

Generalized Measurement Models: Modeling Observed Polytomous Data

Lecture 4d

Today's Lecture Objectives

1. Show how to estimate unidimensional latent variable models with polytomous data*
 - Also known as Polytomous Item Repsonse Theory (IRT) or Item Factor Analysis (IFA) models
2. Distributions appropriate for polytomous (discrete; data with lower/upper limits)

Example Data: Conspiracy Theories

Today's example is from a bootstrap resample of 177 undergraduate students at a large state university in the Midwest. The survey was a measure of 10 questions about their beliefs in various conspiracy theories that were being passed around the internet in the early 2010s. Additionally, gender was included in the survey. All items responses were on a 5- point Likert scale with:

1. Strongly Disagree
2. Disagree
3. Neither Agree or Disagree
4. Agree
5. Strongly Agree

PLEASE NOTE, THE PURPOSE OF THIS SURVEY WAS TO STUDY INDIVIDUAL BELIEFS REGARDING CONSPIRACIES. THE QUESTIONS CAN PROVOKE SOME STRONG EMOTIONS GIVEN THE WORLD WE LIVE IN CURRENTLY. ALL QUESTIONS WERE APPROVED BY UNIVERSITY IRB PRIOR TO THEIR USE.

Our purpose in using this instrument is to provide a context that we all may find

relevant as many of these conspiracy theories are still prevalent today.

Conspiracy Theory Questions 1-5

Questions:

1. The U.S. invasion of Iraq was not part of a campaign to fight terrorism, but was driven by oil companies and Jews in the U.S. and Israel.
2. Certain U.S. government officials planned the attacks of September 11, 2001 because they wanted the United States to go to war in the Middle East.
3. President Barack Obama was not really born in the United States and does not have an authentic Hawaiian birth certificate.
4. The current financial crisis was secretly orchestrated by a small group of Wall Street bankers to extend the power of the Federal Reserve and further their control of the world's economy.
5. Vapor trails left by aircraft are actually chemical agents deliberately sprayed in a clandestine program directed by government officials.

Conspiracy Theory Questions 6-10

Questions:

6. Billionaire George Soros is behind a hidden plot to destabilize the American government, take control of the media, and put the world under his control.
7. The U.S. government is mandating the switch to compact fluorescent light bulbs because such lights make people more obedient and easier to control.
8. Government officials are covertly Building a 12-lane "NAFTA superhighway" that runs from Mexico to Canada through America's heartland.
9. Government officials purposely developed and spread drugs like crack-cocaine and diseases like AIDS in order to destroy the African American community.
10. God sent Hurricane Katrina to punish America for its sins.

From Previous Lectures: CFA (Normal Outcomes)

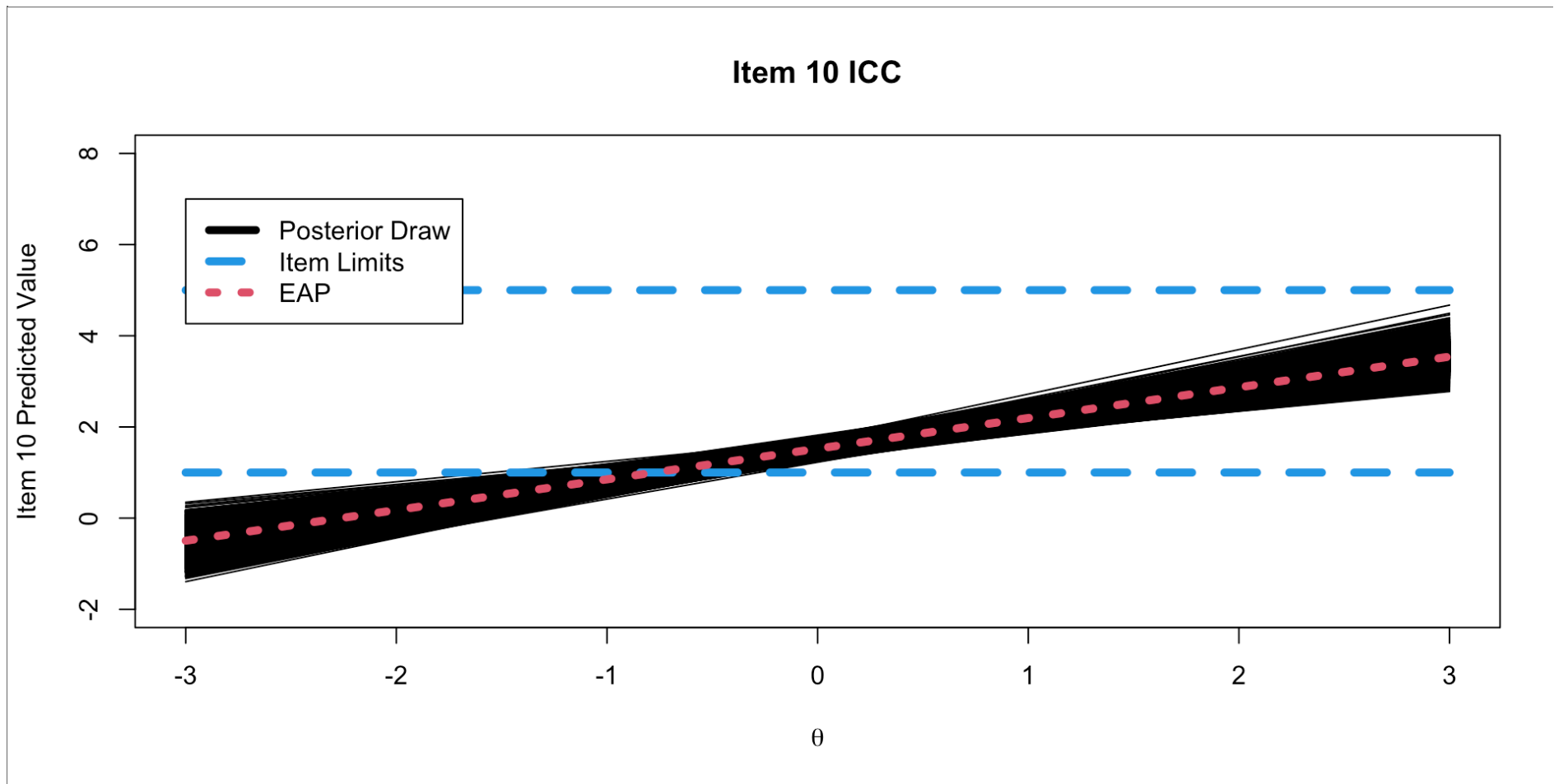
For comparisons today, we will be using the model where we assumed each outcome was (conditionally) normally distributed:

For an item i the model is:

$$Y_{pi} = \mu_i + \lambda_i\theta_p + e_{p,i}; \quad e_{p,i} \sim N(0, \psi_i^2)$$

Recall that this assumption wasn't a good one as the type of data (discrete, bounded, some multimodality) did not match the normal distribution assumption

Plotting CFA (Normal Outcome) Results



Polytomous Data Distributions

Polytomous Data Characteristics

As we have done with each observed variable, we must decide which distribution to use

- To do this, we need to map the characteristics of our data on to distributions that share those characteristics

Our observed data:

- Discrete responses
- Small set of known categories: 1, 2, 3, 4, 5
- Some observed item responses may be multimodal

Choice of distribution must match

- Be capable of modeling a small set of known categories
 - Discrete distribution
 - Limited number of categories (with fixed upper bound)
- Possible multimodality

Discrete Data Distributions

Stan has a list of distributions for bounded discrete data: <https://mc-stan.org/docs/functions-reference/bounded-discrete-distributions.html>

- Binomial distribution
 - Pro: Easy to use/code
 - Con: Unimodal distribution
- Beta-binomial distribution
 - Not often used in psychometrics (but could be)
 - Generalizes binomial distribution to have different probability for each trial
- Hypergeometric distribution
 - Not often used in psychometrics
- Categorical distribution (sometimes called multinomial)
 - Most frequently used
 - Base distribution for graded response, partial credit, and nominal response models

- Discrete range distribution (sometimes called uniform)
 - Not useful—doesn't have much information about latent variables

Binomial Distribution Models

Binomial Distribution Models

The binomial distribution is one of the easiest to use for polytomous items

- However, it assumes the distribution of responses are unimodal

Binomial probability mass function (i.e., pdf):

$$P(Y = y) = \binom{n}{y} p^y (1 - p)^{(n-y)}$$

Parameters:

- n – “number of trials” (range: $n \in \{0, 1, \dots\}$)
- y – “number of successes” out of n “trials” (range: $y \in \{0, 1, \dots, n\}$)
- p – probability of “success” (range: $[0, 1]$)

Mean: np

Variance: $np(1 - p)$

Adapting the Binomial for Item Response Models

Although it doesn't seem like our items fit with a binomial, we can actually use this distribution

- Item response: number of successes y_i
 - Needed: recode data so that lowest category is 0 (subtract one from each item)
- Highest (recoded) item response: number of trials n
 - For all our items, once recoded, $n_i = 4$ ($\forall i$)
- Then, use a link function to model each item's p_i as a function of the latent trait:

$$p_i = \frac{\exp(\mu_i + \lambda_i \theta_p)}{1 + \exp(\mu_i + \lambda_i \theta_p)}$$

Note:

- Shown with a logit link function (but could be any link)
- Shown in slope/intercept form (but could be discrimination/difficulty for unidimensional items)
- Could also include asymptote parameters (c_i or d_i)

Binomial Item Repsonse Model

The item response model, put into the PDF of the binomial is then:

$$P(Y_{pi} | \theta_p) = \binom{n_i}{Y_{pi}} \left(\frac{\exp(\mu_i + \lambda_i \theta_p)}{1 + \exp(\mu_i + \lambda_i \theta_p)} \right)^{Y_{pi}} \left(1 - \left(\frac{\exp(\mu_i + \lambda_i \theta_p)}{1 + \exp(\mu_i + \lambda_i \theta_p)} \right) \right)^{(n_i - Y_{pi})}$$

Further, we can use the same priors as before on each of our item parameters

- μ_i : Normal prior $N(0, 1000)$
- λ_i Normal prior $N(0, 1000)$

Likewise, we can identify the scale of the latent variable as before, too:

- $\theta_p \sim N(0, 1)$

Estimating the Binomial Model in Stan

```

1 model {
2
3   lambda ~ multi_normal(meanLambda, covLambda); // Prior for item discrimination/factor loadings
4   mu ~ multi_normal(meanMu, covMu);             // Prior for item intercepts
5
6   theta ~ normal(0, 1);                          // Prior for latent variable (with mean/sd specified)
7
8   for (item in 1:nItems){
9     Y[item] ~ binomial(maxItem[item], inv_logit(mu[item] + lambda[item]*theta));
10  }
11
12 }
```

Here, the binomial function has two arguments:

- The first (`maxItem[item]`) is the number of “trials” n_i (here, our maximum score minus one)
- The second `inv_logit(mu[item] + lambda[item]*theta)` is the probability from our model (p_i)

The data `Y[item]` must be:

- Type: integer
- Range: 0 through `maxItem[item]`

Binomial Model Parameters Block

```
1 parameters {  
2   vector[nObs] theta;           // the latent variables (one for each person)  
3   vector[nItems] mu;           // the item intercepts (one for each item)  
4   vector[nItems] lambda;       // the factor loadings/item discriminations (one for each item)  
5 }
```

No changes from any of our previous slope/intercept models

Binomial Model Data Block

```

1 data {
2   int<lower=0> nObs;           // number of observations
3   int<lower=0> nItems;        // number of items
4   array[nItems] int<lower=0> maxItem;
5
6   array[nItems, nObs] int<lower=0> Y; // item responses in an array
7
8   vector[nItems] meanMu;      // prior mean vector for intercept parameters
9   matrix[nItems, nItems] covMu; // prior covariance matrix for intercept parameters
10
11  vector[nItems] meanLambda;   // prior mean vector for discrimination parameters
12  matrix[nItems, nItems] covLambda; // prior covariance matrix for discrimination parameters
13 }

```

Note:

- Need to supply **maxItem** (maximum score minus one for each item)
- The data are the same (integer) as in the binary/dichotomous items syntax

Preparing Data for Stan

```
1 # note: data must start at zero
2 conspiracyItemsBinomial = conspiracyItems
3 for (item in 1:ncol(conspiracyItemsBinomial)){
4   conspiracyItemsBinomial[, item] = conspiracyItemsBinomial[, item] - 1
5 }
6
7 # check first item
8 table(conspiracyItemsBinomial[,1])
```

```
0 1 2 3 4
49 51 47 23 7
```

```
1 # determine maximum value for each item
2 maxItem = apply(X = conspiracyItemsBinomial,
3                 MARGIN = 2,
4                 FUN = max)
```

Binomial Model Stan Call

```
1 modelBinomial_samples = modelBinomial_stan$sample(  
2   data = modelBinomial_data,  
3   seed = 12112022,  
4   chains = 4,  
5   parallel_chains = 4,  
6   iter_warmup = 5000,  
7   iter_sampling = 5000,  
8   init = function() list(lambda=rnorm(nItems, mean=5, sd=1))  
9 )
```


Binomial Model Results

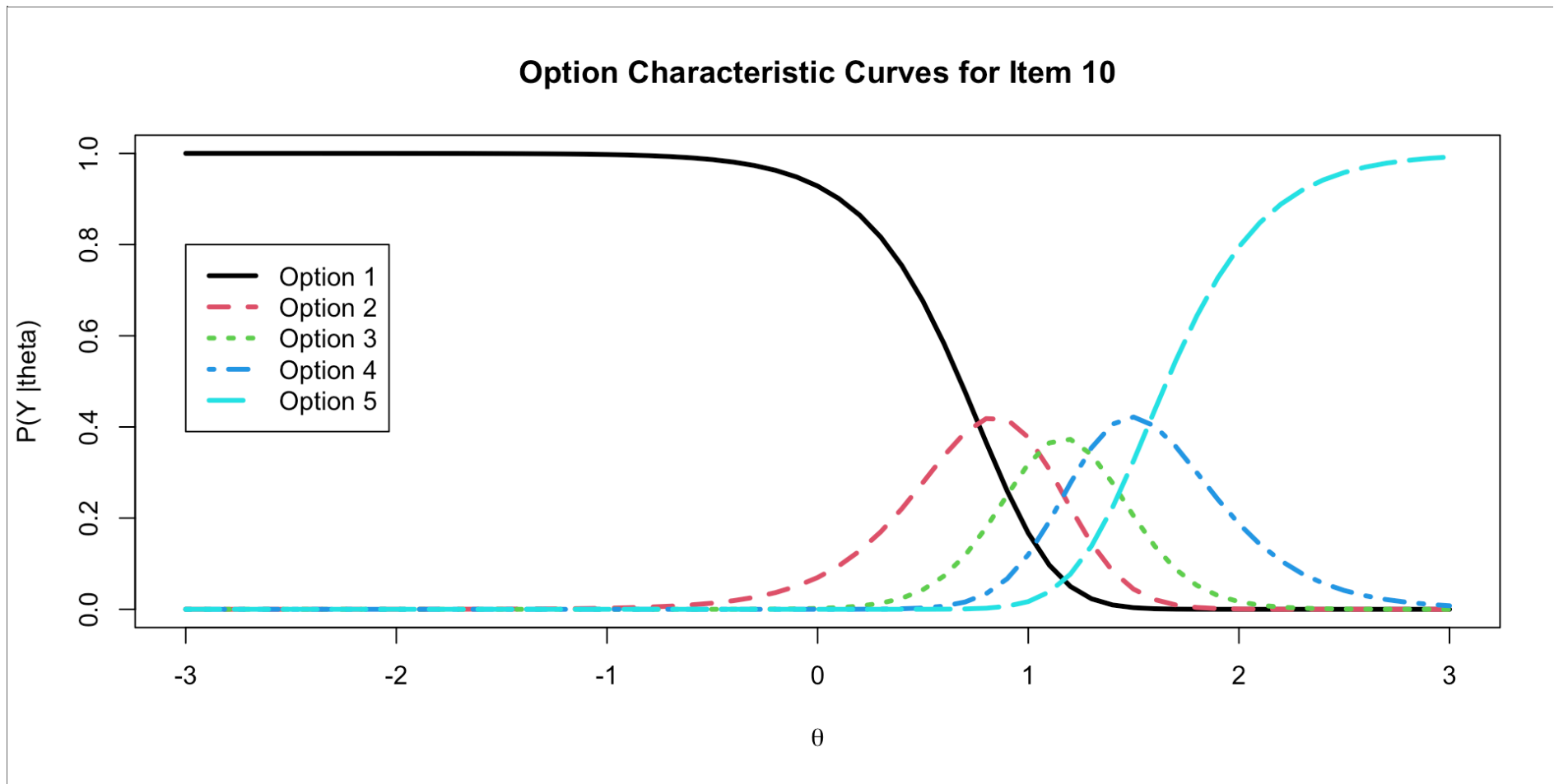
```
1 # checking convergence
2 max(modelBinomial_samples$summary())$rhat, na.rm = TRUE)
```

```
[1] 1.000957
```

