Assignment 3 solutions

10/17/2024

EPSY 6637 Assignment #3 Fall 2024

1) In Week 6 we looked at some of the peculiarities that can arise when using the 3PL. In one example, we showed that if a test started with an easy item, and then had very hard items, a person who missed the first question would have a very hard time improving their theta score, even if they got all the remaining items correct.

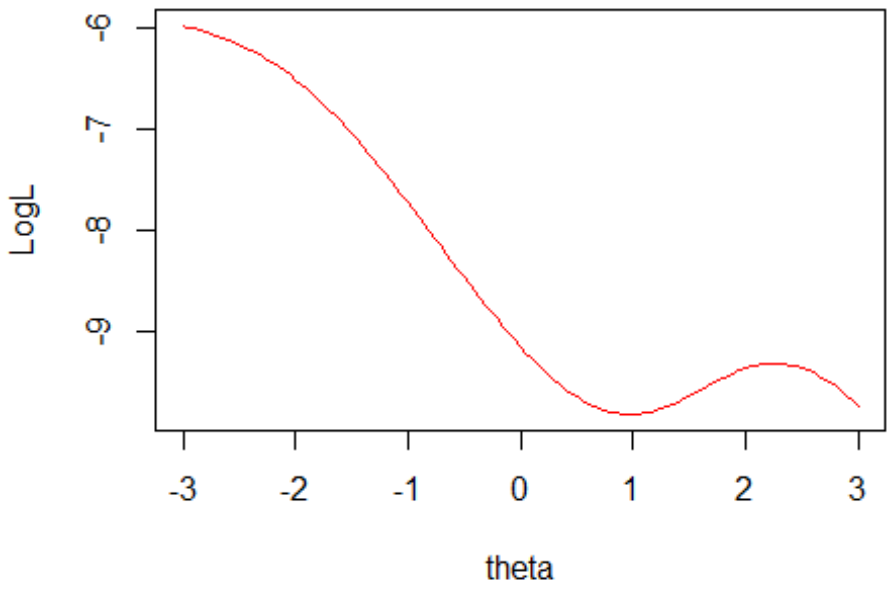
a) Using the example items from the class scripts (params<-matrix(c(1.8,1.8,1.8, 1.8,1.8,1.8, 1.8,1.8, -2,2 ,2,2,2,2,2,2,.25,.25,.25,.25,.25,.25,.25,.25), nrow=8), show that a person with response (0,1,1,1,1) is still estimated to have a very low theta. Make a plot of the likelihood as a function of theta. Why is this person scored so low?

The plot of the log likelihood function shows that the person is still being scored very low. Even after 4 correct answers in a row to very difficult questions, the maximum likelihood estimate for $\hat{\theta}$ is at the minimum on the grid. The problem is the asymmetry in the likelihood function: on the one hand, the guessing parameters for these items are $c = 0.25$ indicating a 1 in 4 chance that the someone with no ability at all could still get the question correct.

But the upper asymptote for the function is 1. For the first item, with $b_1 = -2$, the model predicts that a person with $\theta = 1$ (i.e. above average ability) will have a 0.997 probability of getting it right. So the wrong answer is causing a big problem for scoring.

The upshot is that this particular response pattern is ambiguous. The person has answered 4 very difficult questions correctly, making them seem very capable. However, they also missed a super easy question, and this is inconsistent with the hypothesis that they are strong. As it stands for this person, after answering 5 questions, their ability estimate is extremely low. You can only see the hint at the higher ability level for the idea that the easy question is the anomaly, and the harder ones legit, but the lower estimate still wins out.

```
params<-matrix(c(1.8,1.8,1.8,1.8,1.8,1.8,1.8,1.8,-
2,2,2,2,2,2,2,2,.25,.25,.25,.25,.25,.25,.25,.25),nrow=8)


getloglike(5,params,c(0,1,1,1,1))->weird
plot(seq(-3,3,.05),weird, col="red", type="l", ylab="LogL",xlab="theta")
```




```
which.max(weird)
```

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

b) Take the getloglike function code, and modify it slightly so thar a person has a 0.98 chance of getting the item correct if they are of very high ability. Save it as a different function name (you could call it getloglikeu), and recalculate the likelihood function that you did in part (a). What is the new score for this person? Why would it be different from part (a)?

