

TEACHER'S CORNER

Structural Equation Modeling With the `sem` Package in R

John Fox
McMaster University

R is free, open-source, cooperatively developed software that implements the S statistical programming language and computing environment. The current capabilities of R are extensive, and it is in wide use, especially among statisticians. The `sem` package provides basic structural equation modeling facilities in R, including the ability to fit structural equations in observed variable models by two-stage least squares, and to fit latent variable models by full information maximum likelihood assuming multinormality. This article briefly describes R, and then proceeds to illustrate the use of the `tsls` and `sem` functions in the `sem` package. The article also demonstrates the integration of the `sem` package with other facilities available in R, for example for computing polychoric correlations and for bootstrapping.

R (Ihaka & Gentleman, 1996; R Development Core Team, 2005) is a free, open-source, cooperatively developed implementation of the S statistical programming language and computing environment (Becker, Chambers, & Wilks, 1988; Chambers, 1998; Chambers & Hastie, 1992).¹ Since its introduction in the mid-1990s, R has rapidly become one of the most widely used facilities for statistical computing, especially among statisticians, and arguably now has broader coverage of statistical methods than any other statistical software. The basic R system, with capabilities roughly comparable to (say) a basic installation of SAS, can be augmented by contributed packages, which now number more than 500. These

Correspondence should be addressed to John Fox, Department of Sociology, McMaster University, 1280 Main Street West, Hamilton, Ontario, Canada L8S 4M4. E-mail: jfox@mcmaster.ca

¹There is also a commercial implementation of the S statistical computing environment called S-PLUS, which antedates R, and which is distributed by Insightful Corporation (<http://www.insightful.com/products/splus/>). The `sem` package for R described in this article could be adapted for use with S-PLUS without too much trouble, but it does not work with S-PLUS in its current form.

packages, along with the basic R software, are available on the Comprehensive R Archive Network (CRAN) Web sites, with the main CRAN archive in Vienna (at <http://cran.r-project.org/>; see also the R home page, at <http://www.r-project.org/>). R runs on all of the major computing platforms, including Linux/UNIX systems, Microsoft Windows systems, and Macintoshes under OS/X.

This article describes the `sem` package in R, which provides a basic structural equation modeling (SEM) facility, including the ability to estimate structural equations in observed variable models by two-stage least squares (2SLS), and to fit general (including latent variable) models by full information maximum likelihood (FIML) assuming multinormality. There is, in addition, the `systemfit` package, not described here, which implements a variety of observed variable structural equation estimators.

The first section of this article provides a brief introduction to computing in R. Subsequent sections describe the use of the `tsls` function for 2SLS estimation and the `sem` function for fitting general structural equation models. A concluding section suggests possible future directions for the `sem` package.

BACKGROUND: A BRIEF INTRODUCTION TO R

It is not possible within the confines of this article to give more than a cursory introduction to R. The purpose of this section is to provide some background and orienting information. Beyond that, R comes with a complete set of manuals, including a good introductory manual; other documentation is available on the R Web site and in a number of books (e.g., Fox, 2002; Venables & Ripley, 2002). R also has an extensive online help system, reachable through the `help` command, `? operator`, and some other commands, such as `help.search`.

Although one can build graphical interfaces to R—for example, the `Rcmdr` (“R Commander”) package provides a basic statistics graphical-user interface—R is fundamentally a command-driven system. The user interacts with the R interpreter either directly at a command prompt in the R console or through a programming editor; the Windows version of R incorporates a simple script editor.

Figure 1 shows the main R window as it appears at start-up on a Microsoft Windows XP system. The greater than (>) symbol to the left of the block cursor in the R console is the command prompt; commands entered at the prompt are statements in the S language, and may include mathematical and other expressions to be evaluated along with function calls.

