Multivariate Statistics and Methodology with R

Dimension Reduction

2025-2026

Overview

- Week 1: Dimension Reduction (Principal Components Analysis and Exploratory Factor Analysis)
- Week 2: SEM I - Confirmatory Factor Analysis
- Week 3: SEM II - Path Analysis
- Week 4: SEM III - Full SEM
- Week 5: SEM IV - Practical Issues in SEM

This Week

- Techniques
  - *Principal Components Analysis (PCA)*
  - *Exploratory Factor Analysis (EFA)*
- Key R Functions
  - *vss( )*
  - *fa.parallel( )*
  - *principal( )*
  - *fa( )*
- Reading: *Principal Components Analysis* and *Exploratory Factor Analysis* Chapters (on *Learn*)
- Office hours: by appointment (aja.murray@ed.ac.uk)

Dimension Reduction

- Summarise a set of variables in terms of a smaller number of dimensions

  - *e.g., can 10 aggression items summarised in terms of 'physical' and 'verbal' aggression dimensions?*

    1. I hit someone

    2. I kicked someone

    3. I shoved someone

    4. I battered someone

    5. I physically hurt someone on purpose

    6. I deliberately insulted someone

    7. I swore at someone

    8. I threatened to hurt someone

    9. I called someone a nasty name to their face

10. I shouted mean things at someone

Uses of dimension reduction techniques

- Theory testing
  - *What are the number and nature of dimensions that best describe a theoretical construct?*
- Test construction
  - *How should I group my items into subscales?*
  - *Which items are the best measures of my constructs?*
- Pragmatic
  - *I have multicollinearity issues/too many variables, how can I defensibly combine my variables?*

Our running example

- A researcher has collected n=1000 responses to our 10 aggression items
- We'll use this data to illustrate dimension reduction techniques

```
library(psych)
describe(agg.items)[c('vars','n','mean','sd','min','max')]
```

```
##        vars    n mean   sd   min  max
## item1     1 1000 0.06 0.95 -3.22 3.07
## item2     2 1000 0.04 1.02 -3.10 3.35
## item3     3 1000 0.07 1.02 -3.17 3.30
## item4     4 1000 0.04 1.03 -3.23 2.87
## item5     5 1000 0.04 0.99 -3.01 4.24
## item6     6 1000 0.00 1.01 -3.10 3.22
## item7     7 1000 0.03 1.01 -2.97 3.21
## item8     8 1000 0.04 0.99 -3.01 3.08
## item9     9 1000 0.04 1.01 -3.19 3.47
## item10   10 1000 0.04 1.00 -3.19 3.49
```

PCA

- Starts with a correlation matrix

```
#compute the correlation matrix for the aggression items
round(cor(agg.items),2)
```

```
##        item1 item2 item3 item4 item5 item6 item7 item8 item9 item10
## item1   1.00  0.56  0.47  0.46  0.56  0.06  0.15  0.11  0.18   0.08
## item2   0.56  1.00  0.58  0.55  0.66  0.06  0.16  0.13  0.18   0.11
## item3   0.47  0.58  1.00  0.47  0.58  0.06  0.16  0.11  0.15   0.08
## item4   0.46  0.55  0.47  1.00  0.58  0.07  0.16  0.13  0.16   0.10
## item5   0.56  0.66  0.58  0.58  1.00  0.03  0.12  0.09  0.15   0.06
## item6   0.06  0.06  0.06  0.07  0.03  1.00  0.56  0.59  0.40   0.47
## item7   0.15  0.16  0.16  0.16  0.12  0.56  1.00  0.79  0.60   0.63
## item8   0.11  0.13  0.11  0.13  0.09  0.59  0.79  1.00  0.60   0.61
## item9   0.18  0.18  0.15  0.16  0.15  0.40  0.60  0.60  1.00   0.48
## item10  0.08  0.11  0.08  0.10  0.06  0.47  0.63  0.61  0.48   1.00
```

What PCA does

- Repackages the variance from the correlation matrix into a set of **components**
- Components = orthogonal (uncorrelated) linear combinations of the original variables
  - *1st component is the linear combination that accounts for the most possible variance*
  - *2nd accounts for second-largest after the variance accounted for by the first is removed*
  - *3rd…etc…*
- Each component accounts for as much remaining variance as possible
- There are as many components are there were variables in original correlation matrix

Eigendecomposition

- Components are formed using an **eigen-decomposition** of the correlation matrix
- Eigen-decomposition is a transformation of the correlation matrix to re-express it in terms of **eigenvalues** and **eigenvectors**

Eigenvalues and eigenvectors

```
## [1] "e1" "e2" "e3" "e4" "e5"
```

```
##       component1 component2 component3 component4 component5
## item1 "w11"      "w12"      "w13"      "w14"      "w15"
## item2 "w21"      "w22"      "w23"      "w24"      "w25"
## item3 "w31"      "w32"      "w33"      "w34"      "w35"
## item4 "w41"      "w42"      "w43"      "w44"      "w45"
## item5 "w51"      "w52"      "w53"      "w54"      "w55"
```

- One eigenvector and one eigenvalue for each component
- Eigenvalues are a measure of the size of the variance packaged into a component
  - *Larger eigenvalue = component accounts for larger proportion of the variance in original correlation matrix*
- Eigenvectors are sets of **weights** (one weight per variable in original correlation matrix)
  - *Larger weight = variable makes a bigger contribution to the component*

Eigen-decomposition of aggression item correlation matrix

- We can use the eigen() function to conduct an eigen-decomposition for our 10 aggression items

```
eigen(cor(agg.items))
```

```
## eigen() decomposition
## $values
##  [1] 3.8309985 2.6926222 0.6007752 0.5397370 0.5273152 0.4957247 0.4066606
##  [8] 0.3687049 0.3351602 0.2023015
##
## $vectors
##             [,1]       [,2]         [,3]         [,4]         [,5]        [,6]
##  [1,] -0.2899201  0.3083765  0.225163086  0.655986563 -0.367419171  0.24541217
##  [2,] -0.3206870  0.3434986 -0.037030321 -0.001506561  0.080026050  0.14403914
##  [3,] -0.2918209  0.3157989 -0.186431589  0.013380783  0.750020579 -0.13529293
##  [4,] -0.2933252  0.3047722 -0.117392886 -0.610086868 -0.504253640 -0.24416979
##  [5,] -0.3046599  0.3726670 -0.024739322 -0.034256344 -0.004440499  0.03268042
##  [6,] -0.2749803 -0.3075372 -0.697792682  0.350175856 -0.181086139 -0.27520887
##  [7,] -0.3732449 -0.3120661  0.051613626 -0.059907620  0.056826412  0.01463529
##  [8,] -0.3586401 -0.3389710  0.013173634 -0.038017144  0.018289887 -0.06716292
##  [9,] -0.3320490 -0.2319783  0.639534898  0.045286753  0.057700437 -0.45550756
## [10,] -0.3086814 -0.3077941 -0.004843031 -0.257706392  0.045731527  0.74302248
##              [,7]        [,8]         [,9]       [,10]
```

```
##  [1,] -0.36923225 -0.07445609 -0.031945842  0.04138553
##  [2,]  0.54476093  0.18498983 -0.647077368 -0.03080181
##  [3,] -0.38153968 -0.22551095 -0.002702482  0.05390537
##  [4,] -0.28622021 -0.17435683 -0.092955422  0.01528197
##  [5,]  0.38624296  0.23162058  0.749633754 -0.02541482
##  [6,]  0.19747867 -0.26250758  0.022434957 -0.05766650
##  [7,] -0.25506626  0.44449120 -0.027543214 -0.70015484
##  [8,] -0.13965071  0.48250127 -0.028738815  0.70489429
##  [9,]  0.25207195 -0.38726830  0.012873585 -0.01935388
## [10,]  0.05590792 -0.41960052  0.086127575  0.05254038
```

QUIZ QUESTION 1

- What is the name of the process by which a correlation matrix is transformed into eigenvectors and eigenvalues?
    - *1. eigen-sedimentation*
    - *2. eigen-consolidation*
    - *3. eigen-diversification*
    - *4. eigen-decomposition*

How many components to keep?