Here's how to edit the function:

```
getloglikeupper <- function(nitems,params,response) {

  pq<-array(0,c(121,nitems))
  theta<-seq(-3,3,.05)

  for (j in 1:nitems) {

    temp<- params[j,1]*(theta - params[j,2])

    p <- params[j,3] + (0.98-params[j,3])/(1 + exp(-temp))

    pq[,j]<- response[j]*p + (1-response[j])*(1-p)

  }
```

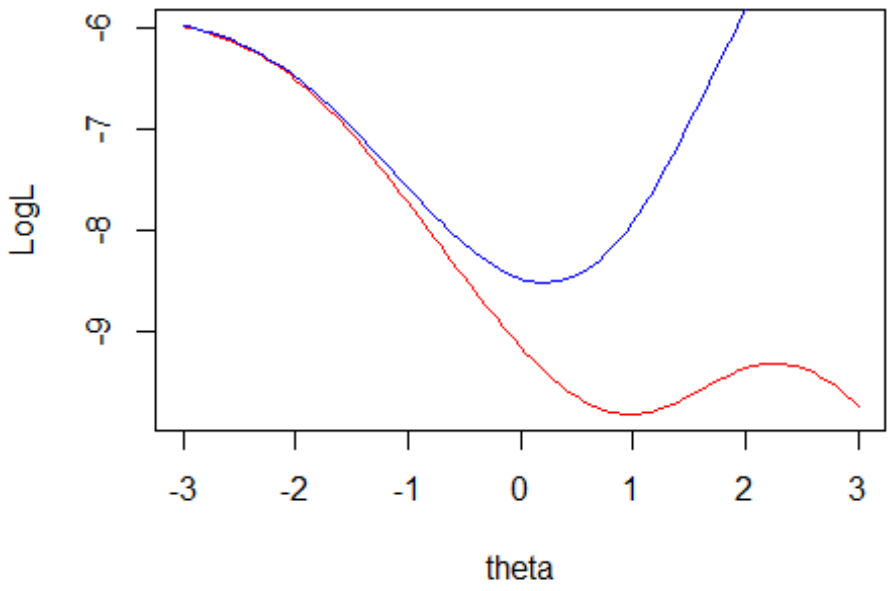
```
  loglike<-apply(log(pq),1,sum)

  loglike
}
```

Our new tailor made model, with an upper asymptote of 0.98, makes things look very different. Now the early wrong answer to the easy question may have resulted from the 1/50 chance of a mistake. The new likelihood for the same parameters and responses gives us the blue line to compare. Under the new model, the likelihood that the person has high ability is clearly highest.

To be very concrete - here's the dynamic that creates the blue line. IF the person has very low ability, then they have just guessed four answers in a row - $(1/4)^4 = 1/256$ *which means these guesses have probability about .004. IF the person has high ability, then they had a slip-up that can occur* $1 - 0.98 = 0.02$ *or 1/50 times. Under the previous model, the* $1/256$ *still looked better than the wrong answer which was seen as super unlikely. After we allow that a wrong answer can happen, then the more likely scenario is that the four correct answers are due to high ability. The blue line is actually heading to positive infinity.*

We made a very small tweak to our model function, which made a slightly different assumption. But it allowed for a different explanation for the data, and a very different conclusion.

```
getloglikeupper(5,params,c(0,1,1,1,1))->weird2
plot(seq(-3,3,.05),weird, col="red", type="l", ylab="LogL",xlab="theta")
points(seq(-3,3,.05),weird2, col="blue", type="l")
```




```
which.max(weird)
```

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

```
which.max(weird2)
```

```
## [1] 121
```

2) In the workspace, there is a dataset called physics that has dimension (739,24). (Check dim(physics) so you know you're working with the right dataset.)