The following are some examples of simple R commands:

```
> 1 + 2*3^2 # an arithmetic expression
[1] 19

> c(1, 2, 3)*2 # vectorized arithmetic
[1] 2 4 6
```

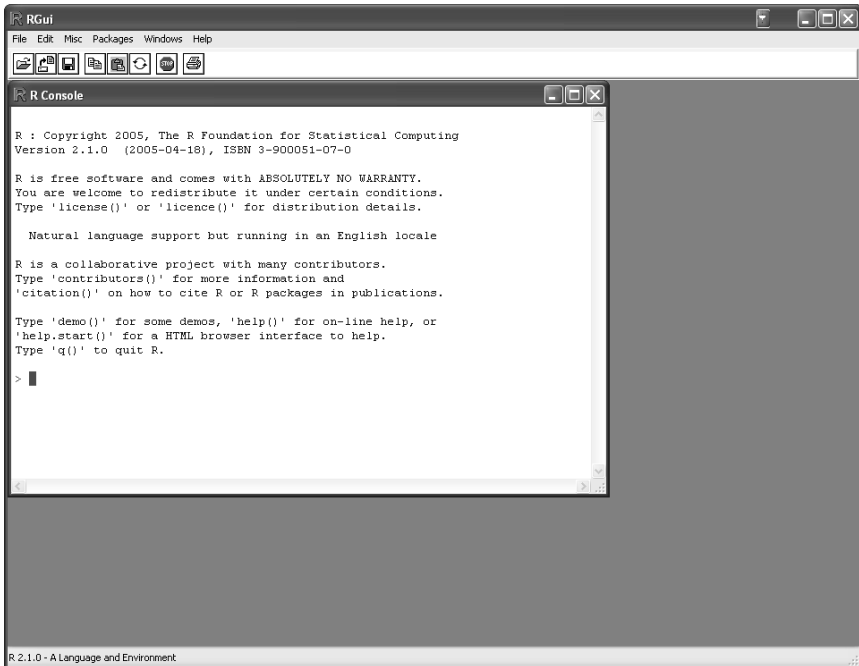


FIGURE 1 The Windows version of R at start-up, showing the main R window and the R console.

```
> 2:5 # the sequence operator
[1] 2 3 4 5

> (2:5)/c(2, 2, 3, 3)
[1] 1.000000 1.500000 1.333333 1.666667

> x <- rnorm(1:10) # random normal numbers
> x # print x
[1] 0.7115833 1.4496708 0.2819413 0.9289971 1.1364372 0.6987258
[7] -2.2418103 -0.2712084 0.1998054 -1.1573525

> mean(x)
[1] 0.1736790

> mean(rnorm(100))
[1] -0.2394827
```

- The first command is an arithmetic expression representing $1 + 2 \times 3^2$; the normal precedence of arithmetic operators applies, and so exponentiation precedes multiplication, which precedes addition. The spaces in the command are optional,

and are meant to clarify the expression. The pound sign (#) is a comment character: Everything to its right is ignored by the R interpreter.

- The second command illustrates vectorized arithmetic, in which each element of a three-element vector is multiplied by 2; here `c` is the combine function, which constructs a vector from its arguments. As is general in S, the arguments to the `c` function are specified in parentheses and are separated by commas; arguments may be specified by position or by (abbreviated) name, in which case they need not appear in order. In many commands, some arguments have default values and therefore need not be specified explicitly.

- In the third command, the sequence operator (`:`) is used to generate a vector of consecutive integers, and in the following command, this vector is divided by a vector of the same length, with the operation performed element-wise on corresponding entries of the two vectors. As here, parentheses may be used to clarify expressions, or to alter the normal order of evaluation.

- In the next command, the `rnorm` function is called with the argument `10` to sample 10 pseudo-random numbers from the standard normal distribution; the result is assigned to a variable named `x`. Enter `?rnorm` or `help(rnorm)` at the command prompt to see the help page for the `rnorm` function. The symbol `<-`, composed of a less than sign and a hyphen, is the assignment operator; an equals sign (`=`) may also be used for assignment. Variables (and other objects) in R can have names of arbitrary length, composed of uppercase and lowercase letters, numerals, underscores, and periods, but must start with a letter or a period; nonstandard names incorporating other characters are supported, but are less convenient. R is case-sensitive, and so, for example, the names `x` and `X` are distinguished.

- Notice that nothing is printed following an assignment. Subsequently typing the name of the variable `x` prints its contents. The number in square brackets at the start of each line of output gives the index of the first element displayed in the line.

