Generalized Measurement Models: Modeling Multidimensional Latent Variables

Lecture 4e

Today's Lecture Objectives

1. Show how to estimate multidimensional latent variable models with polytomous data

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.

Latent Variables

The Latent Variables

Latent variables are built by specification:

- What is their distribution? (nearly all are normally distributed)
- How do you specify their scale: mean and standard deviation? (step two in our model analysis steps)

  You create latent variables by specifying which items measure which latent variables in an analysis model
- Called different names by different fields:
  - Alignment (educational measurement)
  - Factor pattern matrix (factor analysis)
  - Q-matrix (Question matrix; diagnostic models and multidimensional IRT)

From Q-matrix to Model

The alignment provides a specification of which latent variables are measured by which items

- Sometimes we say items “load onto” factors

  The mathematical definition of either of these terms is simply whether or not a latent variable appears as a predictor for an item

- For instance, item one appears to measure nongovernment conspiracies, meaning its alignment (row vector of the Q-matrix) would be:

```
Gov NonGov
  0      1
```

The model for the first item is then built with only the factors measured by the item as being present:

$$f(E\left(Y_{p1} \mid \boldsymbol{\theta}_p\right)) = \mu_1 + \lambda_{11}\theta_{p1}$$

From Q-matrix to Model

The model for the first item is then built with only the factors measured by the item as being present:

$f(E\left(Y_{p1} \mid \boldsymbol{\theta}_p\right)) = \mu_1 + \lambda_{11}\theta_{p1}$

Where:

- μ_1 is the item intercept
- λ_{11} is the factor loading for item 1 (the first subscript) for factor 1 (the second subscript)
- θ_{p1} is the value of the latent variable for person p and factor 1

The second factor is not included in the model for the item.

More on the Q-matrix

We could show the model with the Q-matrix entries:

$$f(E\left(Y_{p1} \mid \boldsymbol{\theta}_p\right)) = \mu_1 + q_{11}\left(\lambda_{11}\theta_{p1}\right) + q_{12}\left(\lambda_{12}\theta_{p2}\right) = \mu_1 + \boldsymbol{\theta}_p \operatorname{diag}(\boldsymbol{q}_i)\boldsymbol{\lambda}_1$$

Where:

- $\boldsymbol{\lambda}_1 = \begin{bmatrix} \lambda_{11} \\ \lambda_{12} \end{bmatrix}$ contains all *possible* factor loadings for item 1 (size 2×1)
- $\boldsymbol{\theta}_p = \begin{bmatrix} \theta_{p1} & \theta_{p2} \end{bmatrix}$ contains the factor scores for person p (size 1×2)
- $\operatorname{diag}(\boldsymbol{q}_i) = \boldsymbol{q}_i \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}$ is a diagonal matrix of ones times the vector of Q-matrix entries for item 1

Even More on the Q-matrix

The matrix product then gives:

$$\boldsymbol{\theta}_p \,\mathrm{diag}\,(\boldsymbol{q}_i)\,\boldsymbol{\lambda}_1 = \begin{bmatrix} \theta_{p1} & \theta_{p2} \end{bmatrix} \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \lambda_{11} \\ \lambda_{12} \end{bmatrix} = \begin{bmatrix} \theta_{p1} & \theta_{p2} \end{bmatrix} \begin{bmatrix} 0 \\ \lambda_{12} \end{bmatrix} = \lambda_{12}\theta_{p2}$$

The Q-matrix functions like a partial version of the model (predictor) matrix that we saw in linear models

Today's Q-matrix

	Gov	NonGov
item1	0	1
item2	1	0
item3	0	1
item4	0	1
item5	1	0
item6	0	1
item7	1	0
item8	1	0
item9	1	0
item10	0	1

