Tutorial for Chapter 5: BLOT Invariance Rasch Analysis with the eRm package

Tara L. Valladares & Moritz Heene

July 22, 2020

# Preliminary steps before you can run the BLOT invariance analysis

1. Create a folder called “BLOT\_invariance” on your desktop, for example. Download the data set ARMChapter5.csv and the R code Chapter\_5\_eRm.R from the website and save both in *that* folder. This folder will serve as your working directory containing all files you need to conduct the analysis and to store optional output (i.e., code, data, and figures). If the R code and the .csv files are not in the same folder, you will not be able to load the data using the code below.
2. Open the file Chapter\_5\_eRm.R in RStudio by clicking on the file. This will open the file in RStudio.
3. Go to “Session” “Set Working Directory” “To Source File Location”. This defines the folder you named above as your working directory in which you are currently working and where R expects all data sets to be.

You are now ready to run the Rasch invariance analysis of the BLOT data used in ARM4 Chapter 5. Please use the following instructions and explanations of the R code.

# Rasch Model Invariance Analysis: Data Preparation

First, we will load the necessary packages we will need for this analysis. If you have not yet, you will need to install the packages using the install.packages() function below. This will only need to be done once. Then, we will load the packages into R using library(). This will need to be done each time you open R.

# Installing the required packages for the tutorials.   
# This is only required once.   
# Uncomment the code below to run the install.packages function.  
# install.packages(c("eRm", "dplyr", "ggplot2", "pairwise", "TAM", "psych"))  
  
# Load the packages required for the analysis  
library(eRm)  
library(dplyr)

The line below says, read a .csv file and store it (<-) as an object in R that we will name “df” for short. The file is called ARMChapter5.csv, it has no header (header = FALSE) and a semi-colon is used as the field separator (sep = ";").

# Reading in the comma-seperated data set  
df <- read.csv("ARMChapter5.csv", header = FALSE, sep = ";")

We will need to preprocess and format our data. The following lines of code define column names of the data frame and then create an object called “blot\_items” containing the BLOT items we need for the Rasch analysis. Each BLOT item is represented by a column. Each participant has their own row. There are 35 multiple-choice items administered to 150 persons. Each answer was scored 1 if correct, 0 if not correct.

Our object df also has a column for group and a column for gender. Participants with raw scores of 0-26 are in the Low group (L); 27-35 are in the High group (H). The gender column has the following labels: B = Boy, G = Girl, x = not known. We will use these columns to look for invariance in the BLOT test.

# Defining the name of the first column of the data frame called "df"  
colnames(df)[1] <- c("group")   
colnames(df)[2] <- c("gender")   
# Defining the names of the columns referring to the items  
colnames(df)[3:37] <- paste("Blot", (1:35), sep="\_")   
# Selecting only the BLOT items  
blot\_items <- dplyr::select(df, Blot\_1:Blot\_35)   
# Looking at the first five rows of the blot dataframe  
head(blot\_items)

## Blot\_1 Blot\_2 Blot\_3 Blot\_4 Blot\_5 Blot\_6 Blot\_7 Blot\_8 Blot\_9 Blot\_10  
## 1 1 1 1 1 1 1 1 1 1 1  
## 2 1 1 1 1 1 1 1 1 1 1  
## 3 1 1 0 1 0 1 1 1 1 1  
## 4 1 1 1 1 1 1 1 1 1 1  
## 5 1 1 1 1 1 1 1 1 1 1  
## 6 1 1 1 1 1 1 1 1 1 1  
## Blot\_11 Blot\_12 Blot\_13 Blot\_14 Blot\_15 Blot\_16 Blot\_17 Blot\_18 Blot\_19  
## 1 0 1 1 0 1 0 1 1 0  
## 2 1 1 1 1 1 1 1 1 1  
## 3 1 1 1 1 0 1 1 1 1  
## 4 1 1 1 1 1 1 1 1 1  
## 5 1 1 0 1 1 1 1 1 1  
## 6 1 1 1 0 1 1 1 1 0  
## Blot\_20 Blot\_21 Blot\_22 Blot\_23 Blot\_24 Blot\_25 Blot\_26 Blot\_27 Blot\_28  
## 1 1 0 1 1 1 1 1 1 1  
## 2 1 1 1 1 1 1 1 1 0  
## 3 1 0 1 1 1 1 1 1 0  
## 4 1 1 0 1 1 1 1 1 1  
## 5 1 0 1 1 1 1 1 1 1  
## 6 1 0 1 1 1 1 1 1 1  
## Blot\_29 Blot\_30 Blot\_31 Blot\_32 Blot\_33 Blot\_34 Blot\_35  
## 1 1 0 1 1 1 1 1  
## 2 1 1 1 1 1 1 1  
## 3 1 0 1 1 1 1 1  
## 4 1 1 1 1 1 1 1  
## 5 1 1 1 1 1 1 1  
## 6 1 1 1 1 1 1 1

We are now ready to run the Rasch invariance analysis. In accordance with the corresponding Winsteps analysis, we will

* Fit a Rasch model
* Perform a global likelihood ratio test based on scores
* Fit separate Rasch models for each score group
* Examine ICCs for by gender

First, we will run a Rasch model that contains all the scores from all examinees, regardless of group or gender. Then we will run Andersen’s Likelihood Ratio Test on the Rasch model to look for invariance. Invariance is a pretty general term - what are we referring to? One way to operationalize, or define, invariance is by saying that the *true* difficulty of an item does not change depending on who is responding to it. This is not the same as saying that an item may be *relatively* easier or harder for a participant to pass if they have more or less ability. Someone with a high ability should always have an easier time passing an item than someone with a low ability. But if we gave the same test to a group of high ability persons and to a group of low ability persons, and then placed the results on the same scale, then our estimated item difficulties should be the same. If the item parameters are estimated the same in different contexts, we can say that the item is *invariant*. Invariance doesn’t have to only involve high and low scoring participants. We could also look for invariance across other relevant dimensions. If we were validating an academic test, we could look for invariance across the different schools we sampled or among students from different socio-economic backgrounds.