- In the next command, the `mean` function is used to calculate the average of the entries in `x`.

- The final preliminary example shows how the result of one function (100 random normal values returned by `rnorm`) can be passed as an argument to another function (`mean`). This style is common in formulating S commands.

Data can be input into R from many different sources: entered directly at the keyboard, read from plain-text files in a variety of formats, imported from other statistical packages and spreadsheet programs, read from database management systems, and so on. Case-by-variable data sets can be stored in data frame objects, which are analogous to internal data sets in statistical packages such as SAS or SPSS.

R also supports many different kinds of data structures (e.g., vectors, matrices, arrays, lists, and objects created in two object-oriented programming systems) and types (e.g., numeric, character, and logical data). Indeed, an advantage of working in a statistical computing environment, as opposed to a traditional statistical package, is that in the former data are flexibly manipulable by the user, both directly

and through programs. Moreover, the functions (programs) that the user writes are syntactically indistinguishable from the functions provided with R.

Related sets of functions, data, and documentation can be collected into R packages, and either maintained for private use or contributed to CRAN. The sophisticated tools provided for writing, maintaining, building, and checking packages are one of the strengths of R.

2SLS ESTIMATION OF OBSERVED VARIABLE MODELS

The `tsls` function in the `sem` package fits structural equations by two-stage least squares (2SLS) using the general S “formula” interface. S model formulas implement a variant of Wilkinson and Rogers’s (1973) notation for linear models; formulas are used in a wide variety of model-fitting functions in R (e.g., the basic `lm` and `glm` functions for fitting linear and generalized linear models, respectively).

2SLS estimation is illustrated using a classical application of SEM in econometrics: Klein’s “Model I” of the U.S. economy (Klein, 1950; see also, e.g., Greene, 1993, pp. 581–582). Klein’s data, a time-series data set for the years 1920 to 1941, are included in the data frame `Klein` in the `sem` package:

```
> library(sem)
> data(Klein)
> Klein
  Year   C   P   Wp   I K.lag   X   Wg   G   T
1 1920 39.8 12.7 28.8  2.7 180.1 44.9 2.2  2.4  3.4
2 1921 41.9 12.4 25.5 -0.2 182.8 45.6 2.7  3.9  7.7
3 1922 45.0 16.9 29.3  1.9 182.6 50.1 2.9  3.2  3.9
. . .
21 1940 65.0 21.1 45.0  3.3 201.2 75.7 8.0  7.4  9.6
22 1941 69.7 23.5 53.3  4.9 204.5 88.4 8.5 13.8 11.6
```

The `library` command loads the `sem` package, and the `data` command reads the `Klein` data set into memory. (The ellipses, `...`, represent lines elided from the output.)

Greene (1993, p. 581) wrote Klein’s model as follows:

$$C_t = \alpha_0 + \alpha_1 P_t + \alpha_2 P_{t-1} + \alpha_3 (W^p_t + W^s_t) + \varepsilon_{1t}$$

$$I_t = \beta_0 + \beta_1 P_t + \beta_2 P_{t-1} + \beta_3 K_{t-1} + \varepsilon_{2t}$$

$$W^p_t = \gamma_0 + \gamma_1 X_t + \gamma_2 X_{t-1} + \gamma_3 A_t + \varepsilon_{3t}$$

$$X_t = C_t + I_t + G_t$$

$$P_t = X_t - T_t - W^p_t$$

$$K_t = K_{t-1} + I_t$$

The last three equations are identities, and do not figure directly in the 2SLS estimation of the model. The variables in the model, again as given by Greene, are C

(consumption), I (investment), W^p (private wages), W^g (government wages), X (equilibrium demand), P (private profits), K (capital stock), A (a trend variable, expressed as year-1931), and G (government nonwage spending). The subscript t indexes observations.