a) Fit the Rasch, 2PL and 3PL to the data, commenting on the relative overall fit. Show how the AIC and BIC values for the 2PL is calculated using the log-likelihood value, the sample size and the number of parameters.

```
library(mirt)
```

```
## Loading required package: stats4
```

```
## Loading required package: lattice
```

```
mirt(physics,1,"Rasch") ->m1
mirt(physics,1,"2PL") ->m2
mirt(physics,1,"3PL") ->m3
```

```
## EM cycles terminated after 500 iterations.
```

Model	LogL	AIC	BIC	G^2	Params
1PL	-6244	12521	12599	3708	17

Model	LogL	AIC	BIC	G^2	Params
2PL	-6023	12110	12258	3268	32
3PL	-5932	11961	12182	3087	48

For the 2PL - the log-likelihood is -8508. To get the AIC:

$$AIC = -2 * logL + 2 * params = 12046 + 2 * 32 = 12110$$

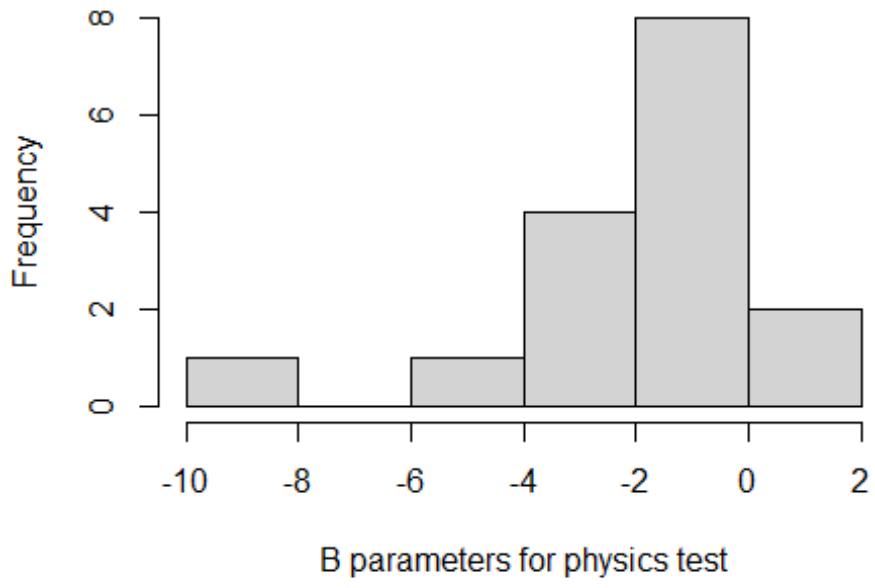
To get the BIC:

$$BIC = -2 * logL + log(N) * params = 12046 + log(739) * 32 = 12258$$

The model comparison shows a clear preference for the 3PL. However, looking at the coefficients, there are some wild estimates for a values (4 slopes are above 30!) so we would need to take a look to see how to get those under control and see if we really need to use that model (we could maybe add prior information to control the slopes).

b) Make a histogram of the b parameters from the 2PL. Comment on the overall apparent difficulty of the test.

```
hist(coef(m2,IRTpars=T,simplify=T)$items[,2], xlab="B parameters for physics
test", main="")
```



Most of the items have negative b values. This means that the test is mostly composed of easy items. (This is true for many tests in large college classes if the intention is to have a