Andersen’s LR-test is a global goodness-of-fit test that we can use to look for invariance. It looks for differential item functioning on all items at once. A significant result suggests that the two groups are not responding to the test in the same way; the item difficulties are likely not invariant.

This is what we see below when we run the LR-test on the high and low scoring groups using the LRtest() function. We split the data by high and low scorers using the group column in the *df* object (splitcr = df$group). A significant result means that there likely is a difference between our two groups.

# Fit a Rasch model for the BLOT data  
fit\_rasch\_blot <- RM(blot\_items)  
# Perform the Likelihood Ratio test based on the group variable  
(lrtest <- LRtest(fit\_rasch\_blot, splitcr = df$group))

## Warning in LRtest.Rm(fit\_rasch\_blot, splitcr = df$group):   
## The following items were excluded due to inappropriate response patterns within  
## subgroups:  
## Blot\_7 Blot\_12  
##   
## Full and subgroup models are estimated without these items!

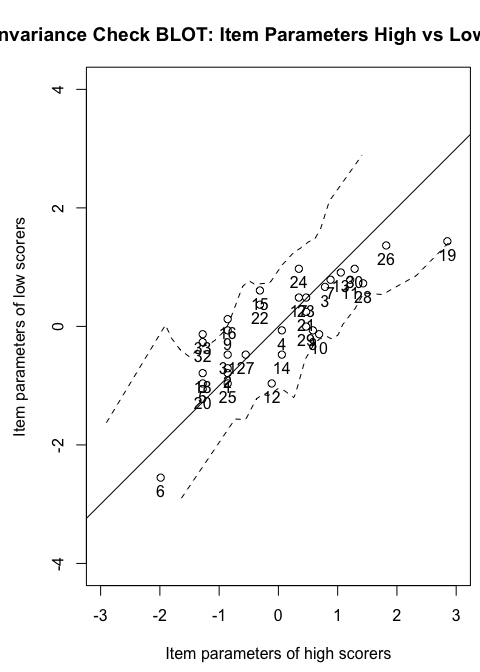
##   
## Andersen LR-test:   
## LR-value: 49.619   
## Chi-square df: 32   
## p-value: 0.024

Because this is a global test, it does not tell us where, exactly, the differences are at. We will need to look at the items individually to see which items are causing problems. You may also see an error come out when you ran the LRtest() suggesting that some of our items have suspicious response patterns. That’s okay - because our result is already significant, we will need to examine all the items more closely anyway.

# Plots

We can take a closer look at the differences in the item parameters by plotting the results of our LR-test. In the graph below, we will plot the estimated item parameters using our low scorers against the estimated item parameters from our high scorers. The item parameters of the low scorers are on the y-axis, while the item parameters of the high scorers are on the x-axis. The diagonal line in the middle marks the identity line - where both x and y are equal. Each dot represents an item. When items are close to the identity line, the item parameters from high and low scores are more similar. The further away the dots are from the line, the more different the items were estimated to be. When items are above the line, they are harder for low scoring participants. When items are below the line, they are harder for high scoring participants. The dotted lines are the 95% confidence bands. If the plot is hard to read, try making the plot pane bigger and re-running the code, or clicking on the zoom button. Don’t worry too much about reading the plotting code - it can be quite complex in R!

plotGOF(lrtest, col = "black", tlab = "number",   
 main = "Invariance Check BLOT: Item Parameters High vs Low Scorers",   
 xlab = "Item parameters of high scorers", ylab = "Item parameters of low scorers",   
 xlim = c(-3, 3), ylim = c(-3, 3), pos = 1,   
 smooline = list(gamma = .95, col = "black", lty = "dashed"))



In a perfect world, invariant items would lie exactly on the identity line, but this is not a realistic assumption. How can we determine how far is too far, then? This is where the confidence bands are used. Items that lie between the confidence bands (dotted lines) are assumed to be invariant. Items that lie outside of the bands, like item 32, are assumed to have item parameters that are statistically significantly different between the two groups. The distance between the confidence bands is partly based on how many participants we have; the more participants who take the test, the more sure we can be that any difference between the item parameters is not due to sample variation.

When we look at the BLOT plot, we can see that there are a couple of misfitting items, like items 32, 33, and 16. Most of our items fall within the confidence bands and do not seem to be different between the two groups. We also have some suspicious items, like 19, which fall very closely to the boundary lines. We will look at our items more closely in the next section to determine where these differences are coming from.

Finally, we could also use this plot to look for patterns in which item are different between groups. Here, there isn’t an obvious pattern, but in other situations you may see that items on a certain topic or of a certain difficulty show significant variance.

## Getting more Detailed Information on Item Parameters in Both Score Groups:

###Separate Analysis for high vs low scorers

First, let’s run a Rasch model with the responses from the high scorers. We will subset our data using the which function. This will produce a vector of TRUE’s and FALSE’s depending on if each element of df$group equals “H”.