Because the model includes lagged variables that are not directly supplied in the data set, the observation for the first year, 1920, is effectively lost to estimation. The lagged variables can be added to the data frame as follows, printing the first three observations:

```
> Klein$P.lag <- c(NA, Klein$P[-22])
> Klein$X.lag <- c(NA, Klein$X[-22])
> Klein[1:3, ]
  Year   C   P   Wp   I K.lag   X   Wg   G   T P.lag X.lag
1 1920 39.8 12.7 28.8  2.7 180.1 44.9 2.2 2.4 3.4   NA   NA
2 1921 41.9 12.4 25.5 -0.2 182.8 45.6 2.7 3.9 7.7 12.7 44.9
3 1922 45.0 16.9 29.3  1.9 182.6 50.1 2.9 3.2 3.9 12.4 45.6
```

In S, NA (not available) represents missing data, and, consistent with standard statistical notation, a negative subscript, such as -22 , drops observations. Square brackets are used to index objects such as data frames (e.g., `Klein[1:3,]`), vectors (e.g., `Klein$P[-22]`), matrices, arrays, and lists. The dollar sign (\$) can be used to index elements of data frames or lists.

The available instrumental variables are the exogenous variables G , T , W^g , A , and the constant regressor, and the predetermined variables K_{t-1} , P_{t-1} , and X_{t-1} . Using these instrumental variables, the structural equations can be estimated:

```
> Klein.eqn1 <- tsls(C ~ P + P.lag + I(Wp + Wg),
+   instruments=~G + T + Wg + I(Year - 1931) + K.lag + P.lag + X.lag,
+   data=Klein)

> Klein.eqn2 <- tsls(I ~ P + P.lag + K.lag,
+   instruments=~G + T + Wg + I(Year - 1931) + K.lag + P.lag + X.lag,
+   data=Klein)

> Klein.eqn3 <- tsls(Wp ~ X + X.lag + I(Year - 1931),
+   instruments=~G + T + Wg + I(Year - 1931) + K.lag + P.lag + X.lag,
+   data=Klein)
```

- The `tsls` function returns an object, which, in each case, has been saved in a variable; for example, `Klein.eqn1` for the first structural equation. Because of the assignment, no results are printed. We can create a brief printout by entering the name of the object, or a more complete listing via the `summary` function (see later). If desired, further computation could be performed on the object, such as extracting residuals or fitted values (via the `residuals` and `fitted.values` functions), or comparing two nested models by an F test (via the `anova` function). Generic

functions such as `summary`, `residuals`, `fitted.values`, and `anova` have methods for handling `tsls` objects appropriately. The same generics are used for other classes of objects, such as linear and generalized linear models.

- We can perform arithmetic operations and function calls within a model formula. Thus $I(W_P + W_G)$ forms a single regressor by summing the two wage variables; it is necessary to protect this operation with the identify function I (which returns its argument unchanged) because otherwise an arithmetic operator such as $+$ would be accorded special meaning in a model formula; specifically, $+$ means add a term to the model, and therefore, without protection, $W_P + W_G$ would enter the two variables into the regression separately. We can read the model formula for Equation 1 as, “Regress C on P , $P.lag$, and the sum of W_P and W_G .” Were factors (categorical predictors) included in the model, R would automatically have generated contrasts to represent the factors, by default using dummy-coded (0/1) regressors. Interactions, nesting, transformations of variables, etc., can also be specified on the right side of a model formula, and ordinary arithmetic operations and function calls on the left side. Unless it is explicitly suppressed (by including -1 on the right side of the model formula), a constant regressor is included in the model.

- The `instruments` argument to `tsls` specifies the instrumental variables as a one-sided model formula. As in the specification of the structural equation, the constant variable is automatically included among the instruments.

- When an R command is syntactically incomplete, it is continued to the next line, as indicated in the listing by the $+$ prompt, which is supplied at the beginning of each continuation line by the interpreter.

To produce a printed summary for the first structural equation:

```
> summary(Klein.eqn1)
2SLS Estimates
Model Formula: C ~ P + P.lag + I(Wp + Wg)
Instruments: ~G + T + Wg + I(Year - 1931) + K.lag + P.lag + X.lag

Residuals:
      Min.      1st Qu.      Median      Mean      3rd Qu.      Max.
-1.89e+00 -6.16e-01 -2.46e-01 -6.60e-11  8.85e-01  2.00e+00

              Estimate Std. Error t value Pr(>|t|)
(Intercept)  16.55476      1.46798  11.2772 2.587e-09
P              0.01730      0.13120   0.1319 8.966e-01
P.lag         0.21623      0.11922   1.8137 8.741e-02
I(Wp + Wg)    0.81018      0.04474  18.1107 1.505e-12
Residual standard error: 1.1357 on 17 degrees of freedom
```

To save space, the summaries for Equations 2 and 3 are omitted.

