Advanced Measurement Theory Course Notebook

William M. Murrah

2021-08-23

Contents

Introduction to the Course

Welcome! This is a notebook for ERMA 8350 Advanced Measurement Theory. The class will be using the textbook *Modern Psychometrics with R* (Mair, 2018), which will be the primary source for the course. I will use this notebook to make available additional information to help you learn the material. It may include some examples from the textbook, with some elaborations, additional readings, and some more details about implementing the methods in R. These web-based notes will make it easy for you to use code, by allowing you to copy and paste code found within. Some of you will have experience with R and others not. So I will try to also point you to additional resources that may be helpful. For example, in this preface I will provide links to resources to help you setup R and RStudio. RStudio is a platform to make using R more productive. I will use it extensively in this course.

Software

There are at least two way you can access the software needed for this course. You can use the virtual labs on campus. I know at least the education virtual labls have R and RStudio installed. IF you go this route you can watch the following video. Note you will need Duo setup for this to work.

Using Vlab to acces R/RStudio

A better option if you have a laptop, you can install both programs on your computer. They are both absolutely free and available on all major operating systems, so you will not have to worry about transferring information across computers, limited connection speeds, or other hassles inherent with the VLab route.

The following links take you to videos instructing you how to install them.

Installing R and RStudio

Organizing Projects in RStudio

Resources for Learning R

While such experience is certainly helpful, I do not assume you have prior knowledge of using R. I will demonstrate the use of R and provide (particularly in this notebook) the R code needed to use the methods we will learn. However, even if you have prior experience with R, you should plan to spend time learning to program in R. Some people find this intimidating initially, but most of you will grow to find R programming rewarding, and even fun by the end of the course. But, there will be frustration for sure.

Here are some good places to start learning R:

CRAN

R Packages

R is, among other paradigms, a functional programming language, which means is heavily utilizes functions. R's functions are stored in packages. While base R has a long list of very useful functions, to fully realize the power of R you will have to use additional packages. So, learning how to **install** packages (downloading from the web to your computer) and **loading** packages (making the package's functions accessible to your current R session) are important skills to master.

How To Use These Notes

Before going further, it may be helpful to watch the following video about how to use the code in this notebook with Rstudio:

How to Use RStudio with this Notebook

Chapter 1

Classic Test Theory

1.1 Classical True Score Model

The true score model is:

$$X = T + E$$

where X is the **observed score**, T is the **true score**, which is unknown, and E is the **error**

To demonstrate this let's assume we have the following data (R script is at end of this chapter),

```r
source("code/simulate_CTTdata.R")
CTTdata
```

```
   id time x1 x2 x3 x4 x5 x6 x7 x8 x9 x10 Tau
1   1    1  3  6  5  3  5  5  4  5  3   3   4
2   1    2  6  3  5  3  4  2  4  4  3   5   4
3   1    3  4  4  2  4  4  3  5  3  4   5   4
4   2    1  3  6  8  6  5  4  5  5  5   5   5
5   2    2  6  4  6  6  4  6  6  5  4   4   5
6   2    3  4  6  6  5  5  5  1  3  6   4   5
7   3    1  6  5  6  6  6  6  9  6  6   5   6
8   3    2  6  6  6  7  5  6  6  6  6   7   6
9   3    3  6  5  8  6  6  6  7  7  5   7   6
10  4    1  4  3  5  4  2  3  3  5  5   2   4
11  4    2  4  5  5  4  5  5  3  5  3   4   4
12  4    3  2  4  4  4  6  4  3  4  5   4   4
13  5    1  5  6  5  4  5  5  5  6  5   6   5
14  5    2  6  6  4  6  4  5  4  5  5   5   5
15  5    3  6  4  5  4  5  5  4  4  5   5   5
16  6    1  6  6  7  8  6  6  7  6  6   4   6
```

```
17  6     2  4  5  7  5  5  7  4  5  6   7   6
18  6     3  5  6  6  6  4  5  4  5  7   6   6
```

where `id` is a variable indicating individual test-takers, `time` indicated which of 3 times each individual was assessed, `x1 - x10` are the scores on 10 items that comprise the test, and `Tau` is the true value of the individuals ability. I use `Tau` here instead of `T`, because `T` is a protected symbol in R which is short-hand for `TRUE`. Note that we would not know `Tau` in most situations, but because this is simulated data we will pretend we do.

We can create a composite score for the ten items for each individual on each occasion by averaging columns 3 through 12.

```
CTTdata$X <- rowMeans(CTTdata[ ,3:12])
```

And we can also create `E`, the error with:

```
CTTdata$E <- CTTdata$X - CTTdata$Tau
```

Again, in practice we would not be able to directly compute `E` because we would not know `Tau`, but we will use it to build an understanding of what error is.

