Using the psych package to generate and test structural models

William Revelle
Northwestern University

Contents

Written to accompany the psych package. Comments should be directed to William Revelle `revelle@northwestern.edu`

The psych package

Preface

The *psych* package Revelle (2009) has been developed to include those functions most useful for teaching and learning basic psychometrics and personality theory. Functions have been developed for many parts of the analysis of test data, including basic descriptive statistics (`describe` and `pairs.panels`), dimensionality analysis (`ICLUST`, `VSS`, `principal`, `factor.pa`), reliability analysis (`omega`, `guttman`) and eventual scale construction (`cluster.cor`, `score.items`). The use of these and other functions is described in more detail in the complete user's manual and the relevant help pages. This vignette is concerned with the problem of modeling structural data and using the *psych* package as a front end for the much more powerful *sem* package of John Fox Fox (2006, 2008).

Creating and modeling structural relations

One common application of `psych` is the creation of simulated data matrices with particular structures to use as examples for principal components analysis, factor analysis, cluster analysis, and structural equation modeling. This vignette describes some of the functions used for creating, analyzing, and displaying such data sets. The examples use two other packages: *Rgraphviz* and *sem*. Although not required to use the *psych* package, these two libraries are required for these examples. *Rgraphviz* is used for the graphical displays, but the analyses themselves require only the *sem* package to do the structural modeling.

Functions for generating correlational matrices with a particular structure

The `sim` family of functions create data sets with particular structure. Most of these functions have default values that will produce useful examples. Although graphical summaries of these structures will be shown here, some of the options of the graphical displays will be discussed in a later section.

sim.congeneric

Classical test theory considers tests to be *tau* equivalent if they have the same covariance with a vector of latent true scores, but perhaps different error variances. Tests are considered *congeneric* if they each have the same true score component (perhaps to a different degree) and independent error components. The `sim.congeneric` function may be used to generate either structure.

The first example considers four tests with equal loadings on a latent factor. If the number of subjects is not specified, a population correlation matrix will be generated. If N is specified, then the sample correlation matrix is returned. If the "short" option is FALSE, then the population matrix, sample matrix, and sample data are all returned as elements of a list.

```
> tau <- sim.congeneric(loads = c(0.8, 0.8, 0.8, 0.8))
> tau.samp <- sim.congeneric(loads = c(0.8, 0.8, 0.8, 0.8), N = 100)
> round(tau.samp, 2)
```

```
     V1   V2   V3   V4
V1 1.00 0.63 0.62 0.62
V2 0.63 1.00 0.64 0.69
V3 0.62 0.64 1.00 0.68
V4 0.62 0.69 0.68 1.00
```

```
> tau.samp <- sim.congeneric(loads = c(0.8, 0.8, 0.8, 0.8), N = 100, short = FALSE)
> tau.samp
```

```
 $model (Population correlation matrix)
     V1   V2   V3   V4
V1 1.00 0.64 0.64 0.64
V2 0.64 1.00 0.64 0.64
V3 0.64 0.64 1.00 0.64
V4 0.64 0.64 0.64 1.00

$r  (Sample correlation matrix  for sample size =  100 )
     V1   V2   V3   V4
V1 1.00 0.56 0.56 0.65
V2 0.56 1.00 0.55 0.65
V3 0.56 0.55 1.00 0.60
V4 0.65 0.65 0.60 1.00
```

```
> dim(tau.samp$observed)
```

```
[1] 100   4
```

In this last case, the generated data are retrieved from tau.samp$observed. Congeneric data are created by specifying unequal loading values. The default values are loadings of c(.8,.7,.6,.5). As seen in Figure 1, tau equivalence is the special case where all paths are equal.

```
> cong <- sim.congeneric(N = 100)
> round(cong, 2)
```

```
     V1   V2   V3   V4
V1 1.00 0.58 0.49 0.37
V2 0.58 1.00 0.49 0.35
V3 0.49 0.49 1.00 0.17
V4 0.37 0.35 0.17 1.00
```

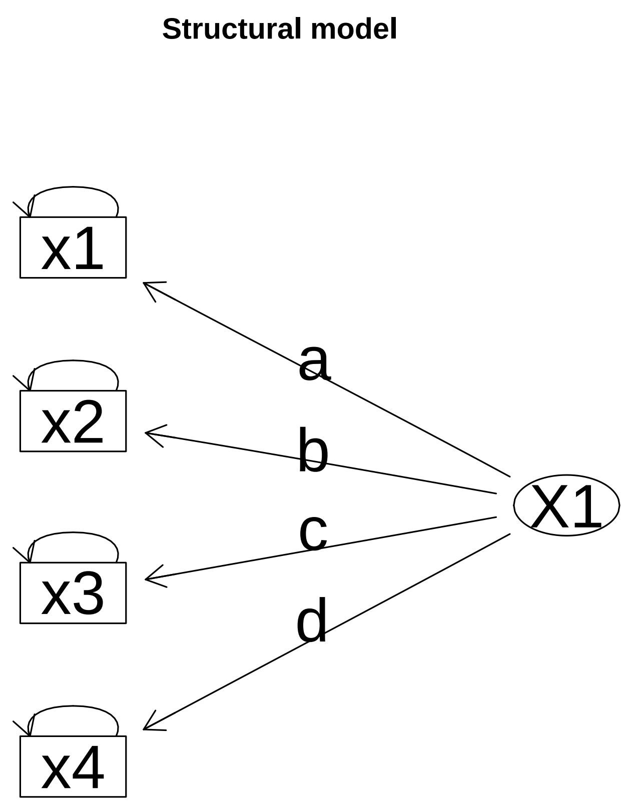

Figure 1. Tau equivalent tests are special cases of congeneric tests. Tau equivalence assumes a=b=c=d

sim.hierarchical

The previous function, `sim.congeneric`, is used when one factor accounts for the pattern of correlations. A slightly more complicated model is when one broad factor and several narrower factors are observed. An example of this structure might be the structure of mental abilities, where there is a broad factor of general ability and several narrower factors (e.g., spatial ability, verbal ability, working memory capacity). Another example is in the measure of psychopathology where a broad general factor of neuroticism is seen along with more specific anxiety, depression, and aggression factors. This kind of structure may be simulated with `sim.hierarchical` specifying the loadings of each sub factor on a general factor (the g-loadings) as well as the loadings of individual items on the lower order factors (the f-loadings). An early paper describing a *bifactor* structure was by Holzinger & Swineford (1937). A helpful description of what makes a good general factor is that of Jensen & Weng (1994). Three data sets are included in the `bifactor` data set. One is the original 14 variable problem of Holzinger & Swineford (1937) (holzinger), a second is a nine variable problem from Thurstone (used as an example in the SAS manual and discussed in great detail by McDonald (1999)), the third is from a recent paper by Reise et al. (2007) with 16 measures of patient reports of interactions with their health care provider.

```
> 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
> 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
```

These data can be represented as either a *bifactor* (Figure 2) or *hierarchical* (Figure 3) factor solution.

sim.item and sim.circ

Many personality questionnaires are thought to represent multiple, independent factors. A particularly interesting case is when there are two factors and the items either have *simple structure* or *circumplex structure*. Examples of such items with a circumplex structure are measures of emotion (Rafaeli & Revelle, 2006) where many different emotion terms can be arranged in a two dimensional space, but where there is no obvious clustering of items. Typical personality scales are constructed to have simple structure, where items load on one and only one factor.

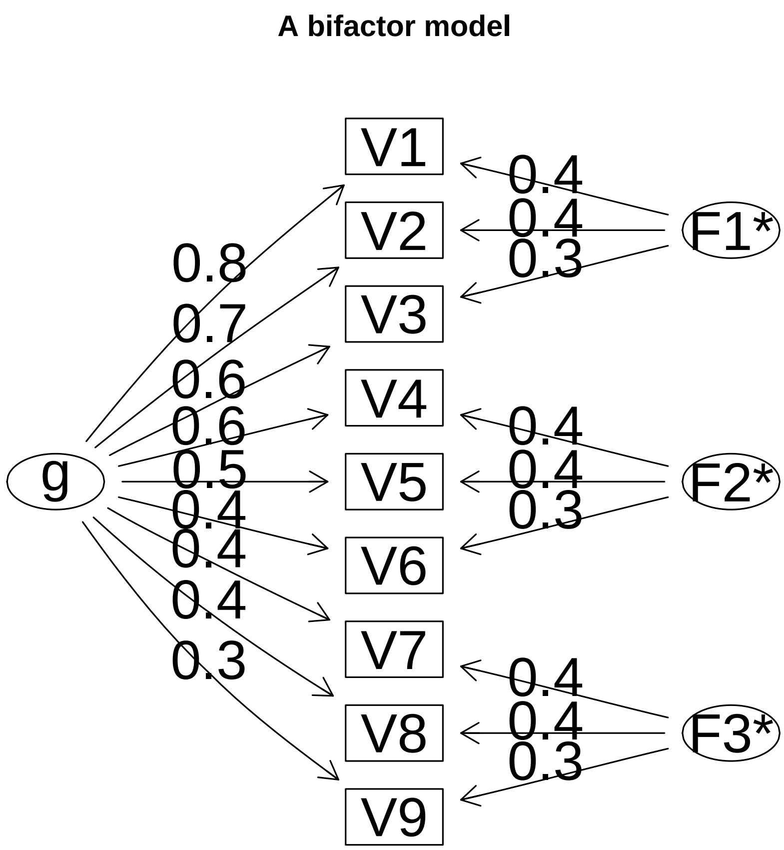

Figure 2. A bifactor solution represents each test in terms of a general factor and a residualized group factor.

An additional challenge to measurement with emotion or personality items is that the items can be highly skewed and are assessed with a small number of discrete categories (do not agree, somewhat agree, strongly agree).

The more general `sim.item` function, and the more specific, `sim.circ` functions simulate items with a two dimensional structure, with or without skew, and varying the number of categories for the items.

