```
---
Signif. codes:  0 âĂŸ***âĂŹ 0.001 âĂŸ**âĂŹ 0.01 âĂŸ*âĂŹ 0.05 âĂŸ.âĂŹ 0.1 âĂŸ âĂŹ 1
```

```
> print(model.tables(recall.aov, "means"), digits = 3)
```

```
Tables of means
Grand mean

18.53704

 time
time
 long short
18.43 18.65

 study
study
  d45   d90
17.30 19.78

 time:study
       study
time    d45   d90
  long  16.37 20.48
  short 18.22 19.07
```

8.5 Multi-level models as an alternative to repeated measures ANOVA

Suppose we measure anger within subjects over four weeks.

9 Day 4: Multivariate analysis

We will use several built in data sets. The first is the `Thurstone` data set found in the `bifactor` data set.

9.1 Factor analysis and Principal Components Analysis

```
> data(bifactor)
> colnames(Thurstone) <- c("Sentences", "Vocab", "S.comp", "F.letter", "4.letter", "Suffix"
```

```
+     "Series", "Pedi", "letters")
> round(Thurstone, 2)
```

	Sentences	Vocab	S.comp	F.letter	4.letter	Suffix	Series	Pedi	letters
Sentences	1.00	0.83	0.78	0.44	0.43	0.45	0.45	0.54	0.38
Vocabulary	0.83	1.00	0.78	0.49	0.46	0.49	0.43	0.54	0.36
Sent.Completion	0.78	0.78	1.00	0.46	0.42	0.44	0.40	0.53	0.36
First.Letters	0.44	0.49	0.46	1.00	0.67	0.59	0.38	0.35	0.42
4.Letter.Words	0.43	0.46	0.42	0.67	1.00	0.54	0.40	0.37	0.45
Suffixes	0.45	0.49	0.44	0.59	0.54	1.00	0.29	0.32	0.32
Letter.Series	0.45	0.43	0.40	0.38	0.40	0.29	1.00	0.56	0.60
Pedigrees	0.54	0.54	0.53	0.35	0.37	0.32	0.56	1.00	0.45
Letter.Group	0.38	0.36	0.36	0.42	0.45	0.32	0.60	0.45	1.00

There are many flavors of factor analysis. The primary FA function is the `factanal` function in the core R.

```
> f3 <- factanal(covmat = Thurstone, factors = 3, n.obs = 213)
> f3

Call:
factanal(factors = 3, covmat = Thurstone, n.obs = 213)

Uniquenesses:
Sentences     Vocab    S.comp  F.letter  4.letter    Suffix    Series      Pedi   letters
    0.175     0.165     0.268     0.268     0.372     0.504     0.282     0.496     0.473

Loadings:
               Factor1 Factor2 Factor3
Sentences       0.834   0.244   0.264
Vocabulary      0.827   0.318   0.223
Sent.Completion 0.775   0.284   0.227
First.Letters   0.228   0.792   0.230
4.Letter.Words  0.213   0.706   0.291
Suffixes        0.314   0.616   0.134
Letter.Series   0.232   0.179   0.795
Pedigrees       0.446   0.166   0.527
Letter.Group    0.154   0.311   0.638

               Factor1 Factor2 Factor3
SS loadings      2.454   1.902   1.642
Proportion Var   0.273   0.211   0.182
Cumulative Var   0.273   0.484   0.666
```

```
Test of the hypothesis that 3 factors are sufficient.
The chi square statistic is 2.82 on 12 degrees of freedom.
The p-value is 0.997
```

Compare this solution to a two factor solution. The chi^2 value is much better for the 3 factor solution.

```
> f2 <- factanal(covmat = Thurstone, factors = 2, n.obs = 213)
```

9.1.1 The number of factors/components problem

A fundamental question in both components and factor analysis is how many components or factors to extract? While it is clear that more factors will fit better than fewer factors, and that more components will always summarize the data better than fewer such an improvement in fit has a cost in parsimony. Henry Kaiser once said that "a solution to the number-of factors problem in factor analysis is easy", that he used to make up one every morning before breakfast. But the problem, of course is to find *the* solution, or at least a solution that others will regard quite highly not as the best" Horn and Engstrom (1979). There are at least eight procedures that have been suggested:

1) Extracting factors until the χ^2 of the residual matrix is not significant. Although statistically this makes sense, it is very sensitive to the number of subjects in the data set. That is, the more subjects analyzed, the more factors or components that will be extracted. The rule is also sensitive to departures from normality in the data as well as the assumption that residual error is random rather than systematic but small. χ^2 estimates are reported for the maximum likelihood solution done by the `factanal` function.

2) Extracting factors until the change in χ^2 from factor n to factor n+1 is not significant. The same arguments apply to this rule as the previous rule.

3) Extracting factors until the eigenvalues of the real data are less than the corresponding eigenvalues of a random data set of the same size (*parallel analysis)*(Humphreys and Montanelli, 1975; Montanelli and Humphreys, 1976). The `fa.parallel` function plots the eigenvalues for a principal components solution as well as a principal axis factor solution for a given data set as well as that of n (default value = 20) randomly generated parallel data sets (Figure 18). A typical application is shown for 24 mental ability tests discussed by Harman (1960, 1976), reported originally by Holzinger and Swineford, and available as the `Harman74.cor` data set. Parallel analysis is partially sensitive to sample size in that for large samples the eigenvalues of random factors will tend to be very small and thus the number of components or factors will tend to be more than other rules.