Now we have:

```
CTTdata
```

```
   id time x1 x2 x3 x4 x5 x6 x7 x8 x9 x10 Tau   X    E
1   1    1  3  6  5  3  5  5  4  5  3   3   4 4.2  0.2
2   1    2  6  3  5  3  4  2  4  4  3   5   4 3.9 -0.1
3   1    3  4  4  2  4  4  3  5  3  4   5   4 3.8 -0.2
4   2    1  3  6  8  6  5  4  5  5  5   5   5 5.2  0.2
5   2    2  6  4  6  6  4  6  6  5  4   4   5 5.1  0.1
6   2    3  4  6  6  5  5  5  1  3  6   4   5 4.5 -0.5
7   3    1  6  5  6  6  6  6  9  6  6   5   6 6.1  0.1
8   3    2  6  6  6  7  5  6  6  6  6   7   6 6.1  0.1
9   3    3  6  5  8  6  6  6  7  7  5   7   6 6.3  0.3
10  4    1  4  3  5  4  2  3  3  5  5   2   4 3.6 -0.4
11  4    2  4  5  5  4  5  5  3  5  3   4   4 4.3  0.3
12  4    3  2  4  4  4  6  4  3  4  5   4   4 4.0  0.0
13  5    1  5  6  5  4  5  5  5  6  5   6   5 5.2  0.2
14  5    2  6  6  4  6  4  5  4  5  5   5   5 5.0  0.0
15  5    3  6  4  5  4  5  5  4  4  5   5   5 4.7 -0.3
16  6    1  6  6  7  8  6  6  7  6  6   4   6 6.2  0.2
17  6    2  4  5  7  5  5  7  4  5  6   7   6 5.5 -0.5
18  6    3  5  6  6  6  4  5  4  5  7   6   6 5.4 -0.6
```

Look over the last three columns and make sure you understand their relation. For example, in the first row, note that `X` is .2 points above `Tau`, which is exactly the value of `E` we computed ($X_1 - T_1 = E_1 = 4.2 - 4 = .2$). The 1 subscript in the previous expression indicated row 1 (i.e. i = 1).

```
CTTdata$X_t <- round(ave(CTTdata$X, CTTdata$id, FUN = mean),1)
```

```
CTTdata
```

```
   id time x1 x2 x3 x4 x5 x6 x7 x8 x9 x10 Tau   X    E X_t
1   1    1  3  6  5  3  5  5  4  5  3   3   4 4.2  0.2 4.0
2   1    2  6  3  5  3  4  2  4  4  3   5   4 3.9 -0.1 4.0
3   1    3  4  4  2  4  4  3  5  3  4   5   4 3.8 -0.2 4.0
4   2    1  3  6  8  6  5  4  5  5  5   5   5 5.2  0.2 4.9
5   2    2  6  4  6  6  4  6  6  5  4   4   5 5.1  0.1 4.9
6   2    3  4  6  6  5  5  5  1  3  6   4   5 4.5 -0.5 4.9
7   3    1  6  5  6  6  6  6  9  6  6   5   6 6.1  0.1 6.2
8   3    2  6  6  6  7  5  6  6  6  6   7   6 6.1  0.1 6.2
9   3    3  6  5  8  6  6  6  7  7  5   7   6 6.3  0.3 6.2
10  4    1  4  3  5  4  2  3  3  5  5   2   4 3.6 -0.4 4.0
11  4    2  4  5  5  4  5  5  3  5  3   4   4 4.3  0.3 4.0
12  4    3  2  4  4  4  6  4  3  4  5   4   4 4.0  0.0 4.0
13  5    1  5  6  5  4  5  5  5  6  5   6   5 5.2  0.2 5.0
14  5    2  6  6  4  6  4  5  4  5  5   5   5 5.0  0.0 5.0
15  5    3  6  4  5  4  5  5  4  4  5   5   5 4.7 -0.3 5.0
16  6    1  6  6  7  8  6  6  7  6  6   4   6 6.2  0.2 5.7
17  6    2  4  5  7  5  5  7  4  5  6   7   6 5.5 -0.5 5.7
18  6    3  5  6  6  6  4  5  4  5  7   6   6 5.4 -0.6 5.7
```

1.2 Reliability

$$\text{reliability} = \frac{\sigma_T^2}{\sigma_X^2} = \frac{\sigma_T^2}{\sigma_T^2 + \sigma_E^2} = \rho_{XT}^2$$

```
Tau <- CTTdata$Tau
X <- CTTdata$X
E <- CTTdata$X - CTTdata$Tau
```

```
var(Tau)/var(X)
```

```
[1] 0.9170806
```

```
var(Tau)/(var(Tau) + var(E))
```

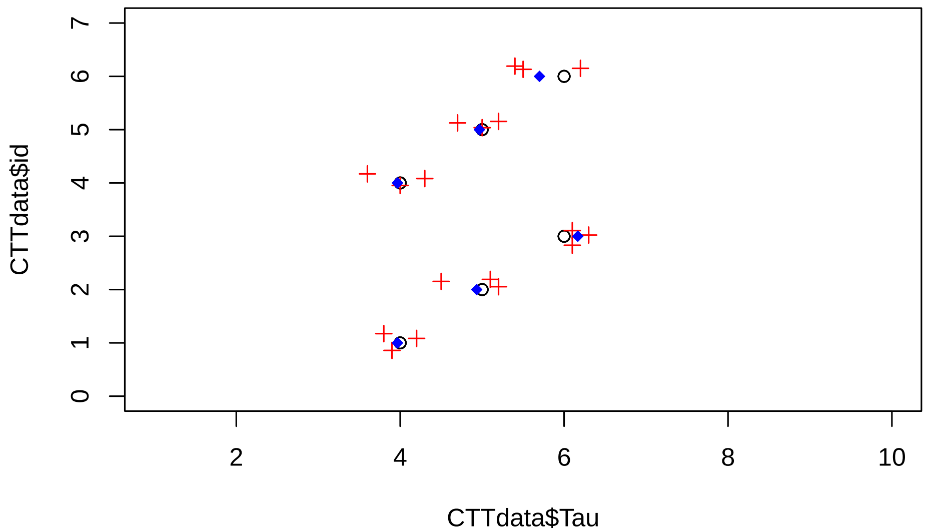
```
[1] 0.8898776
```

```
cor(Tau, X)^2
```

```
[1] 0.886766
```

```
plot(x = CTTdata$Tau, y = CTTdata$id, xlim = c(1,10),
     ylim = c(0,7))
points(x = CTTdata$X, y = jitter(CTTdata$id), pch = 3, col = "red")
```

```
points(x = ave(x = CTTdata$X, factor(CTTdata$id), FUN = mean), y = CTTdata$id,
       col = "blue", pch = 18)
```




```
# points(x = CTTdata$X_t, pch = 2, factor(CTTdata$id))
```

1.2.1 Cronbach's α

In the notes for this chapter, I demonstrate aspects of classical test theory, reliability and generalizability theory using data from a study exploring the motivation of R package authors (Mair et al., 2015). This tutorial is based on Chapter 1 of Mair (2018), which can be consulted for a more in depth exposition of the underlying theory. Here I focus on demonstrating some of those concepts in R, as well as describing how to get certain results in R.