- Eigen-decomposition repackages the variance but does not reduce our dimensions
- Dimension reduction comes from keeping only the largest components
- Assumes the others can be dropped with little loss of information
- Our decisions on how many components to keep can be guided by several methods
  - *Scree plot*
  - *Minimum average partial test (MAP)*
  - *Parallel analysis*

Other considerations in how many components to keep

- Substantive considerations
  - *Do the selected components make theoretical sense?*
- Practical considerations
  - *Are some components too 'minor' to be reliable?*

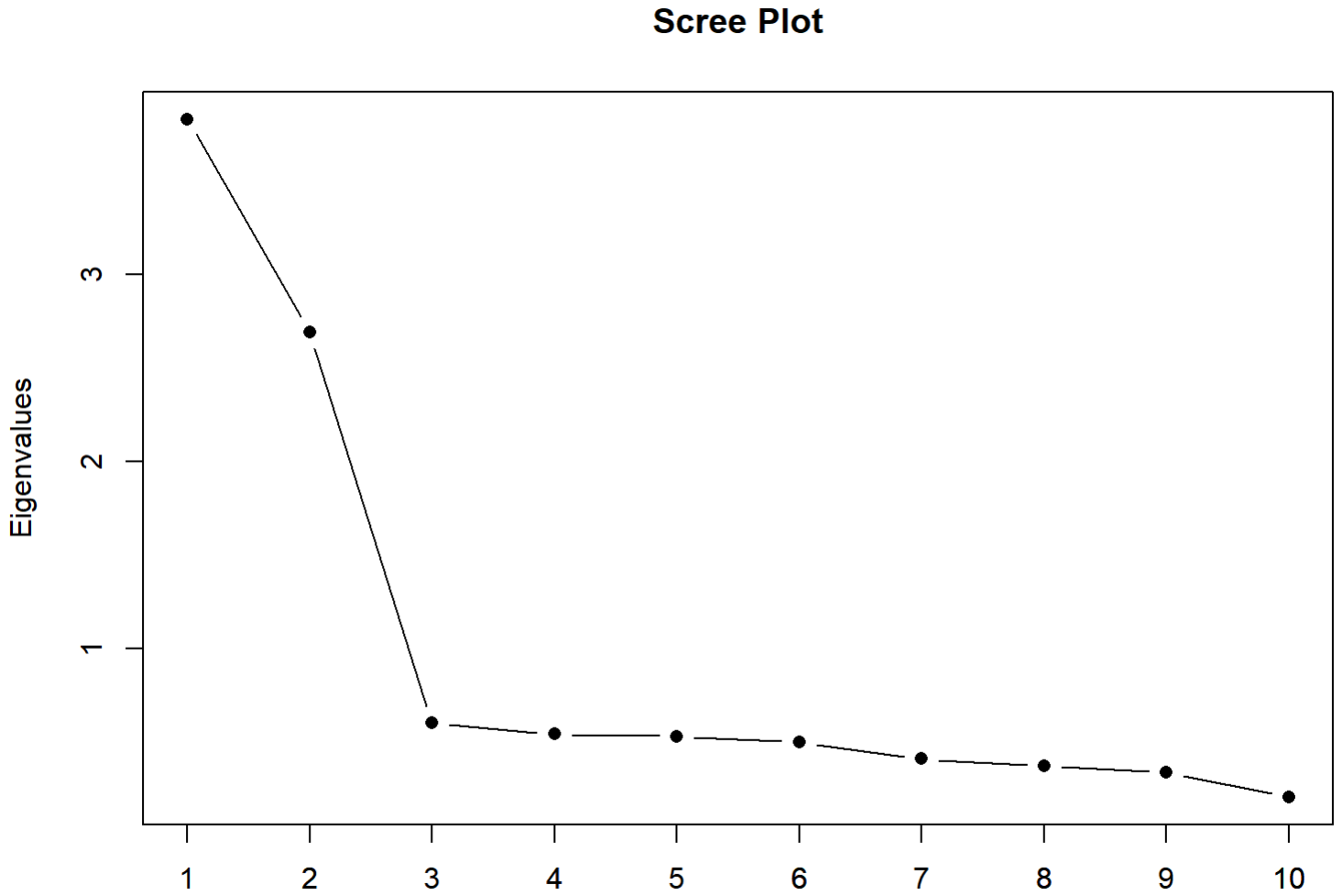
Kaiser criterion

- Keeps number of components with eigenvalue >1
- DO NOT USE Kaiser criterion
- Often suggests keeping far too many components

Scree plot

- Based on plotting the eigenvalues
- Looking for a sudden change of slope
- Assumed to potentially reflect point at which components become substantively unimportant

Constructing a scree plot

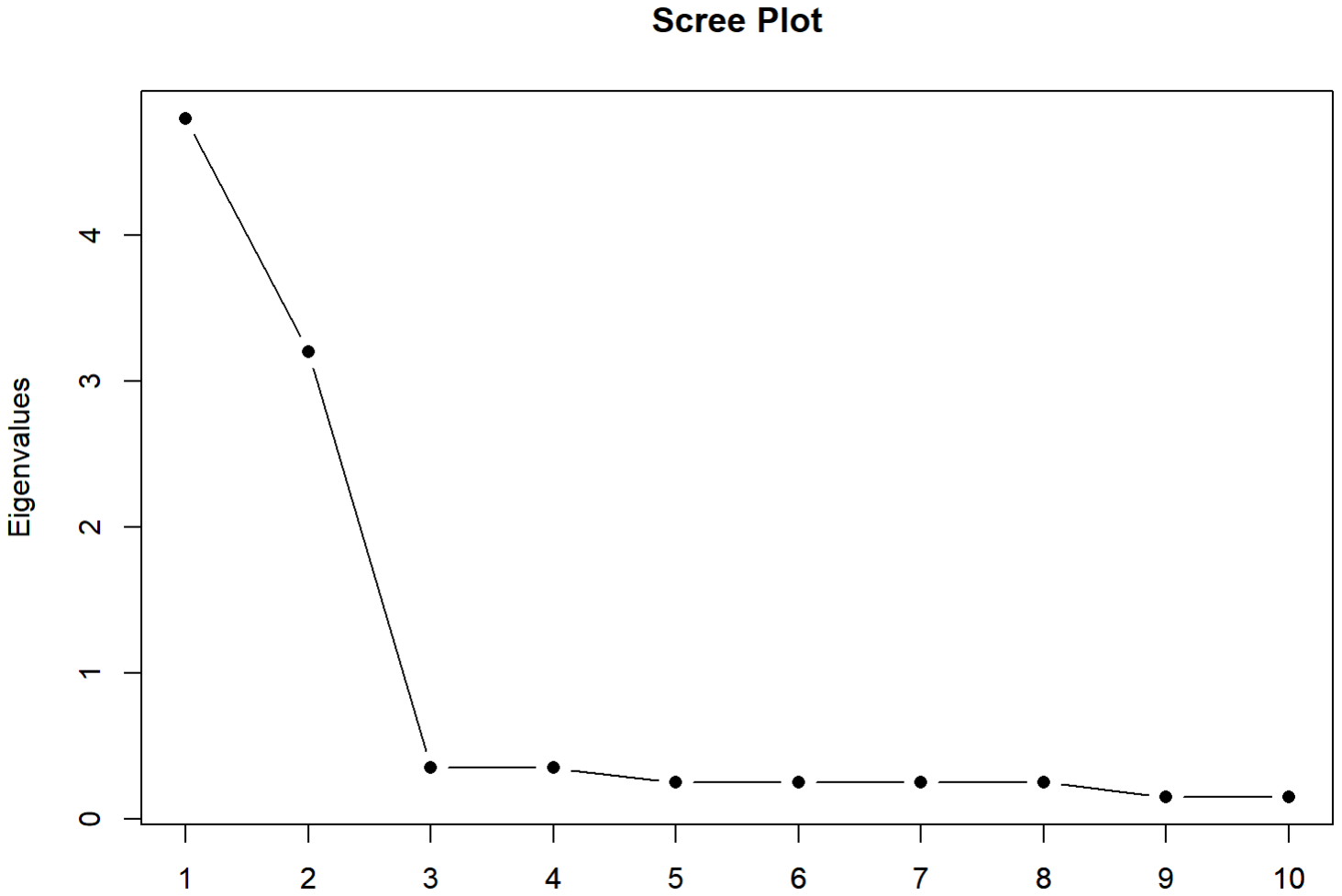


- Eigenvalue plot
  - *x-axis is component number*
  - *y-axis is eigenvalue for each component*
- Keep the components with eigenvalues above a kink in the plot