4) Plotting the magnitude of the successive eigenvalues and applying the *scree test* (a

```
> fa.parallel(Thurstone, n.obs = 213)
```

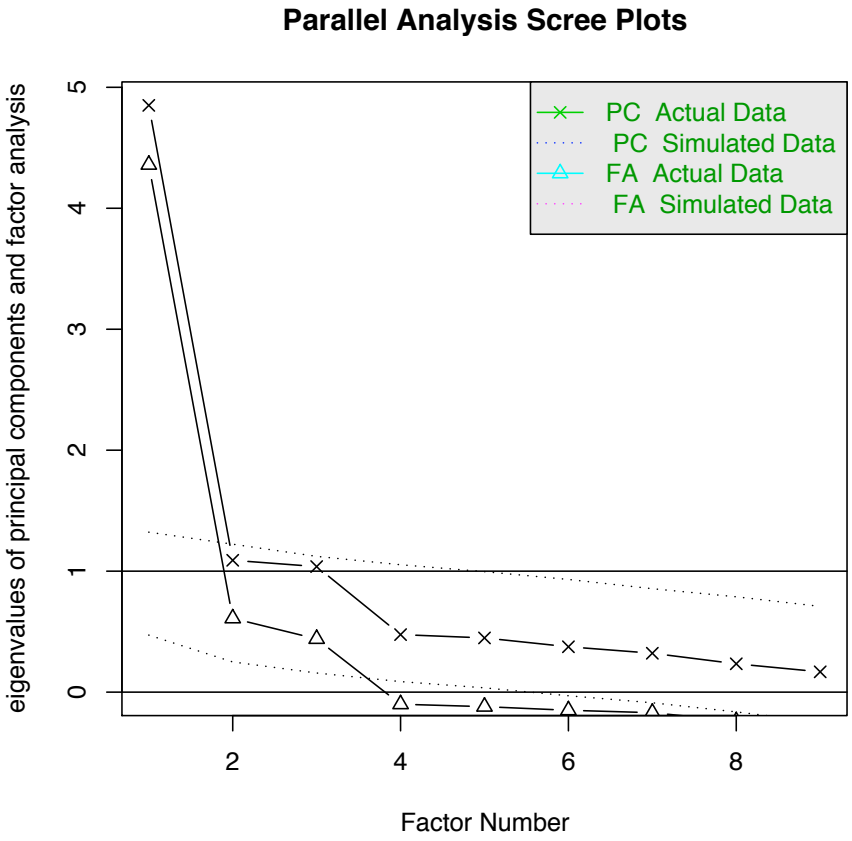

Figure 18: A parallel analysis of 9 mental ability tests is found by the `fa.parallel` function. The eigenvalues of the principal components solution of 20 random data sets suggests 1 components. A parallel solution for factor analysis suggests 3 factors.

sudden drop in eigenvalues analogous to the change in slope seen when scrambling up the talus slope of a mountain and approaching the rock face) (Cattell, 1966). In the example of the 9 mental tests case of Thurstone (Figure 18), a strong argument could be made for either one factor or three factors.

5) Extracting factors as long as they are interpretable. A surprisingly compelling rule for the number of factors or components to extract. This basically reflects common sense. The disadvantage is that investigators differ in their ability or desire to interpret factors. While some will find a two factor solution most interpretable, others will prefer five.

6) Using the *Very Simple Structure* Criterion (`VSS`), (Revelle and Rocklin, 1979) (Figure 19). Most people, when interpreting a factor solution, will highlight the large (salient) loadings and ignore the small loadings. That is, they are interpreting the factor matrix as if it had simple structure. How well does this simplified matrix reproduce the original matrix. That is, if c is the complexity (number of non zero loadings) of an item and max_c means the greatest (absolute) c loadings for an item, then find the *Very Simple Structure* matrix, $\vec{S}_c$, where

$$s_{cij} = \begin{pmatrix} f_{ij} & if(f_{ij} = max_c(fi.)) \\ 0 & otherwise \end{pmatrix}$$

Then let

$$\vec{R}^*_{sc} = R - \vec{SS}' \tag{1}$$

and

$$VSS_c = 1 - \frac{\vec{1}\vec{R}^{*\,2}_{sc}\vec{1}' - diag(\vec{R}^*_{sc})}{\vec{1}\vec{R}^2\vec{1}' - diag(\vec{R})}$$

That is, the VSS criterion for a complexity c is 1 - the squared residual correlations divided by the squared observed correlations, where the residuals are found by the "simplified" factor equation 1. Compare this result to those suggested by the *scree* test or the *parallel factors* tests seen in Figure 18. Unlike $\vec{R}^*$ as found in equation **??**, $\vec{R}^*_{sc}$ as found in equation 1 is sensitive to rotation and can also be used for evaluating alternative rotations. The *Very Simple Structure* criterion is implemented in the `VSS` and `VSS.plot` functions in the **psych** package.

7) Using the Minimum Average Partial criterion (MAP). Velicer (1976) proposed that the appropriate number of components to extract is that which minimizes the average (squared) partial correlation. Considering the $(n+p)x(n+p)$ super matrix composed of the *nxn* correlation matrix $\vec{R}$, the *nxp* component matrix $\vec{C}$, and the *pxp* covariance matrix of the components $\vec{CC}'$

$$\begin{pmatrix} \vec{R} & \vec{C}' \\ \vec{C} & C\vec{C}' \end{pmatrix} = \begin{pmatrix} \vec{R} & \vec{C}' \\ \vec{C} & \vec{I} \end{pmatrix}$$

```
> vss <- VSS(Thurstone, n.obs = 213, SMC = FALSE)
```