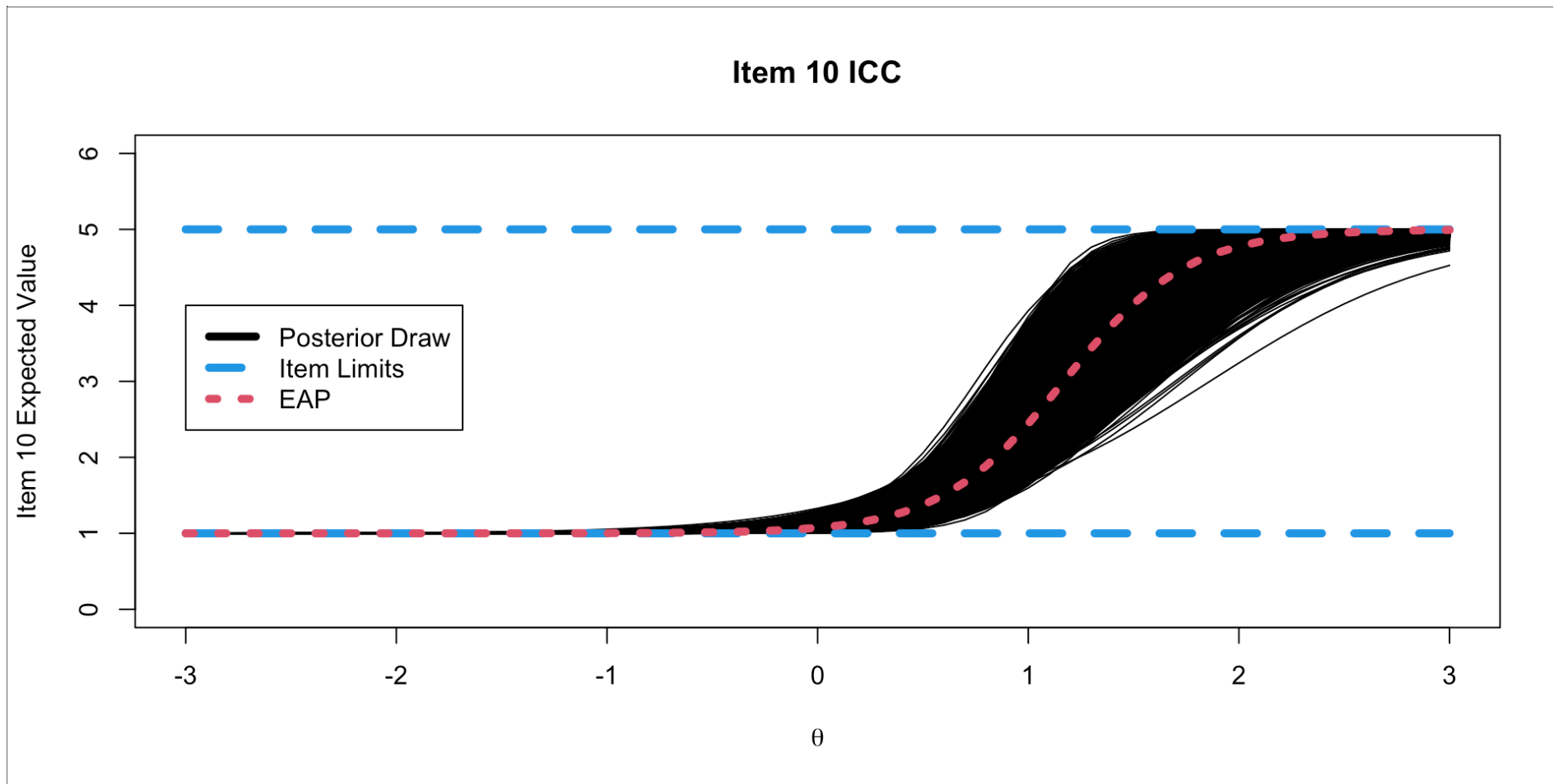
```
1 # item parameter results
2 print(modelBinomial_samples$summary(variables = c("mu", "lambda"))) ,n=Inf)
```

```
# A tibble: 20 × 10
  variable      mean median    sd    mad    q5    q95  rhat ess_bulk ess_tail
  <chr>      <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>   <dbl>   <dbl>
1 mu[1]    -0.842 -0.839 0.127 0.124 -1.05 -0.636 1.00   2527.   5864.
2 mu[2]    -1.84  -1.83 0.214 0.213 -2.20 -1.50  1.00   2413.   5088.
3 mu[3]    -1.99  -1.98 0.223 0.220 -2.38 -1.64  1.00   2461.   6559.
4 mu[4]    -1.73  -1.72 0.210 0.208 -2.08 -1.40  1.00   1896.   4566.
5 mu[5]    -2.00  -1.99 0.252 0.253 -2.44 -1.61  1.00   2233.   5590.
6 mu[6]    -2.10  -2.09 0.243 0.242 -2.52 -1.72  1.00   2176.   5810.
7 mu[7]    -2.45  -2.43 0.265 0.262 -2.90 -2.03  1.00   2360.   6116.
8 mu[8]    -2.16  -2.15 0.239 0.237 -2.57 -1.79  1.00   2162.   5316.
9 mu[9]    -2.40  -2.39 0.278 0.275 -2.88 -1.97  1.00   2506.   6622.
10 mu[10]   -3.98  -3.94 0.519 0.502 -4.89 -3.19  1.00   2823.   7291.
11 lambda[1]  1.11   1.10 0.143 0.142  0.881  1.35  1.00   5976.   9415.
12 lambda[2]  1.90   1.89 0.244 0.238  1.52   2.33  1.00   5167.   8013.
13 lambda[3]  1.92   1.90 0.256 0.253  1.52   2.36  1.00   4960.   7639.
14 lambda[4]  1.89   1.88 0.239 0.234  1.52   2.31  1.00   3959.   5469.
15 lambda[5]  2.28   2.26 0.284 0.282  1.84   2.77  1.00   4135.   5926.
16 lambda[6]  2.15   2.13 0.275 0.271  1.73   2.63  1.00   4391.   7256.
17 lambda[7]  2.13   2.11 0.295 0.288  1.68   2.65  1.00   4773.   6502.
18 lambda[8]  2.08   2.07 0.268 0.268  1.66   2.54  1.00   3914.   5351.
19 lambda[9]  2.34   2.33 0.316 0.311  1.86   2.89  1.00   4729.   7527.
20 lambda[10] 3.40   3.36 0.557 0.544  2.57   4.38  1.00   4772.   7579.
```