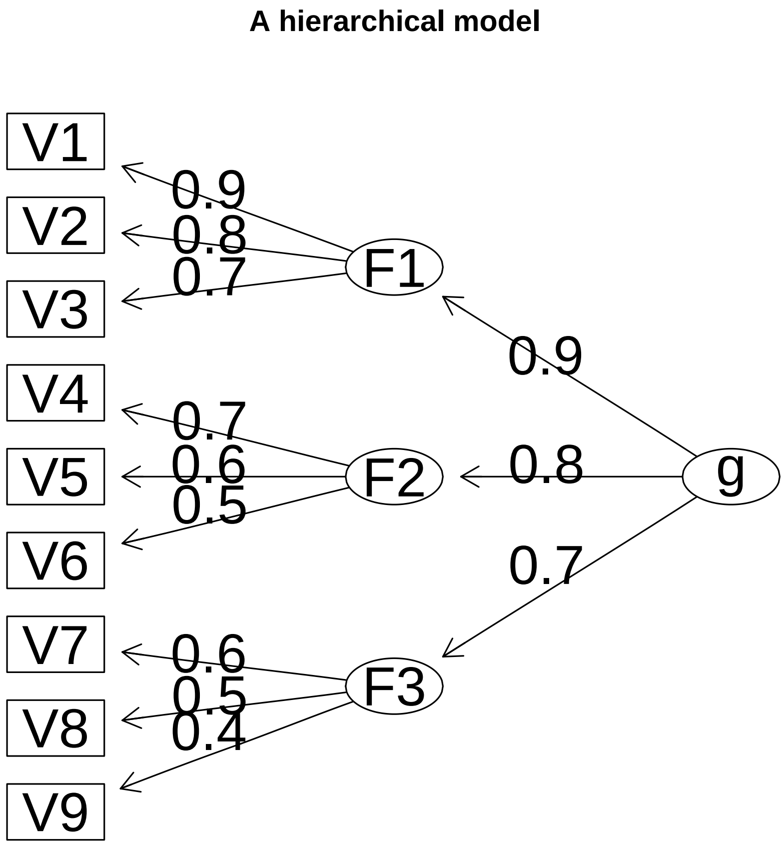

Figure 3. A hierarchical factor solution has g as a second order factor accounting for the correlations between the first order factors.

sim.structure

A more general case is to consider three matrices, $\vec{f}_x, \vec{\phi_{xy}}, \vec{f}_y$ which describe, in turn, a measurement model of x variables, $\vec{f}_x$, a measurement model of y variables, $\vec{f}_x$, and a covariance matrix between and within the two sets of factors. If $\vec{f}_x$ is a vector and $\vec{f}_y$ and $\vec{phi}_{xy}$ are NULL, then this is just the congeneric model. If $\vec{f}_x$ is a matrix of loadings with n rows and c columns, then this is a measurement model for n variables across c factors. If $\vec{phi}_{xy}$ is not null, but $\vec{f}_y$ is NULL, then the factors in $\vec{f}_x$ are correlated. Finally, if all three matrices are not NULL, then the data show the standard linear structural relations (LISREL) structure.

Consider the following examples:

$\vec{f}_x$ is a vector implies a congeneric model.

```
> fx <- c(0.9, 0.8, 0.7, 0.6)
> cong1 <- sim.structure(fx)
> cong1

 $model (Population correlation matrix)
     V1   V2   V3   V4
V1 1.00 0.72 0.63 0.54
V2 0.72 1.00 0.56 0.48
V3 0.63 0.56 1.00 0.42
V4 0.54 0.48 0.42 1.00

$reliability (population reliability)
[1] 0.81 0.64 0.49 0.36
```

$\vec{f}_x$ is a matrix implies an independent factors model:.

```
> fx <- 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), ncol =
> three.fact <- sim.structure(fx)
> three.fact

 $model (Population correlation matrix)
     V1   V2   V3   V4   V5   V6   V7  V8   V9
V1 1.00 0.72 0.63 0.00 0.00 0.00 0.00 0.0 0.00
V2 0.72 1.00 0.56 0.00 0.00 0.00 0.00 0.0 0.00
V3 0.63 0.56 1.00 0.00 0.00 0.00 0.00 0.0 0.00
V4 0.00 0.00 0.00 1.00 0.42 0.35 0.00 0.0 0.00
V5 0.00 0.00 0.00 0.42 1.00 0.30 0.00 0.0 0.00
V6 0.00 0.00 0.00 0.35 0.30 1.00 0.00 0.0 0.00
V7 0.00 0.00 0.00 0.00 0.00 0.00 1.00 0.3 0.24
V8 0.00 0.00 0.00 0.00 0.00 0.00 0.30 1.0 0.20
V9 0.00 0.00 0.00 0.00 0.00 0.00 0.24 0.2 1.00

$reliability (population reliability)
[1] 0.81 0.64 0.49 0.49 0.36 0.25 0.36 0.25 0.16
```

$\vec{f}_x$ is a matrix and $Phi \neq I$ is a correlated factors model.

```
> Phi = matrix(c(1, 0.5, 0.3, 0.5, 1, 0.2, 0.3, 0.2, 1), ncol = 3)
> cor.f3 <- sim.structure(fx, Phi)
```

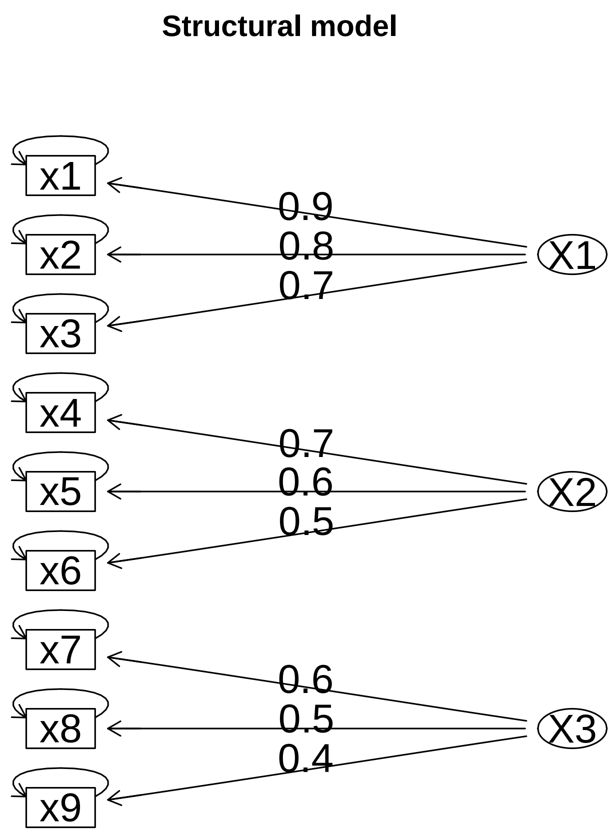

Figure 4. Three uncorrelated factors generated using the sim.structure function and drawn using structure.graph.

```
> fx

     [,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
```

```
> Phi
```

```
     [,1] [,2] [,3]
[1,]  1.0  0.5  0.3
[2,]  0.5  1.0  0.2
[3,]  0.3  0.2  1.0
```

```
> cor.f3
```

```
 $model (Population correlation matrix)
     V1    V2    V3    V4    V5   V6    V7   V8    V9
V1 1.00 0.720 0.630 0.315 0.270 0.23 0.162 0.14 0.108
V2 0.72 1.000 0.560 0.280 0.240 0.20 0.144 0.12 0.096
V3 0.63 0.560 1.000 0.245 0.210 0.17 0.126 0.10 0.084
V4 0.32 0.280 0.245 1.000 0.420 0.35 0.084 0.07 0.056
V5 0.27 0.240 0.210 0.420 1.000 0.30 0.072 0.06 0.048
V6 0.23 0.200 0.175 0.350 0.300 1.00 0.060 0.05 0.040
V7 0.16 0.144 0.126 0.084 0.072 0.06 1.000 0.30 0.240
V8 0.14 0.120 0.105 0.070 0.060 0.05 0.300 1.00 0.200
V9 0.11 0.096 0.084 0.056 0.048 0.04 0.240 0.20 1.000

$reliability (population reliability)
[1] 0.81 0.64 0.49 0.49 0.36 0.25 0.36 0.25 0.16
```

This can be shown with symbolic loadings and path coefficients by using the `structure.list` and `phi.list` functions to create the fx and Phi matrices (Figure 5).

```
> fxs <- structure.list(9, list(F1 = c(1, 2, 3), F2 = c(4, 5, 6), F3 = c(7, 8, 9)))
> Phis <- phi.list(3, list(F1 = c(2, 3), F2 = c(1, 3), F3 = c(1, 2)))
> fxs
```

```
      F1   F2   F3
 [1,] "a1" "0"  "0"
 [2,] "a2" "0"  "0"
 [3,] "a3" "0"  "0"
 [4,] "0"  "b4" "0"
 [5,] "0"  "b5" "0"
 [6,] "0"  "b6" "0"
 [7,] "0"  "0"  "c7"
 [8,] "0"  "0"  "c8"
 [9,] "0"  "0"  "c9"
```

```
> Phis
```

```
   F1    F2    F3
F1 "1"   "rba" "rca"
F2 "rab" "1"   "rcb"
F3 "rac" "rbc" "1"
```

The `structure.list` and `phi.list` functions allow for for creation of fx, Phi, and fy matrices in a very compact form, just by specifying the relevant variables.

```
> corf3.mod <- structure.graph(fxs, Phis)
```

