\_invariant, random\_non\_invariant,   
 h\_nmi, h\_nmi\_p)

## term non\_invariant random\_non\_invariant h\_nmi h\_nmi\_p  
## 1 RS ~1 1 0.002 3.052120 0.971520  
## 2 RS1 ~1 1 0.001 3.078337 0.979865  
## 3 RS10 ~1 1 0.002 3.052120 0.971520  
## 4 RS11 ~1 1 0.002 3.052120 0.971520  
## 5 RS12 ~1 1 0.002 3.052120 0.971520  
## 6 RS13 ~1 1 0.002 3.052120 0.971520  
## 7 RS14 ~1 1 0.002 3.052120 0.971520  
## 8 RS2 ~1 1 0.002 3.052120 0.971520  
## 9 RS3 ~1 1 0.002 3.052120 0.971520  
## 10 RS4 ~1 1 0.002 3.052120 0.971520  
## 11 RS5 ~1 1 0.002 3.052120 0.971520  
## 12 RS6 ~1 1 0.002 3.052120 0.971520  
## 13 RS7 ~1 1 0.001 3.078337 0.979865  
## 14 RS8 ~1 1 0.002 3.052120 0.971520  
## 15 RS9 ~1 1 0.002 3.052120 0.971520

In this output, we see that all the bootstrapped runs of the real data are non-invariant, even when each parameter is relaxed individually. A few runs of the random data are *non*-invariant (meaning most are actually invariant when randomized). This indicates that no one parameter is likely the reason for non-invariance, as they all show large non-replication effects. If we use the boot\_summary, we can see the effect size for each parameter when the two intercepts are compared to each other (as the chart above shows the overall model invariance effect).

### Each Parameter’s Standardized Difference Score.

boot.partial.rs$boot\_summary %>%   
 dplyr::select(term, invariant, n\_boot, d\_boot)

## term invariant n\_boot d\_boot  
## 1 RS ~1 FALSE 1000 NA  
## 2 RS1 ~1 FALSE 1000 0.2913872  
## 3 RS10 ~1 FALSE 1000 0.3596961  
## 4 RS11 ~1 FALSE 1000 -0.6589317  
## 5 RS12 ~1 FALSE 1000 -0.3010202  
## 6 RS13 ~1 FALSE 1000 -0.4005088  
## 7 RS14 ~1 FALSE 1000 -0.5665988  
## 8 RS2 ~1 FALSE 1000 -0.5540349  
## 9 RS3 ~1 FALSE 1000 -0.6218628  
## 10 RS4 ~1 FALSE 1000 -0.6245830  
## 11 RS5 ~1 FALSE 1000 -0.3334520  
## 12 RS6 ~1 FALSE 1000 -0.5014247  
## 13 RS7 ~1 FALSE 1000 -0.2381644  
## 14 RS8 ~1 FALSE 1000 -0.6333750  
## 15 RS9 ~1 FALSE 1000 -0.5986866

boot.partial.rs$boot\_summary %>%   
 dplyr::select(term, invariant, n\_random, d\_random)

## term invariant n\_random d\_random  
## 1 RS ~1 FALSE 2 NA  
## 2 RS1 ~1 FALSE 1 NA  
## 3 RS10 ~1 FALSE 2 NA  
## 4 RS11 ~1 FALSE 2 NA  
## 5 RS12 ~1 FALSE 2 NA  
## 6 RS13 ~1 FALSE 2 NA  
## 7 RS14 ~1 FALSE 2 NA  
## 8 RS2 ~1 FALSE 2 NA  
## 9 RS3 ~1 FALSE 2 NA  
## 10 RS4 ~1 FALSE 2 NA  
## 11 RS5 ~1 FALSE 2 NA  
## 12 RS6 ~1 FALSE 2 NA  
## 13 RS7 ~1 FALSE 1 NA  
## 14 RS8 ~1 FALSE 2 NA  
## 15 RS9 ~1 FALSE 2 NA

With the bootstrap summary, we see that no invariant intercept runs are found (not all columns shown to save space). Note that the RS~1 does not calculate *d* because the *sd* is zero (thus, this would cause an error as *sd* is part of the denominator). The random runs also do not show a *d* score because the effect size is only calculated when at least 10% of the runs in n\_boot or n\_random are found. Since we only have a few in that column, no effect size is calculated. The differences in the intercepts for the real data (boot columns) appear to be medium to large, showing *d* scores from 0.3 to 0.6. Therefore, we might expect that the English and Chinese samples have different average endorsement levels of the RS14.

boot.partial.rs$density\_plot

A diagram of different types of graphs

Description automatically generated with medium confidence

ggsave(paste0("figures/rs14-chinese-density\_plot.png"),   
 dpi = 300,   
 width = 8,   
 units = "in")

The examination of the density plot shows how group 1 (Aiena English) tends to show lower average scores than group 2 (Chen Chinese) for most but not all items.

We can view the mean difference or standardized mean difference by using:

boot.partial.rs$invariance\_plot

A graph with blue and red dots

Description automatically generated

ggsave(paste0("figures/rs14-chinese-invariance\_plot.png"),   
 dpi = 300,   
 width = 8,   
 units = "in")  
  
boot.partial.rs$effect\_invariance\_plot

A graph with red dots

Description automatically generated

ggsave(paste0("figures/rs14-chinese-effect\_invariance\_plot.png"),   
 dpi = 300,   
 width = 8,   
 units = "in")

When effects are non-invariant in the randomized data, the mean difference is still fairly small, but we see large mean differences in intercepts when the bootstrapped data is non-invariant. In the effect size graph, we can see that this effect is medium to large for all the parameters.