Contents

Preface ........................................................................................................................................ 3

1 Introduction and R Setup ......................................................................................................... 4
   1.1 Caveat on this document ........................................................................................................ 5
   1.2 Resources ............................................................................................................................. 5

2 Prepare and Describe the Data ................................................................................................. 6
   2.1 The Data Set ........................................................................................................................ 6
   2.2 Numeric and Graphical Description of the Data ..................................................................... 7
   2.3 Bivariate Characteristics of the data set ............................................................................... 12
   2.4 Covariances and Zero Order Correlations ......................................................................... 13

3 Using the lavaan package for CFA .......................................................................................... 15
   3.1 Implement the CFA, First Model ......................................................................................... 15
   3.2 Generate a second model and compare ............................................................................. 21
   3.3 Compare Model 1 and Model 2 ......................................................................................... 26
   3.4 An additional perspective on estimation and optimization ................................................. 26

4 Using the sem package for CFA ............................................................................................... 30
   4.1 Example one ....................................................................................................................... 30
   4.2 Example two ....................................................................................................................... 36
   4.3 Compare the two CFA models produced by sem ................................................................. 40

5 Using the OpenMx Package for CFA ...................................................................................... 41
   5.1 First OpenMx Model - Single Factor .................................................................................. 41
   5.2 Second OpenMx Model - the bifactor model ..................................................................... 44
   5.3 Third OpenMx Model .......................................................................................................... 49
   5.4 Compare the OpenMx models ............................................................................................. 53

6 Summary and Reproducibility .................................................................................................. 54
Preface

The data set and the models evaluated are those used by James Boswell in his APSY613 Multivariate Analysis class in the Psychology Department at the University at Albany. The data set is the WISC-R data set that the multivariate statistics textbook by the Tabachnick textbook (Tabachnick et al., 2019) employs for confirmatory factor analysis illustration. The goal of this document is to outline rudiments of Confirmatory Factor Analysis strategies implemented with three different packages in R. The illustrations here attempt to match the approach taken by Boswell with SAS. The document is targeted to UAlbany graduate students who have already had instruction in R in their introductory statistics courses.

This book/monograph uses the bookdown package (Xie, 2018a) for R (R Core Team, 2018), which was built on top of rmarkdown (Allaire et al., 2018) and knitr (Xie, 2015). RStudio (RStudio Team, 2015) was used for all writing and programming.
Chapter 1

Introduction and R Setup

This short monograph outlines three approaches to implementing Confirmatory Factor Analysis with R, by using three separate packages. The illustration is simple, employing a 175 case data set of scores on subsections of the WISC. The idea is to fit a bifactor model where the two latent factors are the verbal and performance constructs. In this primary two-factor model, each observed variable is associated with only one latent factor. Then a second model is fit. It includes a path from both latent factors to one of the variables. Comparisons of models are then performed.

Several R packages are required for the implementations outlined in the succeeding chapters. Since CFA is implemented as a structural equation model, commercial software (e.g., LISREL, EQS, SAS) as well as open-source approaches to CFA all use SEM routines. The three primary R packages to illustrate CFA are lavaan, sem and OpenMx, along with the drawing package, semPlot. One major advantage of using R for implementation of these methods is that semPlot provides a user-friendly method for producing path diagrams of many styles by simply taking a model object from the CFA fitting functions of the other packages.

Other “housekeeping” packages are loaded here, but the three analytical packages for CFA are loaded at the point in the sequence of their usage since some common function names are shared - thus load order is important.

```r
library(car)
library(semPlot)
library(psych)
library(knitr)
library(kableExtra)
library(MVN)
library(dplyr)
library(magrittr)
library(tidyr)
library(corrplot)
library(ggraph)
```

Package citations for packages loaded here (in the above order): car (Fox et al., 2018), semPlot (Epskamp and with contributions from Simon Stuber, 2017), psych (Revelle, 2019), knitr (Xie, 2018b), kableExtra (Zhu, 2019), MVN (Korkmaz et al., 2018), dplyr (Wickham et al., 2018), magrittr (Bache and Wickham, 2014), tidyr (Wickham and Henry, 2018), corrplot (Wei and Simko, 2017)

Package citations for packages loaded elsewhere in this document: bookdown (Xie, 2018a), rmarkdown (Allaire et al., 2018), sem (Fox et al., 2017), lavaan (Rosseel, 2018), OpenMx (Boker et al., 2019)
1.1 Caveat on this document

The present treatment of the CFA procedures is not intended to be an exhaustive analysis of this particular data set. Nor is it intended to be a thorough treatment of the CFA approaches available in R, CFA in general, or SEM in general. Rather, it is intended as a bit more than a simple introduction to CFA using R (and by implication, the nice capabilities of for Structural Equation Modeling). It provides students, who have a basic understanding of how to use R, with a reasonable introduction to CFA modeling code. The R approaches can then be compared to their class coverage of the same analysis, done with SAS. This document provides some capabilities that may not have been covered in class, and it misses others. The learning curve for software is never at an asymptote......

1.2 Resources

The following list will provide a good start for those needing a broader in CFA modeling and more detailed sources for the primary packages employed in this document.

- A comprehensive textbook treatment of SEM and CFA: (Tabachnick et al., 2019)
- Tim Brown’s well-regarded book on CFA: (Brown, 2015)
- Rosseel’s extensive original article on lavaan: (Rosseel, 2012)
- El-Sheik, et al on a comparison of software for SEM: (El-Sheikh et al., 2017)
- Narayanan’s review of eight SEM software approaches (Narayanan, 2012)
- Espkamp’s helpful original article on the semPlot package: (Narayanan, 2012)

In addition, the following internet resources can be helpful.

- Lavaan package home: [http://lavaan.ugent.be/]
- Google Group for Lavaan: [https://groups.google.com/forum/#!forum/lavaan]
- OpenMx package home: [https://openmx.ssri.psu.edu/wiki/projects]
- OpenMx package online forums: [https://openmx.ssri.psu.edu/forums]
- SEM package page on CRAN: [https://cran.r-project.org/web/packages/sem/index.html]
- lavaan package page on CRAN: [https://cran.r-project.org/web/packages/lavaan/index.html]
- OpenMx package page on Cran: [https://cran.r-project.org/web/packages/OpenMx/index.html]
- MVN package page on Cran: [https://cran.r-project.org/web/packages/MVN/index.html]
- semPlot package page on Cran: [https://cran.r-project.org/web/packages/semPlot/index.html]
Chapter 2

Prepare and Describe the Data

This chapter prepares the data set and does some univariate and multivariate description of its characteristics prior to the CFA implementation in later chapters. Both numeric and graphical description and inference about distribution shape are quickly available with R functions from the psych and MVN packages.

2.1 The Data Set

The 175 case data set (no missing observations) is loaded from a .csv file. The .csv file was exported from the SPSS system file that is available from the website for the Tabachnick textbook (Tabachnick et al., 2019). It has eleven subscales from the WISC:

- info (Information)
- comp (Comprehension)
- arith (Arithmetic)
- simil (Similarities)
- vocab (Vocabulary)
- digit (Digit Span)
- pictcomp (Picture Completion)
- parang (Picture Arrangement)
- block (Block Design)
- object (Object Assembly)
- coding (Coding - not sure if it is A or B, or a combination)

The user may recognize these scales as commonly discussed subtests of the WISC. The first 6 variables comprise the set of manifest variables for the latent factor known as Verbal. The last five are associated with Performance.

The original data file also contains an ID variable that is dropped for the working object created as wisc2 here.

```r
# import the primary data file
wisc1 <- read.csv("wisc1.csv")
knitr::kable(head(wisc1), booktabs=TRUE, format="markdown")
```

<table>
<thead>
<tr>
<th>ID</th>
<th>info</th>
<th>comp</th>
<th>arith</th>
<th>simil</th>
<th>vocab</th>
<th>digit</th>
<th>pictcomp</th>
<th>parang</th>
<th>block</th>
<th>object</th>
<th>coding</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>8</td>
<td>7</td>
<td>13</td>
<td>9</td>
<td>12</td>
<td>9</td>
<td>6</td>
<td>11</td>
<td>12</td>
<td>7</td>
<td>9</td>
</tr>
<tr>
<td>4</td>
<td>9</td>
<td>6</td>
<td>8</td>
<td>7</td>
<td>11</td>
<td>12</td>
<td>6</td>
<td>8</td>
<td>7</td>
<td>12</td>
<td>14</td>
</tr>
<tr>
<td>5</td>
<td>13</td>
<td>18</td>
<td>11</td>
<td>16</td>
<td>15</td>
<td>6</td>
<td>18</td>
<td>8</td>
<td>11</td>
<td>12</td>
<td>9</td>
</tr>
<tr>
<td>6</td>
<td>8</td>
<td>11</td>
<td>6</td>
<td>12</td>
<td>9</td>
<td>7</td>
<td>13</td>
<td>4</td>
<td>7</td>
<td>12</td>
<td>11</td>
</tr>
<tr>
<td>7</td>
<td>10</td>
<td>3</td>
<td>8</td>
<td>9</td>
<td>12</td>
<td>9</td>
<td>7</td>
<td>7</td>
<td>11</td>
<td>4</td>
<td>10</td>
</tr>
</tbody>
</table>
# create the working data frame by removing the ID variable
wisc2 <- wisc[,2:12]

A note about tables in this document: Many of the tables generated by the various R functions in this document are reformatted so that they do not appear as the plain text that is typically output into the R console. The `kable` function in the `knitr` package permits formatting that is well-rendered with `rmarkdown` and `bookdown` document production. `kable` is used frequently.

## 2.2 Numeric and Graphical Description of the Data

We can explore univariate characteristics of the data with summaries, plots, and evaluation of normality characteristics.

### 2.2.1 Univariate descriptive statistics from the psych package.

```r
knitr::kable(describe(wisc2, type=2, fast=T), booktabs=TRUE, format="markdown")
```

<table>
<thead>
<tr>
<th>vars</th>
<th>n</th>
<th>mean</th>
<th>sd</th>
<th>min</th>
<th>max</th>
<th>range</th>
<th>se</th>
</tr>
</thead>
<tbody>
<tr>
<td>info</td>
<td>1</td>
<td>9.4971</td>
<td>2.9123</td>
<td>3</td>
<td>19</td>
<td>16</td>
<td>0.22015</td>
</tr>
<tr>
<td>comp</td>
<td>2</td>
<td>10.0000</td>
<td>2.9653</td>
<td>0</td>
<td>18</td>
<td>18</td>
<td>0.22416</td>
</tr>
<tr>
<td>arith</td>
<td>3</td>
<td>9.0000</td>
<td>2.3069</td>
<td>4</td>
<td>16</td>
<td>12</td>
<td>0.17439</td>
</tr>
<tr>
<td>simil</td>
<td>4</td>
<td>10.6114</td>
<td>3.1836</td>
<td>2</td>
<td>18</td>
<td>16</td>
<td>0.24066</td>
</tr>
<tr>
<td>vocab</td>
<td>5</td>
<td>10.7029</td>
<td>2.9327</td>
<td>2</td>
<td>19</td>
<td>17</td>
<td>0.22169</td>
</tr>
<tr>
<td>digit</td>
<td>6</td>
<td>8.7314</td>
<td>2.7042</td>
<td>0</td>
<td>16</td>
<td>16</td>
<td>0.20442</td>
</tr>
<tr>
<td>pictcomp</td>
<td>7</td>
<td>10.6800</td>
<td>2.9342</td>
<td>2</td>
<td>19</td>
<td>17</td>
<td>0.22181</td>
</tr>
<tr>
<td>parang</td>
<td>8</td>
<td>10.3714</td>
<td>2.6597</td>
<td>2</td>
<td>17</td>
<td>15</td>
<td>0.20105</td>
</tr>
<tr>
<td>block</td>
<td>9</td>
<td>10.3143</td>
<td>2.7098</td>
<td>2</td>
<td>18</td>
<td>16</td>
<td>0.20484</td>
</tr>
<tr>
<td>object</td>
<td>10</td>
<td>10.9029</td>
<td>2.8440</td>
<td>3</td>
<td>19</td>
<td>16</td>
<td>0.21498</td>
</tr>
<tr>
<td>coding</td>
<td>11</td>
<td>8.5486</td>
<td>2.8721</td>
<td>0</td>
<td>15</td>
<td>15</td>
<td>0.21711</td>
</tr>
</tbody>
</table>

### 2.2.2 Univariate Distribution Tests and Plots plus Evaluation of Multivariate Normality

The `MVN` package provides univariate and multivariate normality tests. It is an efficient way to explore characteristics of a set of variables. Several options are available for testing both univariate and Multivariate normality. First, explicit calls for univariate and multivariate tests are made, and then an approach is shown that obtains all at once plus a useful set of plots.

```r
x_vars <- wisc2

# use the mvn function for an extensive evaluation
# note that different kinds of tests can be specified with changes in the arguments
result <- mvn(data= x_vars, mvnTest="mardia", univariateTest="AD")

kable(result$univariateNormality, booktabs=TRUE, format="markdown")
```