First, I load the packages used in this tutorial:

```
# Packages used:
library(psych)
library(MPsychoR)
```

Next, I load the full data set from the `MPsychoR` package (Mair, 2020), then as in the chapter, I subset the data to only include hybrid motivation items, followed by removing rows with missing values.

```
data("Rmotivation")

# Create data frame with only Hybrid Motivation items.
HybMot <- subset(Rmotivation,
                        select = grep("hyb", names(Rmotivation)))
# Remove rows  with any missing data.
```

```
HybMot <- na.omit(HybMot)
```

This leads to a data set with 777 authors and 19 items.

```
# How many authors(rows) and items(columns)?
dim(HybMot)
```

```
[1] 777  19
```

```
# Note they are all dichotomous items.
head(HybMot)
```

```
  hyb1 hyb2 hyb3 hyb4 hyb5 hyb6 hyb7 hyb8 hyb9 hyb10 hyb11 hyb12 hyb13 hyb14
1    1    0    1    0    1    1    0    0    0     1     1     1     1     0
3    0    0    1    0    1    0    0    0    0     1     0     1     1     0
4    1    1    1    1    1    0    1    0    0     1     0     1     1     0
5    1    0    0    1    1    0    0    0    0     1     1     1     1     0
8    1    1    1    1    1    1    1    1    1     1     0     1     1     1
9    1    0    0    1    1    0    0    0    0     1     0     0     1     0
  hyb15 hyb16 hyb17 hyb18 hyb19
1     1     1     1     1     1
3     1     0     0     1     0
4     0     1     1     1     1
5     1     0     1     1     1
8     1     1     1     1     1
9     0     1     1     1     1
```

```
# Variance/Covariance Matrix
vcmat <- cov(HybMot)
scroll_box(kable(vcmat, digits = 2), width = "100%")
```

	hyb1	hyb2	hyb3	hyb4	hyb5	hyb6	hyb7	hyb8	hyb9	hyb10	hyb11	hyb12
hyb1	0.18	0.06	0.04	0.03	0.03	0.05	0.01	0.05	0.04	0.04	0.03	0.03
hyb2	0.06	0.25	0.06	0.05	0.03	0.05	-0.01	0.04	0.05	0.02	0.04	0.04
hyb3	0.04	0.06	0.23	0.13	0.03	0.05	0.00	0.03	0.05	0.04	0.05	0.06
hyb4	0.03	0.05	0.13	0.21	0.03	0.04	0.01	0.03	0.04	0.03	0.05	0.05
hyb5	0.03	0.03	0.03	0.03	0.11	0.02	0.00	0.01	0.01	0.03	0.03	0.03
hyb6	0.05	0.05	0.05	0.04	0.02	0.24	0.01	0.11	0.15	0.05	0.06	0.06
hyb7	0.01	-0.01	0.00	0.01	0.00	0.01	0.22	0.04	0.01	0.03	0.00	0.02
hyb8	0.05	0.04	0.03	0.03	0.01	0.11	0.04	0.25	0.10	0.06	0.05	0.06
hyb9	0.04	0.05	0.05	0.04	0.01	0.15	0.01	0.10	0.20	0.04	0.04	0.05
hyb10	0.04	0.02	0.04	0.03	0.03	0.05	0.03	0.06	0.04	0.15	0.03	0.06
hyb11	0.03	0.04	0.05	0.05	0.03	0.06	0.00	0.05	0.04	0.03	0.23	0.10
hyb12	0.03	0.04	0.06	0.05	0.03	0.06	0.02	0.06	0.05	0.06	0.10	0.23
hyb13	0.03	0.03	0.02	0.02	0.03	0.02	0.00	0.02	0.01	0.03	0.03	0.04
hyb14	0.03	0.03	0.02	0.02	0.02	0.07	0.00	0.04	0.05	0.02	0.04	0.03
hyb15	0.04	0.03	0.06	0.04	0.04	0.06	0.01	0.06	0.04	0.04	0.10	0.11
hyb16	0.05	0.03	0.02	0.02	0.02	0.05	0.02	0.05	0.04	0.04	0.03	0.04
hyb17	0.04	0.01	0.03	0.03	0.02	0.05	0.02	0.05	0.03	0.04	0.03	0.03
hyb18	0.03	0.00	0.02	0.02	0.01	0.02	0.01	0.03	0.01	0.03	0.01	0.02
hyb19	0.06	0.03	0.04	0.04	0.03	0.07	0.02	0.06	0.05	0.05	0.02	0.04

```r
k <- ncol(HybMot)
sigma2_Xi <- tr(vcmat) # trace of matrix or sum(diag(vmat))
sigma2_X <- sum(vcmat)
```

1.2.2 Other Reliability Coefficients

1.3 Generalizability Theory

Generalizability theory, or G-theory for short, is an extension of CTT, which decomposes the one error term in CTT into multiple sources of error called *facets*. These could include sources such as items, raters, or measurement occasions. These were each given a subscript on page 2 of the text.