# Estimates Rasch model parameters for high scorers   
fit\_rasch\_blot\_high <- eRm::RM(blot\_items[which(df$group == "H"), ])

## Warning:   
## The following items have no 0-responses:  
## Blot\_7 Blot\_12  
## Responses are shifted such that lowest category is 0.

## Warning:   
## The following items were excluded due to complete 0/full responses:  
## Blot\_7 Blot\_12

You will see some warnings come out when running the Rasch model for high scorers: The following items have no 0-responses: Blot\_7 Blot\_12. Responses are shifted such that lowest category is 0and The following items were excluded due to complete 0/full responses: Blot\_7 Blot\_12.

The RM function is warning us that for two of our items (7 and 12), everyone got this item correct. Because everyone got it right, the Rasch model cannot estimate its difficulty; it could be anywhere from 4 to infinity. Some programs will give an educated guess about the true difficulty of the item, but *eRm*’s RM function does not. It is not surprising that we ran into this as everyone in this model was a high scorer; these were easier items.

Now let’s run a separate Rasch model for the low scorers in the same way.

# Estimates Rasch model parameters for low scorers  
fit\_rasch\_blot\_low <- eRm::RM(blot\_items[which(df$group == "L"), ])

Now that we have separate models, we can look at the item fit statistics for each model.

# Item fit statistics for low scorers  
pp\_low <- person.parameter(fit\_rasch\_blot\_low)  
item\_fit\_low <- itemfit(pp\_low)  
print(item\_fit\_low)

##   
## Itemfit Statistics:   
## Chisq df p-value Outfit MSQ Infit MSQ Outfit t Infit t Discrim  
## Blot\_1 67.546 70 0.561 0.951 0.992 -0.198 0.002 0.307  
## Blot\_2 72.877 70 0.384 1.026 1.029 0.207 0.246 0.246  
## Blot\_3 69.495 70 0.495 0.979 0.982 -0.180 -0.246 0.286  
## Blot\_4 70.070 70 0.475 0.987 0.989 -0.085 -0.079 0.316  
## Blot\_5 73.885 70 0.352 1.041 0.985 0.258 -0.031 0.274  
## Blot\_6 46.232 70 0.987 0.651 0.997 -0.520 0.132 0.293  
## Blot\_7 77.171 70 0.260 1.087 1.082 0.647 0.700 0.154  
## Blot\_8 58.320 70 0.839 0.821 0.873 -1.728 -1.944 0.470  
## Blot\_9 70.340 70 0.466 0.991 1.004 -0.049 0.071 0.288  
## Blot\_10 71.053 70 0.442 1.001 0.992 0.043 -0.050 0.300  
## Blot\_11 62.794 70 0.717 0.884 0.892 -1.017 -1.091 0.443  
## Blot\_12 29.933 70 1.000 0.422 0.673 -2.160 -1.371 0.767  
## Blot\_13 78.394 70 0.230 1.104 1.086 0.969 1.266 0.108  
## Blot\_14 76.401 70 0.281 1.076 1.083 0.417 0.520 0.161  
## Blot\_15 65.395 70 0.634 0.921 0.970 -0.648 -0.401 0.301  
## Blot\_16 78.591 70 0.225 1.107 1.110 0.742 0.873 0.163  
## Blot\_17 65.469 70 0.631 0.922 0.957 -0.792 -0.646 0.340  
## Blot\_18 67.978 70 0.546 0.957 0.966 -0.412 -0.387 0.324  
## Blot\_19 70.861 70 0.449 0.998 1.004 0.010 0.086 0.272  
## Blot\_20 63.678 70 0.690 0.897 0.911 -0.501 -0.532 0.436  
## Blot\_21 89.971 70 0.054 1.267 1.050 1.448 0.483 0.064  
## Blot\_22 62.095 70 0.738 0.875 0.880 -0.495 -0.619 0.449  
## Blot\_23 75.601 70 0.302 1.065 1.074 0.714 0.974 0.176  
## Blot\_24 66.401 70 0.600 0.935 0.919 -0.694 -1.140 0.402  
## Blot\_25 90.707 70 0.049 1.278 1.108 2.779 1.566 0.074  
## Blot\_26 73.728 70 0.357 1.038 1.037 0.348 0.512 0.185  
## Blot\_27 53.080 70 0.934 0.748 0.804 -1.219 -1.149 0.578  
## Blot\_28 77.106 70 0.262 1.086 1.016 0.554 0.190 0.154  
## Blot\_29 66.424 70 0.599 0.936 0.941 -0.392 -0.427 0.385  
## Blot\_30 80.637 70 0.181 1.136 1.131 1.288 1.920 0.057  
## Blot\_31 68.892 70 0.515 0.970 0.983 -0.252 -0.157 0.311  
## Blot\_32 58.600 70 0.833 0.825 0.886 -1.462 -1.575 0.435  
## Blot\_33 83.562 70 0.128 1.177 1.147 1.172 1.137 0.061  
## Blot\_34 75.257 70 0.312 1.060 1.071 0.508 0.669 0.190  
## Blot\_35 69.578 70 0.492 0.980 0.997 -0.139 0.007 0.298

# Item fit statistics for high scorers  
pp\_high <- person.parameter(fit\_rasch\_blot\_high)  
item\_fit\_high <- itemfit(pp\_high)