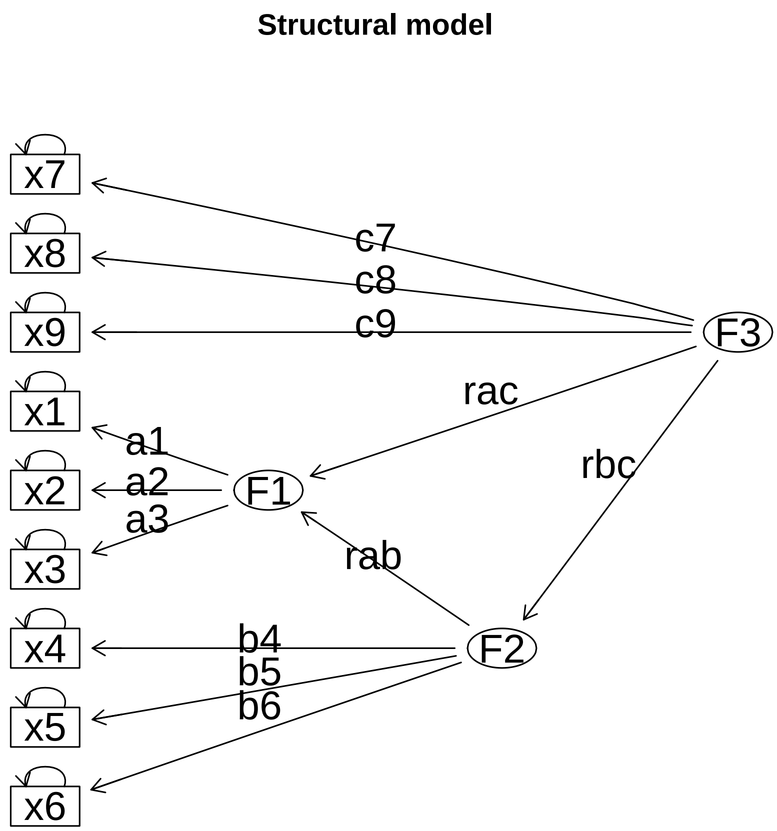

Figure 5. Three correlated factors with symbolic paths. Created using structure.graph and structure.list and phi.list for ease of input.

Alternatively, this result can represent the estimated factor loadings and oblique correlations found using `factanal` (Maximum Likelihood factoring) or `factor.pa` (Principal axis factoring) followed by a promax rotation using the `Promax` function (Figure 6. Comparing this figure with the previous one (Figure 5), it will be seen that one path was dropped because it was less than the arbitrary "cut" value of .2.

```
> f3.p <- Promax(factor.pa(cor.f3$model, 3))
```

```
> mod.f3p <- structure.graph(f3.p, cut = 0.2)
```

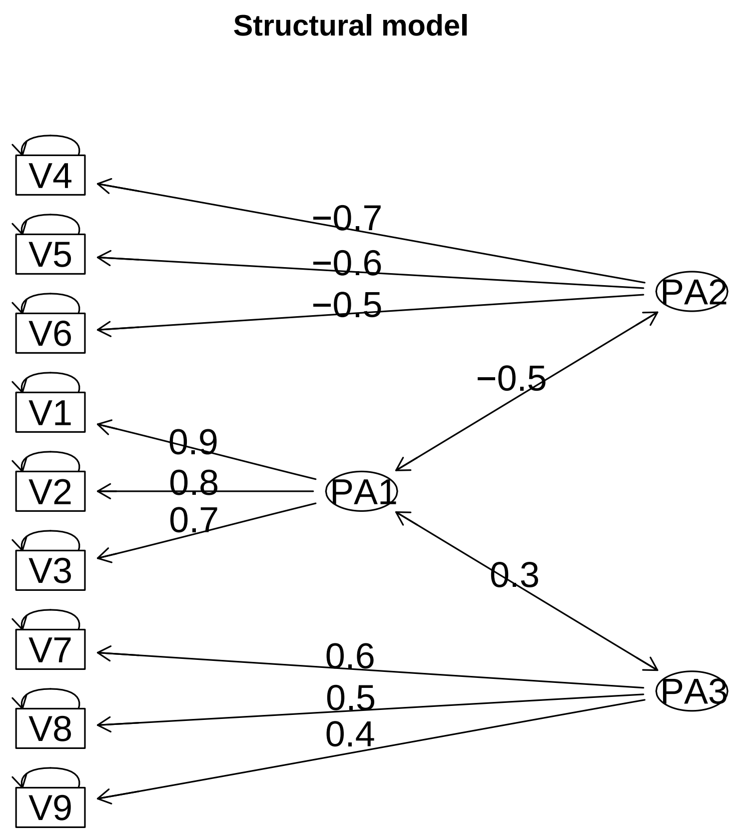

Figure 6. The empirically fitted structural model. Paths less than cut (.2 in this case, the default is .3) are not shown.

$\vec{f}_x$ and $\vec{f}_y$ are matrices, and $Phi \neq I$ represents their correlations. A more complicated model is when there is a $\vec{f}_y$ vector or matrix representing a set of Y latent variables that are associated with the a set of y variables. In this case, the Phi matrix is a set of correlations within the X set and between the X and Y set.

```
> fx <- 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), ncol =
> fy <- c(0.6, 0.5, 0.4)
```

```
> Phi <- matrix(c(1, 0.48, 0.32, 0.4, 0.48, 1, 0.32, 0.3, 0.32, 0.32, 1, 0.2, 0.4, 0.3, 0.2
+     ncol = 4)
> twelveV <- sim.structure(fx, Phi, fy)$model
> colnames(twelveV) <- rownames(twelveV) <- c(paste("x", 1:9, sep = ""), paste("y", 1:3, se
> round(twelveV, 2)
```

```
     x1   x2   x3   x4   x5   x6   x7   x8   x9   y1   y2   y3
x1 1.00 0.72 0.63 0.30 0.26 0.22 0.17 0.14 0.12 0.22 0.18 0.14
x2 0.72 1.00 0.56 0.27 0.23 0.19 0.15 0.13 0.10 0.19 0.16 0.13
x3 0.63 0.56 1.00 0.24 0.20 0.17 0.13 0.11 0.09 0.17 0.14 0.11
x4 0.30 0.27 0.24 1.00 0.42 0.35 0.13 0.11 0.09 0.13 0.10 0.08
x5 0.26 0.23 0.20 0.42 1.00 0.30 0.12 0.10 0.08 0.11 0.09 0.07
x6 0.22 0.19 0.17 0.35 0.30 1.00 0.10 0.08 0.06 0.09 0.08 0.06
x7 0.17 0.15 0.13 0.13 0.12 0.10 1.00 0.30 0.24 0.07 0.06 0.05
x8 0.14 0.13 0.11 0.11 0.10 0.08 0.30 1.00 0.20 0.06 0.05 0.04
x9 0.12 0.10 0.09 0.09 0.08 0.06 0.24 0.20 1.00 0.05 0.04 0.03
y1 0.22 0.19 0.17 0.13 0.11 0.09 0.07 0.06 0.05 1.00 0.30 0.24
y2 0.18 0.16 0.14 0.10 0.09 0.08 0.06 0.05 0.04 0.30 1.00 0.20
y3 0.14 0.13 0.11 0.08 0.07 0.06 0.05 0.04 0.03 0.24 0.20 1.00
```

Data with this structure may be created using the `sim.structure` function, and shown either with the numeric values or symbolically using the `structure.graph` function (Figure 7).

```
> fxs <- structure.list(9, list(X1 = c(1, 2, 3), X2 = c(4, 5, 6), X3 = c(7, 8, 9)))
> phi <- phi.list(4, list(F1 = c(4), F2 = c(4), F3 = c(4), F4 = c(1, 2, 3)))
> fyx <- structure.list(3, list(Y = c(1, 2, 3)), "Y")
```

A hierarchical structure among the latent predictors.. Measures of intelligence and psychopathology frequently have a general factor as well as multiple group factors. The general factor then is thought to predict some dependent latent variable. Compare this with the previous model (see Figure 7).

These two models can be compared using structural modeling procedures (see below).

Exploratory functions for analyzing structure

Given a correlation matrix such as seen above for congeneric or bifactor models, the question becomes how best to estimate the underlying structure. Because these data sets were generated from a known model, the question becomes how well does a particular model recover the underlying structure.

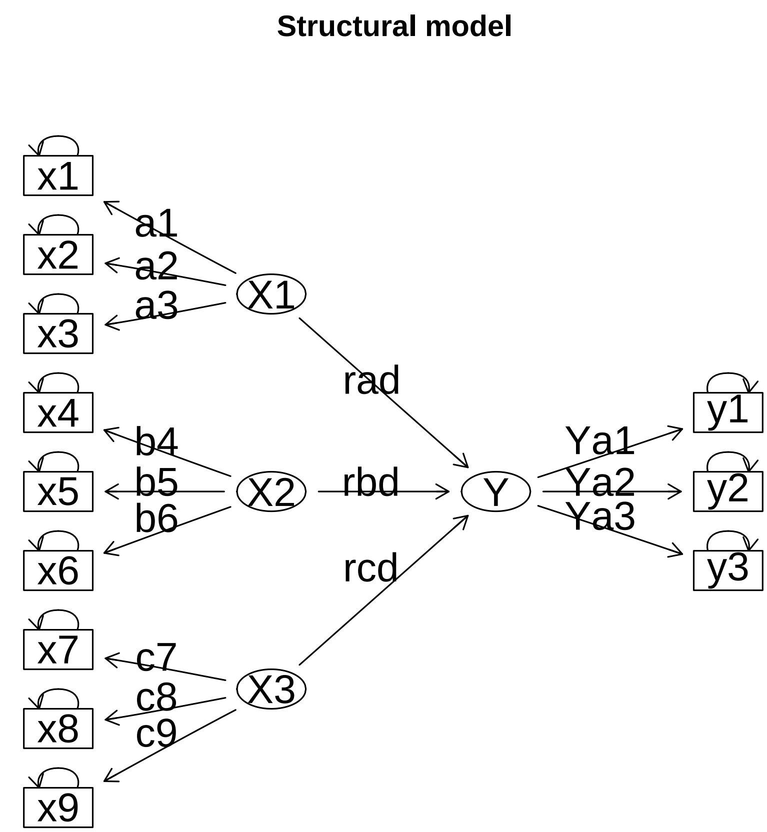

Figure 7. A symbolic structural model. Three independent latent variables are regressed on a latent Y.