<table>
<thead>
<tr>
<th>Test</th>
<th>Variable</th>
<th>Statistic</th>
<th>p value</th>
<th>Normality</th>
</tr>
</thead>
<tbody>
<tr>
<td>Anderson-Darling</td>
<td>info</td>
<td>1.2049</td>
<td>0.0037</td>
<td>NO</td>
</tr>
<tr>
<td>Anderson-Darling</td>
<td>comp</td>
<td>1.3546</td>
<td>0.0016</td>
<td>NO</td>
</tr>
<tr>
<td>Test</td>
<td>Variable</td>
<td>Statistic</td>
<td>p value</td>
<td>Normality</td>
</tr>
<tr>
<td>-------------------</td>
<td>----------</td>
<td>-----------</td>
<td>---------</td>
<td>-----------</td>
</tr>
<tr>
<td>Anderson-Darling</td>
<td>arith</td>
<td>1.8656</td>
<td>1e-04</td>
<td>NO</td>
</tr>
<tr>
<td>Anderson-Darling</td>
<td>simil</td>
<td>0.9336</td>
<td>0.0175</td>
<td>NO</td>
</tr>
<tr>
<td>Anderson-Darling</td>
<td>vocab</td>
<td>1.4992</td>
<td>7e-04</td>
<td>NO</td>
</tr>
<tr>
<td>Anderson-Darling</td>
<td>digit</td>
<td>2.5087</td>
<td>&lt;0.001</td>
<td>NO</td>
</tr>
<tr>
<td>Anderson-Darling</td>
<td>pictcomp</td>
<td>1.1472</td>
<td>0.0052</td>
<td>NO</td>
</tr>
<tr>
<td>Anderson-Darling</td>
<td>parang</td>
<td>1.3391</td>
<td>0.0017</td>
<td>NO</td>
</tr>
<tr>
<td>Anderson-Darling</td>
<td>block</td>
<td>1.5225</td>
<td>6e-04</td>
<td>NO</td>
</tr>
<tr>
<td>Anderson-Darling</td>
<td>object</td>
<td>1.1593</td>
<td>0.0049</td>
<td>NO</td>
</tr>
<tr>
<td>Anderson-Darling</td>
<td>coding</td>
<td>1.2217</td>
<td>0.0034</td>
<td>NO</td>
</tr>
</tbody>
</table>

```
kable(result$multivariateNormality, booktabs=TRUE, format="markdown")
```

<table>
<thead>
<tr>
<th>Test</th>
<th>Statistic</th>
<th>p value</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mardia Skewness</td>
<td>348.496907430775</td>
<td>0.00673137335852766</td>
<td>NO</td>
</tr>
<tr>
<td>Mardia Kurtosis</td>
<td>2.30789388829536</td>
<td>0.0210050390817529</td>
<td>NO</td>
</tr>
<tr>
<td>MVN</td>
<td>NA</td>
<td>NA</td>
<td>NO</td>
</tr>
</tbody>
</table>

```
kable(mvn(data= x_vars, univariatePlot="histogram"), booktabs=TRUE, format="markdown")
```

<table>
<thead>
<tr>
<th>Test</th>
<th>Statistic</th>
<th>p value</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mardia Skewness</td>
<td>348.496907430775</td>
<td>0.00673137335852766</td>
<td>NO</td>
</tr>
<tr>
<td>Mardia Kurtosis</td>
<td>2.30789388829536</td>
<td>0.0210050390817529</td>
<td>NO</td>
</tr>
<tr>
<td>MVN</td>
<td>NA</td>
<td>NA</td>
<td>NO</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Test</th>
<th>Variable</th>
<th>Statistic</th>
<th>p value</th>
<th>Normality</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shapiro-Wilk</td>
<td>info</td>
<td>0.9814</td>
<td>0.0194</td>
<td>NO</td>
</tr>
<tr>
<td>Shapiro-Wilk</td>
<td>comp</td>
<td>0.9810</td>
<td>0.0170</td>
<td>NO</td>
</tr>
<tr>
<td>Shapiro-Wilk</td>
<td>arith</td>
<td>0.9709</td>
<td>0.0010</td>
<td>NO</td>
</tr>
<tr>
<td>Shapiro-Wilk</td>
<td>simil</td>
<td>0.9866</td>
<td>0.0937</td>
<td>YES</td>
</tr>
<tr>
<td>Shapiro-Wilk</td>
<td>vocab</td>
<td>0.9789</td>
<td>0.0093</td>
<td>NO</td>
</tr>
<tr>
<td>Shapiro-Wilk</td>
<td>digit</td>
<td>0.9693</td>
<td>0.0007</td>
<td>NO</td>
</tr>
<tr>
<td>Shapiro-Wilk</td>
<td>pictcomp</td>
<td>0.9836</td>
<td>0.0373</td>
<td>NO</td>
</tr>
<tr>
<td>Shapiro-Wilk</td>
<td>parang</td>
<td>0.9814</td>
<td>0.0193</td>
<td>NO</td>
</tr>
<tr>
<td>Shapiro-Wilk</td>
<td>block</td>
<td>0.9800</td>
<td>0.0127</td>
<td>NO</td>
</tr>
<tr>
<td>Shapiro-Wilk</td>
<td>object</td>
<td>0.9853</td>
<td>0.0638</td>
<td>YES</td>
</tr>
<tr>
<td>Shapiro-Wilk</td>
<td>coding</td>
<td>0.9820</td>
<td>0.0230</td>
<td>NO</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>n</th>
<th>Mean</th>
<th>Std.Dev</th>
<th>Median</th>
<th>Min</th>
<th>Max</th>
<th>25th</th>
<th>75th</th>
<th>Skew</th>
<th>Kurtosis</th>
</tr>
</thead>
<tbody>
<tr>
<td>info</td>
<td>175</td>
<td>9.4971</td>
<td>2.9123</td>
<td>10</td>
<td>3</td>
<td>8</td>
<td>11.5</td>
<td>0.08341</td>
<td>-0.08148</td>
</tr>
<tr>
<td>comp</td>
<td>175</td>
<td>10.0000</td>
<td>2.9653</td>
<td>10</td>
<td>0</td>
<td>8</td>
<td>12.0</td>
<td>0.08547</td>
<td>0.32620</td>
</tr>
<tr>
<td>arith</td>
<td>175</td>
<td>9.0000</td>
<td>2.3069</td>
<td>9</td>
<td>4</td>
<td>16</td>
<td>7</td>
<td>0.39097</td>
<td>-0.17897</td>
</tr>
<tr>
<td>simil</td>
<td>175</td>
<td>10.6114</td>
<td>3.1836</td>
<td>11</td>
<td>2</td>
<td>18</td>
<td>9</td>
<td>0.02233</td>
<td>-0.22985</td>
</tr>
<tr>
<td>vocab</td>
<td>175</td>
<td>10.7029</td>
<td>2.9327</td>
<td>10</td>
<td>2</td>
<td>19</td>
<td>9</td>
<td>0.26702</td>
<td>0.28751</td>
</tr>
<tr>
<td>digit</td>
<td>175</td>
<td>8.7314</td>
<td>2.7042</td>
<td>8</td>
<td>0</td>
<td>16</td>
<td>7</td>
<td>0.26631</td>
<td>0.07205</td>
</tr>
<tr>
<td>pictcomp</td>
<td>175</td>
<td>10.6800</td>
<td>2.9342</td>
<td>11</td>
<td>2</td>
<td>19</td>
<td>9</td>
<td>-0.07150</td>
<td>0.29116</td>
</tr>
<tr>
<td>parang</td>
<td>175</td>
<td>10.3714</td>
<td>2.6597</td>
<td>10</td>
<td>2</td>
<td>17</td>
<td>9</td>
<td>-0.19909</td>
<td>-0.05850</td>
</tr>
</tbody>
</table>
2.2. NUMERIC AND GRAPHICAL DESCRIPTION OF THE DATA

<table>
<thead>
<tr>
<th></th>
<th>n</th>
<th>Mean</th>
<th>Std.Dev</th>
<th>Median</th>
<th>Min</th>
<th>Max</th>
<th>25th</th>
<th>75th</th>
<th>Skew</th>
<th>Kurtosis</th>
</tr>
</thead>
<tbody>
<tr>
<td>block</td>
<td>175</td>
<td>10.3143</td>
<td>2.7098</td>
<td>10</td>
<td>2</td>
<td>18</td>
<td>9</td>
<td>12.0</td>
<td>-0.21972</td>
<td>0.50196</td>
</tr>
<tr>
<td>object</td>
<td>175</td>
<td>10.9029</td>
<td>2.8440</td>
<td>11</td>
<td>3</td>
<td>19</td>
<td>9</td>
<td>13.0</td>
<td>-0.12289</td>
<td>0.14745</td>
</tr>
<tr>
<td>coding</td>
<td>175</td>
<td>8.5486</td>
<td>2.8721</td>
<td>9</td>
<td>0</td>
<td>15</td>
<td>6</td>
<td>11.0</td>
<td>-0.05283</td>
<td>-0.45376</td>
</tr>
</tbody>
</table>

![Histograms](image)
2.2.3 Multivariate Outlier tests

The **MVN** package permits a good array of diagnostic tests/plots for univariate/multivariate shape and outliers.

First, multivariate outliers are examined with the Mahalanobis distance:

\[
\text{result2} \leftarrow \text{mvn(data=x_vars, multivariateOutlierMethod="quan")}
\]

![Chi-Square Q-Q Plot](image-url)
Next, multivariate outliers are evaluated with the Adjusted-Mahalanobis distance:

```r
result2 <- mvn(data=x_vars, multivariateOutlierMethod="adj")
```
2.3  Bivariate Characteristics of the data set

We can quickly explore numerical and graphical summaries of the eleven variables.

2.3.1  SPLOM

Among the many scatterplot matrix capabilities in R, John Fox’ _scatterplot.matrix_ function in his _car_ package has probably been seen by most students.

```
scatterplotMatrix(wisc2,cex=.2,
   smooth=list(col.smooth="red", spread=F, lwd.smooth=.3),
   col="skyblue1",
   regLine=list(lwd=.3,col="black"))
```

Even with some control over colors and sizes of points/lines, this SPLOM probably has too many variables to be effective - each plot is very small. Nonetheless, the sense of fairly linear relationships among all pairs is somewhat apparent, as is the relative univariate normality of each of the eleven.

Note that the image can be enlarged if the reader is using a pdf version of this document simply by using the increase/decrease size capability of pdf readers. If the user is reading an html version of this document, then try to do a right mouse click on the image and “view image” (in Windows). Then the image can be increased in size in a browser.
2.4 Covariances and Zero Order Correlations

The covariance matrix is the basic input for the CFA algorithms outlined in later chapters.

```r
covmatrix1 <- round(cov(wisc2), digits=3)
knitr::kable(covmatrix1, booktabs=TRUE, format="markdown")
```

<table>
<thead>
<tr>
<th></th>
<th>info</th>
<th>comp</th>
<th>arith</th>
<th>simil</th>
<th>vocab</th>
<th>digit</th>
<th>pictcomp</th>
<th>parang</th>
<th>block</th>
<th>object</th>
<th>coding</th>
</tr>
</thead>
<tbody>
<tr>
<td>info</td>
<td>8.481</td>
<td>4.034</td>
<td>3.322</td>
<td>4.758</td>
<td>5.338</td>
<td>2.720</td>
<td>1.965</td>
<td>1.561</td>
<td>1.808</td>
<td>1.531</td>
<td>0.059</td>
</tr>
<tr>
<td>comp</td>
<td>4.034</td>
<td>8.793</td>
<td>2.684</td>
<td>4.816</td>
<td>4.621</td>
<td>1.891</td>
<td>3.540</td>
<td>1.471</td>
<td>2.966</td>
<td>2.718</td>
<td>0.517</td>
</tr>
<tr>
<td>arith</td>
<td>3.322</td>
<td>2.684</td>
<td>5.322</td>
<td>2.713</td>
<td>2.621</td>
<td>1.678</td>
<td>1.052</td>
<td>1.391</td>
<td>1.701</td>
<td>0.282</td>
<td>0.598</td>
</tr>
<tr>
<td>simil</td>
<td>4.758</td>
<td>4.816</td>
<td>2.713</td>
<td>10.136</td>
<td>5.022</td>
<td>2.234</td>
<td>3.450</td>
<td>2.524</td>
<td>2.255</td>
<td>2.433</td>
<td>-0.372</td>
</tr>
<tr>
<td>vocab</td>
<td>5.338</td>
<td>4.621</td>
<td>2.621</td>
<td>5.022</td>
<td>8.601</td>
<td>2.334</td>
<td>2.456</td>
<td>1.031</td>
<td>2.364</td>
<td>1.546</td>
<td>0.842</td>
</tr>
<tr>
<td>digit</td>
<td>2.720</td>
<td>1.891</td>
<td>1.678</td>
<td>2.234</td>
<td>2.334</td>
<td>7.313</td>
<td>0.597</td>
<td>1.066</td>
<td>0.533</td>
<td>0.267</td>
<td>1.344</td>
</tr>
<tr>
<td>pictcomp</td>
<td>1.965</td>
<td>3.540</td>
<td>1.052</td>
<td>3.450</td>
<td>2.456</td>
<td>0.597</td>
<td>8.610</td>
<td>1.941</td>
<td>3.038</td>
<td>3.032</td>
<td>-0.605</td>
</tr>
<tr>
<td>parang</td>
<td>1.561</td>
<td>1.471</td>
<td>1.391</td>
<td>2.524</td>
<td>1.031</td>
<td>1.066</td>
<td>1.941</td>
<td>7.074</td>
<td>2.532</td>
<td>1.916</td>
<td>0.289</td>
</tr>
<tr>
<td>block</td>
<td>1.808</td>
<td>2.966</td>
<td>1.701</td>
<td>2.255</td>
<td>2.364</td>
<td>0.533</td>
<td>3.038</td>
<td>2.532</td>
<td>7.343</td>
<td>3.077</td>
<td>0.832</td>
</tr>
<tr>
<td>object</td>
<td>1.531</td>
<td>2.718</td>
<td>0.282</td>
<td>2.433</td>
<td>1.546</td>
<td>0.267</td>
<td>3.032</td>
<td>1.916</td>
<td>3.077</td>
<td>8.088</td>
<td>0.433</td>
</tr>
<tr>
<td>coding</td>
<td>0.059</td>
<td>0.517</td>
<td>0.598</td>
<td>-0.372</td>
<td>0.842</td>
<td>1.344</td>
<td>-0.605</td>
<td>0.289</td>
<td>0.832</td>
<td>0.433</td>
<td>8.249</td>
</tr>
</tbody>
</table>
We can use the \texttt{Corrplot} package to produce a useful combination of a schematic and correlation matrix.

```r
mat1 <- cor(wisc2)
corrplot(mat1, type="upper", tl.pos="tp")
corrplot(mat1, add=T, type="lower", method="number",
          col="black", diag=FALSE, tl.pos="n", cl.pos="n")
```