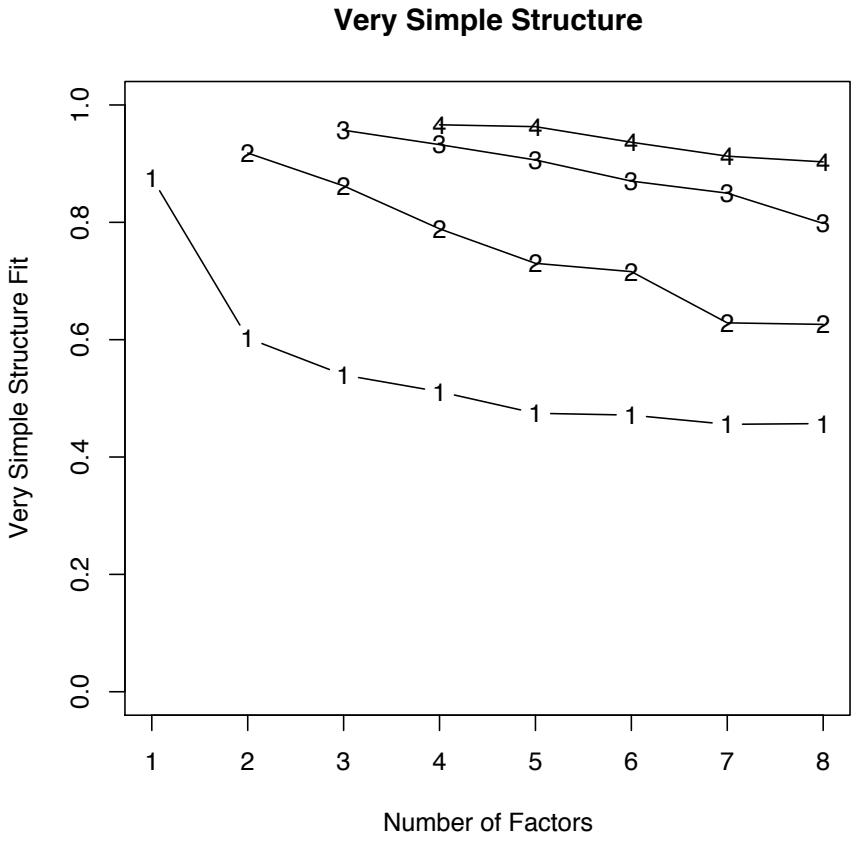

Figure 19: Very Simple Structure for 9 mental ability tests The complexity 1 solution suggests a large general factor but for complexity two or three, the best solution seems to be three factors.

then, partialling the components out of the correlation matrix produces a matrix of partial covariances, $\vec{R}^* = \vec{R} - \vec{C}\vec{C}'$ and the partial correlation matrix $\vec{R}^{\#}$ is found by dividing the partial covariances by their respective partial variances

$$\vec{R}^{\#} = \vec{D}^{-1/2}\vec{R}^*\vec{D}^{-1/2} \tag{2}$$

where $\vec{D} = diag(\vec{R}^*)$. The MAP criterion is just the sum of squared off diagonal elements of $\vec{R}^{\#}$. The logic of the MAP criterion is that although the residual correlations in $\vec{R}^*$ will become smaller as more factors are extracted, so will residual variances ($diag(\vec{R}^*)$. The magnitude of the partial correlations will thus diminish and then increase again when too many components are extracted. `MAP` is implemented in **psych** as part of the `VSS` function. For the Harman data set, MAP is at a minimum at four components, for the two dimensional circumplex strucure, at two:

```
> vss
```

```
Very Simple Structure
Call: VSS(x = Thurstone, n.obs = 213, SMC = FALSE)
VSS complexity 1 achieves a maximimum of 0.88  with  1  factors
VSS complexity 2 achieves a maximimum of 0.92  with  2  factors

The Velicer MAP criterion achieves a minimum of 1  with  3  factors

Velicer MAP
[1] 0.07 0.07 0.07 0.11 0.20 0.31 0.59 1.00

Very Simple Structure Complexity 1
[1] 0.88 0.60 0.54 0.51 0.47 0.47 0.46 0.46

Very Simple Structure Complexity 2
[1] 0.00 0.92 0.86 0.79 0.73 0.72 0.63 0.63
```

```
> round(vss$vss.stats[, 1:3], 2)
```

```
  dof  chisq prob
1  27 255.28 0.00
2  19  86.65 0.00
3  12   3.10 0.99
4   6   1.58 0.95
5   1   1.99 0.16
6  -3   1.15   NA
7  -6   0.01   NA
8  -8   0.00   NA
```

8) Extracting principal components until the eigenvalue <1 (Kaiser, 1970). This is probably the most used and most disparaged rule for the number of components. The logic behind it is that a component should be at least as large as a single variable. Unfortunately, in practice the $\lambda > 1$ rule seems to be a very robust estimate of the number of variables divided by three (Revelle and Rocklin, 1979; Velicer and Jackson, 1990).

Each of the procedures has its advantages and disadvantages. Using either the χ^2 test or the change in χ^2 test is, of course, sensitive to the number of subjects and leads to the nonsensical condition that if one wants to find many factors, one simlpy runs more subjects. The scree test is quite appealling but can lead to differences of interpretation as to when the scree "breaks". Extracting interpretable factors means that the number of factors reflects the investigators creativity more than the data. *VSS*, while very simple to understand, will not work very well if the data are very factorially complex. (Simulations suggests it will work fine if the complexities of some of the items are no more than 2). The eigenvalue of 1 rule, although the default for many programs, is fairly insensitive to the correct number and suggests that the number of factors is roughly 1/3 of the number of variables (Revelle and Rocklin, 1979; Velicer and Jackson, 1990). It is the least recommended of the procedures. MAP and VSS have the advantage that simulations show that they achieve a minimum (MAP) or maximum (VSS) at the correct number of components or factors. The number of factors problem was aptly summarized by Clyde Coombs who would say that determining the number of factors was like saying how many clouds were in the sky. On a clear day, it was easy, but when it was overcast, the problem became much more complicated. That the number of factors problem is important and that the standard default option in commercial packages such as SPSS is inappropriate has been frequently bemoaned Preacher and MacCallum (2003) and remains a challenge to the interpretation of many factor analyses.