Further scree plot examples

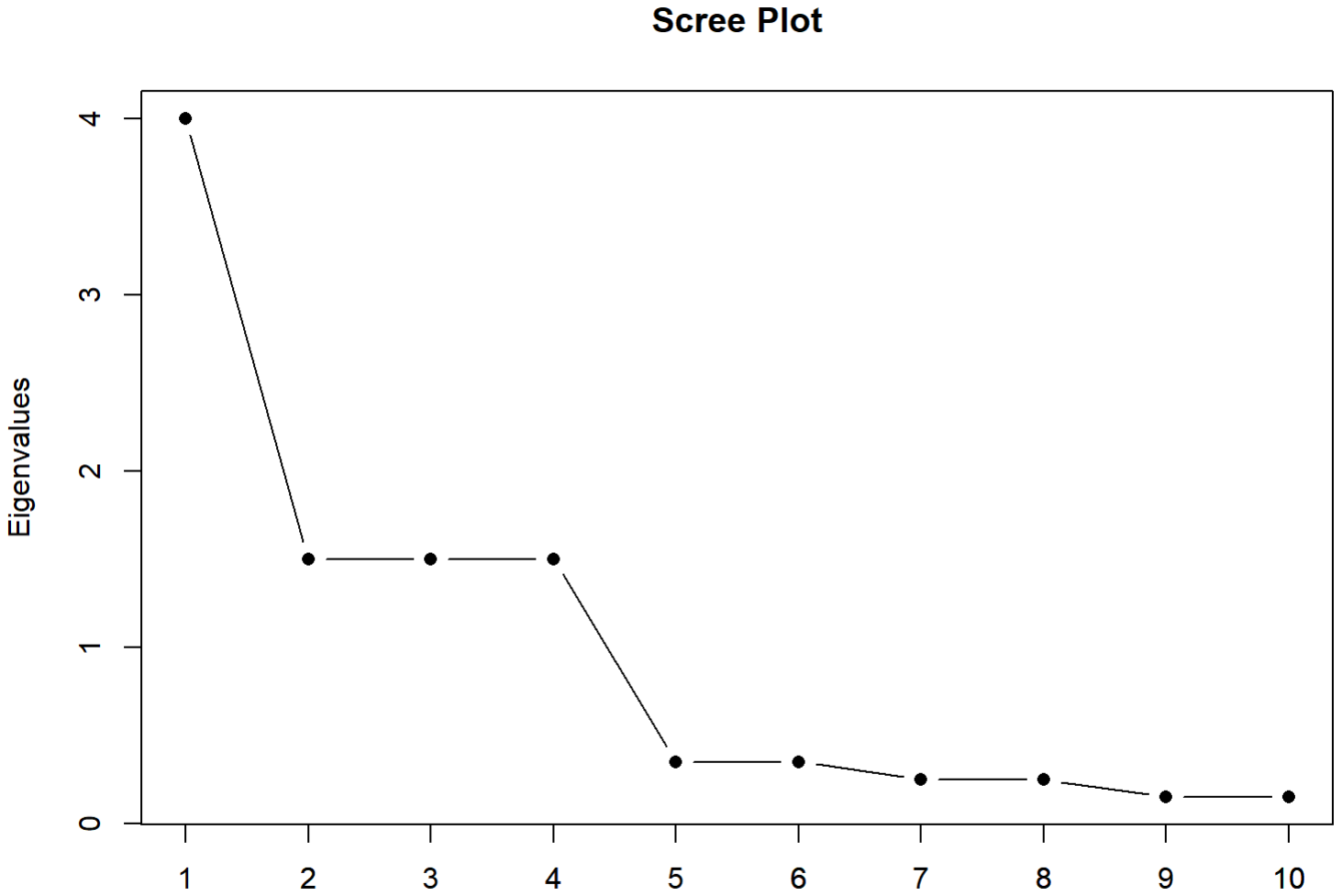
- Scree plots vary in how easy it is to interpret them

```
## [1] 10
```


Scree Plot
Eigenvalues
0
1
2
3
4
1
2
3
4
5
6
7
8
9
10

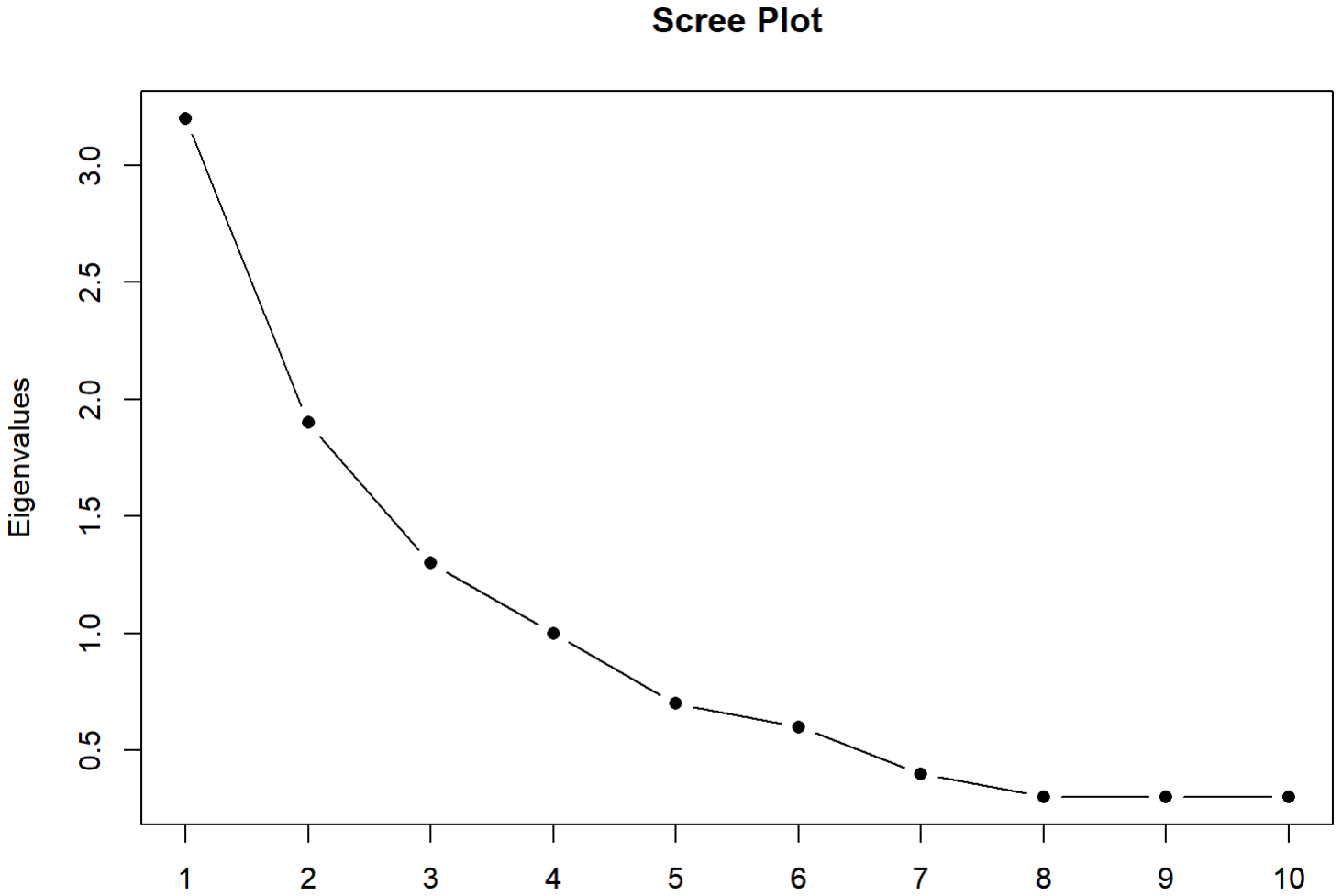
Further scree plot examples

```
## [1] 10
```


Scree Plot
Eigenvalues
0
1
2
3
4
1
2
3
4
5
6
7
8
9
10

Further scree plot examples

```
## [1] 10
```


Scree Plot
Eigenvalues
0.5
1.0
1.5
2.0
2.5
3.0
1
2
3
4
5
6
7
8
9
10

Minimum average partial test (MAP)

- Extracts components iteratively from the correlation matrix
- Computes the average squared partial correlation after each extraction
- At first this quantity goes down with each component extracted but then it starts to increase again
- MAP keeps the components from point at which the average squared partial correlation is at its smallest

MAP test for the aggression items

- We can obtain the results of the MAP test via the vss( ) function from the psych package