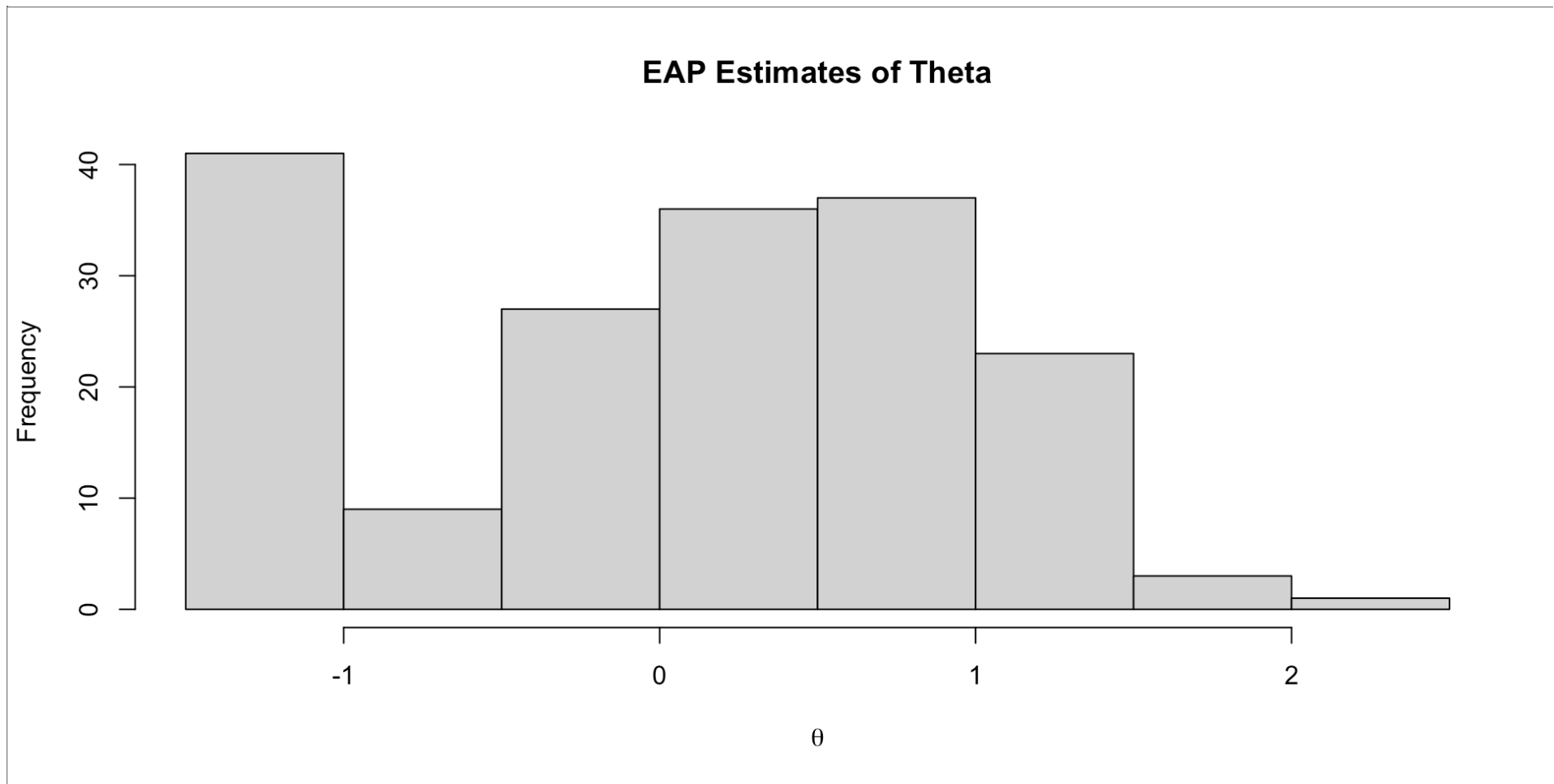

Option Characteristic Curves



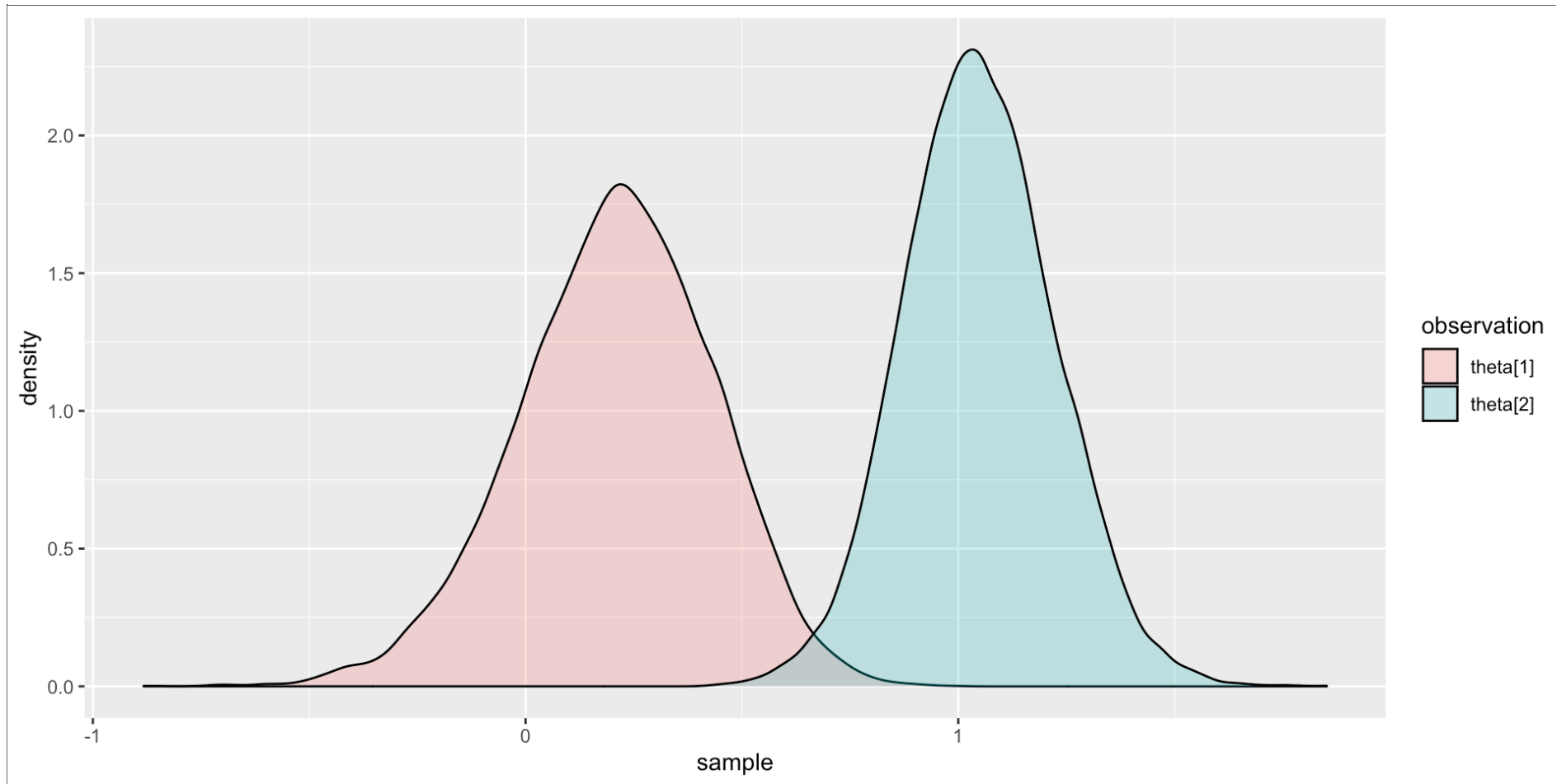
ICC Plots



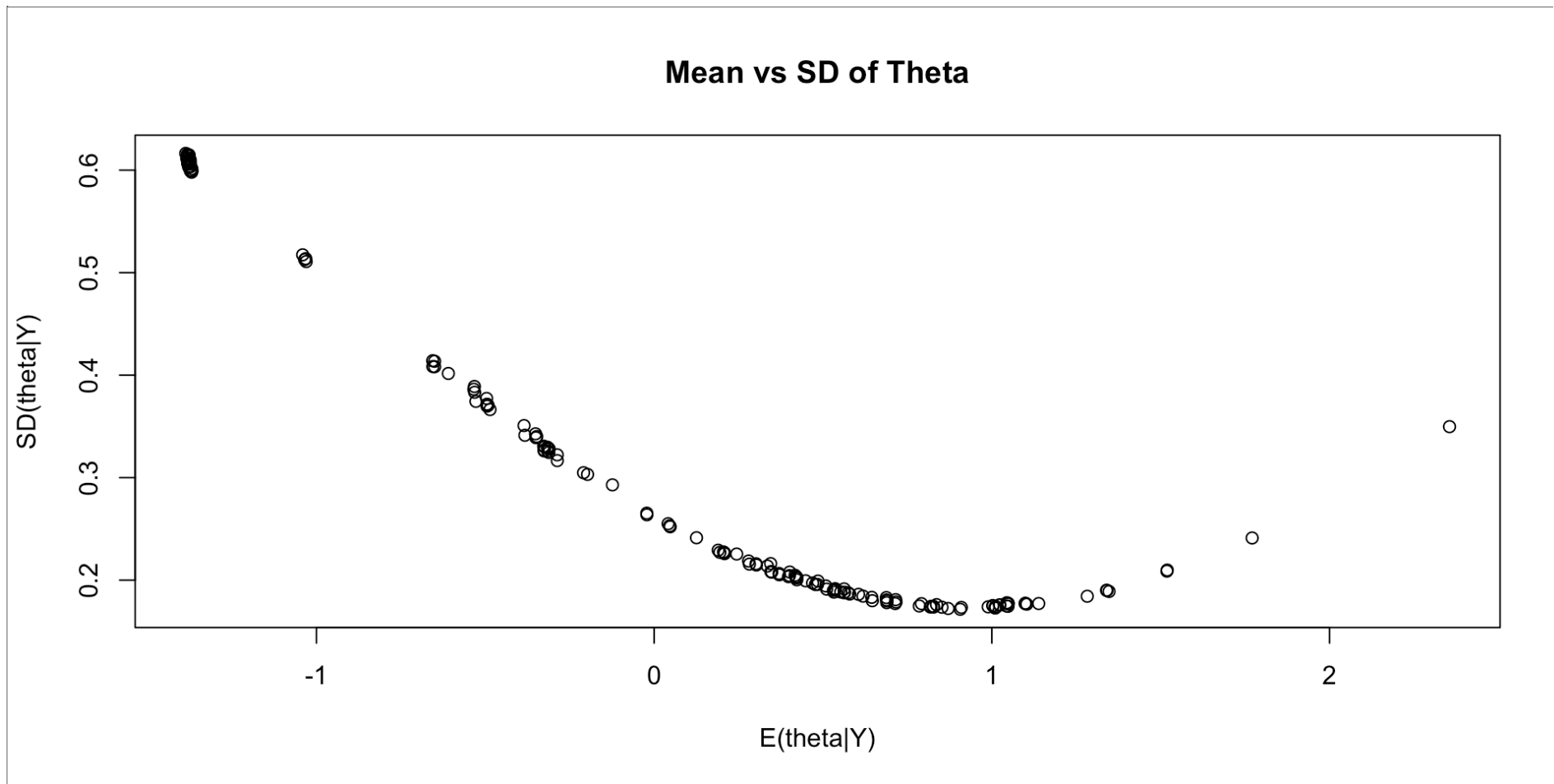
Investigating Latent Variable Estimates



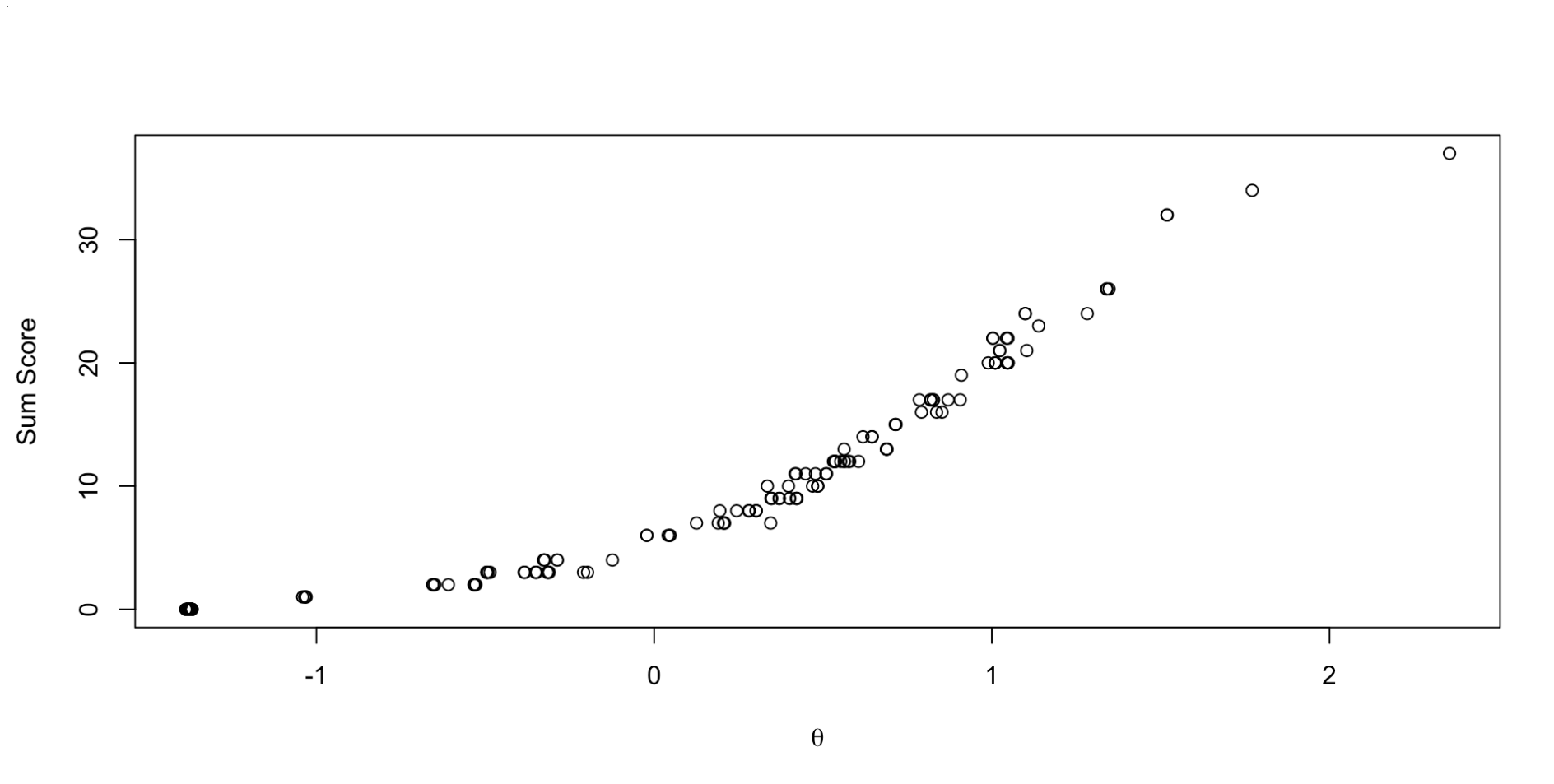
Comparing Two Latent Variable Posterior Distributions



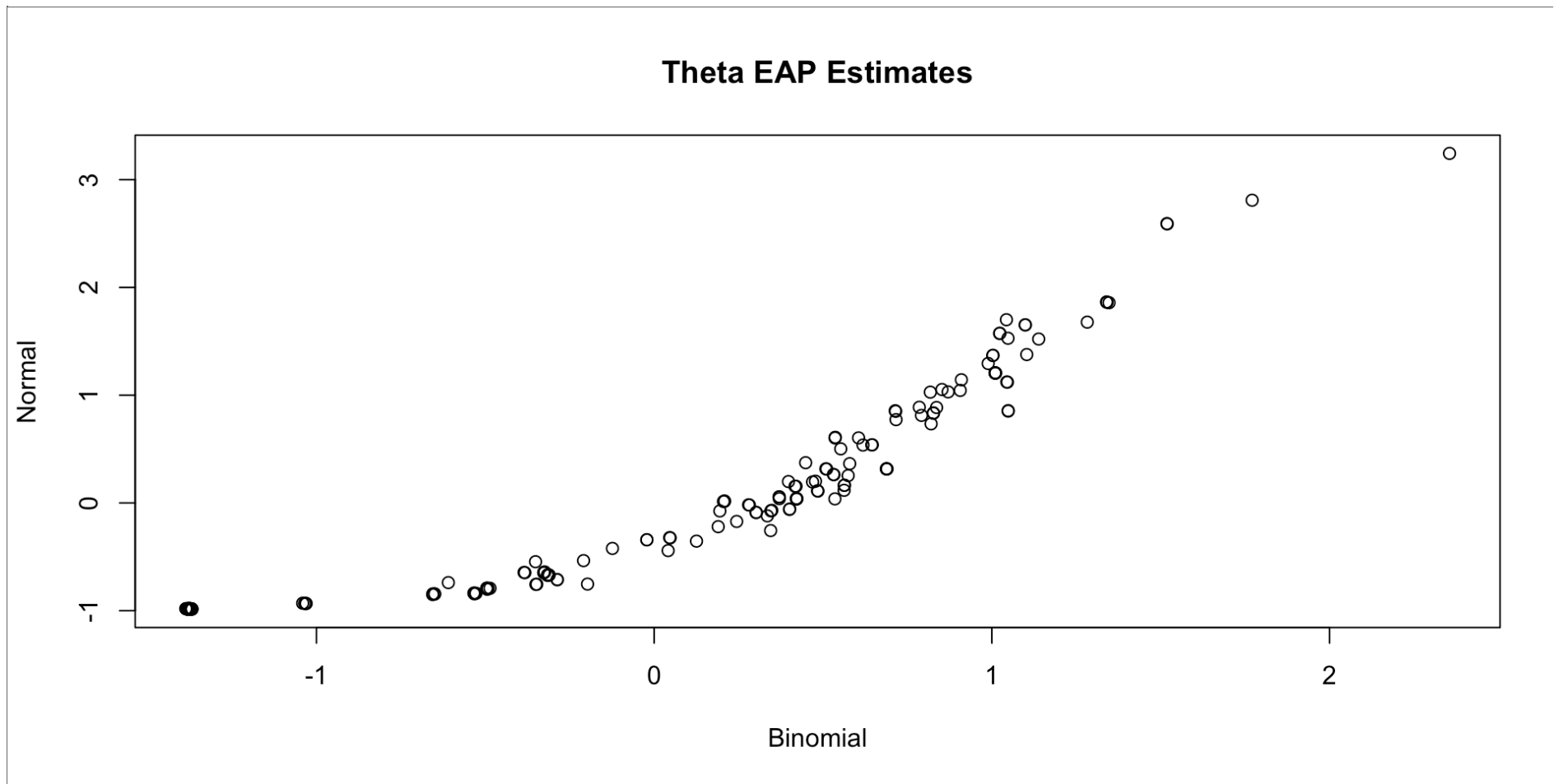
Comparing Latent Variable Posterior Mean and SDs



Comparing EAP Estimates with Sum Scores



Comparing Thetas: Binomial vs Normal



Categorical/Multinomial Distribution Models

Categorical/Multinomial Distribution Models

Although the binomial distribution is easy, it may not fit our data well

- Instead, we can use the categorical (sometimes called multinomial) distribution, with pmf (pdf):

$$P(Y = y) = \frac{n!}{y_1! \cdots y_C!} p_1^{y_1} \cdots p_C^{y_C}$$

Here:

- n : number of “trials”
- y_c : number of events in each of c categories ($c \in \{1, \dots, C\}$; $\sum_c y_c = n$)
- p_c : probability of observing an event in category c

Adapting the Multinomial Distribution for Item Response Models

With some definitions, we can make a multinomial distribution into one we can use for polytomous item response models

- The number of “trials” is set to one for all items $n_i = 1$ ($\forall i$)
 - With $n_i = 1$, this is called the categorical distribution

$$P(Y_{pi} \mid \theta_p) = p_{i1}^{I(y_{pi}=1)} \cdots p_{iC_i}^{I(y_{pi}=C_i)}$$

- The number of categories is equal to the number of options on an item (C_i)
- The item response model is specified for the set of probabilities p_{ic} , with $\sum_c p_{ic} = 1$
 - Then, use a link function to model each item’s set of p_{ic} as a function of the latent trait

Choices for Models for Probability of Each Category p_{ic}

The most-frequently used polytomous item response models all use the categorical distribution for observed data

- They differ in how the model function builds the conditional response probabilities
 - Graded response models: set of ordered intercepts (or thresholds) and a single loading
 - Called proportional odds models in categorical data analysis
 - Partial credit models: set of unordered difficulty parameters and a single loading
 - Nominal response models: set of unordered intercepts and set of loadings
 - Called generalized logit models in categorical data analysis
- Terminology note:
 - Terms graded response and partial credit come from educational measurement
 - Data need not be graded response/partial credit to you

Graded Response Model

The graded response model is an ordered logistic regression model where:

$$P(Y_{ic} | \theta) = \begin{cases} 1 - P^*(Y_{i1} | \theta) & \text{if } c = 1 \\ P^*(Y_{ic-1} | \theta) - P^*(Y_{ic} | \theta) & \text{if } 1 < c < C_i \\ P^*(Y_{iC_i-1} | \theta) & \text{if } c = C_i \end{cases}$$

Where:

$$P^*(Y_{ic} | \theta) = \frac{\exp(\mu_{ic} + \lambda_i \theta_p)}{1 + \exp(\mu_{ic} + \lambda_i \theta_p)}$$

With:

- $C_i - 1$ Ordered intercepts: $\mu_1 > \mu_2 > \dots > \mu_{C_i-1}$

Estimating the Graded Response Model in Stan

```

1 model {
2
3   lambda ~ multi_normal(meanLambda, covLambda); // Prior for item discrimination/factor loadings
4   theta ~ normal(0, 1); // Prior for latent variable (with mean/sd specified)
5
6   for (item in 1:nItems){
7     thr[item] ~ multi_normal(meanThr[item], covThr[item]); // Prior for item intercepts
8     Y[item] ~ ordered_logistic(lambda[item]*theta, thr[item]);
9   }
10
11 }

```

Notes:

- `ordered_logistic` is a built in Stan function that makes the model easy to implement
 - Instead of intercepts, however, it uses thresholds: $\eta_{ic} = -\theta_{ic}$
 - First argument is the linear predictor (the non-intercept portion of the model)
 - Second argument is the set of thresholds for the item
- The function expects the responses of Y to start at one and go to maxCategory ($Y_{pi} \in \{1, \dots, C_i\}$)
 - No transformation needed to data (unless some categories have zero

observations)

Graded Response Model **parameters** Block

```

1 parameters {
2   vector[nObs] theta;           // the latent variables (one for each person)
3   array[nItems] ordered[maxCategory-1] thr; // the item thresholds (one for each item category minus one)
4   vector[nItems] lambda;       // the factor loadings/item discriminations (one for each item)
5 }

```

Notes:

- Threshold parameters: **array[nItems] ordered[maxCategory-1] thr;**
 - Is an array (each item has maxCategory-1) parameters
 - Is of type **ordered**: Automatically ensures order is maintained

$$\tau_{i1} < \tau_{i2} < \dots < \tau_{iC-1}$$

Graded Response Model **generated quantities** Block

```
1 generated quantities{  
2   array[nItems] vector[maxCategory-1] mu;  
3   for (item in 1:nItems){  
4     mu[item] = -1*thr[item];  
5   }  
6 }
```

We use generated quantities to convert threshold parameters into intercepts

Graded Response Model **data** block

```

1 data {
2   int<lower=0> nObs;           // number of observations
3   int<lower=0> nItems;        // number of items
4   int<lower=0> maxCategory;
5
6   array[nItems, nObs] int<lower=1, upper=5> Y; // item responses in an array
7
8   array[nItems] vector[maxCategory-1] meanThr; // prior mean vector for intercept parameters
9   array[nItems] matrix[maxCategory-1, maxCategory-1] covThr; // prior covariance matrix for intercept p
10
11  vector[nItems] meanLambda; // prior mean vector for discrimination parameters
12  matrix[nItems, nItems] covLambda; // prior covariance matrix for discrimination parameters
13 }

```

Notes:

- The input for the prior mean/covariance matrix for threshold parameters is now an array (one mean vector and covariance matrix per item)

Graded Response Model Data Preparation

To match the array for input for the threshold hyperparameter matrices, a little data manipulation is needed

```
1 # item threshold hyperparameters
2 thrMeanHyperParameter = 0
3 thrMeanVecHP = rep(thrMeanHyperParameter, maxCategory-1)
4 thrMeanMatrix = NULL
5 for (item in 1:nItems){
6   thrMeanMatrix = rbind(thrMeanMatrix, thrMeanVecHP)
7 }
8
9 thrVarianceHyperParameter = 1000
10 thrCovarianceMatrixHP = diag(x = thrVarianceHyperParameter, nrow = maxCategory-1)
11 thrCovArray = array(data = 0, dim = c(nItems, maxCategory-1, maxCategory-1))
12 for (item in 1:nItems){
13   thrCovArray[item, , ] = diag(x = thrVarianceHyperParameter, nrow = maxCategory-1)
14 }
```

- The R array matches stan's array type

Graded Response Model Stan Call

```
1 modelOrderedLogit_samples = modelOrderedLogit_stan$sample(  
2   data = modelOrderedLogit_data,  
3   seed = 121120221,  
4   chains = 4,  
5   parallel_chains = 4,  
6   iter_warmup = 5000,  
7   iter_sampling = 5000,  
8   init = function() list(lambda=rnorm(nItems, mean=5, sd=1))  
9 )
```

Note: Using positive starting values for the λ parameters

Graded Response Model Results

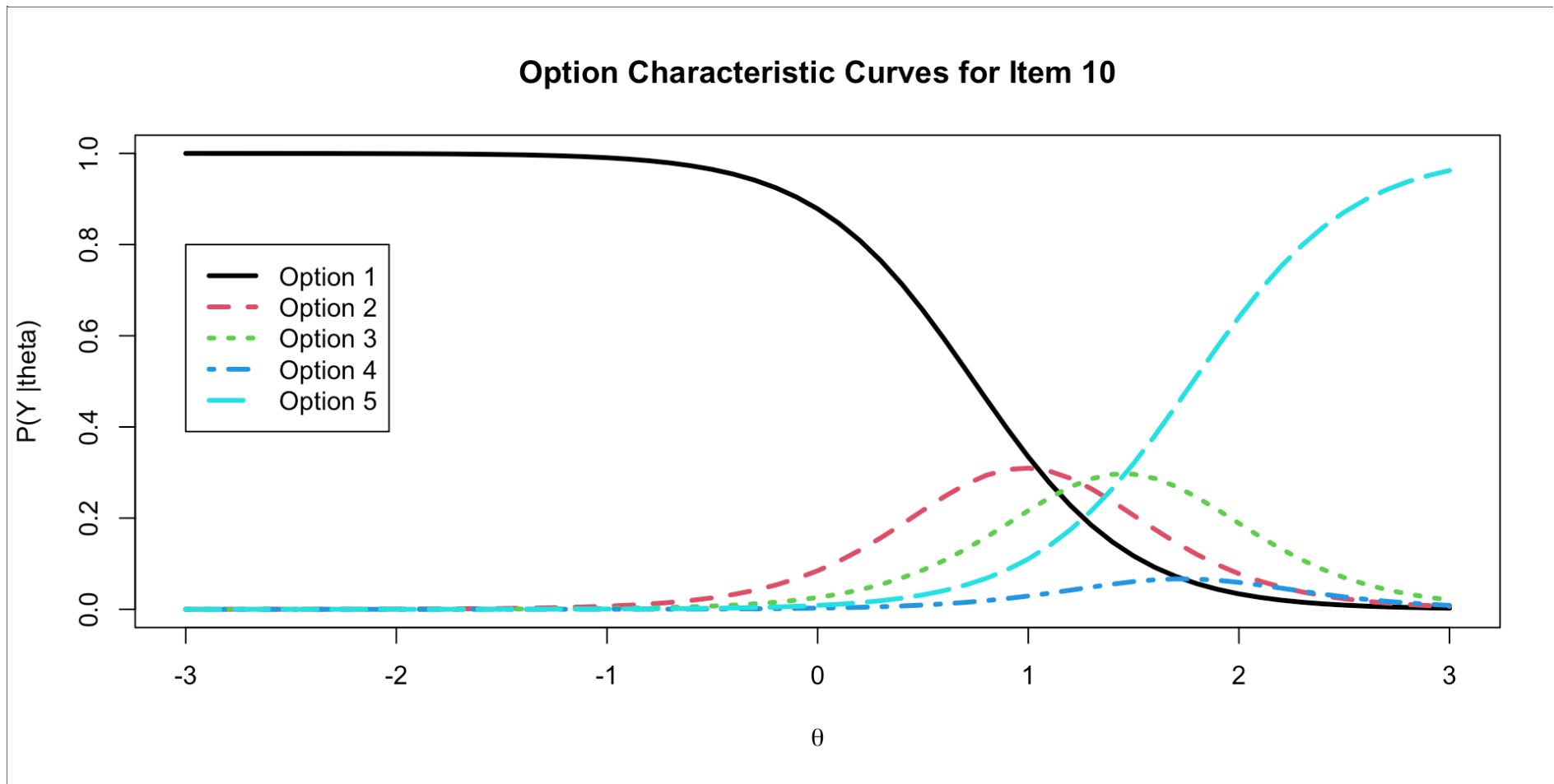
```
1 # checking convergence
2 max(modelOrderedLogit_samples$summary())$rhat, na.rm = TRUE)
```

```
[1] 1.002496
```