Exploratory simple structure models

The technique of *principal components* provides a set of weighted linear composites that best aproximates a particular correlation or covariance matrix. If these are then *rotated* to provide a more interpretable solution, the components are no longer the *principal* components. The `principal` function will extract the first n principal components (default value is 1) and if n>1, rotate to *simple structure* using a `varimax`, `quartimin`, or `Promax` criterion.

```
> principal(cong1$model)
```

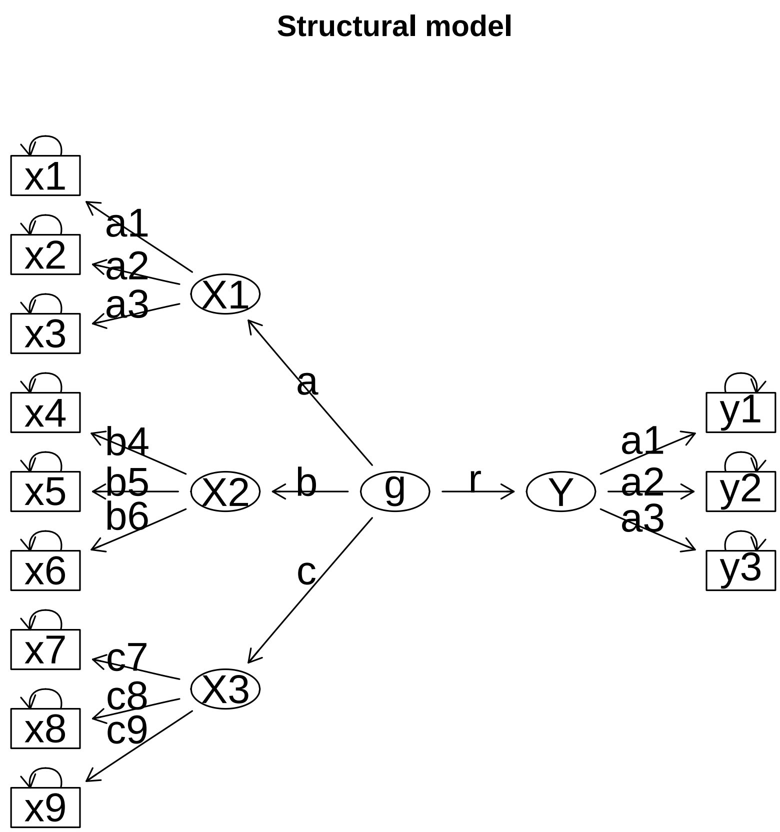

Figure 8. A symbolic structural model with a general factor and three group factors. The general factor is regressed on the latent Y variable.

```
  V  PA1
1 1 0.89
2 2 0.85
3 3 0.80
4 4 0.73

                PA1
SS loadings    2.69
Proportion Var 0.67

Test of the hypothesis that 1 factor is sufficient.
```

```
The degrees of freedom for the model is 2  and the fit was  0.14
```

```
> factor.pa(cong1$model)
```

```
  V PA1
1 1 0.9
2 2 0.8
3 3 0.7
4 4 0.6

                PA1
SS loadings    2.30
Proportion Var 0.58

Test of the hypothesis that 1 factor is sufficient.

The degrees of freedom for the model is 2  and the fit was  0
```

It is important to note that although the `principal` components function does not exactly reproduce the model parameters, the `factor.pa` function, implementing principal axes factor analysis, does.

Consider the case of three underlying factors as seen in the bifact example above. Because the number of observations is not specified, there is no associated χ^2 value. The `factor.congruence` function reports the cosine of the angle between the factors.

```
> pc3 <- principal(bifact, 3)
> pa3 <- factor.pa(bifact, 3)
> ml3 <- factanal(covmat = bifact, factors = 3)
> pc3
```

```
   V  PC1  PC3  PC2
V1 1 0.82
V2 2 0.82
V3 3 0.82
V4 4 0.32 0.68
V5 5      0.70
V6 6      0.77
V7 7           0.66
V8 8           0.68
V9 9           0.71

                PC1  PC3  PC2
```

```
SS loadings    2.26 1.73 1.53
Proportion Var 0.25 0.19 0.17
Cumulative Var 0.25 0.44 0.61

Test of the hypothesis that 3 factors are sufficient.

The degrees of freedom for the model is 12  and the fit was  0.71
```

```
> pa3
```

```
   V  PA1   PA3  PA2
V1 1 0.78 -0.34
V2 2 0.70 -0.30
V3 3 0.61
V4 4      -0.63
V5 5      -0.54
V6 6      -0.45
V7 7            0.55
V8 8            0.47
V9 9            0.37

                PA1  PA3  PA2
SS loadings    1.66 1.21 0.93
Proportion Var 0.18 0.13 0.10
Cumulative Var 0.18 0.32 0.42

Test of the hypothesis that 3 factors are sufficient.

The degrees of freedom for the model is 12  and the fit was  0
```

```
> ml3
```

```
Call:
factanal(factors = 3, covmat = bifact)

Uniquenesses:
  V1   V2   V3   V4   V5   V6   V7   V8   V9
0.19 0.36 0.51 0.51 0.64 0.75 0.64 0.75 0.84

Loadings:
   Factor1 Factor2 Factor3
V1 0.785   0.336   0.286
V2 0.697   0.298   0.254
```

```
V3 0.610   0.261   0.222
V4 0.242   0.631   0.182
V5 0.207   0.541   0.156
V6 0.173   0.451   0.130
V7 0.167   0.148   0.557
V8 0.139   0.124   0.464
V9 0.112           0.371

               Factor1 Factor2 Factor3
SS loadings      1.666   1.212   0.933
Proportion Var   0.185   0.135   0.104
Cumulative Var   0.185   0.320   0.423

The degrees of freedom for the model is 12 and the fit was 0
```

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

```
     PA1   PA3  PA2
PC1 0.99 -0.70 0.65
PC3 0.57 -0.96 0.51
PC2 0.45 -0.43 0.95
```

```
> factor.congruence(pa3, ml3)
```

```
    Factor1 Factor2 Factor3
PA1    1.00    0.72    0.67
PA3   -0.72   -1.00   -0.63
PA2    0.67    0.63    1.00
```

By default, all three of these procedures use the varimax rotation criterion. Perhaps it is useful to apply an oblique transformation such as `Promax` or `oblimin` to the results. The `Promax` function in *psych* differs slightly from the standard `promax` in that it reports the factor intercorrelations.

```
> ml3p <- Promax(ml3)
> ml3p
```

```
   V Factor1 Factor2 Factor3
V1 1  0.8329
V2 2  0.7403
V3 3  0.6478
V4 4          0.6913
V5 5          0.5925
V6 6          0.4938
```

```
V7 7                   0.598
V8 8                   0.498
V9 9                   0.399

               Factor1 Factor2 Factor3
SS loadings       1.66    1.08    0.77
Proportion Var    0.18    0.12    0.09
Cumulative Var    0.18    0.30    0.39

 With factor correlations of
        Factor1 Factor2 Factor3
Factor1    1.00    0.67    0.59
Factor2    0.67    1.00    0.55
Factor3    0.59    0.55    1.00
```

Exploratory hierarchical models

In addition to the conventional oblique factor model, an alternative model is to consider the correlations between the factors to represent a higher order factor. This can be shown either as a *bifactor* solution Holzinger & Swineford (1937); Schmid & Leiman (1957) with a general factor for all variables and a set of residualized group factors, or as a hierarchical structure. An exploratory hierarchical model can be applied to this kind of data structure using the `omega` function. Graphic options include drawing a Schmid - Leiman bifactor solution (Figure 9) or drawing a hierarchical factor solution f(Figure 10).

A bifactor solution.

The *bifactor* solution has a general factor loading for each variable as well as a set of residual group factors. This approach has been used extensively in the measurement of ability and has more recently been used in the measure of psychopathology (Reise et al., 2007). Data sets included in the `bifactor` data include the original (Holzinger & Swineford, 1937) data set (`holzinger`) as well as a set from Reise et al. (2007) (`reise`) and a nine variable problem from Thurstone.

A hierarchical solution.

Both of these graphical representations are reflected in the output of the `omega` function. The first was done using a Schmid-Leiman transformation, the second was not. As will be seen later, the objects returned from these two analyses may be used as models for a `sem` analysis. It is also useful to examine the estimates of reliability reported by `omega`.

```
> om.bi <- omega(bifact)
```

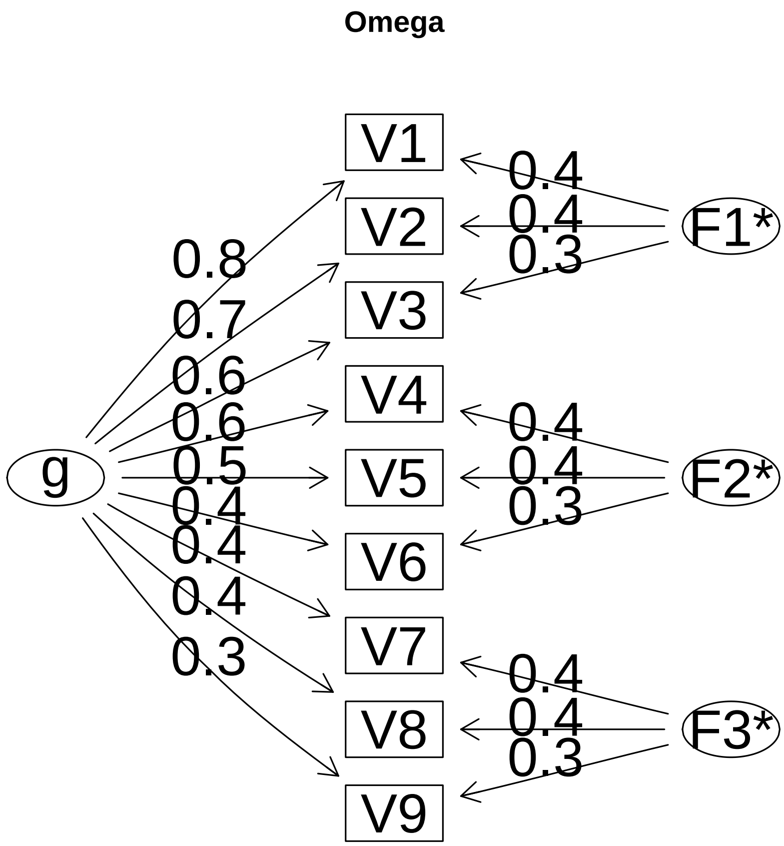

Figure 9. An exploratory bifactor solution to the nine variable problem

```
> om.bi

Omega
Alpha:  0.7899659
Lambda.6:
Omega Hierarchical:   0.715484
Omega Total   0.828264

Schmid Leiman Factor loadings greater than  0.2
   g    F1*  F2*  F3*  h2   u2
V1 0.81 0.39           0.81
```

```
> om.hi <- omega(bifact, sl = FALSE)
```