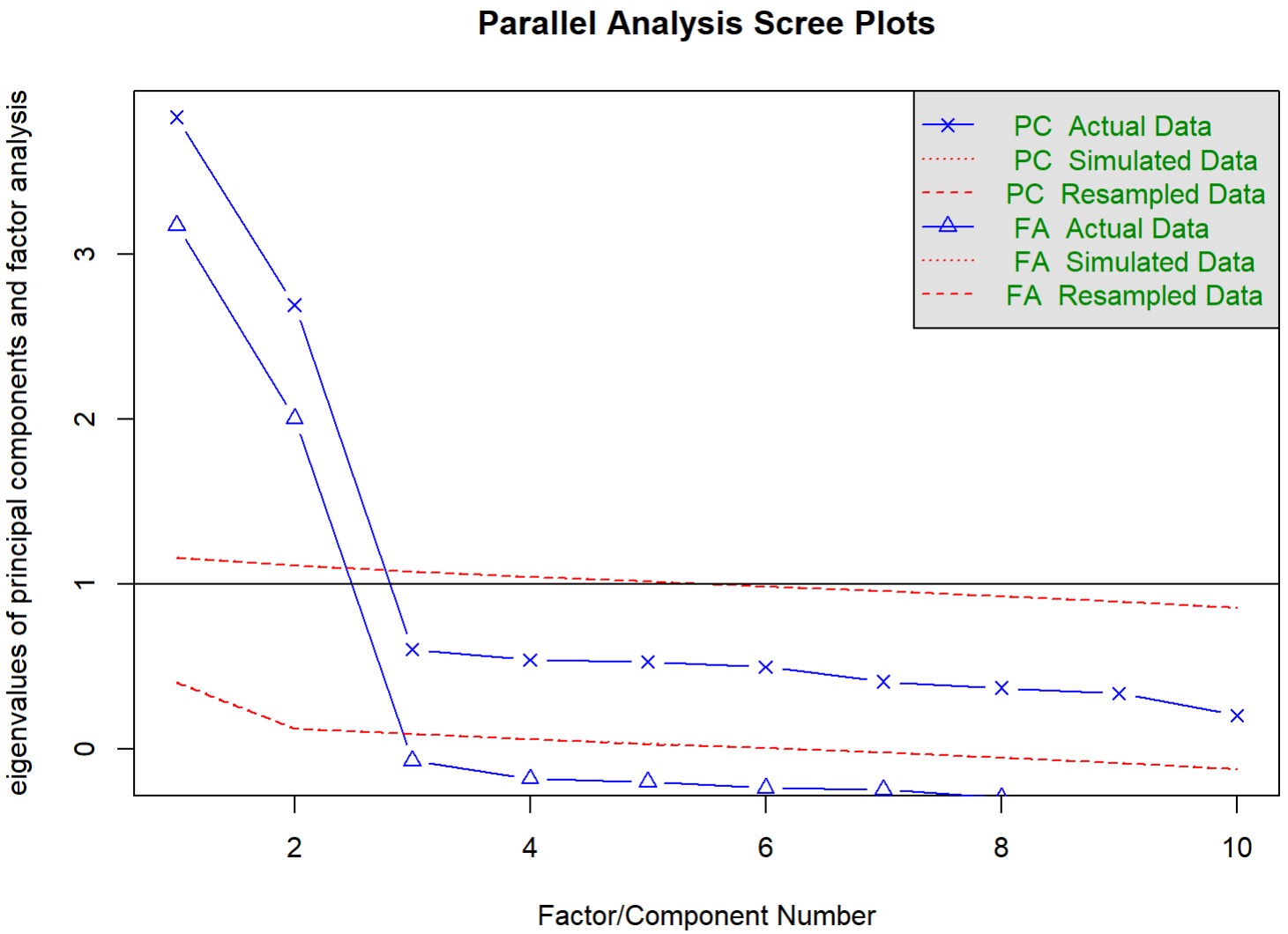
```
library(psych)
vss(agg.items, plot=F)
```

```
##
## Very Simple Structure
## Call: vss(x = agg.items, plot = F)
## Although the VSS complexity 1 shows  8  factors, it is probably more reasonable to think
about  2  factors
## VSS complexity 2 achieves a maximimum of 0.93  with  6  factors
##
## The Velicer MAP achieves a minimum of 0.03  with  2  factors
## BIC achieves a minimum of  -155.98  with  2  factors
## Sample Size adjusted BIC achieves a minimum of  -73.4  with  2  factors
##
## Statistics by number of factors
##   vss1 vss2   map dof   chisq prob sqresid  fit RMSEA  BIC SABIC complex
## 1 0.60 0.00 0.154  35 2.5e+03 0.00     9.4 0.60  0.27 2274  2385     1.0
## 2 0.88 0.91 0.029  26 2.4e+01 0.60     2.0 0.91  0.00 -156   -73     1.0
## 3 0.79 0.91 0.060  18 1.7e+01 0.51     1.7 0.93  0.00 -107   -50     1.1
## 4 0.70 0.91 0.098  11 7.5e+00 0.75     1.5 0.94  0.00  -68   -34     1.2
## 5 0.69 0.91 0.156   5 3.1e+00 0.68     1.4 0.94  0.00  -31   -16     1.3
## 6 0.63 0.93 0.247   0 8.7e-01   NA     1.0 0.96    NA   NA    NA     1.4
## 7 0.88 0.92 0.438  -4 1.5e-02   NA     1.5 0.94    NA   NA    NA     1.3
```

```
## 8 0.88 0.90 0.476  -7 4.1e-07   NA     1.6 0.93    NA   NA    NA     1.2
##    eChisq    SRMR  eCRMS eBIC
## 1 4.6e+03 2.3e-01 0.2561 4349
## 2 8.2e+00 9.6e-03 0.0126 -171
## 3 4.4e+00 7.0e-03 0.0111 -120
## 4 1.8e+00 4.5e-03 0.0091  -74
## 5 8.0e-01 3.0e-03 0.0089  -34
## 6 1.9e-01 1.4e-03     NA   NA
## 7 6.2e-03 2.6e-04     NA   NA
## 8 1.1e-07 1.1e-06     NA   NA
```

Parallel analysis for the aggression items

```
fa.parallel(agg.items, n.iter=500)
```



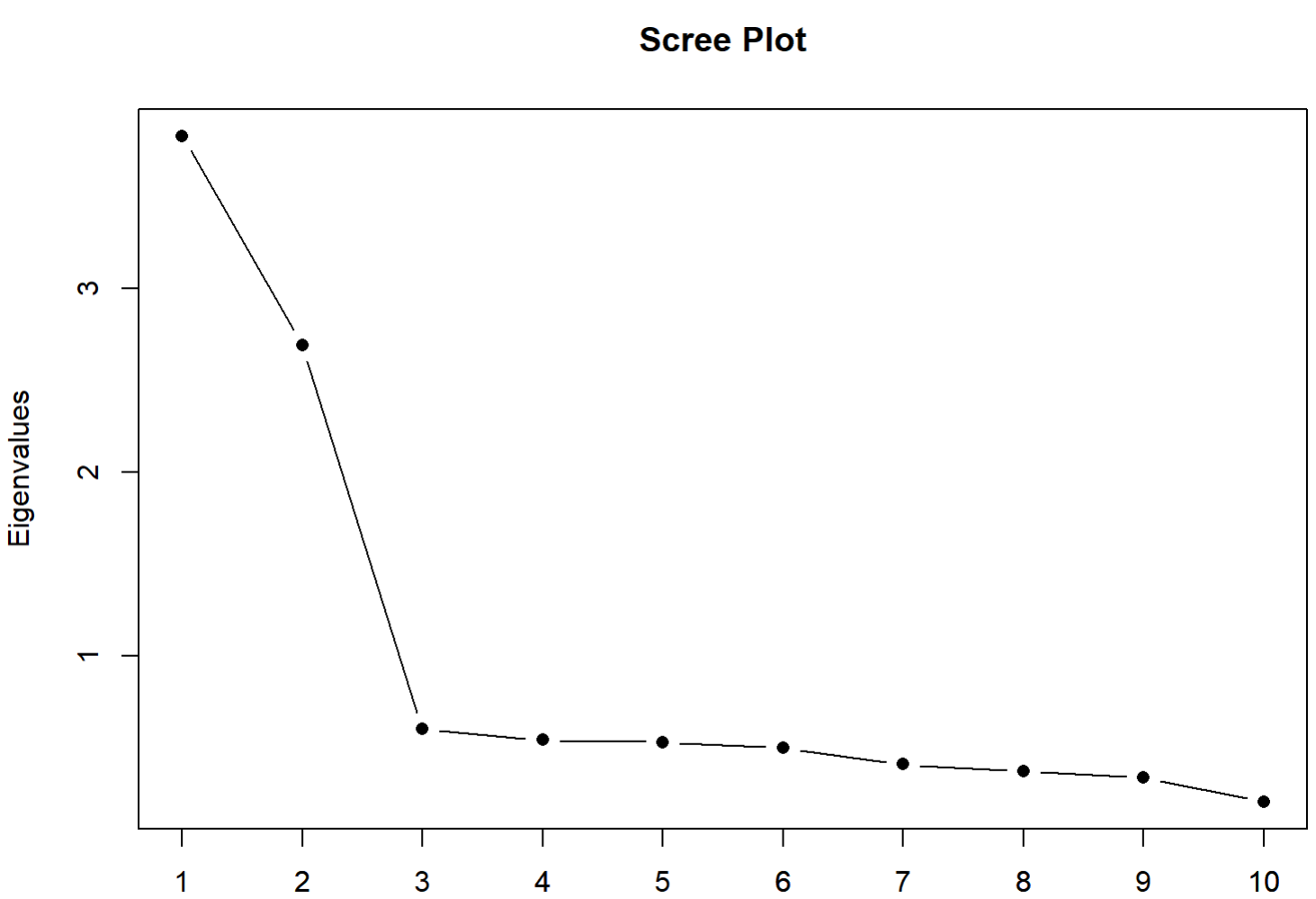

```
## Parallel analysis suggests that the number of factors =  2  and the number of components =
2
```

The fa.parallel( ) function

- Notice the function also gives us a scree plot
- We can use this to find a point of inflection
  - *Use the 'PC Actual Data' datapoints*
- However, if we want to include a scree plot in a report we should construct our own…