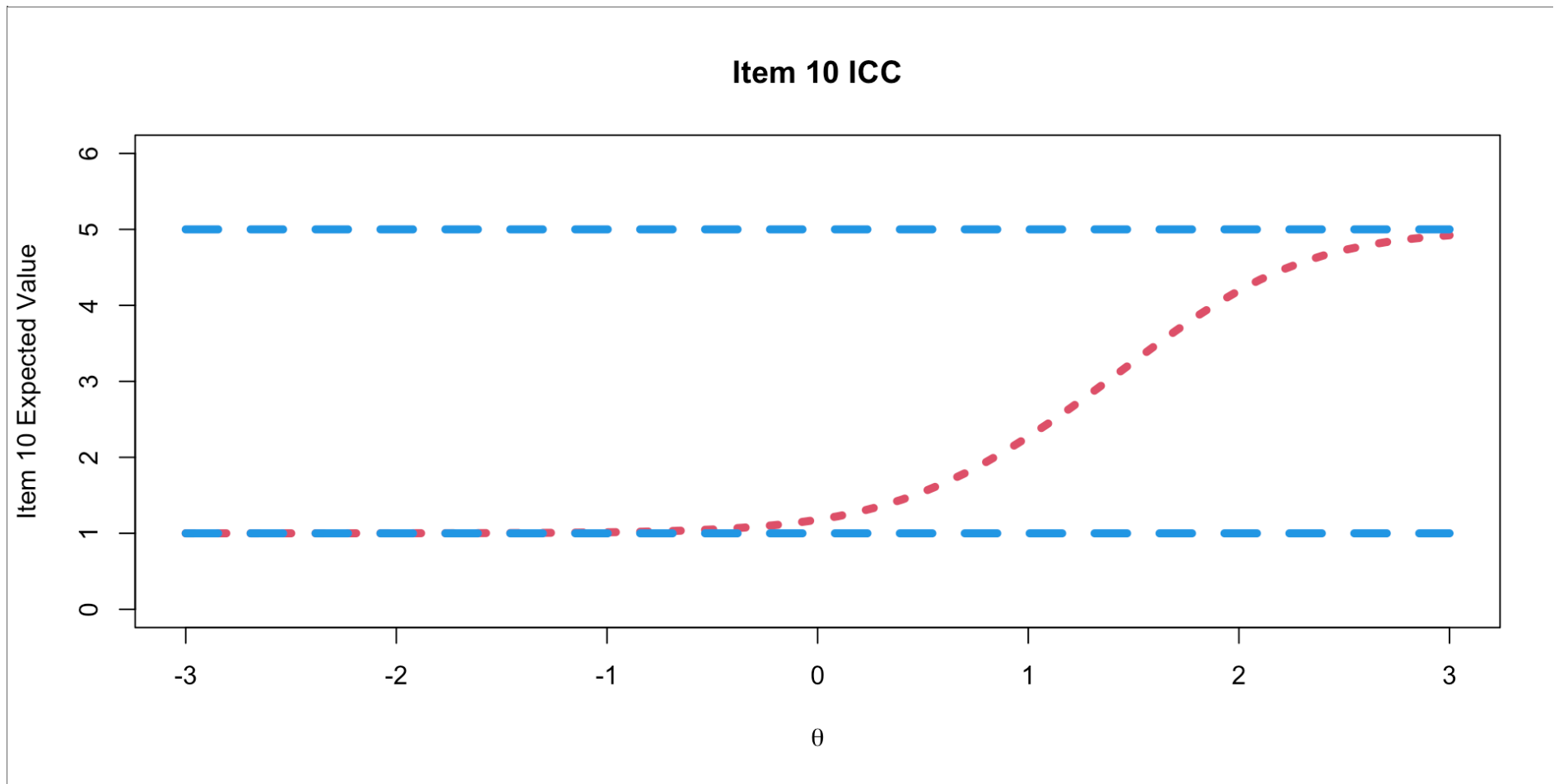
```
1 # item parameter results
2 print(modelOrderedLogit_samples$summary(variables = c("lambda", "mu")) ,n=Inf)
```

```
# A tibble: 50 × 10
  variable      mean median      sd      mad      q5      q95 rhat ess_bulk
  <chr>      <dbl>   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>   <dbl>
1 lambda[1]    2.14    2.13  0.289 0.285    1.69  2.63    1.00   5439.
2 lambda[2]    3.08    3.05  0.432 0.424    2.42  3.84    1.00   5761.
3 lambda[3]    2.60    2.57  0.389 0.380    2.00  3.27    1.00   5731.
4 lambda[4]    3.11    3.08  0.429 0.425    2.45  3.85    1.00   5565.
5 lambda[5]    5.02    4.95  0.768 0.744    3.88  6.37    1.00   4909.
6 lambda[6]    5.04    4.98  0.761 0.745    3.92  6.39    1.00   4960.
7 lambda[7]    3.25    3.22  0.500 0.485    2.49  4.12    1.00   5283.
8 lambda[8]    5.37    5.28  0.863 0.829    4.12  6.91    1.00   5165.
9 lambda[9]    3.15    3.12  0.484 0.476    2.42  4.00    1.00   5564.
10 lambda[10]  2.67    2.64  0.474 0.464    1.94  3.50    1.00   6777.
11 mu[1,1]     1.49    1.48  0.289 0.284    1.03  1.97    1.00   2640.
12 mu[2,1]     0.173   0.178  0.339 0.331   -0.384 0.721    1.00   1769.
13 mu[3,1]    -0.456  -0.447  0.305 0.300   -0.976 0.0319    1.00   1986.
14 mu[4,1]     0.346   0.346  0.341 0.333   -0.222 0.900    1.00   1711.
15 mu[5,1]     0.321   0.326  0.507 0.501   -0.511 1.14    1.00   1429.
16 mu[6,1]    -0.0440  -0.0295 0.509 0.493   -0.909 0.765    1.00   1451.
17 mu[7,1]    -0.822  -0.804  0.374 0.366   -1.47 -0.243    1.00   1832.
18 mu[8,1]    -0.395  -0.374  0.548 0.527   -1.33  0.465    1.00   1469.
19 mu[9,1]    -0.746  -0.733  0.360 0.355   -1.36 -0.185    1.00   1812.
20 mu[10,1]   -1.98   -1.95  0.399 0.390   -2.68 -1.37    1.00   2533.
21 mu[1,2]    -0.659  -0.651  0.262 0.261   -1.10 -0.245    1.00   2192.
22 mu[2,2]    -2.21   -2.20  0.400 0.394   -2.90 -1.60    1.00   2370.
23 mu[3,2]    -1.68   -1.66  0.336 0.330   -2.26 -1.15    1.00   2279.
```

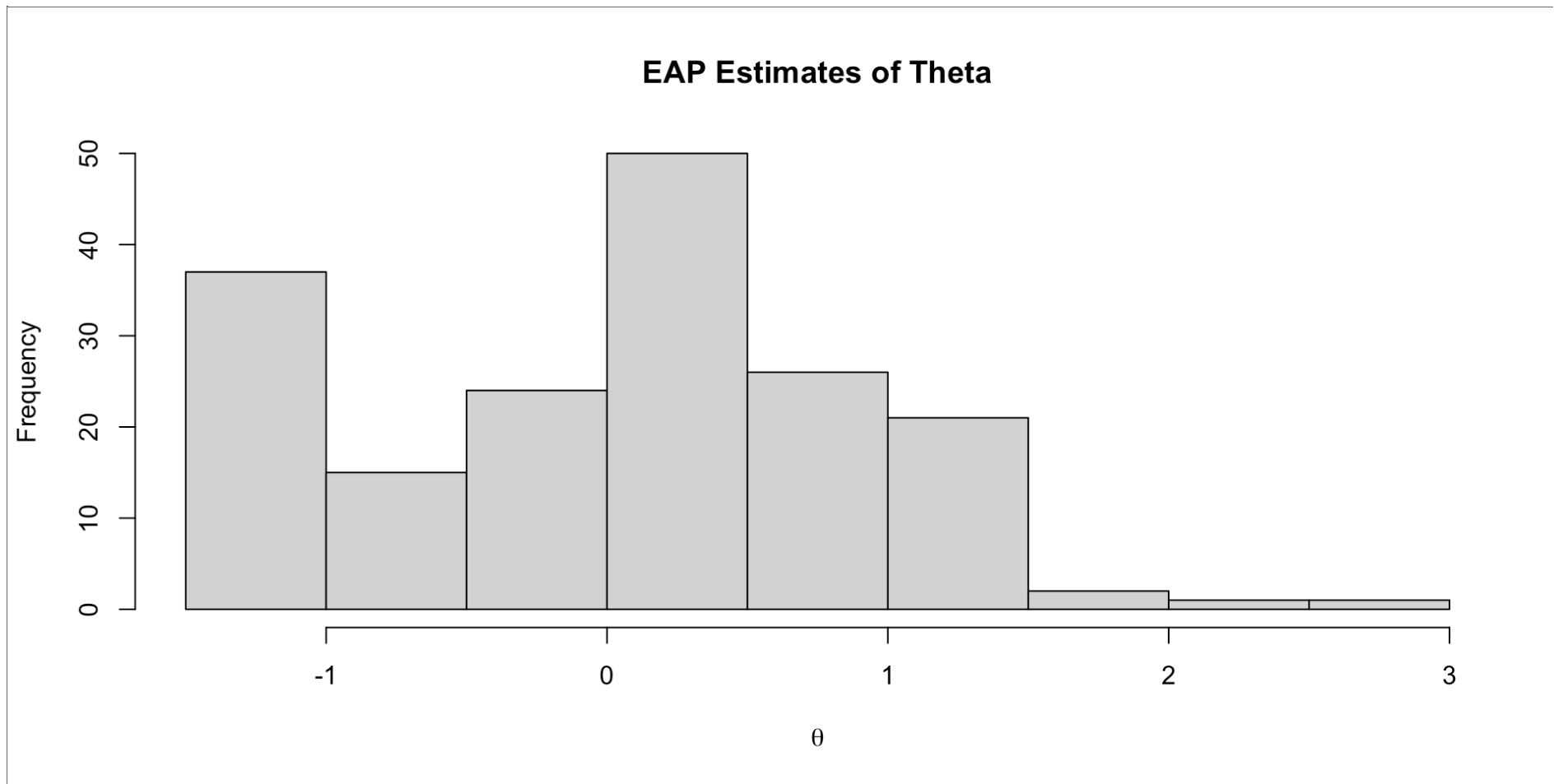
Graded Response Model Option Characteristic Curves



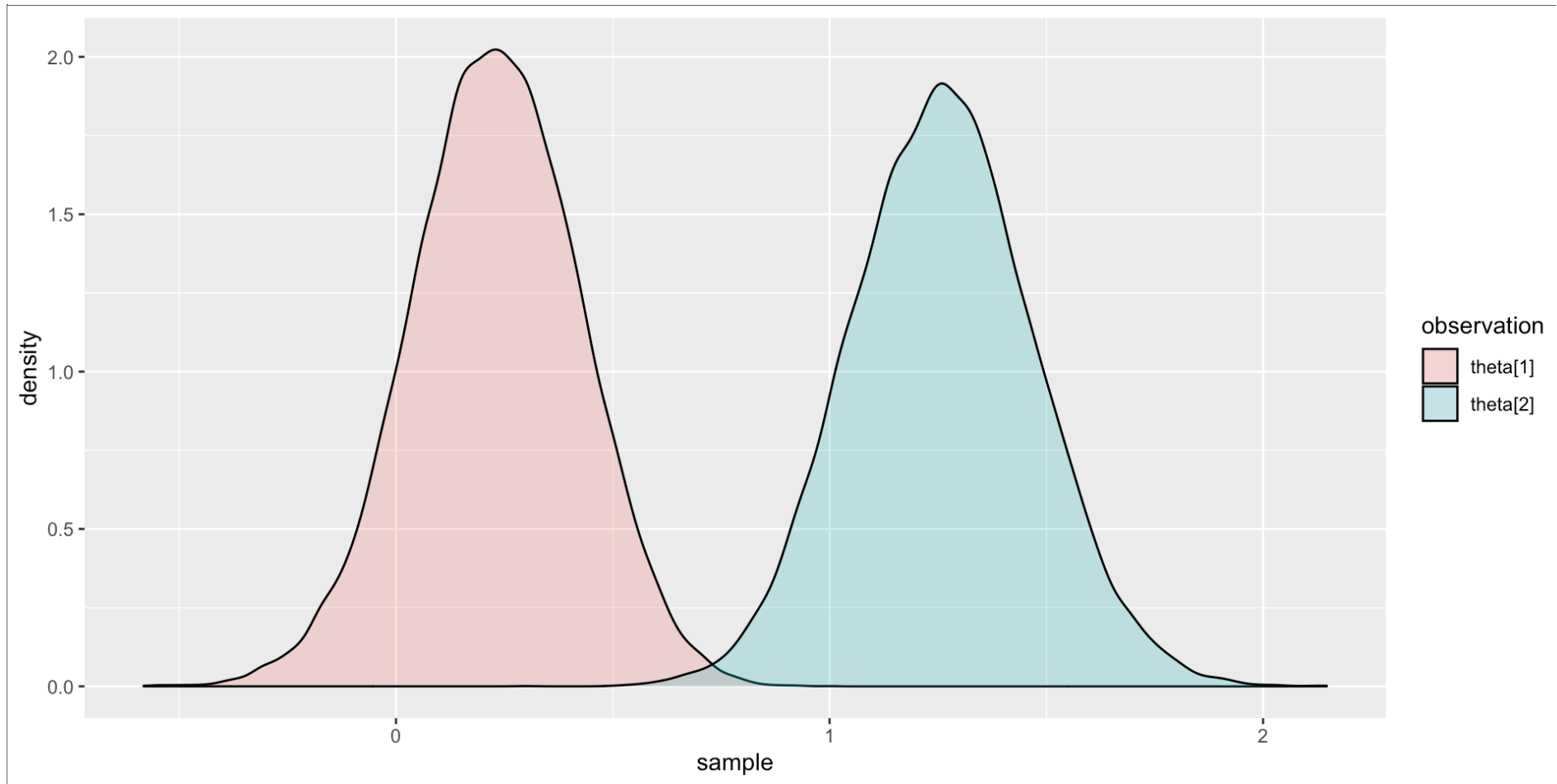
Graded Response Model Item Characteristic Curves