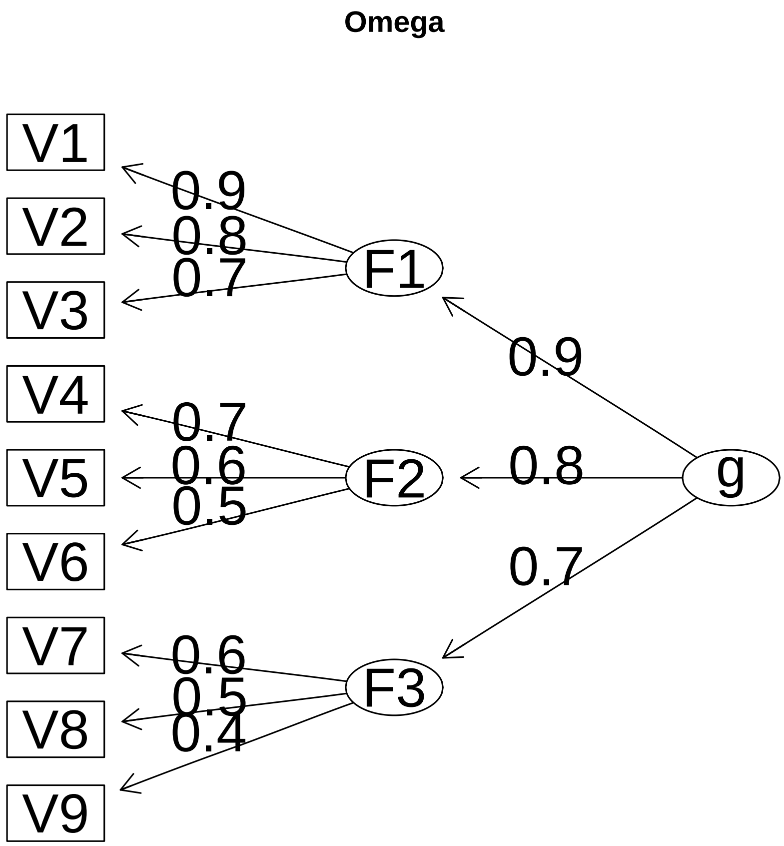

Figure 10. An exploratory hierarchical solution to the nine variable problem.

```
V2 0.72 0.35           0.64 0.36
V3 0.63 0.31           0.49 0.51
V4 0.56      0.42      0.49 0.51
V5 0.48      0.36      0.36 0.64
V6 0.40      0.30      0.25 0.75
V7 0.42           0.43 0.36 0.64
V8 0.35           0.36 0.25 0.75
V9 0.28           0.29      0.84

With eigenvalues of:
   g  F1*  F2*  F3*
```

```
2.65 0.37 0.40 0.40

general/max  6.66   max/min =   1.08
The degrees of freedom for the model is 12  and the fit was  0
```

Yet one more way to treat the hierarchical structure of a data set is to consider hierarchical cluster analysis using the `ICLUST` algorithm (Figure 11). ICLUST is most appropriate for forming item composites.

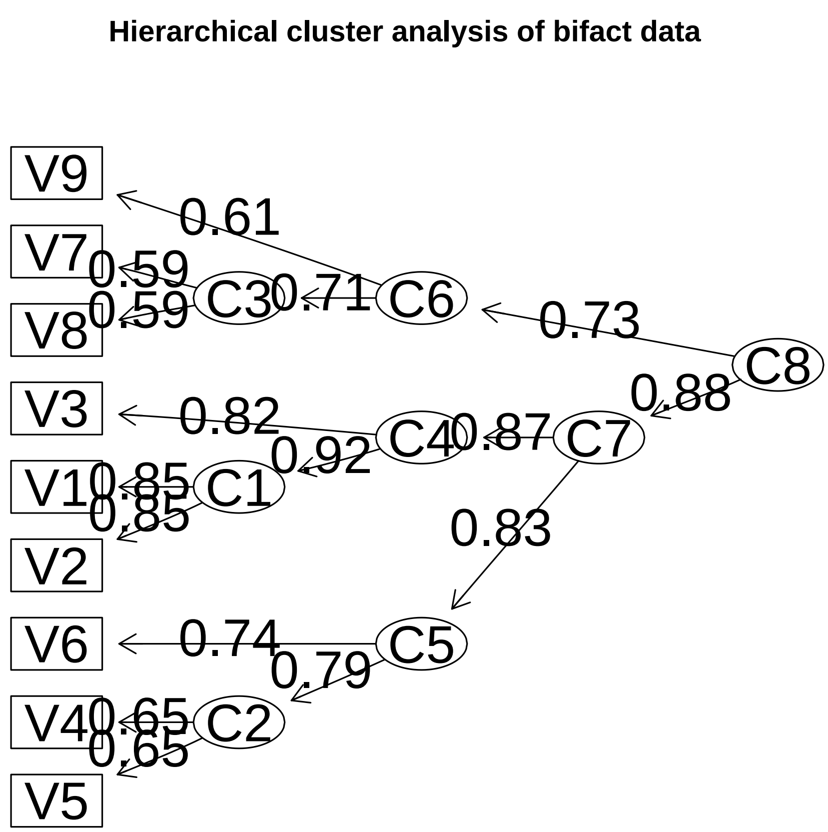

Figure 11. A hierarchical cluster analysis of the bifact data set using ICLUST

Confirmatory models

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, 2008).

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`.

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 Value
[1,] "X1->V1"   "a"       NA
[2,] "X1->V2"   "a"       NA
[3,] "X1->V3"   "a"       NA
[4,] "X1->V4"   "a"       NA
[5,] "V1<->V1"  "x1e"     NA
[6,] "V2<->V2"  "x2e"     NA
[7,] "V3<->V3"  "x3e"     NA
```

```
 [8,] "V4<->V4" "x4e"     NA
 [9,] "X1<->X1" NA        "1"
```

```
> sem.tau <- sem(mod.tau, cong, 100)
> summary(sem.tau)
```

```
 Model Chisquare =  11.947   Df =  5 Pr(>Chisq) = 0.035525
 Chisquare (null model) =  95.451   Df =  6
 Goodness-of-fit index =  0.94612
 Adjusted goodness-of-fit index =  0.89225
 RMSEA index =  0.11846   90% CI: (0.0286, 0.20654)
 Bentler-Bonnett NFI =  0.87484
 Tucker-Lewis NNFI =  0.9068
 Bentler CFI =  0.92234
 SRMR =  0.14102
 BIC =  -11.079

 Normalized Residuals
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
-2.2800 -0.8670 -0.0414 -0.2520  0.4680  1.2900

 Parameter Estimates
    Estimate Std Error z value Pr(>|z|)
a   0.67221  0.064400  10.4380 0.0000e+00 V1 <--- X1
x1e 0.45256  0.085350   5.3024 1.1429e-07 V1 <--> V1
x2e 0.45742  0.085949   5.3220 1.0261e-07 V2 <--> V2
x3e 0.60849  0.105361   5.7753 7.6800e-09 V3 <--> V3
x4e 0.79473  0.131889   6.0257 1.6837e-09 V4 <--> V4

 Iterations =  11
```

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 Value
 [1,] "X1->V1"  "a"       NA
 [2,] "X1->V2"  "b"       NA
 [3,] "X1->V3"  "c"       NA
 [4,] "X1->V4"  "d"       NA
 [5,] "V1<->V1" "x1e"     NA
```

```
 [6,] "V2<->V2" "x2e"     NA
 [7,] "V3<->V3" "x3e"     NA
 [8,] "V4<->V4" "x4e"     NA
 [9,] "X1<->X1" NA        "1"
```

```
> sem.cong <- sem(mod.cong, cong, 100)
> summary(sem.cong)
```

```
 Model Chisquare =  2.8163   Df =  2 Pr(>Chisq) = 0.24459
 Chisquare (null model) =  95.451   Df =  6
 Goodness-of-fit index =  0.98654
 Adjusted goodness-of-fit index =  0.93272
 RMSEA index =  0.064209   90% CI: (NA, 0.22022)
 Bentler-Bonnett NFI =  0.9705
 Tucker-Lewis NNFI =  0.97262
 Bentler CFI =  0.99087
 SRMR =  0.034671
 BIC =  -6.394

 Normalized Residuals
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
-0.9550 -0.0306  0.0381 -0.0431  0.1580  0.3300

 Parameter Estimates
    Estimate Std Error z value Pr(>|z|)
a   0.77539  0.10321   7.5130  5.7732e-14 V1 <--- X1
b   0.77062  0.10312   7.4727  7.8604e-14 V2 <--- X1
c   0.61623  0.10229   6.0244  1.6977e-09 V3 <--- X1
d   0.43467  0.10775   4.0343  5.4772e-05 V4 <--- X1
x1e 0.39877  0.10882   3.6645  2.4778e-04 V1 <--> V1
x2e 0.40614  0.10829   3.7504  1.7658e-04 V2 <--> V2
x3e 0.62026  0.10602   5.8506  4.8976e-09 V3 <--> V3
x4e 0.81106  0.12311   6.5880  4.4568e-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      2.8163
Model 2        5     11.9468  3   9.1305    0.02761 *
```

```
---
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.

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.

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

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

```
> bifact <- round(bifact, 5)
> sem.one <- sem(mod.one, bifact, 100)
```

```
> summary(sem.one)

