Full tutorial

Danny

1/5/2020

Introduction

This is meant to be a very general introduction for using the Rasch model to help construct measures and surveys in discipline based education research. This is meant to get you started but is by no means where you should stop. Please see the references section for where to go next.

Installing R and R-Studio

Instructions for installing R:

1. Go to this web page: http://cran.stat.ucla.edu/
2. Under the "Download and Install R" heading, select your operating system (Windows, Mac, Linux).
3. The directions diverge at this stage, depending on your OS.

For Mac, do the following:

1. Under the "Latest Release" heading, select the top ".pkg" link. Save the file to your computer.
2. This is the basic installer file.

For Windows do the following:

1. Under the "Subdirectories" heading, select the top "base" link. Save the file to your computer.
2. This is the basic installer file.

For Linux, you already know what you're doing. I'll stay out of your way.

1. Download and open the installer file. Now, just follow the instructions to set up R. The default settings are fine. No need to open the program yet.
2. Now, we're going to download R-Studio, which is the user interface that makes R faster and easier to use. It's an integrated development environment (IDE)
3. Once you have R-Studio, you won't need to open the "base R" GUI anymore, since R-Studio does this for you.

Instructions for installing R-Studio:

1. Go to this web page: https://www.rstudio.com/products/rstudio/download/#download
2. Under the `Installers for Supported Platforms` header, select your operating system (Windows, Mac, Linux).
3. Download and open the installer file and follow the instructions. The default settings are fine.
4. Once R-Studio is installed, go ahead and open the program from your applications list (Start Menu/Launchpad/Desktop).

Note: if R-Studio does not open, one common reason is that you have installed multiple versions of base R and R-Studio does not know which one to access. This can happen if you have updated R over time or uninstalled it and reinstalled it in the past. To remedy this situation, follow the instructions in the Appendix of this document. If that doesn't work, contact Anthony at `clairmont@ucsb.edu`.

- You may be asked whether you want to install "Command Line Developer Tools." R-Studio needs these to function, so say "Install" and "Agree" to the terms. When R-Studio is done installing its necessary tools, it is read to use! That's all you need to do for now.

Appendix: Troubleshooting R-Studio installation

- On Mac, go to MacintoshHD/Library/Frameworks/R.framework/Versions and delete files associated with the older version of R. When you try to launch R-Studio again, it should automatically find the remaining, current version of R.
- On Windows or Linux, this guide will show you how to tell R-Studio which version of base R to run: https://support.rstudio.com/hc/en-us/articles/200486138-Changing-R-versions-for- RStudio-desktop

The Rasch Model

set your working directory to the folder where you downloaded the CSV file.

1.Go to Session -> Set Working Directory -> Choose Directory

2. install TAM and the WrightMap package

```
install.packages("TAM")
install.packages("WrightMap")
```

Take a {#Rasch} CSV from outside of R and read it in. This means that it is something you can now work with in R. The .csv file will be read in as something called a data frame or (dataframe). This is a type of object in R, that's essentially a spreadsheet that your're used to working with.

```
hls <- read.csv("hls_dic_scale.csv")
```

See the first few rows and columns

```
head(hls)
```

```
##   Hls1 Hls2 Hls3 Hls4 Hls5 Hls6 Hls7 Hls8 Hls9 Hls10 Hls11 Hls12 Hls13
## 1    0    0    0    0    0    0    0    1    0     0     1     0     0
## 2    1    0    0    0    1    0    0    1    0     0     1     1     1
## 3    0    0    0    0    0    0    0    1    0     0     0     1     0
## 4    0    0    0    0    1    0    0    0    0     0     1     0     0
## 5    0    0    0    0    0    0    0    1    0     0     1     1     1
## 6    0    0    0    0    1    0    0    0    0     0     1     1     1
##   Hls14 Hls15 Hls16
## 1     0     0     0
## 2     0     0     1
## 3     0     0     0
## 4     0     1     0
## 5     0     0     1
## 6     0     1     0
```

If you want to see view the data frame:

```
View(hls)
```

Now we call the TAM library you installed in a prior step. This tells R to use the set of functions in available in TAM

```
library(TAM)
```

Run the Rasch Model

This command runs a Rasch model on the selected data frame. `mod1` is an object in R that holds the data from our Rasch model (along with a lot of other information). This is the main computation step, now we just ask TAM questions about this model.

Note that the object `hls` has to contain only items and no other information.

```
mod1 <- tam(hls)
```

```
summary(mod1)
```

```
## -----------------------------------------------------------------
## TAM 3.3-10 (2019-08-23 13:42:07)
## R version 3.5.2 (2018-12-20) x86_64, mingw32 | nodename=LAPTOP-K7402PLE | login=katzd
##
## Date of Analysis: 2020-01-08 11:44:41
## Time difference of 0.0830009 secs
## Computation time: 0.0830009
##
## Multidimensional Item Response Model in TAM
##
## IRT Model: 1PL
## Call:
## tam.mml(resp = resp)
##
## -----------------------------------------------------------------
## Number of iterations = 46
## Numeric integration with 21 integration points
##
## Deviance = 3761.76
## Log likelihood = -1880.88
## Number of persons = 317
## Number of persons used = 317
## Number of items = 16
## Number of estimated parameters = 17
##      Item threshold parameters = 16
##      Item slope parameters = 0
##      Regression parameters = 0
##      Variance/covariance parameters = 1
##
## AIC = 3796   | penalty = 34      | AIC=-2*LL + 2*p
## AIC3 = 3813  | penalty = 51      | AIC3=-2*LL + 3*p
## BIC = 3860   | penalty = 97.9    | BIC=-2*LL + log(n)*p
## aBIC = 3806  | penalty = 43.77    | aBIC=-2*LL + log((n-2)/24)*p  (adjusted BIC)
## CAIC = 3877  | penalty = 114.9    | CAIC=-2*LL + [log(n)+1]*p  (consistent AIC)
## AICc = 3798  | penalty = 36.05    | AICc=-2*LL + 2*p + 2*p*(p+1)/(n-p-1)  (bias corrected AIC)
##
## -----------------------------------------------------------------
## EAP Reliability
## [1] 0.772
## -----------------------------------------------------------------
## Covariances and Variances
##        [,1]
## [1,] 2.932
## -----------------------------------------------------------------
```

```
## Correlations and Standard Deviations (in the diagonal)
##         [,1]
## [1,] 1.712
## ----------------------------------------------------------------
## Regression Coefficients
##         [,1]
## [1,]     0
## ----------------------------------------------------------------
## Item Parameters -A*Xsi
##      item   N     M xsi.item AXsi_.Cat1 B.Cat1.Dim1
## 1    Hls1 317 0.158    2.429      2.429           1
## 2    Hls2 317 0.107    3.004      3.004           1
## 3    Hls3 317 0.038    4.362      4.362           1
## 4    Hls4 317 0.028    4.707      4.707           1
## 5    Hls5 317 0.271    1.491      1.491           1
## 6    Hls6 317 0.098    3.134      3.134           1
## 7    Hls7 317 0.047    4.087      4.087           1
## 8    Hls8 317 0.356    0.919      0.919           1
## 9    Hls9 317 0.085    3.324      3.324           1
## 10  Hls10 317 0.047    4.087      4.087           1
## 11  Hls11 317 0.486    0.122      0.122           1
## 12  Hls12 317 0.426    0.487      0.487           1
## 13  Hls13 317 0.281    1.424      1.424           1
## 14  Hls14 317 0.227    1.821      1.821           1
## 15  Hls15 317 0.369    0.839      0.839           1
## 16  Hls16 317 0.199    2.053      2.053           1
##
## Item Parameters in IRT parameterization
##      item alpha  beta
## 1    Hls1     1 2.429
## 2    Hls2     1 3.004
## 3    Hls3     1 4.362
## 4    Hls4     1 4.707
## 5    Hls5     1 1.491
## 6    Hls6     1 3.134
## 7    Hls7     1 4.087
## 8    Hls8     1 0.919
## 9    Hls9     1 3.324
## 10  Hls10     1 4.087
## 11  Hls11     1 0.122
## 12  Hls12     1 0.487
## 13  Hls13     1 1.424
## 14  Hls14     1 1.821
## 15  Hls15     1 0.839
## 16  Hls16     1 2.053
```

Item Difficulties

So how difficult were those items? let's ask TAM. We'll extract difficulties (xsi) from the mod1 object (think of mod1 like a list). We'll access this via indexing. The $ sign means, access mod1 and extract the object xsi which exists in mod1. The command mod1xsixsi accesses just the column xsi, though, you may want other information at times.

Assign those values to a column in the environment called ItemDiff using <-

```
mod1$xsi
```

```
##             xsi     se.xsi
## Hls1   2.4293464 0.1770696
## Hls2   3.0036806 0.2044455
## Hls3   4.3619882 0.3185594
## Hls4   4.7075062 0.3628821
## Hls5   1.4909467 0.1501134
## Hls6   3.1336976 0.2120718
## Hls7   4.0869798 0.2884162
## Hls8   0.9192688 0.1419071
## Hls9   3.3237826 0.2242965
## Hls10 4.0869798 0.2884162
## Hls11 0.1220405 0.1384023
## Hls12 0.4867469 0.1389679
## Hls13 1.4239100 0.1488687
## Hls14 1.8211298 0.1574645
## Hls15 0.8391479 0.1411647
## Hls16 2.0532023 0.1639480
```

```
ItemDiff <- mod1$xsi$xsi
ItemDiff
```

```
##  [1] 2.4293464 3.0036806 4.3619882 4.7075062 1.4909467 3.1336976 4.0869798
##  [8] 0.9192688 3.3237826 4.0869798 0.1220405 0.4867469 1.4239100 1.8211298
## [15] 0.8391479 2.0532023
```

Visualize

We may want to visualize or describe the distribution of item difficulties (if you want to play with binwidth, you can).

Get Item Characteristic Curves

```
plot(mod1, export = F)
```

```
## Iteration in WLE/MLE estimation  1   | Maximal change  2.5494
## Iteration in WLE/MLE estimation  2   | Maximal change  2.2203
## Iteration in WLE/MLE estimation  3   | Maximal change  0.4122
## Iteration in WLE/MLE estimation  4   | Maximal change  0.0234
## Iteration in WLE/MLE estimation  5   | Maximal change  0.003
## Iteration in WLE/MLE estimation  6   | Maximal change  4e-04
## Iteration in WLE/MLE estimation  7   | Maximal change  1e-04
## ----
##  WLE Reliability= 0.615
```
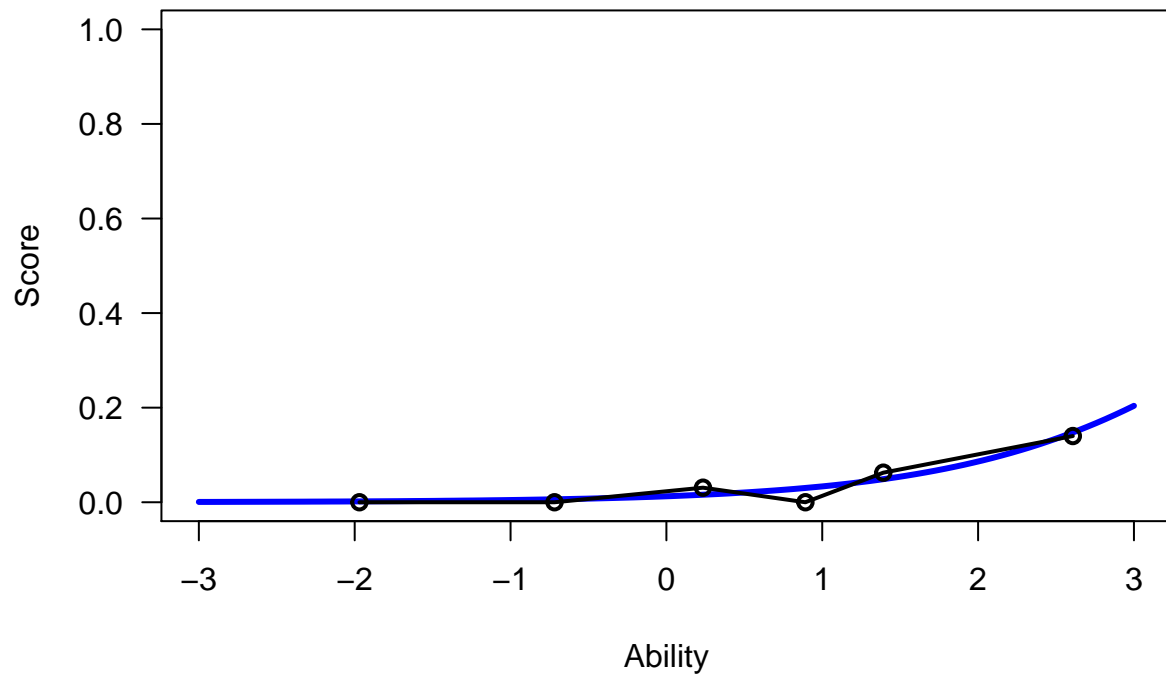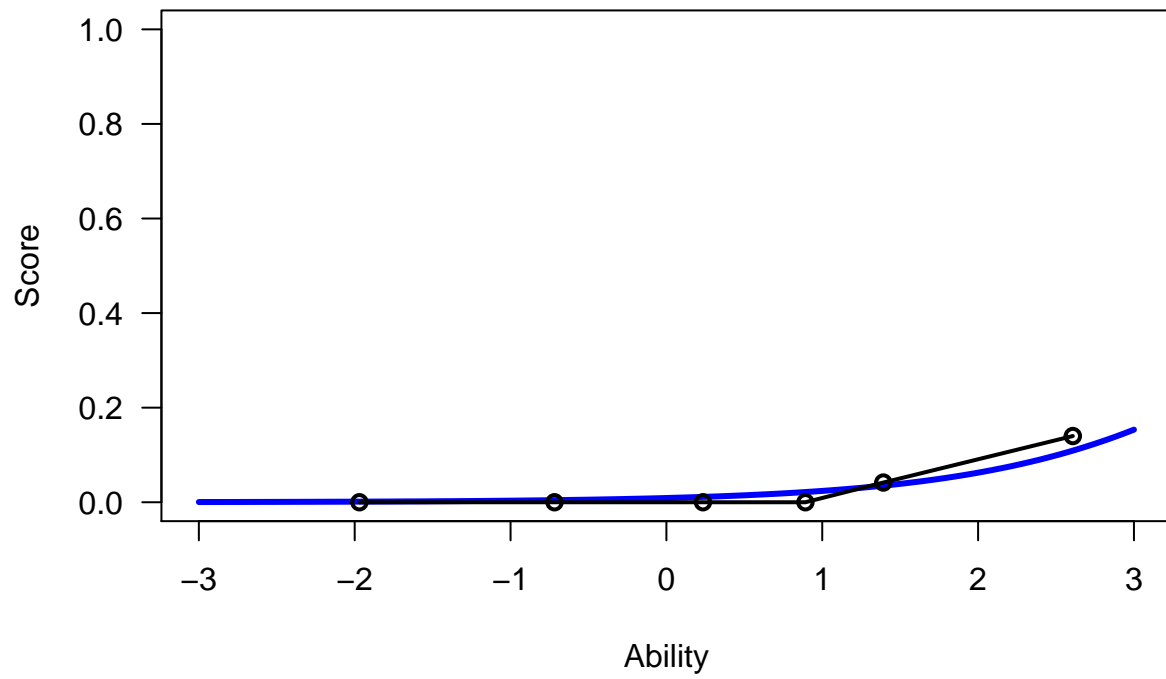
Expected Scores Curve – Item HIs1
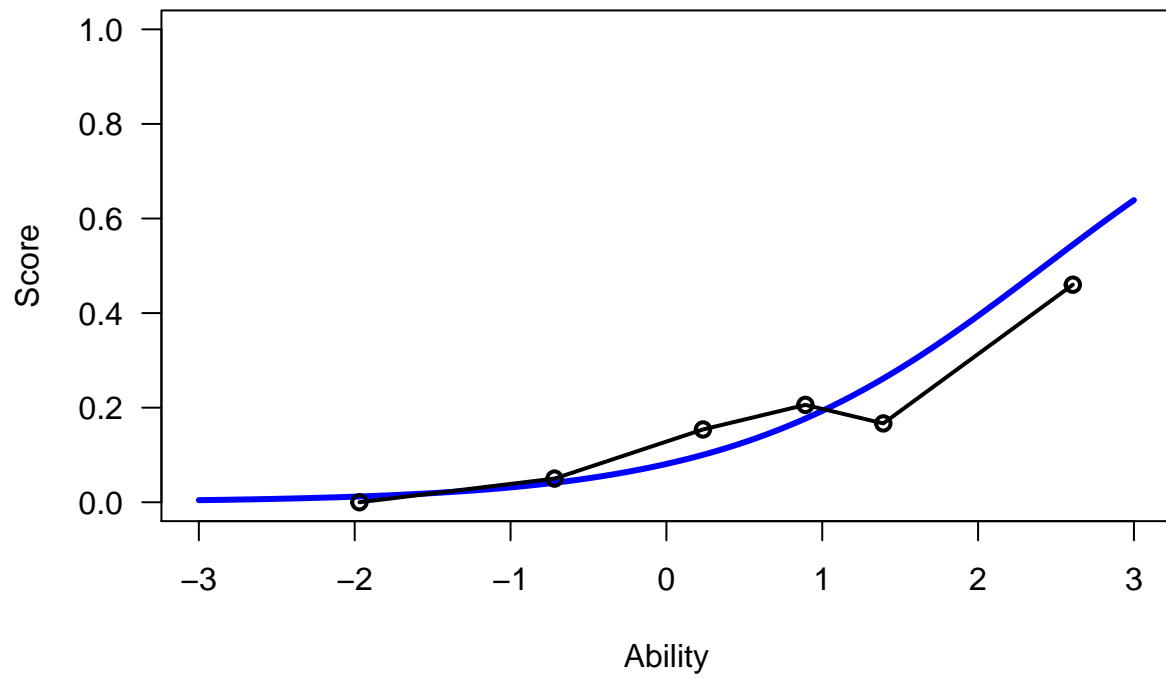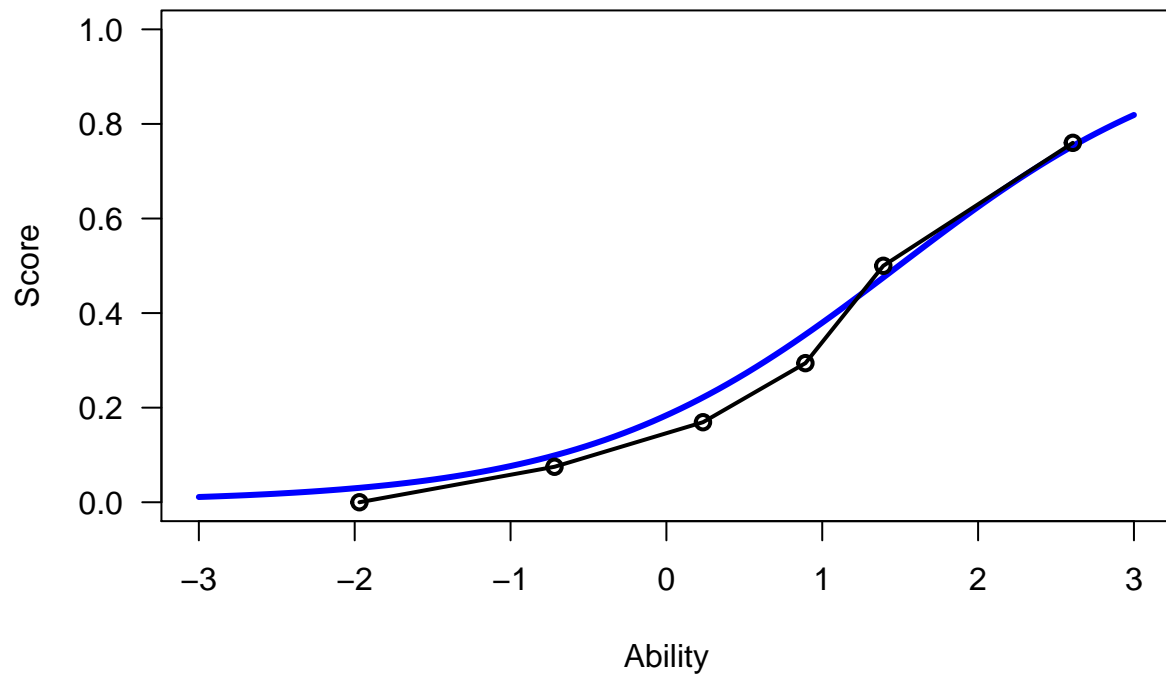
Expected Scores Curve – Item HIs2
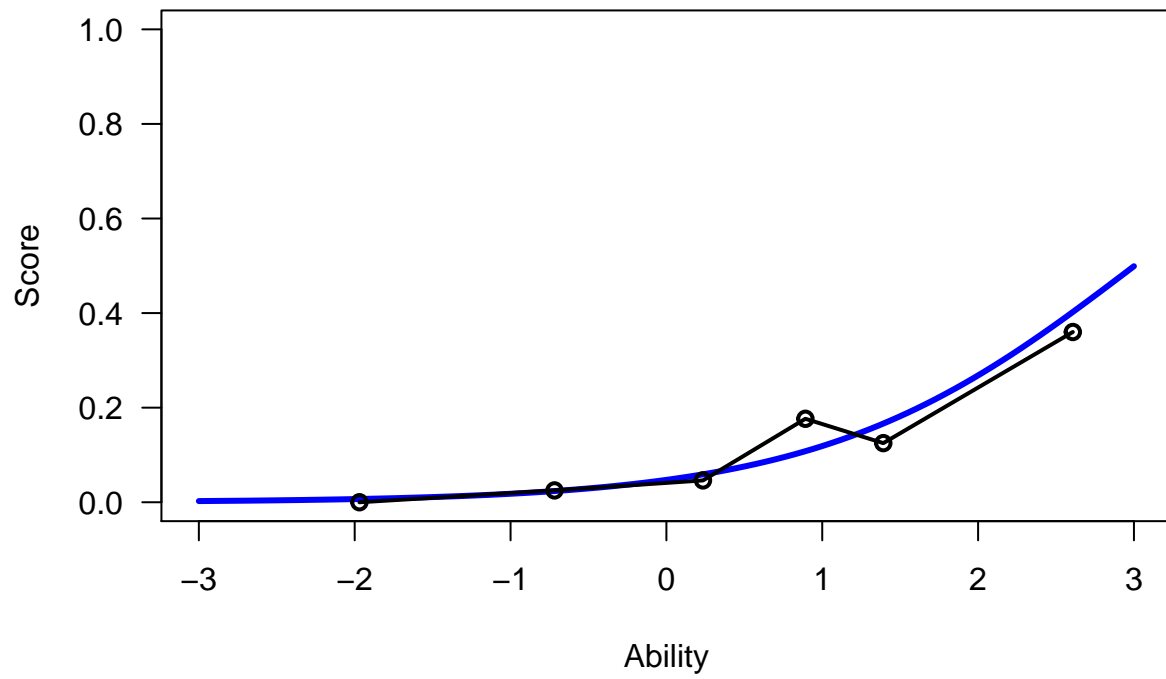
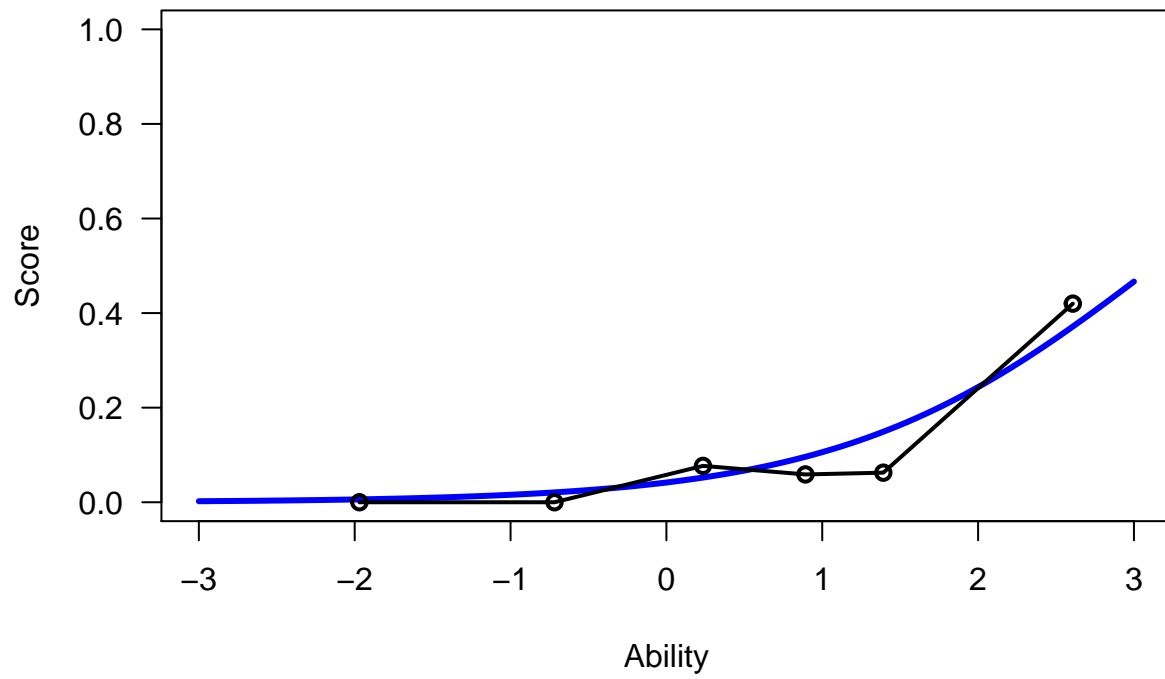# Expected Scores Curve – Item HIs3