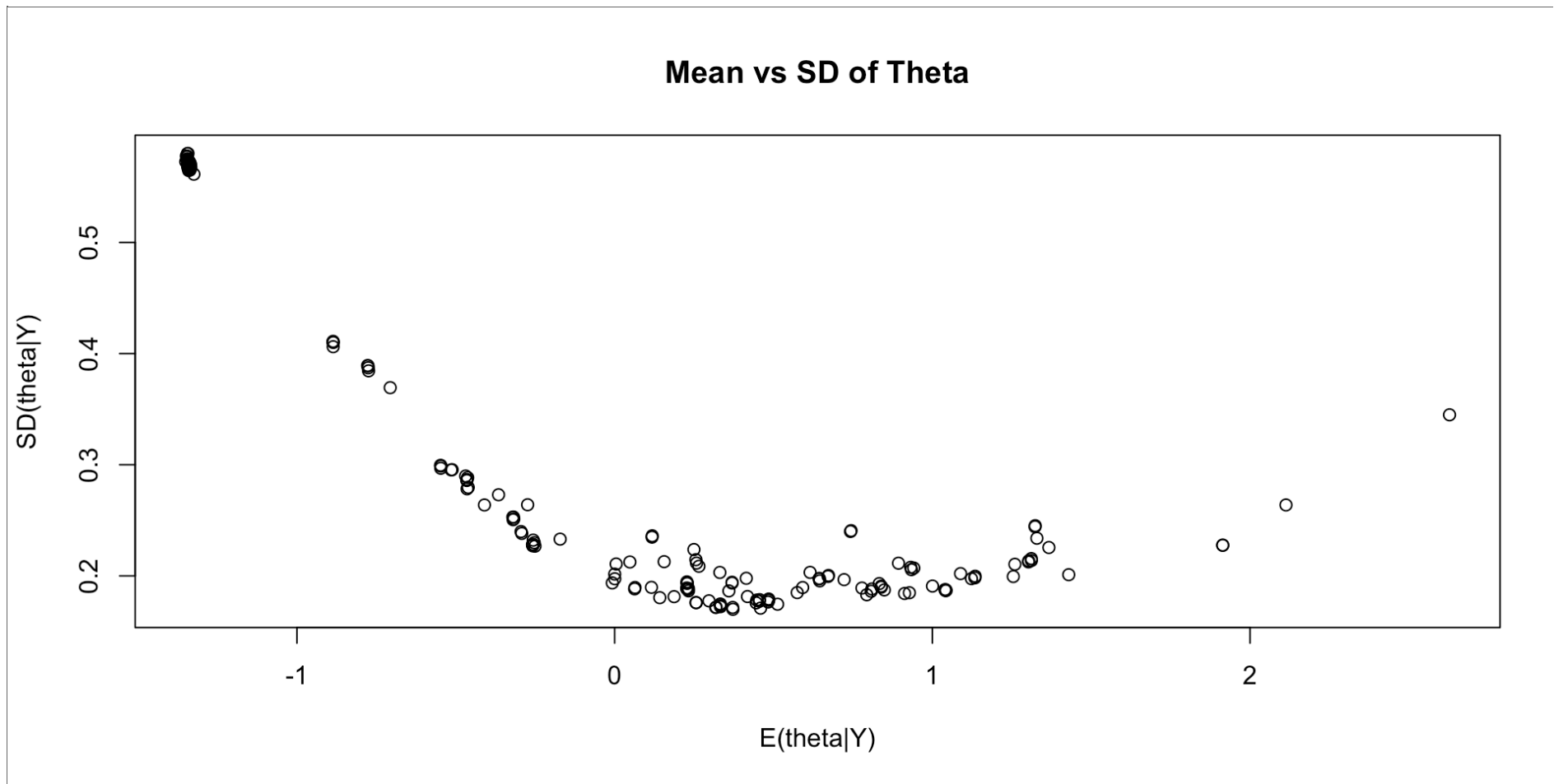
Investigating Latent Variables



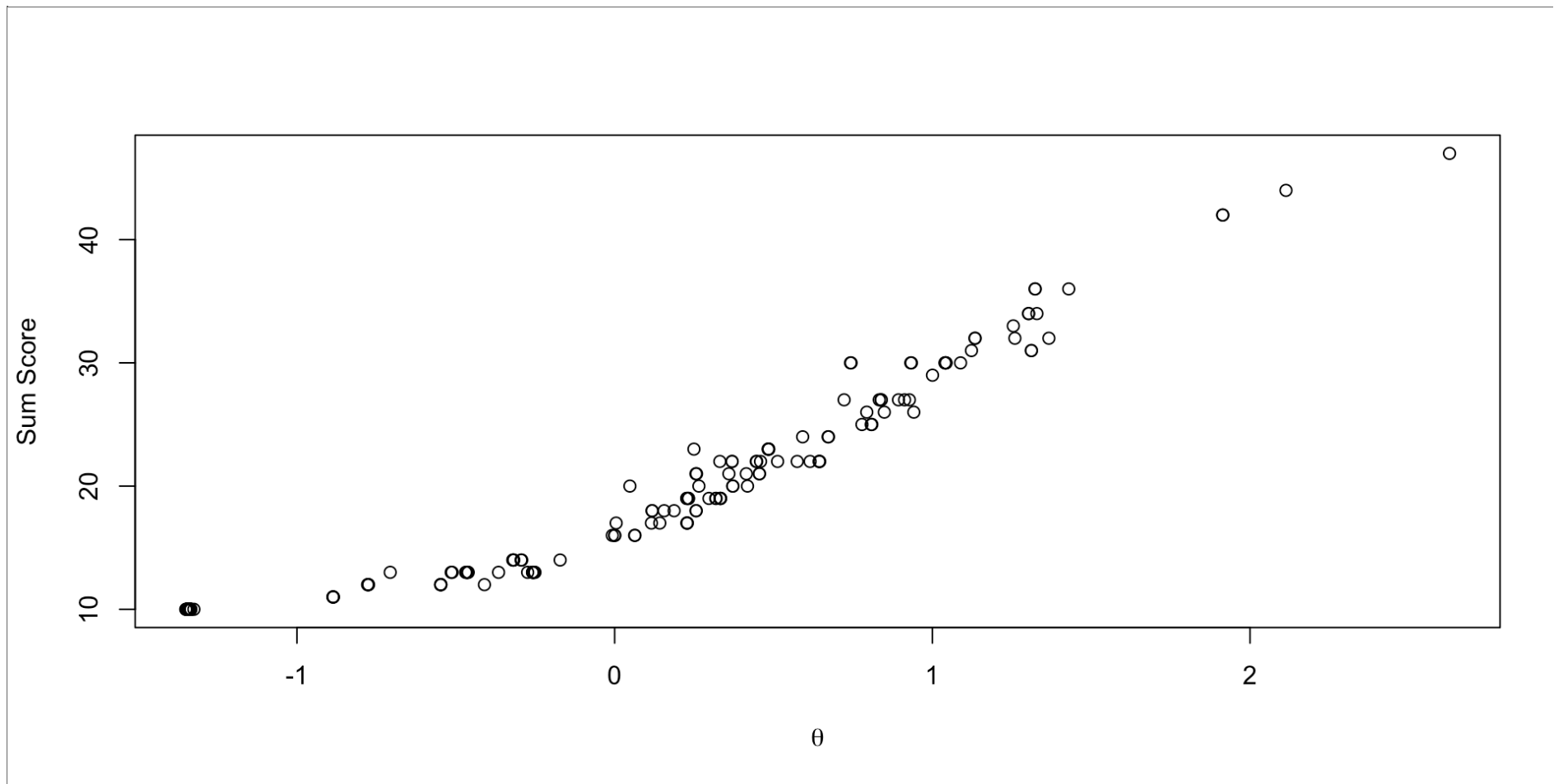
Comparing Two Latent Variable Posterior Distributions



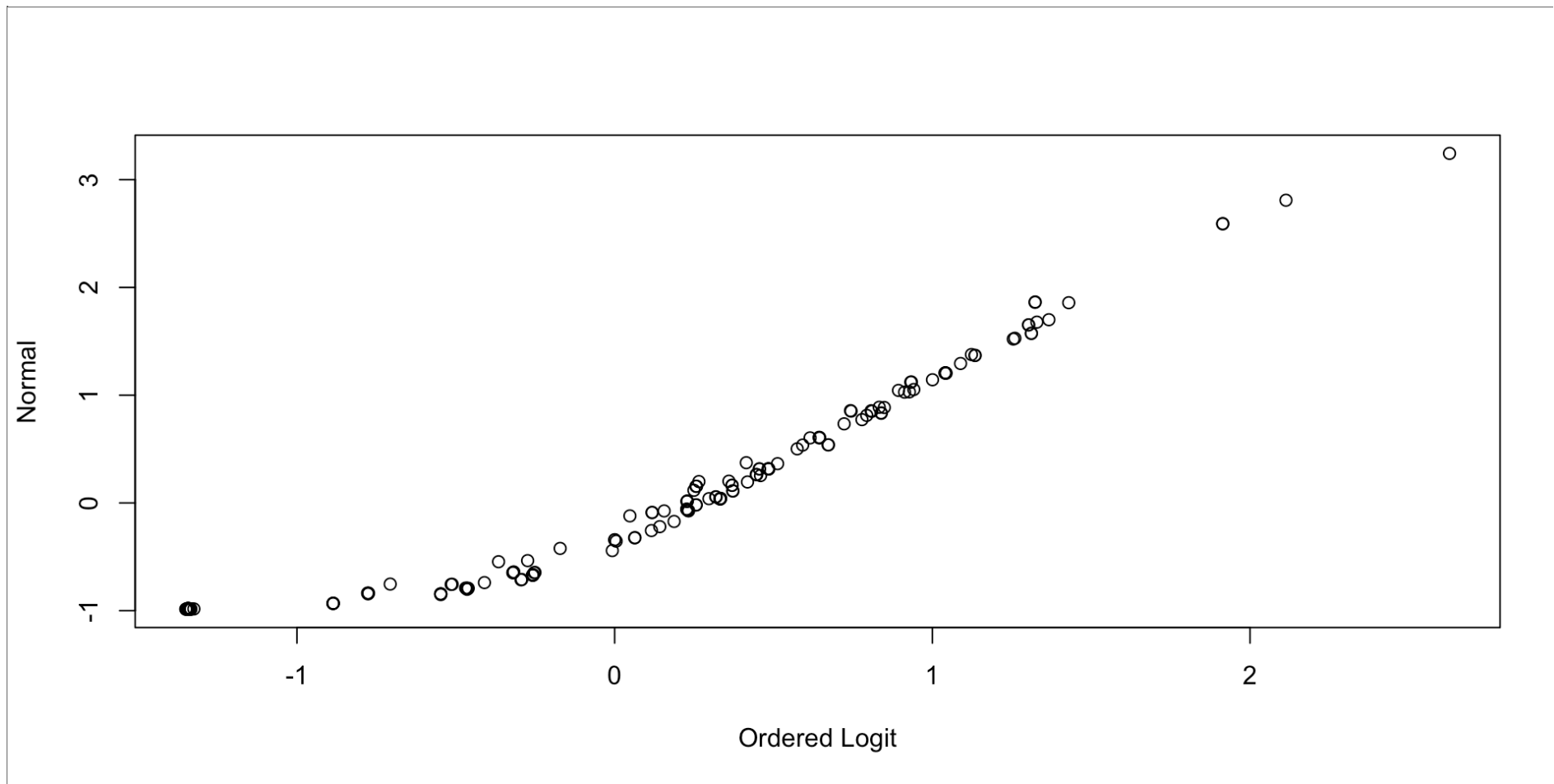
Comparing Latent Variable EAP Estimates with Posterior SDs



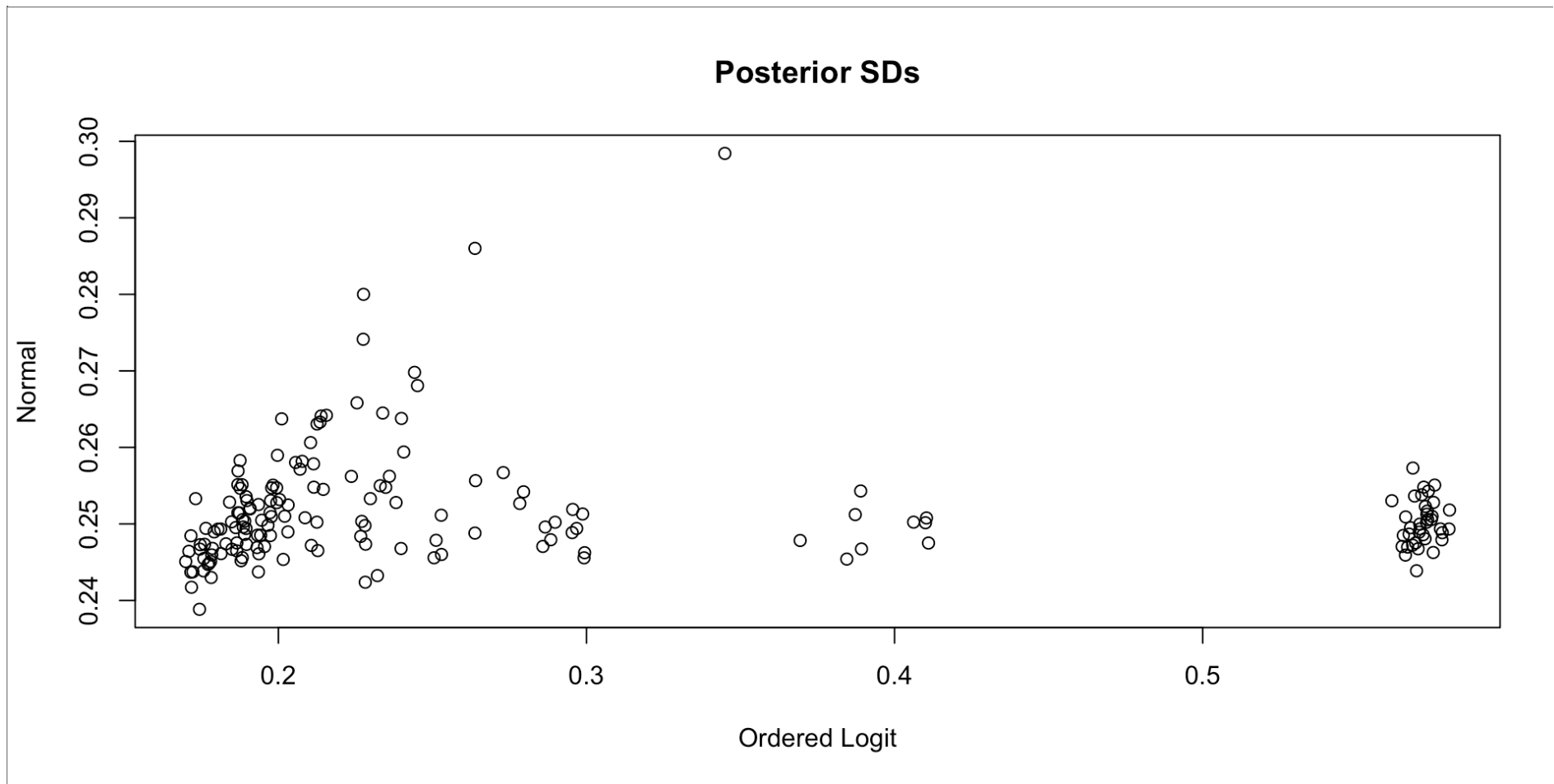
Comparing Latent Variable EAP Estimates with Sum Scores



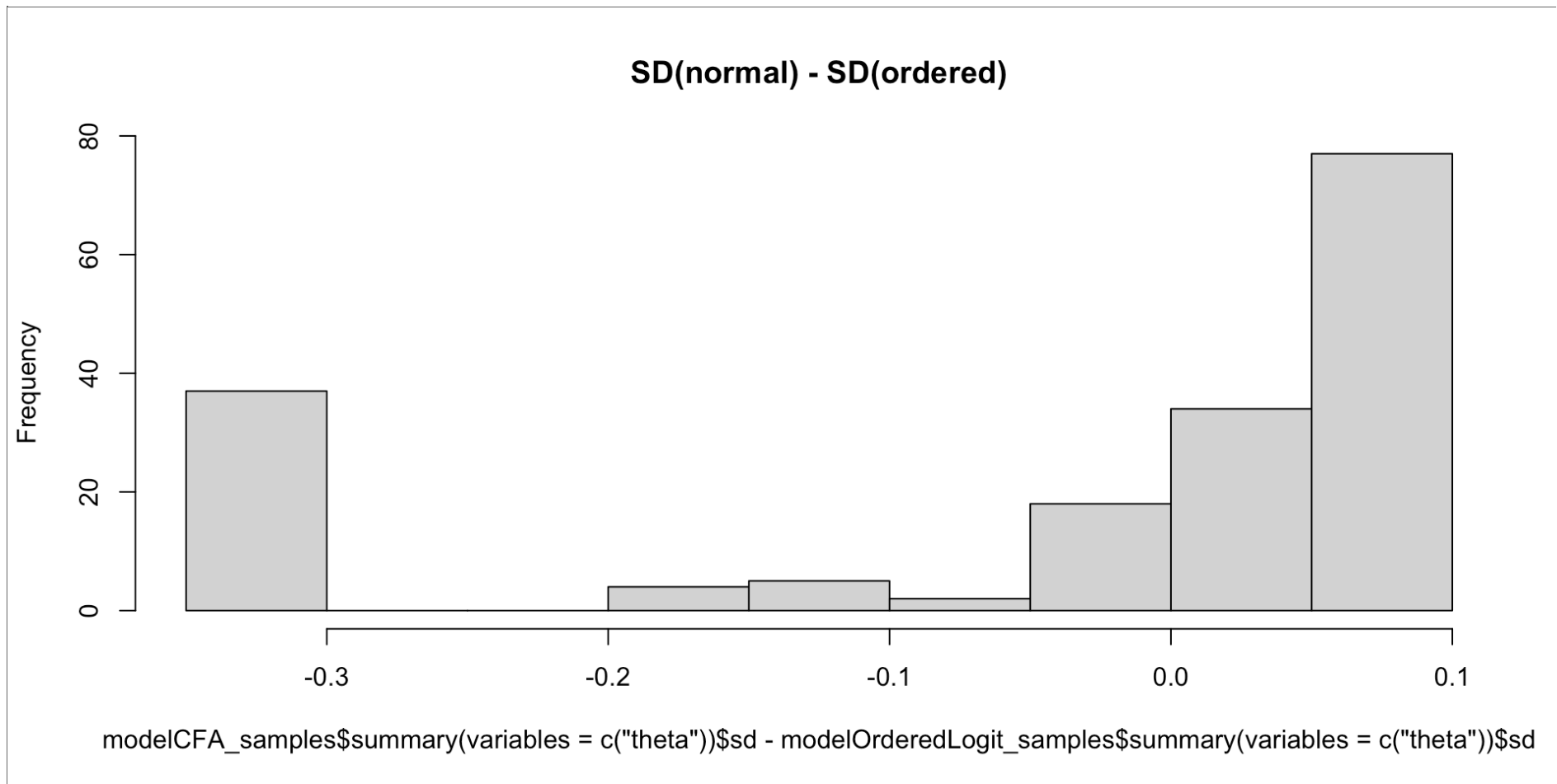
Comparing Latent Variable Posterior Means: GRM vs CFA