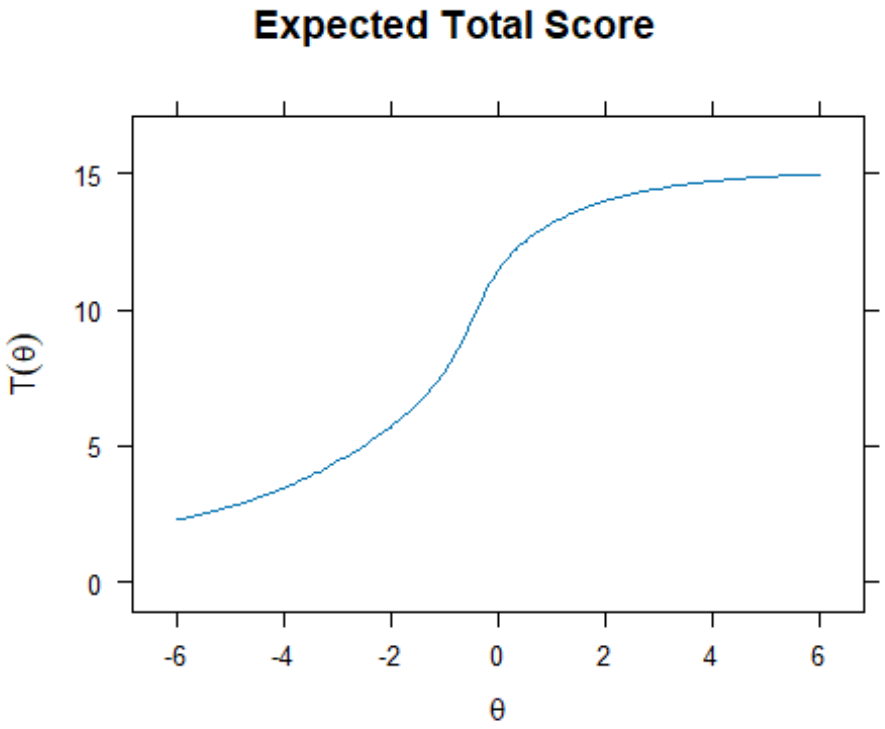
decently high class average. Optimal for the grading scheme - not optimal for information about ability!)

FWIW the default plot function in mirt gets the test response function. This is simply the sum of all the item functions, but it then shows a mapping between θ and expected total score. Notice where the grades A and B and C would be in terms of the