The coefficient estimates are identical to those in Greene (1993), but the coefficient standard errors differ slightly, because the `summary` method for `tsls` ob-

jects uses residual degrees of freedom (17) rather than the number of observations (21) to estimate the error variance. To recover Greene's asymptotic standard errors, the covariance matrix of the coefficients can be extracted and adjusted, illustrating a computation on a `tsls` object:

```
> sqrt(diag(vcov(Klein.eqn1)*17/21))
(Intercept)          P          P.lag  I(Wp + Wg)
 1.32079242  0.11804941  0.10726796  0.04024971
```

`vcov` is a generic function that returns a variance–covariance matrix, here for the `tsls` object `Klein.eqn1`, and `diag` extracts the main diagonal of this matrix.

ESTIMATING GENERAL STRUCTURAL EQUATION MODELS

A Structural Equation Model With Latent Exogenous and Endogenous Variables

Like Klein's Model I, Wheaton, Muthén, Alwin, and Summers's (1977) panel data on the stability of alienation have been a staple of the SEM literature, making an appearance, for example, in both the Lisrel manual (Jöreskog & Sörbom, 1989) and in the documentation for the CALIS procedure in SAS (SAS Institute, 2004). The path diagram in Figure 2 is for a model fit to the Wheaton et al. data in the Lisrel manual (Jöreskog & Sörbom, 1989, pp. 169–177). This diagram employs the usual conventions, representing observed variables by Roman letters enclosed in rectangles and unobserved variables (including latent variables and errors) by Greek letters enclosed in ellipses and circles. Directed arrows designate regression coefficients, and bidirectional arrows signify covariances. The covariances represented by the two bidirectional arrows with broken lines are not included in an initial model specified for these data later. The directed arrows are labeled with Greek letters representing the corresponding regression coefficients.

Anomia and Powerlessness are two subscales of a standard alienation scale, with data collected on a panel of individuals from rural Illinois in both 1967 and 1971. (In this version of the model, data on these variables from a 1966 wave of the study are ignored.) Education is measured in years, and SEI represents a socioeconomic index based on the respondent's occupation. The latent variable SES stands for socioeconomic status.²

²It is curious that the various forms of the Wheaton model treat SES as a cause of the indicators Education and SEI, rather than treating these variables as constitutive of SES (i.e., reversing the arrows in the path diagram, which produces an observationally indistinguishable model), but for consistency with the literature that specification is retained here.