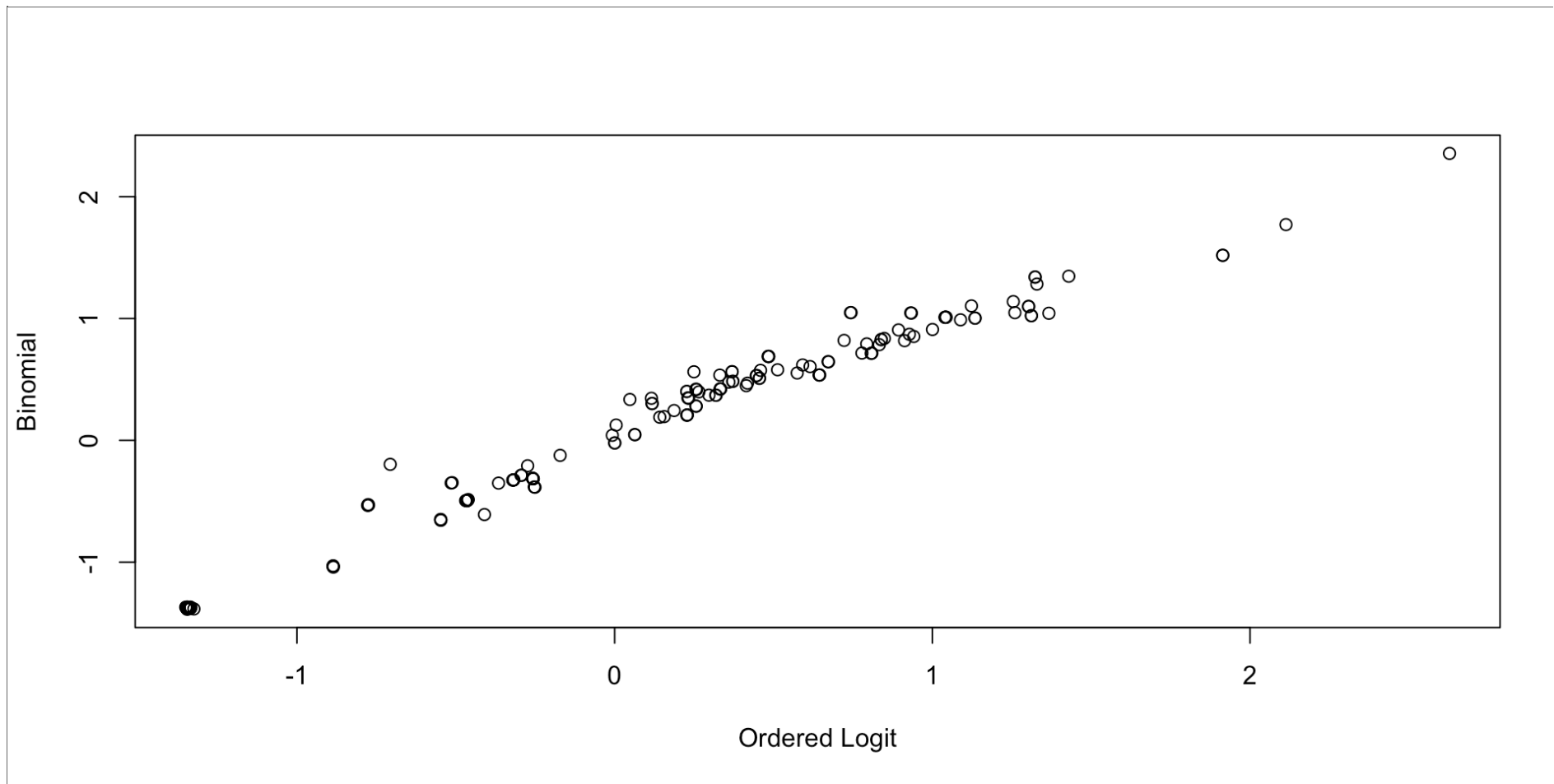
Comparing Latent Variable Posterior SDs: GRM vs Normal



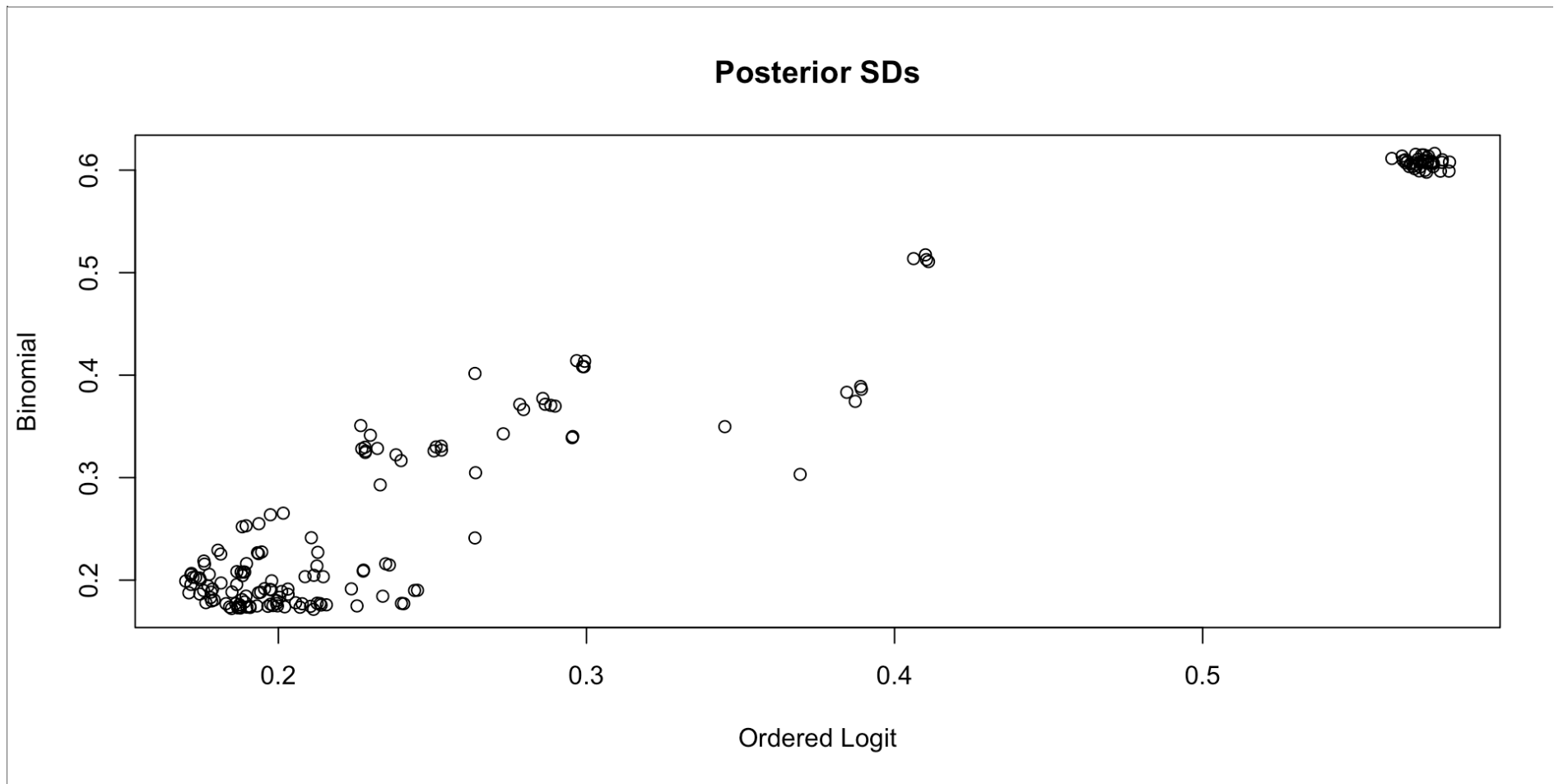
Which Posterior SD is Larger: GRM vs. CFA



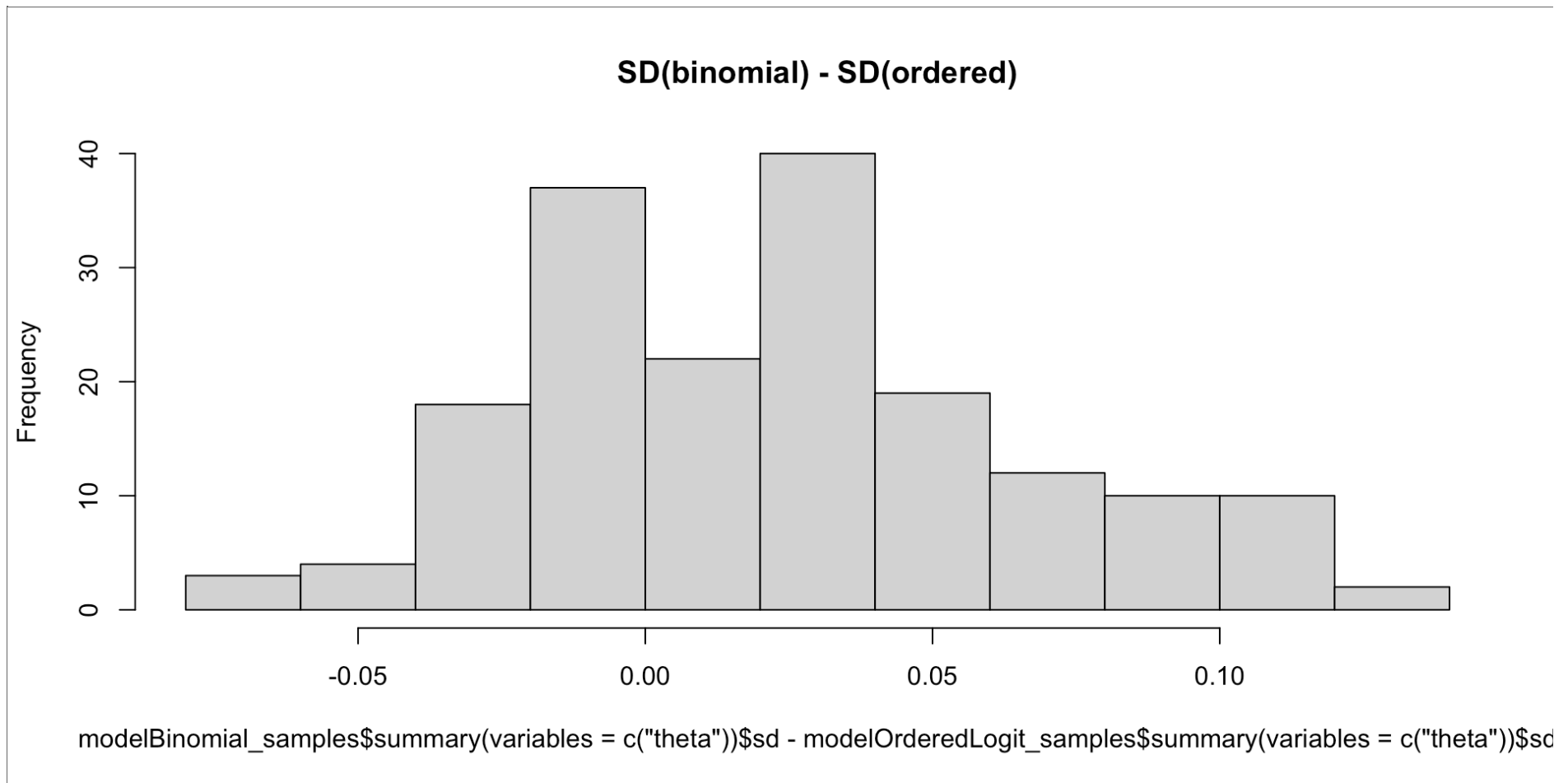
Comparing Thetas: GRM vs Binomial



Comparing Latent Variable Posterior SDs: GRM vs Binomial



Which Posterior SD is Larger: GRM vs. Binomial



Nominal Response Models (Generalized Logit Models)

Adapting the Multinomial Distribution for Item Response Models

With some definitions, we can make a multinomial distribution into one we can use for polytomous item response models

- The number of “trials” is set to one for all items $n_i = 1$ ($\forall i$)
 - With $n_i = 1$, this is called the categorical distribution

$$P(Y_{pi} \mid \theta_p) = p_{i1}^{I(y_{pi}=1)} \cdots p_{iC_i}^{I(y_{pi}=C_i)}$$

- The number of categories is equal to the number of options on an item (C_i)
- The item response model is specified for the set of probabilities p_{ic} , with $\sum_c p_{ic} = 1$
 - Then, use a link function to model each item’s set of p_{ic} as a function of the latent trait

Nominal Response Model (Generalized Logit Model)

The nominal response model is an ordered logistic regression model where:

$$P(Y_{ic} | \theta) = \frac{\exp(\mu_{ic} + \lambda_{ic}\theta_p)}{\sum_{c=1}^{C_i} \exp(\mu_{ic} + \lambda_{ic}\theta_p)}$$

Where:

- One constraint per parameter (one of these options):
 - Sum to zero: $\sum_{c=1}^{C_i} \mu_{ic} = 0$ and $\sum_{c=1}^{C_i} \lambda_{ic} = 0$
 - Fix one category's parameters to zero: $\mu_{iC_i} = 0$ and $\lambda_{iC_i} = 0$

Estimating the NRM in Stan

```

1  model {
2
3    vector[maxCategory] probVec;
4
5    theta ~ normal(0,1);
6
7    for (item in 1:nItems){
8      initMu[item] ~ multi_normal(meanMu[item], covMu[item]); // Prior for item intercepts
9      initLambda[item] ~ multi_normal(meanLambda[item], covLambda[item]); // Prior for item loadings
10
11     for (obs in 1:nObs) {
12       for (category in 1:maxCategory){
13         probVec[category] = mu[item, category] + lambda[item, category]*theta[obs];
14       }
15       Y[item, obs] ~ categorical_logit(probVec);
16     }
17   }
18 }

```

- Probability vector is built category-by-category
 - Code is not vectorized (takes longer to run)
- `categorical_logit` model takes input, applies inverse logit function, evaluates categorical distribution pmf
- Set-to-zero constraints used:
 - `initMu` and `initLambda` have only estimated values in them

- Will build μ and λ in transformed parameters block

Nominal Response Model `parameters` Block

```
1 parameters {  
2   array[nItems] vector[maxCategory - 1] initMu;  
3   array[nItems] vector[maxCategory - 1] initLambda;  
4   vector[nObs] theta;           // the latent variables (one for each person)  
5 }
```

Set-to-zero constraints used:

- `initMu` and `initLambda` have only estimated values in them
- Will build `mu` and `lambda` in transformed parameters block

Nominal Response Model `transformed parameters` Block

```
1 transformed parameters {  
2   array[nItems] vector[maxCategory] mu;  
3   array[nItems] vector[maxCategory] lambda;  
4  
5   for (item in 1:nItems){  
6     mu[item, 2:maxCategory] = initMu[item, 1:(maxCategory-1)];  
7     mu[item, 1] = 0.0; // setting one category's intercept to zero  
8  
9     lambda[item, 2:maxCategory] = initLambda[item, 1:(maxCategory-1)];  
10    lambda[item, 1] = 0.0; // setting one category's lambda to zero  
11  }  
12 }  
13 }
```

Notes:

- Here, we set the first category's μ_{i1} and λ_{i1} to zero
- We use `mu` and `lambda` for the model itself (not `initMu` or `initLambda`)

Nominal Response Model **data** Block

```
1 data {  
2   int maxCategory;  
3   int nObs;  
4   int nItems;  
5  
6   array[nItems, nObs] int Y;  
7  
8   array[nItems] vector[maxCategory-1] meanMu;    // prior mean vector for intercept parameters  
9   array[nItems] matrix[maxCategory-1, maxCategory-1] covMu;    // prior covariance matrix for intercept pa  
10  
11  array[nItems] vector[maxCategory-1] meanLambda;    // prior mean vector for discrimination parameters  
12  array[nItems] matrix[maxCategory-1, maxCategory-1] covLambda; // prior covariance matrix for discriminati  
13  
14 }
```

Almost the same as graded response model

- Lambda now needs an array

Nominal Response Model Data Preparation

```

1 # Data needs: successive integers from 1 to highest number (recode if not consistent)
2 maxCategory = 5
3
4 # data dimensions
5 nObs = nrow(conspiracyItems)
6 nItems = ncol(conspiracyItems)
7
8 # item threshold hyperparameters
9 muMeanHyperParameter = 0
10 muMeanVecHP = rep(muMeanHyperParameter, maxCategory-1)
11 muMeanMatrix = NULL
12 for (item in 1:nItems){
13   muMeanMatrix = rbind(muMeanMatrix, muMeanVecHP)
14 }
15
16 muVarianceHyperParameter = 1000
17 muCovarianceMatrixHP = diag(x = muVarianceHyperParameter, nrow = maxCategory-1)
18 muCovArray = array(data = 0, dim = c(nItems, maxCategory-1, maxCategory-1))
19 for (item in 1:nItems){
20   muCovArray[item, , ] = diag(x = muVarianceHyperParameter, nrow = maxCategory-1)
21 }
22
23 # item discrimination/factor loading hyperparameters
24 lambdaMeanHyperParameter = 0
25 lambdaMeanVecHP = rep(lambdaMeanHyperParameter, maxCategory-1)
26 lambdaMeanMatrix = NULL
27 for (item in 1:nItems){
28   lambdaMeanMatrix = rbind(lambdaMeanMatrix, lambdaMeanVecHP)
29 }
30
31 lambdaVarianceHyperParameter = 1000