9.2 Principal Axis factor analyis

An alternative to maximum likelihood factor analysis is principal axis factor analysis. This is just an interated eigen value decomposition replacing the diagonals of the correlation matrix with successively estimated communalities. It is sometimes a better solution, particularly in the case when the factor residuals are not normally distributed.

```
> pa3 <- factor.pa(Thurstone, 3)
> pa3
```

```
          V  PA1  PA2  PA3
Sentences 1 0.83
Vocab     2 0.83 0.32
S.comp    3 0.78
```

```
F.letter  4      0.79
4.letter  5      0.71
Suffix    6 0.31 0.62
Series    7           0.79
Pedi      8 0.45      0.53
letters   9      0.31 0.64

                PA1  PA2  PA3
SS loadings    2.46 1.91 1.64
Proportion Var 0.27 0.21 0.18
Cumulative Var 0.27 0.49 0.67

Test of the hypothesis that 3 factors are sufficient.

The degrees of freedom for the model is 12  and the fit was  0.01
```

9.3 Principal components analysis

Typically we do not want to exactly reproduce the original $n * n$ correlation matrix, for there is no gain in parsimony (the rank of the matrix is the same as the rank of the original matrix) but rather want to approximate it with a matrix of lower rank ($k < n$). This may be done by using just the first k principal components. This requires selecting and rescaling the first k components returned by the functions `princomp` and `prcomp` (Anderson, 1963). Alternatively, the `principal` function will provide the first k components scaled appropriately. `principal` returns an object of class `factanal` to be compatible with the output of that and other factor analysis functions.

Consider just the first principal component of the matrix (Table **??**). The *loadings* matrix shows the correlations of each variable with the component. The *uniquenesses*, a concept from factor analysis, reflect the variance not explained for each variable. As is seen in Table **??**, just one component does not reproduce the matrix very well, for it overestimates the correlations and underestimates the elements on the diagonal. The components solution, in attempting to account for the entire matrix, underestimates the importance of the major variables, and overestimates the importance of the least important variables. This is due to the influence of the diagonal elements of the matrix which are also being fitted. This is most clearly seen by examining the residual matrix of the difference between and the model of which is the product of the first principal component with its transpose. Increasing the number of components used will provide a progressively better approximation to the original matrix, but at a cost of a reduction in parsimony.

If the goal is simple and parsimonious description of a correlation or covariance matrix,

the first k principal components will do a better job than any other k-dimensional solution.

Although the PCA and factor analysis solutions look very similar, they are in fact, very different models.

```
> pc3 <- principal(Thurstone, 3)
> pc3
```

```
                V  PC1  PC2  PC3
Sentences       1 0.86
Vocabulary      2 0.85 0.31
Sent.Completion 3 0.85
First.Letters   4      0.82
4.Letter.Words  5      0.79 0.30
Suffixes        6 0.31 0.77
Letter.Series   7           0.83
Pedigrees       8 0.53      0.61
Letter.Group    9      0.31 0.80

                PC1  PC2  PC3
SS loadings    2.72 2.24 1.97
Proportion Var 0.30 0.25 0.22
Cumulative Var 0.30 0.55 0.77

Test of the hypothesis that 3 factors are sufficient.

The degrees of freedom for the model is 12  and the fit was  0.62
```

9.4 Comparing factor solutions

To compare factor solutions, we may use the `factor.congruence` function. This finds the cosines of the angles between the factors. Unlike a correlation, it does not remove the mean value of the loadings.

```
> factor.congruence(list(f3, pa3, pc3))
```

```
        Factor1 Factor2 Factor3  PA1  PA2  PA3  PC1  PC2  PC3
Factor1    1.00    0.64    0.62 1.00 0.64 0.62 1.00 0.59 0.55
Factor2    0.64    1.00    0.62 0.63 1.00 0.62 0.61 0.99 0.57
Factor3    0.62    0.62    1.00 0.62 0.62 1.00 0.61 0.56 0.99
PA1        1.00    0.63    0.62 1.00 0.64 0.62 1.00 0.58 0.55
```

```
PA2       0.64    1.00    0.62 0.64 1.00 0.62 0.61 0.99 0.57
PA3       0.62    0.62    1.00 0.62 0.62 1.00 0.61 0.56 0.99
PC1       1.00    0.61    0.61 1.00 0.61 0.61 1.00 0.56 0.54
PC2       0.59    0.99    0.56 0.58 0.99 0.56 0.56 1.00 0.51
PC3       0.55    0.57    0.99 0.55 0.57 0.99 0.54 0.51 1.00
```

9.5 Cluster Analysis, Multidimensional Scaling

9.6 Structural Equation Modeling

The following is adapted from the vignette *psych for sem*.

Although the exploratory models shown above do estimate the goodness of fit of the model and compare the residual matrix to a zero matrix using a χ^2 statistic, they estimate more parameters than are necessary if there is indeed a simple structure, and they do not allow for tests of competing models. The `sem` function in the *sem* package by John Fox allows for confirmatory tests. The interested reader is referred to the *sem* manual for more detail (Fox, 2009).

9.7 Using psych as a front end for the sem package

Because preparation of the `sem` commands is a bit tedious, several of the *psych* package functions have been designed to provide the appropriate commands. That is, the functions `structure.list`, `phi.list`, `structure.graph`, `structure.sem`, and `omega.graph` may be used as a front end to `sem`. Usually with no modification, but sometimes with just slight modification, the model output from the `structure.graph`, `structure.sem`, and `omega.graph` functions is meant to provide the appropriate commands for `sem`.

9.8 Testing a congeneric model versus a tau equivalent model

The congeneric model is a one factor model with possibly unequal factor loadings. The tau equivalent model model is one with equal factor loadings. Tests for these may be done by creating the appropriate structures. Either the `structure.graph` function which requires `Rgraphviz` or the `structure.sem` function which does not may be used.