![Correlation Matrix Diagram](image)

<table>
<thead>
<tr>
<th>info</th>
<th>comp</th>
<th>arith</th>
<th>simil</th>
<th>vocab</th>
<th>digit</th>
<th>pictcomp</th>
<th>parang</th>
<th>block</th>
<th>object</th>
<th>coding</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.47</td>
<td>0.49</td>
<td>0.51</td>
<td>0.51</td>
<td>0.53</td>
<td>0.35</td>
<td>0.23</td>
<td>0.2</td>
<td>0.26</td>
<td>0.18</td>
<td>0.01</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.49</td>
<td>0.39</td>
<td>0.51</td>
<td>0.24</td>
<td>0.41</td>
<td>0.39</td>
<td>0.27</td>
<td>0.32</td>
<td>0.06</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.51</td>
<td>0.37</td>
<td>0.27</td>
<td>0.37</td>
<td>0.54</td>
<td>0.26</td>
<td>0.25</td>
<td>0.04</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.54</td>
<td>0.26</td>
<td>0.29</td>
<td>0.51</td>
<td>0.41</td>
<td>0.27</td>
<td>0.27</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.29</td>
<td>0.08</td>
<td>0.39</td>
<td>0.35</td>
<td>0.18</td>
<td>0.09</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.08</td>
<td>0.37</td>
<td>0.07</td>
<td>0.29</td>
<td>0.15</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.37</td>
<td>0.26</td>
<td>0.07</td>
<td>0.08</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.38</td>
<td>0.38</td>
<td>0.03</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.36</td>
<td>0.06</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.11</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Chapter 3

Using the lavaan package for CFA

One of the primary tools for SEM in R is the lavaan package. It permits path specification with a simple syntax.

3.1 Implement the CFA, First Model

Using the lavaan package, we can implement directly the CFA with only a few steps. Since this document contains three different packages’ approach to CFA, the packages used for each will be loaded at that point, so as to not have confusion over common function names.

```r
library(lavaan)
```

3.1.1 Define and fit the first model

The latent variables and their paths to the manifest variables are initially defined as simple textual specifications. The =~ symbol is the key to defining paths. We have two latent variables and no manifest variable has duplicate paths from both latents. This is a so-called “simple structure.”

Note that this text string uses variable names that match what is in the wisc2 data set.

```r
wisc.model1 <- "verbal =~ info + comp + arith + simil + vocab + digit
performance =~ pictcomp + parang + block + object + coding"
```

Fit the model and obtain the basic summary. Note that in this default approach, the latent factors are permitted to covary and the model estimates this covariance.

One R syntax note…. the format here to call the cfa function (lavaan::cfa(.....)) is employed to ensure no ambiguity that the correct cfa function is the one from the lavaan package. This precludes confusion when multiple packages contain functions with the same name as is the case with both lavaan and sem which is also used in this document. Even though sem is loaded later in this document, if there is a chance that it may simultaneously exist in the R environment with lavaan then the approach here is safer.

```r
fit1 <- lavaan::cfa(wisc.model1, data=wisc2, std.lv=TRUE)
summary(fit1, fit.measures=T, standardized=T)
```

```
## lavaan 0.6-3 ended normally after 24 iterations
##
## Optimization method    :       NLMINB
## Number of free parameters:       23
## Number of observations     :      175
```
## Estimator ML

## Model Fit Test Statistic 70.640

## Degrees of freedom 43

## P-value (Chi-square) 0.005

## Model test baseline model:

## Minimum Function Test Statistic 519.204

## Degrees of freedom 55

## P-value 0.000

## User model versus baseline model:

## Comparative Fit Index (CFI) 0.940

## Tucker-Lewis Index (TLI) 0.924

## Loglikelihood and Information Criteria:

## Loglikelihood user model (H0) -4491.822

## Loglikelihood unrestricted model (H1) -4456.502

## Number of free parameters 23

## Akaike (AIC) 9029.643

## Bayesian (BIC) 9102.433

## Sample-size adjusted Bayesian (BIC) 9029.600

## Root Mean Square Error of Approximation:

## RMSEA 0.061

## 90 Percent Confidence Interval 0.033 0.085

## P-value RMSEA <= 0.05 0.233

## Standardized Root Mean Square Residual:

## SRMR 0.059

## Parameter Estimates:

## Information Expected

## Information saturated (h1) model Structured

## Standard Errors Standard

## Latent Variables:

| Estimate | Std.Err | z-value | P(>|z|) | Std.lv | Std.all |
|----------|---------|---------|---------|--------|---------|
| verbal =~
| info     | 2.206   | 0.200   | 11.029  | 0.000  | 2.206   | 0.760 |
| comp     | 2.042   | 0.210   | 9.709   | 0.000  | 2.042   | 0.691 |
| arith    | 1.300   | 0.172   | 7.555   | 0.000  | 1.300   | 0.565 |
| simil    | 2.232   | 0.225   | 9.940   | 0.000  | 2.232   | 0.703 |
| vocab    | 2.250   | 0.200   | 11.225  | 0.000  | 2.250   | 0.770 |
| digit    | 1.053   | 0.212   | 4.967   | 0.000  | 1.053   | 0.390 |
| performance =~
| pictcomp | 1.742   | 0.242   | 7.187   | 0.000  | 1.742   | 0.595 |
## 3.1. IMPLEMENT THE CFA, FIRST MODEL

### parang 1.253 0.224 5.582 0.000 1.253 0.473
### block 1.846 0.222 8.311 0.000 1.846 0.683
### object 1.605 0.236 6.800 0.000 1.605 0.566
### coding 0.207 0.254 0.814 0.416 0.207 0.072

### Covariances:
| estimate | std.err | z-value | p(|z|) | std.lv | std.all |
|----------|---------|---------|--------|--------|---------|
| verbal~~performance | 0.589 | 0.075 | 7.814 | 0.000 | 0.589 | 0.589 |

### Variances:
| estimate | std.err | z-value | p(|z|) | std.lv | std.all |
|----------|---------|---------|--------|--------|---------|
| .info | 3.566 | 0.507 | 7.034 | 0.000 | 3.566 | 0.423 |
| .comp | 4.572 | 0.585 | 7.815 | 0.000 | 4.572 | 0.523 |
| .arith | 3.602 | 0.420 | 8.571 | 0.000 | 3.602 | 0.681 |
| .simil | 5.096 | 0.662 | 7.702 | 0.000 | 5.096 | 0.506 |
| .vocab | 3.487 | 0.506 | 6.886 | 0.000 | 3.487 | 0.408 |
| .digit | 6.162 | 0.680 | 9.056 | 0.000 | 6.162 | 0.848 |
| .pictcomp | 5.526 | 0.757 | 7.296 | 0.000 | 5.526 | 0.646 |
| .parang | 5.463 | 0.658 | 8.298 | 0.000 | 5.463 | 0.777 |
| .block | 3.894 | 0.640 | 6.083 | 0.000 | 3.894 | 0.533 |
| .object | 5.467 | 0.719 | 7.600 | 0.000 | 5.467 | 0.680 |
| .coding | 8.159 | 0.874 | 9.335 | 0.000 | 8.159 | 0.995 |
| verbal | 1.000 | 1.000 | 1.000 |
| performance | 1.000 | 1.000 | 1.000 |

### 3.1.2 Obtain coefficients

It is possible to obtain the output from the above `summary` function that does not contain the parameter coefficients part of the table. If that had been the implementation, then the coefficients can be obtained simply, but it is better to obtain the full table of parameter estimates and errors, etc.

```r
# obtain only the coefficients
kable(coef(fit1), booktabs=TRUE, format="markdown")
```

### 3.1.3 Complete parameter listing

Instead of directly printing the parameter table, I prefer to reformat it with `kable` when using R Markdown. Without using `kable`, the code would be the following:

```r
parameterEstimates(fit1, standardized=T)
```

Format the table and clean it up using `kable`.

```r
parameterEstimates(fit1, standardized=TRUE) %>%
  filter(op == "~") %>%
  select('Latent Factor'=lhs, Indicator=rhs, B=est, SE=se, Z=z, 'p-value'=pvalue, Beta=std.all) %>%
  knitr::kable(digits = 3, booktabs=TRUE, format="markdown", caption="Factor Loadings")
```

<table>
<thead>
<tr>
<th>Latent Factor</th>
<th>Indicator</th>
<th>B</th>
<th>SE</th>
<th>Z</th>
<th>p-value</th>
<th>Beta</th>
</tr>
</thead>
<tbody>
<tr>
<td>verbal</td>
<td>info</td>
<td>2.206</td>
<td>0.200</td>
<td>11.029</td>
<td>0.000</td>
<td>0.760</td>
</tr>
<tr>
<td>verbal</td>
<td>comp</td>
<td>2.042</td>
<td>0.210</td>
<td>9.709</td>
<td>0.000</td>
<td>0.691</td>
</tr>
<tr>
<td>verbal</td>
<td>arith</td>
<td>1.300</td>
<td>0.172</td>
<td>7.555</td>
<td>0.000</td>
<td>0.565</td>
</tr>
<tr>
<td>verbal</td>
<td>simil</td>
<td>2.232</td>
<td>0.225</td>
<td>9.940</td>
<td>0.000</td>
<td>0.703</td>
</tr>
<tr>
<td>verbal</td>
<td>vocab</td>
<td>2.250</td>
<td>0.200</td>
<td>11.225</td>
<td>0.000</td>
<td>0.770</td>
</tr>
</tbody>
</table>
### CHAPTER 3. USING THE LAVAAN PACKAGE FOR CFA

<table>
<thead>
<tr>
<th>Latent Factor</th>
<th>Indicator</th>
<th>B</th>
<th>SE</th>
<th>Z</th>
<th>p-value</th>
<th>Beta</th>
</tr>
</thead>
<tbody>
<tr>
<td>verbal</td>
<td>digit</td>
<td>1.053</td>
<td>0.212</td>
<td>4.967</td>
<td>0.000</td>
<td>0.390</td>
</tr>
<tr>
<td>performance</td>
<td>pictcomp</td>
<td>1.742</td>
<td>0.242</td>
<td>7.187</td>
<td>0.000</td>
<td>0.595</td>
</tr>
<tr>
<td>performance</td>
<td>parang</td>
<td>1.253</td>
<td>0.224</td>
<td>5.582</td>
<td>0.000</td>
<td>0.473</td>
</tr>
<tr>
<td>performance</td>
<td>block</td>
<td>1.846</td>
<td>0.222</td>
<td>8.311</td>
<td>0.000</td>
<td>0.683</td>
</tr>
<tr>
<td>performance</td>
<td>object</td>
<td>1.605</td>
<td>0.236</td>
<td>6.800</td>
<td>0.000</td>
<td>0.566</td>
</tr>
<tr>
<td>performance</td>
<td>coding</td>
<td>0.207</td>
<td>0.254</td>
<td>0.814</td>
<td>0.416</td>
<td>0.072</td>
</tr>
</tbody>
</table>

#### 3.1.4 Residuals correlation matrix

Residuals can be examined.

```r
cor_table <- residuals(fit1, type = "cor")$cov
# cor_table[upper.tri(cor_table)] <- # erase the upper triangle
# diag(cor_table) <- NA # erase the diagonal 0's
knitr::kable(cor_table, digits=3, format="markdown", booktabs=TRUE) # makes a nice table and rounds everything
```