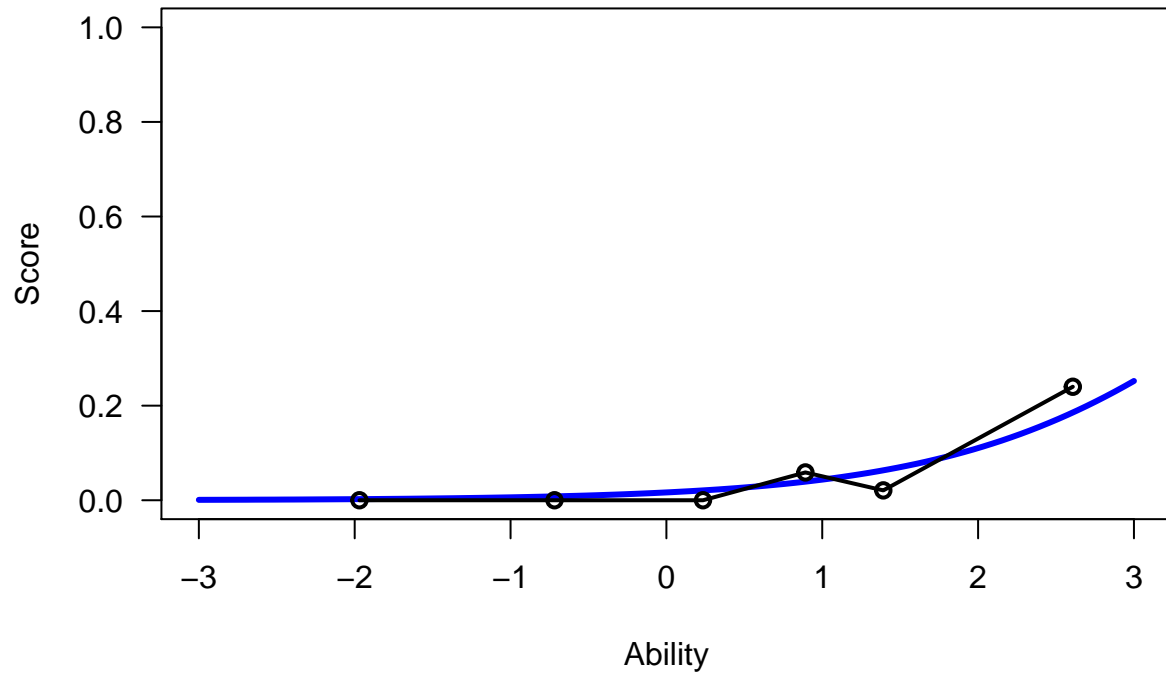# Expected Scores Curve – Item HIs4
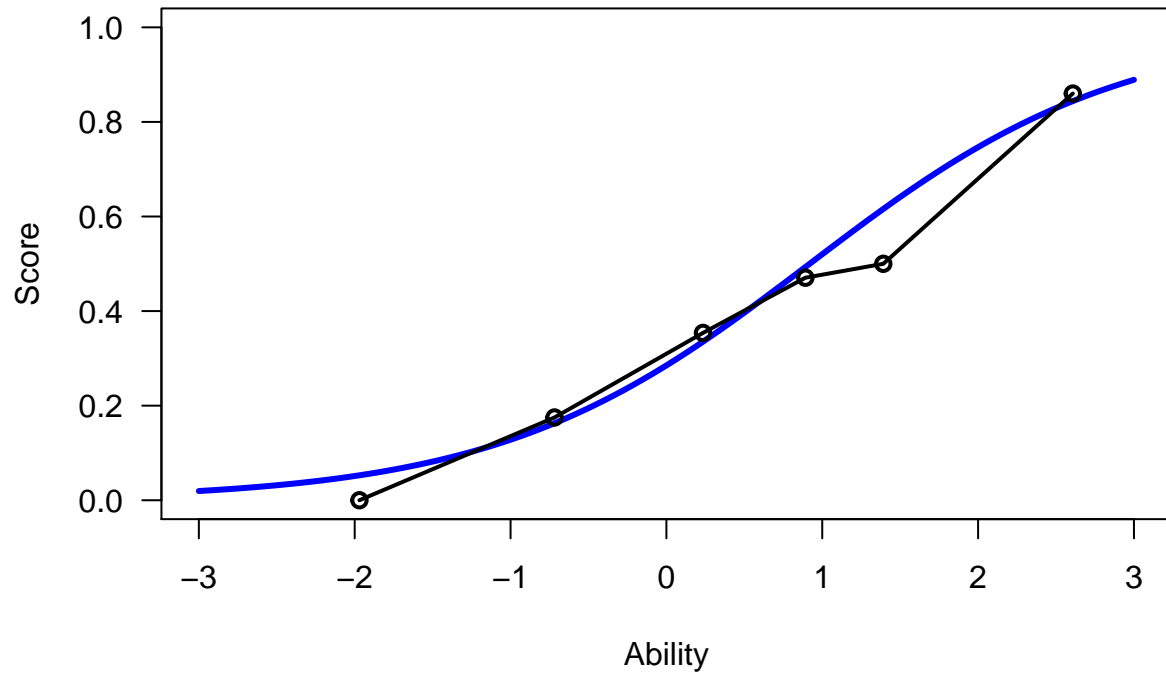
Expected Scores Curve – Item HIs5

Expected Scores Curve – Item HIs6

Expected Scores Curve – Item Hls7

Expected Scores Curve – Item HIs8

Expected Scores Curve – Item HIs9
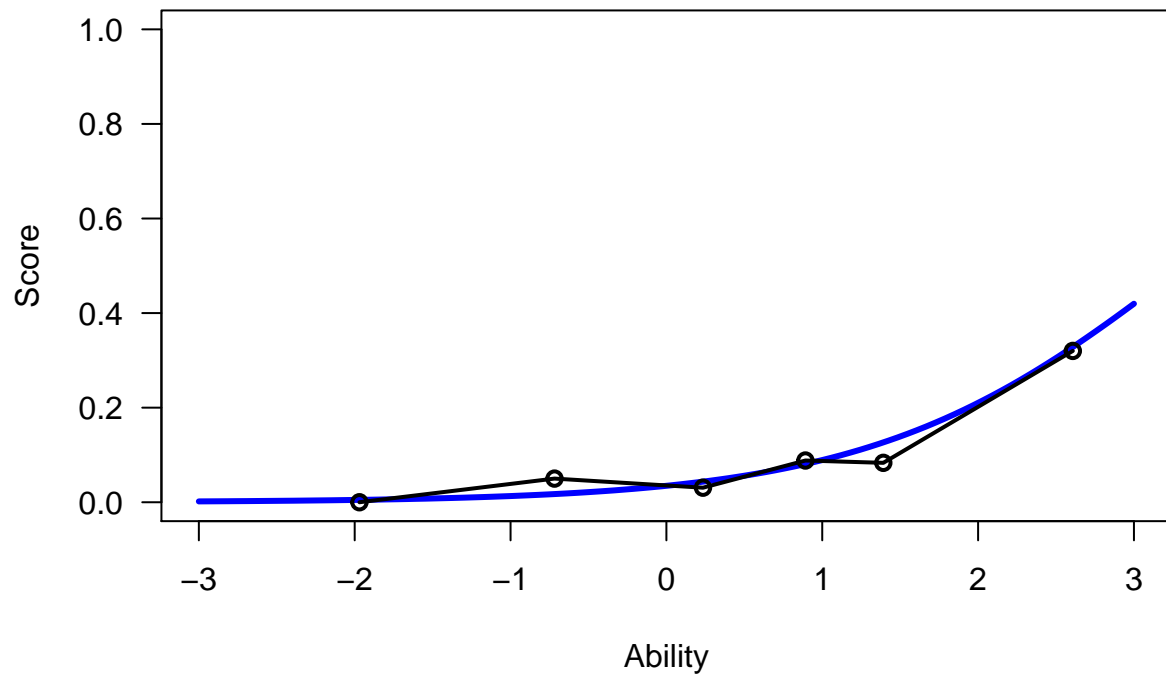
Expected Scores Curve – Item HIs10

