

HUDM6052 Psychometric II Homework_04

Chenguang Pan

2023-11-14

Contents

Q1-a-Parameters-Estimation	1
Q1-a-(1)	4
Q1-a-(2)	4
Q1-a-(3)	5
Q1-a-(4)	6
Q1-a-(5)	7
Q1-b	7

Q1-a-Parameters-Estimation

Fit the 1PL, 2PL, and 3PL models...report the estimated item parameters in separated tables

My Solution:

To make the layout concise and good-looking, I intentionally omitted the codes for data cleaning and some instant display of running outcomes. I attached the estimated item parameters from all three models into one table to save space.

```
> # load the binary response dataset
> library(mirt)
> df <- read.csv("/Users/panpeter/Desktop/PhD_Learning/HUDM6052 Psychometric II/HUDM6052_Psychometric_II.
> df <- df[,-1]
>
> # -----
> #                               Run 1PL
> # -----
>
> # specify the model for 33 items loading on 1 dimension
> # and constrain all the item slope to be equal for 1PL estimation
> spec <- 'F = 1-33
+ CONSTRAIN = (1-33, a1)'
>
> # estimated the model
> # since I constrained all slopes to be equal, here the argument "2PL" is safe
> irt_1pl <- mirt(df, model = spec, itemtype = "2PL", SE=T)
Iteration: 1, Log-Lik: -20276.377, Max-Change: 0.18304Iteration: 2, Log-Lik: -20265.381, Max-Change: 0.0
```

```

Calculating information matrix...
> irt_1pl

Call:
mirt(data = df, model = spec, itemtype = "2PL", SE = T)

Full-information item factor analysis with 1 factor(s).
Converged within 1e-04 tolerance after 25 EM iterations.
mirt version: 1.40
M-step optimizer: BFGS
EM acceleration: Ramsay
Number of rectangular quadrature: 61
Latent density type: Gaussian

Information matrix estimated with method: Oakes
Second-order test: model is a possible local maximum
Condition number of information matrix = 50.40657

Log-likelihood = -20260.22
Estimated parameters: 66
AIC = 40588.44
BIC = 40756.74; SABIC = 40648.75

```

```

> # -----
> #                               Run 2PL
> # -----
> irt_2pl <- mirt(df, model = 1, itemtype = "2PL", SE=T)
Iteration: 1, Log-Lik: -20276.377, Max-Change: 0.63072Iteration: 2, Log-Lik: -20095.717, Max-Change: 0.1

Calculating information matrix...
> irt_2pl

Call:
mirt(data = df, model = 1, itemtype = "2PL", SE = T)

Full-information item factor analysis with 1 factor(s).
Converged within 1e-04 tolerance after 32 EM iterations.
mirt version: 1.40
M-step optimizer: BFGS
EM acceleration: Ramsay
Number of rectangular quadrature: 61
Latent density type: Gaussian

Information matrix estimated with method: Oakes
Second-order test: model is a possible local maximum
Condition number of information matrix = 11.16451

Log-likelihood = -20078.62
Estimated parameters: 66
AIC = 40289.23
BIC = 40615.92; SABIC = 40406.3

```

```

> # -----
> #                               Run 3PL
> # -----
> # specify the model
> spec <- 'F = 1-33
+ PRIOR = (1-33, g, norm, -1.1, 2)'
> irt_3pl <- mirt(df, model = spec, itemtype = "3PL", SE = T)

```

```

> print(irt_3pl)

```

Call:

```
mirt(data = df, model = spec, itemtype = "3PL", SE = T)
```

Full-information item factor analysis with 1 factor(s).
 Converged within 1e-04 tolerance after 136 EM iterations.
 mirt version: 1.40