<table>
<thead>
<tr>
<th></th>
<th>info</th>
<th>comp</th>
<th>arith</th>
<th>simil</th>
<th>vocab</th>
<th>digit</th>
<th>pictcomp</th>
<th>parang</th>
<th>block</th>
<th>object</th>
<th>coding</th>
</tr>
</thead>
<tbody>
<tr>
<td>info</td>
<td>0.000</td>
<td>-0.058</td>
<td>0.065</td>
<td>-0.021</td>
<td>0.040</td>
<td>0.049</td>
<td>-0.036</td>
<td>-0.010</td>
<td>-0.076</td>
<td>-0.068</td>
<td>-0.025</td>
</tr>
<tr>
<td>comp</td>
<td>-0.058</td>
<td>0.000</td>
<td>0.002</td>
<td>0.025</td>
<td>0.000</td>
<td>-0.034</td>
<td>0.165</td>
<td>-0.006</td>
<td>0.091</td>
<td>0.092</td>
<td>0.031</td>
</tr>
<tr>
<td>arith</td>
<td>0.065</td>
<td>0.002</td>
<td>0.000</td>
<td>-0.028</td>
<td>-0.047</td>
<td>0.048</td>
<td>-0.043</td>
<td>0.069</td>
<td>0.045</td>
<td>-0.145</td>
<td>0.066</td>
</tr>
<tr>
<td>simil</td>
<td>-0.021</td>
<td>0.025</td>
<td>-0.028</td>
<td>0.000</td>
<td>-0.003</td>
<td>-0.015</td>
<td>0.123</td>
<td>0.102</td>
<td>-0.021</td>
<td>0.034</td>
<td>-0.071</td>
</tr>
<tr>
<td>vocab</td>
<td>0.040</td>
<td>0.000</td>
<td>-0.047</td>
<td>-0.003</td>
<td>0.000</td>
<td>-0.006</td>
<td>0.016</td>
<td>-0.082</td>
<td>-0.012</td>
<td>-0.071</td>
<td>0.067</td>
</tr>
<tr>
<td>digit</td>
<td>0.049</td>
<td>-0.034</td>
<td>0.048</td>
<td>-0.015</td>
<td>-0.006</td>
<td>0.000</td>
<td>-0.062</td>
<td>0.040</td>
<td>-0.084</td>
<td>-0.095</td>
<td>0.156</td>
</tr>
<tr>
<td>pictcomp</td>
<td>-0.036</td>
<td>0.165</td>
<td>-0.043</td>
<td>0.123</td>
<td>0.016</td>
<td>-0.062</td>
<td>0.000</td>
<td>-0.033</td>
<td>-0.025</td>
<td>0.026</td>
<td>-0.115</td>
</tr>
<tr>
<td>parang</td>
<td>-0.010</td>
<td>-0.006</td>
<td>0.069</td>
<td>0.102</td>
<td>-0.082</td>
<td>0.040</td>
<td>-0.033</td>
<td>0.000</td>
<td>0.028</td>
<td>-0.014</td>
<td>0.004</td>
</tr>
<tr>
<td>block</td>
<td>-0.076</td>
<td>0.091</td>
<td>0.045</td>
<td>-0.021</td>
<td>-0.012</td>
<td>-0.084</td>
<td>-0.025</td>
<td>0.028</td>
<td>0.000</td>
<td>0.013</td>
<td>0.058</td>
</tr>
<tr>
<td>object</td>
<td>-0.068</td>
<td>0.092</td>
<td>-0.145</td>
<td>0.034</td>
<td>-0.071</td>
<td>-0.095</td>
<td>0.026</td>
<td>-0.014</td>
<td>0.013</td>
<td>0.000</td>
<td>0.012</td>
</tr>
<tr>
<td>coding</td>
<td>-0.025</td>
<td>0.031</td>
<td>0.066</td>
<td>-0.071</td>
<td>0.067</td>
<td>0.156</td>
<td>-0.115</td>
<td>0.004</td>
<td>0.058</td>
<td>0.012</td>
<td>0.000</td>
</tr>
</tbody>
</table>

#### 3.1.5 Plot the residuals

```r
# extract the residuals from the fit1 model
# get rid of the duplicates and diagonal values
# create a vector for a
res1 <- residuals(fit1, type = "cor")$cov
res1[upper.tri(res1, diag=T)] <- NA
v1 <- as.vector(res1)
v2 <- v1[!is.na(v1)]
qqPlot(v2, id=F)
```
3.1. IMPLEMENT THE CFA, FIRST MODEL

3.1.6 Modification Indices

Modification indices provide……

\[
kable(modificationIndices(fit1, sort=TRUE, minimum.value=3), booktabs=TRUE, format="markdown")
\]

<table>
<thead>
<tr>
<th>lhs</th>
<th>op</th>
<th>rhs</th>
<th>mi</th>
<th>epc</th>
<th>sepc.lv</th>
<th>sepc.all</th>
<th>sepc.nox</th>
</tr>
</thead>
<tbody>
<tr>
<td>32</td>
<td>performance =~ comp</td>
<td>9.8232</td>
<td>0.93721</td>
<td>0.93721</td>
<td>0.31696</td>
<td>0.31696</td>
<td></td>
</tr>
<tr>
<td>62</td>
<td>arith =~ object</td>
<td>6.3624</td>
<td>-0.94453</td>
<td>-0.94453</td>
<td>-0.21285</td>
<td>-0.21285</td>
<td></td>
</tr>
<tr>
<td>37</td>
<td>info =~ comp</td>
<td>5.2214</td>
<td>-0.98473</td>
<td>-0.98473</td>
<td>-0.24388</td>
<td>-0.24388</td>
<td></td>
</tr>
<tr>
<td>81</td>
<td>digit =~ coding</td>
<td>4.9267</td>
<td>1.20723</td>
<td>1.20723</td>
<td>0.17025</td>
<td>0.17025</td>
<td></td>
</tr>
<tr>
<td>51</td>
<td>comp =~ pictcomp</td>
<td>4.6854</td>
<td>0.96931</td>
<td>0.96931</td>
<td>0.19284</td>
<td>0.19284</td>
<td></td>
</tr>
<tr>
<td>85</td>
<td>pictcomp =~ coding</td>
<td>4.5748</td>
<td>-1.22491</td>
<td>-1.22491</td>
<td>-0.18242</td>
<td>-0.18242</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>performance =~ info</td>
<td>4.4766</td>
<td>-0.59552</td>
<td>-0.59552</td>
<td>-0.20507</td>
<td>-0.20507</td>
<td></td>
</tr>
<tr>
<td>40</td>
<td>info =~ vocab</td>
<td>4.4029</td>
<td>0.91242</td>
<td>0.91242</td>
<td>0.25875</td>
<td>0.25875</td>
<td></td>
</tr>
<tr>
<td>73</td>
<td>vocab =~ parang</td>
<td>4.1459</td>
<td>-0.79773</td>
<td>-0.79773</td>
<td>-0.18277</td>
<td>-0.18277</td>
<td></td>
</tr>
<tr>
<td>38</td>
<td>info =~ arith</td>
<td>4.1333</td>
<td>0.69702</td>
<td>0.69702</td>
<td>0.19450</td>
<td>0.19450</td>
<td></td>
</tr>
<tr>
<td>67</td>
<td>simil =~ parang</td>
<td>3.3732</td>
<td>0.83050</td>
<td>0.83050</td>
<td>0.15741</td>
<td>0.15741</td>
<td></td>
</tr>
<tr>
<td>70</td>
<td>simil =~ coding</td>
<td>3.2711</td>
<td>-0.95602</td>
<td>-0.95602</td>
<td>-0.14827</td>
<td>-0.14827</td>
<td></td>
</tr>
<tr>
<td>66</td>
<td>simil =~ pictcomp</td>
<td>3.2695</td>
<td>0.86025</td>
<td>0.86025</td>
<td>0.16212</td>
<td>0.16212</td>
<td></td>
</tr>
</tbody>
</table>

3.1.7 Path Diagram for the bifactor Model 1

The \texttt{semPlot} package provides a capability of drawing path diagrams for cfa and other sem models. The \texttt{semPaths} function will take a cfa model object and draw the diagram, with several options available. The
diagram produced here takes control over font/label sizes, display of residuals, and color of paths/coefficients. These and many more control options are available. There is a challenge in producing these path diagrams to have font sizes large enough for most humans to read. I’ve taken control of the font sizes for the “edges” with a cex argument. But this causes overlap in the values if the default layout is used. I found that “circle2” worked best here.

```r
# Note that the base plot, including standardized path coefficients plots positive coefficients green # and negative coefficients red. Red-green colorblindness issues anyone?
# I redrew it here to choose a blue and red. But all the coefficients in this example are # positive, so they are shown with the skyblue.
# more challenging to use colors other than red and green. not in this doc
semPaths(fit1, residuals=F, sizeMan=7, "std",
  posCol=c("skyblue4", "red"),
  #edge.color="skyblue4",
  edge.label.cex=1.2, layout="circle2")
```

```r
# or we could draw the paths in such a way to include the residuals:
#semPaths(fit1, sizeMan=7,"std",edge.color="skyblue4",edge.label.cex=1,layout="circle2")
# the base path diagram can be drawn much more simply:
#semPaths(fit1)
# or
semPaths(fit1,"std")
```
3.1.8 Orthogonal fit comparison

Compare to a one factor solution.

```r
fitlorth <- lavaan::cfa(wisc.model1, data=wisc1, orthogonal=T)
kable(anova(fit1, fitlorth), booktabs=TRUE, format="markdown")
```

<table>
<thead>
<tr>
<th>Df</th>
<th>AIC</th>
<th>BIC</th>
<th>Chisq</th>
<th>Chisq diff</th>
<th>Df diff</th>
<th>Pr(&gt;Chisq)</th>
</tr>
</thead>
<tbody>
<tr>
<td>fit1</td>
<td>43</td>
<td>9029.6</td>
<td>9102.4</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>fitlorth</td>
<td>44</td>
<td>9065.6</td>
<td>9135.2</td>
<td>70.64</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>37.97</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

3.2 Generate a second model and compare

Next, generate a second CFA model. There are theoretical reasons why paths from both latent factors to “comp” might be warranted. The “comp” variable also has the largest Modification index in the first model. This second model implements this one-path parameter change.

3.2.1 Add a path (Perf to comp) and Fit the second CFA model

Define the additional path in the model text string.

```r
wisc.model2 <- 'verbal =~ info + comp + arith + simil + vocab + digit
  performance =~ pictcomp + parang + block + object + coding + comp'
```

Fit the model and obtain the basic summary.
fit2 <- lavaan::cfa(wisc.model2, data=wisc1, std.lv=TRUE)
summary(fit2, fit.measures=T, standardized=T)

## lavaan 0.6-3 ended normally after 26 iterations
##
## Optimization method  NLMINB
## Number of free parameters  24
##
## Number of observations  175
##
## Estimator  ML
## Model Fit Test Statistic  60.642
## Degrees of freedom  42
## P-value (Chi-square)  0.031
##
## Model test baseline model:
##
## Minimum Function Test Statistic  519.204
## Degrees of freedom  55
## P-value  0.000
##
## User model versus baseline model:
##
## Comparative Fit Index (CFI)  0.960
## Tucker-Lewis Index (TLI)  0.947
##
## Loglikelihood and Information Criteria:
##
## Loglikelihood user model (H0)  -4486.823
## Loglikelihood unrestricted model (H1)  -4456.502
##
## Number of free parameters  24
## Akaike (AIC)  9021.645
## Bayesian (BIC)  9097.600
## Sample-size adjusted Bayesian (BIC)  9021.600
##
## Root Mean Square Error of Approximation:
##
## RMSEA  0.050
## 90 Percent Confidence Interval  0.016  0.077
## P-value RMSEA <= 0.05  0.465
##
## Standardized Root Mean Square Residual:
##
## SRMR  0.054
##
## Parameter Estimates:
##
## Information  Expected
## Information saturated (h1) model  Structured
## Standard Errors  Standard
##
## Latent Variables:
##
## Estimate  Std.Err  z-value  P(>|z|)  Std.lv  Std.all
3.2. GENERATE A SECOND MODEL AND COMPARE

## verbal =~
| Indicator | Estimate | Std.Err | z-value | P(>|z|) | Std.lv | Std.all |
|-----------|----------|---------|---------|--------|-------|---------|
| info      | 2.256    | 0.199   | 11.318  | 0.000  | 2.256 | 0.777   |
| comp      | 1.491    | 0.254   | 5.877   | 0.000  | 1.491 | 0.504   |
| arith     | 1.307    | 0.172   | 7.584   | 0.000  | 1.307 | 0.568   |
| simil     | 2.205    | 0.226   | 9.748   | 0.000  | 2.205 | 0.695   |
| vocab     | 2.273    | 0.201   | 11.329  | 0.000  | 2.273 | 0.777   |
| digit     | 1.075    | 0.212   | 5.068   | 0.000  | 1.075 | 0.399   |

## Covariances:

## verbal =~
| performance | Estimate | Std.Err | z-value | P(>|z|) | Std.lv | Std.all |
|-------------|----------|---------|---------|--------|-------|---------|
| pictcomp    | 1.790    | 0.239   | 7.495   | 0.000  | 1.790 | 0.612   |
| parang      | 1.189    | 0.224   | 5.317   | 0.000  | 1.189 | 0.448   |
| block       | 1.823    | 0.219   | 8.334   | 0.000  | 1.823 | 0.675   |
| object      | 1.633    | 0.233   | 7.010   | 0.000  | 1.633 | 0.576   |
| coding      | 0.200    | 0.253   | 0.793   | 0.428  | 0.200 | 0.070   |
| comp        | 0.884    | 0.266   | 3.324   | 0.000  | 0.884 | 0.299   |

## performance =~
| Indicator | Estimate | Std.Err | z-value | P(>|z|) | Std.lv | Std.all |
|-----------|----------|---------|---------|--------|-------|---------|
| info      | 0.533    | 0.081   | 6.594   | 0.000  | 0.533 | 0.533   |

## Variances:

## info
| Estimate | Std.Err | z-value | P(>|z|) | Std.lv | Std.all |
|----------|---------|---------|--------|-------|---------|
| 3.343    | 0.502   | 6.665   | 0.000  | 3.343 | 0.396   |
| comp     | 4.331    | 0.554   | 7.819   | 0.000  | 4.331 | 0.495   |
| arith    | 3.584    | 0.420   | 8.533   | 0.000  | 3.584 | 0.677   |
| simil    | 5.217    | 0.675   | 7.726   | 0.000  | 5.217 | 0.518   |
| vocab    | 3.383    | 0.508   | 6.655   | 0.000  | 3.383 | 0.396   |
| digit    | 6.116    | 0.677   | 9.032   | 0.000  | 6.116 | 0.841   |
| pictcomp | 5.358    | 0.741   | 7.231   | 0.000  | 5.358 | 0.626   |
| parang   | 5.620    | 0.663   | 8.480   | 0.000  | 5.620 | 0.799   |
| block    | 3.979    | 0.623   | 6.385   | 0.000  | 3.979 | 0.545   |
| object   | 5.376    | 0.707   | 7.603   | 0.000  | 5.376 | 0.668   |
| coding   | 8.162    | 0.874   | 9.337   | 0.000  | 8.162 | 0.995   |

3.2.2 Obtain coefficients

Coefficients can be obtained simply with the coef function, but it is better to do the full parameter listing in the next section.

```r
knitr::kable(coef(fit2), booktabs=TRUE, format="markdown")
```