 Model Chisquare =  18.729   Df =  27 Pr(>Chisq) = 0.87967
 Chisquare (null model) =  234.74   Df =  36
 Goodness-of-fit index =  0.95526
 Adjusted goodness-of-fit index =  0.92543
 RMSEA index =  0   90% CI: (NA, 0.039523)
 Bentler-Bonnett NFI =  0.92022
 Tucker-Lewis NNFI =  1.0555
 Bentler CFI =  1
 SRMR =  0.052506
 BIC =  -105.61

 Normalized Residuals
     Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
-2.67e-01 -1.85e-01 -1.40e-06  1.37e-01  1.20e-01  1.61e+00

 Parameter Estimates
    Estimate Std Error z value Pr(>|z|)
a   0.88014  0.084098  10.4657 0.0000e+00 V1 <--- X1
b   0.79786  0.087665   9.1013 0.0000e+00 V2 <--- X1
c   0.69867  0.092309   7.5688 3.7748e-14 V3 <--- X1
d   0.54016  0.099013   5.4555 4.8843e-08 V4 <--- X1
e   0.46911  0.101142   4.6381 3.5161e-06 V5 <--- X1
f   0.39443  0.102945   3.8315 1.2738e-04 V6 <--- X1
g   0.40361  0.102583   3.9344 8.3390e-05 V7 <--- X1
h   0.34005  0.103944   3.2714 1.0701e-03 V8 <--- X1
i   0.27422  0.105061   2.6101 9.0526e-03 V9 <--- X1
x1e 0.22535  0.061293   3.6765 2.3644e-04 V1 <--> V1
x2e 0.36342  0.068545   5.3019 1.1461e-07 V2 <--> V2
x3e 0.51186  0.083791   6.1087 1.0042e-09 V3 <--> V3
x4e 0.70822  0.107282   6.6015 4.0701e-11 V4 <--> V4
x5e 0.77993  0.115697   6.7412 1.5708e-11 V5 <--> V5
x6e 0.84442  0.123326   6.8471 7.5369e-12 V6 <--> V6
x7e 0.83710  0.122367   6.8409 7.8686e-12 V7 <--> V7
x8e 0.88437  0.128072   6.9053 5.0109e-12 V8 <--> V8
x9e 0.92481  0.132971   6.9549 3.5274e-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
```

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
      x4e       x5e       x6e       x7e       x8e       x9e
0.7082243 0.7799344 0.8444243 0.8371012 0.8843691 0.9248059

 Iterations =  14
```

The answers are, of course, identical.

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)
```

```
 Model Chisquare =  49.362   Df =  26 Pr(>Chisq) = 0.0037439
 Chisquare (null model) =  234.74   Df =  36
 Goodness-of-fit index =  0.89584
 Adjusted goodness-of-fit index =  0.81972
 RMSEA index =  0.095268   90% CI: (0.053304, 0.13543)
 Bentler-Bonnett NFI =  0.78972
 Tucker-Lewis NNFI =  0.83724
 Bentler CFI =  0.88245
 SRMR =  0.19571
 BIC =  -70.373

 Normalized Residuals
     Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
-2.04e-05  1.92e-05  1.76e+00  1.66e+00  2.63e+00  4.01e+00

 Parameter Estimates
     Estimate Std Error z value Pr(>|z|)
F1V1 0.79231  0.093980  8.4306  0.0000e+00 V1 <--- PA1
F2V1 0.23013  0.089392  2.5744  1.0043e-02 V1 <--- PA3
F1V2 0.80000  0.093251  8.5790  0.0000e+00 V2 <--- PA1
F1V3 0.70000  0.095002  7.3683  1.7275e-13 V3 <--- PA1
F2V4 0.70000  0.129238  5.4164  6.0827e-08 V4 <--- PA3
F2V5 0.60000  0.123717  4.8498  1.2359e-06 V5 <--- PA3
F2V6 0.50000  0.120027  4.1657  3.1037e-05 V6 <--- PA3
F3V7 0.60000  0.189530  3.1657  1.5470e-03 V7 <--- PA2
F3V8 0.50000  0.167439  2.9862  2.8250e-03 V8 <--- PA2
F3V9 0.40000  0.146908  2.7228  6.4733e-03 V9 <--- PA2
x1e  0.19428  0.073779  2.6333  8.4554e-03 V1 <--> V1
x2e  0.36000  0.085408  4.2150  2.4973e-05 V2 <--> V2
x3e  0.51000  0.089431  5.7028  1.1787e-08 V3 <--> V3
x4e  0.51000  0.151833  3.3589  7.8242e-04 V4 <--> V4
x5e  0.64000  0.135626  4.7188  2.3721e-06 V5 <--> V5
x6e  0.75000  0.130148  5.7627  8.2777e-09 V6 <--> V6
x7e  0.64000  0.219308  2.9183  3.5198e-03 V7 <--> V7
x8e  0.75000  0.174853  4.2893  1.7921e-05 V8 <--> V8
x9e  0.84000  0.148755  5.6468  1.6343e-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.

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`.

```
> bifact.s <- sim.hierarchical()
> bifact.s <- round(bifact.s, 5)
> f3 <- factor.pa(bifact.s, 3)
> f3.p <- Promax(f3)
> mod.f3p <- structure.sem(f3.p)
> mod.f3p
```

```
      Path        Parameter Value
 [1,] "PA1->V1"   "F1V1"    NA
 [2,] "PA1->V2"   "F1V2"    NA
 [3,] "PA1->V3"   "F1V3"    NA
 [4,] "PA3->V4"   "F2V4"    NA
 [5,] "PA3->V5"   "F2V5"    NA
 [6,] "PA3->V6"   "F2V6"    NA
 [7,] "PA2->V7"   "F3V7"    NA
 [8,] "PA2->V8"   "F3V8"    NA
 [9,] "PA2->V9"   "F3V9"    NA
[10,] "V1<->V1"   "x1e"     NA
[11,] "V2<->V2"   "x2e"     NA
[12,] "V3<->V3"   "x3e"     NA
[13,] "V4<->V4"   "x4e"     NA
[14,] "V5<->V5"   "x5e"     NA
```

```
[15,] "V6<->V6"   "x6e"     NA
[16,] "V7<->V7"   "x7e"     NA
[17,] "V8<->V8"   "x8e"     NA
[18,] "V9<->V9"   "x9e"     NA
[19,] "PA3<->PA1" "rF2F1"   NA
[20,] "PA2<->PA1" "rF3F1"   NA
[21,] "PA2<->PA3" "rF3F2"   NA
[22,] "PA1<->PA1" NA        "1"
[23,] "PA3<->PA3" NA        "1"
[24,] "PA2<->PA2" NA        "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)
```

```
 Model Chisquare =  7.0943   Df =  26 Pr(>Chisq) = 0.99991
 Chisquare (null model) =  234.74   Df =  36
 Goodness-of-fit index =  0.98577
 Adjusted goodness-of-fit index =  0.97537
 RMSEA index =  0   90% CI: (NA, NA)
 Bentler-Bonnett NFI =  0.96978
 Tucker-Lewis NNFI =  1.1317
 Bentler CFI =  1
 SRMR =  0.11957
 BIC =  -112.64
```

```
 Normalized Residuals
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 -1.970  -0.790  -0.514  -0.673  -0.344  -0.113

 Parameter Estimates
      Estimate Std Error z value Pr(>|z|)
F1V2  0.86964  0.088687  9.8057  0.0000e+00 V2 <--- PA1
F1V3  0.76175  0.094439  8.0661  6.6613e-16 V3 <--- PA1
F2V5  0.65068  0.124457  5.2282  1.7118e-07 V5 <--- PA3
F2V6  0.54625  0.125530  4.3515  1.3520e-05 V6 <--- PA3
F3V7  0.62995  0.133715  4.7111  2.4638e-06 V7 <--- PA2
F3V8  0.52495  0.130730  4.0155  5.9309e-05 V8 <--- PA2
F3V9  0.41996  0.130765  3.2116  1.3201e-03 V9 <--- PA2
x1e   0.17952  0.063477  2.8281  4.6832e-03 V1 <--> V1
x2e   0.36528  0.071261  5.1259  2.9612e-07 V2 <--> V2
x3e   0.51300  0.084177  6.0943  1.0992e-09 V3 <--> V3
x4e   0.38948  0.124838  3.1199  1.8090e-03 V4 <--> V4
x5e   0.67448  0.116554  5.7869  7.1709e-09 V5 <--> V5
x6e   0.77059  0.122091  6.3116  2.7618e-10 V6 <--> V6
x7e   0.64000  0.143188  4.4696  7.8356e-06 V7 <--> V7
x8e   0.75000  0.134931  5.5584  2.7229e-08 V8 <--> V8
x9e   0.84000  0.135238  6.2113  5.2542e-10 V9 <--> V9
rF2F1 0.73025  0.081115  9.0026  0.0000e+00 PA1 <--> PA3
rF3F1 0.67008  0.113054  5.9271  3.0842e-09 PA1 <--> PA2
rF3F2 0.58397  0.143537  4.0684  4.7329e-05 PA3 <--> PA2

 Iterations =  21
```

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

Extract a bifactor solution using `omega` and then test that model using `sem`

A bifactor solution has previously been shown (Figure 9). The output from the `omega` function includes the sem commands for the analysis. As an example of doing this with real rather than simulated data, consider 9 variables from Thurstone. For completeness, the `std.coef` from *sem* is used as well as the `summary` function.

sem of Thurstone 9 variable problem. The sem manual includes an example of a hierarchical solution to 9 mental abilities originally reported by Thurstone and used in the SAS manual for PROC CALIS and discussed in detail by McDonald (1999). The data