The following example tests the hypothesis (which is actually false) that the correlations found in the cong data set (see **??**) are tau equivalent. Because the variable labels in that data set were V1 ... V4, we specify the labels to match those.

```
> library(sem)
> mod.tau <- structure.graph(c("a", "a", "a", "a"), labels = paste("V", 1:4, sep = ""))
> mod.tau

  Path    Parameter StartValue
1 X1->V1  a
2 X1->V2  a
3 X1->V3  a
4 X1->V4  a
5 V1<->V1 x1e
6 V2<->V2 x2e
7 V3<->V3 x3e
8 V4<->V4 x4e
9 X1<->X1 <fixed>   1

> sem.tau <- sem(mod.tau, cong, 100)
> summary(sem.tau, digits = 2)

 Model Chisquare =  12   Df =  5 Pr(>Chisq) = 0.037
 Chisquare (null model) =  84   Df =  6
 Goodness-of-fit index =  0.94
 Adjusted goodness-of-fit index =  0.88
 RMSEA index =  0.12   90% CI: (0.027, 0.21)
 Bentler-Bonnett NFI =  0.86
 Tucker-Lewis NNFI =  0.9
 Bentler CFI =  0.91
 SRMR =  0.11
 BIC =  -11

 Normalized Residuals
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 -1.040  -0.760  -0.521  -0.150   0.064   2.000

 Parameter Estimates
    Estimate Std Error z value Pr(>|z|)
a   0.64     0.064     10.0    0.0e+00  V1 <--- X1
x1e 0.51     0.095      5.3    9.3e-08  V1 <--> V1
x2e 0.50     0.095      5.3    9.9e-08  V2 <--> V2
x3e 0.67     0.115      5.8    5.0e-09  V3 <--> V3
x4e 0.75     0.126      5.9    3.0e-09  V4 <--> V4

 Iterations =  9
```

Test whether the data are congeneric. That is, whether a one factor model fits. Compare this to the prior model using the anova function.

```
> mod.cong <- structure.sem(c("a", "b", "c", "d"), labels = paste("V", 1:4, sep = ""))
> mod.cong

  Path    Parameter StartValue
1 X1->V1  a
2 X1->V2  b
3 X1->V3  c
4 X1->V4  d
5 V1<->V1 x1e
6 V2<->V2 x2e
7 V3<->V3 x3e
8 V4<->V4 x4e
9 X1<->X1 <fixed>   1

> cong <- sim.congeneric(N = 100)
> sem.cong <- sem(mod.cong, cong, 100)
> summary(sem.cong, digits = 2)

 Model Chisquare =  2.7   Df =  2 Pr(>Chisq) = 0.26
 Chisquare (null model) =  84   Df =  6
 Goodness-of-fit index =  0.99
 Adjusted goodness-of-fit index =  0.93
 RMSEA index =  0.061   90% CI: (NA, 0.22)
 Bentler-Bonnett NFI =  0.97
 Tucker-Lewis NNFI =  0.97
 Bentler CFI =  1
 SRMR =  0.037
 BIC =  -6.5

 Normalized Residuals
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 -0.373  -0.075   0.017   0.062   0.055   1.010

 Parameter Estimates
    Estimate Std Error z value Pr(>|z|)
a   0.77     0.10      7.4     1.0e-13  V1 <--- X1
b   0.78     0.10      7.5     5.2e-14  V2 <--- X1
c   0.50     0.11      4.7     2.7e-06  V3 <--- X1
d   0.43     0.11      3.9     8.9e-05  V4 <--- X1
x1e 0.40     0.11      3.7     2.4e-04  V1 <--> V1
```

```
x2e 0.39     0.11      3.5     4.5e-04  V2 <--> V2
x3e 0.75     0.12      6.3     2.8e-10  V3 <--> V3
x4e 0.82     0.12      6.5     5.8e-11  V4 <--> V4

 Iterations =  14
```

```
> anova(sem.cong, sem.tau)
```

```
LR Test for Difference Between Models

        Model Df Model Chisq Df LR Chisq Pr(>Chisq)
Model 1        2      2.7299
Model 2        5     11.8341  3   9.1041    0.02794 *
---
Signif. codes:  0 âĂŸ***âĂŹ 0.001 âĂŸ**âĂŹ 0.01 âĂŸ*âĂŹ 0.05 âĂŸ.âĂŹ 0.1 âĂŸ âĂŹ 1
```

The anova comparison of the congeneric versus tau equivalent model shows that the change in χ^2 is significant given the change in degrees of freedom.

9.9 Testing the dimensionality of a hierarchical data set by creating the model

The bifact correlation matrix was created to represent a hierarchical structure. Various confirmatory models can be applied to this matrix.