$$\theta$$

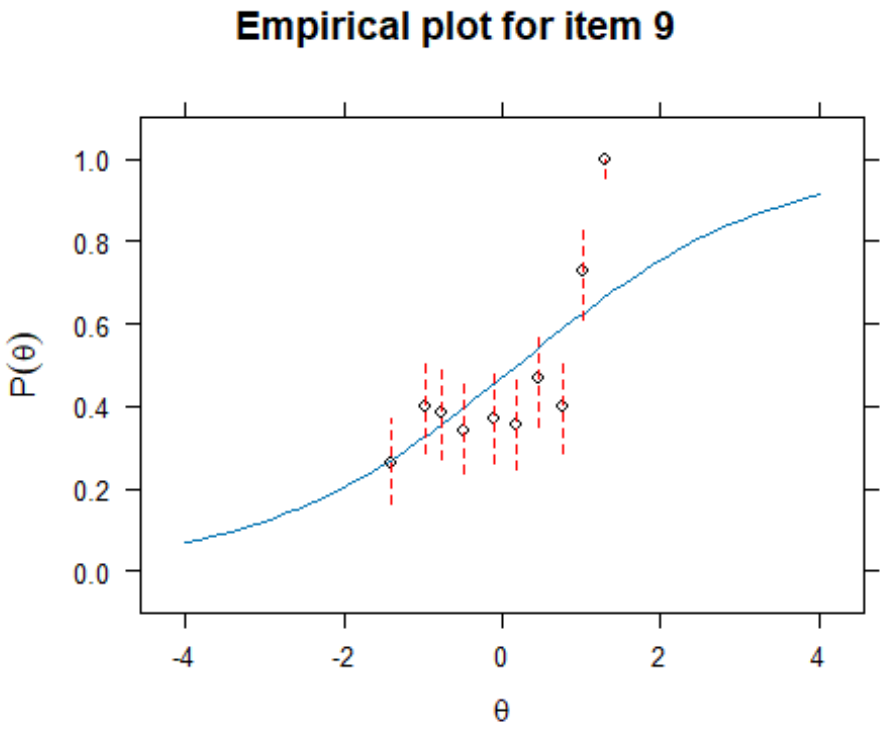
.

```
plot(m2)
```

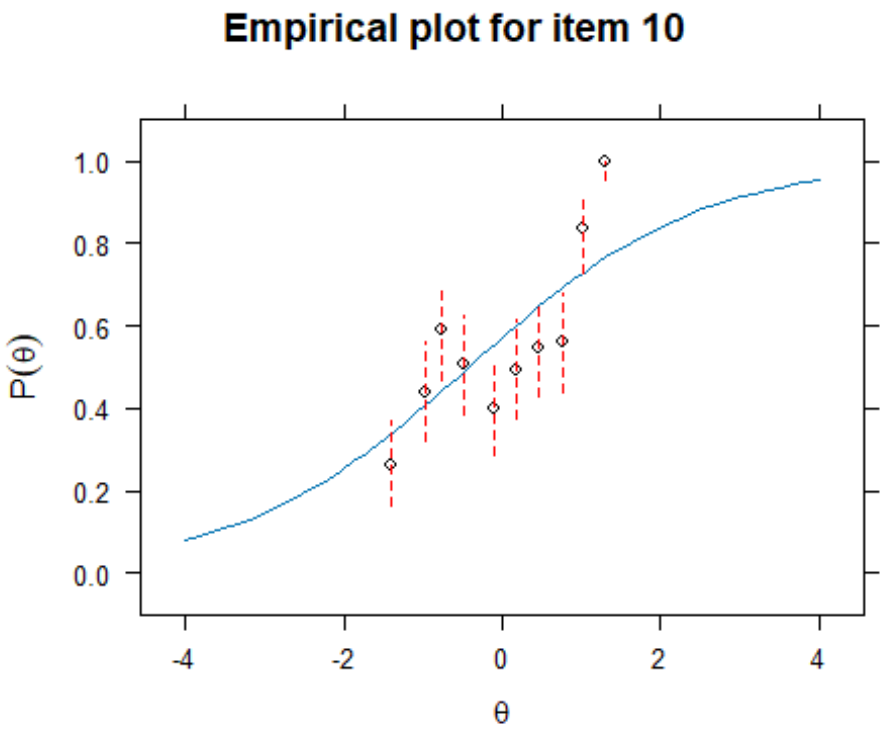


c) Still using the 2PL (for this and the rest of the questions), make an empirical plot of items 9 and 10. Do you see any systematic differences between the predicted and observed scores? Do you have any ideas as to why this pattern might happen?

```
itemfit(m2,empirical.plot = 9)
```




```
itemfit(m2,empirical.plot = 10)
```



Both item curves capture an upward trend of the data. However, they both show some strange behavior. They seem flat for about 8 of the bins, and then a very sharp spike for the last two. The fit doesn't seem all that good - and it's not surprising that for the 3PL the slope is higher, and the guessing rate quite large.

Incidentally, item 11 (not asked for) is a real clunker, with a negative slope and a very poor empirical plot. It looks like an item that either had a wrong key, or that students really didn't know what to do with.

d) Use fscores(model, full.scores.SE = T, method = "ML")[1:10,] to get the first 10 estimates, along with their SE, for method = ML, MAP and EAP. Explain how the estimates are different and why.

```
fscores(m2, full.scores.SE = T, method = "ML")[1:10,]
```

```
## Warning: The following factor score estimates failed to converge
successfully:
##
6,25,45,74,121,131,133,145,189,211,298,307,311,315,336,370,413,472,499,506,58
5,605,623,636,690
```

```
##                F1     SE_F1
##  [1,] -0.75398559 0.3338127
##  [2,]  0.69992664 0.8020600
##  [3,] -0.60651191 0.2979091
##  [4,] -1.25138923 0.5358398
##  [5,] -0.01285903 0.3885207
##  [6,] 19.99997959        NA
##  [7,]  0.02047327 0.4057991
##  [8,]  3.31536501 2.1924118
##  [9,] -1.25125069 0.5357737
## [10,] -0.89299153 0.3797763
```

```
fscores(m2, full.scores.SE = T, method = "MAP")[1:10,]
```

```
##                F1     SE_F1
##  [1,] -0.68244670 0.3000197
##  [2,]  0.45589822 0.5540103
##  [3,] -0.55835509 0.2783636
##  [4,] -1.01818242 0.3945018
##  [5,] -0.01116978 0.3628392
##  [6,]  1.29002093 0.7399428
##  [7,]  0.01758852 0.3747999
##  [8,]  1.06658641 0.7033822
##  [9,] -1.01810732 0.3944771
## [10,] -0.78985002 0.3258525
```

```
fscores(m2, full.scores.SE = T, method = "EAP")[1:10,]
```

```
##                F1     SE_F1
##  [1,] -0.7401989 0.3300369
```

```
##  [2,]  0.6248767 0.5489472
##  [3,] -0.5883290 0.3098208
##  [4,] -1.1329396 0.4174925
##  [5,]  0.1146377 0.4038396
##  [6,]  1.4104538 0.7165210
##  [7,]  0.1500021 0.4139651
##  [8,]  1.2014661 0.6795477
##  [9,] -1.1328555 0.4174717
## [10,] -0.8689268 0.3550337
```