It is simple to obtain the full parameter list, but I prefer to use kable for tables when I can.

```r
parameterEstimates(fit2, standardized=TRUE)
```

Reformat the table and clean it up by using kable.

```r
parameterEstimates(fit2, standardized=TRUE) %>%
  filter(op == "=") %>%
  select(Latent Factor=lhs, Indicator=rhs, B=est, SE=se, Z=z, 'p-value'=pvalue, Beta=std.all) %>%
  knitr::kable(digits = 3, format="markdown", booktabs=TRUE, caption="Factor Loadings")
```

<table>
<thead>
<tr>
<th>Latent Factor</th>
<th>Indicator</th>
<th>B</th>
<th>SE</th>
<th>Z</th>
<th>p-value</th>
<th>Beta</th>
</tr>
</thead>
<tbody>
<tr>
<td>verbal</td>
<td>info</td>
<td>2.256</td>
<td>0.199</td>
<td>11.318</td>
<td>0.000</td>
<td>0.777</td>
</tr>
</tbody>
</table>
### 3.2.3 Residuals correlation matrix

Residuals can be examined.

```r
residuals(fit2, type = "cor")$cov

# erase the upper triangle
# diag(cor_table) <- NA # erase the diagonal 0's
knitr::kable(cor_table2, digits=3, format="markdown", booktabs=TRUE) # makes a nice table and rounds everyhing to 2 digits
```

### 3.2.4 Modification Indices for Model 2

Modification indices provide.................

```r
kable(modificationIndices(fit2, sort=TRUE, minimum.value=3), booktabs=TRUE, format="markdown")
```
3.2. GENERATE A SECOND MODEL AND COMPARE

<table>
<thead>
<tr>
<th>lhs</th>
<th>op</th>
<th>rhs</th>
<th>mi</th>
<th>epc</th>
<th>sepc.lv</th>
<th>sepc.all</th>
<th>sepc.nox</th>
</tr>
</thead>
<tbody>
<tr>
<td>61</td>
<td>arith</td>
<td>~~~</td>
<td>block</td>
<td>3.2569</td>
<td>0.60993</td>
<td>0.60993</td>
<td>0.16152</td>
</tr>
<tr>
<td>38</td>
<td>info</td>
<td>~~~</td>
<td>arith</td>
<td>3.2166</td>
<td>0.62088</td>
<td>0.62088</td>
<td>0.17938</td>
</tr>
<tr>
<td>70</td>
<td>simil</td>
<td>~~~</td>
<td>coding</td>
<td>3.1652</td>
<td>-0.94894</td>
<td>-0.94894</td>
<td>-0.14543</td>
</tr>
</tbody>
</table>

3.2.5 Path Diagram for Model 2

The semPlot package provides a capability of drawing path diagrams for cfa and other sem models. The semPaths function will take a cfa model object and draw the diagram, with several options available. The diagram produced here takes control over font/label sizes, display of residuals, and color of paths/coefficients. These and many more control options are available. There is a challenge in producing these path diagrams to have font sizes large enough for most humans to read. I've taken control of the font sizes for the “edges” with a cex argument. But this causes overlap in the values if the default layout is used. I found that “circle2” worked best here.

```R
# Note that the base plot, including standardized path coefficients plots positive coefficients green
# and negative coefficients red. Red-green colorblindness issues anyone?
# I redrew it here to choose a blue and red. But all the coefficients in this example are
# positive, so they are shown with the skyblue.
# more challenging to use colors other than red and green. not in this doc
semPaths(fit2, residuals=F, sizeMan=7, "std",
    posCol=c("skyblue4", "red"),
    #edge.color="skyblue4",
    edge.label.cex=1.2, layout="circle2")
```
3.3 Compare Model 1 and Model 2

Model 1 is first/base. Model 2 adds a path from Perf to comp... Compare these models.

```r
kable(anova(fit1, fit2), booktabs = TRUE, format = "markdown")
```

<table>
<thead>
<tr>
<th></th>
<th>Df</th>
<th>AIC</th>
<th>BIC</th>
<th>Chisq</th>
<th>Chisq diff</th>
<th>Df diff</th>
<th>Pr(&gt;Chisq)</th>
</tr>
</thead>
<tbody>
<tr>
<td>fit2</td>
<td>42</td>
<td>9021.6</td>
<td>9097.6</td>
<td>60.642</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>fit1</td>
<td>43</td>
<td>9029.6</td>
<td>9102.4</td>
<td>70.640</td>
<td>9.998</td>
<td>1</td>
<td>0.00157</td>
</tr>
</tbody>
</table>

3.4 An additional perspective on estimation and optimization

Subtle differences in algorithmic strategies for Structural Equation Modeling exist among software packages. Users are often familiar with only the default approaches and will only be moved to learn other approaches when the default approach fails to converge or produces problematic models. The student first confronted with these methods will often assume that the same SEM model evaluated in different software (e.g., LISREL, MPlus, EQS, lavaan, sem, OpenMx) will produce the same model outcome. This may not be a safe
3.4. AN ADDITIONAL PERSPECTIVE ON ESTIMATION AND OPTIMIZATION

assumption. Different “dialects” exist for the various software products. The commercial products may use algorithms that are proprietary and not available to understand their approach. The open source products (e.g., lavaan) have made source code available for inspection.

The perceptive reader will notice that the default solutions given by the three R packages employed in this document (lavaan, sem, OpenMx) all give the same values for parameter estimates and goodness of fit statistics for the same two models run with each. However, there are slight differences in these quantities compared to the published LISREL analysis of the same data in the Tabachnik, et al textbook (2019). Other readers will have noticed that the LISREL output in this textbook matches SAS output that they may have seen with class coverage of the CFA topic. The R packages, while agreeing with each other, vary slightly from the LISREL and SAS values. The degree of difference is slight, but its existence may puzzle some students. The answer to understanding these differences goes well beyond the scope of this document and involves an advanced understanding of the computational algorithms employed. Even though all of the approaches are Maximum Likelihood methods some optimization and estimation strategies can differ.

The lavaan package permits some insight into this with one of the arguments available for the cfa function used in this chapter. The reader might want to examine the help docs for this function: ?cfa. That help page directs readers to another help document on Lavaan Options (lavoptions). There, one can find an argument that can be passed to cfa, called “mimic”. Here is the text from that section, describing the various “mimic” possibilities:

“If”Mplus”, an attempt is made to mimic the Mplus program. If”EQS”, an attempt is made to mimic the EQS program. If”default”, the value is (currently) set to “lavaan”, which is very close to”Mplus”.

The reader may have been exposed to a SAS Proc Calis approach to this problem that employed the default Method called LINEQS. The following rerun of the first model from this chapter above, employs the mimic argument to be specified as “EQS”. The product of this model is a set of parameter values and fit statistics (e.g., Chi Sq) that match the SAS output (and the Tabachnick et al LISREL output) exactly. Demonstrating the equivalence with the addition of the mimic argument does not fully explain why such differences originally existed, but that, again, is well beyond the scope of this document. The reference section to this document includes some articles that address these differences (El-Sheikh et al., 2017; Narayanan, 2012). Rosseel (2012) has discussed use of the ‘mimic’ function that guides lavaan to emulate the approach of some of the commercial products. His website is also a valuable resource on this.

```r
wisc.model1 <- "verbal =~ info + comp + arith + simil + vocab + digit
performance =~ pictcomp + parang + block + object + coding"

fit1eqs <- lavaan::cfa(wisc.model1, data=wisc2, std.lv=TRUE, mimic="EQS")
summary(fit1eqs, fit.measures=T, standardized=T)
```

```
## lavaan 0.6-3 ended normally after 25 iterations
##
## Optimization method       NLMINB
## Number of free parameters 23
##
## Number of observations    175
##
## Estimator     ML
## Model Fit Test Statistic 70.236
## Degrees of freedom     43
## P-value (Chi-square) 0.005
##
## Model test baseline model:
##
## Minimum Function Test Statistic 516.237
## Degrees of freedom   55
## P-value                0.000
```
## User model versus baseline model:

## Comparative Fit Index (CFI) 0.941
## Tucker-Lewis Index (TLI) 0.924

## Loglikelihood and Information Criteria:
## Loglikelihood user model (H0) -4497.337
## Loglikelihood unrestricted model (H1) -4462.018
## Number of free parameters 23
## Akaike (AIC) 9040.675
## Bayesian (BIC) 9113.465
## Sample-size adjusted Bayesian (BIC) 9040.631

## Root Mean Square Error of Approximation:
## RMSEA 0.060
## 90 Percent Confidence Interval 0.033 0.085
## P-value RMSEA <= 0.05 0.239

## Standardized Root Mean Square Residual:
## SRMR 0.059

## Parameter Estimates:

## Information
## Information saturated (h1) model Structured
## Standard Errors Standard

## Latent Variables:

| Estimate | Std.Err | z-value | P(>|z|) | Std.lv | Std.all |
|----------|---------|---------|---------|--------|---------|
| verbal =~
| info 2.212 0.201 10.997 0.000 2.212 0.760 |
| comp 2.048 0.212 9.682 0.000 2.048 0.691 |
| arith 1.304 0.173 7.534 0.000 1.304 0.565 |
| simil 2.238 0.226 9.911 0.000 2.238 0.703 |
| vocab 2.257 0.202 11.193 0.000 2.257 0.770 |
| digit 1.056 0.213 4.952 0.000 1.056 0.390 |

## Covariances:

| Estimate | Std.Err | z-value | P(>|z|) | Std.lv | Std.all |
|----------|---------|---------|---------|--------|---------|
| verbal ~ performance 0.589 0.076 7.792 0.000 0.589 0.589 |

## Variances:
## Estimate Std.Err  z-value  P(>|z|)  Std.lv  Std.all
## .info 3.586 0.511  7.014  0.000  3.586  0.423
## .comp 4.599 0.590  7.793  0.000  4.599  0.523
## .arith 3.623 0.424  8.547  0.000  3.623  0.681
## .simil 5.125 0.667  7.680  0.000  5.125  0.506
## .vocab 3.507 0.511  6.866  0.000  3.507  0.408
## .digit 6.198 0.686  9.030  0.000  6.198  0.848
## .pictcomp 5.558 0.764  7.276  0.000  5.558  0.646
## .parang 5.494 0.664  8.275  0.000  5.494  0.777
## .block 3.916 0.646  6.066  0.000  3.916  0.533
## .object 5.499 0.726  7.578  0.000  5.499  0.680
## .coding 8.206 0.882  9.309  0.000  8.206  0.995
## verbal 1.000 1.000  1.000  1.000
## performance 1.000 1.000  1.000  1.000
Chapter 4

Using the sem package for CFA

In this chapter, we use the sem package to implement the same two CFA analyses that we produced with lavaan in chapter 3. sem provides an equally simple way to obtain the models and only the basics are shown here. The code in this chapter is modeled after a document by James Steiger

4.1 Example one

Once again, the bifactor model with Verbal and Performance factors is specified. Each manifest factor has a path from only one of the two factors.

4.1.1 Data Setup

In sem, it is helpful to have covariance and correlation matrices available as objects, as well as a sample size object.

```r
# same data file and extraction of the wisc2 data frame with only the 11 manifests
#wisc1 <- read.csv("wisc1.csv")
wisc2 <- wisc1[,2:12]
# covariance and correlation matrices are saved as objects
covwisc <- cov(wisc2)
corwisc <- cor(wisc2)
# list of manifest variables for potential use
manifests <- names(wisc2)
# this gives an object that is the sample size
wobs <- length(wisc2)
```

We also need to load the package.

```r
library(sem)
```

4.1.2 Define the first model

In sem, the structure of the model is created with a text string to define the paths. The specifyModel function permits this in several ways. The simplest is to enter text as an argument. Some explanation of the structure is needed.

- Each line defines a path, a label for the parameter, and the starting value for the parameter value.

- This is symbolized as: , , .
• Note that paths can be double or single-headed arrows.

• The unique name specified by the parameter symbol is for free parameters.

• If the parameter value is NA, then its starting point value is system determined.

• A numerical value following the NA can fix a variance at the value. E.g.,: F1 <-> F1, NA, 1 would fix the factor variance a 1.
• Unique variances can be specified for manifest variables. e.g.,: manifest1 <-> manifest1, e1, NA. If they are not specified, they default to “free to vary”
• Factors can be set to a fixed relationship to each other, e.g, 0, or 1. Or they can be left free (estimable) as in the example here.