## Warning in cor.smooth(R): Matrix was not positive definite, smoothing was done

## In smc, smcs < 0 were set to .0

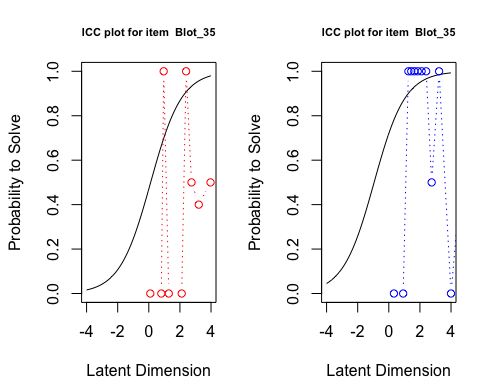
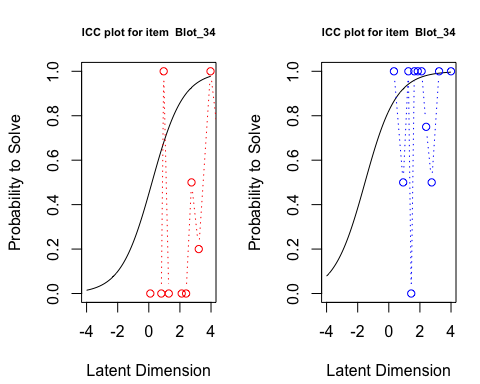
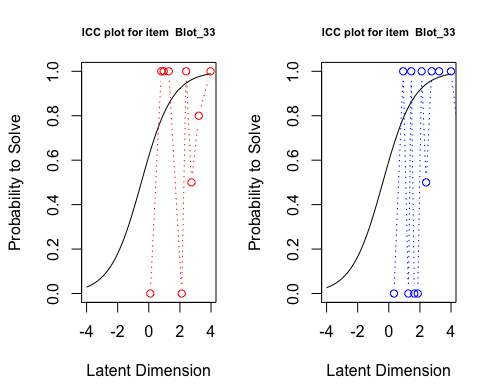
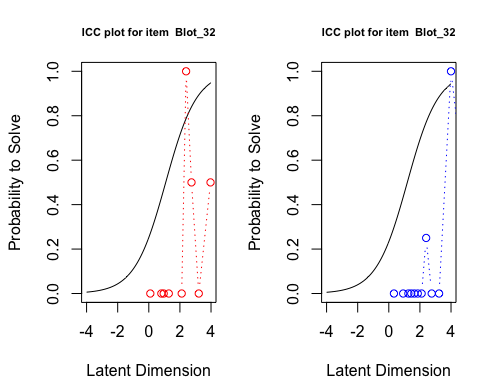
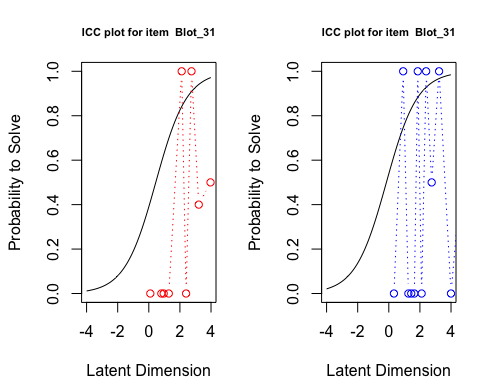
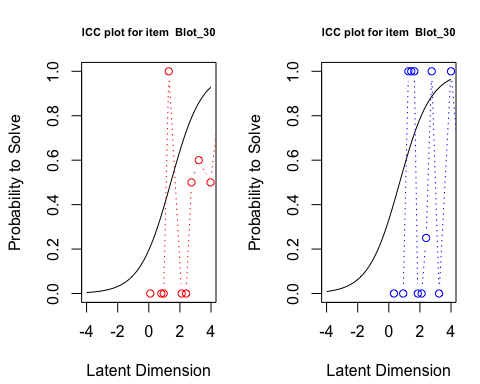
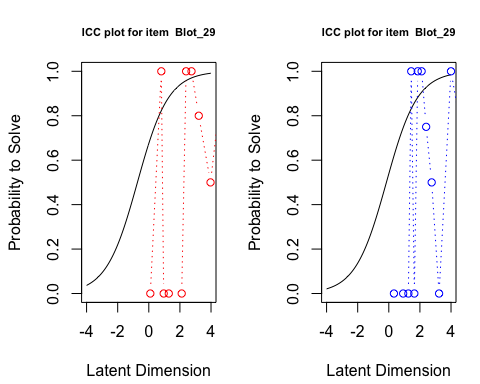
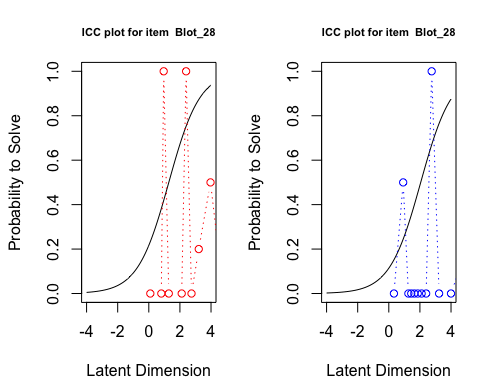
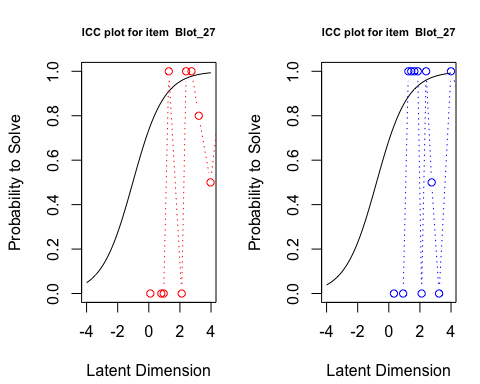
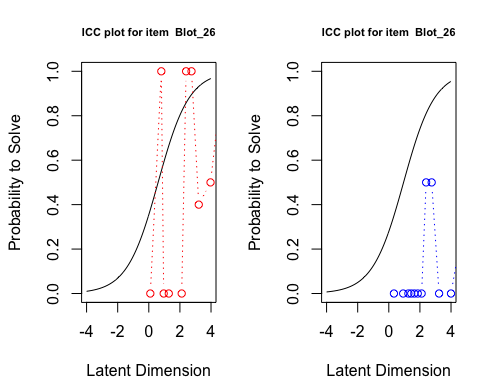