The first example creates the model directly, the next several create models based upon exploratory factor analyses. mod.one is a congeneric model of one factor accounting for the relationships between the nine variables. Although not correct, with 100 subjects, this model can not be rejected. However, an examination of the residuals suggests serious problems with the model.

```
> set.seed(42)
> gload = matrix(c(0.9, 0.8, 0.7), nrow = 3)
> fload <- matrix(c(0.9, 0.8, 0.7, rep(0, 9), 0.7, 0.6, 0.5, rep(0, 9), 0.6, 0.5, 0.4), nco
> fload
```

```
     [,1] [,2] [,3]
 [1,]  0.9  0.0  0.0
 [2,]  0.8  0.0  0.0
 [3,]  0.7  0.0  0.0
 [4,]  0.0  0.7  0.0
 [5,]  0.0  0.6  0.0
 [6,]  0.0  0.5  0.0
```

```
 [7,]  0.0  0.0  0.6
 [8,]  0.0  0.0  0.5
 [9,]  0.0  0.0  0.4
```

```
> bifact <- sim.hierarchical(gload = gload, fload = fload)
> round(bifact, 2)
```

```
     V1   V2   V3   V4   V5   V6   V7   V8   V9
V1 1.00 0.72 0.63 0.45 0.39 0.32 0.34 0.28 0.23
V2 0.72 1.00 0.56 0.40 0.35 0.29 0.30 0.25 0.20
V3 0.63 0.56 1.00 0.35 0.30 0.25 0.26 0.22 0.18
V4 0.45 0.40 0.35 1.00 0.42 0.35 0.24 0.20 0.16
V5 0.39 0.35 0.30 0.42 1.00 0.30 0.20 0.17 0.13
V6 0.32 0.29 0.25 0.35 0.30 1.00 0.17 0.14 0.11
V7 0.34 0.30 0.26 0.24 0.20 0.17 1.00 0.30 0.24
V8 0.28 0.25 0.22 0.20 0.17 0.14 0.30 1.00 0.20
V9 0.23 0.20 0.18 0.16 0.13 0.11 0.24 0.20 1.00
```

```
> mod.one <- structure.sem(letters[1:9], labels = paste("V", 1:9, sep = ""))
> mod.one
```

```
   Path    Parameter StartValue
1  X1->V1  a
2  X1->V2  b
3  X1->V3  c
4  X1->V4  d
5  X1->V5  e
6  X1->V6  f
7  X1->V7  g
8  X1->V8  h
9  X1->V9  i
10 V1<->V1 x1e
11 V2<->V2 x2e
12 V3<->V3 x3e
13 V4<->V4 x4e
14 V5<->V5 x5e
15 V6<->V6 x6e
16 V7<->V7 x7e
17 V8<->V8 x8e
18 V9<->V9 x9e
19 X1<->X1 <fixed>   1
```

```
> sem.one <- sem(mod.one, bifact, 100)
```

```
> summary(sem.one, digits = 2)

 Model Chisquare =  19   Df =  27 Pr(>Chisq) = 0.88
 Chisquare (null model) =  235   Df =  36
 Goodness-of-fit index =  0.96
 Adjusted goodness-of-fit index =  0.93
 RMSEA index =  0   90% CI: (NA, 0.040)
 Bentler-Bonnett NFI =  0.92
 Tucker-Lewis NNFI =  1.1
 Bentler CFI =  1
 SRMR =  0.053
 BIC =  -106

 Normalized Residuals
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
-2.7e-01 -1.8e-01 -1.4e-06  1.4e-01  1.2e-01  1.6e+00

 Parameter Estimates
    Estimate Std Error z value Pr(>|z|)
a   0.88     0.084     10.5    0.0e+00  V1 <--- X1
b   0.80     0.088      9.1    0.0e+00  V2 <--- X1
c   0.70     0.092      7.6    3.8e-14  V3 <--- X1
d   0.54     0.099      5.5    4.9e-08  V4 <--- X1
e   0.47     0.101      4.6    3.5e-06  V5 <--- X1
f   0.39     0.103      3.8    1.3e-04  V6 <--- X1
g   0.40     0.103      3.9    8.3e-05  V7 <--- X1
h   0.34     0.104      3.3    1.1e-03  V8 <--- X1
i   0.27     0.105      2.6    9.1e-03  V9 <--- X1
x1e 0.23     0.061      3.7    2.4e-04  V1 <--> V1
x2e 0.36     0.069      5.3    1.1e-07  V2 <--> V2
x3e 0.51     0.084      6.1    1.0e-09  V3 <--> V3
x4e 0.71     0.107      6.6    4.1e-11  V4 <--> V4
x5e 0.78     0.116      6.7    1.6e-11  V5 <--> V5
x6e 0.84     0.123      6.8    7.5e-12  V6 <--> V6
x7e 0.84     0.122      6.8    7.9e-12  V7 <--> V7
x8e 0.88     0.128      6.9    5.0e-12  V8 <--> V8
x9e 0.92     0.133      7.0    3.5e-12  V9 <--> V9

 Iterations =  14

> round(residuals(sem.one), 2)
```

```
      V1    V2    V3    V4    V5    V6    V7    V8    V9
V1  0.00  0.02  0.02 -0.02 -0.02 -0.02 -0.02 -0.02 -0.01
V2  0.02  0.00  0.00 -0.03 -0.03 -0.03 -0.02 -0.02 -0.02
V3  0.02  0.00  0.00 -0.02 -0.03 -0.02 -0.02 -0.02 -0.02
V4 -0.02 -0.03 -0.02  0.00  0.17  0.14  0.02  0.01  0.01
V5 -0.02 -0.03 -0.03  0.17  0.00  0.11  0.01  0.01  0.01
V6 -0.02 -0.03 -0.02  0.14  0.11  0.00  0.01  0.01  0.00
V7 -0.02 -0.02 -0.02  0.02  0.01  0.01  0.00  0.16  0.13
V8 -0.02 -0.02 -0.02  0.01  0.01  0.01  0.16  0.00  0.11
V9 -0.01 -0.02 -0.02  0.01  0.01  0.00  0.13  0.11  0.00
```

9.10 Testing the dimensionality based upon an exploratory analysis

Alternatively, the output from an exploratory factor analysis can be used as input to the structure.sem function.

```
> f1 <- factanal(covmat = bifact, factors = 1)
> mod.f1 <- structure.sem(f1)
> sem.f1 <- sem(mod.f1, bifact, 100)
> sem.f1

 Model Chisquare =  18.72871   Df =  27

       V1        V2        V3        V4        V5        V6        V7        V8        V9
0.8801449 0.7978613 0.6986695 0.5401625 0.4691098 0.3944311 0.4036073 0.3400459 0.2742160 0
      x3e       x4e       x5e       x6e       x7e       x8e       x9e
0.5118600 0.7082243 0.7799344 0.8444243 0.8371012 0.8843691 0.9248059

 Iterations =  14
```

The answers are, of course, identical.