M-step optimizer: BFGS

EM acceleration: Ramsay

Number of rectangular quadrature: 61

Latent density type: Gaussian

Information matrix estimated with method: Oakes

Second-order test: model is a possible local maximum

Condition number of information matrix = 783.5905

Log-posterior = -20064.48

Estimated parameters: 99

AIC = 40212.78

BIC = 40702.82; SABIC = 40388.38

Items	1PL				2PL				3PL			
	a	b	g	u	a	b	g	u	a	b	g	u
mathc1	0.960	-0.881	0	1	1.078	-0.816	0	1	1.385	-0.250	0.238	1
mathc2	0.960	-1.706	0	1	1.135	-1.516	0	1	1.198	-1.170	0.198	1
mathc3	0.960	-0.162	0	1	1.307	-0.145	0	1	1.529	0.080	0.095	1
mathc4	0.960	-0.412	0	1	1.308	-0.349	0	1	2.095	0.170	0.235	1
mathc5	0.960	1.104	0	1	0.627	1.556	0	1	2.282	1.443	0.187	1
mathc6	0.960	-0.696	0	1	1.352	-0.569	0	1	1.428	-0.432	0.060	1
mathc7	0.960	-0.457	0	1	1.044	-0.434	0	1	1.247	-0.081	0.141	1
mathc8	0.960	-0.562	0	1	0.897	-0.589	0	1	0.951	-0.395	0.071	1
mathc9	0.960	-2.089	0	1	1.415	-1.609	0	1	1.464	-1.443	0.129	1
mathc11	0.960	-0.668	0	1	1.243	-0.571	0	1	1.956	0.092	0.293	1
mathc12	0.960	0.470	0	1	0.856	0.514	0	1	1.130	0.802	0.114	1
mathc13	0.960	-1.212	0	1	1.535	-0.910	0	1	2.546	-0.207	0.360	1
mathc14	0.960	0.552	0	1	0.775	0.655	0	1	0.887	0.819	0.060	1
mathc15	0.960	0.498	0	1	0.726	0.624	0	1	0.858	0.838	0.075	1
mathc16	0.960	-0.216	0	1	0.829	-0.237	0	1	0.942	0.027	0.090	1
mathc17	0.960	0.715	0	1	0.837	0.795	0	1	1.254	1.050	0.125	1
mathc18	0.960	0.767	0	1	1.107	0.688	0	1	1.253	0.757	0.034	1
mathc19	0.960	0.290	0	1	0.551	0.475	0	1	1.432	1.302	0.297	1
mathc21	0.960	-0.004	0	1	0.607	0.011	0	1	3.411	1.149	0.407	1
mathc22	0.960	-0.253	0	1	0.990	-0.250	0	1	1.401	0.295	0.212	1
mathc23	0.960	-0.085	0	1	0.688	-0.099	0	1	1.428	0.863	0.316	1
mathc24	0.960	-0.052	0	1	0.986	-0.053	0	1	1.460	0.460	0.202	1
mathc25	0.960	-0.170	0	1	1.230	-0.155	0	1	1.517	0.132	0.122	1
mathc26	0.960	0.332	0	1	0.893	0.350	0	1	1.177	0.672	0.125	1
mathc27	0.960	-0.282	0	1	1.479	-0.232	0	1	1.719	-0.018	0.094	1
mathc28	0.960	-0.026	0	1	1.949	-0.044	0	1	2.260	0.100	0.058	1
mathc29	0.960	0.119	0	1	1.229	0.092	0	1	1.928	0.476	0.173	1
mathc31	0.960	1.250	0	1	0.723	1.564	0	1	1.699	1.576	0.150	1
mathc32	0.960	-0.476	0	1	0.837	-0.522	0	1	1.600	0.481	0.350	1
mathc33	0.960	0.176	0	1	1.284	0.134	0	1	2.790	0.582	0.217	1
mathc34	0.960	0.165	0	1	0.678	0.229	0	1	0.863	0.645	0.131	1
mathc35	0.960	0.667	0	1	0.360	1.577	0	1	1.480	1.899	0.288	1
mathc36	0.960	1.449	0	1	1.060	1.352	0	1	1.601	1.359	0.070	1

Due to the limitation of `mirt` package, I can't constrain all the α to be 1. Rather, I can only set them to be equal across all the items. Therefore, in the estimation for the 1PL, the estimated universal α is .96 here.

Q1-a-(1)

Does it appear reasonable to assume all the items having an equal slope...

My Solution:

No.

From a aspect of test development, since this is a test about math placement, we should expect that items can discriminate students with different traits well. In addition, comparing the estimated parameters from 2PL model versus 1PL, these items' levels of discrimination spread along a wide range. It is reasonable to have items with higher levels of discrimination than others.

From a mathematics perspective, since the 1PL model is nested in the 2PL model, I conducted the Likelihood Ratio Test to compare the two models as followed:

$$D = -2[\ln(L_{1pl}) - \ln(L_{2pl})].$$

Plug the log likelihood estimated from the above code chunk, then one can have $D = 363.2$ at the degree of freedom of $df = df_{2pl} - df_{1pl} = 66 - 34 = 32$. Based on the Chi-squared distribution, the p value is lower than .001. Therefore, 2PL is better than 1PL, which means the discrimination is preferred.