```

Nominal Response Model Stan Call

```
1 modelCategoricalLogit_samples = modelCategoricalLogit_stan$sample(  
2   data = modelOrderedLogit_data,  
3   seed = 121120222,  
4   chains = 4,  
5   parallel_chains = 4,  
6   iter_warmup = 1000,  
7   iter_sampling = 1000,  
8   init = function() list(initLambda=rnorm(nItems*(maxCategory-1), mean=1, sd=.1))  
9 )
```

Nominal Response Model Results

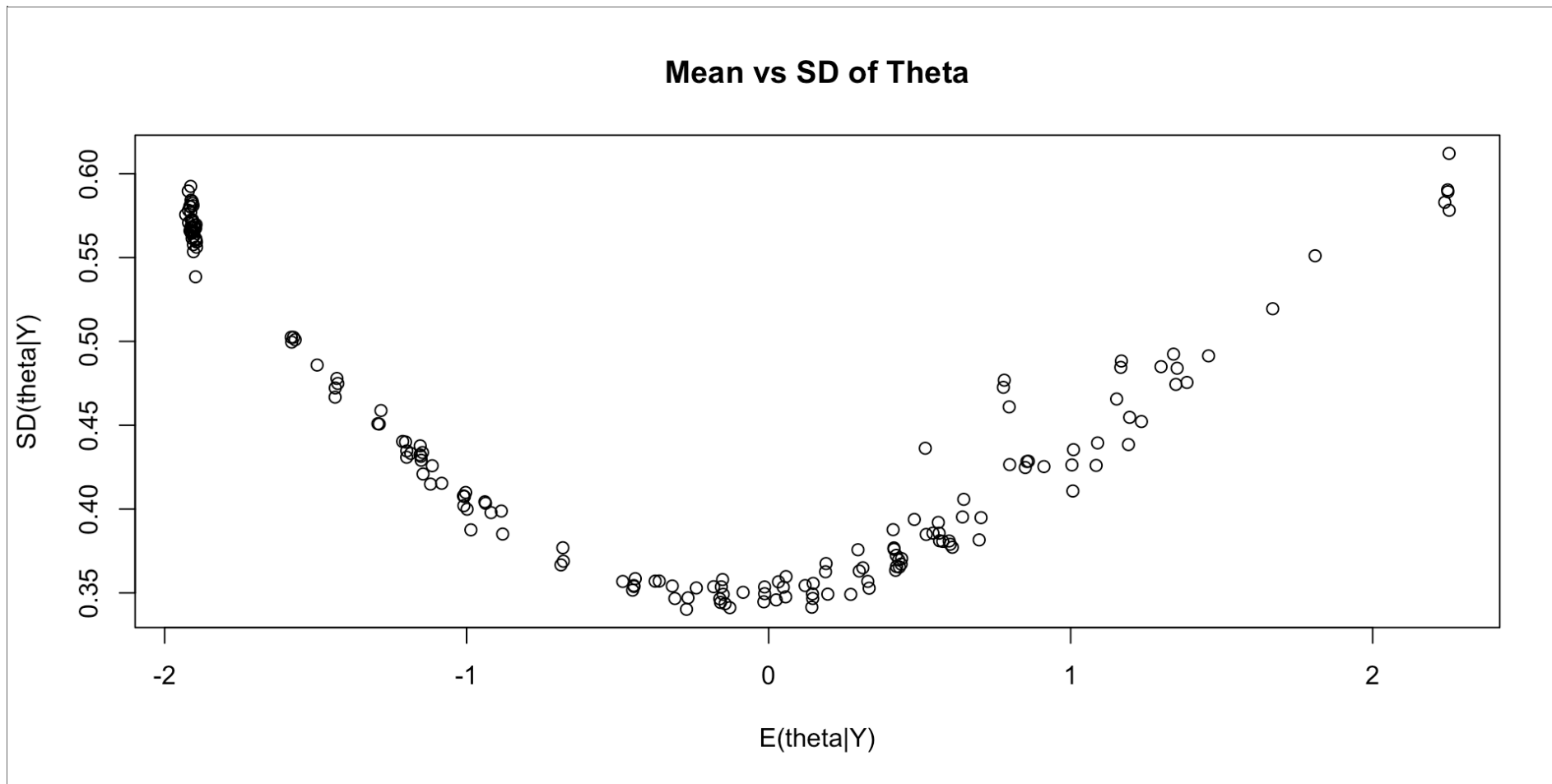
```
1 # checking convergence
2 max(modelCategoricalLogit_samples$summary())$rhat, na.rm = TRUE)
```

```
[1] 1.005476
```

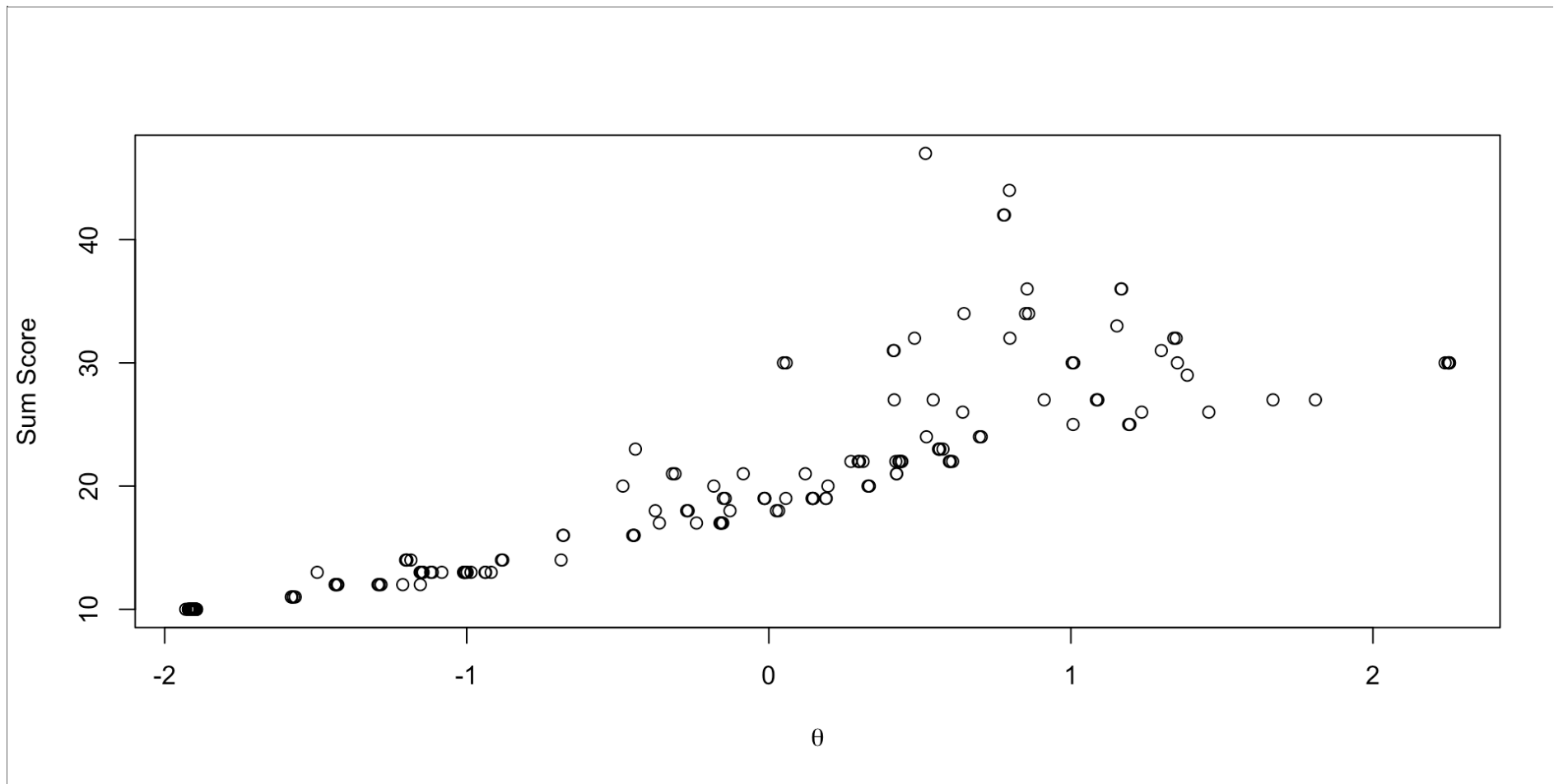
```
1 print(modelCategoricalLogit_samples$summary(variables = c("mu", "lambda")), n=Inf)
```

```
# A tibble: 100 × 10
  variable      mean median    sd    mad    q5    q95    rhat ess_bulk
  <chr>      <dbl>   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>   <dbl>
1 mu[1,1]      0         0      0      0      0      0     NA      NA
2 mu[2,1]      0         0      0      0      0      0     NA      NA
3 mu[3,1]      0         0      0      0      0      0     NA      NA
4 mu[4,1]      0         0      0      0      0      0     NA      NA
5 mu[5,1]      0         0      0      0      0      0     NA      NA
6 mu[6,1]      0         0      0      0      0      0     NA      NA
7 mu[7,1]      0         0      0      0      0      0     NA      NA
8 mu[8,1]      0         0      0      0      0      0     NA      NA
9 mu[9,1]      0         0      0      0      0      0     NA      NA
10 mu[10,1]     0         0      0      0      0      0     NA      NA
11 mu[1,2]     1.06      1.06  0.294 0.287  0.588  1.56    1.00    1611.
12 mu[2,2]     0.369     0.369 0.251 0.254 -0.0311 0.793    1.00    2166.
13 mu[3,2]    -0.494    -0.492 0.265 0.263 -0.931 -0.0611  1.00    2402.
14 mu[4,2]     0.561     0.558 0.278 0.281  0.113  1.02    1.00    1795.
15 mu[5,2]     0.646     0.642 0.264 0.265  0.224  1.09    1.00    2166.
16 mu[6,2]     0.641     0.631 0.285 0.284  0.189  1.12    1.00    1827.
17 mu[7,2]    -0.311    -0.308 0.241 0.238 -0.713  0.0873  1.00    2368.
18 mu[8,2]     0.321     0.319 0.268 0.273 -0.122  0.762    1.00    2019.
19 mu[9,2]    -0.345    -0.347 0.242 0.239 -0.734  0.0552  1.00    2026.
20 mu[10,2]   -1.46     -1.45  0.258 0.260 -1.89  -1.05    1.00    3373.
21 mu[1,3]     0.934     0.927 0.306 0.310  0.433  1.44    1.00    1600.
22 mu[2,3]    -0.740    -0.734 0.375 0.368 -1.37  -0.138  1.00    1764.
23 mu[3,3]    -0.218    -0.216 0.260 0.261 -0.650  0.211    1.00    1979.
```

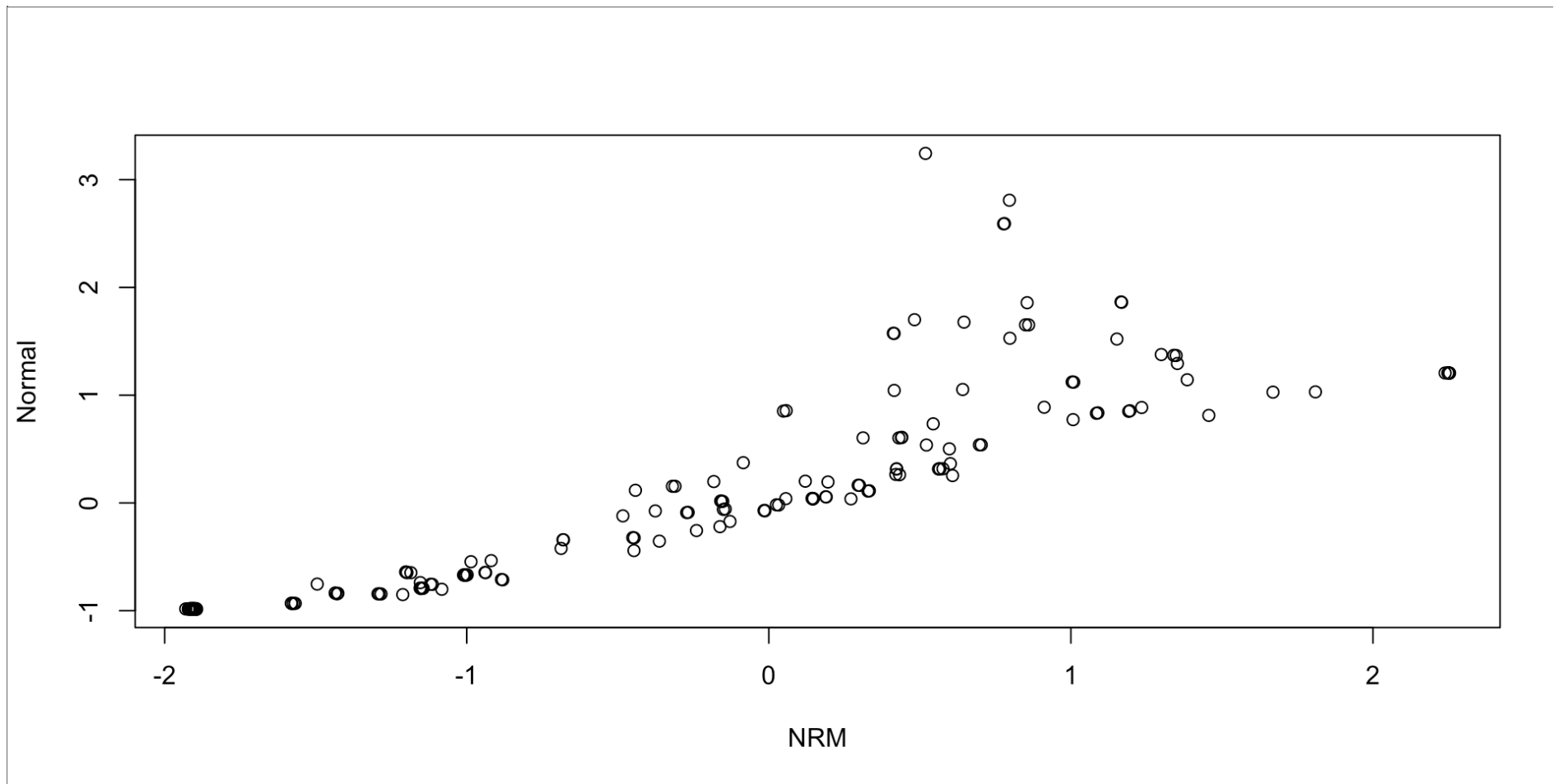
Comparing EAP Estimates with Posterior SDs