Example code for a scree plot

```
eigenvalues<-eigen(cor(agg.items))$values
plot(eigenvalues, type = 'b', pch = 16, main = "Scree Plot", xlab="", ylab="Eigenvalues")
axis(1, at = 1:10, labels = 1:10)
```


Scree Plot
Eigenvalues
1
2
3
1
2
3
4
5
6
7
8
9
10

Limitations of scree, MAP, and parallel analysis

- No one right answer about the number of components to retain
- Scree plot, MAP and parallel analysis frequently disagree
- Each method has weaknesses
  - *Scree plots are subjective and may have multiple or no obvious kinks*
  - *Parallel analysis sometimes suggest too many components*
  - *MAP sometimes suggests too few components*

Quiz question 2

- Which components are retained based on a scree plot?
  - *1. Those with eigenvalues up to and including the kink*
  - *2. Those with eigenvalues >2*
  - *3. Those with eigenvalues before the kink*
  - *4. Those up to the point where the average squared partial correlation is at its minimum*

Running a PCA with a reduced number of components

- We can run a PCA keeping just a selected number of components
- We do this using the principal() function from the psych package
- We supply the dataframe or correlation matrix as the first argument
- We specify the number of components to retain with the nfactors= argument
- We specify rotate='none' to keep the components uncorrelated

```
PC2<-principal(agg.items, nfactors=2, rotate='none')
```

Interpreting the components

- Once we have decided how many components to keep we interpret the PCA solution
- We do this based on the component loadings
  - *Component loadings are calculated from the values in the eigenvectors*
  - *They can be interpreted as the correlations between variables and components*

The component loadings

- Component loading matrix
- RC1 and RC2 columns show the component loadings

```
PC2<-principal(r=agg.items, nfactors=2, rotate='none')
PC2$loadings
```

```
##
## Loadings:
##        PC1    PC2
## item1   0.567  0.506
## item2   0.628  0.564
## item3   0.571  0.518
## item4   0.574  0.500
## item5   0.596  0.612
## item6   0.538 -0.505
## item7   0.731 -0.512
## item8   0.702 -0.556
## item9   0.650 -0.381
## item10  0.604 -0.505
##
##                  PC1   PC2
## SS loadings    3.831 2.693
## Proportion Var 0.383 0.269
## Cumulative Var 0.383 0.652
```

Interpreting the components

1. I hit someone
2. I kicked someone
3. I shoved someone
4. I battered someone
5. I physically hurt someone on purpose
6. I deliberately insulted someone
7. I swore at someone
8. I threatened to hurt someone
9. I called someone a nasty name to their face
10. I shouted mean things at someone

How good is my PCA solution?

- A good PCA solution explains the variance of the original correlation matrix in as few components as possible

```
PC2$loadings
```

```
##
## Loadings:
##        PC1    PC2
## item1   0.567  0.506
## item2   0.628  0.564
## item3   0.571  0.518
## item4   0.574  0.500
## item5   0.596  0.612
## item6   0.538 -0.505
## item7   0.731 -0.512
## item8   0.702 -0.556
## item9   0.650 -0.381
## item10  0.604 -0.505
##
##                  PC1   PC2
## SS loadings    3.831 2.693
## Proportion Var 0.383 0.269
## Cumulative Var 0.383 0.652
```

Computing scores for the components

- After conducting a PCA you may want to create scores for the new dimensions
  - *e.g., to use in a regression*
- Simplest method is to sum the scores for all items with loadings >|.3|
- Better method is to compute them taking into account the weights

Computing component scores in R

```
PC<-principal(r=agg.items, nfactors=2, rotate='none')
scores<-PC$scores
head(scores)
```

```
##              PC1        PC2
## [1,]  0.2711811  1.9216506
## [2,] -1.2291255  0.2168795
## [3,]  1.1834948 -0.1865432
## [4,]  0.4967953  0.6992322
## [5,] -0.4075699  1.4900556
## [6,] -0.3676440  0.4053956
```

Reporting a PCA

- Main principles: transparency and reproducibility
- Method
  - *Methods used to decide on number of components*
- Results
  - *Results of MAP, parallel analysis, scree test (& any other considerations in choice of number of components)*
  - *How many components retained*
  - *The loading matrix for the chosen solution*
  - *Variance explained by components*
  - *Labelling and interpretation of the components*

PCA Summary

- PCA is a common dimension reduction technique
- Steps are:
  - *Decide how many components to keep (scree plot, parallel analysis, MAP test)*
  - *Interpret solution (loadings, variance explained)*
  - *(Optional) calculate and use scores*

END OF PCA

Exploratory factor analysis

- Used to identify number & nature of dimensions that describe a psychological construct and their inter-relations
- Procedurally similar to PCA but differs in important ways
  - *Uses only the common variance in its calculation*
  - *Can give quite different results to PCA under some circumstances*
  - *Resulting dimensions are called* ***factors***
  - *EFA based on a* ***latent variable model***

Latent variable models

- Divides the world into **observed variables** and **latent variables** (factors)
  - *Observed variables can be measured directly*
    - e.g., scores on IQ subtests
  - *Latent variables inferred based on patterns of observed variable associations*
    - e.g., Spearman's *g*
- Latent variables generate the correlations between observed variables
  - *e.g., higher g causes higher subtest scores*
- Observed variables are imperfect **indicators** (measures) of latent variables
  - *Observed variable scores have both a systematic and a random error component*

Doing EFA

- Like PCA, but some extra decisions:
  - *How many factors?*
  - ***Which rotation?***
  - ***Which extraction method?***
- In EFA we also have to choose an extraction/estimation method and rotation

How many factors?

- As in PCA, we can use the following tools to help us decide how many factors to retain:
  - *Scree test*
  - *Parallel analysis*
  - *MAP test*
- It is also important to examine the factor solutions for varying numbers of factors
  - *Which solutions make more sense based on our background knowledge of the construct?*
  - *Do some solutions have deficiencies such as minor factors?*

Our running example

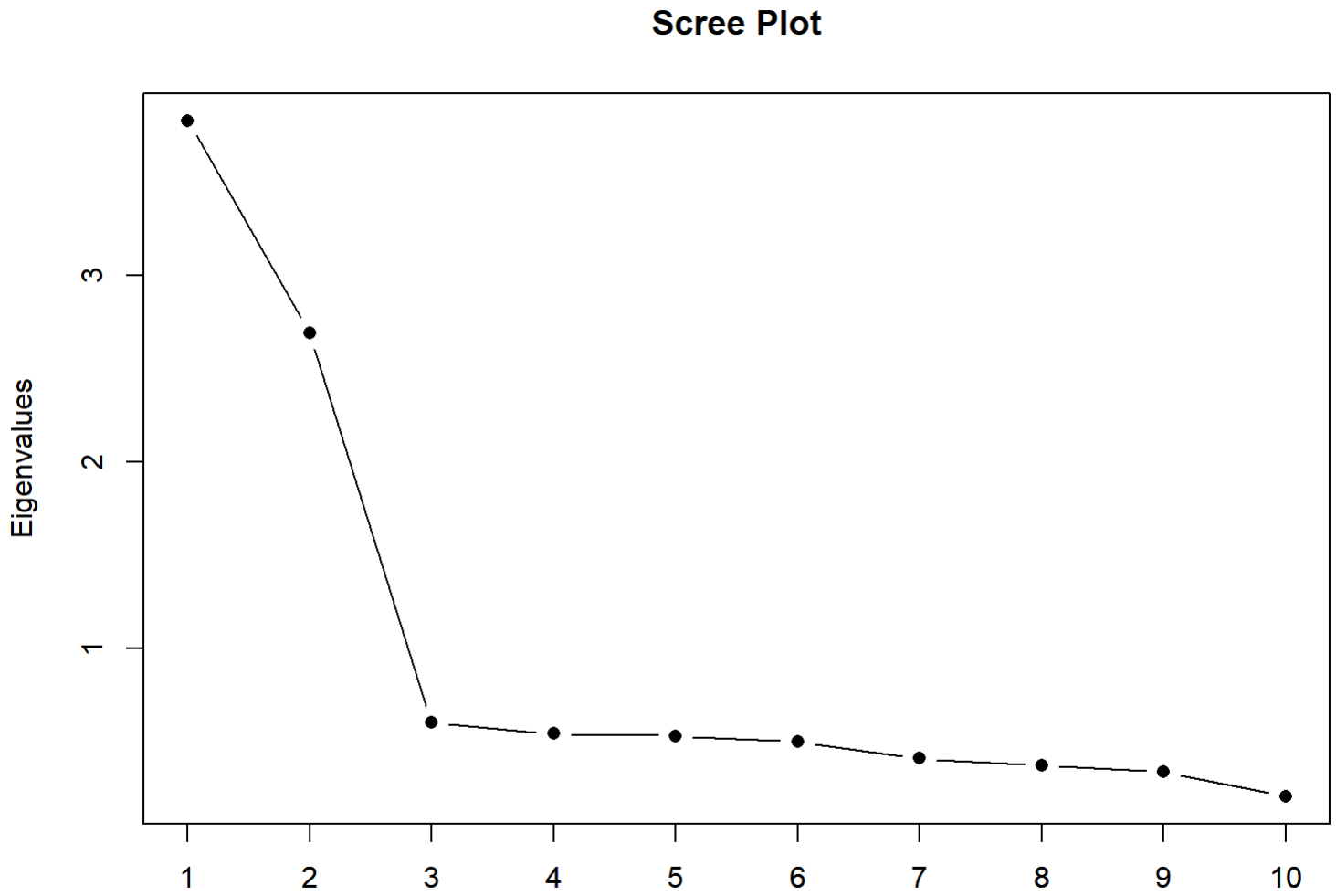
- Let's return to our aggression example and now run an EFA
- We had n=1000 participants with data on the following 10 items:

1. *I hit someone*
2. *I kicked someone*
3. *I shoved someone*
4. *I battered someone*
5. *I physically hurt someone on purpose*
6. *I deliberately insulted someone*
7. *I swore at someone*
8. *I threatened to hurt someone*
9. *I called someone a nasty name to their face*

10. I shouted mean things at someone

How many aggression factors? Scree test

- We can plot the eigenvalues and look for a kink in the plot:



How many aggression factors? MAP

- We can conduct a MAP test using vss( ):

```
library(psych)
vss(agg.items, plot=F)
```

```
##
## Very Simple Structure
## Call: vss(x = agg.items, plot = F)
## Although the VSS complexity 1 shows  8  factors, it is probably more reasonable to think
about  2  factors
## VSS complexity 2 achieves a maximimum of 0.93  with  6  factors
##
## The Velicer MAP achieves a minimum of 0.03  with  2  factors
## BIC achieves a minimum of  -155.98  with  2  factors
## Sample Size adjusted BIC achieves a minimum of  -73.4  with  2  factors
##
## Statistics by number of factors
##   vss1 vss2   map dof   chisq prob sqresid  fit RMSEA  BIC SABIC complex
## 1 0.60 0.00 0.154  35 2.5e+03 0.00     9.4 0.60  0.27 2274  2385     1.0
## 2 0.88 0.91 0.029  26 2.4e+01 0.60     2.0 0.91  0.00 -156   -73     1.0
## 3 0.79 0.91 0.060  18 1.7e+01 0.51     1.7 0.93  0.00 -107   -50     1.1
## 4 0.70 0.91 0.098  11 7.5e+00 0.75     1.5 0.94  0.00  -68   -34     1.2
## 5 0.69 0.91 0.156   5 3.1e+00 0.68     1.4 0.94  0.00  -31   -16     1.3
## 6 0.63 0.93 0.247   0 8.7e-01   NA     1.0 0.96    NA   NA    NA     1.4
## 7 0.88 0.92 0.438  -4 1.5e-02   NA     1.5 0.94    NA   NA    NA     1.3
## 8 0.88 0.90 0.476  -7 4.1e-07   NA     1.6 0.93    NA   NA    NA     1.2
##    eChisq    SRMR  eCRMS eBIC
```

```
## 1 4.6e+03 2.3e-01 0.2561 4349
## 2 8.2e+00 9.6e-03 0.0126 -171
## 3 4.4e+00 7.0e-03 0.0111 -120
## 4 1.8e+00 4.5e-03 0.0091  -74
## 5 8.0e-01 3.0e-03 0.0089  -34
## 6 1.9e-01 1.4e-03     NA   NA
## 7 6.2e-03 2.6e-04     NA   NA
## 8 1.1e-07 1.1e-06     NA   NA
```

Examining the factor solutions

- Finally, we draw on information from the factor solutions themselves
- We run a series of factor analysis models with different numbers of factors
- Look at the loadings and factor correlations:
  - *Are important distinctions blurred when the number of factors is smaller?*
  - *Are there minor or 'methodological' factors when the number of factors is larger?*
  - *Are the factor correlations very high?*
  - *Do the factor solutions make theoretical sense?*
- In this case, given the MAP, scree and parallel analysis results we would likely want to examine the 1,2 and 3 factor solutions

Rotation of factors

- Rotation takes an initial EFA solution and transforms it to make it more interpretable
- An initial EFA solution typically has:
  - *has high loadings on the first component*
  - *has a mix of positive and negative loadings on subsequent components*
  - *is difficult to interpret*
- We typically try to achieve *simple structure* with a rotation
  - *each item has a high loading on one component and close to zero loading on all others*

Initial EFA solution for the aggression items

```
FA_initial<-fa(r=agg.items, nfactors=2, rotate='none')
FA_initial$loadings
```

```
##
## Loadings:
##        MR1    MR2
## item1   0.506  0.458
## item2   0.592  0.554
## item3   0.513  0.473
## item4   0.513  0.453
## item5   0.565  0.606
## item6   0.492 -0.413
## item7   0.741 -0.489
## item8   0.716 -0.536
## item9   0.597 -0.311
## item10  0.566 -0.428
##
##                  MR1   MR2
## SS loadings    3.432 2.289
## Proportion Var 0.343 0.229
## Cumulative Var 0.343 0.572
```

Different types of rotation

- The initial (unrotated) loading matrix is transformed by multiplication by a *transformation matrix*
- Different transformation matrices are used to achieve different transformations
- The most important distinction is between *orthogonal* versus *oblique* rotations
  - *Orthogonal rotations force the components to remain uncorrelated*
    - They include varimax, quartimax and equamax
  - *Oblique rotations allow the components to be correlated*
    - They include oblimin, promax, direct oblimin, and quartimin

Choosing a rotation

- Orthogonal rotations are useful for e.g. reducing multicollinearity in regression
- Oblique rotations better reflect the reality that psychological constructs tend to be correlated
- Advice: use an oblique rotation and switch to orthogonal if correlation is very low
  - *Oblimin is a good choice for oblique rotation*
  - *Varimax is a good choice for orthogonal rotation*
  - *… but trying a few and comparing is a good idea*

Interpreting an oblique rotation

- When an orthogonal rotation is used only one loading matrix is produced
- When an oblique rotation is used two loading matrices are produced:
  - *structure matrix (correlations between the components and the variables)*
  - *pattern matrix (regression weights from the components to the variables)*
- Pattern is likely to be most useful for interpreting the components

EFA solution for the aggression items using an oblique rotation

```
FA2<-fa(r=agg.items, nfactors=2, rotate='oblimin')
```

```
## Loading required namespace: GPArotation
```

```
FA2$loadings
```

```
##
## Loadings:
##        MR1    MR2
## item1           0.678
## item2           0.808
## item3           0.694
## item4           0.678
## item5           0.836
## item6   0.650
## item7   0.882
## item8   0.899
## item9   0.649
## item10  0.714
##
##                  MR1   MR2
## SS loadings    2.944 2.767
```

```
## Proportion Var 0.294 0.277
## Cumulative Var 0.294 0.571
```

EFA solution for the aggression items using an oblique rotation

```
FA2<-principal(r=agg.items, nfactors=2, rotate='oblimin')
FA2$Phi
```

```
##           TC1       TC2
## TC1 1.0000000 0.1695367
## TC2 0.1695367 1.0000000
```

Quiz question 1

- Quiz question:
  - *Why do we conduct a factor rotation?*
    - 1. increase the amount of variance explained
    - 2. improve the reliability of the factors
    - 3. make the factors more interpretable
    - 4. rotations should always be avoided

Conducting EFA in R

- We can run our factor analyses using the fa() function
- The first argument is the dataset with the items we want to factor analyse
- We also need to mention the number of factors we want to extract, e.g., nfactors=1

```
onef<-fa(agg.items, nfactors=1) #EFA with 1 factor
```

The one-factor solution

- To help us choose an optimal number of factors, we can look at the one-factor solution…

```
onef<-fa(agg.items, nfactors=1) #EFA with 1 factor
onef$loadings #inspect the factor loadings
```

```
##
## Loadings:
##        MR1
## item1  0.472
## item2  0.531
## item3  0.476
## item4  0.479
## item5  0.495
## item6  0.500
## item7  0.729
## item8  0.694
## item9  0.617
## item10 0.572
##
##                  MR1
## SS loadings    3.176
## Proportion Var 0.318
```

The two-factor solution

- And compare with the two-factor solution…

```
library(psych)
twof<-fa(agg.items, nfactors=2, rotate='oblimin') #EFA with 2 factors
twof$loadings ##inspect the factor loadings
```

```
##
## Loadings:
##        MR1    MR2
## item1           0.678
## item2           0.808
## item3           0.694
## item4           0.678
## item5           0.836
## item6   0.650
## item7   0.882
## item8   0.899
## item9   0.649
## item10  0.714
##
##                  MR1   MR2
## SS loadings    2.944 2.767
## Proportion Var 0.294 0.277
## Cumulative Var 0.294 0.571
```