9.11 Specifying a three factor model

An alternative model is to extract three factors and try this solution. The `factor.pa` factor analysis function is used to detect the structure. Alternatively, the `factanal` could have been used.

```
> f3 <- factor.pa(bifact, 3)
> mod.f3 <- structure.sem(f3)
```

```
> sem.f3 <- sem(mod.f3, bifact, 100)
> summary(sem.f3, digits = 2)

 Model Chisquare =  49   Df =  26 Pr(>Chisq) = 0.0037
 Chisquare (null model) =  235   Df =  36
 Goodness-of-fit index =  0.9
 Adjusted goodness-of-fit index =  0.82
 RMSEA index =  0.095   90% CI: (0.053, 0.14)
 Bentler-Bonnett NFI =  0.79
 Tucker-Lewis NNFI =  0.84
 Bentler CFI =  0.88
 SRMR =  0.20
 BIC =  -70

 Normalized Residuals
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
-2.0e-05  1.9e-05  1.8e+00  1.7e+00  2.6e+00  4.0e+00

 Parameter Estimates
     Estimate Std Error z value Pr(>|z|)
F1V1 0.79     0.094     8.4     0.0e+00  V1 <--- PA1
F2V1 0.23     0.089     2.6     1.0e-02  V1 <--- PA3
F1V2 0.80     0.093     8.6     0.0e+00  V2 <--- PA1
F1V3 0.70     0.095     7.4     1.7e-13  V3 <--- PA1
F2V4 0.70     0.129     5.4     6.1e-08  V4 <--- PA3
F2V5 0.60     0.124     4.8     1.2e-06  V5 <--- PA3
F2V6 0.50     0.120     4.2     3.1e-05  V6 <--- PA3
F3V7 0.60     0.190     3.2     1.5e-03  V7 <--- PA2
F3V8 0.50     0.167     3.0     2.8e-03  V8 <--- PA2
F3V9 0.40     0.147     2.7     6.5e-03  V9 <--- PA2
x1e  0.19     0.074     2.6     8.5e-03  V1 <--> V1
x2e  0.36     0.085     4.2     2.5e-05  V2 <--> V2
x3e  0.51     0.089     5.7     1.2e-08  V3 <--> V3
x4e  0.51     0.152     3.4     7.8e-04  V4 <--> V4
x5e  0.64     0.136     4.7     2.4e-06  V5 <--> V5
x6e  0.75     0.130     5.8     8.3e-09  V6 <--> V6
x7e  0.64     0.219     2.9     3.5e-03  V7 <--> V7
x8e  0.75     0.175     4.3     1.8e-05  V8 <--> V8
x9e  0.84     0.149     5.6     1.6e-08  V9 <--> V9

 Iterations =  34
```

```
> round(residuals(sem.f3), 2)
```

```
     V1   V2   V3   V4   V5   V6   V7   V8   V9
V1 0.13 0.09 0.08 0.29 0.25 0.21 0.34 0.28 0.23
V2 0.09 0.00 0.00 0.40 0.35 0.29 0.30 0.25 0.20
V3 0.08 0.00 0.00 0.35 0.30 0.25 0.26 0.22 0.18
V4 0.29 0.40 0.35 0.00 0.00 0.00 0.24 0.20 0.16
V5 0.25 0.35 0.30 0.00 0.00 0.00 0.20 0.17 0.13
V6 0.21 0.29 0.25 0.00 0.00 0.00 0.17 0.14 0.11
V7 0.34 0.30 0.26 0.24 0.20 0.17 0.00 0.00 0.00
V8 0.28 0.25 0.22 0.20 0.17 0.14 0.00 0.00 0.00
V9 0.23 0.20 0.18 0.16 0.13 0.11 0.00 0.00 0.00
```

The residuals show serious problems with this model. Although the residuals within each of the three factors are zero, the residuals between groups are much too large.

9.12 Allowing for an oblique solution

That solution is clearly very bad. What would happen if the exploratory solution were allowed to have correlated (oblique) factors? This analysis is done on a sample of size 100 with the bifactor structure created by `sim.hierarchical`.

```
> set.seed(42)
> bifact.s <- sim.hierarchical()
> f3 <- factor.pa(bifact.s, 3)
> f3.p <- Promax(f3)
> mod.f3p <- structure.sem(f3.p)
> mod.f3p
```

```
   Path      Parameter StartValue
1  PA1->V1   F1V1
2  PA1->V2   F1V2
3  PA1->V3   F1V3
4  PA3->V4   F2V4
5  PA3->V5   F2V5
6  PA3->V6   F2V6
7  PA2->V7   F3V7
8  PA2->V8   F3V8
9  PA2->V9   F3V9
10 V1<->V1   x1e
11 V2<->V2   x2e
12 V3<->V3   x3e
```

```
13 V4<->V4   x4e
14 V5<->V5   x5e
15 V6<->V6   x6e
16 V7<->V7   x7e
17 V8<->V8   x8e
18 V9<->V9   x9e
19 PA3<->PA1 rF2F1
20 PA2<->PA1 rF3F1
21 PA2<->PA3 rF3F2
22 PA1<->PA1 <fixed>   1
23 PA3<->PA3 <fixed>   1
24 PA2<->PA2 <fixed>   1
```

Unfortunately, the model as created automatically by `structure.sem` is not identified and would fail to converge if run. The problem is that the covariances between items on different factors is a product of the factor loadings and the between factor covariance. Multiplying the factor loadings by a constant can be compensated for by dividing the between factor covariances by the same constant. Thus, one of these paths must be fixed to provide a scale for the solution. That is, it is necessary to fix some of the paths to set values in order to properly identify the model. This can be done using the `edit` function and hand modification of particular paths. Set one path for each latent variable to be fixed.

e.g.,

mod.adjusted <- edit(mod.f3p)

Alternatively, the model can be adjusted by specifying the changes directly.