Measurement Model Implied by Today's Q-matrix

Given the Q-matrix each item has its own model using the alignment specifications

$$f(E\left(Y_{p1} \mid \boldsymbol{\theta}_p\right)) = \mu_1 + \lambda_{12}\theta_{p2}$$
$$f(E\left(Y_{p2} \mid \boldsymbol{\theta}_p\right)) = \mu_2 + \lambda_{21}\theta_{p1}$$
$$f(E\left(Y_{p3} \mid \boldsymbol{\theta}_p\right)) = \mu_3 + \lambda_{32}\theta_{p2}$$
$$f(E\left(Y_{p4} \mid \boldsymbol{\theta}_p\right)) = \mu_4 + \lambda_{42}\theta_{p2}$$
$$f(E\left(Y_{p5} \mid \boldsymbol{\theta}_p\right)) = \mu_5 + \lambda_{51}\theta_{p1}$$
$$f(E\left(Y_{p6} \mid \boldsymbol{\theta}_p\right)) = \mu_6 + \lambda_{62}\theta_{p2}$$
$$f(E\left(Y_{p7} \mid \boldsymbol{\theta}_p\right)) = \mu_7 + \lambda_{71}\theta_{p1}$$
$$f(E\left(Y_{p8} \mid \boldsymbol{\theta}_p\right)) = \mu_8 + \lambda_{81}\theta_{p1}$$
$$f(E\left(Y_{p9} \mid \boldsymbol{\theta}_p\right)) = \mu_9 + \lambda_{91}\theta_{p1}$$
$$f(E\left(Y_{p,10} \mid \boldsymbol{\theta}_p\right)) = \mu_{10} + \lambda_{10,2}\theta_{p2}$$

Our Example: Multivariate Normal Distribution

For our example, we will assume the set of traits follows a multivariate normal distribution

$$f\left(\boldsymbol{\theta}_p\right) = (2\pi)^{-\frac{D}{2}} \det\left(\boldsymbol{\Sigma}_\theta\right)^{-\frac{1}{2}} \exp\left[-\frac{1}{2}\left(\boldsymbol{\theta}_p - \boldsymbol{\mu}_\theta\right)^T \boldsymbol{\Sigma}_\theta^{-1} \left(\boldsymbol{\theta}_p - \boldsymbol{\mu}_\theta\right)\right]$$

Where:

- $\pi \approx 3.14$
- D is the number of latent variables (dimensions)
- $\boldsymbol{\Sigma}_\theta$ is the covariance matrix of the latent variables
  - $\boldsymbol{\Sigma}_\theta^{-1}$ is the inverse of the covariance matrix
- $\boldsymbol{\mu}_\theta$ is the mean vector of the latent variables
- $\det(\cdot)$ is the matrix determinant function
- $(\cdot)^T$ is the matrix transpose operator

Alternatively, we would specify $\boldsymbol{\theta}_p \sim N_D\left(\boldsymbol{\mu}_\theta, \boldsymbol{\Sigma}_\theta\right)$; but, we cannot always estimate $\boldsymbol{\mu}_\theta$ and $\boldsymbol{\Sigma}_\theta$

Identification of Latent Traits, Part 1

Psychometric models require two types of identification to be valid:

1. Empirical Identification

- The minimum number of items that must measure each latent variable
- From CFA: three observed variables for each latent variable (or two if the latent variable is correlated with another latent variable)

  Bayesian priors can help to make models with fewer items than these criteria suggest estimable
- The parameter estimates (item parameters and latent variable estimates) often have MCMC convergence issues and should not be trusted
- Use the CFA standard in your work

Identification of Latent Traits, Part 2

Psychometric models require two types of identification to be valid:

2. Scale Identification (i.e., what the mean/variance is for each latent variable)

- The additional set of constraints needed to set the mean and standard deviation (variance) of the latent variables
- Two main methods to set the scale:
  - Marker item parameters
    - For variances: Set the loading/slope to one for one observed variable per latent variable
      - Can estimate the latent variable's variance (the diagonal of $\boldsymbol{\Sigma}_\theta$)
    - For means: Set the item intercept to one for one observed variable perlatent variable
      - Can estimate the latent variable's mean (in $\boldsymbol{\mu}_\theta$)
  - Standardized factors
    - Set the variance for all latent variables to one

- Set the mean for all latent variables to zero
- Estimate all unique off-diagonal correlations (covariances) in $\boldsymbol{\Sigma}_\theta$

More on Scale Identification

Bayesian priors can let you believe you can estimate more parameters than the non-Bayesian standards suggest

- For instance, all item parameters and the latent variable means/variances

Like empirical identification, these estimates are often unstable and are not recommended
Most common:

- Standardized latent variables
  - Used for scale development and/or when scores are of interest directly
- Marker item for latent variables and zero means
  - Used for cases where latent variables may become outcomes (and that variance needs explained)