The two-factor solution factor correlations

```
twof$Phi  ## inspect the factor correlations
```

```
##           MR1       MR2
## MR1 1.0000000 0.1888657
## MR2 0.1888657 1.0000000
```

The three-factor solution

- And the three-factor solution

```
library(psych)
threef<-fa(agg.items, nfactors=3, rotate='oblimin') #EFA with 3 factors
threef$loadings #inspect the factor loadings
```

```
##
## Loadings:
##        MR2    MR1    MR3
## item1   0.673
## item2   0.810
## item3   0.697
## item4   0.680
## item5   0.835
## item6          0.696
## item7          0.865
## item8          0.881
## item9                 0.995
## item10         0.704
##
##                  MR2   MR1   MR3
## SS loadings    2.757 2.511 0.998
## Proportion Var 0.276 0.251 0.100
## Cumulative Var 0.276 0.527 0.627
```

The three-factor solution factor correlations

```
threef$Phi # inspect the factor correlations
```

```
##           MR2       MR1       MR3
## MR2 1.0000000 0.1726127 0.2137118
## MR1 0.1726127 1.0000000 0.6649856
## MR3 0.2137118 0.6649856 1.0000000
```

Factor extraction in EFA

- **Factor extraction** refers to the method of deriving the factors
- PCA is itself an extraction method
- In EFA there are a number of factor extraction options:
  - *principal axis factoring (PAF)*
  - *ordinary least squares (OLS)*
  - *weighted least squares (WLS)*
  - *minres*
  - *maximum likelihood (ML)*

Principal axis factoring (PAF)

- Traditional method
- An eigendecomposition of a reduced form of correlation matrix
  - *Diagonals are replaced by communalities*
  - *Communalities estimates used as starting point*
    - Estimates of the variance shared with other indicators
    - Based on e.g. multiple squared R
  - *Iteratively updated across successive PAFs*
  - *Process terminates when estimates change little across iterations*
- Focus on *common* rather than all variance is key EFA vs PCA distinction

Other extraction methods

- **OLS** finds the factor solution that minimises difference between observed and model-implied covariance matrices
  - *specifically, minimises the sum of squared residuals*
- **WLS** up-weights the variables with higher communalities
- **minres** ignores the diagonals
- **ML** finds the factor solution that maximises the likelihood of the observed covariance matrix

Which to use?

- PAF is a good option
- minres can provide EFA solutions when other methods fail
  - *minres is the default for the fa( ) function*
- choice of extraction method usually makes little difference if:
  - *communalities are similar*
  - *sample size is large*
  - ***the number of variables is large***

PAF

- We can do a factor analysis with PAF by setting fm='pa' in the fa() function:

```
library(psych)
twof<-fa(agg.items, nfactors=2, rotate='oblimin', fm='pa') #EFA with 2 factors
twof$loadings ##inspect the factor loadings
```

```
##
## Loadings:
##        PA1    PA2
## item1           0.678
## item2           0.808
## item3           0.694
## item4           0.678
## item5           0.835
## item6   0.650
## item7   0.882
## item8   0.898
## item9   0.650
## item10  0.714
##
##                  PA1   PA2
## SS loadings    2.943 2.767
## Proportion Var 0.294 0.277
## Cumulative Var 0.294 0.571
```

```
twof$Phi  ## inspect the factor correlations
```

```
##           PA1       PA2
## PA1 1.0000000 0.1889406
## PA2 0.1889406 1.0000000
```

minres

- minres is the default method but we can also explicitly set fm='minres':

```
library(psych)
twof<-fa(agg.items, nfactors=2, rotate='oblimin', fm='minres') #EFA with 2 factors
twof$loadings ##inspect the factor loadings
```

```
##
## Loadings:
##        MR1    MR2
## item1           0.678
## item2           0.808
## item3           0.694
## item4           0.678
## item5           0.836
## item6   0.650
## item7   0.882
## item8   0.899
## item9   0.649
## item10  0.714
##
##                  MR1   MR2
## SS loadings    2.944 2.767
## Proportion Var 0.294 0.277
## Cumulative Var 0.294 0.571
```

```
twof$Phi  ## inspect the factor correlations
```

```
##           MR1       MR2
## MR1 1.0000000 0.1888657
## MR2 0.1888657 1.0000000
```

Interpreting the factor solution

- Label factors on basis of high loading items

```
library(psych)
twof<-fa(agg.items, nfactors=2, rotate='oblimin', fm='minres') #EFA with 2 factors
twof$loadings ##inspect the factor loadings
```

```
##
## Loadings:
##        MR1    MR2
## item1           0.678
## item2           0.808
## item3           0.694
## item4           0.678
## item5           0.836
## item6   0.650
## item7   0.882
## item8   0.899
## item9   0.649
## item10  0.714
##
##                  MR1   MR2
## SS loadings    2.944 2.767
## Proportion Var 0.294 0.277
## Cumulative Var 0.294 0.571
```

Interpreting the factor solution

- Factor 1 could be labelled *verbal aggression* and factor 2 could be labelled *physical aggression*

1. ***I hit someone***
2. ***I kicked someone***
3. ***I shoved someone***
4. ***I battered someone***
5. ***I physically hurt someone on purpose***
6. *I deliberately insulted someone*
7. *I swore at someone*
8. *I threatened to hurt someone*
9. *I called someone a nasty name to their face*
10. *I shouted mean things at someone*

The magnitude of factor loadings

- How large are the loadings?
- Comfrey & Lee (1992) offered the following rules of thumb:
    - *.71 (50% overlapping variance) are considered excellent*
    - *.63 (40% overlapping variance) is very good*
    - *.55 (30% overlapping variance) is good*
    - *.45 (20% overlapping variance) is fair*
    - *.32 (10% overlapping variance) is poor*

The magnitude of factor correlations

- How distinct are the factors?

```
library(psych)
twof<-fa(agg.items, nfactors=2, rotate='oblimin', fm='minres') #EFA with 2 factors
twof$Phi  ## inspect the factor correlations
```

```
##           MR1       MR2
## MR1 1.0000000 0.1888657
## MR2 0.1888657 1.0000000
```

How much variance is accounted for by the factors?

- We can also check how much variance overall is accounted for by the factors

```
##
## Loadings:
##        MR1    MR2
## item1           0.678
## item2           0.808
## item3           0.694
## item4           0.678
## item5           0.836
## item6   0.650
## item7   0.882
## item8   0.899
## item9   0.649
## item10  0.714
##
##                  MR1   MR2
## SS loadings    2.944 2.767
## Proportion Var 0.294 0.277
## Cumulative Var 0.294 0.571
```

Quiz question 2

- Quiz question:
  - *Which of these best describe principal axis factoring extraction?*
    - A PCA of a correlation matrix with communalities on the diagonals?
    - A PCA of a correlation matrix ignoring the diagonals?
    - A PCA of a correlation matrix where the the off-diagonals are replaced with communalities?
    - A PCA of a correlation matrix where all elements are replaced by communalities?