Before looking at these different sources of error, let's calculate Cronbach's α in a different way, that will allow this decomposition going forward.

We will first need to reshape the data from wide to long format. A great tutorial on reshaping data with the `reshape2` package can be found here:

https://seananderson.ca/2013/10/19/reshape/

Basically, we need to transform the data so that instead of each item being in a separate column are reshaped so there is one column with the cell values, and one column that identifies which item the score is from.

```r
library("reshape2")
# Add person variable
```

```
Hyb1 <- data.frame(HybMot, person = 1:nrow(HybMot))
Hyblong <- melt(Hyb1, id.vars = c("person"), variable.name = "item")
Hyblong$person <- as.factor(Hyblong$person)
```

1.3.1 Reliability and Generalizability

```
summary(aov(value ~ person + item, data = Hyblong))
```

```
               Df Sum Sq Mean Sq F value Pr(>F)
person        776  663.0    0.85   5.549 <2e-16 ***
item           18  573.8   31.88 207.048 <2e-16 ***
Residuals   13968 2150.5    0.15
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

From this output we can calculate Cronbach's α with the following values:

```
MSp <- 0.85
MSe <- 0.15

alpha <- (MSp - MSe)/MSp
print(alpha, digits = 2)
```

```
[1] 0.82
```

```
ICC(HybMot)
```

```
Call: ICC(x = HybMot)

Intraclass correlation coefficients
                         type  ICC   F df1   df2        p lower bound
Single_raters_absolute   ICC1 0.15 4.4 776 13986 7.2e-279        0.14
Single_random_raters     ICC2 0.16 5.5 776 13968  0.0e+00        0.14
Single_fixed_raters      ICC3 0.19 5.5 776 13968  0.0e+00        0.18
Average_raters_absolute ICC1k 0.77 4.4 776 13986 7.2e-279        0.75
Average_random_raters   ICC2k 0.78 5.5 776 13968  0.0e+00        0.75
Average_fixed_raters    ICC3k 0.82 5.5 776 13968  0.0e+00        0.80
                        upper bound
Single_raters_absolute         0.17
Single_random_raters           0.18
Single_fixed_raters            0.21
Average_raters_absolute        0.79
Average_random_raters          0.81
Average_fixed_raters           0.83

 Number of subjects = 777     Number of Judges =  19
```

```
icchyb <- ICC(HybMot)
sqrt((0.85-0.15)/19)
```

```
[1] 0.191943
```

```
sqrt((31.88-0.15)/777)
```

```
[1] 0.2020806
```

```
library("lme4")
```

```
Loading required package: Matrix
```

```
VarCorr(lmer(value ~ (1|person) + (1|item), data = Hyblong))
```

```
 Groups   Name        Std.Dev.
 person   (Intercept) 0.19200
 item     (Intercept) 0.20206
 Residual             0.39238
```

```
library("gtheory")
gfit <- gstudy(data = Hyblong, formula = value ~ (1|person) + (1|item))
dfit <- dstudy(gfit, colname.objects = "person", colname.scores = "value",
               data = Hyblong)
round(dfit$generalizability, 3)
```

```
[1] 0.82
```

1.3.2 Multiple Sources of Error

Generalizability theory acknowledges that multiple sources of error can impact scores simultaneously, and allow for estimating the effects of each (Raykov and Marcoulides, 2011). These various sources of error, or facets (e.g., items, raters, measurement occasions). All measurements of behavior are conceptualized as being sampled from a *universe* of admissible observations (Raykov and Marcoulides, 2011). If the observed score is expected to vary across a facet (e.g. vary across occasions, or vary depending on the items included, or the rater scoring), then that facet is a defining characteristic of the universe. The idea of reliability is replace with the idea of generalizability, which ,instead of asking how accurately observed scores can reflect the true score (CTT), generalizability theory asks how accurately observed scores allow us to generalize about behavior of an individual in a particular universe.

Below is the code from the Mair (2018) text.

```
library(gtheory)

data("Lakes")
phydat <- subset(Lakes, subtest == "physical")
```

```r
phydat$item <- droplevels(phydat$item)
head(phydat)
```

```
      personID raterID item score  subtest
12611        1       7 phy1     5 physical
12612        1       1 phy1     5 physical
12613        1       3 phy1     5 physical
12614        1       8 phy1     4 physical
12615        1       5 phy1     6 physical
12616        2       3 phy1     5 physical
```

```r
formula <- score ~ (1|personID) + (1|raterID) + (1|item) +
  (1|personID:raterID) + (1|personID:item) + (1|raterID:item)
gfit <- gstudy(formula = formula, data = phydat)
gfit
```

```
$components
            source         var percent n
1 personID:raterID 0.127298630    10.7 1
2    personID:item 0.151229568    12.7 1
3         personID 0.458766113    38.5 1
4     raterID:item 0.008802081     0.7 1
5          raterID 0.139048763    11.7 1
6             item 0.033388424     2.8 1
7         Residual 0.272703419    22.9 1

attr(,"class")
[1] "gstudy" "list"
```

```r
dfit <- dstudy(gfit, colname.objects = "personID", colname.scores = "score",
               data = phydat)
dfit$components
```

```
            source          var percent  n
1 personID:raterID 0.0254597259     4.3  5
2    personID:item 0.0504098560     8.5  3
3         personID 0.4587661131    77.4  1
4     raterID:item 0.0005868054     0.1 15
5          raterID 0.0278097526     4.7  5
6             item 0.0111294746     1.9  3
7         Residual 0.0181802279     3.1 15
```