The ML (Maximum Likelihood) estimates are based only on the likelihood function - with the implicit assumption that prior to the data the θ may have been any value from $(-\infty, \infty)$. Notice that for person 6 the estimate value is in fact ∞ because they got all the questions right.

The MAP (mode of the posterior distribution) and EAP (expected value of the posterior distribution) differ by assuming a standard normal prior distribution. That is, if the mean and variance of the population distribution of θ are going to be 0 and 1, then why not assume before getting any data that the person comes from that distribution? This means that estimates are available for all the people. It also means that the results are pulled in, or shrunk towards zero, because the normal prior had more density around the middle. Every value is pulled in a bit from the ML results - and the degree to which they are pulled is greater for the largest absolute value ML estimates. ML estimates close to 0 are hardly pulled in at all; estimates farther from zero are pulled in quite a bit. See for instance persons 6 and 8.

The standard errors for the EAP and MAP are smaller because the prior information is exactly that - a small amount of information added to the data. Also, notice everywhere that at lower $\hat{\theta}$ the SE is smaller. This is consistent with what we learned when we saw that most of the items had below average difficulties.

e) The 2PL model has a G2 value of 3267.6. Explain what this value refers to. It also has a ridiculous number of degrees of freedom (df = 65503). How are those degrees of freedom calculated?

First of all, the G^2 value is computing the likelihood ratio test for all the available answer patterns. If you ask to see how many unique lines there in the physics data - there are 519. There are also 519 unique $\hat{\theta}$ values. The G^2 value compares the probabilities for the 519 cells as predicted by the model to the actual observed probabilities. The discrepancy is called the deviance, comparing our 2PL model with 48 parameters to a saturated model that exactly fits the 519 cells.

Where does the $df = 65503$ come from? In theory, for a 16 item test with binary (0/1) scoring, there could be $2^{16} = 65536$. That means that theoretically there are $df = 65536 - 1$. We used 32 parameters (and a and b for each of 24 items) to fit the 2PL. Therefore, for our model $df = 65536 - 1 - 32 = 65503$.

```r
dim(unique(physics))
```

```
## [1] 519  16
```

```r
length(unique(fscores(m2)))
```

```
## [1] 519
```

f) Using the residuals(model, "Q3") command, evaluate the model assumption of conditional independence. Explain