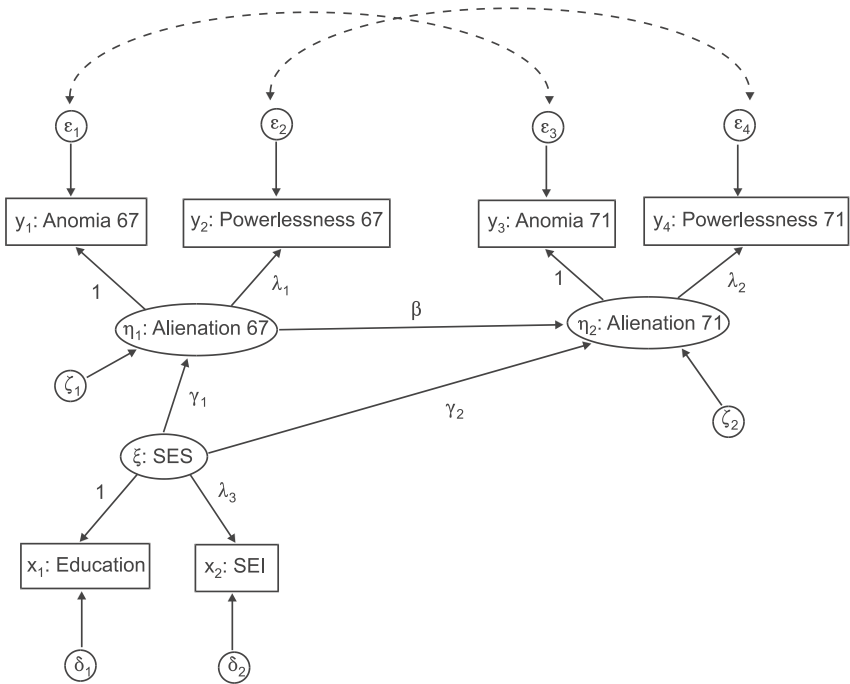


FIGURE 2 Conventional path diagram for Jöreskog and Sörbom’s model for the Wheaton alienation data. Adapted with permission from Figure 6.5 in Jöreskog and Sörbom (1989, 171).

Using the common Lisrel notation, this model consists of a structural submodel with two equations for the two latent endogenous variables (Alienation 67 and Alienation 71), and a measurement submodel with equations for the six indicators of the latent variables:

Structural submodel

$$\begin{aligned} \eta_1 &= \gamma_1 \xi + \zeta_1 \\ \eta_2 &= \gamma_2 \xi + \beta \eta_1 + \zeta_2 \end{aligned}$$

Measurement submodel

$$\begin{aligned} y_1 &= 1\eta_1 + \varepsilon_1 \\ y_2 &= \lambda_1 \eta_1 + \varepsilon_2 \\ y_3 &= 1\eta_2 + \varepsilon_3 \\ y_4 &= \lambda_2 \eta_2 + \varepsilon_4 \\ x_1 &= 1\xi + \delta_1 \\ x_2 &= \lambda_3 \xi + \delta_2 \end{aligned}$$

In these equations, the variables—observed and unobserved—are expressed as deviations from their expectations, suppressing the regression constant in each equation.³ The parameters of the model to be estimated include not just regression coefficients (i.e., structural parameters and factor loadings relating observed indicators to latent variables), but also the measurement-error variances, $V(\varepsilon_i) = \theta^{\varepsilon_{ii}}$, $V(\delta_j) = \theta^{\delta_{jj}}$; the variances of the structural disturbances, $V(\zeta_i) = \psi_{ii}$; the variance of the latent exogenous variable, $V(\xi) = \phi$; and, in some models considered next, certain measurement-error covariances $C(\varepsilon_i, \varepsilon_r) = \theta^{\varepsilon_{ir}}$. The 1s in the measurement submodel reflect normalizing restrictions, establishing the scales of the latent variables.

Internally, the `sem` function, which is used to fit general structural equation models in R, employs the recticular action model (RAM) formulation of the model, due to McArdle (1980) and McArdle and McDonald (1984), and it is therefore helpful to understand the structure of this model; the notation used here is from McDonald and Hartmann (1992).

In the RAM model, the vector v contains indicator variables, directly observed exogenous variables, and latent exogenous and endogenous variables; the vector u (which may overlap with v) contains directly observed and latent exogenous variables, measurement-error variables, and structural-error variables (i.e., the inputs to the system). Not all classes of variables are present in every model; for example, there are no directly observed exogenous variables in the Wheaton model.

The v and u vectors are related by the equation $v = Av + u$, and, therefore, the matrix A contains regression coefficients (both structural parameters and factor loadings). For example, for the Wheaton model, we have

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ x_1 \\ x_2 \\ \eta_1 \\ \eta_2 \\ \xi \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \lambda_1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_3 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \gamma_1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \beta & \gamma_2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ x_1 \\ x_2 \\ \eta_1 \\ \eta_2 \\ \xi \end{bmatrix} + \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \varepsilon_3 \\ \varepsilon_4 \\ \delta_1 \\ \delta_2 \\ \zeta_1 \\ \zeta_2 \\ \xi \end{bmatrix}$$

³A model with intercepts can be estimated by the `sem` function (described later) using a `raw` (i.e., uncorrected) moment matrix of mean sums of squares and cross-products in place of the covariance matrix among the observed variables in the model. This matrix includes sums of squares and products with a vector of ones, representing the constant regressor (see, e.g., McArdle & Epstein, 1987). The `raw.moments` function in the `sem` package will compute a raw-moments matrix from a model formula, numeric data frame, or numeric data matrix. To get correct degrees of freedom, set the argument `raw = TRUE` in `sem`.