Today, we will use standardized factors

- Covariance matrix diagonal (factor variances) all ones (factor correlation matrix)

Observed Item Model

For each item, we will use a graded- (ordinal logit) response modelwhere:

$$P\left(Y_{ic} = c \mid \boldsymbol{\theta}_p\right) = \begin{cases} 1 - P^*\left(Y_{i1} > 1 \mid \boldsymbol{\theta}_p\right) & \text{if } c = 1 \\ P^*\left(Y_{ic-1} > c - 1 \mid \boldsymbol{\theta}_p\right) - P^*\left(Y_{ic} > c \mid \boldsymbol{\theta}_p\right) & \text{if } 1 < c < C_i \\ P^*\left(Y_{iC_i-1} > C_i - 1 \mid \boldsymbol{\theta}_p\right) - 0 & \text{if } c = C_i \end{cases}$$

Where:

$$P^*\left(Y_{ic} > c \mid \theta\right) = \frac{\exp(\mu_{ic} + \boldsymbol{\theta}_p \operatorname{diag}\left(\boldsymbol{q}_i\right) \boldsymbol{\lambda}_i)}{1 + \exp(\mu_{ic} + \boldsymbol{\theta}_p \operatorname{diag}\left(\boldsymbol{q}_i\right) \boldsymbol{\lambda}_i)}$$

With:

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

Building the Multidimensional Model in Stan

Observed Item Model in Stan

As Stan uses thresholds instead of intercepts, our model then becomes:

$$P\left(Y_{ic} = c \mid \boldsymbol{\theta}_p\right) = \begin{cases} 1 - P\left(Y_{i1} > 1 \mid \boldsymbol{\theta}_p\right) & \text{if } c = 1 \\ P\left(Y_{ic-1} > c - 1 \mid \boldsymbol{\theta}_p\right) - P\left(Y_{ic} > c \mid \boldsymbol{\theta}_p\right) & \text{if } 1 < c < C_i \\ P\left(Y_{iC_i-1} > C_i - 1 \mid \boldsymbol{\theta}_p\right) & \text{if } c = C_i \end{cases}$$

Where:

$$P\left(Y_{ic} > c \mid \theta\right) = \frac{\exp(-\tau_{ic} + \boldsymbol{\theta}_p \operatorname{diag}\left(\boldsymbol{q}_i\right) \boldsymbol{\lambda}_i)}{1 + \exp(-\tau_{ic} + \boldsymbol{\theta}_p \operatorname{diag}\left(\boldsymbol{q}_i\right) \boldsymbol{\lambda}_i)}$$

With:

- $C_i - 1$ Ordered thresholds: $\tau_1 < \tau_2 < \ldots < \tau_{C_i-1}$

We can convert thresholds to intercepts by multiplying by negative one: $\mu_c = -\tau_c$

Stan Model Block

```
model {

  lambda ~ multi_normal(meanLambda, covLambda);
  thetaCorrL ~ lkj_corr_cholesky(1.0);
  theta ~ multi_normal_cholesky(meanTheta, thetaCorrL);


  for (item in 1:nItems){
    thr[item] ~ multi_normal(meanThr[item], covThr[item]);
    Y[item] ~ ordered_logistic(thetaMatrix*lambdaMatrix[item,1:nFactors]', thr[item]);
  }


}
```

- `thetaMatrix` is matrix of latent variables for each person Θ (size $N \times D$; N is number of people, D is number of dimensions)
- `lambdaMatrix` is, for each item, a matrix of factor loading/discrimination parameters along with zeros by Q-matrix (size $I \times D$; I is number of items)
  - The transpose of one row of the lambda matrix is used, resulting in a scalar for each person
- `lambda` is a vector of estimated factor loadings (we use a different block to put these into `lambdaMatrix`)

- Each factor loading has a (univariate) normal distribution for a prior

More Model Block Notes:

```
model {

  lambda ~ multi_normal(meanLambda, covLambda);
  thetaCorrL ~ lkj_corr_cholesky(1.0);
  theta ~ multi_normal_cholesky(meanTheta, thetaCorrL);


  for (item in 1:nItems){
    thr[item] ~ multi_normal(meanThr[item], covThr[item]);
    Y[item] ~ ordered_logistic(thetaMatrix*lambdaMatrix[item,1:nFactors]', thr[item]);
  }


}
```