Expected Scores Curve – Item Hls11
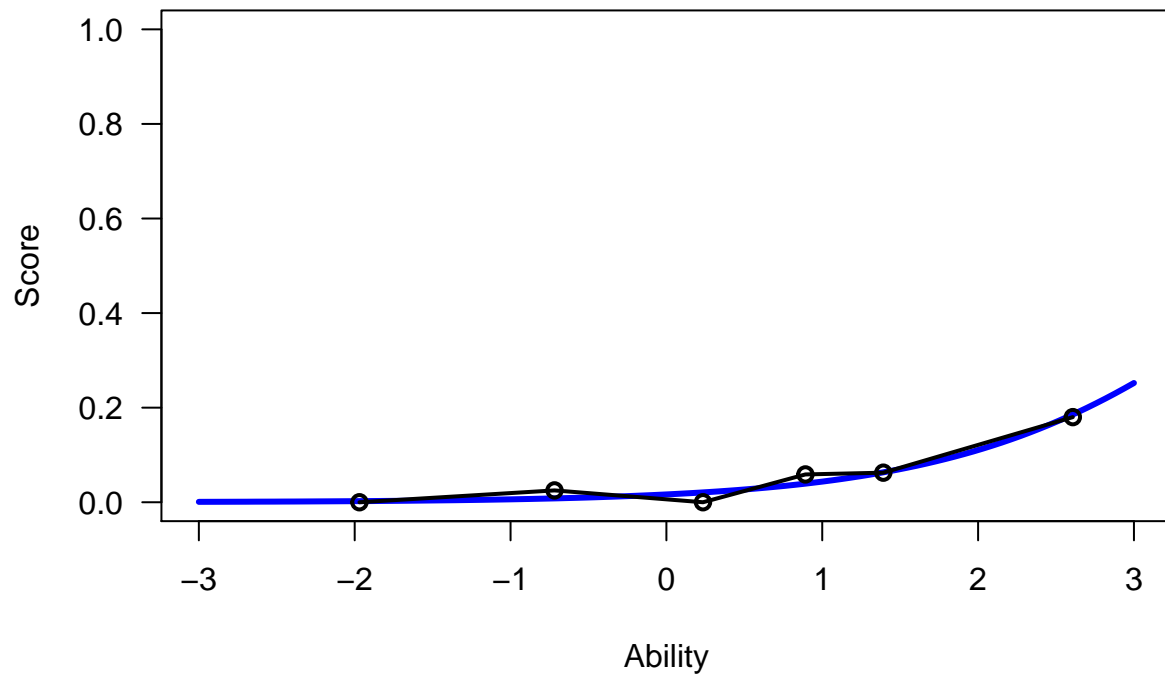
Expected Scores Curve – Item Hls12
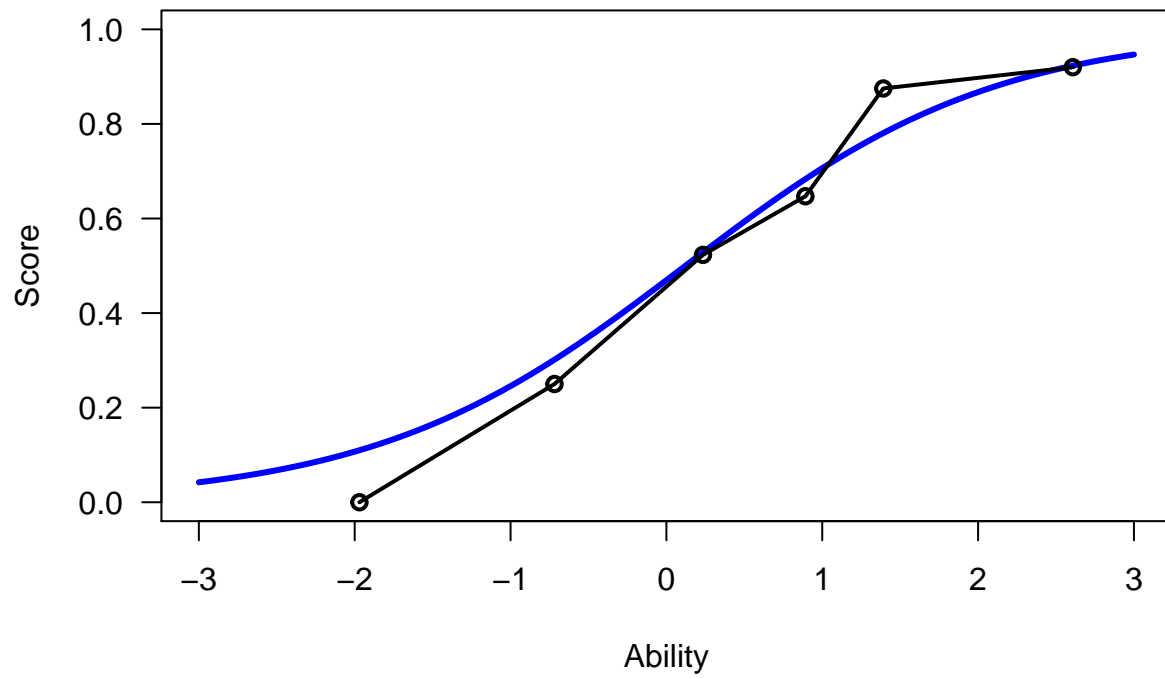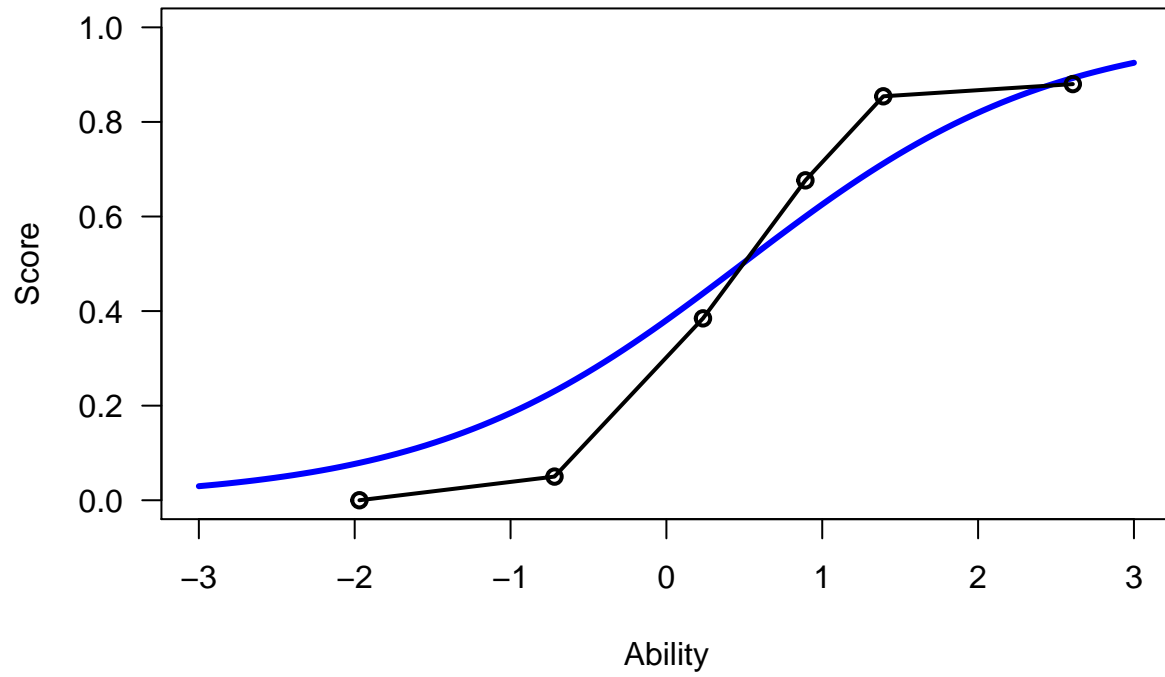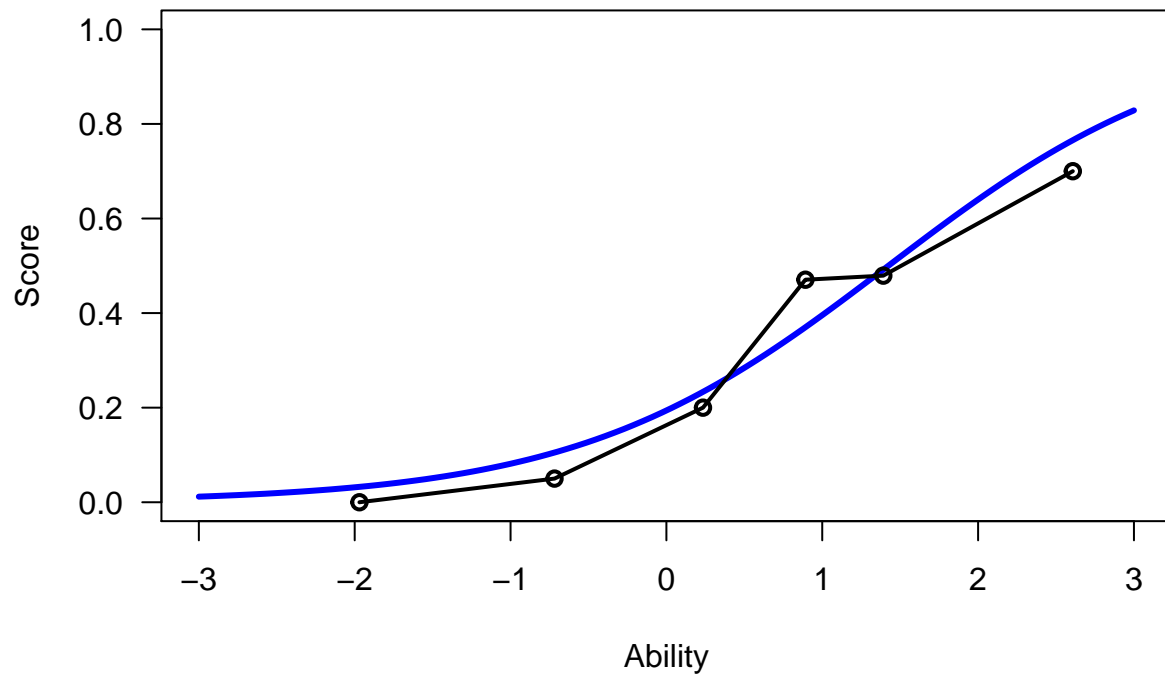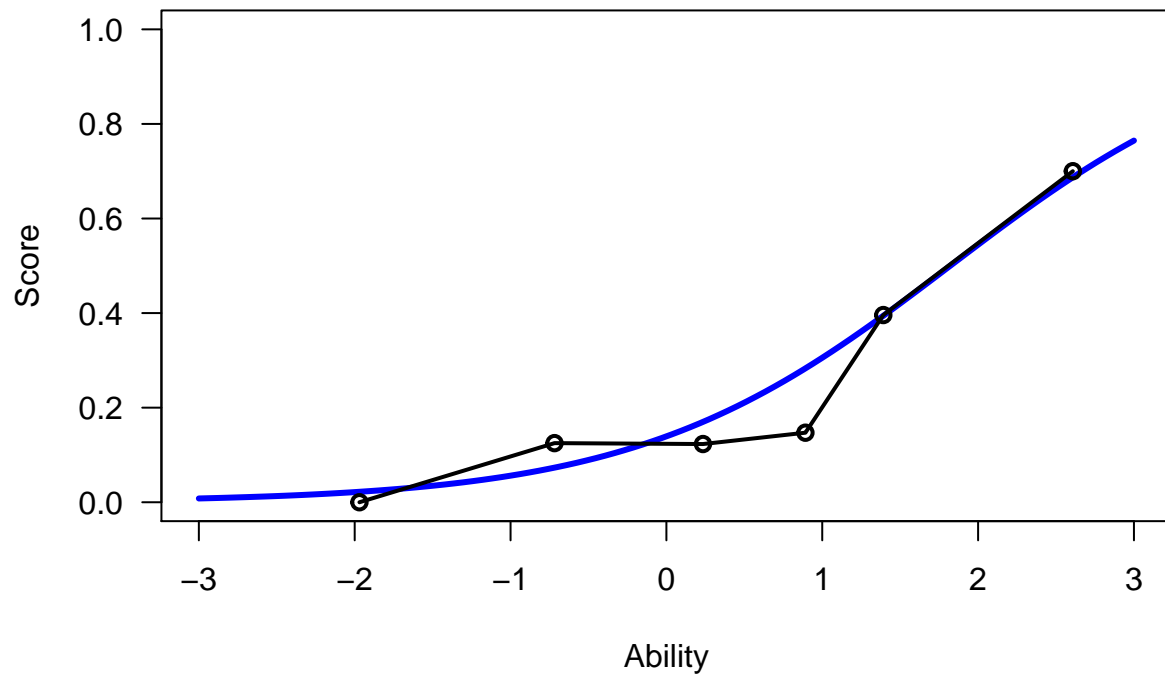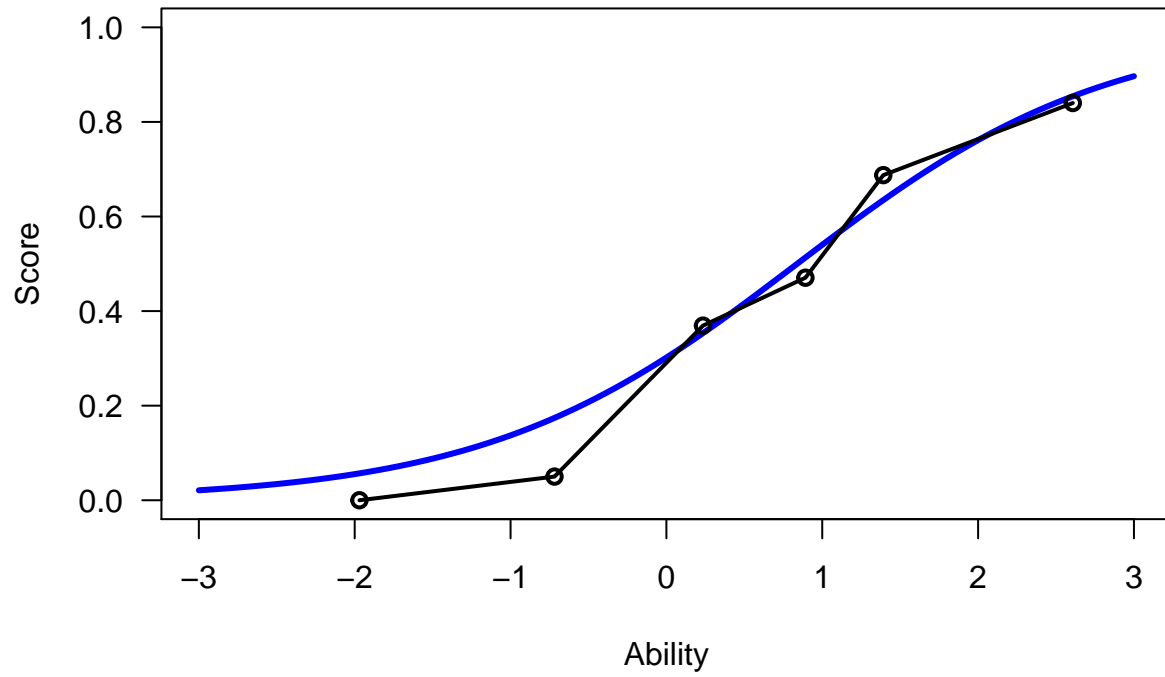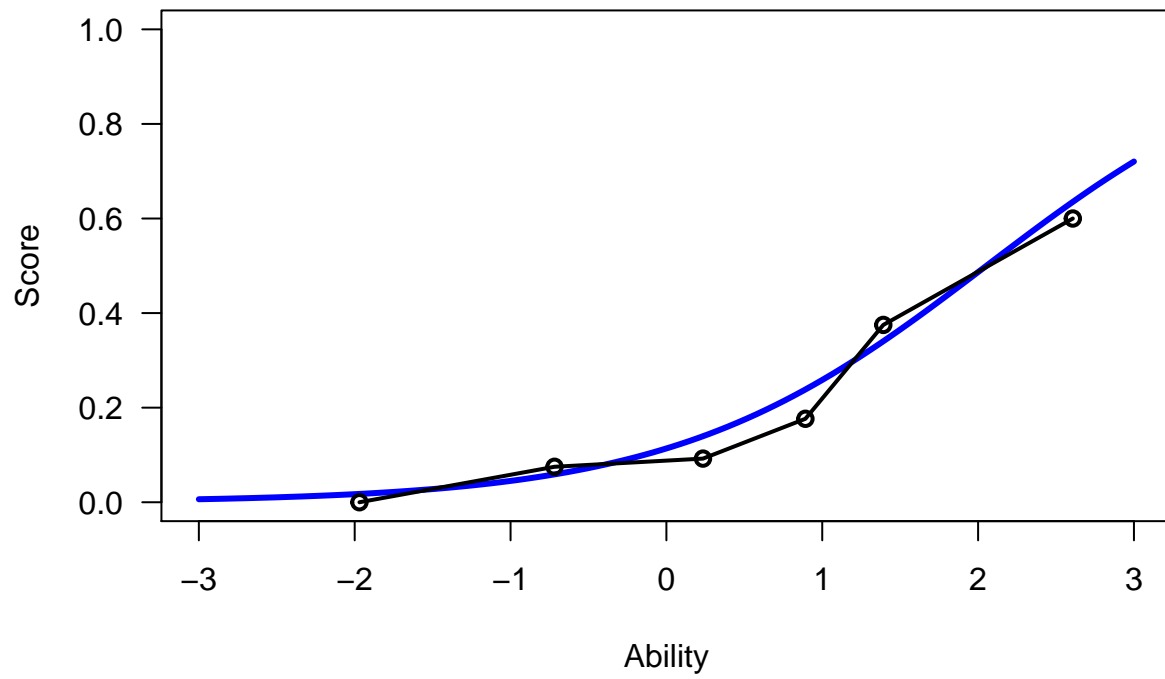
Expected Scores Curve – Item HIs13

Expected Scores Curve – Item HIs14

Expected Scores Curve – Item HIs15

Expected Scores Curve – Item HIs16


```r
hist(ItemDiff)
```

Histogram of ItemDiff
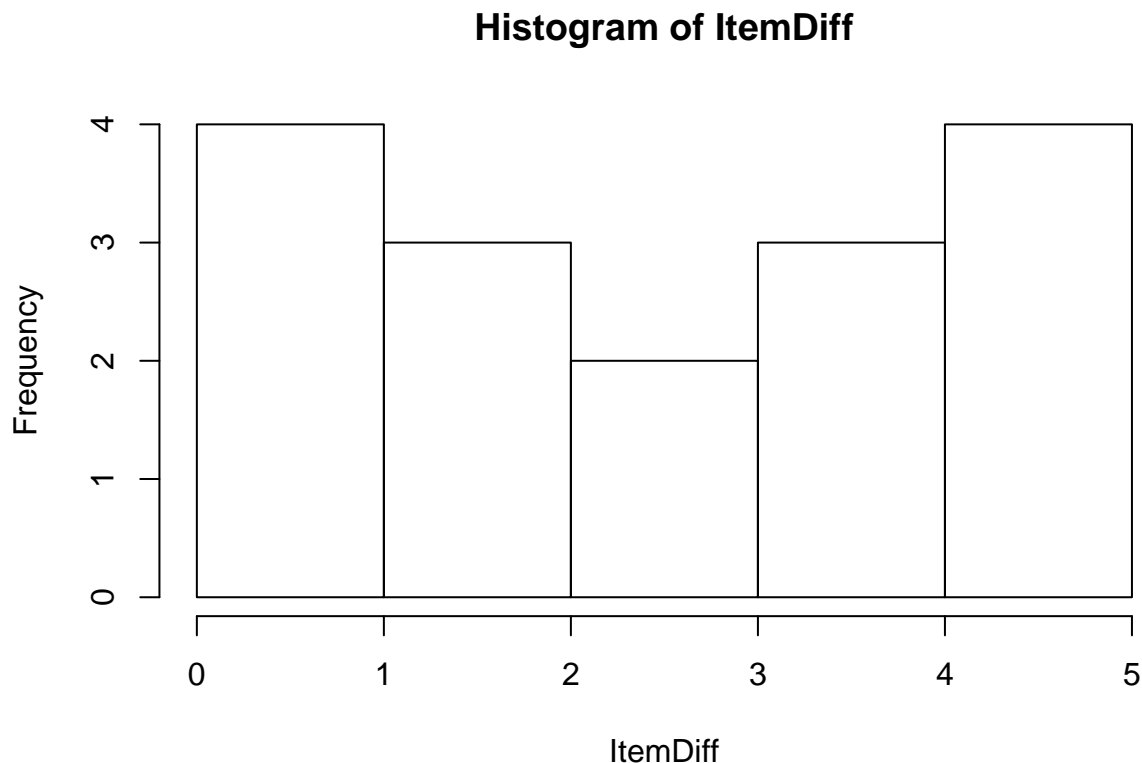

```r
mean(ItemDiff)
```

```
## [1] 2.393147
```

```r
sd(ItemDiff)
```

```
## [1] 1.468218
```

Exercise:

1. Which item is the hardest? The easiest? The closest to the mean? **Hint**: try to use the commands such as `max()`, `min()`.

Person Abilities

Person abilities are also of interest. We can look at the person side of the model by computing person abilities. Compute person abilities using the `tam.wle` function and assign to an object called `Abil`. Extract person abilities (θ_p) from `Abil` and create an object in the **environment** called `PersonAbility` which will essentially be a column vector. **Note**: You may want more information than this at times (such as standard errors) so you may not always want to subset this way.

```r
#generates a data frame
Abil <- tam.wle(mod1)
```

```
## Iteration in WLE/MLE estimation  1   | Maximal change  2.5494
## Iteration in WLE/MLE estimation  2   | Maximal change  2.2203
## Iteration in WLE/MLE estimation  3   | Maximal change  0.4122
## Iteration in WLE/MLE estimation  4   | Maximal change  0.0234
## Iteration in WLE/MLE estimation  5   | Maximal change  0.003
```

```
## Iteration in WLE/MLE estimation   6    | Maximal change   4e-04
## Iteration in WLE/MLE estimation   7    | Maximal change   1e-04
## ----
##   WLE Reliability= 0.615
```

See the first few rows of Abil. Notice you get:

1. `pid`: person id assigned by TAM.
2. `N.items`: Number of items the person was given (this becomes interesting when you have linked test forms where students may not all see the same number of items)
3. `PersonScores`: Number of items the student got right/selected an option for in the survey case.
4. `PersonMax`: Max that person could have gotten right/selected an option for
5. `theta`: estimated person ability
6. `error`: estimated measurement error
7. `WLE.rel`: estimated person seperation reliability.

```
head(Abil)
```

```
# or
```

```
View(Abil)
```

```
## Object of class 'tam.wle'
## Call: tam.wle(tamobj = mod1)
##
##   WLEs for 317 observations and 1 dimension
##
##   WLE Reliability=0.615
##   Average error variance=1.012
##   WLE mean=0.178
##   WLE variance=2.629
```