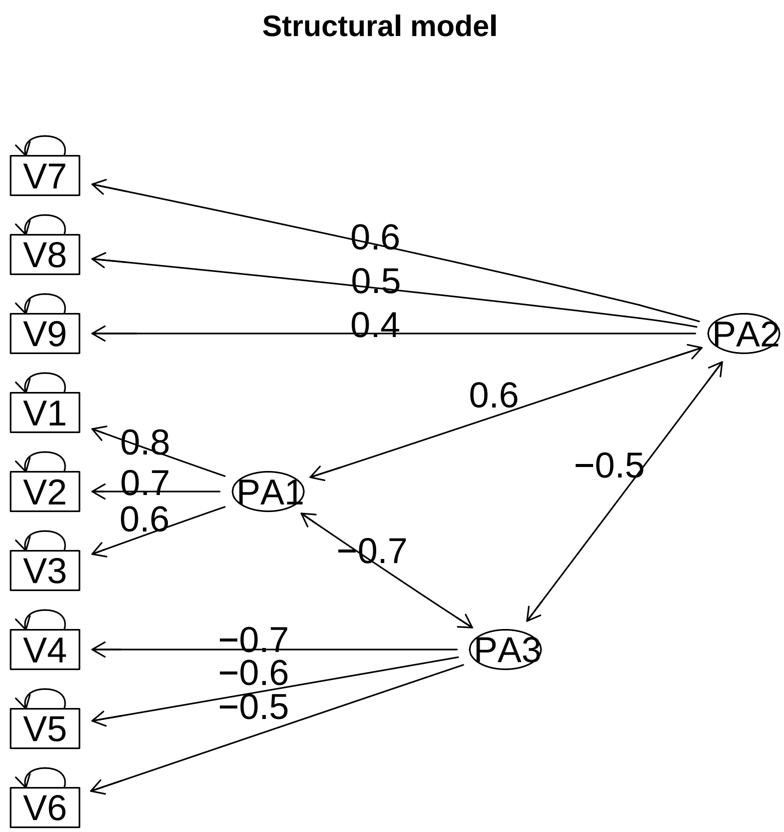

Figure 12. A three factor, oblique solution.

matrix, as reported by Fox may be found in the `bifactor` data set. Using the commands just shown, it is possible to analyze this data set using a bifactor solution (Figure 13).

```
> sem.bi <- sem(om.th.bi$model, Thurstone, 213)
> summary(sem.bi)
```

```
 Model Chisquare =  24.216   Df =  18 Pr(>Chisq) = 0.14807
 Chisquare (null model) =  1101.9   Df =  36
 Goodness-of-fit index =  0.97578
 Adjusted goodness-of-fit index =  0.93944
 RMSEA index =  0.040361   90% CI: (NA, 0.077994)
 Bentler-Bonnett NFI =  0.97802
 Tucker-Lewis NNFI =  0.98834
```

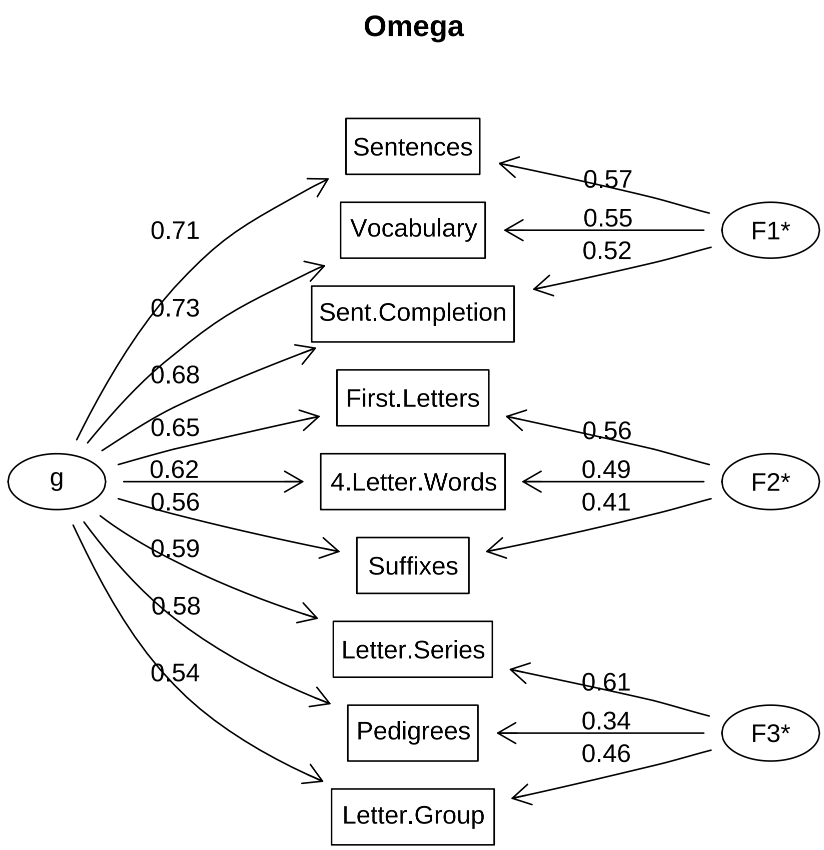

Figure 13. A bifactor solution to the Thurstone 9 variable problem. All items load on a general factor of ability, the residual factors account for the correlations between items within groups.

```
 Bentler CFI =  0.99417
 SRMR =  0.034895
 BIC =  -72.287

 Normalized Residuals
     Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
-8.21e-01 -3.34e-01 -8.92e-07  2.82e-02  1.56e-01  1.80e+00

 Parameter Estimates
                  Estimate Std Error z value  Pr(>|z|)
Sentences         0.76787  0.072626  10.57292 0.0000e+00 Sentences <--- g
```

```
Vocabulary         0.79092  0.072418  10.92170 0.0000e+00 Vocabulary <--- g
Sent.Completion    0.75362  0.073402  10.26709 0.0000e+00 Sent.Completion <--- g
First.Letters      0.60838  0.072201   8.42617 0.0000e+00 First.Letters <--- g
4.Letter.Words     0.59733  0.073851   8.08843 6.6613e-16 4.Letter.Words <--- g
Suffixes           0.57179  0.071492   7.99792 1.3323e-15 Suffixes <--- g
Letter.Series      0.56689  0.074271   7.63281 2.2871e-14 Letter.Series <--- g
Pedigrees          0.66233  0.069321   9.55455 0.0000e+00 Pedigrees <--- g
Letter.Group       0.52995  0.078985   6.70955 1.9522e-11 Letter.Group <--- g
F1*Sentences       0.48787  0.085457   5.70898 1.1366e-08 Sentences <--- F1*
F1*Vocabulary      0.45232  0.090422   5.00233 5.6640e-07 Vocabulary <--- F1*
F1*Sent.Completion 0.40445  0.093402   4.33024 1.4895e-05 Sent.Completion <--- F1*
F2*First.Letters   0.61405  0.085794   7.15733 8.2268e-13 First.Letters <--- F2*
F2*4.Letter.Words  0.50581  0.084848   5.96130 2.5024e-09 4.Letter.Words <--- F2*
F2*Suffixes        0.39432  0.078289   5.03671 4.7360e-07 Suffixes <--- F2*
F3*Letter.Series   0.72730  0.159499   4.55988 5.1183e-06 Letter.Series <--- F3*
F3*Pedigrees       0.24684  0.089011   2.77317 5.5513e-03 Pedigrees <--- F3*
F3*Letter.Group    0.40915  0.122180   3.34875 8.1177e-04 Letter.Group <--- F3*
e1                 0.17236  0.034113   5.05265 4.3571e-07 Sentences <--> Sentences
e2                 0.16984  0.030037   5.65438 1.5641e-08 Vocabulary <--> Vocabulary
e3                 0.26847  0.033188   8.08958 6.6613e-16 Sent.Completion <--> Sent.Complet
e4                 0.25281  0.079472   3.18115 1.4669e-03 First.Letters <--> First.Letters
e5                 0.38735  0.063194   6.12960 8.8103e-10 4.Letter.Words <--> 4.Letter.Word
e6                 0.51757  0.059639   8.67838 0.0000e+00 Suffixes <--> Suffixes
e7                 0.14967  0.223242   0.67044 5.0257e-01 Letter.Series <--> Letter.Series
e8                 0.50039  0.059655   8.38800 0.0000e+00 Pedigrees <--> Pedigrees
e9                 0.55175  0.084725   6.51223 7.4043e-11 Letter.Group <--> Letter.Group

 Iterations =  72
```

> std.coef(sem.bi)

```
                                       Std. Estimate
Sentences          Sentences           0.76787      Sentences <--- g
Vocabulary         Vocabulary          0.79092      Vocabulary <--- g
Sent.Completion    Sent.Completion     0.75362      Sent.Completion <--- g
First.Letters      First.Letters       0.60838      First.Letters <--- g
4.Letter.Words     4.Letter.Words      0.59733      4.Letter.Words <--- g
Suffixes           Suffixes            0.57179      Suffixes <--- g
Letter.Series      Letter.Series       0.56690      Letter.Series <--- g
Pedigrees          Pedigrees           0.66233      Pedigrees <--- g
Letter.Group       Letter.Group        0.52995      Letter.Group <--- g
F1*Sentences       F1*Sentences        0.48787      Sentences <--- F1*
```

```
F1*Vocabulary      F1*Vocabulary      0.45232       Vocabulary <--- F1*
F1*Sent.Completion F1*Sent.Completion 0.40445       Sent.Completion <--- F1*
F2*First.Letters   F2*First.Letters   0.61405       First.Letters <--- F2*
F2*4.Letter.Words  F2*4.Letter.Words  0.50581       4.Letter.Words <--- F2*
F2*Suffixes        F2*Suffixes        0.39432       Suffixes <--- F2*
F3*Letter.Series   F3*Letter.Series   0.72730       Letter.Series <--- F3*
F3*Pedigrees       F3*Pedigrees       0.24684       Pedigrees <--- F3*
F3*Letter.Group    F3*Letter.Group    0.40915       Letter.Group <--- F3*
```

Compare this solution to the one reported below, and to the sem manual.