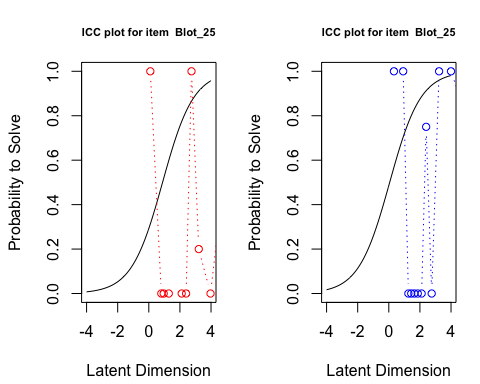
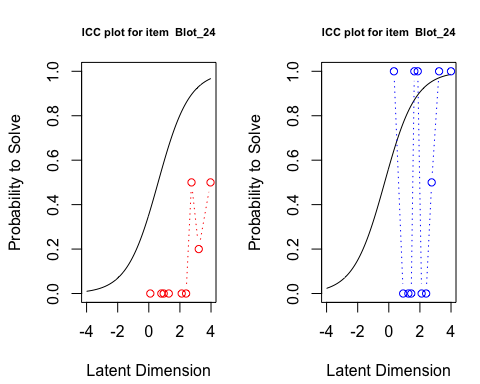
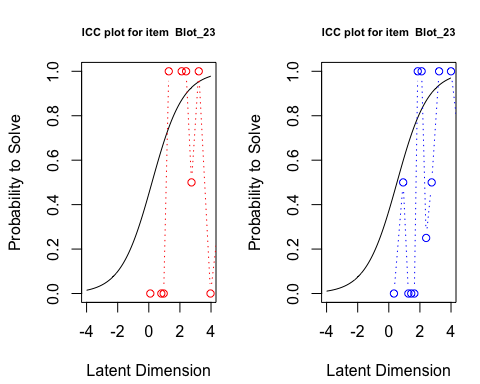
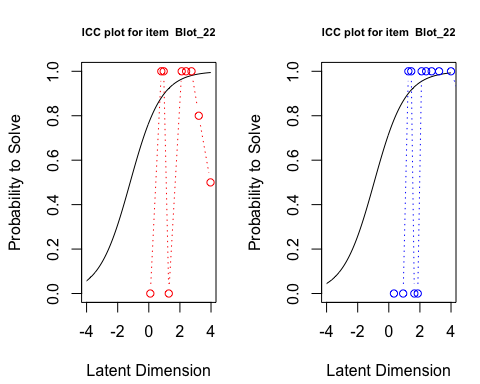
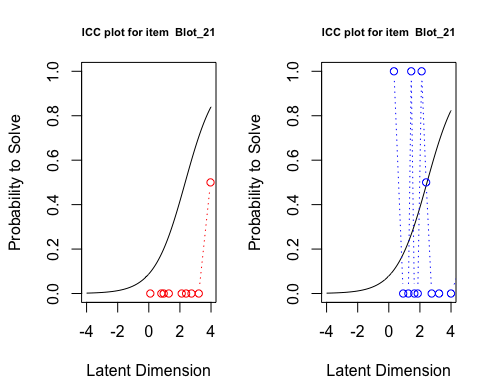
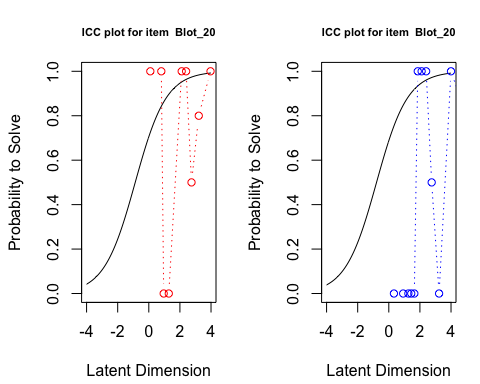
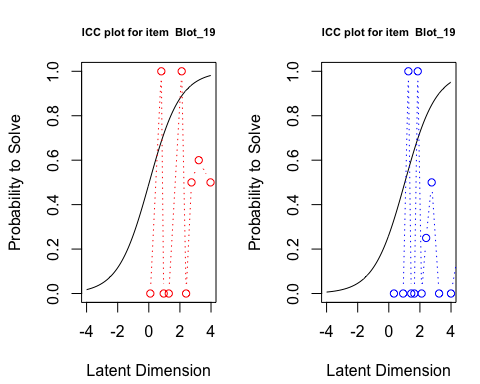
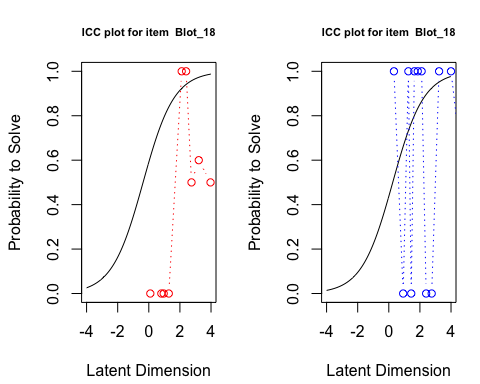
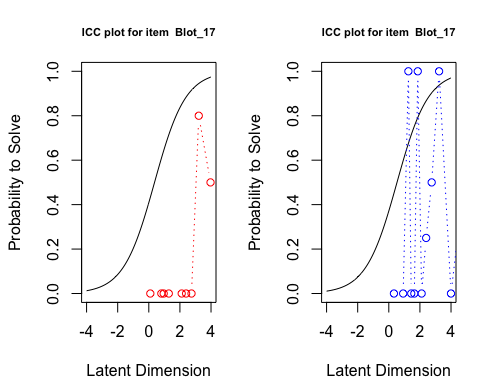
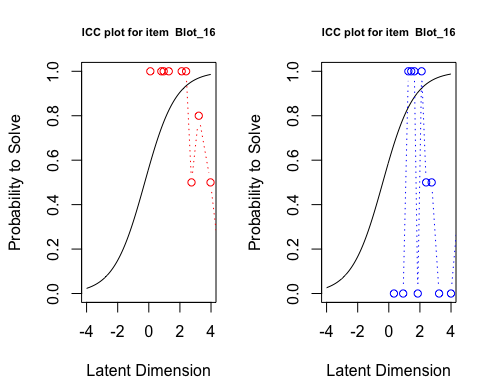
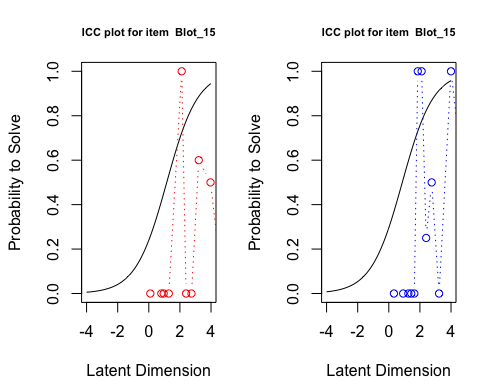
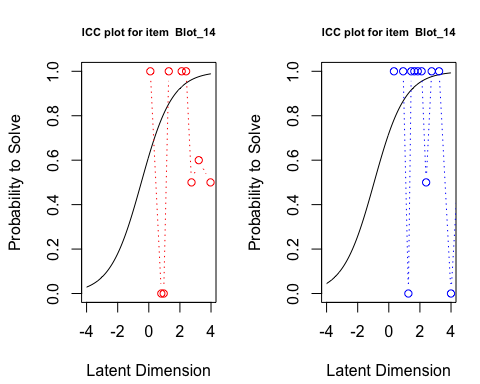
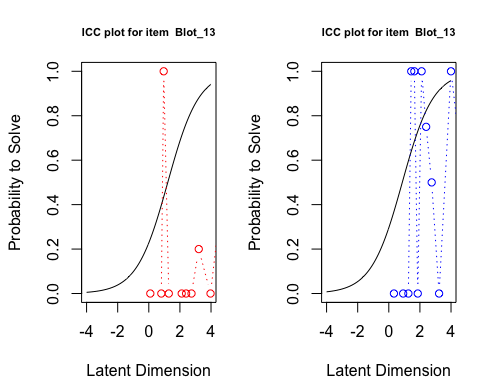