Q1-a-(2)

Does it appear useful to include a guessing parameter in the model...

My Solution:

Yes, it is useful.

Intuitively, it is reasonable to include a guessing parameter since this a test with multiple choice and guessing is very possible. In addition, by looking through all the guessing parameters, one can find that the `matchc13`, `matchc21`, and `matchc32` do have quite high guessing rate, i.e., all above .30.

However, in terms of model comparison, when using the LRT test again to compare the 2PL vs 3PL model, one can have $D = 28.28$ at 33 degree of freedom, $P = .701$. Based on the parsimony rule, one should endorse the simpler model, i.e., the 2PL.

Therefore, my overall conclusion is including a guessing parameter is useful in this scenario. A practitioner should choose the either model based on their purpose since these two models do not differ a lot.

Q1-a-(3)

Evaluate the goodness of fit of the items with the option of the chi-square test...

My Solution:

I conduct the item fit analysis on each model and summarize the results into one table to make the layout concise.

```
> # get the item fit indices for each model
> item_fit_1pl <- itemfit(irt_1pl, na.rm = T)
> item_fit_2pl <- itemfit(irt_2pl, na.rm = T)
> item_fit_3pl <- itemfit(irt_3pl, na.rm = T)
>
> # combine all the outputs into one table
> item_fit_all <- cbind(item_fit_1pl[,c("item")],
+                       round(item_fit_1pl[,c("S_X2", "p.S_X2")],4),
+                       round(item_fit_2pl[,c("S_X2", "p.S_X2")],4),
+                       round(item_fit_3pl[,c("S_X2", "p.S_X2")],4))
> names(item_fit_all)[1] <- "item"
>
> # get all the item fit indices for 1PL, 2PL, and 3PL model
> item_fit_all
      item      S_X2 p.S_X2      S_X2 p.S_X2      S_X2 p.S_X2
1  mathc1  24.8711 0.4128 23.0566 0.4575 23.8800 0.3536
2  mathc2  28.1234 0.2112 30.4743 0.1074 28.6024 0.1239
3  mathc3  27.2722 0.2919 22.4566 0.4929 21.7064 0.4775
4  mathc4  26.5449 0.3261 19.2254 0.6314 17.4558 0.6831
5  mathc5  97.3404 0.0000 52.8362 0.0014 39.3238 0.0183
6  mathc6  35.2844 0.0643 26.8461 0.2171 23.9949 0.2933
7  mathc7  30.6375 0.1645 30.4324 0.1708 30.8269 0.1271
8  mathc8  31.1674 0.1490 31.5235 0.1393 23.0747 0.4564
9  mathc9  31.5513 0.0854 16.4299 0.6896 12.8670 0.7994
10 mathc11 24.7300 0.4205 17.4787 0.7364 16.8747 0.7187
11 mathc12 23.5755 0.4861 22.8300 0.5298 20.6982 0.5995
12 mathc13 35.9083 0.0422 24.6484 0.2627 19.3900 0.3682
13 mathc14 27.5382 0.2800 23.1188 0.5128 20.5069 0.6112
14 mathc15 33.5261 0.0934 21.0409 0.6903 17.2351 0.7976
15 mathc16 26.6813 0.3195 25.3271 0.3882 22.7550 0.5343
16 mathc17 21.2673 0.6229 17.9679 0.8046 17.1584 0.8014
17 mathc18 21.5747 0.6046 22.3173 0.5012 19.1247 0.6376
```

18	mathc19	54.0475	0.0004	20.7629	0.7058	20.1995	0.6854
19	mathc21	61.7627	0.0000	33.2577	0.1247	23.3450	0.4408
20	mathc22	20.4045	0.6736	20.4046	0.6736	20.8114	0.5926
21	mathc23	20.8365	0.6483	14.4939	0.9524	13.5269	0.9566
22	mathc24	20.9306	0.6428	21.0813	0.6339	21.0628	0.5773
23	mathc25	24.9448	0.4088	21.3330	0.5608	21.4360	0.4939
24	mathc26	19.3543	0.7328	17.6730	0.8186	18.2240	0.7452
25	mathc27	45.0529	0.0057	29.1187	0.1112	27.4429	0.1567
26	mathc28	65.4501	0.0000	24.1670	0.2352	23.9470	0.1982
27	mathc29	20.6986	0.6564	16.3725	0.8389	14.2529	0.8923
28	mathc31	38.2013	0.0242	24.7382	0.4771	22.2939	0.5026
29	mathc32	26.7508	0.3162	22.5725	0.5451	21.6350	0.5424
30	mathc33	29.0666	0.2176	21.2045	0.5686	8.0644	0.9949
31	mathc34	37.3521	0.0403	26.8718	0.3623	24.8798	0.4123
32	mathc35	109.6358	0.0000	32.7703	0.2048	35.3966	0.1033
33	mathc36	28.9987	0.2202	25.1823	0.3410	23.7681	0.3595