Examining a hierarchical solution

A hierarchical solution to this data set was previously found by the `omega` function (Figure 10). The output of that analysis can be used as a model for a `sem` analysis. Once again, the `std.coef` function helps see the structure. Alternatively, using the `omega` function on the `Thurstone` data (in the `bifactor` data set) will create the model for this particular data set.

```
> sem.hi <- sem(om.hi$model, Thurstone, 213)
> summary(sem.hi)
```

```
 Model Chisquare =  38.196   Df =  24 Pr(>Chisq) = 0.033101
 Chisquare (null model) =  1101.9   Df =  36
 Goodness-of-fit index =  0.95957
 Adjusted goodness-of-fit index =  0.9242
 RMSEA index =  0.052822   90% CI: (0.015262, 0.083067)
 Bentler-Bonnett NFI =  0.96534
 Tucker-Lewis NNFI =  0.98002
 Bentler CFI =  0.98668
 SRMR =  0.043595
 BIC =  -90.475

 Normalized Residuals
    Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
-9.72e-01 -4.16e-01 -6.36e-07  4.01e-02  9.39e-02  1.63e+00

 Parameter Estimates
                 Estimate Std Error z value Pr(>|z|)
gF1              1.44381  0.264174  5.4654  4.6187e-08 F1 <--- g
gF2              1.25383  0.216597  5.7888  7.0905e-09 F2 <--- g
gF3              1.40655  0.279332  5.0354  4.7682e-07 F3 <--- g
```

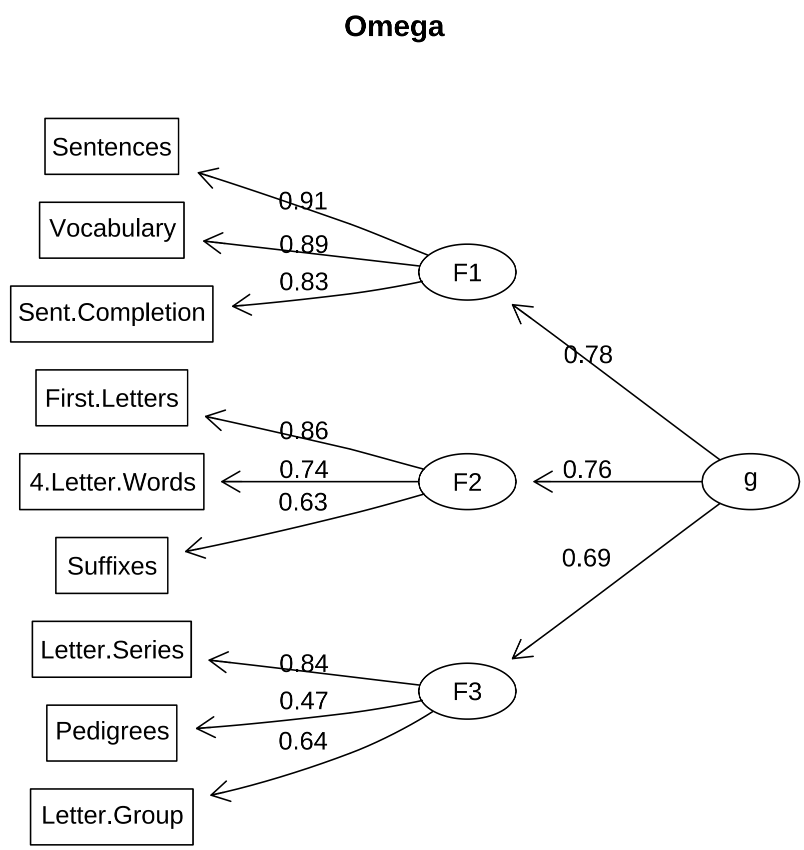

Figure 14. Hierarchical analysis of the Thurstone 9 variable problem using an exploratory algorithm can provide the appropriate sem code for analysis using the sem package.

```
F1Sentences        0.51512  0.064964  7.9293  2.2204e-15 Sentences <--- F1
F1Vocabulary       0.52031  0.065162  7.9849  1.3323e-15 Vocabulary <--- F1
F1Sent.Completion  0.48743  0.062422  7.8087  5.7732e-15 Sent.Completion <--- F1
F2First.Letters    0.52112  0.063137  8.2538  2.2204e-16 First.Letters <--- F2
F24.Letter.Words   0.49707  0.059673  8.3298  0.0000e+00 4.Letter.Words <--- F2
F2Suffixes         0.43806  0.056479  7.7562  8.6597e-15 Suffixes <--- F2
F3Letter.Series    0.45244  0.071371  6.3392  2.3101e-10 Letter.Series <--- F3
F3Pedigrees        0.41729  0.061037  6.8367  8.1024e-12 Pedigrees <--- F3
F3Letter.Group     0.40763  0.064524  6.3175  2.6584e-10 Letter.Group <--- F3
e1                 0.18150  0.028400  6.3907  1.6517e-10 Sentences <--> Sentences
e2                 0.16493  0.027797  5.9334  2.9679e-09 Vocabulary <--> Vocabulary
```

```
e3                 0.26713  0.033468  7.9816  1.5543e-15 Sent.Completion <--> Sent.Completio
e4                 0.30150  0.050686  5.9484  2.7073e-09 First.Letters <--> First.Letters
e5                 0.36450  0.052358  6.9617  3.3618e-12 4.Letter.Words <--> 4.Letter.Words
e6                 0.50641  0.059963  8.4455  0.0000e+00 Suffixes <--> Suffixes
e7                 0.39033  0.061599  6.3367  2.3474e-10 Letter.Series <--> Letter.Series
e8                 0.48137  0.065388  7.3618  1.8141e-13 Pedigrees <--> Pedigrees
e9                 0.50510  0.065227  7.7437  9.5479e-15 Letter.Group <--> Letter.Group

 Iterations =  53

> std.coef(sem.hi)

                                     Std. Estimate
gF1                gF1                0.82208       F1 <--- g
gF2                gF2                0.78180       F2 <--- g
gF3                gF3                0.81501       F3 <--- g
F1Sentences        F1Sentences        0.90471       Sentences <--- F1
F1Vocabulary       F1Vocabulary       0.91382       Vocabulary <--- F1
F1Sent.Completion  F1Sent.Completion  0.85608       Sent.Completion <--- F1
F2First.Letters    F2First.Letters    0.83576       First.Letters <--- F2
F24.Letter.Words   F24.Letter.Words   0.79718       4.Letter.Words <--- F2
F2Suffixes         F2Suffixes         0.70256       Suffixes <--- F2
F3Letter.Series    F3Letter.Series    0.78081       Letter.Series <--- F3
F3Pedigrees        F3Pedigrees        0.72016       Pedigrees <--- F3
F3Letter.Group     F3Letter.Group     0.70349       Letter.Group <--- F3

> anova(sem.hi, sem.bi)

LR Test for Difference Between Models

        Model Df Model Chisq Df LR Chisq Pr(>Chisq)
Model 1       24      38.196
Model 2       18      24.216  6   13.980    0.02986 *
---
Signif. codes:  0 âĂŸ***âĂŹ 0.001 âĂŸ**âĂŹ 0.01 âĂŸ*âĂŹ 0.05 âĂŸ.âĂŹ 0.1 âĂŸ âĂŹ 1
```

Using the Thurstone data set, we see what happens when a hierarchical model is applied to real data. The exploratory structure derived from the `omega` function (Figure 14) provides estimates in close approximation to those found using `sem`. The model definition created by using `omega` is the same hierarchical model discussed in the `sem` help page. The *bifactor* model, with 6 more parameters does provide a better fit to the data than the hierarchical model.

Similar analyses can be done with other data that are organized hierarchically. Examples of these analyses are analyzing the 14 variables of `holzinger` and the 16 variables of `reise`. The output from the following analyses has been limited to just the comparison between the bifactor and hierarchical solutions.

```
> data(bifactor)
> om.holz.bi <- omega(Holzinger, 4)
> sem.holz.bi <- sem(om.holz.bi$model, Holzinger, 355)
> om.holz.hi <- omega(Holzinger, 4, sl = FALSE)
> sem.holz.hi <- sem(om.holz.hi$model, Holzinger, 355)
> anova(sem.holz.bi, sem.holz.hi)
```

```
LR Test for Difference Between Models

        Model Df Model Chisq Df LR Chisq Pr(>Chisq)
Model 1       63     147.179
Model 2       73     183.047 10   35.868  8.868e-05 ***
---
Signif. codes:  0 âĂŸ***âĂŹ 0.001 âĂŸ**âĂŹ 0.01 âĂŸ*âĂŹ 0.05 âĂŸ.âĂŹ 0.1 âĂŸ âĂŹ 1
```

Summary and conclusion

The use of exploratory and confirmatory models for understanding real data structures is an important advance in psychological research. To understand these approaches it is helpful to try them first on "baby" data sets. To the extent that the models we use can be tested on simple, artificial examples, it is perhaps easier to practice their application. The *psych* tools for simulating structural models and for specifying models are a useful supplement to the power of packages such as *sem*. The techniques that can be used on simulated data set can also be applied to real data sets.

References

Fox, J. (2006). Structural equation modeling with the sem package in R. *Structural Equation Modeling*, *13*, 465-486.

Fox, J. (2008). *sem: Structural equation models.* (R package version 0.9-13)

Holzinger, K., & Swineford, F. (1937, 03 27). The bi-factor method. *Psychometrika*, *2*(1), 41–54.

Jensen, A. R., & Weng, L.-J. (1994). What is a good g? *Intelligence*, *18*(3), 231-258.

McDonald, R. P. (1999). *Test theory: A unified treatment.* Mahwah, N.J.: L. Erlbaum Associates.

Rafaeli, E., & Revelle, W. (2006). A premature consensus: Are happiness and sadness truly opposite affects? *Motivation and Emotion*, *30*(1), 1-12.

Reise, S., Morizot, J., & Hays, R. (2007, 08 25). The role of the bifactor model in resolving dimensionality issues in health outcomes measures. *Quality of Life Research*, *16*(0), 19–31.

Revelle, W. (2009). *psych: Procedures for personality and psychological research.* (R package version 1.0-64)

Schmid, J. J., & Leiman, J. M. (1957). The development of hierarchical factor solutions. *Psychometrika*, *22*(1), 83-90.

Index