The Q3 statistic calculates the correlations among all items of the residuals. It creates, therefore, a big 16 by 16 correlation matrix. The assumption of conditional independence means that these correlations are expected to be zero. Of course due to sampling variation it won't be exactly zero, but we can look for larger values in the matrix. In this case there are some large values - for instance, items 6 and 7 have a residual correlation of 0.4. One possibility is that those two items had a common diagram or set-up. There are also some relatively high correlations among items 3, 4, 5, 6.

```r
residuals(m2,"Q3")->xx
```

```
## Q3 summary statistics:
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  -0.321  -0.082  -0.005  -0.020   0.045   0.403
##
##         V1     V2     V3     V4     V5     V6     V7     V8     V9    V10
## V1   1.000  0.173  0.015 -0.087 -0.105 -0.044  0.029  0.167 -0.120  0.007
## V2   0.173  1.000 -0.086  0.015 -0.120 -0.119  0.001  0.031 -0.042  0.060
## V3   0.015 -0.086  1.000 -0.257 -0.248 -0.246 -0.192  0.098 -0.164 -0.125
## V4  -0.087  0.015 -0.257  1.000 -0.219 -0.321 -0.270 -0.056 -0.062 -0.111
## V5  -0.105 -0.120 -0.248 -0.219  1.000 -0.113 -0.109 -0.100 -0.091 -0.081
## V6  -0.044 -0.119 -0.246 -0.321 -0.113  1.000  0.403 -0.011  0.079  0.063
## V7   0.029  0.001 -0.192 -0.270 -0.109  0.403  1.000 -0.067  0.027 -0.027
## V8   0.167  0.031  0.098 -0.056 -0.100 -0.011 -0.067  1.000 -0.070  0.050
## V9  -0.120 -0.042 -0.164 -0.062 -0.091  0.079  0.027 -0.070  1.000  0.089
## V10  0.007  0.060 -0.125 -0.111 -0.081  0.063 -0.027  0.050  0.089  1.000
## V11  0.040  0.062  0.026  0.058 -0.013 -0.080 -0.014  0.044 -0.020  0.067
## V12 -0.014 -0.018 -0.098 -0.125 -0.019  0.097  0.044 -0.021  0.059 -0.005
## V13  0.109  0.069 -0.147 -0.105 -0.055  0.005 -0.005  0.151 -0.031 -0.005
## V14  0.099  0.063 -0.154 -0.258 -0.056  0.030 -0.003  0.017  0.019  0.047
## V15 -0.073 -0.060 -0.133 -0.132  0.027  0.017 -0.029 -0.052  0.118  0.037
## V16  0.001 -0.002 -0.097 -0.100 -0.032 -0.005 -0.034 -0.037  0.051  0.068
##        V11    V12    V13    V14    V15    V16
## V1   0.040 -0.014  0.109  0.099 -0.073  0.001
## V2   0.062 -0.018  0.069  0.063 -0.060 -0.002
## V3   0.026 -0.098 -0.147 -0.154 -0.133 -0.097
## V4   0.058 -0.125 -0.105 -0.258 -0.132 -0.100
## V5  -0.013 -0.019 -0.055 -0.056  0.027 -0.032
## V6  -0.080  0.097  0.005  0.030  0.017 -0.005
## V7  -0.014  0.044 -0.005 -0.003 -0.029 -0.034
```

```
## V8   0.044 -0.021  0.151  0.017 -0.052 -0.037
## V9  -0.020  0.059 -0.031  0.019  0.118  0.051
## V10  0.067 -0.005 -0.005  0.047  0.037  0.068
## V11  1.000  0.006 -0.031  0.044 -0.001  0.035
## V12  0.006  1.000  0.038  0.051  0.017  0.050
## V13 -0.031  0.038  1.000  0.058  0.035  0.057
## V14  0.044  0.051  0.058  1.000  0.062  0.080
## V15 -0.001  0.017  0.035  0.062  1.000  0.057
## V16  0.035  0.050  0.057  0.080  0.057  1.000
```

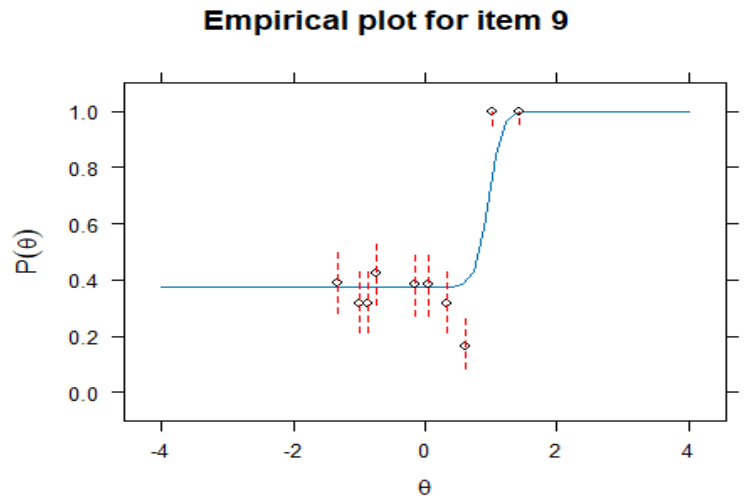
```
diag(xx)<-0
round(sort(abs(xx),decreasing=T)[1:20],2)
```

```
##  [1] 0.40 0.40 0.32 0.32 0.27 0.27 0.26 0.26 0.26 0.26 0.25 0.25 0.25 0.25
0.22
## [16] 0.22 0.19 0.19 0.17 0.17
```