```r
dfit$var.error.abs
```

```
[1] 0.1335758
```

```r
dfit$sem.abs
```

```
[1] 0.3654803
```

```
dfit$var.error.rel
```

```
[1] 0.09404981
```

```
dfit$sem.rel
```

```
[1] 0.3066754
```

```
dfit$dependability
```

```
[1] 0.7744954
```

```
dfit$generalizability
```

```
[1] 0.8298714
```

1.4 Additional Readings

For more information of G theory, see Raykov and Marcoulides (2011). For an example using the R package `lavaan` with G theory, see Jorgensen (2021).

1.5 R Scripts

1.5.1 Simulating CTT data

```
#***************************************************************************
# Title: simulate_CTTdata
# Author: William Murrah
# Description: Simulate data to demonstrate CTT and reliability
# Created: Monday, 09 August 2021
# R version: R version 4.1.0 (2021-05-18)
# Project(working) directory: /Users/wmm0017/Projects/Courses/AdvancedMeasurementTheor
#***************************************************************************

simx <- function(truescore, sigmax = 1) {
  x <- rnorm(18, truescore, sigmax)
  return(round(x))
}
id <- rep(1:6, each = 3)
Tau <- rep(rep(4:6, each = 3),2)
set.seed(20210805)
CTTdata <- data.frame(
  id = id,
  time = rep(1:3, 6),
  x1 = simx(Tau),
```

```
  x2 = simx(Tau),
  x3 = simx(Tau),
  x4 = simx(Tau),
  x5 = simx(Tau),
  x6 = simx(Tau),
  x7 = simx(Tau),
  x8 = simx(Tau),
  x9 = simx(Tau),
  x10 = simx(Tau),
  Tau = Tau
)
rm(id, Tau, simx)
```

Chapter 2

Factor Analysis

2.1 Correlation Coefficient

Pearson product-moment correlation:

$$r_{xy} = \frac{\Sigma_{n=1}^{n}(x_k - \bar{x})(y_i - \bar{y})}{\sqrt{\Sigma_{n=1}^{n}(x_i - \bar{x})^2}\sqrt{\Sigma_{n=1}^{n}(y_i - \bar{y})^2}} = \frac{s_{xy}}{s_x s_y}.$$

The eauation looks very daunting, until you see that it is just the covariance of x and y divided by the product of their standard deviations.

```
library("MPsychoR")
data("YouthDep")
item1 <- YouthDep[, 1]
levels(item1) <- c("0", "1", "1")
item2 <- YouthDep[, 14]
levels(item2) <- c("0", "1", "1")
table(item1, item2)
```

```
     item2
item1    0    1
    0 1353  656
    1  115  166
```

```
## ------ correlation coefficients
library("psych")
tetcor <- tetrachoric(cbind(item1, item2))
```

```
tetcor
```

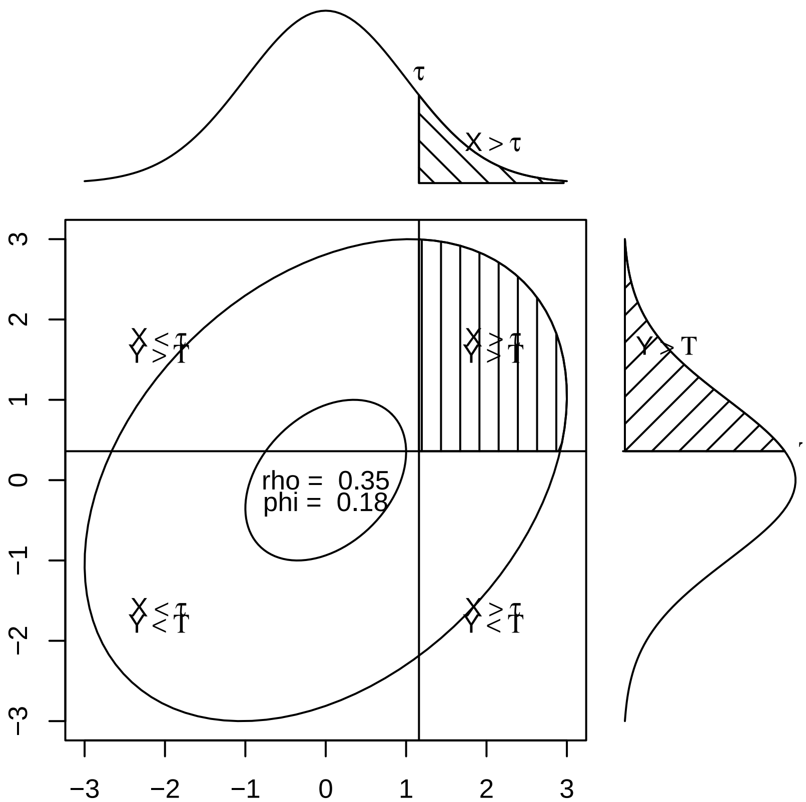
```
Call: tetrachoric(x = cbind(item1, item2))
tetrachoric correlation
```

```
      item1 item2
item1 1.00
item2 0.35  1.00

 with tau of
item1 item2
 1.16  0.36
```

```
item1 <- YouthDep[, 1]
item2 <- YouthDep[, 14]
polcor <- polychoric(cbind(item1, item2))
polcor
```