Checking the suitability of data for EFA

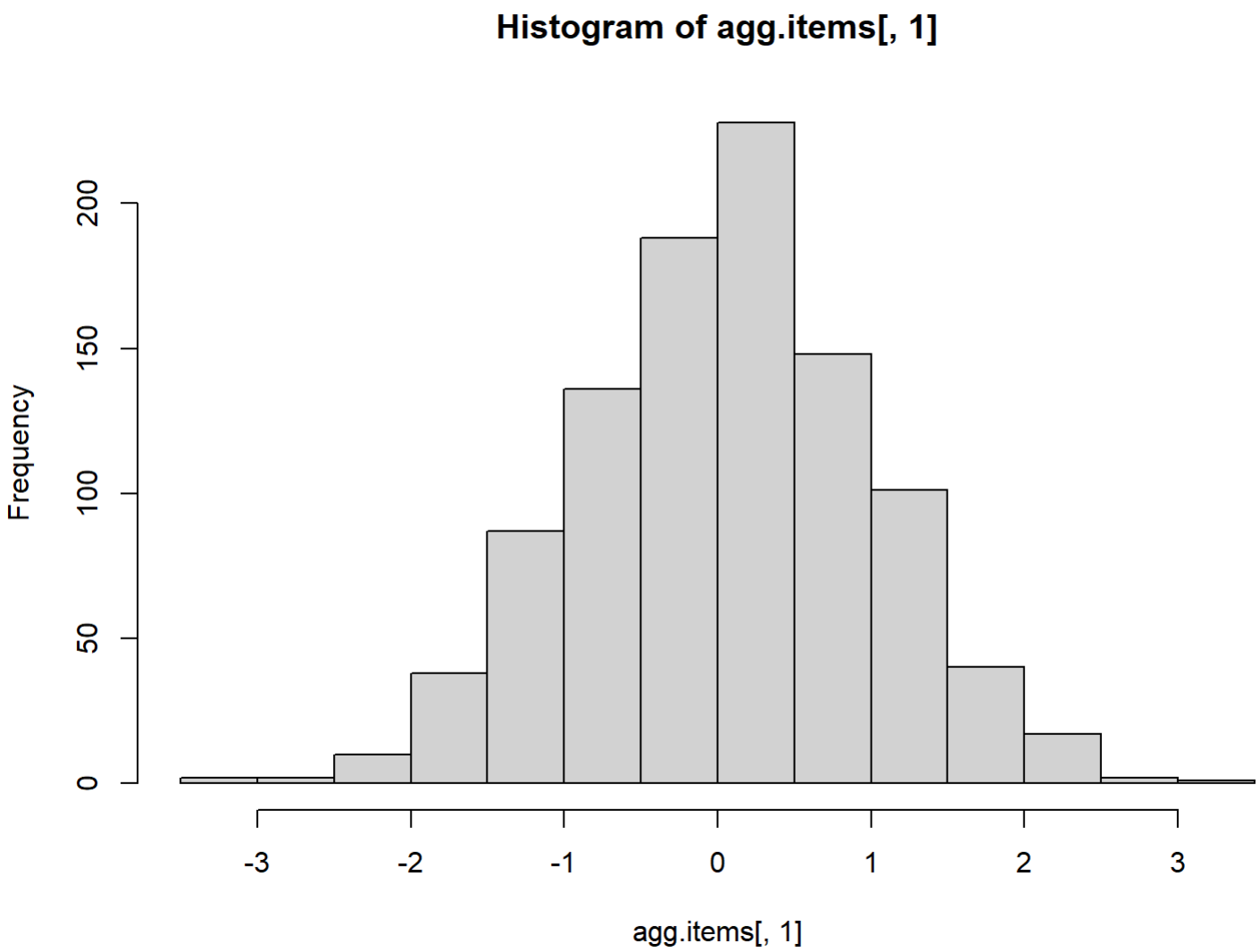
- The first step in an EFA is actually to check the appropriateness of the data:
  - *Does the data look multivariate normal?*
  - *Do the relations look linear?*
  - *Does the correlation matrix have good factorability?*

Multivariate normality

- Do the variables have (approximately) continuous measurement scales?
  - *5 or more response options*
- Examining univariate distributions using histograms

Univariate histogram example

```
hist(agg.items[ ,1])
```


Histogram of agg.items[, 1]
Frequency
0
50
100
150
200
-3
-2
-1
0
1
2
3
agg.items[, 1]

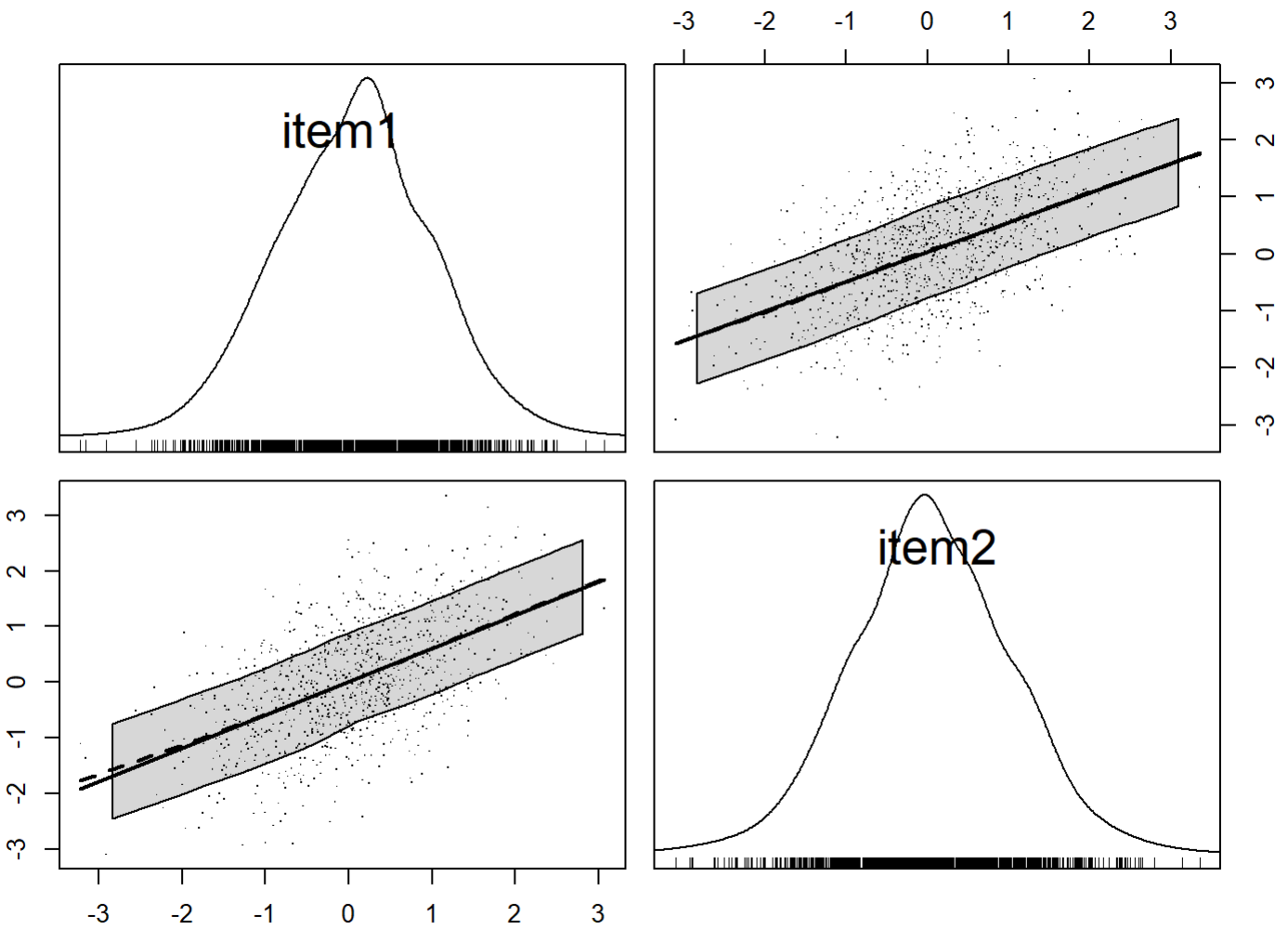
Linearity

- Plot linear and lowess lines for pairwise relations and compare

```
## Loading required package: carData
```

```
##
## Attaching package: 'car'
```

```
## The following object is masked from 'package:psych':
##
##     logit
```


item1
item2
-3
-2
-1
0
1
2
3

Factorability

- EFA focuses on variance **common** to items
  - *Not much point in an EFA if little variance in common*
- Use Kaiser-Meyer-Olkin (KMO) test
  - *Provides measure of proportion of variance shared between variables*
  - *Can be computed for individual variables or whole correlation matrix*
  - *Overall values >.60 and no variable <.50 is ideal*

KMO in R

```
KMO(agg.items)
```

```
## Kaiser-Meyer-Olkin factor adequacy
## Call: KMO(r = agg.items)
## Overall MSA =  0.87
## MSA for each item =
##  item1  item2  item3  item4  item5  item6  item7  item8  item9 item10
##   0.90   0.85   0.89   0.90   0.84   0.92   0.82   0.81   0.92   0.91
```

Reporting EFA

- Transparency and reproducibility
- Methods
  - *Methods to determine number of factors*
  - *Extraction method*
  - *Rotation method*
- Results
  - *Information about data suitability for EFA*
  - *Number of factors (and why)*
  - *Loading matrix*
  - *Factor correlations*
  - *Interpretation of factors (and why)*
  - *Variance explained by factors*

Summary

- Steps in EFA are similar to PCA but…
  - *The underlying theory and interpretation is quite different*
  - *Their results can differ if there is not a lot of common variance*
- EFA involves:
  - *Checking data suitability*
  - *Choosing number of factors*
  - *Factor extraction*
  - *Rotation*
  - *Interpretation of factors*

Live coding

- Each week I'll add some analysis based on what we cover
- This will be written up in a real peer-reviewed paper
- You can check the progress of the paper each week on the Learn weekly folder
- Simulated version of the dataset in weekly folders

Previous MSMR live coding publications

- SBQ ADHD:
  https://journals.sagepub.com/doi/full/10.1177/01650254241268865
- Ageism towards older adults in Colombia:
  https://osf.io/preprints/psyarxiv/fd7ac_v1

The live coding example this year

- WHO ageism towards older adults scale (WHO-TOPS) in Libya
- Developed to measure stereotypes, prejudice and discrimination against older adults
- Translated into Arabic
- Data collected from a sample of older adults in Libya
- This week: EFA of the data