- `theta` is an array of latent variables (we use a different block to put these into `thetaMatrix`)
- `theta` follows a multivariate normal distribution as a prior where:
  - We set all means of `theta` to zero (the mean for each factor is set to zero)
  - We set all variances of `theta` to one (the variance for each factor is set to one)
  - We estimate a correlation matrix of factors $\mathbf{R}_\theta$ using a so-called LKJ prior distribution

    - LKJ priors are found in Stan and make modeling correlations very easy

- Of note: we are using the Cholesky forms of the functions `lkj_corr_cholesky` and `multi_normal_cholesky`

LKJ Priors for Correlation Matrices

From **Stan's Functions Reference**, for a correlation matrix $\mathbf{R}_\theta$
Correlation Matrix Properties:

- Positive definite (determinant greater than zero; $\det(R_\theta) > 0$
- Symmetric
- Diagonal values are all one

LKJ Prior, with hyperparameter η, is proportional to the determinant of the correlation matrix
$\text{LKJ}(\mathbf{R}_\theta \mid \eta) \propto \det(\mathbf{R}_\theta)^{(\eta-1)}$

- Density is uniform over correlation matrices with $\eta = 1$
- With $\eta > 1$, identity matrix is modal (moves correlations toward zero)
- With $0 < \eta < 1$, density has a trough at the identity (moves correlations away from zero)

For this example, we set $\eta = 1$, noting a uniform prior over all correlation matrices

Cholesky Decomposition

The functions we are using do not use the correlation matrix directly

THE PROBLEM:

- The issue: We need the inverse and determinant to calculate the density of a multivariate normal distribution
  - Inverses and determinants are numerically unstable (computationally) in general form

THE SOLUTION:

- Convert R_θ to a triangular form using the Cholesky decomposition

$$\mathbf{R}_\theta = \mathbf{L}_\theta \mathbf{L}_\theta^T$$

Cholesky Decomposition Math

- $\det(\mathbf{R}_\theta) = \det\left(\mathbf{L}_\theta \mathbf{L}_\theta^T\right) = \det(\mathbf{L}_\theta)\det\left(\mathbf{L}_\theta^T\right) = \det(\mathbf{L}_\theta)^2 = \left(\prod_{d=1}^{D} l_{d,d}\right)^2$ (note: product becomes sum for log-likelihood)
- For the inverse, we work with the term in the exponent of the MVN pdf:

  $$-\frac{1}{2}\left(\boldsymbol{\theta}_p - \boldsymbol{\mu}_\theta\right)^T \mathbf{R}_\theta^{-1}\left(\boldsymbol{\theta}_p - \boldsymbol{\mu}_\theta\right) = -\frac{1}{2}\left(\boldsymbol{\theta}_p - \boldsymbol{\mu}_\theta\right)^T \left(\mathbf{L}_\theta \mathbf{L}_\theta^T\right)^{-1}\left(\boldsymbol{\theta}_p - \boldsymbol{\mu}_\theta\right)$$

  $$-\frac{1}{2}\left(\boldsymbol{\theta}_p - \boldsymbol{\mu}_\theta\right)^T \mathbf{L}_\theta^{-T}\mathbf{L}_\theta^{-1}\left(\boldsymbol{\theta}_p - \boldsymbol{\mu}_\theta\right)$$

  Then, we solve by back substitution: $\left(\boldsymbol{\theta}_p - \boldsymbol{\mu}_\theta\right)^T \mathbf{L}_\theta^{-T}$
- We can then multiply this result by itself to form the term in the exponent (times $-\frac{1}{2}$)
- Back substitution involves far fewer steps, minimizing the amount of rounding error in the process of forming the inverse
- Most algorithms using MVN distributions use some variant of this process (perhaps with a different factorization method such as QR)

Stan Parameters Block

```
parameters {
  array[nObs] vector[nFactors] theta;               // the latent variables (one for each person)
  array[nItems] ordered[maxCategory-1] thr; // the item thresholds (one for each item category minus one)
  vector[nLoadings] lambda;             // the factor loadings/item discriminations (one for each item)
  cholesky_factor_corr[nFactors] thetaCorrL;
}
```

Notes:

- `theta` is an array (for the MVN prior)
- `thr` is the same as the unidimensional model
- `lambda` is the vector of all factor loadings to be estimated (needs `nLoadings`)
- `thetaCorrL` is of type `cholesky_factor_corr`, a built in type that identifies this as the lower diagonal of a Cholesky-factorized correlation matrix

We need a couple other blocks to link our parameters to the model

Stan Transformed Data Block

```
transformed data{
  int<lower=0> nLoadings = 0;                                    // number of loadings in model

  for (factor in 1:nFactors){
    nLoadings = nLoadings + sum(Qmatrix[1:nItems, factor]);
  }

  array[nLoadings, 2] int loadingLocation;                    // the row/column positions of each loading
  int loadingNum=1;

  for (item in 1:nItems){
    for (factor in 1:nFactors){
      if (Qmatrix[item, factor] == 1){
        loadingLocation[loadingNum, 1] = item;
        loadingLocation[loadingNum, 2] = factor;
        loadingNum = loadingNum + 1;
      }
    }
  }

}
```

- The `transformed data {}` block runs prior to the Markov chain
  - We use it to create variables that will stay constant throughout the chain
- This syntax works for any Q-matrix (but only has main effects in the model)

Stan Transformed Data Block

```
transformed data{
  int<lower=0> nLoadings = 0;                                    // number of loadings in model

  for (factor in 1:nFactors){
    nLoadings = nLoadings + sum(Qmatrix[1:nItems, factor]);
  }

  array[nLoadings, 2] int loadingLocation;                    // the row/column positions of each loading
  int loadingNum=1;

  for (item in 1:nItems){
    for (factor in 1:nFactors){
      if (Qmatrix[item, factor] == 1){
        loadingLocation[loadingNum, 1] = item;
        loadingLocation[loadingNum, 2] = factor;
        loadingNum = loadingNum + 1;
      }
    }
  }

}
```

- Here, we count the number of loadings in the Q-matrix `nLoadings`
  - We then process the Q-matrix to tell Stan the row and column position of each loading in the `loadingsMatrix` used in the `model {}` block
  - For instance, the loading for item one factor two has row position 1 and column position 2

Stan Transformed Parameters Block

```
transformed parameters{
  matrix[nItems, nFactors] lambdaMatrix = rep_matrix(0.0, nItems, nFactors);
  matrix[nObs, nFactors] thetaMatrix;

  // build matrix for lambdas to multiply theta matrix
  for (loading in 1:nLoadings){
    lambdaMatrix[loadingLocation[loading,1], loadingLocation[loading,2]] = lambda[loading];
  }

  for (factor in 1:nFactors){
    thetaMatrix[,factor] = to_vector(theta[,factor]);
  }

}
```

Notes:

- The `transformed parameters {}` block runs prior to each iteration of the Markov chain
  - This means it affects the estimation of each type of parameter
- We use it to create:
  - `thetaMatrix` (converting `theta` from an array to a matrix)
  - `lambdaMatrix` (puts the loadings and zeros from the Q-matrix into correct position)

- `lambdaMatrix` initializes at zero (so we just have to add the loadings in the correct position)

Stan Data Block

```
data {

  // data specifications  ============================================================
  int<lower=0> nObs;                          // number of observations
  int<lower=0> nItems;                        // number of items
  int<lower=0> maxCategory;       // number of categories for each item

  // input data  ============================================================
  array[nItems, nObs] int<lower=1, upper=5>  Y; // item responses in an array

  // loading specifications  ============================================================
  int<lower=1> nFactors;                                    // number of loadings in the model
  array[nItems, nFactors] int<lower=0, upper=1> Qmatrix;

  // prior specifications ============================================================
  array[nItems] vector[maxCategory-1] meanThr;               // prior mean vector for intercept parameters
  array[nItems] matrix[maxCategory-1, maxCategory-1] covThr;  // prior covariance matrix for intercept parame

  vector[nItems] meanLambda;         // prior mean vector for discrimination parameters
  matrix[nItems, nItems] covLambda;  // prior covariance matrix for discrimination parameters

  vector[nFactors] meanTheta;
}
```

- Changes from unidimensional model:
  - `meanTheta`: Factor means (hyperparameters) are added (but we will set these to zero)
  - `nFactors`: Number of latent variables (needed for Q-matrix)