The column in the `Abil` data.frame corresponding to person estimates is the `theta` column. Pull out the ability estimates, theta, column if you would like, though, this creates a list. This makes it a little easier for a few basic tasks below.

```
PersonAbility <- Abil$theta
```

```
# Only the first 20 shown
PersonAbility
```

```
##    [1] -0.04587787  2.01460924 -0.04587787  0.46070913  1.28373867
##    [6]  1.28373867 -1.96956632  0.46070913  1.28373867  2.01460924
##   [11] -0.71791902  1.65424265  2.73309925  2.37233153 -1.96956632
##   [16]  0.46070913 -1.96956632 -1.96956632 -0.04587787 -1.96956632
##   [21]  1.28373867  0.89201500 -0.71791902 -0.71791902  1.28373867
##   [26]  3.10211354  0.46070913 -0.04587787  1.28373867  0.46070913
##   [31]  0.89201500 -1.96956632 -1.96956632 -0.71791902  1.65424265
##   [36] -0.04587787 -0.71791902 -1.96956632 -1.96956632  0.46070913
##   [41] -0.04587787  2.73309925 -1.96956632 -1.96956632  0.89201500
##   [46] -1.96956632 -0.71791902 -1.96956632  0.89201500 -1.96956632
##   [51] -1.96956632 -1.96956632 -0.71791902 -1.96956632 -0.71791902
##   [56]  0.89201500  4.86023062  3.48553489  0.46070913 -1.96956632
##   [61]  1.28373867  0.46070913 -1.96956632  1.28373867  0.89201500
##   [66] -1.96956632  0.89201500  0.89201500  2.01460924 -1.96956632
##   [71] -1.96956632  1.28373867 -1.96956632 -1.96956632 -0.04587787
##   [76] -1.96956632  1.28373867  0.46070913 -1.96956632  1.65424265
```

```
## [81]   1.28373867 -1.96956632  0.89201500 -1.96956632  3.10211354
## [86]   1.28373867  0.89201500  0.89201500 -0.04587787  1.28373867
## [91]   0.46070913 -1.96956632  4.33952720  0.46070913 -1.96956632
## [96]   0.46070913 -1.96956632 -1.96956632  2.01460924  0.89201500
## [101]  3.48553489 -1.96956632  1.65424265  0.46070913  1.65424265
## [106] -1.96956632 -0.04587787 -0.71791902  3.10211354 -0.71791902
## [111]  1.65424265  0.89201500  0.46070913  6.80308915 -0.04587787
## [116] -1.96956632  0.46070913  1.28373867  2.01460924 -0.04587787
## [121]  0.89201500  1.28373867 -0.04587787 -1.96956632  1.65424265
## [126] -0.04587787  0.89201500 -0.71791902 -1.96956632  0.46070913
## [131] -0.71791902  2.01460924  0.46070913  0.89201500  2.73309925
## [136] -0.71791902 -1.96956632 -1.96956632  2.73309925 -0.71791902
## [141] -0.04587787 -0.71791902 -1.96956632  0.46070913  1.65424265
## [146]  1.28373867  1.65424265  1.28373867  0.46070913 -1.96956632
## [151]  2.01460924 -1.96956632  2.01460924 -1.96956632 -0.71791902
## [156] -0.04587787 -1.96956632  0.46070913 -1.96956632 -0.71791902
## [161] -1.96956632  1.28373867 -0.71791902 -1.96956632  0.89201500
## [166] -1.96956632  0.46070913  0.89201500  0.46070913  0.89201500
## [171] -1.96956632 -1.96956632  0.46070913  3.48553489  0.89201500
## [176] -0.71791902 -1.96956632  0.89201500 -0.71791902  1.28373867
## [181]  2.01460924  4.33952720 -1.96956632 -1.96956632  2.73309925
## [186] -1.96956632  2.37233153  0.46070913  2.01460924 -1.96956632
## [191]  2.01460924 -0.71791902  1.65424265 -1.96956632  0.89201500
## [196] -0.04587787 -1.96956632  0.89201500  0.46070913  2.73309925
## [201] -0.04587787 -0.71791902  1.28373867  1.65424265  0.46070913
## [206]  2.01460924 -1.96956632 -0.71791902 -1.96956632  2.37233153
## [211]  1.65424265  2.01460924  2.01460924  0.89201500 -0.71791902
## [216] -1.96956632  0.46070913 -0.04587787  3.10211354 -1.96956632
## [221] -0.71791902  0.46070913  1.28373867 -1.96956632 -0.71791902
## [226]  0.89201500  2.01460924 -1.96956632 -0.71791902  0.46070913
## [231]  0.46070913 -1.96956632  0.89201500  0.89201500 -1.96956632
## [236]  0.46070913 -0.71791902 -0.04587787 -1.96956632 -0.04587787
## [241] -1.96956632 -1.96956632 -1.96956632 -1.96956632  2.01460924
## [246] -0.71791902 -0.71791902  1.28373867  1.28373867 -1.96956632
## [251] -0.04587787  1.65424265  1.28373867 -0.04587787 -0.71791902
## [256]  1.28373867  1.28373867 -0.71791902  2.01460924  1.28373867
## [261]  2.01460924  2.01460924 -0.71791902 -1.96956632 -1.96956632
## [266] -1.96956632  0.89201500  2.73309925  0.46070913  2.37233153
## [271] -0.71791902 -0.04587787  2.73309925 -0.04587787  2.73309925
## [276]  2.01460924 -1.96956632  0.89201500  0.89201500  0.46070913
## [281] -1.96956632  2.37233153  1.28373867 -0.71791902 -1.96956632
## [286]  1.65424265 -1.96956632  0.46070913  2.01460924 -1.96956632
## [291] -0.71791902 -0.71791902 -0.71791902 -0.04587787 -0.04587787
## [296]  0.89201500  2.01460924 -0.71791902  1.28373867  1.28373867
## [301] -1.96956632  0.46070913  0.89201500 -0.04587787 -0.04587787
## [306]  2.01460924  2.37233153  1.28373867  0.46070913  0.89201500
## [311]  1.28373867  0.89201500 -0.71791902  1.28373867  1.28373867
## [316] -0.04587787  2.01460924
```

You can export those estimated abilites to a .csv to save (you can also save directly in R, if you need to).

```
write.csv(Abil,"HLSmod1_thetas.csv")
```

You can find the CSV file in your `Working Directory`. If you need help finding where your `working directory` is:

```
getwd()
```

[1] "C:/Users/katzd/Desktop/Rprojects/DBER_plain_biome/DBER_Rasch"

Descriptives for person ability

```
hist(PersonAbility)
```

Histogram of PersonAbility
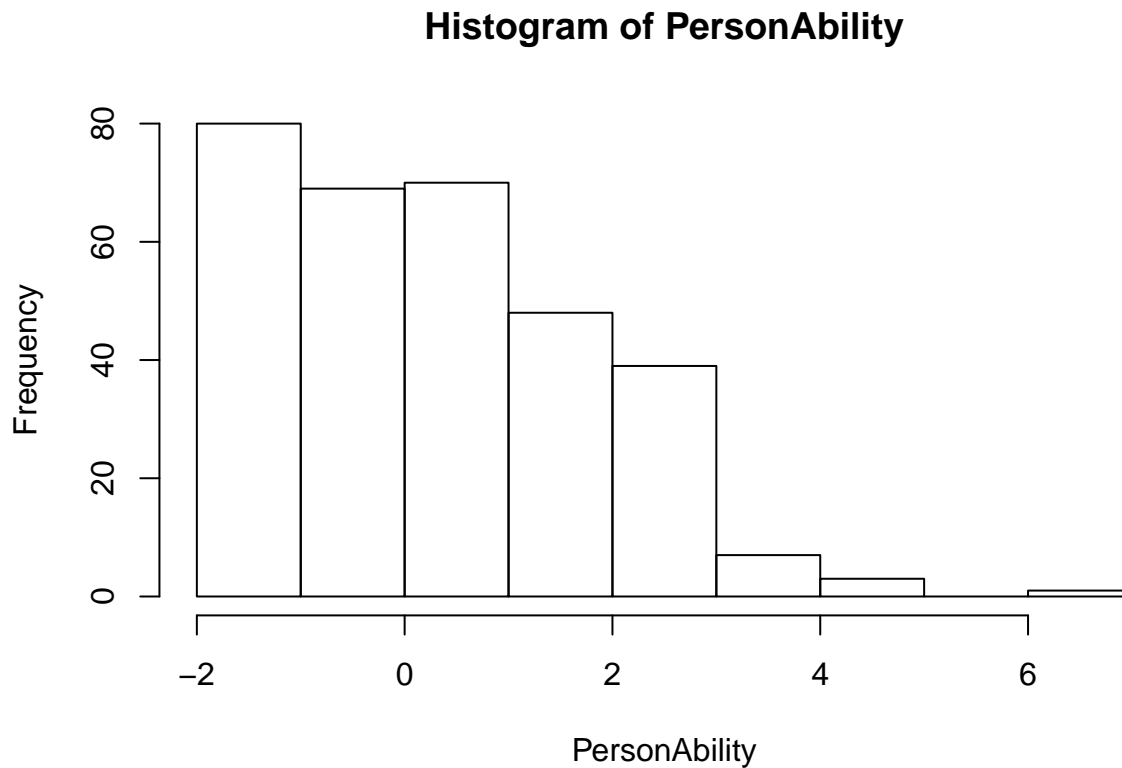

```
mean(PersonAbility)
```

[1] 0.178227

```
sd(PersonAbility)
```

[1] 1.621543

Wright Map

To visualize the relationship between item difficulty and person ability distributions, call the WrightMap package installed previously. We'll generate a simple WrightMap. We'll clean it up a little bit by removing some elements

```
library(WrightMap)
IRT.WrightMap(mod1)
```

Wright Map


```
IRT.WrightMap(mod1, show.thr.lab=FALSE)
```

Wright Map

Exercise:

1. Are the items appropriately targeted to the ability level of the population?
2. Why do you think?

Item Fit

Let's find out if the data fit the model. Use the `tam.fit` function to compute fit statistics, then display.
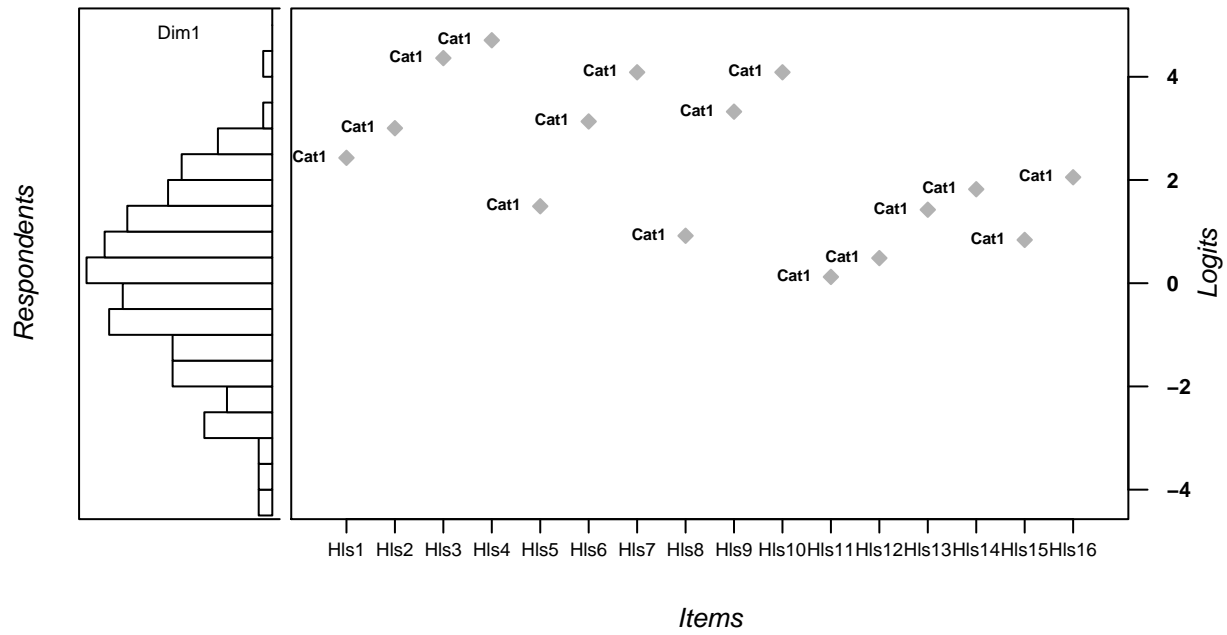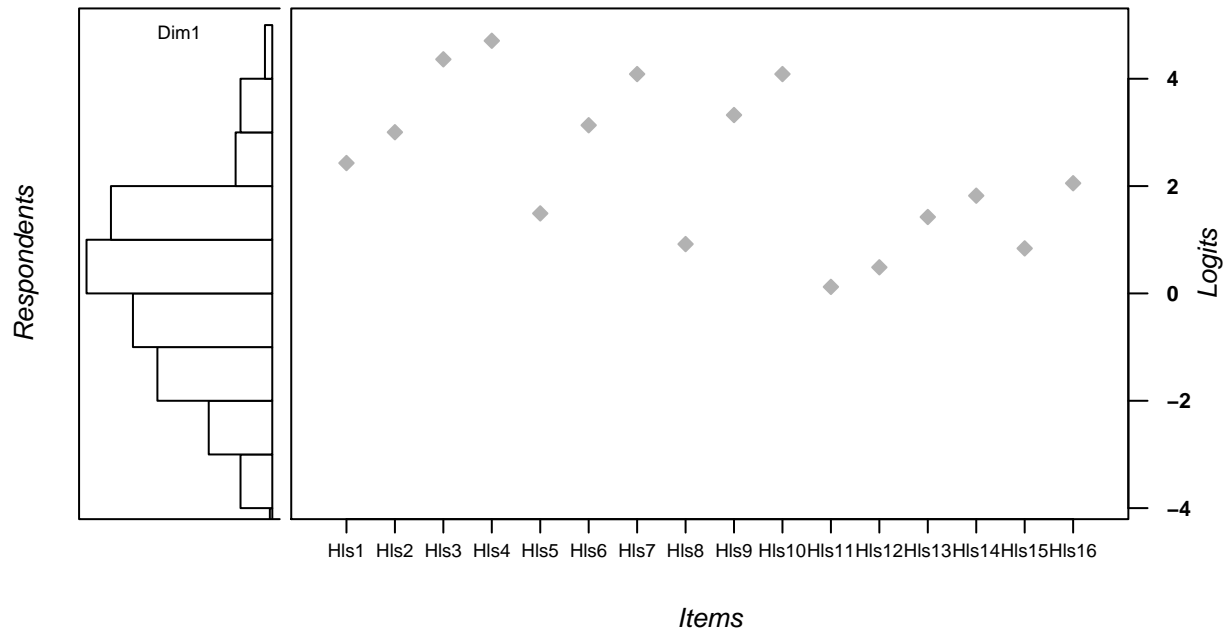
```
fit <- tam.fit(mod1)
```

```
## Item fit calculation based on 100 simulations
## |**********|
## |----------|
```

```
View(fit$itemfit)
```

```
##    parameter    Outfit   Outfit_t    Outfit_p Outfit_pholm     Infit
## 1       Hls1 1.3542455  2.9429756 0.003250741  0.048761108 1.1307487
## 2       Hls2 1.1958983  1.1756804 0.239722627  1.000000000 1.0769183
## 3       Hls3 1.2792558  0.8835458 0.376941446  1.000000000 0.9505660
## 4       Hls4 0.5266458 -1.9306308 0.053528727  0.535287266 0.8736428
## 5       Hls5 0.9009455 -1.4654654 0.142793987  1.000000000 0.9843233
## 6       Hls6 0.9413935 -0.5296596 0.596347981  1.000000000 0.9865001
## 7       Hls7 0.5780420 -2.2449925 0.024768615  0.297223375 0.9401685
## 8       Hls8 1.0245159  0.3193182 0.749485222  1.000000000 1.0568424
## 9       Hls9 1.5880030  2.7823889 0.005396033  0.075544464 1.0011186
## 10     Hls10 1.4739333  1.3978193 0.162167333  1.000000000 1.0804472
```

```
## 11      Hls11 0.8909137 -1.7916485 0.073189288  0.658703590 0.9242078
## 12      Hls12 0.7856933 -3.6458822 0.000266476  0.004263616 0.8671026
## 13      Hls13 0.9122849 -1.2918799 0.196398738  1.000000000 1.0284065
## 14      Hls14 1.2171900  2.0552618 0.039853723  0.438390951 1.0244919
## 15      Hls15 0.8402058 -2.6496143 0.008058370  0.104758807 0.9569603
## 16      Hls16 1.0569807  0.3888759 0.697367950  1.000000000 1.0258899
##          Infit_t      Infit_p Infit_pholm
## 1   1.24753293 0.21220215   1.0000000
## 2    0.59971826 0.54869401   1.0000000
## 3   -0.12186065 0.90300938   1.0000000
## 4   -0.35295407 0.72412288   1.0000000
## 5   -0.20270755 0.83936362   1.0000000
## 6   -0.06118877 0.95120887   1.0000000
## 7   -0.21021174 0.83350241   1.0000000
## 8    0.85874435 0.39048158   1.0000000
## 9    0.05251025 0.95812213   1.0000000
## 10   0.42109330 0.67368695   1.0000000
## 11 -1.20114125 0.22969641   1.0000000
## 12 -2.17102358 0.02992939   0.4788702
## 13   0.39737453 0.69109129   1.0000000
## 14   0.31836383 0.75020897   1.0000000
## 15 -0.66485738 0.50614172   1.0000000
## 16   0.31114990 0.75568667   1.0000000
```

Exercise:

1. Look at the `Wright Map` and the histograms of person abilities (θ_p) and item difficulties (δ_i). Do you think this instrument is well-targeted for this sample? 2. How might it be optimized?
2. Relative to other items, which item fit our model the worst?

This concludes the planned instruction in the Rasch model for our workshop. However, we've provided working code for a few other concepts in which you might be interested, including using the Rasch model with polytomous items and with multidimensional models.