# this text could have been saved in a file and read in with the file argument to be efficient
# commented here to show the argument options
# m1.model <- specifyModel(file="sem1.txt")
m1.model <- specifyModel(text="
## Factor 1 is Verbal
Verbal -> info, t01, NA
Verbal -> comp, t02, NA
Verbal -> arith, t03, NA
Verbal -> simil, t04, NA
Verbal -> vocab, t05, NA
Verbal -> digit, t06, NA
## Factor 2 is performance
Performance -> pictcomp, t07, NA
Performance -> parang, t08, NA
Performance -> block, t09, NA
Performance -> object, t10, NA
Performance -> coding, t11, NA
## Set factor variances
Verbal <-> Verbal, NA, 1
Performance <-> Performance, NA, 1
## Set factor covariance to be estimable
Verbal <-> Performance, p1, NA"
)

## NOTE: it is generally simpler to use specifyEquations() or cfa()
## see ?specifyEquations

## NOTE: adding 11 variances to the model

This m1.model is now available to be used in the model function. The first “note” refers to the fact that the model can be specified interactively (I think). I found it easier to do this way, and more reproducible.
4.1.3 Fit model 1 and examine the results

In this chapter, we will just look at the basics of the model fit and draw the path diagram. Other aspects of the model can be extracted, but are passed over here to save space. The interested reader might try `str(m1)` to see what is available from the model object.

```r
# all that is required is the model specification, the covariance matrix, and the sample size
m1 <- sem(m1.model, covwisc, 175)
options(digits=5)
summary(m1)
```

```
##
## Model Chisquare = 70.236 Df = 43 Pr(>Chisq) = 0.0054454
## AIC = 116.24
## BIC = -151.85
##
## Normalized Residuals
## Min. 1st Qu. Median Mean 3rd Qu. Max.
## -1.88413 -0.41375 -0.00001 0.03511 0.45976 2.11208
##
## R-square for Endogenous Variables
## info comp arith simil vocab digit pictcomp parang
## 0.5772 0.4770 0.3193 0.4944 0.5922 0.1524 0.3545 0.2233
## block object coding
## 0.4667 0.3202 0.0052
##
## Parameter Estimates
## Estimate Std Error z value Pr(>|z|)
## t01 2.21249 0.201190 10.99700 3.9507e-28
## t02 2.04806 0.211541 9.68163 3.6090e-22
## t03 1.30357 0.173035 7.53358 4.9367e-14
## t04 2.23845 0.225846 9.91142 4.4281e-29
## t05 2.25691 0.201642 11.19269 4.4281e-29
## t06 1.05579 0.213185 4.95244 7.3287e-07
## t07 1.74699 0.243776 7.16638 7.7008e-13
## t08 1.25683 0.225785 5.56647 2.5995e-08
## t09 1.85123 0.223392 8.28693 1.1621e-16
## t10 1.60918 0.237324 6.78052 1.1974e-11
## t11 0.20761 0.255890 0.81132 4.1718e-01
## pl 0.58883 0.075569 7.79198 6.5965e-15
## V[info] 3.58620 0.511281 7.01414 2.3137e-12
## V[comp] 4.59856 0.590106 7.92777 6.5566e-15
## V[arith] 3.62254 0.423850 8.54675 1.2660e-17
## V[simil] 5.12484 0.667296 7.68001 1.5908e-14
## V[vocab] 3.50720 0.510787 6.86626 6.5906e-12
## V[digit] 6.19783 0.686345 9.03020 1.7136e-19
## V[pictcomp] 5.55770 0.763882 7.27560 3.4488e-13
## V[parang] 5.49428 0.663998 8.27454 1.2895e-16
## V[block] 3.91612 0.645638 6.06550 1.3154e-09
## V[object] 5.49872 0.725619 7.57798 3.5099e-14
## V[coding] 8.20596 0.881552 9.30853 1.2961e-20
##
## t01  info <-- Verbal
## t02  comp <-- Verbal
## t03  arith <-- Verbal
```
The table listing of parameter estimates uses the labeling strategy that was defined in the `modelSpecify` function. The names, such as theta01 are arbitrary.
4.1.4 Can we draw the path diagram from a sem model?

The `semPaths` function from `semPlot` is capable of recognizing a model object from `sem`. This code is identical to what was used in chapter 2 for the `lavaan` object. The fit object name is just changed to `m1` here.

```r
# Note that the base plot, including standardized path coefficients plots positive coefficients green
# and negative coefficients red. Red-green colorblindness issues anyone?
# I redrew it here to choose a blue and red. But all the coefficients in this example are
# positive, so they are shown with the skyblue.
# more challenging to use colors other than red and green. Not in this doc
semPaths(m1, residuals=F, sizeMan=7, "std",
    posCol=c("skyblue4", "red"),
    edge.color="skyblue4",
    edge.label.cex=1.2, layout="circle2")
```

```r
# or we could draw the paths in such a way to include the residuals:
#semPaths(m1, sizeMan=7,"std",edge.color="skyblue4",edge.label.cex=1,layout="circle2")
# the base path diagram can be drawn much more simply:
#semPaths(m1)
# or
semPaths(m1,"std")
```
4.1. EXAMPLE ONE
4.2 Example two

Model with added path still under construction in this chapter.

4.2.1 Define the second model

In the `specifyModel` text argument we just need to add one path, from Performance to “comp”, rerun the model, and compare to the first.

```r
# this text could have been saved in a file and read in with the file argument to be efficient
# commented here to show the argument options
# m1.model <- specifyModel(file="sem2.txt")

m2.model <- specifyModel(text="
## Factor 1 is Verbal
Verbal -> info, t01, NA
Verbal -> comp, t02, NA
Verbal -> arith, t03, NA
Verbal -> simil, t04, NA
Verbal -> vocab, t05, NA
Verbal -> digit, t06, NA
## Factor 2 is performance
Performance -> pictcomp, t07, NA
Performance -> parang, t08, NA
Performance -> block, t09, NA
Performance -> object, t10, NA
Performance -> coding, t11, NA
Performance -> comp, t12, NA
## Set factor variances
Verbal <-> Verbal, NA, 1
Performance <-> Performance, NA, 1
## Set factor covariance to be estimable
Verbal <-> Performance, p1, NA"
)
```

## NOTE: it is generally simpler to use specifyEquations() or cfa()
## see ?specifyEquations
## NOTE: adding 11 variances to the model

4.2.2 Fit model 2 and examine the results

In this chapter, we will just look at the basics of the model fit and draw the path diagram. Other aspects of the model can be extracted, but are passed over here to save space. The interested reader might try `str(m1)` to see what is available from the model object.

```r
# all that is required is the model specification, the covariance matrix, and the sample size
m2 <- sem(m2.model,covwisc,175)
options(digits=5)
summary(m2)
```

## Model Chisquare = 60.295   Df = 42 Pr(>Chisq) = 0.033354
## AIC = 108.3
## BIC = -156.63
##
## Normalized Residuals
##     Min. 1st Qu. Median Mean 3rd Qu. Max.
4.2. EXAMPLE TWO

## -1.7087 -0.3670 0.0000 0.0439 0.4509 2.0855
## R-square for Endogenous Variables
## info comp arith simil vocab digit pictcomp parang
## 0.6036 0.5046 0.3227 0.4823 0.6044 0.1589 0.3741 0.2009
## block object coding
## 0.4551 0.3315 0.0049
## Parameter Estimates
## Estimate Std Error z value Pr(>|z|)
## t01 2.26254 0.200471 11.28609 1.5373e-29
## t02 1.49558 0.255230 5.85973 4.6361e-09
## t03 1.31053 0.173300 7.56223 3.9620e-14
## t04 2.21107 0.227463 9.72056 2.4641e-22
## t05 2.28001 0.201829 11.29675 1.3616e-29
## t06 1.07784 0.213286 5.05351 4.3377e-07
## t07 1.79476 0.240149 7.47352 7.8078e-14
## t08 1.19223 0.224879 5.30164 1.1477e-07
## t09 1.82799 0.219975 8.30998 9.5719e-17
## t10 1.63752 0.234254 6.99037 2.7416e-12
## t11 0.20104 0.254172 0.79098 4.2896e-01
## t12 0.88667 0.267519 3.31443 9.1829e-04
## p1 0.53322 0.081099 6.57487 4.8695e-11
## V[info] 3.36224 0.505946 6.64546 3.0227e-11
## V[comp] 4.35598 0.558735 7.79614 6.3830e-15
## V[arith] 3.60434 0.423598 8.50888 1.7563e-17
## V[simil] 5.24665 0.681040 7.70388 1.3199e-14
## V[vocab] 3.40241 0.512709 6.63614 3.2200e-11
## V[digit] 6.15077 0.682983 9.00574 2.1421e-19
## V[pictcomp] 5.38850 0.747358 7.21007 5.5922e-13
## V[parang] 5.65249 0.668468 8.45589 2.7697e-17
## V[block] 4.00164 0.628568 6.36628 1.9367e-10
## V[object] 5.40673 0.713201 7.58094 3.4306e-14
## V[coding] 8.20865 0.881648 9.31057 1.2715e-20
## t01 info <----- Verbal
## t02 comp <----- Verbal
## t03 arith <----- Verbal
## t04 simil <----- Verbal
## t05 vocab <----- Verbal
## t06 digit <----- Verbal
## t07 pictcomp <----- Performance
## t08 parang <----- Performance
## t09 block <----- Performance
## t10 object <----- Performance
## t11 coding <----- Performance
## t12 comp <----- Performance
## p1 Performance <---> Verbal
## V[info] info <---> info
## V[comp] comp <---> comp
## V[arith] arith <---> arith
## V[simil] simil <---> simil
## V[vocab] vocab <---> vocab
## V[digit] digit <---> digit
The table listing of parameter estimates uses the labeling strategy that was defined in the `modelSpecify` function. The names, such as theta01 are arbitrary.
4.2.3 Can we draw the path diagram from a sem model?

The `semPaths` function from `semPlot` is capable of recognizing a model object from `sem`. This code is identical to what was used in chapter 2 for the `lavaan` object. The fit object name is just changed to `m2` here.

```
# Note that the base plot, including standardized path coefficients plots positive coefficients green # and negative coefficients red. Red-green colorblindness issues anyone?
# I redrew it here to choose a blue and red. But all the coefficients in this example are # positive, so they are shown with the skyblue.
# more challenging to use colors other than red and green. not in this doc
semPaths(m2, residuals=F, sizeMan=7, "std",
    posCol=c("skyblue4", "red"),
    edge.color="skyblue4",
    edge.label.cex=1.2, layout="circle2")
```

# or we could draw the paths in such a way to include the residuals:
#semPaths(m1, sizeMan=7,"std",edge.color="skyblue4",edge.label.cex=1,layout="circle2")
# the base path diagram can be drawn much more simply:
#semPaths(m1)
# or
semPaths(m2,"std")
4.3 Compare the two CFA models produced by sem

We can use the anova function to compare the two models, as we did for lavaan.

```r
kable(anova(m2,m1), booktabs=TRUE, format="markdown")
```

<table>
<thead>
<tr>
<th></th>
<th>Model</th>
<th>Df</th>
<th>Model Chisq</th>
<th>Df</th>
<th>LR Chisq</th>
<th>Pr(&gt;Chisq)</th>
</tr>
</thead>
<tbody>
<tr>
<td>m2</td>
<td>42</td>
<td>60.295</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>m1</td>
<td>43</td>
<td>70.236</td>
<td>1</td>
<td>9.9409</td>
<td>0.00162</td>
<td></td>
</tr>
</tbody>
</table>
Chapter 5

Using the OpenMx Package for CFA

The OpenMx package in R is a port of the well-respected MX analytical software. It handles SEM and can easily be used for CFA. Here, the same two models that were run in lavaan will be run again, but an additional model will be run first.

```r
library(OpenMx)
```

5.1 First OpenMx Model - Single Factor

First, fit a bad model that posits only one underlying factor

5.1.1 set new dataframe and set up basics

Some initial setups

```r
# grab data
wisc2 <- wisc1[2:12] # remove ID from df
# set up basics
manifests = names(wisc2)
observedCov <- cov(wisc2)
umSubjects <- nrow(wisc2)
```

5.1.2 Create the model using the mxModel function

This code initializes the model and sets vars/paths

```r
latents = "F1"
cfa2a <- mxModel("Common Factor Model", type="RAM",
manifestVars = manifests, latentVars = latents,
# Now set the residual variance for manifest variables
mxPath(from=manifests, arrows=2, free=T, values=1, labels=paste("error", 1:11, sep="")),
# set latent factor variance to 1
mxPath(from="F1", arrows=2, free=F, values=1, labels="varF1"),
# specify factor loadings
mxPath(from="F1", to=manifests, arrows=1,
free=T, values=1, labels=paste("i", 1:11, sep="")),
# specify the covariance matrix
mxData(observed=observedCov, type="cov", numObs=numSubjects)
) # close the model
```
5.1.3 Now use \texttt{mxRun} to fit the model

\texttt{mxRun} uses the model defined above.

\begin{verbatim}
cfa2a <- mxRun(cfa2a)
\end{verbatim}

\texttt{## Running Common Factor Model with 22 parameters}

Summarize the fit:

\begin{verbatim}
summary(cfa2a)
\end{verbatim}

\texttt{## Summary of Common Factor Model}

\begin{verbatim}
## free parameters:
## name matrix row col Estimate Std.Error A
## 1 i1 A info F1 2.10843 0.20489
## 2 i2 A comp F1 2.10258 0.20876
## 3 i3 A arith F1 1.27253 0.17310
## 4 i4 A simil F1 2.25978 0.22323
## 5 i5 A vocab F1 2.17498 0.20342
## 6 i6 A digit F1 1.00892 0.21308
## 7 i7 A pictcomp F1 1.35055 0.22867
## 8 i8 A parang F1 0.89834 0.21227
## 9 i9 A block F1 1.01784 0.22717
## 10 i10 A coding F1 0.20128 0.23480
## 11 error1 S info info 3.98738 0.54543
## 12 error2 S comp comp 4.32199 0.56950
## 13 error3 S arith arith 3.67209 0.42704
## 14 error4 S simil simil 4.97100 0.64992
## 15 error5 S vocab vocab 3.82117 0.53031
## 16 error6 S digit digit 6.25280 0.68878
## 17 error7 S pictcomp pictcomp 6.73647 0.76246
## 18 error8 S parang parang 6.22646 0.68288
## 19 error9 S block block 5.78440 0.65307
## 20 error10 S object object 7.00600 0.77245
## 21 error11 S coding coding 8.16142 0.87333
##
## Model Statistics:
## | Parameters | Degrees of Freedom | Fit (-2lnL units) |
## | Model: 22 | 44 | 5492.0 |
## | Saturated: 66 | 0 | 5375.1 |
## | Independence: 11 | 55 | 5894.3 |
## Number of observations/statistics: 175/66
##
## \texttt{## chi-square: $\chi^2$ ( df=44 ) = 116.85, \ p = 1.5993e-08} \n## Information Criteria: \n## | df Penalty | Parameters Penalty | Sample-Size Adjusted |
## | AIC: 28.851 | 160.85 | 167.51 |
## | BIC: -110.400 | 230.48 | 160.81 |
## CFI: 0.84306
## TLI: 0.80383 \ (also known as NNFI) \n## RMSEA: 0.097268 \ [95\% CI (0.071763, 0.12282)] \n## Prob(RMSEA $\leq$ 0.05): 0.00026603 \n## timestamp: 2019-07-11 11:27:19
What is in the model?

```r
slotNames(cfa2a@output)
```

```r
## NULL

names(cfa2a@output)
```

```r
## [1] "matrices"    "algebras"
## [3] "data"        "SaturatedLikelihood"
## [5] "IndependenceLikelihood" "calculatedHessian"
## [7] "standardErrors" "gradient"
## [9] "hessian"     "expectations"
## [11] "fit"         "fitUnits"
## [13] "Minus2LogLikelihood" "maxRelativeOrdinalError"
## [15] "minimum"     "estimate"
## [17] "infoDefinite" "conditionNumber"
## [19] "status"      "iterations"
## [21] "evaluations" "mxVersion"
## [23] "frontendTime" "backendTime"
## [25] "independentTime" "wallTime"
## [27] "timestamp"   "cpuTime"
```
5.2 Second OpenMx Model - the bifactor model

Now fit a model that is the same as the initial model fit with lavaan in chapter 3. Two factors, verbal and performance are established with each manifest variable uniquely specified by only one of the two latent factors.