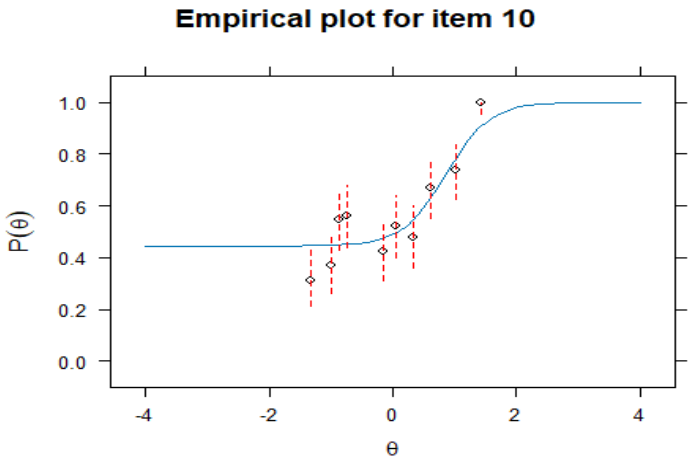
Extra Material:

Plots of Items 9 and 10 with the 3PL

```
itemfit(m3,empirical.plot = 9)
```




```
itemfit(m3,empirical.plot = 10)
```



For fun - do a k-means on the physics data, requesting 3 classes. The code below will run the kmeans, then show the cluster sizes, and then put the cluster means beside the 3PL parameters.

```
kmeans(physics,3) ->xx

xx$size

## [1] 336 144 259

round(cbind(t(xx$centers),coef(m3,IRTpars=T,simplify=T)$items[,c(1:3)]),2)

##         1    2    3     a     b    g
## V1   0.95 0.88 0.89  0.81 -3.30 0.00
## V2   0.85 0.83 0.71  0.58 -2.55 0.00
## V3   0.97 0.96 0.24 41.45 -0.29 0.29
## V4   0.94 0.91 0.04 35.30 -0.24 0.10
## V5   0.91 0.71 0.15  2.80 -0.17 0.14
## V6   0.97 0.15 0.39 45.48  0.31 0.40
## V7   0.93 0.19 0.42 37.18  0.36 0.41
## V8   1.00 1.00 1.00  9.34 -2.94 0.00
## V9   0.63 0.32 0.35 10.59  0.96 0.37
## V10 0.72 0.41 0.45  2.84  0.86 0.45
## V11 0.32 0.40 0.41 -0.17 -3.47 0.01
## V12 0.51 0.22 0.33  1.42  1.70 0.29
## V13 0.95 0.90 0.83  1.02 -2.56 0.00
## V14 0.91 0.69 0.58  1.72 -0.31 0.42
## V15 0.76 0.55 0.52  1.94  0.93 0.53
## V16 0.83 0.74 0.69  0.54 -0.60 0.45
```

And finally, correlate the kmeans and the IRT params:

```
round(cor(cbind(t(xx$centers),coef(m3,IRTpars=T,simplify=T)$items[,c(1:3)])),
2)
```

```
##       1     2     3     a     b     g
## 1  1.00  0.52  0.23  0.43 -0.10 -0.05
## 2  0.52  1.00  0.36 -0.16 -0.58 -0.55
## 3  0.23  0.36  1.00 -0.47 -0.62 -0.31
## a  0.43 -0.16 -0.47  1.00  0.29  0.20
## b -0.10 -0.58 -0.62  0.29  1.00  0.83
## g -0.05 -0.55 -0.31  0.20  0.83  1.00
```