As is typically the case, most of the entries of A are prespecified to be 0, whereas others are set to 1.

In the RAM formulation, the matrix P contains covariances among the elements of u . For the Wheaton model:

$$P = V \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \varepsilon_3 \\ \varepsilon_4 \\ \delta_1 \\ \delta_2 \\ \zeta_1 \\ \zeta_2 \\ \xi \end{bmatrix} = \begin{bmatrix} \theta^{\varepsilon_{11}} & 0 & \theta^{\varepsilon_{13}} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \theta^{\varepsilon_{22}} & 0 & \theta^{\varepsilon_{24}} & 0 & 0 & 0 & 0 & 0 \\ \theta^{\varepsilon_{13}} & 0 & \theta^{\varepsilon_{33}} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \theta^{\varepsilon_{24}} & 0 & \theta^{\varepsilon_{44}} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \theta^{\delta_{11}} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \theta^{\delta_{22}} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \psi_{11} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \psi_{12} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \phi \end{bmatrix}$$

Once again, as is typically true, the P matrix is very sparse.

Let m represent the number of variables in v , and let the first n entries of v be the observed variables of the model. Then the $m \times n$ selection matrix

$$J = \begin{bmatrix} I_n & 0 \\ 0 & 0 \end{bmatrix}$$

picks out the observed variables, where I_n is an order- n identity matrix and the 0s are zero matrices of appropriate order. Covariances among the observed variables are therefore given by

$$C = E(Jv v^T J^T) = J(I_m - A)^{-1} P [(I_m - A)^{-1}]^T J^T$$

Let S denote the covariances among the observed variables computed directly from a sample of data. Estimating the parameters of the model—the unconstrained entries of A and P —entails picking values of the parameters that make C close in some sense to S . In particular, under the assumption that the latent variables and errors are multnormally distributed, maximum likelihood (ML) estimates of the parameters minimize the fitting criterion

$$F(A, P) = \text{trace}(SC^{-1}) - n + \log_e \det C - \log_e \det S$$

The `sem` function minimizes the ML fitting criterion numerically using the `nlm` optimizer in R, which employs a Newton-type algorithm; `sem` by default uses an analytic gradient, but a numerical gradient may be optionally employed. The covariance matrix of the parameter estimates is based on the numerical Hessian re-

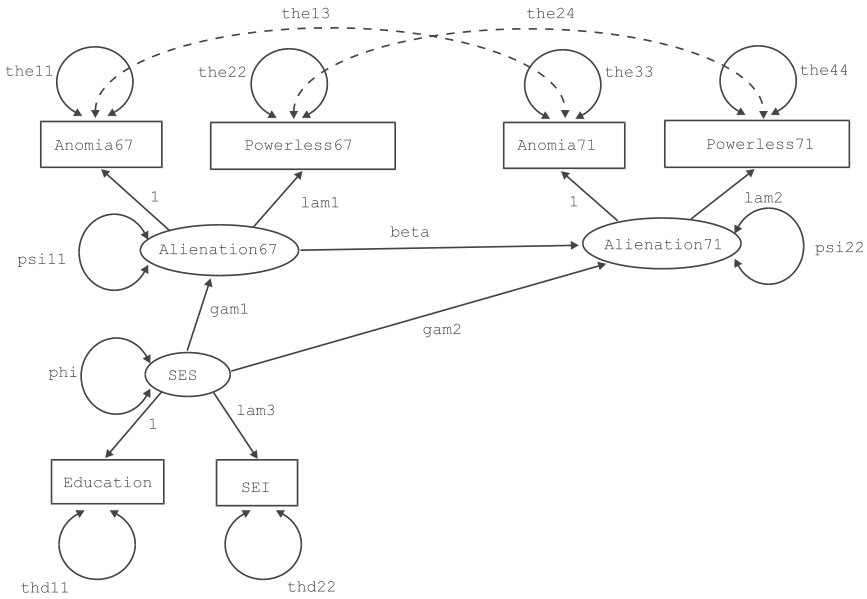


FIGURE 3 Modified RAM-format path diagram, with variables and parameters labeled for input to the `sem` function.

turn by `nlm`. Start values for the parameters may be supplied by the user or are computed by a slightly modified version of the approach given in McDonald and Hartmann (1992).