Item mathc5 shows bad item fit in all three models. In addition, matchc19, matchc21, matchc27, matchc28, matchc31, matchc34, and 'matchc35 show bad fit in 1PL model only.

Q1-a-(4)

Evaluate the overall fit of the model. Which model do you prefer for this data?...

My Solution:

```
> # get the fit indices for 1PL model
> M2(irt_1pl, na.rm = T)
      M2 df p      RMSEA      RMSEA_5      RMSEA_95      SRMSR      TLI
stats 992.2492 527 0 0.03104487 0.02805368 0.03398001 0.05781072 0.9632371
      CFI
stats 0.9633067
>
> # get the fit indices for 2PL model
> M2(irt_2pl, na.rm = T)
      M2 df      p      RMSEA      RMSEA_5      RMSEA_95      SRMSR
stats 678.0246 495 7.852727e-08 0.02009113 0.01617592 0.02371771 0.03274386
      TLI      CFI
stats 0.9846029 0.9855652
>
> # get the fit indices for 3PL model
> M2(irt_3pl, na.rm = T)
      M2 df      p      RMSEA      RMSEA_5      RMSEA_95      SRMSR
stats 604.7095 462 8.359281e-06 0.01836359 0.01400902 0.02228458 0.03106851
      TLI      CFI
stats 0.9871369 0.9887448
```

Past studies have recommended that a TLI of or above .95, a CFI of or above .95, an RMSEA of or below .05, and an SRMR of or below .05 could indicate a very good fit. Based on those criteria, all three model demonstrate very good model fit.

Next, by conducting the Likelihood Ratio Test (LRT) on 1PL vs. 2PL and 2PL vs. 3PL (finished in Q1-a-(1) and Q1-a-(2)), 2PL is preferred since it is significantly different from 1PL. In addition, LRT shows there is no significant difference in 2PL and 3PL. Based on the parsimony rule, I prefer 2PL model on this data.

Q1-a-(5)

Using the estimates obtained from the model you choose in part...

My Solution:

In the selected 2PL model, `matchc 28` shows the highest level of discrimination with $\alpha = 1.949$, and `matchc 35` has the lowest level of discrimination.

Next, to find the most informative item on a given trait level, I write a function that automatically search for the targeted item.

```
> # write a function to find the most informative item for a given trait
>
> most_info <- function(irt_model, trait){
+   # irt_model can be any estimated IRT model
+   # trait can be any given trait level
+   a <- 0
+   item_num <- 0
+   for (i in 1:33){
+     # extract the item information at a given trait
+     info_temp <- iteminfo(extract.item(irt_model, i), trait)
+     # dynamically updated the best target
+     if (info_temp > a){
+       a <- info_temp
+       item_num <- i
+     }
+   }
+   out <- list(item_num = item_num,
+               info_value = a)
+   return(out)
+ }
```

Then, I applied this function to find the most informative items on the given trait levels of $\theta = -1.5$, $\theta = 0$, and $\theta = 1.5$.

```
> # using a for loop to get all results in one sitting
> for (theta in c(-1.5, 0, 1.5)){
+   theta_out <- most_info(irt_2pl, theta)
+   print(paste0("The most informative item on given trait theta=", theta, " is the item ", theta_out$item_num,
+               " with the infomation value of ", round(theta_out$info_value,4)))
+ }
[1] "The most informative item on given trait theta=-1.5 is the item 9, i.e., the mathc9"
[1] "    with the infomation value of 0.4978"
[1] "The most informative item on given trait theta=0 is the item 26, i.e., the mathc28"
[1] "    with the infomation value of 0.9477"
[1] "The most informative item on given trait theta=1.5 is the item 33, i.e., the mathc36"
[1] "    with the infomation value of 0.2789"
```

As shown above, the most informative item on $\theta = -1.5$ is the item `matchc9`. The most informative item on $\theta = 0$ is the item `matchc28`. The most informative item on $\theta = 1.5$ is the item `matchc36`.