5.2.1 set new dataframe and set up basics

Same initial setups as above.

```r
# grab data
wisc2 <- wisc1[2:12]  # remove ID from df
# set up basics
manifests <- names(wisc2)
verbalVars <- names(wisc2[1:6])
perfVars <- names(wisc2[7:11])
latents2 <- c("verbal","perf")
#manifests <- c("verbalVars","perfVars")
observedCov <- cov(wisc2[,1:11])
numSubjects <- nrow(wisc2)
```

5.2.2 Create the model using the mxModel function

Essentially initializes the model and sets vars/paths

```r
cfa2b <- mxModel("Two Factor Model",type="RAM",
  manifestVars = manifests, latentVars = latents2,
  # Now set the residual variance for manifest variables
  mxPath(from = manifestVars, arrows=2,free=T,values=1, labels=paste("e",1:11,sep="")),
  # set latent factor variances
  mxPath(from=latents2,arrows=2,free=F,values=1,labels=c("p1","p2")),
  # specify factor loadings
  # Allow the factors to covary as per the lavaan model
  mxPath(from="verbal", to="perf", arrows=2, free=T, values=1, labels="latcov1"),
  # Specify the latent factor paths to manifests
  mxPath(from="verbal", to = verbalVars,
    arrows=1,
    free=T,values=1),
  mxPath(from="perf", to = perfVars,
    arrows=1,
    free=T,values=1),
  mxData(observed=observedCov,type="cov",numObs=numSubjects)
)  # close the model
```
Now use `mxRun` to fit the model

mxRun uses the model defined above

```r
cfa2b <- mxRun(cfa2b)
```

### Running Two Factor Model with 23 parameters

Summarize the fit

```r
summary(cfa2b)
```

### Summary of Two Factor Model

### free parameters:

<table>
<thead>
<tr>
<th></th>
<th>name</th>
<th>matrix</th>
<th>row</th>
<th>col</th>
<th>Estimate</th>
<th>Std.Error</th>
<th>A</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Two Factor</td>
<td>A</td>
<td>info</td>
<td>verbal</td>
<td>2.20616</td>
<td>0.201309</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Model.A[1,12]</td>
<td>A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Two Factor</td>
<td>A</td>
<td>comp</td>
<td>verbal</td>
<td>2.04220</td>
<td>0.211937</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Two Factor</td>
<td>A</td>
<td>arith</td>
<td>verbal</td>
<td>1.29984</td>
<td>0.172679</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Two Factor</td>
<td>A</td>
<td>simil</td>
<td>verbal</td>
<td>2.23205</td>
<td>0.225198</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Model.A[4,12]</td>
<td>A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Two Factor</td>
<td>A</td>
<td>vocab</td>
<td>verbal</td>
<td>2.25045</td>
<td>0.200598</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>Two Factor</td>
<td>A</td>
<td>digit</td>
<td>verbal</td>
<td>1.05277</td>
<td>0.212308</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>Model.A[6,12]</td>
<td>A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Two Factor</td>
<td>A</td>
<td>pictcomp</td>
<td>perf</td>
<td>1.74200</td>
<td>0.246173</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>Two Factor</td>
<td>A</td>
<td>parang</td>
<td>perf</td>
<td>1.25323</td>
<td>0.224773</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>Two Factor</td>
<td>A</td>
<td>block</td>
<td>perf</td>
<td>1.84593</td>
<td>0.224562</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>Two Factor</td>
<td>A</td>
<td>object</td>
<td>perf</td>
<td>1.60457</td>
<td>0.236048</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>Two Factor</td>
<td>A</td>
<td>coding</td>
<td>perf</td>
<td>0.20701</td>
<td>0.257118</td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>e1</td>
<td>S</td>
<td>info</td>
<td>info</td>
<td>3.56570</td>
<td>0.516752</td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>e2</td>
<td>S</td>
<td>comp</td>
<td>comp</td>
<td>4.57229</td>
<td>0.594874</td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>e3</td>
<td>S</td>
<td>arith</td>
<td>arith</td>
<td>3.60184</td>
<td>0.422011</td>
<td></td>
</tr>
<tr>
<td>26</td>
<td>e4</td>
<td>S</td>
<td>simil</td>
<td>simil</td>
<td>5.09555</td>
<td>0.666203</td>
<td></td>
</tr>
<tr>
<td>27</td>
<td>e5</td>
<td>S</td>
<td>vocab</td>
<td>vocab</td>
<td>3.48717</td>
<td>0.507536</td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>e6</td>
<td>S</td>
<td>digit</td>
<td>digit</td>
<td>6.16241</td>
<td>0.680961</td>
<td></td>
</tr>
<tr>
<td>29</td>
<td>e7</td>
<td>S</td>
<td>pictcomp</td>
<td>pictcomp</td>
<td>5.52590</td>
<td>0.772061</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>e8</td>
<td>S</td>
<td>parang</td>
<td>parang</td>
<td>5.46287</td>
<td>0.658943</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>e9</td>
<td>S</td>
<td>block</td>
<td>block</td>
<td>3.89376</td>
<td>0.651701</td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>e10</td>
<td>S</td>
<td>object</td>
<td>object</td>
<td>5.46733</td>
<td>0.719732</td>
<td></td>
</tr>
<tr>
<td>33</td>
<td>e11</td>
<td>S</td>
<td>coding</td>
<td>coding</td>
<td>8.15907</td>
<td>0.874195</td>
<td></td>
</tr>
<tr>
<td>34</td>
<td>latcov1</td>
<td>S</td>
<td>verbal</td>
<td>perf</td>
<td>0.58883</td>
<td>0.077178</td>
<td></td>
</tr>
</tbody>
</table>

### Model Statistics:

<table>
<thead>
<tr>
<th></th>
<th>Parameters</th>
<th>Degrees of Freedom</th>
<th>Fit (-2lnL units)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model:</td>
<td>23</td>
<td>43</td>
<td>5445.7</td>
</tr>
<tr>
<td>Saturated:</td>
<td>66</td>
<td>0</td>
<td>5375.1</td>
</tr>
<tr>
<td>Independence:</td>
<td>11</td>
<td>55</td>
<td>5894.3</td>
</tr>
</tbody>
</table>

Number of observations/statistics: 175/66

### chi-square: $<\chi^2>$ (df=43) = 70.608, $p = 0.0050089$

### Information Criteria:

<table>
<thead>
<tr>
<th></th>
<th>df Penalty</th>
<th>Parameters Penalty</th>
<th>Sample-Size Adjusted</th>
</tr>
</thead>
<tbody>
<tr>
<td>AIC</td>
<td>-15.392</td>
<td>116.61</td>
<td>123.92</td>
</tr>
<tr>
<td>BIC</td>
<td>-151.478</td>
<td>189.40</td>
<td>116.56</td>
</tr>
<tr>
<td>CFI</td>
<td>0.94053</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TLI</td>
<td>0.92393</td>
<td>(also known as NNFI)</td>
<td></td>
</tr>
<tr>
<td>RMSEA</td>
<td>0.060571</td>
<td>[95% CI (0.026575, 0.089595)]</td>
<td>0.23348</td>
</tr>
</tbody>
</table>
What is in the model?

```r
slotNames(cfa2b@output)
```

```
## NULL
names(cfa2b@output)
```

```r
## [1] "matrices"   "algebras"
## [3] "data"       "SaturatedLikelihood"
## [5] "IndependenceLikelihood" "calculatedHessian"
## [7] "standardErrors" "gradient"
## [9] "hessian"     "expectations"
## [11] "fit"         "fitUnits"
## [13] "Minus2LogLikelihood" "maxRelativeOrdinalError"
## [15] "minimum"    "estimate"
## [17] "infoDefinite" "conditionNumber"
## [19] "status"     "iterations"
## [21] "evaluations" "mxVersion"
## [23] "frontendTime" "backendTime"
## [25] "independentTime" "wallTime"
## [27] "timestamp"  "cpuTime"
```
Can we use semPlot to draw the OpenMx model fit?

The `semPlot` package is very powerful and can recognize many lm and sem model objects. We can use the identical code that we used in chapter 2 for the `lavaan` model.

```r
# Note that the base plot, including standardized path coefficients plots positive coefficients green
# and negative coefficients red. Red-green colorblindness issues anyone?
# I redrew it here to choose a blue and red. But all the coefficients in this example are
# positive, so they are shown with the skyblue.
# more challenging to use colors other than red and green. not in this doc
semPaths(cfa2b, residuals=F, sizeMan=7, "std",
    posCol=c("skyblue4", "red"),
    #edge.color="skyblue4",
    edge.label.cex=1.2,
    layout="circle2")
```

# or we could draw the paths in such a way to include the residuals:
#semPaths(fit1, sizeMan=7,"std",edge.color="skyblue4",edge.label.cex=1,layout="circle2")
# the base path diagram can be drawn much more simply:
#semPaths(fit1)
# or
semPaths(cfa2b,"std")
5.3 Third OpenMx Model

Now fit a model that is the same as the second model fit with lavaan above. Two factors, verbal and performance, plus paths from both latents to “comp”.

5.3.1 Set new dataframe and set up basics

Same initial setups as above.

```r
# grab data
wisc2 <- wisc1[2:12]  # remove ID from df
# set up basics
manifests <- names(wisc2)
verbalVars <- names(wisc2[1:6])
perfVars <- c(names(wisc2[7:11]), "comp")
lats2 <- c("verbal", "perf")
#manifests <- c("verbalVars","perfVars")
observedCov <- cov(wisc2[,1:11])
numSubjects <- nrow(wisc2)
```

5.3.2 Create the model using the mxModel function

Essentially initializes the model and sets vars/paths

```r
cfa2c <- mxModel("TwoFac, Comp Common", type="RAM",
manifestVars = manifests, latentVars = latents2,
  # Now set the residual variance for manifest variables
  mxPath(from = manifests, arrows=2,free=T,values=1, labels=paste("e",1:11,sep="")),
  # set latent factor variances
  mxPath(from=latents2,arrows=2,free=F,values=1,labels=c("p1","p2")),
  # specify factor loadings
  # Allow the factors to covary as per the lavaan model
  mxPath(from="verbal", to="perf", arrows=2,
         free=T, values=1, labels="latcov1"),
  # Specify the latent factor paths to manifests
  mxPath(from="verbal", to=verbalVars, arrows=1, free=T),
  mxPath(from="perf", to=perfVars, arrows=1, free=T),
  mxData( observed=observedCov, type="cov", numObs=numSubjects)
)
```

5.3.3 Now use ‘mxRun’ to fit the model

mxRun uses the model defined above

```r
cfa2c <- mxRun(cfa2c)
```

## Running TwoFac, Comp Common with 24 parameters

Summarize the fit

```r
summary(cfa2c)
```

## Summary of TwoFac, Comp Common
## free parameters:

<table>
<thead>
<tr>
<th>name</th>
<th>matrix</th>
<th>row</th>
<th>col</th>
<th>Estimate</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwoFac, Comp</td>
<td>Common.A[1,12]</td>
<td>A</td>
<td>info verbal</td>
<td>2.25606</td>
</tr>
<tr>
<td>e1</td>
<td>S info</td>
<td>info</td>
<td>info</td>
<td>3.34302</td>
</tr>
<tr>
<td>e2</td>
<td>S comp</td>
<td>comp</td>
<td>comp</td>
<td>4.33109</td>
</tr>
<tr>
<td>e3</td>
<td>S arith</td>
<td>arith</td>
<td>arith</td>
<td>3.58375</td>
</tr>
<tr>
<td>e4</td>
<td>S simil</td>
<td>simil</td>
<td>simil</td>
<td>5.21667</td>
</tr>
<tr>
<td>e5</td>
<td>S vocab</td>
<td>vocab</td>
<td>vocab</td>
<td>3.38297</td>
</tr>
<tr>
<td>e6</td>
<td>S digit</td>
<td>digit</td>
<td>digit</td>
<td>6.11561</td>
</tr>
<tr>
<td>e7</td>
<td>S pictcomp</td>
<td>pictcomp</td>
<td>pictcomp</td>
<td>5.35770</td>
</tr>
<tr>
<td>e8</td>
<td>S parang</td>
<td>parang</td>
<td>parang</td>
<td>5.62019</td>
</tr>
<tr>
<td>e9</td>
<td>S block</td>
<td>block</td>
<td>block</td>
<td>3.97877</td>
</tr>
<tr>
<td>e10</td>
<td>S object</td>
<td>object</td>
<td>object</td>
<td>5.37584</td>
</tr>
<tr>
<td>e11</td>
<td>S coding</td>
<td>coding</td>
<td>coding</td>
<td>8.16174</td>
</tr>
<tr>
<td>latcov1</td>
<td>S verbal</td>
<td>verbal</td>
<td>perf</td>
<td>0.53322</td>
</tr>
</tbody>
</table>