When this is done

```
> mod.f3p.adjusted <- mod.f3p
> mod.f3p.adjusted[c(1, 4), 2] <- NA
> mod.f3p.adjusted[c(1, 4), 3] <- "1"
> sem.f3p.adjusted <- sem(mod.f3p.adjusted, bifact.s, 100)
> summary(sem.f3p.adjusted, digits = 2)

 Model Chisquare =  7.1   Df =  26 Pr(>Chisq) = 1
 Chisquare (null model) =  235   Df =  36
 Goodness-of-fit index =  0.99
 Adjusted goodness-of-fit index =  0.98
 RMSEA index =  0   90% CI: (NA, NA)
 Bentler-Bonnett NFI =  0.97
 Tucker-Lewis NNFI =  1.1
 Bentler CFI =  1
```

```
 SRMR =  0.12
 BIC =  -113

 Normalized Residuals
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  -1.97   -0.79   -0.51   -0.67   -0.34   -0.11

 Parameter Estimates
      Estimate Std Error z value Pr(>|z|)
F1V2  0.87     0.089     9.8     0.0e+00  V2 <--- PA1
F1V3  0.76     0.094     8.1     6.7e-16  V3 <--- PA1
F2V5  0.65     0.124     5.2     1.7e-07  V5 <--- PA3
F2V6  0.55     0.126     4.4     1.4e-05  V6 <--- PA3
F3V7  0.63     0.134     4.7     2.5e-06  V7 <--- PA2
F3V8  0.52     0.131     4.0     5.9e-05  V8 <--- PA2
F3V9  0.42     0.131     3.2     1.3e-03  V9 <--- PA2
x1e   0.18     0.063     2.8     4.7e-03  V1 <--> V1
x2e   0.37     0.071     5.1     3.0e-07  V2 <--> V2
x3e   0.51     0.084     6.1     1.1e-09  V3 <--> V3
x4e   0.39     0.125     3.1     1.8e-03  V4 <--> V4
x5e   0.67     0.117     5.8     7.2e-09  V5 <--> V5
x6e   0.77     0.122     6.3     2.8e-10  V6 <--> V6
x7e   0.64     0.143     4.5     7.8e-06  V7 <--> V7
x8e   0.75     0.135     5.6     2.7e-08  V8 <--> V8
x9e   0.84     0.135     6.2     5.3e-10  V9 <--> V9
rF2F1 0.73     0.081     9.0     0.0e+00  PA1 <--> PA3
rF3F1 0.67     0.113     5.9     3.1e-09  PA1 <--> PA2
rF3F2 0.58     0.144     4.1     4.7e-05  PA3 <--> PA2

 Iterations =  21
```

The structure being tested may be seen using `structure.graph`

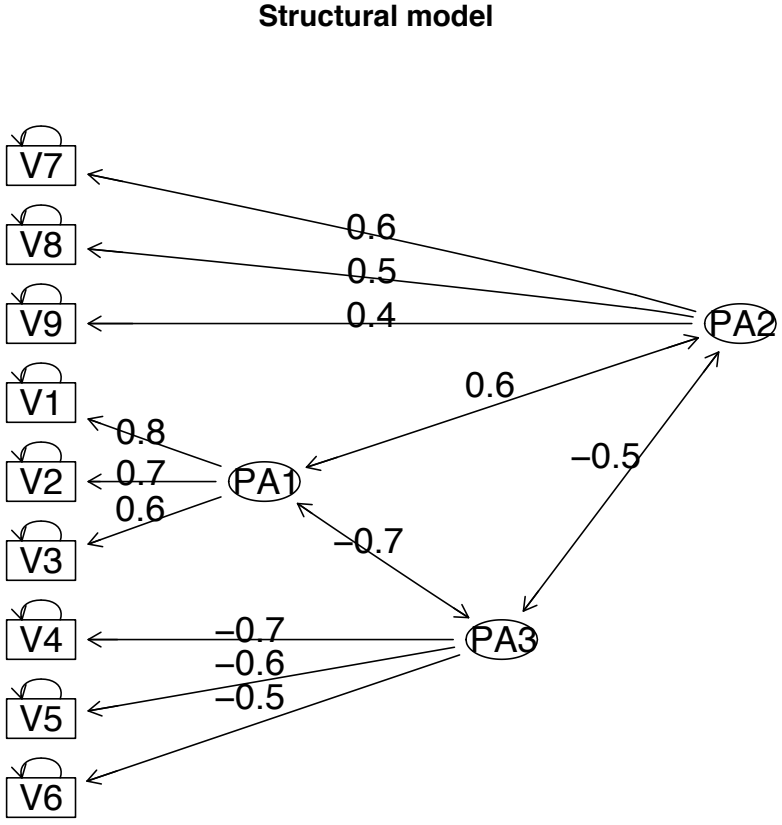

Figure 20: A three factor, oblique solution.

9.13 Scale reliability

9.14 Scoring an inventory

10 Day 5: R as a programming language

10.1 R in the lab

10.2 R in the classroom

10.3 Using R and Latex or OpenOffice to prepare documents

LaTeXis a text processing and formating language that can be combined with the `Sweave` function in R to integrate statistics within a manuscript. This is also possible to do with OpenOffice.

11 Various web resources

`http://www.rseek.org/`R seek: a search engine for R

`http://artsweb.uwaterloo.ca/~jalockli/R_exp_psy.pdf`A psychology graduate students learns R

Draft of April 1, 2009.