Q1-b

Using the estimates obtained from the model you choose in part...

My Solution:

The 1PL model is reasonable to solve this question since total score is sufficient to estimate the parameters, which means all the students with the same total score will have the same estimated trait level.

Since some of the response vectors contain NA values and `mirt` will impute the missing data, I need to select the students who got 20 in total raw score and are without any missing responses.

```
> # get the estimated person trait
> mirt_trait_out <- fscores(irt_1pl, method = 'EAP', full.scores = F, full.scores.SE = T)

Method: EAP

Empirical Reliability:

      F
0.8481
> mirt_trait <- as.data.frame(mirt_trait_out)
> sum_score <- apply(mirt_trait[,c(1:33)], 1, function(x) sum(x, na.rm = T))
> mirt_trait$sum_score <- sum_score
> trait_20 <- mirt_trait[which(sum_score==20 & complete.cases(mirt_trait[,c(1:33)])),]
> head(trait_20[,c(34:36)])
      F      SE_F sum_score
98  0.4018117 0.3657516      20
270 0.4018117 0.3657516      20
292 0.4018117 0.3657516      20
317 0.4018117 0.3657516      20
323 0.4018117 0.3657516      20
338 0.4018117 0.3657516      20
```

The results show that students who got 20 true score have the same trait, $\theta = .402$. Therefore, we need to select 10 items to have the maximum information on the $\theta = .402$.

Next, I try to get the information values of each item on $\theta = .402$ and sort these values in decreasing order. Finally, select the first ten items to make the shorter test.

```
> info_set <- c()
> # using a for loop to get all info value at this given trait
> for (i in 1:33) {
+   info_tempt <- iteminfo(extract.item(irt_1pl, i), .4018117)
+   info_set[i] <- info_tempt
+ }
> # make a new df
> info_matrix <- data.frame(
+   item = item_fit_1pl[,c("item")],
+   info_value = info_set
+ )
> # sort this df in decreasing order
> info_matrix <- info_matrix[order(-info_matrix$info_value),]
> # get the first 10 items
> info_matrix[c(1:10),]
      item info_value
11 mathc12  0.2303953
24 mathc26  0.2303806
14 mathc15  0.2301504
18 mathc19  0.2299760
```



```

13 mathc14 0.2294367
30 mathc33 0.2279574
31 mathc34 0.2276869
32 mathc35 0.2269344
27 mathc29 0.2264486
16 mathc17 0.2254932

```

As shown above, I will select these ten items to create a shorter version of test.

```

> df_short <- df[,which(names(df) %in% info_matrix$item[1:10])]
> # fit 1PL on this short test
> spec <- 'F = 1-10
+ CONSTRAIN = (1-10, a1)'
>
> irt_1pl_short <- mirt(df_short, model = spec, itemtype = "2PL", SE=T)
Iteration: 1, Log-Lik: -6666.931, Max-Change: 0.07529Iteration: 2, Log-Lik: -6664.492, Max-Change: 0.003

Calculating information matrix...
> irt_1pl_short

Call:
mirt(data = df_short, model = spec, itemtype = "2PL", SE = T)

Full-information item factor analysis with 1 factor(s).
Converged within 1e-04 tolerance after 9 EM iterations.
mirt version: 1.40
M-step optimizer: BFGS
EM acceleration: Ramsay
Number of rectangular quadrature: 61
Latent density type: Gaussian

Information matrix estimated with method: Oakes
Second-order test: model is a possible local maximum
Condition number of information matrix = 9.243253

Log-likelihood = -6664.226
Estimated parameters: 20
AIC = 13350.45
BIC = 13404.88; SABIC = 13369.94

```

Now to get the person trait estimation.

```

> # get the estimated person trait
> mirt_trait_out <- fscores(irt_1pl_short, method = 'EAP', full.scores = F, full.scores.SE = T)

Method: EAP

Empirical Reliability:

      F
0.5769
> mirt_trait <- as.data.frame(mirt_trait_out)
> mirt_trait <- round(mirt_trait[which(complete.cases(mirt_trait)),c(11:12)],2)

```

```
> head(mirt_trait)
      F SE_F
1 -1.50 0.72
2 -1.10 0.69
3 -1.10 0.69
4 -0.72 0.66
6 -1.10 0.69
7 -0.72 0.66
> head(mirt_trait[which(mirt_trait$F == .40),])
[1] F      SE_F
<0 rows> (or 0-length row.names)
```