## Std.Error A

<table>
<thead>
<tr>
<th>name</th>
<th>Std.Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>e1</td>
<td>0.200315</td>
</tr>
<tr>
<td>e2</td>
<td>0.256354</td>
</tr>
<tr>
<td>e3</td>
<td>0.173048</td>
</tr>
<tr>
<td>e4</td>
<td>0.227320</td>
</tr>
<tr>
<td>e5</td>
<td>0.200572</td>
</tr>
<tr>
<td>e6</td>
<td>0.212373</td>
</tr>
<tr>
<td>e7</td>
<td>0.270140</td>
</tr>
<tr>
<td>e8</td>
<td>0.242338</td>
</tr>
<tr>
<td>e9</td>
<td>0.225100</td>
</tr>
<tr>
<td>e10</td>
<td>0.221685</td>
</tr>
<tr>
<td>e11</td>
<td>0.233235</td>
</tr>
<tr>
<td>e12</td>
<td>0.255009</td>
</tr>
<tr>
<td>e13</td>
<td>0.509394</td>
</tr>
<tr>
<td>e14</td>
<td>0.557177</td>
</tr>
<tr>
<td>e15</td>
<td>0.422007</td>
</tr>
<tr>
<td>e16</td>
<td>0.682585</td>
</tr>
<tr>
<td>e17</td>
<td>0.507400</td>
</tr>
<tr>
<td>e18</td>
<td>0.677590</td>
</tr>
<tr>
<td>e19</td>
<td>0.755634</td>
</tr>
<tr>
<td>e20</td>
<td>0.665637</td>
</tr>
<tr>
<td>e21</td>
<td>0.636905</td>
</tr>
<tr>
<td>e22</td>
<td>0.708170</td>
</tr>
<tr>
<td>e23</td>
<td>0.874112</td>
</tr>
<tr>
<td>e24</td>
<td>0.082033</td>
</tr>
</tbody>
</table>

## Model Statistics:
### Third OpenMx Model

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Degrees of Freedom</th>
<th>Fit (-2lnL units)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model</td>
<td>24</td>
<td>42</td>
</tr>
<tr>
<td>Saturated</td>
<td>66</td>
<td>0</td>
</tr>
<tr>
<td>Independence</td>
<td>11</td>
<td>55</td>
</tr>
</tbody>
</table>

Number of observations/statistics: 175/66

chi-square: \( \chi^2 \) (df=42) = 60.61, p = 0.031396

Information Criteria:

<table>
<thead>
<tr>
<th>df Penalty</th>
<th>Parameters Penalty</th>
<th>Sample-Size Adjusted</th>
</tr>
</thead>
<tbody>
<tr>
<td>AIC: -23.39</td>
<td>108.61</td>
<td>116.61</td>
</tr>
<tr>
<td>BIC: -156.31</td>
<td>184.56</td>
<td>108.56</td>
</tr>
</tbody>
</table>

CFI: 0.95991

TLI: 0.9475 (also known as NNFI)

RMSEA: 0.050319 [95% CI (0, 0.081389)]

Prob(RMSEA <= 0.05): 0.4664


Wall clock time: 0.0488 secs

Optimizer: CSOLNP

OpenMx version number: 2.12.2

Need help? See help(mxSummary)

What is in the model?

```r
slotNames(cfa2c@output)
```

## NULL

```r
names(cfa2c@output)
```

## [1] "matrices"   "algebras"
## [3] "data"       "SaturatedLikelihood"
## [5] "IndependenceLikelihood" "calculatedHessian"
## [7] "standardErrors" "gradient"
## [9] "hessian"     "expectations"
## [11] "fit"         "fitUnits"
## [13] "Minus2LogLikelihood" "maxRelativeOrdinalError"
## [15] "minimum"    "estimate"
## [17] "infoDefine"  "conditionNumber"
## [19] "status"     "iterations"
## [21] "evaluations" "mxVersion"
## [23] "frontendTime" "backendTime"
## [25] "independentTime" "wallTime"
## [27] "timestamp"  "cpuTime"
5.3.4 Draw the OpenMx model number 3

The `semPlot` package is very powerful and can recognize many ‘lm’ and SEM model objects. We can use the identical code that we used in chapter 2 for the `lavaan` model.

```r
# Note that the base plot, including standardized path coefficients plots positive coefficients green
# and negative coefficients red. Red-green colorblindness issues anyone?
# I redrew it here to choose a blue and red. But all the coefficients in this example are
# positive, so they are shown with the skyblue.
# more challenging to use colors other than red and green. not in this doc
semPaths(cfa2c, residuals=F, sizeMan=7, "std",
        posCol=c("skyblue4", "red"),
        edge.color="skyblue4",
        edge.label.cex=1.2, layout="circle2")
```

# or we could draw the paths in such a way to include the residuals:
#semPaths(fit1, sizeMan=7,"std",edge.color="skyblue4",edge.label.cex=1,layout="circle2")
# the base path diagram can be drawn much more simply:
#semPaths(fit1)
# or
semPaths(cfa2c,"std")
5.4 Compare the OpenMx models

In OpenMx a convenient function exists for model comparisons. This code compares the same two models that we initially compared in the lavaan approach that used the anova function.

```r
kable(mxCompare(cfa2c, cfa2b), booktabs=TRUE, format="markdown")
```

<table>
<thead>
<tr>
<th>base</th>
<th>comparison</th>
<th>ep</th>
<th>minus2LL</th>
<th>df</th>
<th>AIC</th>
<th>diffLL</th>
<th>diffdf</th>
<th>p</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwoFac, Comp Common</td>
<td>NA</td>
<td>24</td>
<td>5435.7</td>
<td>42</td>
<td>-23.390</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>TwoFac, Comp Common</td>
<td>Two Factor Model</td>
<td>23</td>
<td>5445.7</td>
<td>43</td>
<td>-15.392</td>
<td>9.998</td>
<td>1</td>
<td>0.00157</td>
</tr>
</tbody>
</table>
Chapter 6

Summary and Reproducibility

We have finished a nice book........

Here is some information for for reproducibility:

```r
sessionInfo()
```

```r
## R version 3.5.2 (2018-12-20)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 7 x64 (build 7601) Service Pack 1
##
## Matrix products: default
##
## locale:
## [1] LC_COLLATE=English_United States.1252
## [2] LC_CTYPE=English_United States.1252
## [3] LC_MONETARY=English_United States.1252
## [4] LC_NUMERIC=C
## [5] LC_TIME=English_United States.1252
##
## attached base packages:
## [1] stats  graphics  grDevices  utils  datasets  methods  base
##
## other attached packages:
## [1] OpenMx_2.12.2   sem_3.1-9     lavaan_0.6-3   ggraph_1.0.2
## [5] ggplot2_3.1.0   corrplot_0.84 tidyr_0.8.2    magrittr_1.5
## [9] dplyr_0.8.0.1   MVN_5.6      kableExtra_1.0.1 knitr_1.21
##
## loaded via a namespace (and not attached):
## [1] readxl_1.3.0  backports_1.1.3 Hmisc_4.2-0
## [4] BDgraph_2.55  plyr_1.8.4   igraph_1.2.4
## [7] lazyeval_0.2.2 sp_1.3-1     splines_3.5.2
## [10] digest_0.6.18  htmltools_0.3.6 viridis_0.5.1
## [13] matrixcalc_1.0-3 checkmate_1.9.1 lsr_1ToF_0.1.4
## [16] cluster_2.0.7-1 openxlsx_4.1.0 readr_1.3.1
## [19] jpeg_0.1-8     colorspace_1.4-1 ggrepel_0.8.0
## [22] rvest_0.3.2    rrcov_1.4-7   haven_2.1.0
## [25] xfun_0.4       crayon_1.3.4  lme4_1.1-20
## [28] zoo_1.8-4      survival_2.43-3 glue_1.3.1
```
<table>
<thead>
<tr>
<th>Package</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>gtable_0.3.0</td>
<td>webshot_0.5.1</td>
</tr>
<tr>
<td>kernlab_0.9-27</td>
<td>prabclus_2.2-7</td>
</tr>
<tr>
<td>ggm_2.3</td>
<td>sROC_0.1-2</td>
</tr>
<tr>
<td>scales_1.0-0</td>
<td>xtable_1.8-3</td>
</tr>
<tr>
<td>GGalley_1.4-0</td>
<td>Formula_1.2-3</td>
</tr>
<tr>
<td>viridisLite_0.3.0</td>
<td>VCD_1.4-4</td>
</tr>
<tr>
<td>htmlTable_1.13.1</td>
<td>units_0.6-2</td>
</tr>
<tr>
<td>mlc_5.4.2</td>
<td>pwc_2.1-11.1</td>
</tr>
<tr>
<td>Kernlab_0.9-27</td>
<td>acepack_1.4.1</td>
</tr>
<tr>
<td>NADA_1.6-1</td>
<td>flexmix_2.3-15</td>
</tr>
<tr>
<td>reshape_0.8.8</td>
<td>XML_3.98-1.17</td>
</tr>
<tr>
<td>kutils_1.60</td>
<td>tidyselect_0.2.5</td>
</tr>
<tr>
<td>reshape2_1.4.3</td>
<td>munsell_0.5.0</td>
</tr>
<tr>
<td>pls_2.7-0</td>
<td>moments_0.14</td>
</tr>
<tr>
<td>evaluate_0.13</td>
<td>stringr_1.4.0</td>
</tr>
<tr>
<td>yam_2.2.0</td>
<td>zip_1.0.0</td>
</tr>
<tr>
<td>purrr_0.3.2</td>
<td>glasso_1.10</td>
</tr>
<tr>
<td>nle_3.1-137</td>
<td>whisker_0.3-2</td>
</tr>
<tr>
<td>compiler_3.5.2</td>
<td>rstudioapi_0.9.0</td>
</tr>
<tr>
<td>png_0.1-7</td>
<td>e1071_1.7-0.1</td>
</tr>
<tr>
<td>huge_1.2.7</td>
<td>tweenr_1.0.1</td>
</tr>
<tr>
<td>robCompositions_2.0.10</td>
<td>pbivnorm_0.6.0</td>
</tr>
<tr>
<td>stringi_1.3.1</td>
<td>highr_0.7</td>
</tr>
<tr>
<td>rockchalk_1.8.140</td>
<td>forcats_0.4.0</td>
</tr>
<tr>
<td>lattice_0.20-38</td>
<td>Matrix_1.2-15</td>
</tr>
<tr>
<td>pillar_1.3.1</td>
<td>lmtest_0.9-36</td>
</tr>
<tr>
<td>corpor_1.6.9</td>
<td>R6_2.4.0</td>
</tr>
<tr>
<td>gridExtra_2.3</td>
<td>gridExtra_2.3</td>
</tr>
<tr>
<td>boot_1.3-20</td>
<td>energy_1.7-5</td>
</tr>
<tr>
<td>tools_3.5.2</td>
<td>highr_0.7</td>
</tr>
<tr>
<td>reproc_1.1.0</td>
<td>highr_0.7</td>
</tr>
<tr>
<td>robCompositions_2.0.10</td>
<td>pbivnorm_0.6.0</td>
</tr>
<tr>
<td>assertthat_0.2.1</td>
<td>assertthat_0.2.1</td>
</tr>
<tr>
<td>withr_2.1.2</td>
<td>nortest_1.0-4</td>
</tr>
<tr>
<td>diptest_0.75-7</td>
<td>parallel_3.5.2</td>
</tr>
<tr>
<td>grid_3.5.2</td>
<td>rpart_4.1-13</td>
</tr>
<tr>
<td>class_7.3-15</td>
<td>minqa_1.2.4</td>
</tr>
<tr>
<td>mvoutlier_2.0.9</td>
<td>d3Network_0.5.2.1</td>
</tr>
<tr>
<td>numDeriv_2016.8-1</td>
<td>semTools_0.5-1</td>
</tr>
<tr>
<td>yaml_2.2.0</td>
<td>zCompositions_1.2.0</td>
</tr>
<tr>
<td>zip_1.0.0</td>
<td>pcaPP_1.9-73</td>
</tr>
<tr>
<td>stringi_1.3.1</td>
<td>highr_0.7</td>
</tr>
<tr>
<td>boot_1.3-20</td>
<td>highr_0.7</td>
</tr>
<tr>
<td>assertthat_0.2.1</td>
<td>assertthat_0.2.1</td>
</tr>
<tr>
<td>withr_2.1.2</td>
<td>nortest_1.0-4</td>
</tr>
<tr>
<td>diptest_0.75-7</td>
<td>parallel_3.5.2</td>
</tr>
<tr>
<td>grid_3.5.2</td>
<td>rpart_4.1-13</td>
</tr>
<tr>
<td>class_7.3-15</td>
<td>minqa_1.2.4</td>
</tr>
<tr>
<td>mvoutlier_2.0.9</td>
<td>d3Network_0.5.2.1</td>
</tr>
<tr>
<td>numDeriv_2016.8-1</td>
<td>semTools_0.5-1</td>
</tr>
<tr>
<td>base64enc_0.1-3</td>
<td>base64enc_0.1-3</td>
</tr>
</tbody>
</table>
Bibliography