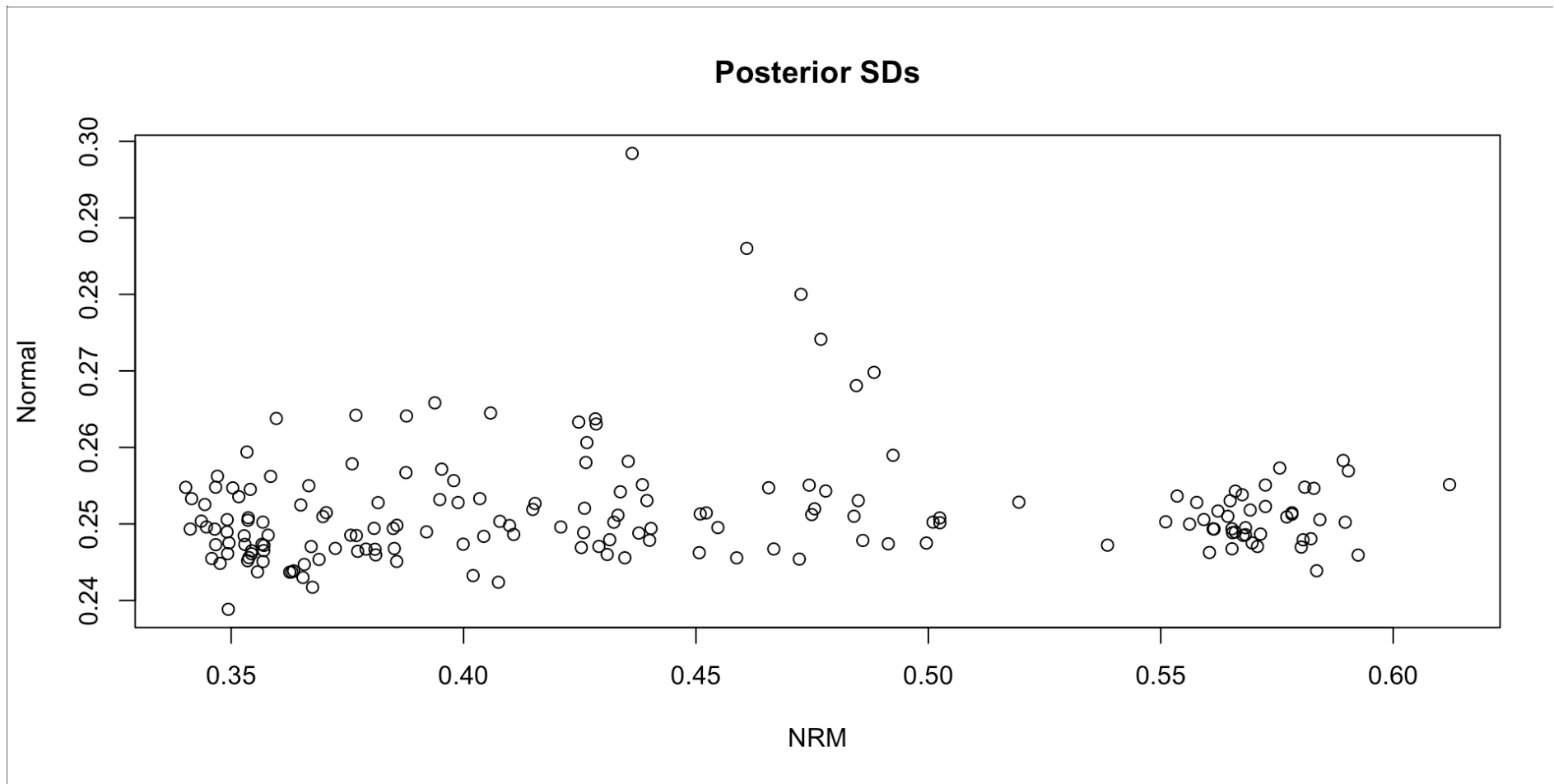
Comparing EAP Estimates with Sum Scores



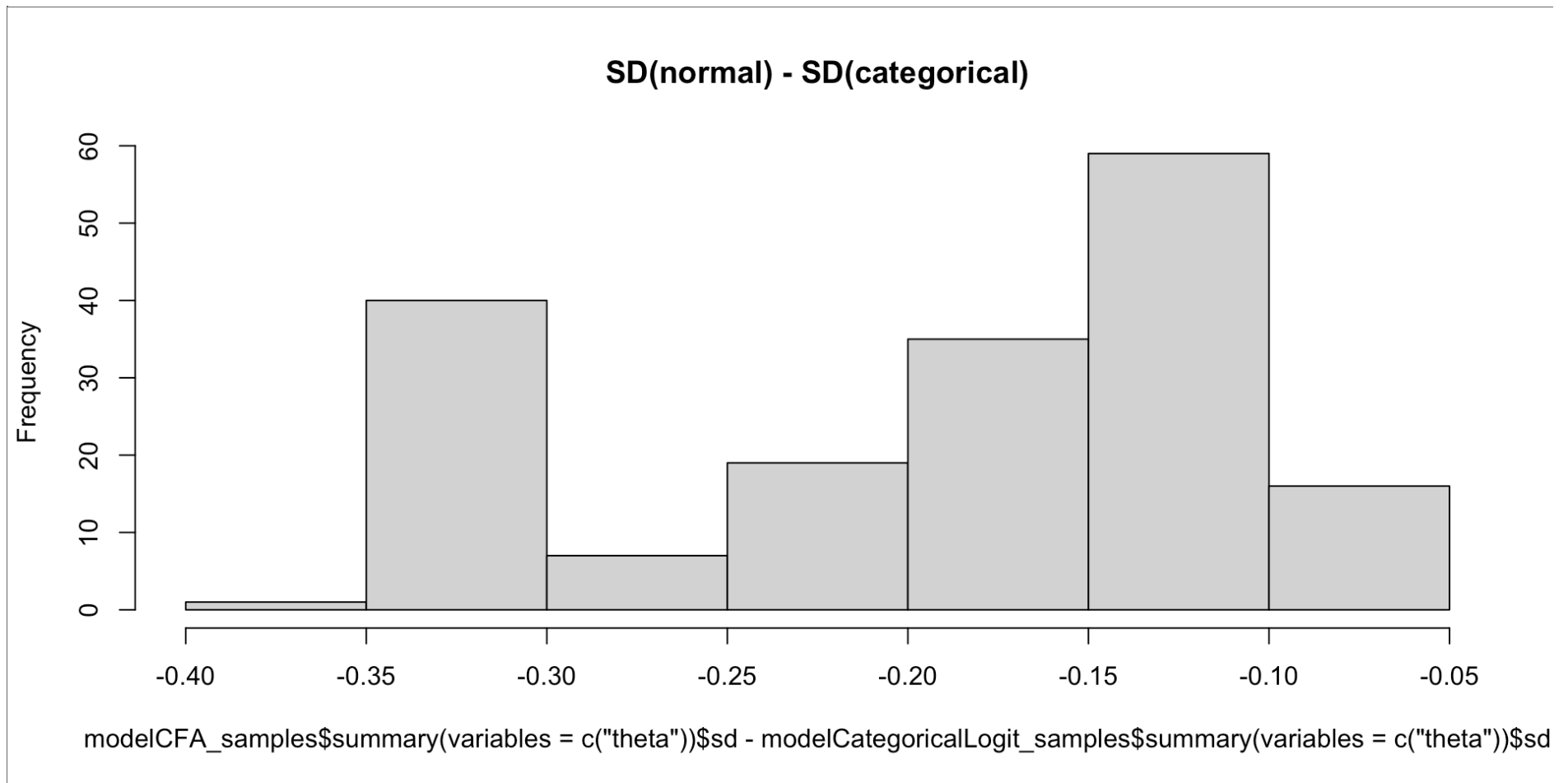
Comparing Thetas: NRM vs Normal:



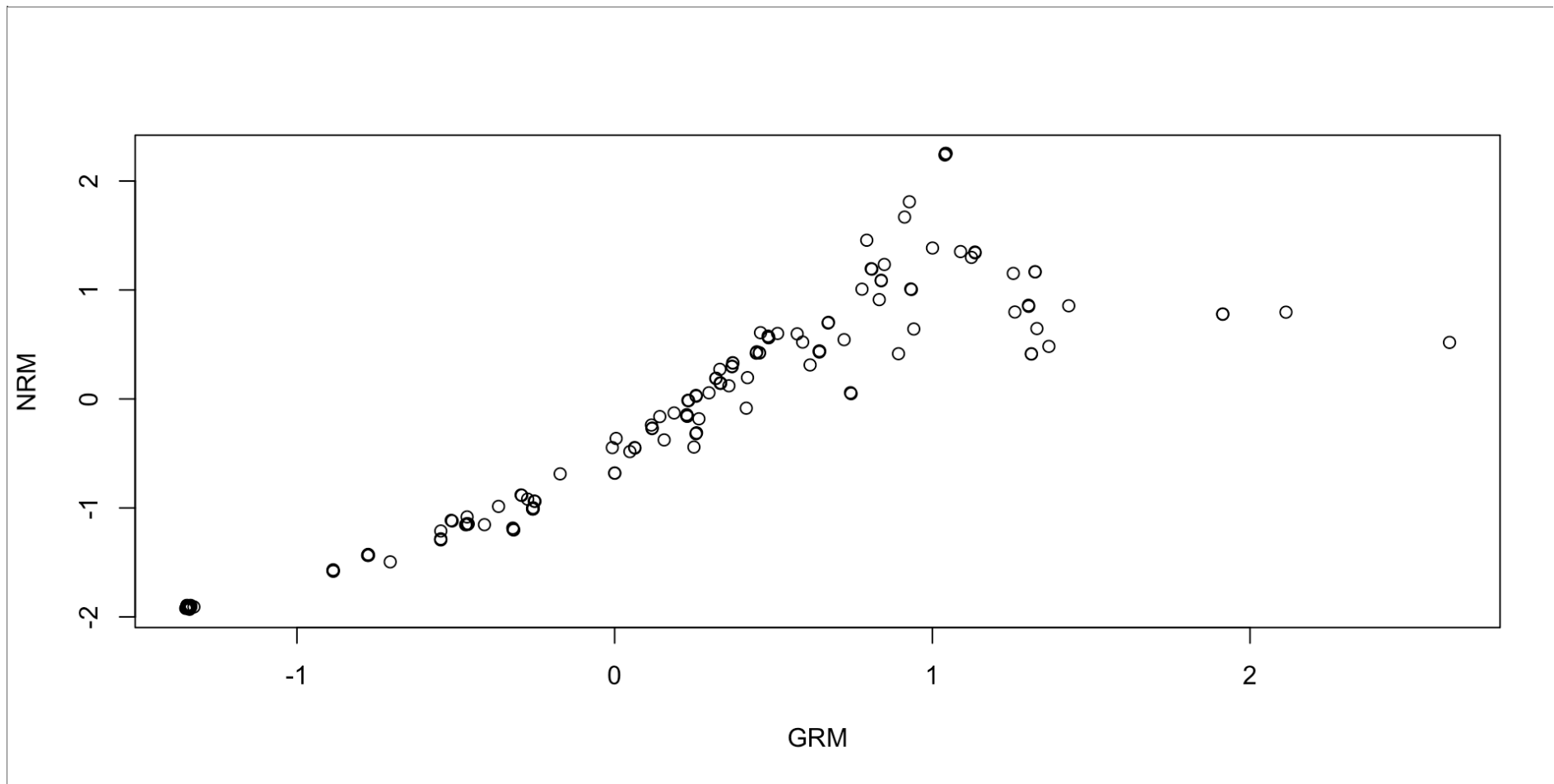
Comparing Theta SDs: NRM vs Normal:



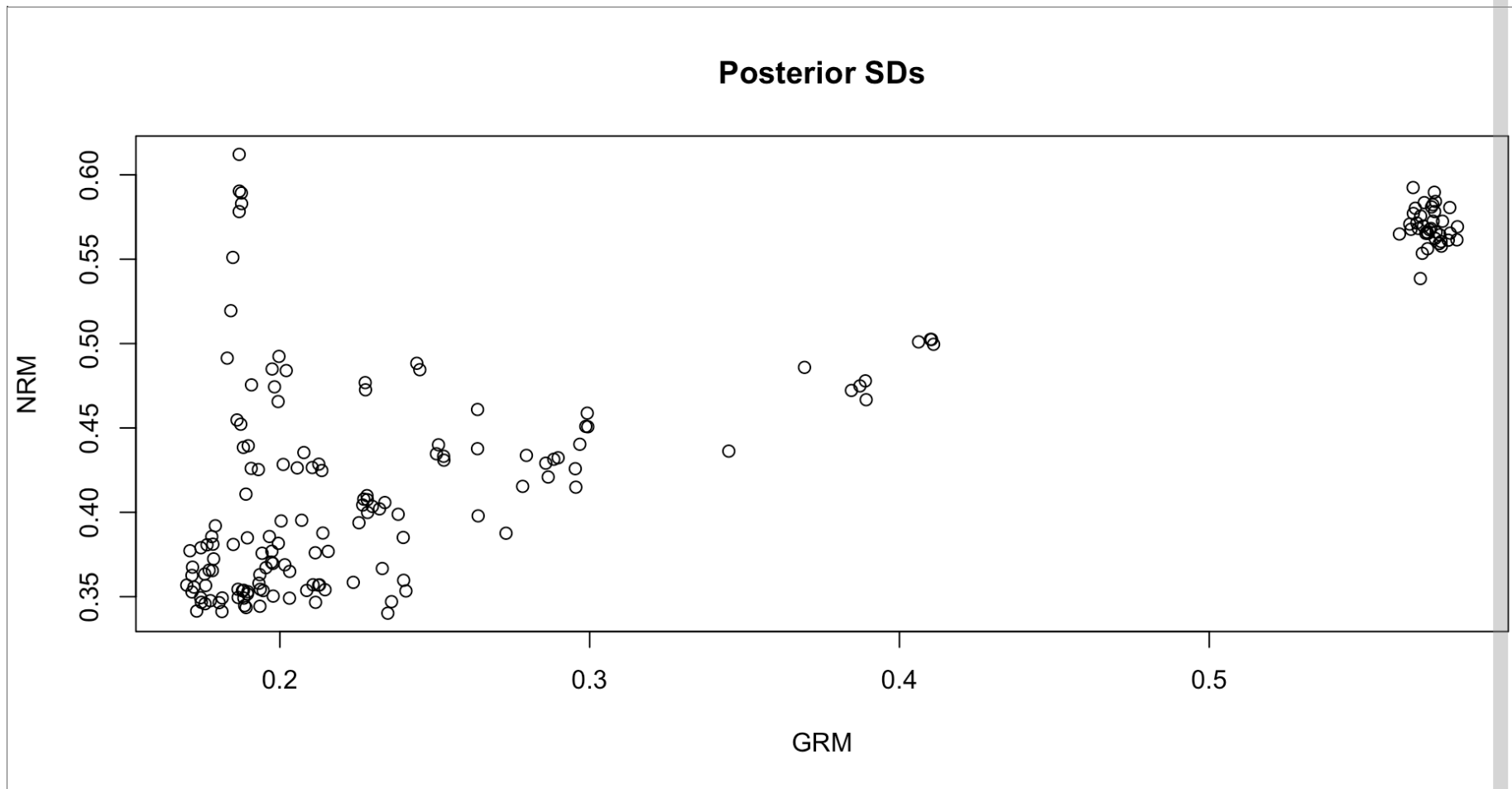
Which SDs are bigger: NRM vs. Normal?



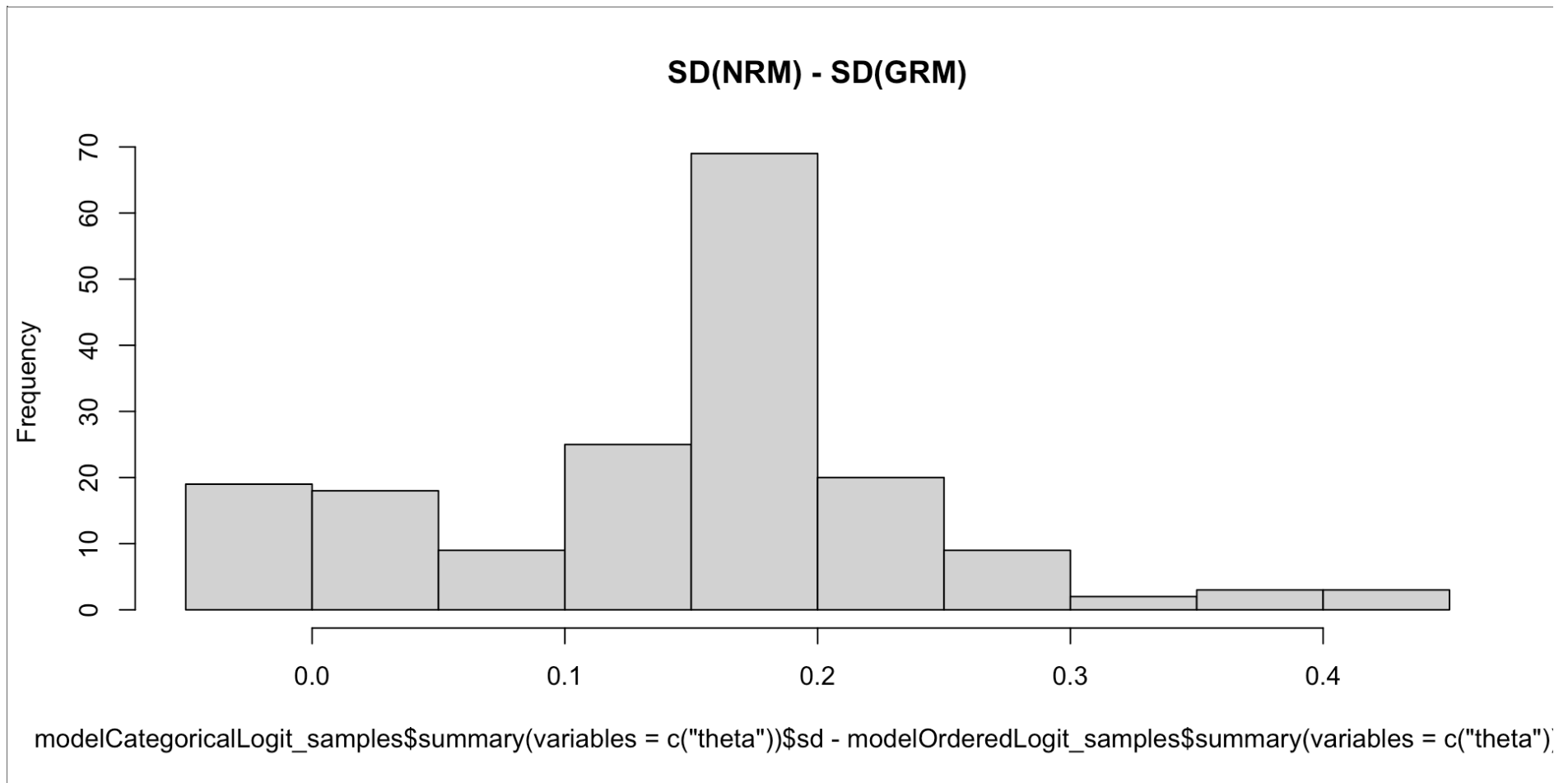
Comparing Thetas: NRM vs GRM:



Comparing Theta SDs: NRM vs GRM:



Which SDs are bigger: NRM vs GRM?



Models With Different Types of Items

Models with Different Types of Items

Although often taught as one type of model that applies to all items, you can mix-and-match distributions

- Recall the posterior distribution of the latent variable θ_p
- For each person, the model (data) likelihood function can be constructed so that each item's conditional PDF is used:

$$f(Y_p | \theta_p) = \prod_{i=1}^I f(Y_{pi} | \theta_p)$$

Here, $\prod_{i=1}^I f(Y_{pi} | \theta_p)$ can be any distribution that you can build

Wrapping Up

Wrapping Up

- There are many different models for polytomous data
 - Almost all use the categorical (multinomial with one trial) distribution
- What we say today is that the posterior SDs of the latent variables are larger with categorical models
 - Much more uncertainty compared to normal models
- What we will need is model fit information to determine what fits best

<https://jonathantemplin.com/bayesian-psychometric-modeling-fall-2022/>