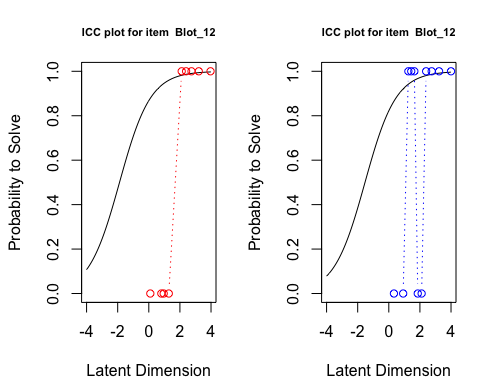
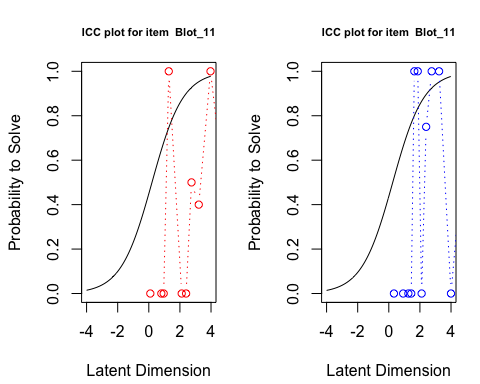
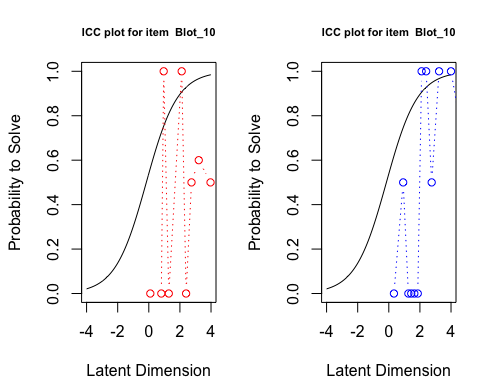
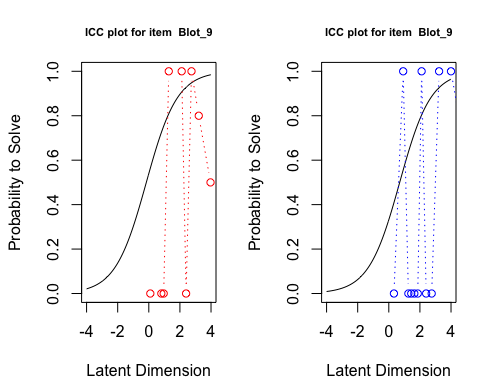
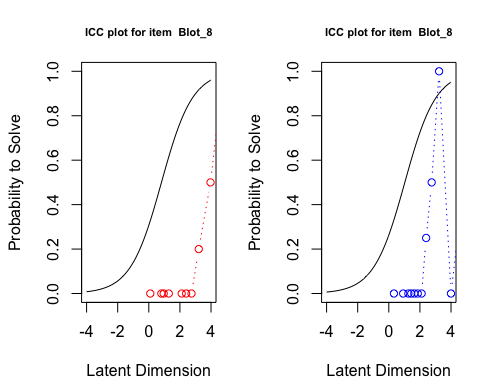
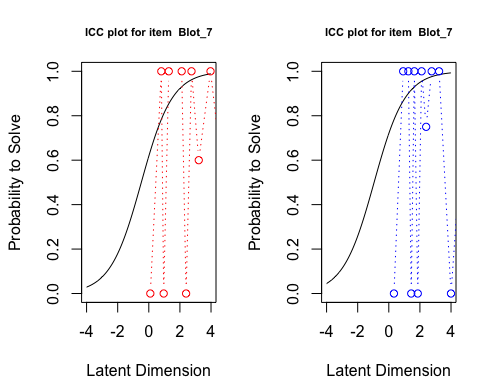
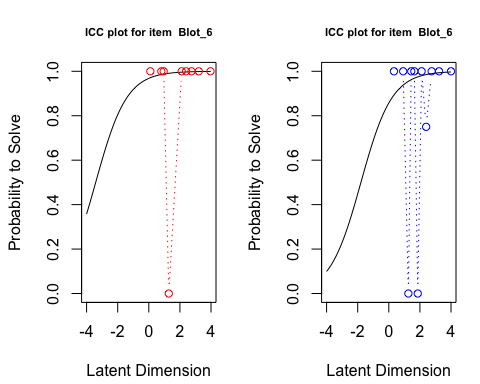
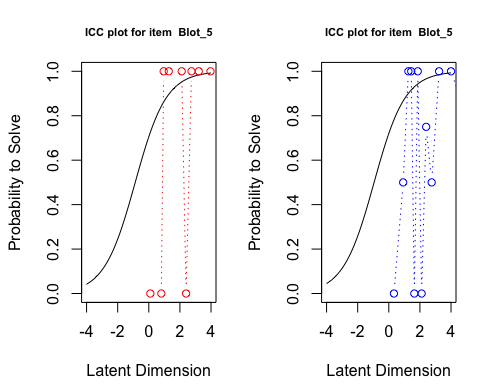
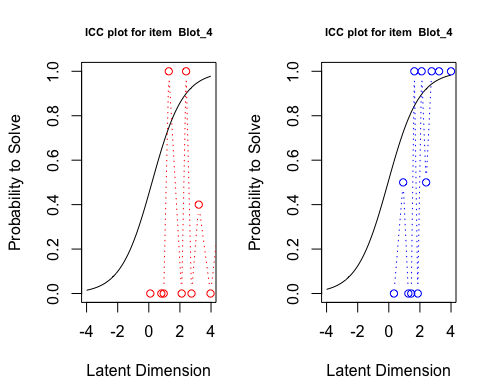
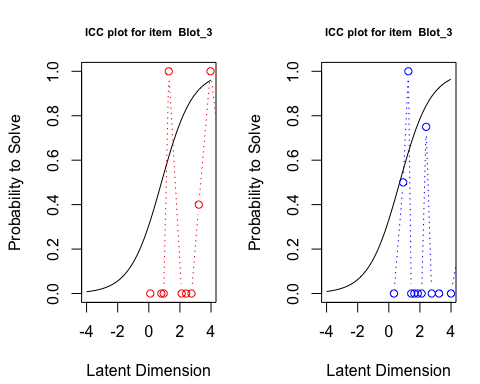
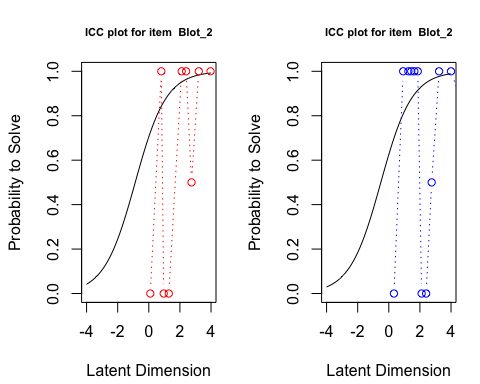
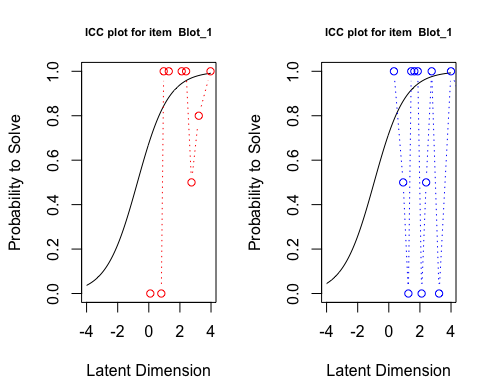
print(item\_fit\_high)