Polytomous Items

Polytymous item types (anything with a rating Scale)

We can use the Rasch Partial Credit Model (PCM) to look at polytomous data too. We'll start by bringing in the polytomous items from the survey. Note that TAM needs the bottom category to be coded as 0, so you may need to recode.

```
hls2 <- read.csv("hls_poly_scale.csv")
```

```
head(hls2)
```

```
##   Hls1 Hls2 Hls3 Hls4 Hls5 Hls6 Hls7 Hls8 Hls9 Hls10 Hls11 Hls12 Hls13
## 1    1    1    1    0    1    1    0    2    1     1     2     1     1
## 2    2    1    1    1    2    1    1    2    1     1     2     2     2
## 3    0    1    1    1    1    1    1    2    1     0     1     2     1
## 4    1    1    0    0    2    1    0    1    0     0     2     1     1
## 5    1    1    0    0    1    0    0    2    0     0     2     2     2
## 6    1    1    1    1    2    1    1    1    1     0     2     2     2
##   Hls14 Hls15 Hls16
## 1     0     1     1
## 2     1     1     2
```

```
## 3      1    1      1
## 4      1    2      1
## 5      1    1      2
## 6      1    2      1
```

View(hls2)

TAM will automatically run the PCM when our data is polytomous. There are other model-types for polytomous data such as the rating scale model. This may be more appropriate for Likert-type items. For more information, read TAM documentation or see the reference list (Bond & Fox, 2007)

mod2 <- **tam**(hls2)

summary(mod2)

```
## -----------------------------------------------------------------
## TAM 3.3-10 (2019-08-23 13:42:07)
## R version 3.5.2 (2018-12-20) x86_64, mingw32 | nodename=LAPTOP-K7402PLE | login=katzd
##
## Date of Analysis: 2020-01-08 11:44:42
## Time difference of 0.1629992 secs
## Computation time: 0.1629992
##
## Multidimensional Item Response Model in TAM
##
## IRT Model: 1PL
## Call:
## tam.mml(resp = resp)
##
## -----------------------------------------------------------------
## Number of iterations = 57
## Numeric integration with 21 integration points
##
## Deviance = 8371.25
## Log likelihood = -4185.63
## Number of persons = 317
## Number of persons used = 317
## Number of items = 16
## Number of estimated parameters = 49
##      Item threshold parameters = 48
##      Item slope parameters = 0
##      Regression parameters = 0
##      Variance/covariance parameters = 1
##
## AIC = 8469  | penalty = 98      | AIC=-2*LL + 2*p
## AIC3 = 8518  | penalty = 147     | AIC3=-2*LL + 3*p
## BIC = 8653  | penalty = 282.19    | BIC=-2*LL + log(n)*p
## aBIC = 8497  | penalty = 126.15    | aBIC=-2*LL + log((n-2)/24)*p  (adjusted BIC)
## CAIC = 8702  | penalty = 331.19    | CAIC=-2*LL + [log(n)+1]*p  (consistent AIC)
## AICc = 8488  | penalty = 116.35    | AICc=-2*LL + 2*p + 2*p*(p+1)/(n-p-1)  (bias corrected AIC)
##
## -----------------------------------------------------------------
## EAP Reliability
## [1] 0.914
## -----------------------------------------------------------------
## Covariances and Variances
```

```
##       [,1]
## [1,] 2.615
## ----------------------------------------------------------------
## Correlations and Standard Deviations (in the diagonal)
##       [,1]
## [1,] 1.617
## ----------------------------------------------------------------
## Regression Coefficients
##       [,1]
## [1,]    0
## ----------------------------------------------------------------
## Item Parameters -A*Xsi
##       item   N     M xsi.item AXsi_.Cat1 AXsi_.Cat2 AXsi_.Cat3 B.Cat1.Dim1
## 1    Hls1 317 0.978    1.427     -1.846      0.452      4.282           1
## 2    Hls2 317 0.874    2.074     -1.502      1.263      6.221           1
## 3    Hls3 317 0.634    2.903     -0.381      3.581      8.709           1
## 4    Hls4 317 0.530    2.809      0.176      4.500      8.428           1
## 5    Hls5 317 1.091    1.198     -1.684     -0.302      3.595           1
## 6    Hls6 317 0.830    1.818     -1.155      1.784      5.455           1
## 7    Hls7 317 0.615    2.455     -0.166      3.587      7.364           1
## 8    Hls8 317 1.237    0.781     -2.136     -1.239      2.342           1
## 9    Hls9 317 0.644    2.098     -0.008      2.982      6.295           1
## 10 Hls10 317 0.615    2.630     -0.183      3.511      7.890           1
## 11 Hls11 317 1.413    0.325     -2.106     -1.934      0.974           1
## 12 Hls12 317 1.338    0.512     -2.318     -1.805      1.537           1
## 13 Hls13 317 1.136    1.162     -2.127     -0.789      3.487           1
## 14 Hls14 317 1.063    1.070     -1.762      0.000      3.209           1
## 15 Hls15 317 1.243    0.792     -2.043     -1.232      2.376           1
## 16 Hls16 317 1.000    1.434     -1.629      0.278      4.303           1
##    B.Cat2.Dim1 B.Cat3.Dim1
## 1            2           3
## 2            2           3
## 3            2           3
## 4            2           3
## 5            2           3
## 6            2           3
## 7            2           3
## 8            2           3
## 9            2           3
## 10           2           3
## 11           2           3
## 12           2           3
## 13           2           3
## 14           2           3
## 15           2           3
## 16           2           3
##
## Item Parameters Xsi
##              xsi se.xsi
## Hls1_Cat1 -1.846  0.177
## Hls1_Cat2  2.298  0.174
## Hls1_Cat3  3.830  0.464
## Hls2_Cat1 -1.502  0.164
## Hls2_Cat2  2.765  0.200
```

```
## Hls2_Cat3   4.958  0.780
## Hls3_Cat1  -0.381  0.138
## Hls3_Cat2   3.962  0.321
## Hls3_Cat3   5.127  1.117
## Hls4_Cat1   0.176  0.133
## Hls4_Cat2   4.325  0.378
## Hls4_Cat3   3.927  0.853
## Hls5_Cat1  -1.684  0.178
## Hls5_Cat2   1.382  0.147
## Hls5_Cat3   3.897  0.394
## Hls6_Cat1  -1.155  0.154
## Hls6_Cat2   2.939  0.211
## Hls6_Cat3   3.671  0.519
## Hls7_Cat1  -0.166  0.136
## Hls7_Cat2   3.753  0.295
## Hls7_Cat3   3.776  0.689
## Hls8_Cat1  -2.136  0.199
## Hls8_Cat2   0.897  0.139
## Hls8_Cat3   3.581  0.324
## Hls9_Cat1  -0.008  0.136
## Hls9_Cat2   2.990  0.229
## Hls9_Cat3   3.313  0.482
## Hls10_Cat1 -0.183  0.136
## Hls10_Cat2  3.695  0.292
## Hls10_Cat3  4.379  0.822
## Hls11_Cat1 -2.106  0.210
## Hls11_Cat2  0.173  0.138
## Hls11_Cat3  2.907  0.235
## Hls12_Cat1 -2.318  0.212
## Hls12_Cat2  0.513  0.136
## Hls12_Cat3  3.342  0.282
## Hls13_Cat1 -2.127  0.194
## Hls13_Cat2  1.339  0.144
## Hls13_Cat3  4.276  0.449
## Hls14_Cat1 -1.761  0.178
## Hls14_Cat2  1.762  0.155
## Hls14_Cat3  3.208  0.331
## Hls15_Cat1 -2.042  0.197
## Hls15_Cat2  0.810  0.138
## Hls15_Cat3  3.609  0.323
## Hls16_Cat1 -1.629  0.172
## Hls16_Cat2  1.907  0.160
## Hls16_Cat3  4.025  0.457
##
## Item Parameters in IRT parameterization
##      item alpha  beta tau.Cat1 tau.Cat2 tau.Cat3
## 1    Hls1     1 1.427   -3.274    0.871    2.403
## 2    Hls2     1 2.074   -3.575    0.691    2.885
## 3    Hls3     1 2.903   -3.284    1.059    2.224
## 4    Hls4     1 2.809   -2.633    1.515    1.118
## 5    Hls5     1 1.198   -2.883    0.184    2.699
## 6    Hls6     1 1.818   -2.974    1.121    1.853
## 7    Hls7     1 2.455   -2.620    1.299    1.322
## 8    Hls8     1 0.781   -2.917    0.116    2.800
```

```
## 9   Hls9    1 2.098   -2.106    0.891    1.215
## 10 Hls10   1 2.630   -2.814    1.065    1.749
## 11 Hls11   1 0.325   -2.431   -0.152    2.583
## 12 Hls12   1 0.512   -2.830    0.001    2.830
## 13 Hls13   1 1.162   -3.290    0.176    3.113
## 14 Hls14   1 1.070   -2.831    0.692    2.139
## 15 Hls15   1 0.792   -2.835    0.018    2.816
## 16 Hls16   1 1.434   -3.063    0.472    2.590
```

Item Difficulties

Now we'll get item and person characteristics just like before

```
mod2$xsi
```

```
##                    xsi     se.xsi
## Hls1_Cat1  -1.846046710 0.1770841
## Hls1_Cat2   2.298334842 0.1736858
## Hls1_Cat3   3.830385868 0.4635448
## Hls2_Cat1  -1.501715752 0.1640914
## Hls2_Cat2   2.764560868 0.2004621
## Hls2_Cat3   4.958324889 0.7804292
## Hls3_Cat1  -0.380834628 0.1378983
## Hls3_Cat2   3.962213547 0.3205437
## Hls3_Cat3   5.127428610 1.1165768
## Hls4_Cat1   0.175976124 0.1331498
## Hls4_Cat2   4.324558567 0.3775359
## Hls4_Cat3   3.927492901 0.8526291
## Hls5_Cat1  -1.684131262 0.1781274
## Hls5_Cat2   1.381933373 0.1467223
## Hls5_Cat3   3.897482578 0.3943440
## Hls6_Cat1  -1.155351142 0.1543659
## Hls6_Cat2   2.939344680 0.2113184
## Hls6_Cat3   3.671463609 0.5189345
## Hls7_Cat1  -0.165823435 0.1358793
## Hls7_Cat2   3.753328170 0.2949727
## Hls7_Cat3   3.776319530 0.6885511
## Hls8_Cat1  -2.135935885 0.1992731
## Hls8_Cat2   0.896643565 0.1385362
## Hls8_Cat3   3.581083599 0.3235907
## Hls9_Cat1  -0.008019089 0.1360316
## Hls9_Cat2   2.989853095 0.2288433
## Hls9_Cat3   3.313295753 0.4819018
## Hls10_Cat1 -0.183297684 0.1360561
## Hls10_Cat2  3.694746057 0.2921816
## Hls10_Cat3  4.379242422 0.8215605
## Hls11_Cat1 -2.106058995 0.2097751
## Hls11_Cat2  0.172650186 0.1377271
## Hls11_Cat3  2.907183948 0.2353937
## Hls12_Cat1 -2.317865929 0.2117123
## Hls12_Cat2  0.513325662 0.1362435
## Hls12_Cat3  3.342199604 0.2821645
## Hls13_Cat1 -2.127336182 0.1938394
## Hls13_Cat2  1.338677184 0.1444056
## Hls13_Cat3  4.275574749 0.4493420
```

```
## Hls14_Cat1 -1.761463128 0.1777770
## Hls14_Cat2  1.762102813 0.1550751
## Hls14_Cat3  3.208316423 0.3314447
## Hls15_Cat1 -2.042459839 0.1969236
## Hls15_Cat2  0.810466042 0.1380178
## Hls15_Cat3  3.608588261 0.3230206
## Hls16_Cat1 -1.628691156 0.1721913
## Hls16_Cat2  1.906727817 0.1604086
## Hls16_Cat3  4.024764112 0.4574405
```

```
ItemDiff2 <- mod2$xsi$xsi
ItemDiff2
```

```
##  [1] -1.846046710  2.298334842  3.830385868 -1.501715752  2.764560868
##  [6]  4.958324889 -0.380834628  3.962213547  5.127428610  0.175976124
## [11]  4.324558567  3.927492901 -1.684131262  1.381933373  3.897482578
## [16] -1.155351142  2.939344680  3.671463609 -0.165823435  3.753328170
## [21]  3.776319530 -2.135935885  0.896643565  3.581083599 -0.008019089
## [26]  2.989853095  3.313295753 -0.183297684  3.694746057  4.379242422
## [31] -2.106058995  0.172650186  2.907183948 -2.317865929  0.513325662
## [36]  3.342199604 -2.127336182  1.338677184  4.275574749 -1.761463128
## [41]  1.762102813  3.208316423 -2.042459839  0.810466042  3.608588261
## [46] -1.628691156  1.906727817  4.024764112
```

```
#note, if you want to see this in your viewer, you can also use View().
```

Person ability (theta) estimates

```
person.ability.poly <- tam.wle(mod2)
```

```
## Iteration in WLE/MLE estimation  1   | Maximal change  2.6967
## Iteration in WLE/MLE estimation  2   | Maximal change  2.1777
## Iteration in WLE/MLE estimation  3   | Maximal change  0.368
## Iteration in WLE/MLE estimation  4   | Maximal change  0.0135
## Iteration in WLE/MLE estimation  5   | Maximal change  3e-04
## Iteration in WLE/MLE estimation  6   | Maximal change  0
## ----
##  WLE Reliability= 0.9
```

```
WLEestimates.poly <- person.ability.poly$theta
```

```
head(WLEestimates.poly)
```

```
## [1]  0.07670224  1.74460893  0.29326740 -0.14106713  0.07670224  1.14369491
```

Item fit statistics

```
Fit.poly <- tam.fit(mod2)
```

```
## Item fit calculation based on 100 simulations
## |*********|
## |----------|
```
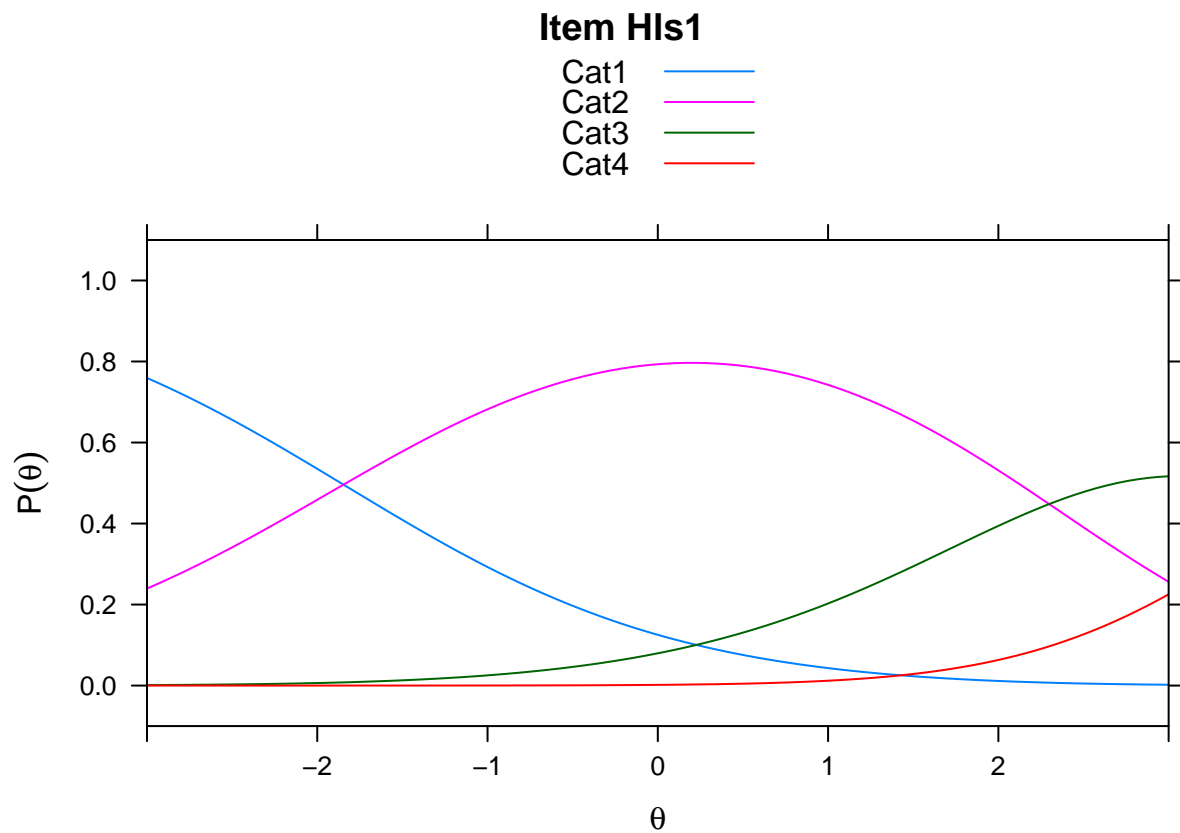
```
Fit.poly$itemfit
```

```r
kable(Fit.poly$itemfit)
```

parameter	Outfit	Outfit_t	Outfit_p	Outfit_pholm	Infit	Infit_t	Infit_p	Infit_pho
Hls1__Cat1	3.1007634	11.3001508	0.0000000	0.0000000	1.0352323	0.3487245	0.7272961	
Hls1__Cat2	4.6240034	16.1090708	0.0000000	0.0000000	1.1163466	1.1305639	0.2582387	
Hls1__Cat3	6248.0019696	94.8923820	0.0000000	0.0000000	0.9629939	0.0200439	0.9840083	
Hls2__Cat1	24.4525937	49.8016152	0.0000000	0.0000000	1.0730547	0.7600367	0.4472326	
Hls2__Cat2	0.9177929	-0.6186671	0.5361357	1.0000000	1.0375272	0.3151812	0.7526241	
Hls2__Cat3	0.7539337	-0.3272716	0.7434624	1.0000000	1.4128023	0.7864336	0.4316135	
Hls3__Cat1	0.9343011	-1.0545509	0.2916308	1.0000000	0.9625829	-0.5803792	0.5616589	
Hls3__Cat2	0.8242510	-0.7174254	0.4731116	1.0000000	0.9092794	-0.2790741	0.7801880	
Hls3__Cat3	0.0148261	-2.4591035	0.0139284	0.3621396	0.8576576	0.0366982	0.9707256	
Hls4__Cat1	0.8597843	-2.6898274	0.0071489	0.2001691	0.9251789	-1.3923342	0.1638212	
Hls4__Cat2	0.6392526	-1.3189716	0.1871786	1.0000000	0.8853431	-0.2839016	0.7764858	
Hls4__Cat3	0.0105119	-3.4187835	0.0006290	0.0213866	0.4148649	-1.0256810	0.3050420	
Hls5__Cat1	0.8407345	-1.6520140	0.0985317	1.0000000	0.8411208	-1.5191467	0.1287256	
Hls5__Cat2	1.4802821	5.6184507	0.0000000	0.0000007	1.0804969	1.1817268	0.2373141	
Hls5__Cat3	1.8249715	1.8853513	0.0593824	1.0000000	1.0745181	0.3245537	0.7455189	
Hls6__Cat1	0.9442274	-0.6886770	0.4910265	1.0000000	0.8727491	-1.5233027	0.1276830	
Hls6__Cat2	1.3050151	1.8245351	0.0680712	1.0000000	1.0169231	0.1580838	0.8743908	
Hls6__Cat3	3.1933872	2.9964944	0.0027310	0.0873930	1.3117089	0.8081612	0.4189978	
Hls7__Cat1	0.8339666	-2.9338879	0.0033475	0.1037710	0.9086193	-1.5603391	0.1186798	
Hls7__Cat2	0.5243376	-2.4840992	0.0129880	0.3506750	0.9025817	-0.3680606	0.7128280	
Hls7__Cat3	0.6784787	-0.9479458	0.3431571	1.0000000	1.1820991	0.4784749	0.6323122	
Hls8__Cat1	0.6201148	-3.5337935	0.0004096	0.0143374	0.8456861	-1.2664570	0.2053495	
Hls8__Cat2	2.2289975	14.0054667	0.0000000	0.0000000	1.0792474	1.3604254	0.1736953	
Hls8__Cat3	1.3745376	1.2057630	0.2279089	1.0000000	0.9514572	-0.1169448	0.9069038	
Hls9__Cat1	0.8820544	-2.0786420	0.0376503	0.8592555	0.9518692	-0.8158483	0.4145869	
Hls9__Cat2	1.4147644	2.1842366	0.0289449	0.7236216	1.0829360	0.5571076	0.5774539	
Hls9__Cat3	0.9649998	-0.2809335	0.7787614	1.0000000	0.9456991	-0.0271491	0.9783408	
Hls10__Cat1	1.0193765	0.3047867	0.7605286	1.0000000	1.0157198	0.2708261	0.7865248	
Hls10__Cat2	116.5378802	41.3241005	0.0000000	0.0000000	1.1434683	0.6691669	0.5033890	
Hls10__Cat3	0.5629843	-0.8766713	0.3806652	1.0000000	1.0302599	0.2485462	0.8037118	
Hls11__Cat1	0.7632463	-2.0818198	0.0373589	0.8592555	0.8582229	-1.1023511	0.2703091	
Hls11__Cat2	0.9248002	-1.3361555	0.1814984	1.0000000	1.0215376	0.3758592	0.7070216	
Hls11__Cat3	2.0014574	4.4032074	0.0000107	0.0003946	0.9998998	0.0477752	0.9618954	
Hls12__Cat1	0.6573307	-3.1138495	0.0018466	0.0609390	0.7245387	-2.2522582	0.0243060	
Hls12__Cat2	1.0537065	0.7994685	0.4240188	1.0000000	0.9895009	-0.1812616	0.8561622	
Hls12__Cat3	3.0087222	5.7805715	0.0000000	0.0000003	1.0379774	0.2406451	0.8098302	
Hls13__Cat1	0.9398563	-0.6099850	0.5418717	1.0000000	0.7677644	-2.0328043	0.0420723	
Hls13__Cat2	1.0069909	0.0837512	0.9332543	1.0000000	1.0516760	0.7957569	0.4261733	
Hls13__Cat3	2.3119772	2.1054498	0.0352522	0.8460521	0.9056078	-0.1421912	0.8869290	
Hls14__Cat1	0.8147351	-1.8331346	0.0667826	1.0000000	0.9123777	-0.8006380	0.4233412	
Hls14__Cat2	2.1786214	10.6091792	0.0000000	0.0000000	1.0741606	0.9475891	0.3433387	
Hls14__Cat3	0.5832513	-2.0305020	0.0423055	0.8884163	0.9724752	-0.0369570	0.9705193	
Hls15__Cat1	0.6930232	-2.8227201	0.0047618	0.1428544	0.8077009	-1.6376298	0.1014989	
Hls15__Cat2	1.1211023	2.0039401	0.0450765	0.9015296	1.0668424	1.1678919	0.2428504	
Hls15__Cat3	4.1818916	6.4484797	0.0000000	0.0000000	0.9623696	-0.0733001	0.9415673	
Hls16__Cat1	1.4931591	4.0212740	0.0000579	0.0020838	0.9485151	-0.4734878	0.6358652	
Hls16__Cat2	1.9373013	8.3760845	0.0000000	0.0000000	1.0612267	0.7287673	0.4661440	
Hls16__Cat3	0.2728508	-2.7942267	0.0052024	0.1508696	0.8233994	-0.3877356	0.6982117	

Item characteristic curves (but now as thresholds).