```
Call: polychoric(x = cbind(item1, item2))
Polychoric correlations
      item1 item2
item1 1.00
item2 0.33  1.00

 with tau of
         1   2
item1 1.16 2.3
item2 0.36 1.2
```

```
draw.tetra(r = .35, t1 = 1.16, t2 = .36)
```




```
DepItems <- YouthDep[,1:26]
Depnum <- data.matrix(DepItems) - 1  ## convert to numeric
Rdep <- polychoric(Depnum)
```

2.2 Think about these situations

What do you do when you have a large number of variables you are considering as predictors of a dependent variable?

- Often, subsets of these variables are measuring the same, or very similar things.
- We might like to reduce the variables to a smaller number of predictors.

What if you are developing a measurement scale and have a large number of items you think measure the same construct

- You might want to see how strongly the items are related to the construct.

2.3 Solutions

1. Principal Components Analysis
   - transforming the original variables into a new set of linear combinations (pricipal components).
2. Factor Analysis
   - setting up a mathematical model to estimate the number or factors

2.4 Principal Components Analysis

- Concerned with explaining variance-covariance structure of a set of variables.
- PCA attempts to explain as much of the total variance among the observed variables as possible with a smaller number of components.
- Because the variables are standardized prior to analysis, the total amount of variance available is the number of variables.
- The goal is **data reduction** for subsequent analysis.
- Variables *cause* components.
- Components are not representative of any underlying theory.

2.5 Factor Analysis

- The goal is understanding underlying constructs.
- Uses a modified correlation matrix (reduced matrix)
- factors *cause* the variables.
- Factors represent theoretical constructs.
- Focuses on the common variance of the variables, and purges the unique variance.

2.6 Components

The principal components partition the total variance (the sum of the variances of the original variables) by finding the linear combination of the variables that account for the maximum amount of variance:

$$PC1 = a_{11}x_1 + a_{12}x_2 ... a_{1p}x_p,$$

This is repeated as many time as there are variables.

2.7 PC Extraction

draw pretty pictures on the board

2.8 Eigenvalues

Eigenvalues represent the variance in the variables explained by the success components.

2.9 Determining the Number of Factors

1. Kaiser criterion: Retain only factors with eigenvalues > 1. (generally accurate)
2. Scree plot: plot eigenvalues and drop factors after leveling off.
3. Parallel analysis: compare observed eigenvalues to parallel set of data from randomly generated data. Retain factors in original if eigenvalue is greater than random eigenvalue.
4. Factor meaningfulness is also very important to consider.

2.10 Example data

```
lower <- "
1.00
0.70 1.00
0.65 0.66 1.00
0.62 0.63 0.60 1.00
"
cormat <- getCov(lower, names = c("d1", "d2", "d3", "d4"))

cormat
```

```
     d1   d2   d3   d4
d1 1.00 0.70 0.65 0.62
d2 0.70 1.00 0.66 0.63
d3 0.65 0.66 1.00 0.60
d4 0.62 0.63 0.60 1.00
```

2.11 Kaiser

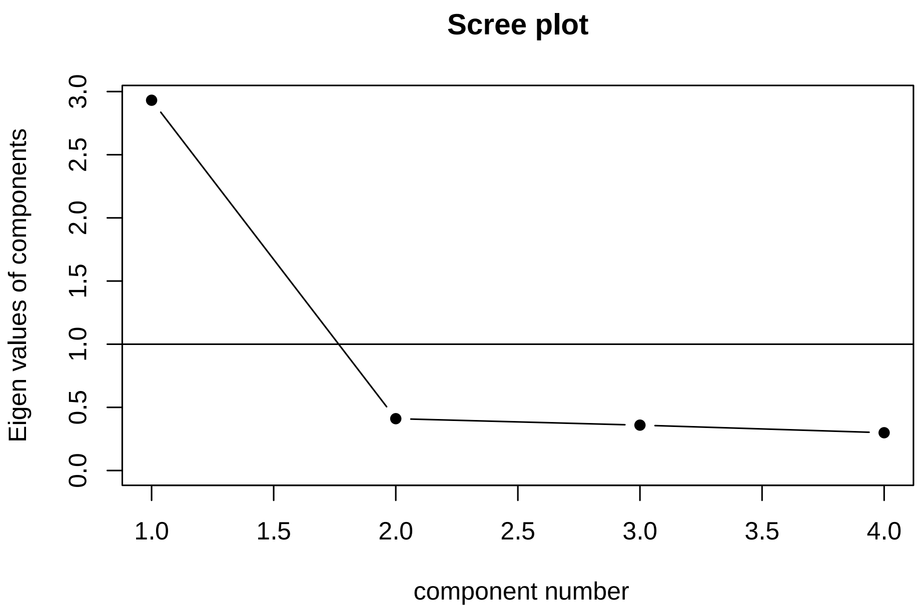
Retain factors with eigenvalues greater than 1

```
eigen(cormat)$values
```

```
[1] 2.9311792 0.4103921 0.3592372 0.2991916
```

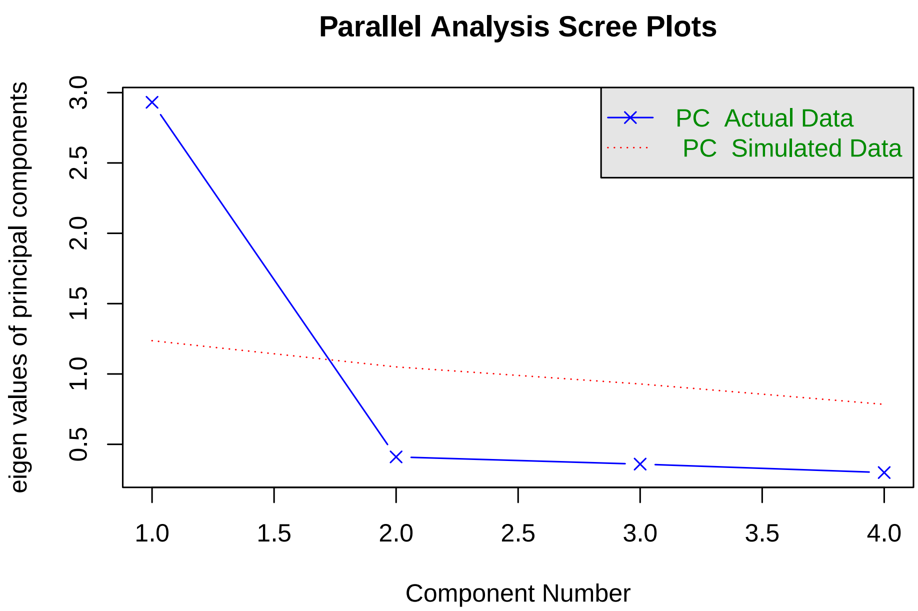
```
scree(cormat, factors = FALSE)
```


2.12 Scree Plot