##   
## Itemfit Statistics:   
## Chisq df p-value Outfit MSQ Infit MSQ Outfit t Infit t Discrim  
## Blot\_1 31.284 75 1.000 0.412 0.885 -0.739 -0.064 0.505  
## Blot\_2 36.511 75 1.000 0.480 0.906 -0.586 -0.018 0.430  
## Blot\_3 61.999 75 0.859 0.816 0.953 -0.529 -0.195 0.321  
## Blot\_4 55.466 75 0.956 0.730 0.968 -0.474 -0.010 0.231  
## Blot\_5 38.702 75 1.000 0.509 0.924 -0.310 0.095 0.238  
## Blot\_6 46.297 75 0.996 0.609 0.942 0.127 0.250 0.104  
## Blot\_8 92.133 75 0.087 1.212 0.965 0.775 -0.146 0.312  
## Blot\_9 55.432 75 0.956 0.729 0.936 -0.735 -0.238 0.356  
## Blot\_10 36.511 75 1.000 0.480 0.906 -0.586 -0.018 0.337  
## Blot\_11 56.442 75 0.946 0.743 0.950 -0.748 -0.189 0.256  
## Blot\_13 101.619 75 0.022 1.337 1.146 1.360 1.052 -0.001  
## Blot\_14 72.146 75 0.572 0.949 1.003 0.074 0.120 0.023  
## Blot\_15 56.807 75 0.942 0.747 0.934 -0.963 -0.388 0.330  
## Blot\_16 55.201 75 0.958 0.726 0.946 -0.483 -0.083 0.216  
## Blot\_17 68.700 75 0.682 0.904 0.984 0.021 0.082 0.168  
## Blot\_18 69.012 75 0.673 0.908 0.985 0.134 0.142 0.060  
## Blot\_19 94.192 75 0.066 1.239 1.099 0.686 0.468 -0.125  
## Blot\_20 76.284 75 0.437 1.004 0.961 0.324 0.154 0.030  
## Blot\_21 72.575 75 0.558 0.955 0.965 -0.370 -0.332 0.321  
## Blot\_22 233.819 75 0.000 3.077 0.993 1.741 0.205 -0.129  
## Blot\_23 55.541 75 0.955 0.731 0.949 -0.668 -0.150 0.271  
## Blot\_24 138.284 75 0.000 1.820 1.056 1.338 0.272 -0.149  
## Blot\_25 101.035 75 0.024 1.329 1.027 0.921 0.192 0.118  
## Blot\_26 47.069 75 0.995 0.619 0.895 -0.965 -0.353 0.389  
## Blot\_27 33.493 75 1.000 0.441 0.896 -0.673 -0.040 0.328  
## Blot\_28 89.537 75 0.121 1.178 1.087 1.121 0.913 0.137  
## Blot\_29 37.285 75 1.000 0.491 0.899 -0.745 -0.100 0.359  
## Blot\_30 68.249 75 0.696 0.898 1.006 -0.444 0.091 0.224  
## Blot\_31 131.201 75 0.000 1.726 1.080 1.720 0.418 -0.015  
## Blot\_32 61.442 75 0.870 0.808 0.965 -0.829 -0.229 0.299  
## Blot\_33 63.784 75 0.819 0.839 0.964 0.036 0.099 0.027  
## Blot\_34 70.012 75 0.641 0.921 0.973 0.235 0.174 0.044  
## Blot\_35 70.012 75 0.641 0.921 0.973 0.235 0.174 0.044