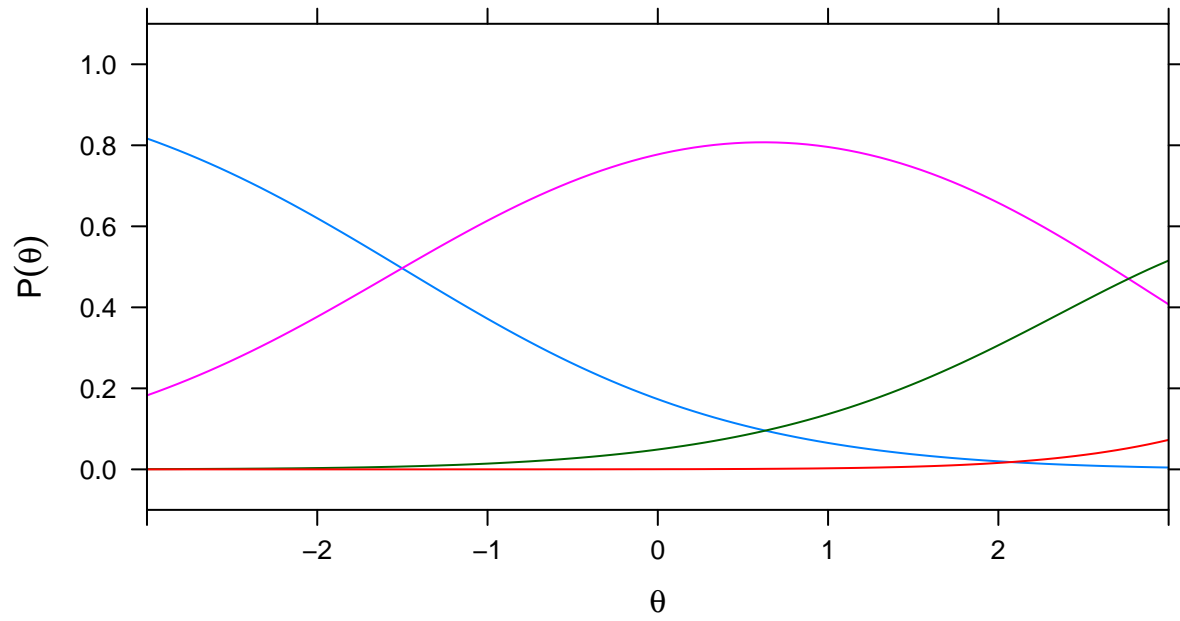There are item characteristic curves (ICCs) for each item choice

```
tthresh.poly <- tam.threshold(mod2)
plot(mod2, type = "items", export=F)
```

```
## Iteration in WLE/MLE estimation  1   | Maximal change  2.6967
## Iteration in WLE/MLE estimation  2   | Maximal change  2.1777
## Iteration in WLE/MLE estimation  3   | Maximal change  0.368
## Iteration in WLE/MLE estimation  4   | Maximal change  0.0135
## Iteration in WLE/MLE estimation  5   | Maximal change  3e-04
## Iteration in WLE/MLE estimation  6   | Maximal change  0
## ----
##  WLE Reliability= 0.9
```

Item HIs2

Cat1
Cat2
Cat3
Cat4
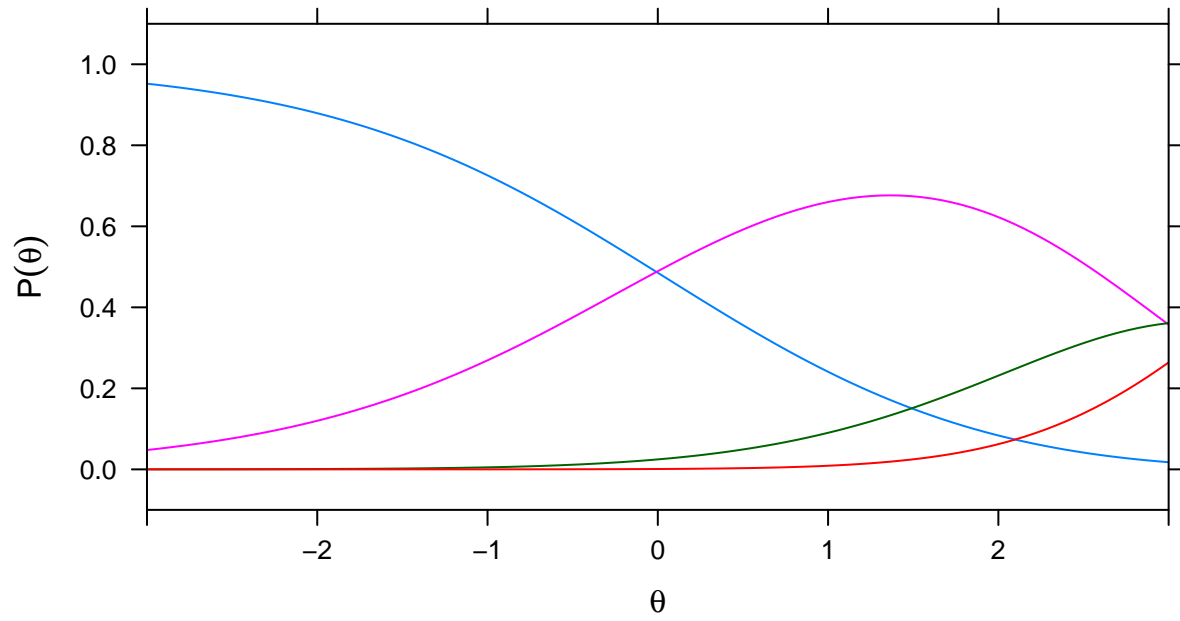
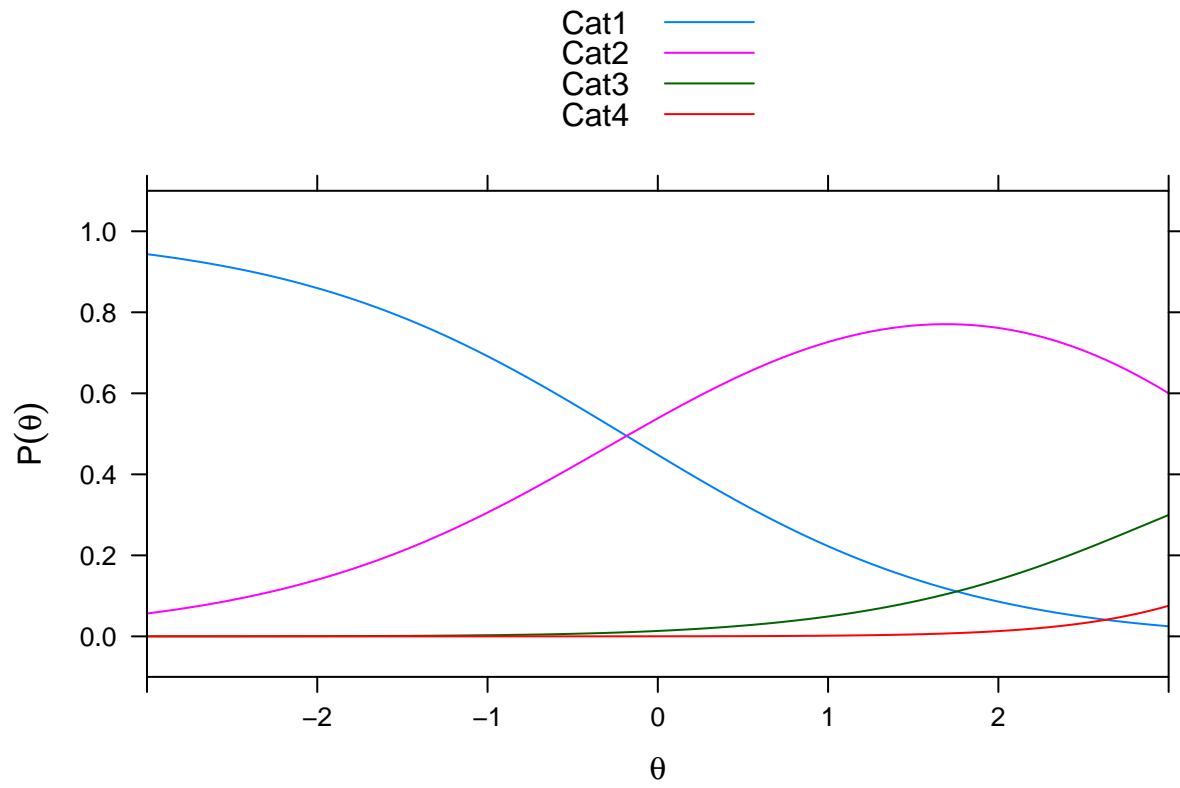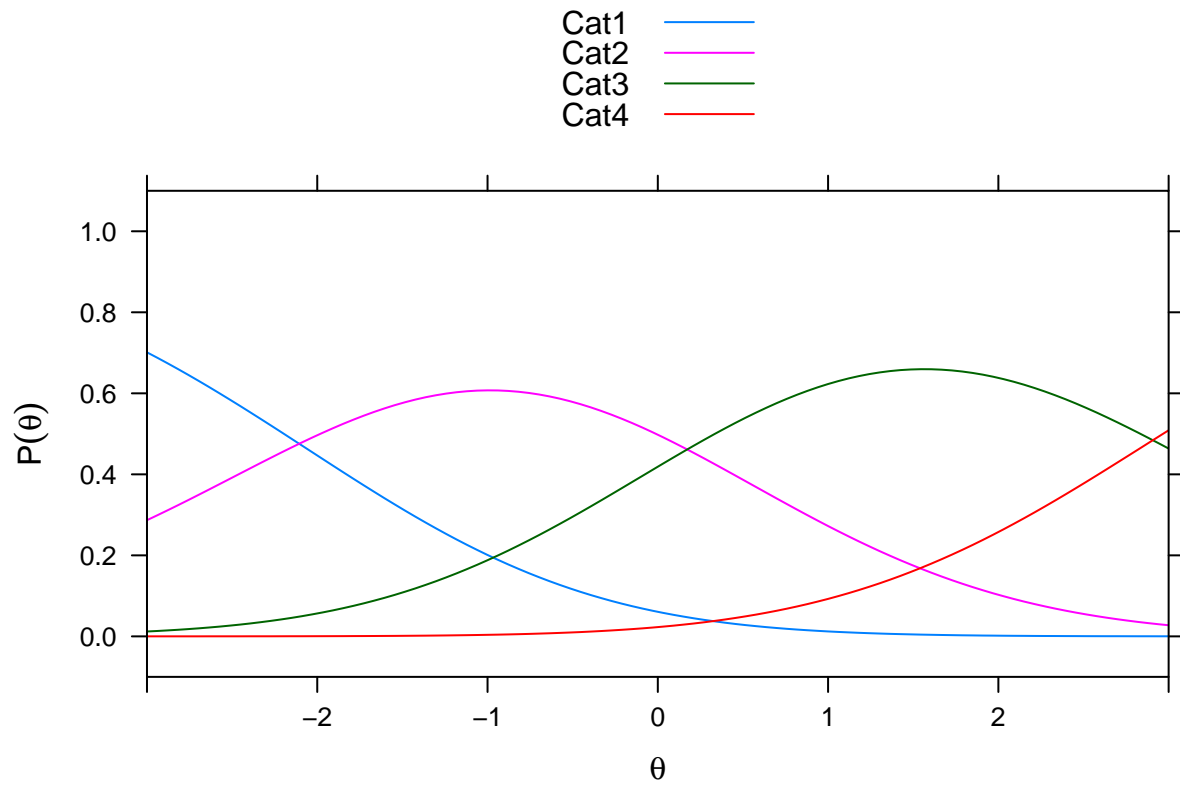# Item HIs3
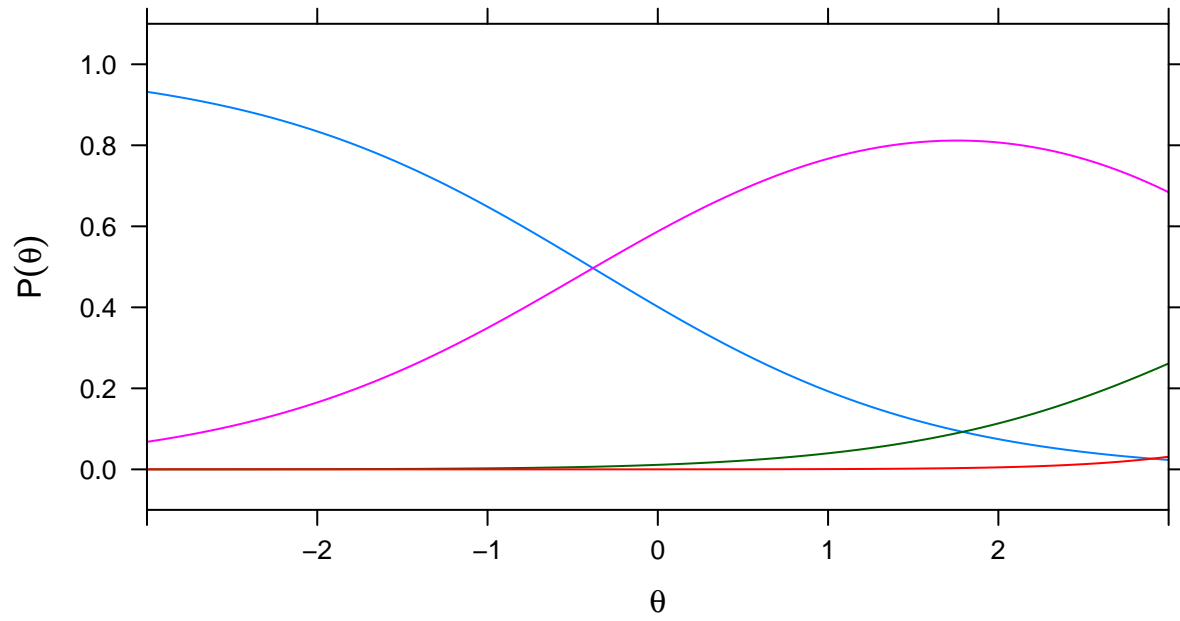
Item HIs4