```
fa.parallel(cormat, fa = "pc")
```

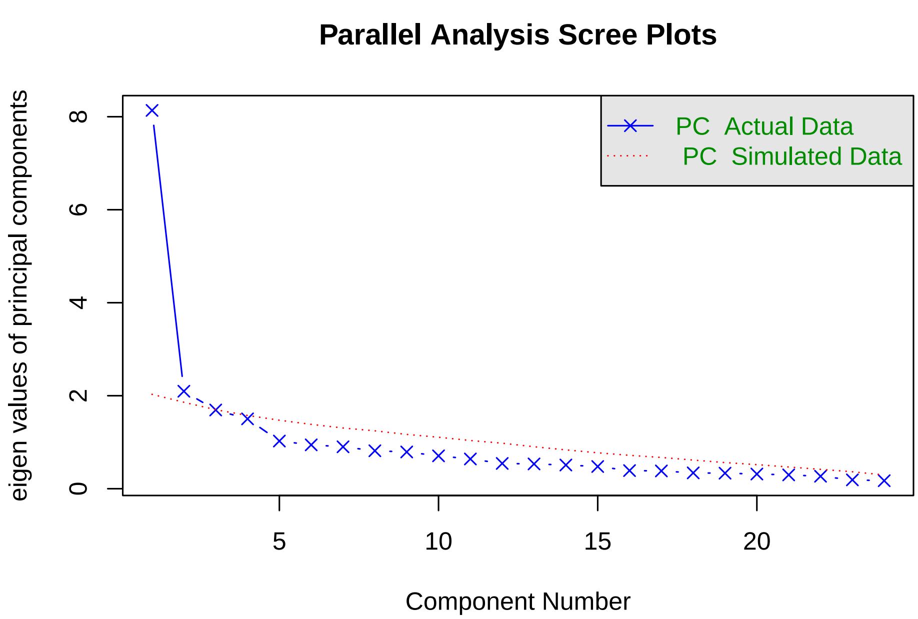
2.13 Horn's Parallel Analysis




```
Parallel analysis suggests that the number of factors =  NA  and the number of components =  1
```

2.14 Another example