## Empirical and theoretical ICC per gender group

The BLOT data also has a column for gender, which is examined at the end of Chapter 5. We will use the ICC plots to look for differential item functioning between the boys and girls. Here, we can plot the expected ICC against the empirical function, that is, the actual frequencies of positive responses across the latent dimension. The relative frequencies of the positive responses are calculated for each raw score group and plotted at the position of the corresponding person parameter. For more about how to understand these plots, see the Chapter 4 tutorial. We will plot ICCs separately for boys (in blue) and girls (in red). You may have to hit enter on your keyboard after running the code to see all the plots.

If we plot the ICCs separately we can look for differences between how the boys and girls are actually responding to each of the items. Look at item 35 and contrast the performance of the boys and girls. In general, boys perform better that expected; performance is generally poorer than expected for girls of the same ability on BLOT. DIF favors boys on this item.



# Uncomment the code below if you would like to plot all the boys together, and then all the girls together  
  
# Actual vs theoretical ICC for boys:  
# plotICC(RM(blot\_items[which(df$gender == "B"), ]), empICC = list("raw", type = "b", col = "blue", lty = "dotted"))  
  
# Actual vs theoretical ICC for girls:  
# plotICC(RM(blot\_items[which(df$gender == "G"), ]), empICC = list("raw", type = "b", col = "red", lty = "dotted"))