Cat1
Cat2
Cat3
Cat4

Item HIs5

Item HIs6

Item HIs7

Cat1 ————
Cat2 ————
Cat3 ————
Cat4 ————

Item HIs8

Item HIs9
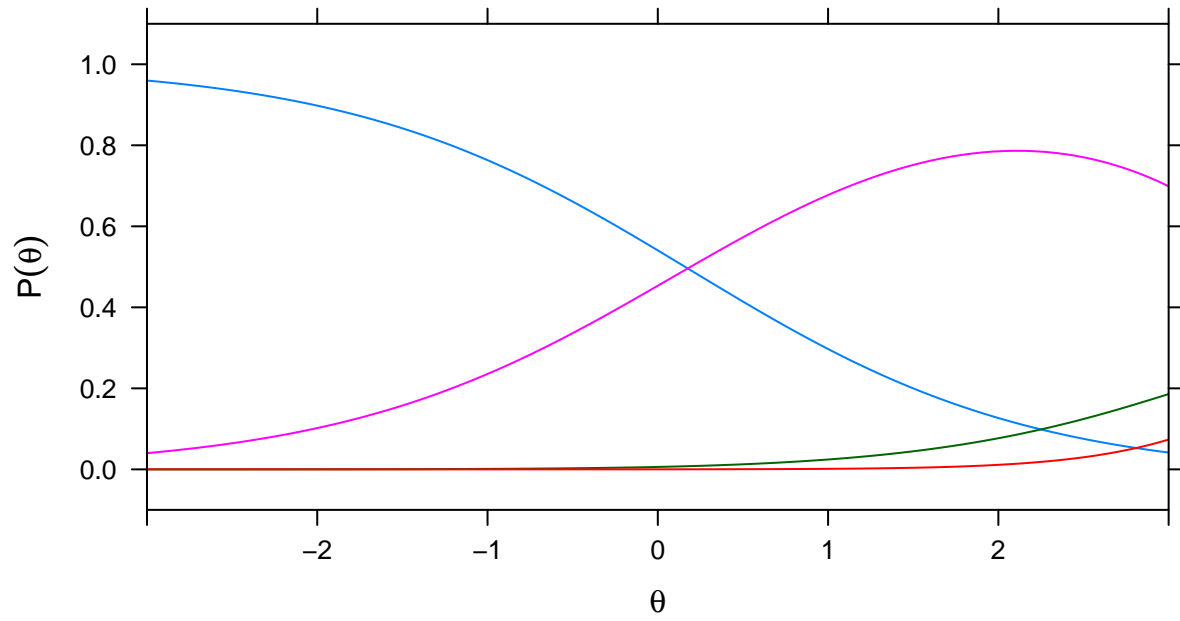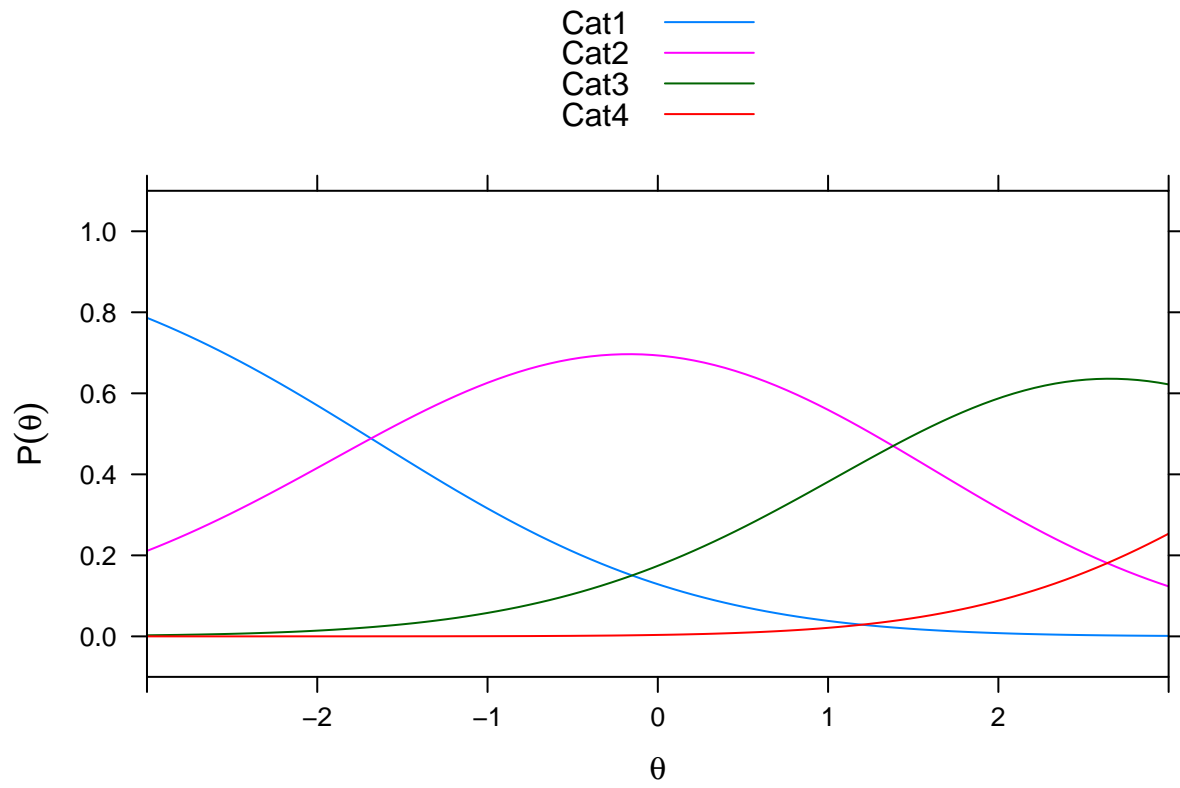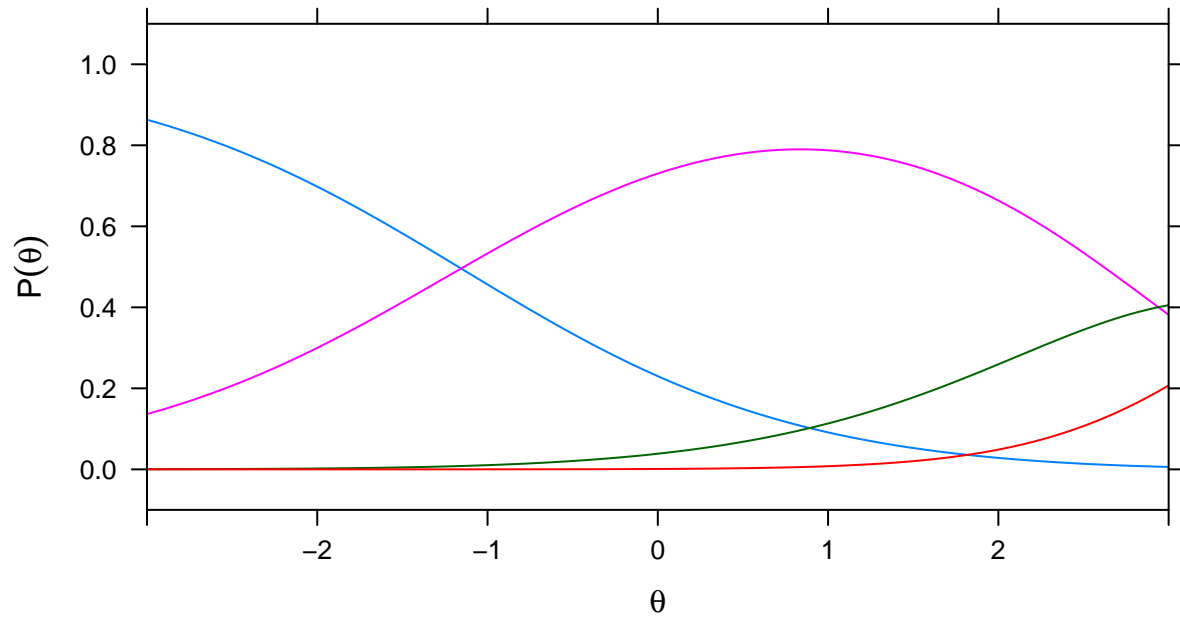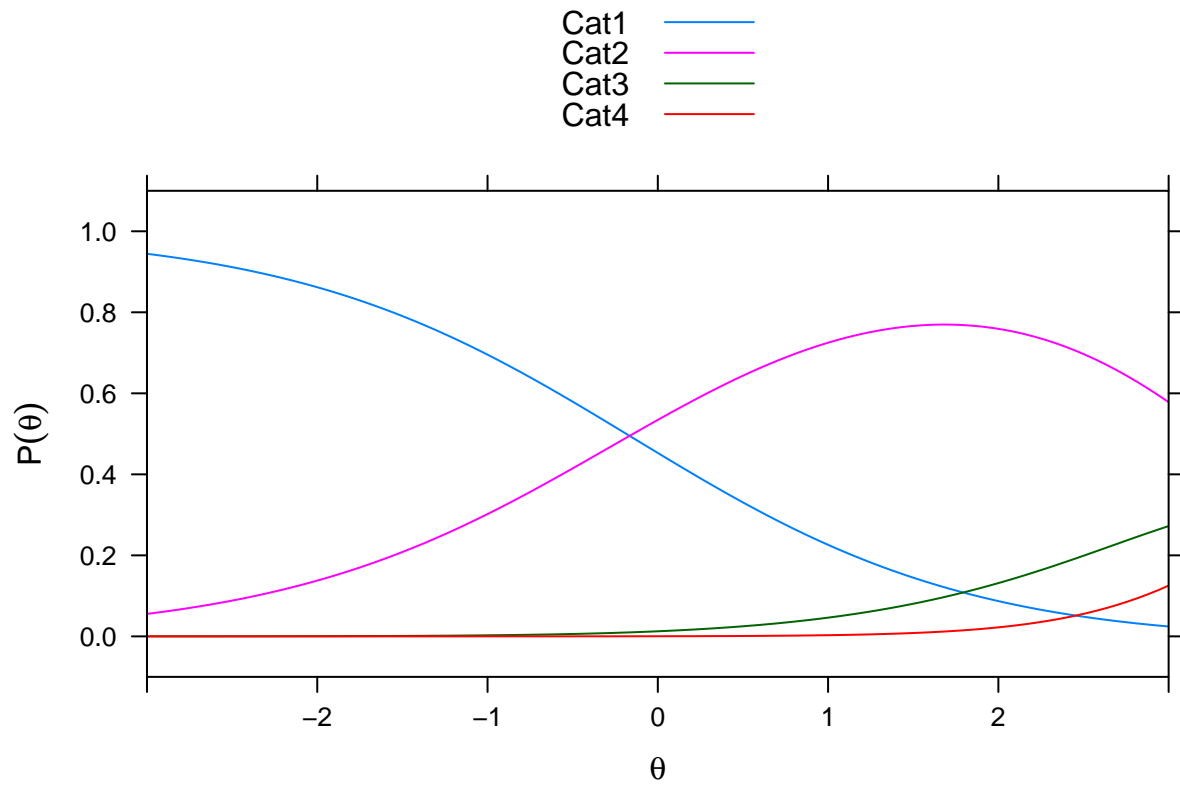
Cat1 ————
Cat2 ————
Cat3 ————
Cat4 ————

Item Hls10

Item HIs11

Cat1 ———
Cat2 ———
Cat3 ———
Cat4 ———

Item HIs12

Item HIs13

Item HIs14

Item HIs15

Cat1 ────────
Cat2 ────────
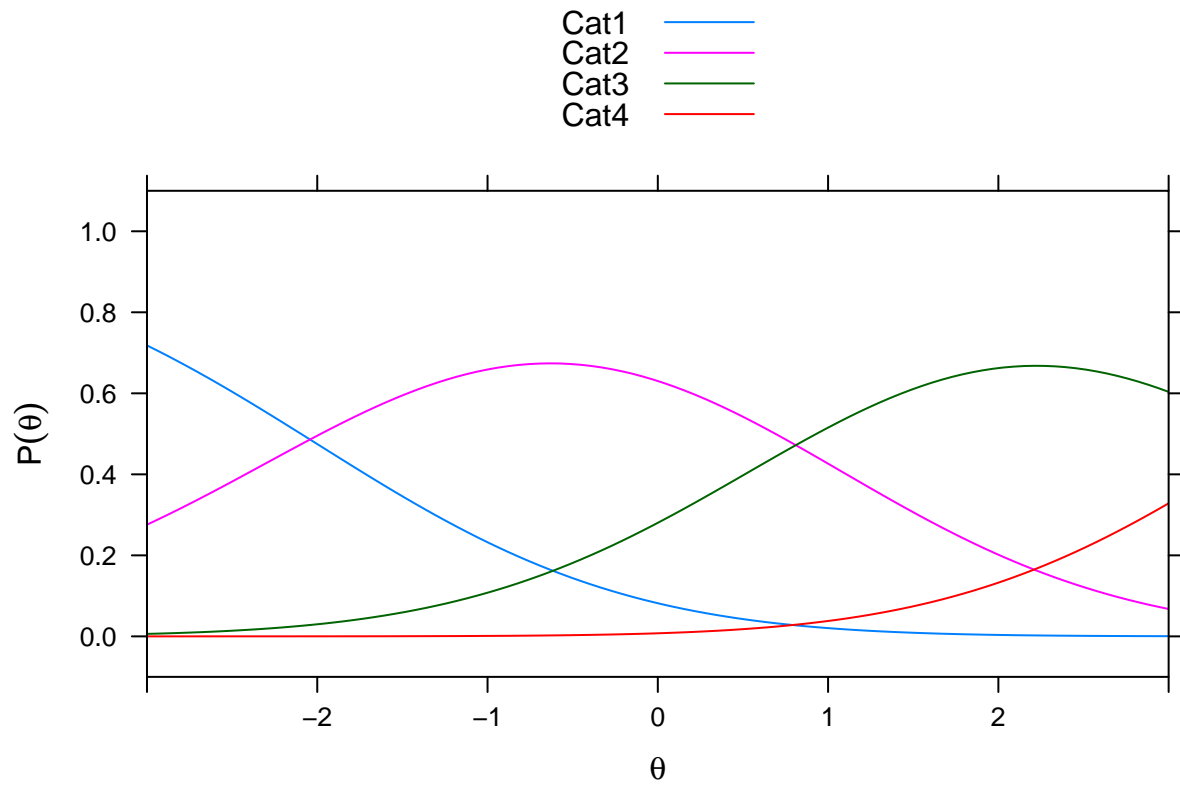Cat3 ────────
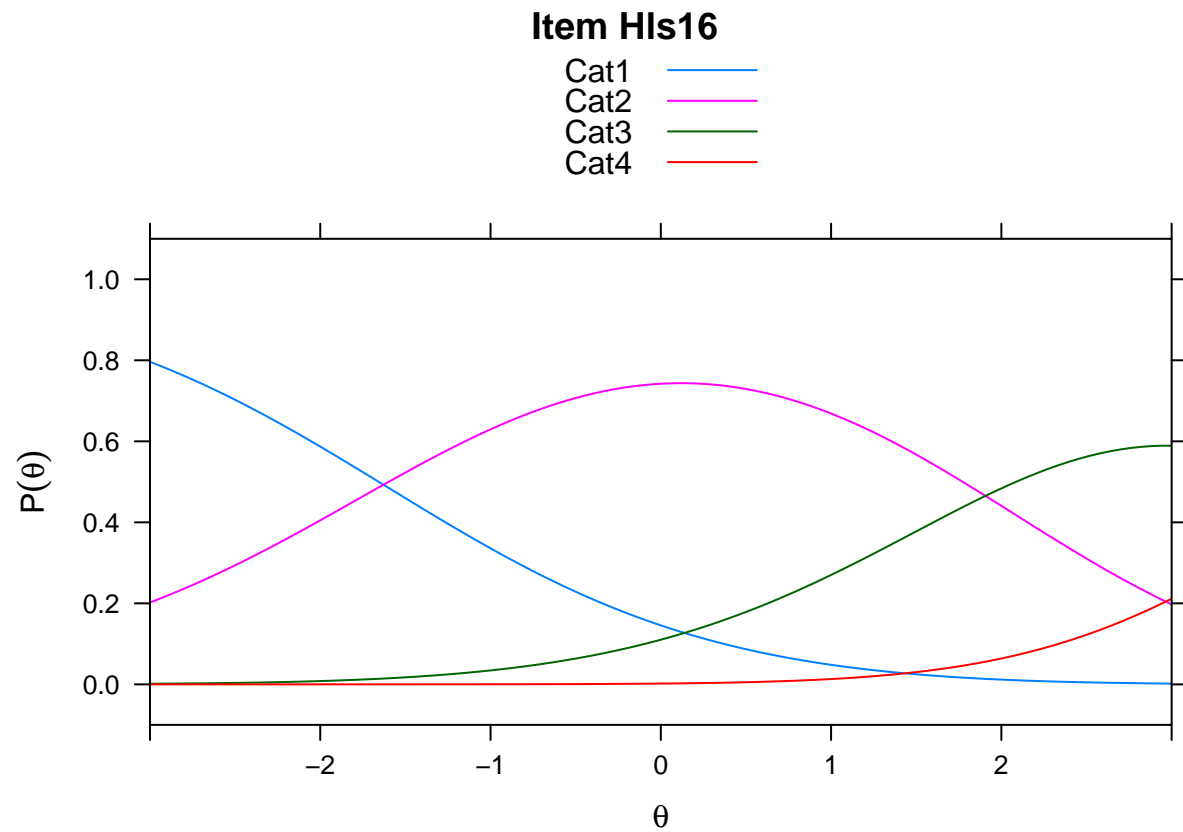Cat4 ────────