- `Qmatrix`: Q-matrix for model

Stan Generated Quantities Block

```
generated quantities{
  array[nItems] vector[maxCategory-1] mu;
  corr_matrix[nFactors] thetaCorr;

  for (item in 1:nItems){
    mu[item] = -1*thr[item];
  }


  thetaCorr = multiply_lower_tri_self_transpose(thetaCorrL);

}
```

Notes:

- Converts thresholds to intercepts `mu`
- Creates `thetaCorr` by multiplying Cholesky-factorized lower triangle with upper triangle
  - We will only use the parameters of `thetaCorr` when looking at model output

Stan Analyses

Estimation in Stan

We run stan the same way we have previously:

```
modelOrderedLogit_samples = modelOrderedLogit_stan$sample(
  data = modelOrderedLogit_data,
  seed = 191120221,
  chains = 4,
  parallel_chains = 4,
  iter_warmup = 2000,
  iter_sampling = 2000,
  init = function() list(lambda=rnorm(nItems, mean=5, sd=1))
)
```

Notes:

- Smaller chain (model takes a lot longer to run)
- Still need to keep loadings positive initially

Analysis Results

```
[1] 1.064337
```

```
# A tibble: 54 × 10
   variable          mean  median      sd     mad       q5    q95  rhat ess_bulk
   <chr>            <dbl>   <dbl>   <dbl>   <dbl>    <dbl>  <dbl> <dbl>    <dbl>
 1 lambda[1]       2.01    1.99   0.267   0.263     1.59    2.46   1.00   3625.
 2 lambda[2]       2.84    2.81   0.383   0.373     2.25    3.50   1.00   3014.
 3 lambda[3]       2.39    2.37   0.344   0.341     1.86    2.99   1.00   3552.
 4 lambda[4]       2.92    2.90   0.389   0.391     2.31    3.59   1.00   2767.
 5 lambda[5]       4.33    4.29   0.600   0.587     3.43    5.40   1.00   2960.
 6 lambda[6]       4.21    4.18   0.574   0.575     3.34    5.20   1.00   2485.
 7 lambda[7]       2.85    2.83   0.414   0.413     2.24    3.59   1.00   3085.
 8 lambda[8]       4.12    4.09   0.565   0.550     3.25    5.10   1.00   2523.
 9 lambda[9]       2.90    2.88   0.421   0.423     2.26    3.63   1.00   3097.
10 lambda[10]      2.44    2.41   0.428   0.423     1.79    3.18   1.00   4365.
11 mu[1,1]         1.88    1.87   0.293   0.298     1.41    2.37   1.00   2177.
12 mu[2,1]         0.810   0.806  0.302   0.306     0.324   1.32   1.00   1201.
13 mu[3,1]         0.0939  0.0909 0.265   0.257    -0.337   0.539  1.00   1394.
14 mu[4,1]         0.993   0.983  0.320   0.318     0.488   1.53   1.00   1290.
15 mu[5,1]         1.31    1.30   0.426   0.411     0.638   2.04   1.00   1142.
16 mu[6,1]         0.967   0.960  0.408   0.407     0.297   1.66   1.00   1039.
17 mu[7,1]        -0.106  -0.105  0.290   0.285    -0.585   0.370  1.00   1282.
18 mu[8,1]         0.693   0.683  0.399   0.393     0.0476  1.35   1.00   1039.
19 mu[9,1]        -0.0641 -0.0608 0.296   0.292    -0.547   0.419  1.00   1217.
20 mu[10,1]       -1.36   -1.35   0.313   0.306    -1.90   -0.867  1.00   1901.
21 mu[1,2]        -0.219  -0.219  0.233   0.233    -0.600   0.166  1.00   1552.
22 mu[2,2]        -1.53   -1.51   0.319   0.320    -2.07   -1.03   1.00   1429.
23 mu[3,2]        -1.11   -1.10   0.278   0.265    -1.57   -0.656  1.00   1621.
```

Note:

- EAP estimate of correlation was 0.993 – indicates only one factor in data

Correlation Chains

Correlation Posterior Distribution

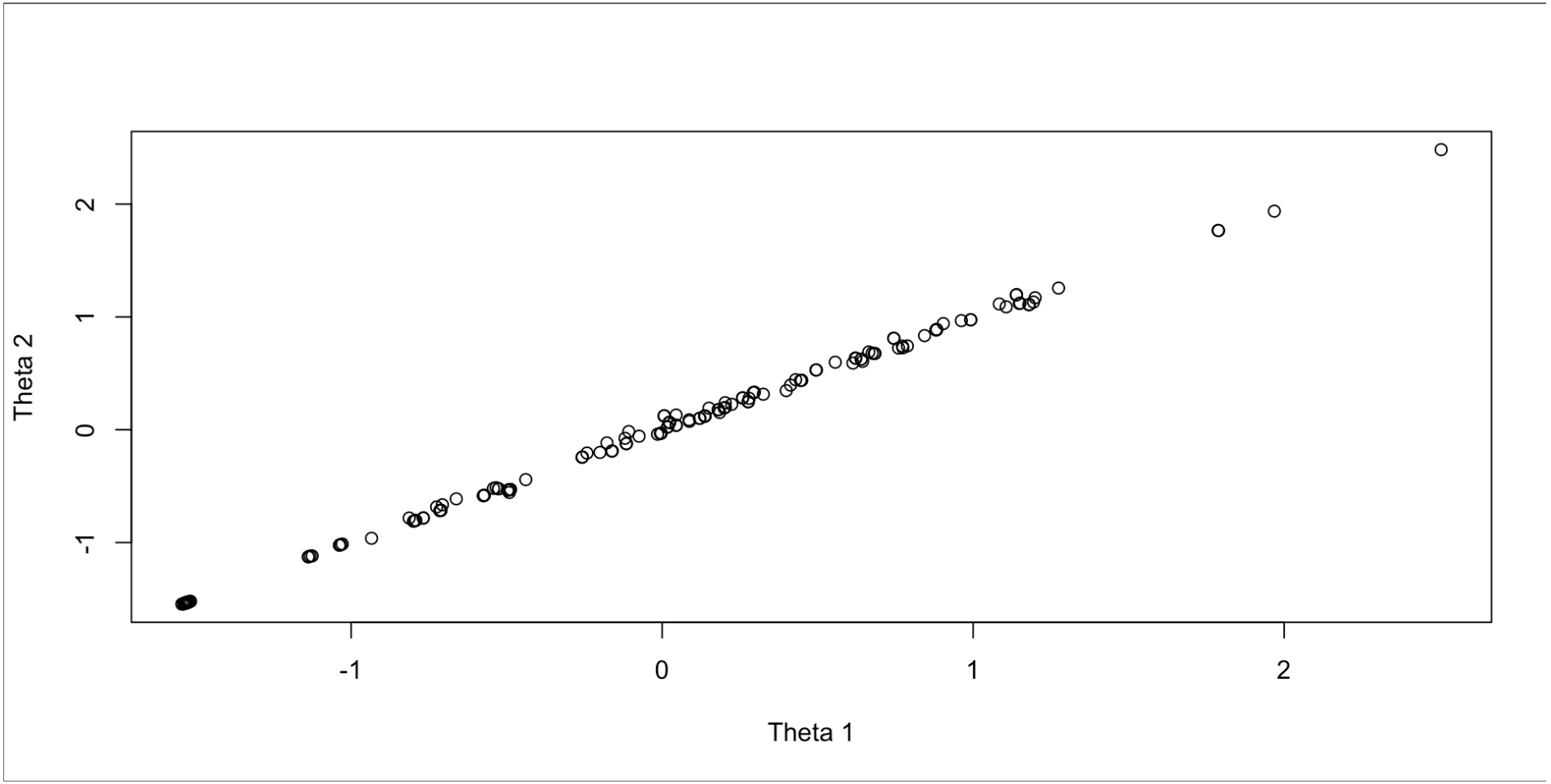
Example Theta Posterior Distribution

Plots of draws for person 1:

```
           theta[1,1] theta[1,2]
theta[1,1]  1.0000000  0.8478048
theta[1,2]  0.8478048  1.0000000
```

EAP Theta Estimates

Plots of draws for person 1:



Wrapping Up

Wrapping Up

Stan makes multidimensional latent variable models fairly easy to implement

- LKJ prior allows for scale identification with standardized factors
- Can use my syntax and import any type of Q-matrix

But…

- Estimation is slower (correlations take time)

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