```
fa.parallel(Harman74.cor$cov, fa = "pc")
```



```
Parallel analysis suggests that the number of factors =  NA  and the number of componen
```

2.15 Rotation

- Principal components are derived to maximize the variance accounted for (data reduction).
- Rotation is done to make the factors more interpretable (i.e. meaningful).
- Two major classes of rotation:
  - Orthogonal - new factors are still uncorrelated, as were the initial factors.
  - Oblique - new factors are allowed to be correlated.

Essentially reallocates the loadings. The first factor may not be the one accounting for the most variance.

2.16 Orthogonal Rotation

1. **Quartimax** - idea is to clean up the *variables*. Rotation done so each variable loads mainly on one factor. Problematic if there is a general factor on which most or all variables load on (think IQ).

2. **Varimax** - to clean up *factors*. So each factor has high correlation with a smaller number of variables, low correlation with the other variables. Generally makes interpretation easier.

2.17 Oblique Rotation

- Often correlated factors are more reasonable.
- Therefore, oblique rotation is often preferred.
- But interpretation is more complicated.

2.18 Factor Matrices

1. Factor pattern matrix:
   - includes *pattern coefficients* analogous to standardized partial regression coefficients.
   - Indicated the unique importance of a factor to a variable, holding other factors constant.
2. Factor structure matrix:
   - includes *structure coefficients* which are simple correlations of the variables with the factors.

2.19 Which matrix should we interpret?

- When orthogonal rotation is used interpret *structural coefficients* (but they are the same as pattern coefficients).
- When oblique rotation is used pattern coefficients are preferred because they account for the correlation between the factors and they are parameters of the correlated factor model (which we will discuss next class).

2.20 Which variables should be used to interpret each factor?

- The idea is to use only those variables that have a strong association with the factor.
- Typical thresholds are $|.30|$ or $|.40|$.
- Content knowledge is critical.

2.21 Examples

Let's look at some examples

2.22 Steps in Factor Analysis

1. Choose extraction method
    - So far we've focused on PCA
2. Determine the number of components/factors
    - Kaiser method: eigenvalues > 1
    - Scree plot: All components before leveling off
    - Horn's parallel analysis: components/factors greater than simulated values from random numbers
3. Rotate Factors
    - Orthogonal
    - Oblique
4. Interpret Components/Factors

2.23 Tom Swift's Eletric Factor Analysis Factory

"Little Jiffy" method of factor analysis

1. Extraction method : PCA
2. Number of factors: eigenvalues > 1
3. Rotation: orthogonal(varimax)

Dimension	Derivation
Thickness	x
Width	y
Length	z
Volume	xyz
Density	d
Weight	$xyzd$
Total surface area	$2(xy + xz +_y z)$
Cross-sectional area	yz
Total edge length	$4(x + y + z)$
Internal diagonal length	$(x^2 + y^2 + z^2)^2$
Cost per pound	c

Table 2.1: Correlations between dimensions

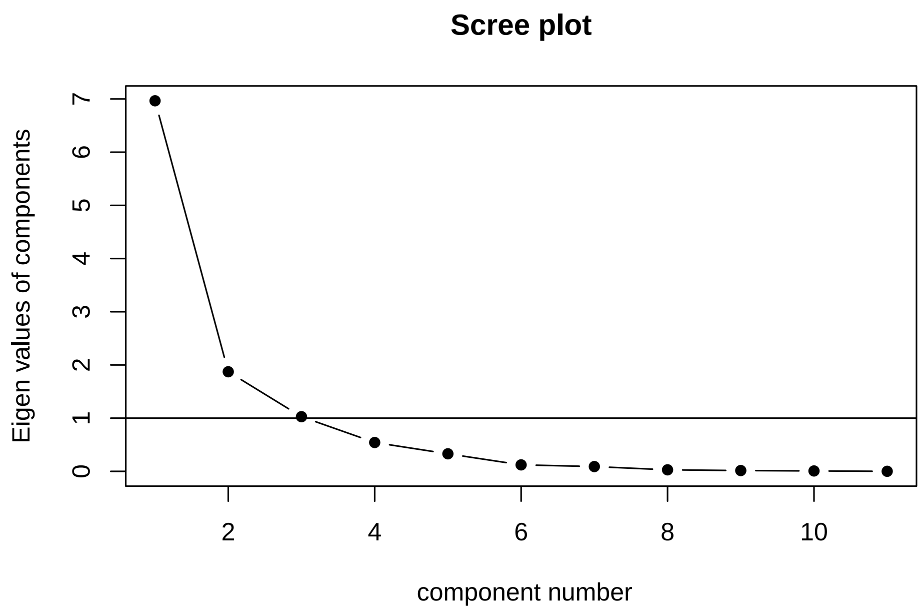
	thick	width	length	volume	density	weight	surface	crosssec	edge	diagonal
thick	1.00									
width	0.49	1.00								
length	0.24	0.61	1.00							
volume	0.84	0.77	0.58	1.00						
density	-0.13	-0.15	-0.02	-0.22	1.00					
weight	0.59	0.55	0.45	0.65	0.44	1.00				
surface	0.74	0.87	0.72	0.97	-0.2	0.65	1.00			
crosssec	0.46	0.92	0.83	0.81	-0.15	0.56	0.92	1.00		
edge	0.61	0.88	0.84	0.87	-0.15	0.61	0.97	0.96	1.00	
diagonal	0.51	0.78	0.86	0.85	-0.18	0.57	0.91	0.93	0.92	1.00
cost	-0.02	0.03	-0.02	-0.11	0.62	0.24	-0.07	-0.03	-0.04	-0.12

4. Interpretation

2.24 Metal Boxes

2.25 Correlations

2.26 Eigenvalues > 1



2.27 Orthogonal Rotation

```
Loadings:
         RC1    RC3    RC2
thick     0.212  0.947 -0.053
width     0.801  0.382 -0.011
length    0.936 -0.006  0.054
volume    0.634  0.744 -0.122
density  -0.102 -0.060  0.930
weight    0.440  0.610  0.509
surface   0.792  0.600 -0.078
crosssec  0.942  0.287 -0.031
edge      0.892  0.422 -0.031
diagonal  0.905  0.327 -0.088
cost     -0.023 -0.012  0.841

                 RC1   RC3   RC2
SS loadings    5.298 2.699 1.868
Proportion Var 0.482 0.245 0.170
Cumulative Var 0.482 0.727 0.897
```

2.28 Orthogonal Rotation with Loadings > .70

```
Loadings:
```

```
         RC1    RC3    RC2
thick           0.947
width     0.801
length    0.936
volume          0.744
density                0.930
weight
surface   0.792
crosssec  0.942
edge      0.892
diagonal  0.905
cost                   0.841

                 RC1   RC3   RC2
SS loadings    5.298 2.699 1.868
Proportion Var 0.482 0.245 0.170
Cumulative Var 0.482 0.727 0.897
```

2.29 R Code for Chapter 2

Chapter 3

Path Analysis and Structural Equation Modeling

We describe our methods in this chapter.

Chapter 4

Item Response Theory

Some *significant* applications are demonstrated in this chapter.

4.1 Example one

4.2 Example two

Chapter 5

Principal Components Analysis

We have finished a nice book.

Chapter 6

Correspondence Analysis

Chapter 7

Gifi Methods

Chapter 8

Multidimensional Scaling

Chapter 9

Graphing Multidimensional Data

Chapter 10

Networks

Chapter 11

Modeling Trajectories and Time Series

Bibliography

Jorgensen, T. D. (2021). How to estimate absolute-error components in structural equation models of generalizability theory. *Psych*, 3(2):113–133.

Mair, P. (2018). *Modern psychometrics with R.* Springer.

Mair, P. (2020). *MPsychoR: Modern Psychometrics with R.* R package version 0.10-8.

Mair, P., Hofmann, E., Gruber, K., Hatzinger, R., Zeileis, A., and Hornik, K. (2015). Motivation, values, and work design as drivers of participation in the r open source project for statistical computing. *Proceedings of the National Academy of Sciences*, 112(48):14788–14792.

Raykov, T. and Marcoulides, G. A. (2011). *Introduction to psychometric theory.* Routledge.