Item HIs16
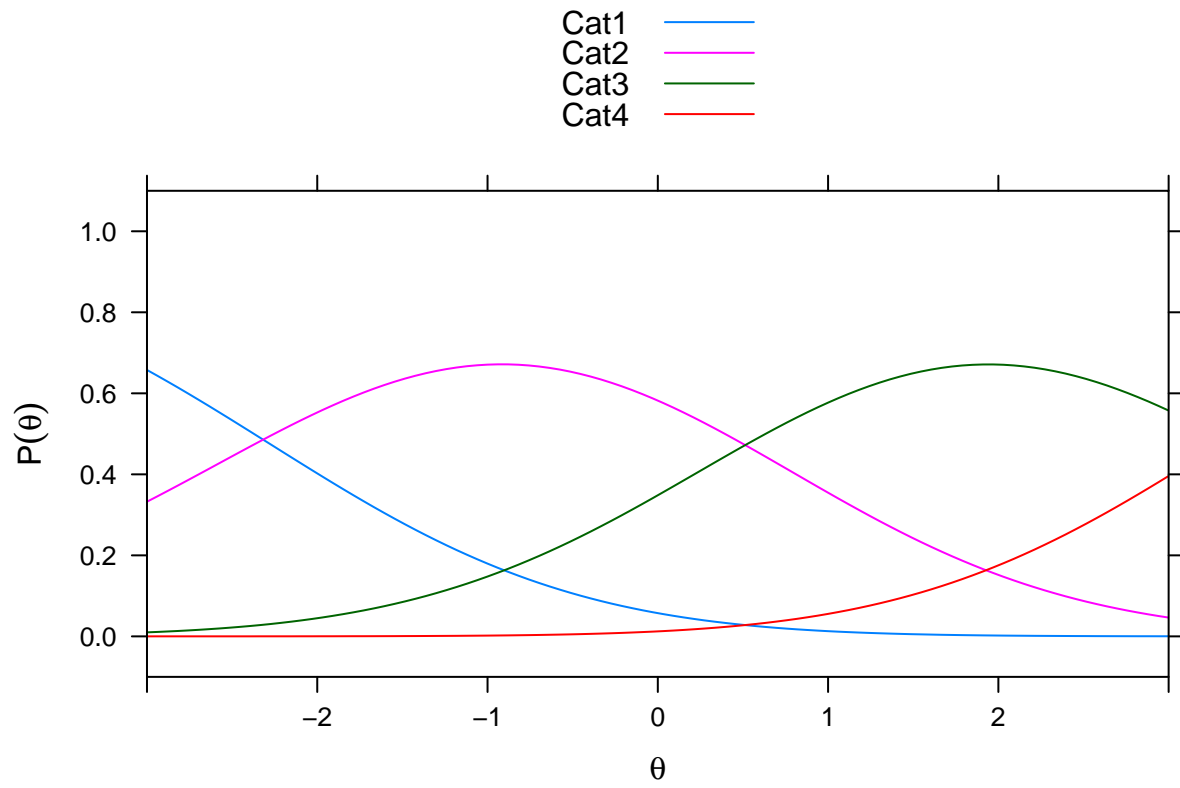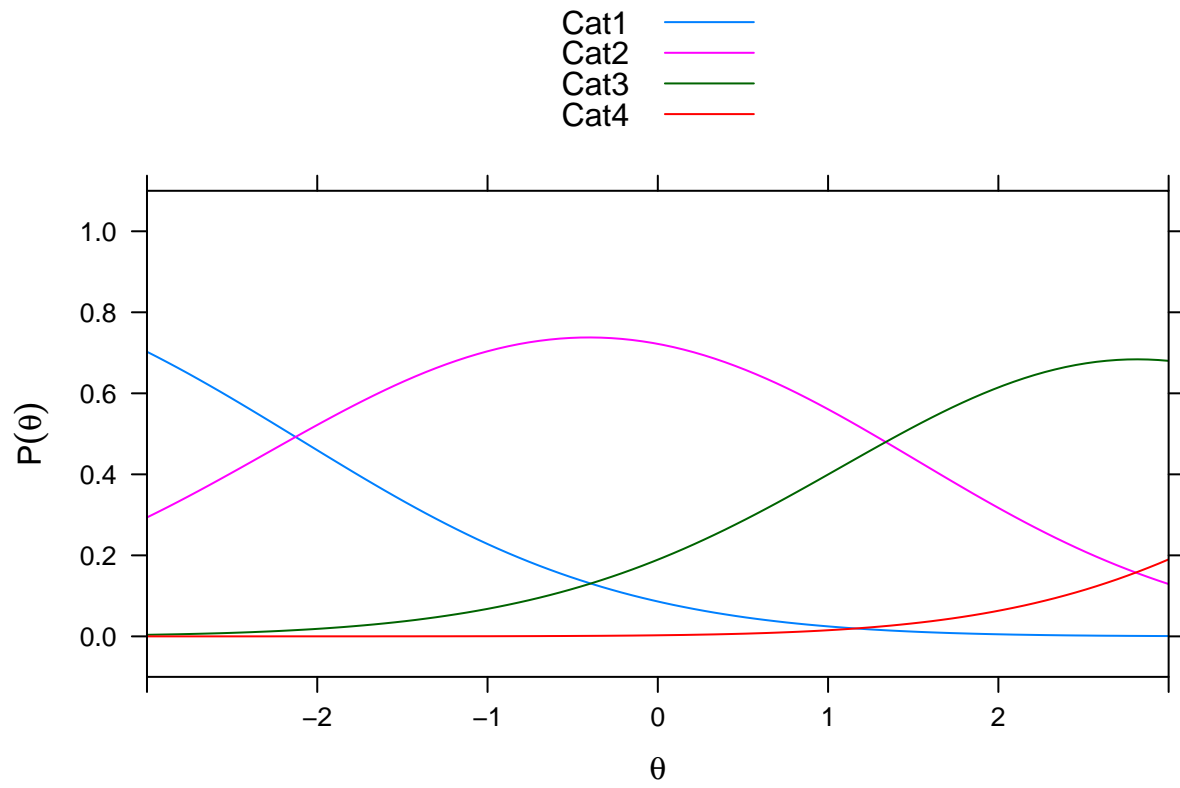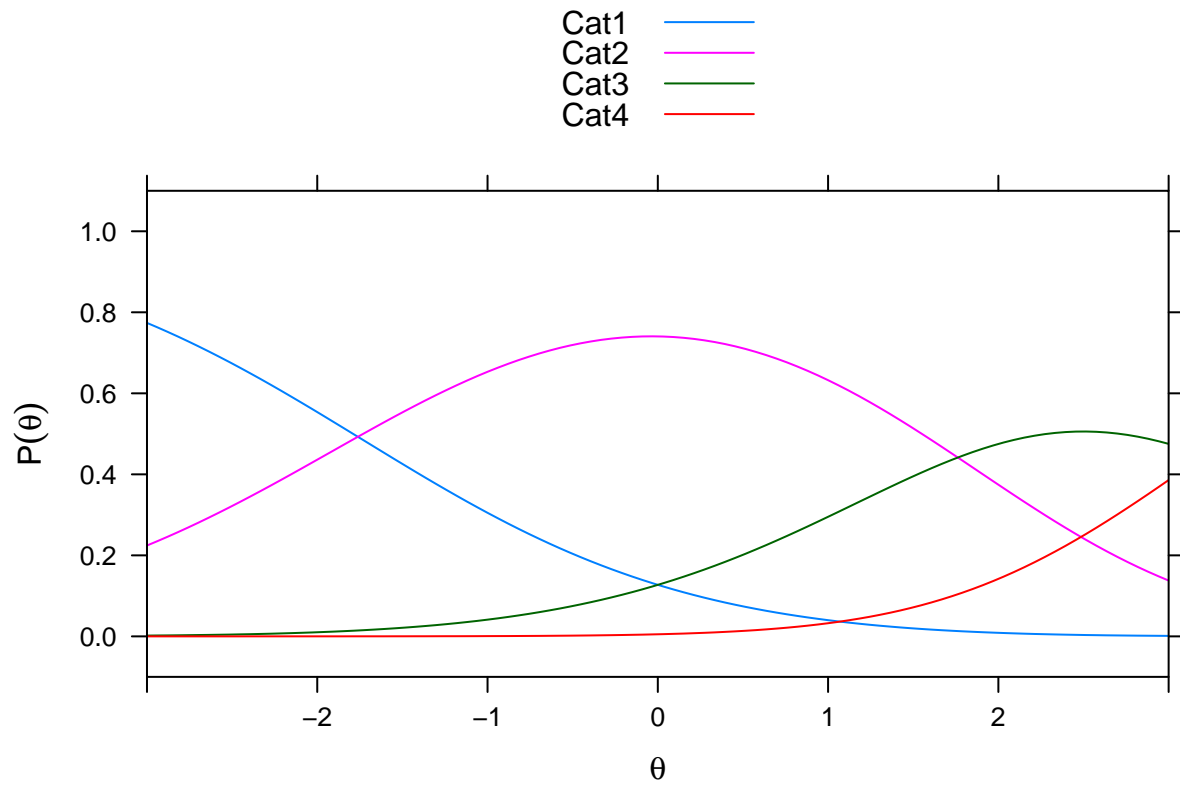
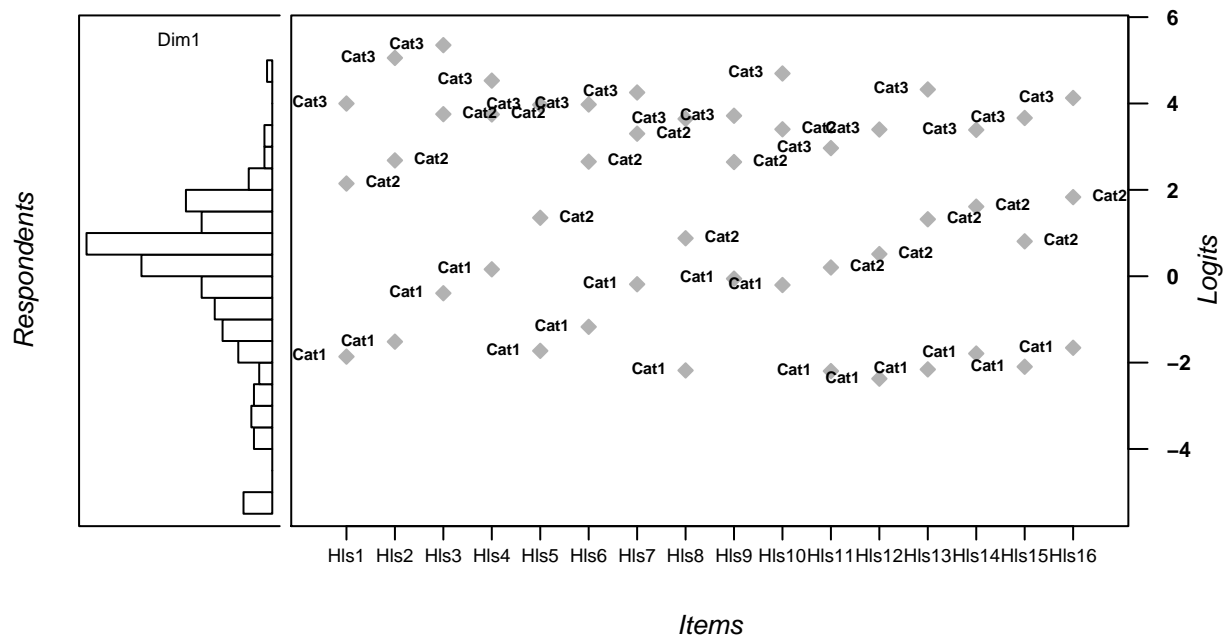## Wright Map

Here's a polytomous Wright Map

```
wrightMap(WLEestimates.poly, tthresh.poly)
```

Wright Map


```
##              Cat1       Cat2     Cat3
## Hls1  -1.86154175 2.1462708 3.998566
## Hls2  -1.51547241 2.6820374 5.054901
## Hls3  -0.39358521 3.7526550 5.350800
## Hls4   0.16012573 3.7458801 4.529205
## Hls5  -1.72787476 1.3532410 3.970184
## Hls6  -1.17178345 2.6529236 3.976410
## Hls7  -0.18539429 3.3007507 4.254547
## Hls8  -2.18106079 0.8795471 3.643341
## Hls9  -0.05612183 2.6443176 3.715851
## Hls10 -0.20352173 3.4025574 4.694550
## Hls11 -2.19607544 0.2026062 2.966949
## Hls12 -2.37240601 0.5131531 3.396881
## Hls13 -2.15725708 1.3193665 4.324860
## Hls14 -1.78994751 1.6114197 3.388824
## Hls15 -2.09591675 0.8077698 3.664764
## Hls16 -1.65664673 1.8318787 4.128021
```

Exercises:

1. Find an item for which Cat 3 is actually easier than the Cat 2 of another item.
2. Find an item that has two categories that are extremely close in severity.
3. Look at the ICC for item 14. Describe what is happening with Cat 3.

Model Comparison

say we want to compare the two models we just ran (note, these aren't really comparable since it's a completely different model - not nested data)

```r
logLik(mod1)
```

```
## 'log Lik.' -1880.882 (df=17)
```

```r
logLik(mod2)
```

```
## 'log Lik.' -4185.626 (df=49)
```

```r
anova(mod1, mod2)
```

```
##   Model   loglike Deviance Npars      AIC      BIC     Chisq df  p
## 1  mod1 -1880.882 3761.763    17 3795.763 3859.665 -4609.489 32  1
## 2  mod2 -4185.626 8371.252    49 8469.252 8653.438        NA NA NA
```

Log likelihood is the foundation of both AIC and BIC. AIC and BIC allow you to compare non-nested models while penalizing for model complexity (BIC penalizes more). In general, the model with a smaller AIC/BIC is the one that the data fit better. The two criteria sometimes disagree.

Multidimensional Rasch Model

What if we envision something that's multidimensional? We can model that with TAM. IN fact, this is one of TAM's great strengths. Do read package documentation, though. As the number of dimensions grows, you'll have to use particular estimation methods else the model will take to long to run.

we start by assigning the items to a dimension using a Q-matrix

If we want to have two dimensions, we'll create a matrix with two columns. A 1 or 0 denotes whether that item belongs to dimension 1 or 2 (or both!)

```r
Q <-
  matrix(c(1,0,0,0,0,0,0,1,1,1,1,1,1,1,1,1,0,1,1,1,1,1,1,0,0,0,0,0,0,0,0
           ,0),ncol=2)

Q
```

```
##       [,1] [,2]
##  [1,]    1    0
##  [2,]    0    1
##  [3,]    0    1
##  [4,]    0    1
##  [5,]    0    1
##  [6,]    0    1
##  [7,]    0    1
##  [8,]    1    0
##  [9,]    1    0
## [10,]    1    0
## [11,]    1    0
## [12,]    1    0
## [13,]    1    0
## [14,]    1    0
## [15,]    1    0
## [16,]    1    0
```

click on the "Q" object in the environment pane to see what we just made

Run the multidimensional Rasch model

```
multi <- TAM::tam.mml(resp=hls, Q=Q)
```

θ and δ

```
persons.multi <- tam.wle(multi)
```

```
## Iteration in WLE/MLE estimation  1   | Maximal change  2.6468
## Iteration in WLE/MLE estimation  2   | Maximal change  1.3204
## Iteration in WLE/MLE estimation  3   | Maximal change  2.3637
## Iteration in WLE/MLE estimation  4   | Maximal change  0.4899
## Iteration in WLE/MLE estimation  5   | Maximal change  0.0964
## Iteration in WLE/MLE estimation  6   | Maximal change  0.0224
## Iteration in WLE/MLE estimation  7   | Maximal change  0.0071
## Iteration in WLE/MLE estimation  8   | Maximal change  0.0023
## Iteration in WLE/MLE estimation  9   | Maximal change  7e-04
## Iteration in WLE/MLE estimation  10  | Maximal change  2e-04
## Iteration in WLE/MLE estimation  11  | Maximal change  1e-04
##
## -------
## WLE Reliability (Dimension1)=0.548
## WLE Reliability (Dimension2)=-0.683
```

```
WLEestimates.multi <- persons.multi$theta
thresholds.multi <- tam.threshold(multi)
```

```
#Fit and reliabilities
Fit.multi <- tam.fit(multi)
```

```
## Item fit calculation based on 100 simulations
## |**********|
## |----------|
```

```
Fit.multi$itemfit
```

```
##     parameter    Outfit    Outfit_t       Outfit_p Outfit_pholm      Infit
## 1        Hls1 1.3564003   2.9934420 2.758500e-03 0.0413774933 1.1459602
## 2        Hls2 1.3082495   1.7495400 8.019773e-02 0.6415818124 1.0692358
## 3        Hls3 0.8193862  -1.2572739 2.086545e-01 1.0000000000 0.9350767
## 4        Hls4 0.4225174  -2.6666186 7.661856e-03 0.0919422769 0.8646364
## 5        Hls5 0.8654329  -1.9224626 5.454758e-02 0.5317082474 0.9893065
## 6        Hls6 0.6716840  -2.7905926 5.261165e-03 0.0683951478 0.9438297
## 7        Hls7 0.8006432  -1.2064163 2.276570e-01 1.0000000000 0.9459093
## 8        Hls8 1.0462433   0.6132611 5.397037e-01 1.0000000000 1.0708640
## 9        Hls9 1.4499380   2.3109451 2.083589e-02 0.2291947675 0.9747220
## 10      Hls10 1.4732872   1.4809172 1.386286e-01 0.8317717671 1.0814151
## 11      Hls11 0.8802598  -1.9335309 5.317082e-02 0.5317082474 0.9195533
## 12      Hls12 0.7595863  -4.0564587 4.982236e-05 0.0007971577 0.8557617
## 13      Hls13 0.9178672  -1.2289097 2.191057e-01 1.0000000000 1.0365018
## 14      Hls14 1.1553715   1.6570111 9.751723e-02 0.6826206248 1.0159429
## 15      Hls15 0.8204242  -2.9315734 3.372496e-03 0.0472149460 0.9439654
## 16      Hls16 1.0541368   0.3642034 7.157061e-01 1.0000000000 1.0147707
```

```
##         Infit_t     Infit_p Infit_pholm
## 1    1.3840870 0.16633180   1.0000000
## 2    0.5351301 0.59255992   1.0000000
## 3   -0.2007844 0.84086713   1.0000000
## 4   -0.4097451 0.68199296   1.0000000
## 5   -0.1283134 0.89790100   1.0000000
## 6   -0.3713297 0.71039201   1.0000000
## 7   -0.1930735 0.84690143   1.0000000
## 8    1.0438936 0.29653463   1.0000000
## 9   -0.1248790 0.90061936   1.0000000
## 10   0.4284881 0.66829580   1.0000000
## 11  -1.2502939 0.21119219   1.0000000
## 12  -2.3272032 0.01995445   0.3192712
## 13   0.5037912 0.61440810   1.0000000
## 14   0.2091055 0.83436588   1.0000000
## 15  -0.8581970 0.39078367   1.0000000
## 16   0.1859782 0.85246185   1.0000000
```
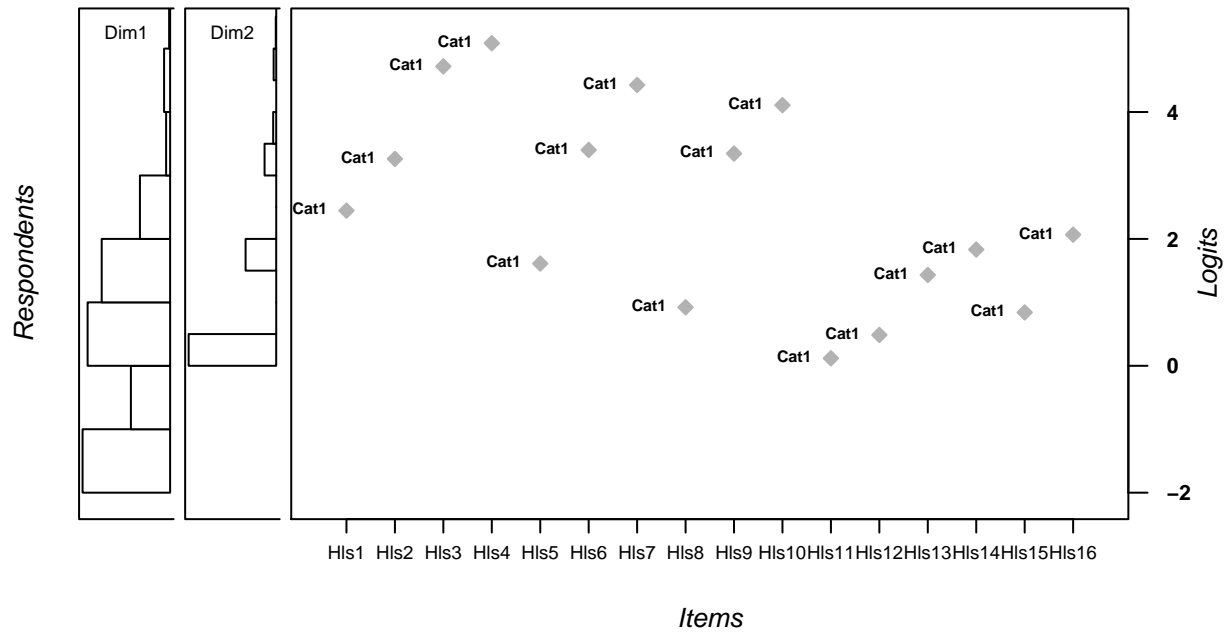
```r
multi$EAP.rel #EAP reliabilities
```

```
##      Dim1      Dim2
## 0.7648527 0.6680034
```

Wright Map

```r
MDthetas.multi <-
  cbind(persons.multi$theta.Dim01,persons.multi$theta.Dim02) #one line
wrightMap(MDthetas.multi, thresholds.multi) #second line
```

Wright Map


```
##              Cat1
## Hls1   2.4458313
## Hls2   3.2610168
## Hls3   4.7192688
## Hls4   5.0847473
## Hls5   1.6114197
## Hls6   3.4018250
## Hls7   4.4266663
## Hls8   0.9223938
## Hls9   3.3454285
## Hls10  4.1084290
## Hls11  0.1181946
## Hls12  0.4860535
## Hls13  1.4316101
## Hls14  1.8324280
## Hls15  0.8414612
## Hls16  2.0666199
```

Compare the first unidimensional model to the multidimensional one

```
logLik(mod1)
```

```
## 'log Lik.' -1880.882 (df=17)
```

```
logLik(multi)
```

```
## 'log Lik.' -1873.238 (df=19)
```

```
anova(mod1, multi)
```

```
##   Model   loglike Deviance Npars      AIC      BIC   Chisq df       p
## 1  mod1 -1880.882 3761.763    17 3795.763 3859.665 15.28774  2 0.00048
## 2 multi -1873.238 3746.476    19 3784.476 3855.895      NA NA      NA
```

Alternatively, you can use IRT.compareModels

```
compare <- CDM::IRT.compareModels(mod1, multi)
compare
```

```
## $IC
##   Model   loglike Deviance Npars Nobs      AIC      BIC     AIC3     AICc
## 1  mod1 -1880.882 3761.763    17  317 3795.763 3859.665 3812.763 3797.810
## 2 multi -1873.238 3746.476    19  317 3784.476 3855.895 3803.476 3787.035
##       CAIC
## 1 3876.665
## 2 3874.895
##
## $LRtest
##   Model1 Model2    Chi2 df           p
## 1   mod1  multi 15.28774  2 0.0004789713
##
## attr(,"class")
## [1] "IRT.compareModels"
```

```
summary(compare)
```

```
## Absolute and relative model fit
##
##   Model   loglike Deviance Npars Nobs      AIC      BIC     AIC3     AICc
## 1  mod1 -1880.882 3761.763    17  317 3795.763 3859.665 3812.763 3797.810
## 2 multi -1873.238 3746.476    19  317 3784.476 3855.895 3803.476 3787.035
##       CAIC
## 1 3876.665
## 2 3874.895
##
## Likelihood ratio tests - model comparison
##
##   Model1 Model2   Chi2 df      p
## 1   mod1  multi 15.2877  2 5e-04
```

We see that model `multi` fits slightly better. However, the log likelihood difference test shows the difference is statististically significant.

```
##   Model1 Model2    Chi2 df           p
## 1   mod1  multi 15.28774  2 0.0004789713
```

```
compare$LRtest
```

Exercises

1. what evidence points towards multidimensionality?
2. compare the multidimensional model to the PCM model