One advantage of the RAM formulation of a structural equation model is that the elements of the A and P matrices can be read off the path diagram for the model, with single-headed arrows corresponding to elements of A and double-headed arrows to elements of P , taking into account the fact that variances (as opposed to covariances) of exogenous variables and errors do not appear directly in the path diagram. To make the variances explicit, it helps to modify the path diagram slightly, as in Figure 3, eliminating the error variables and showing variances as self-directed double-headed arrows. The names of variables and free parameters have been replaced with names suitable for specifying the model in R.

Model specification in the `sem` package is handled most conveniently via the `specify.model` function:

```
> mod.wh.1 <- specify.model()
1: Alienation67 -> Anomia67,      NA,      1
2: Alienation67 -> Powerless67, lam1,    NA
3: Alienation71 -> Anomia71,     NA,      1
4: Alienation71 -> Powerless71, lam2,    NA
```

```

5:  SES           ->  Education,      NA,      1
6:  SES           ->  SEI,              lam3,   NA
7:  Alienation67 ->  Alienation71, beta,   NA
8:  SES           ->  Alienation67, gam1,   NA
9:  SES           ->  Alienation71, gam2,   NA
10: SES           <-> SES,              phi,    NA
11: Alienation67 <-> Alienation67, psi1,   NA
12: Alienation71 <-> Alienation71, psi2,   NA
13: Anomia67     <-> Anomia67,        the11,  NA
14: Powerless67 <-> Powerless67,        the22,  NA
15: Anomia71     <-> Anomia71,        the33,  NA
16: Powerless71 <-> Powerless71,        the44,  NA
17: Education    <-> Education,      thd1,   NA
18: SEI          <-> SEI,              thd2,   NA
19:
Read 18 records

```

This specification is largely self-explanatory, but note the following:

- The line-number prompts are supplied by `specify.model`, which is called here without any arguments. The entries are terminated by a blank line. The `specify.model` function can also read the model specification from a text file.
- There are three entries in each line, separated by commas.
- A single-headed arrow in the first entry indicates a regression coefficient and corresponds to a single-headed arrow in the path diagram; likewise a double-headed arrow represents a variance or covariance and corresponds to a double-headed arrow in the modified path diagram in Figure 3 (disregarding for now the broken arrows labeled `the13` and `the24` in the diagram).
- The second entry in each line gives the (arbitrary) name of a free parameter to be estimated. Entering the name `NA` (missing) indicates that a parameter is to be fixed to a particular value. Assigning the same name to two or more arrows establishes an equality constraint between the corresponding parameters. For example, substituting `lam` for both `lam1` and `lam2` would imply that these two factor loadings are represented by the same parameter and hence are equal.
- The third entry in each line either assigns a value to a fixed parameter or sets a start value for a free parameter; in the latter case, entering `NA` causes `sem` to pick the start value.
- Finally, a word of caution: A common error in specifying models is to omit double-headed arrows representing variances of exogenous variables or error variances.

To estimate the model, the covariance or raw-moment matrix among the observed variables has to be computed. In the case of the Wheaton data, the

covariances rather than the original data are available, and consequently the covariance matrix is entered directly:

```
> S.wh <- matrix(c(
+ 11.834, 0, 0, 0, 0, 0,
+ 6.947, 9.364, 0, 0, 0, 0,
+ 6.819, 5.091, 12.532, 0, 0, 0,
+ 4.783, 5.028, 7.495, 9.986, 0, 0,
+ -3.839, -3.889, -3.841, -3.625, 9.610, 0,
+ -21.899, -18.831, -21.748, -18.775, 35.522, 450.288),
+ 6, 6, byrow=TRUE)
>
> rownames(S.wh) <- colnames(S.wh) <-
+ c('Anomia67', 'Powerless67', 'Anomia71', 'Powerless71', 'Education', 'SEI')
```

The covariance matrix has been entered in lower-triangular form: `sem` will accept a lower triangular, upper triangular, or symmetric covariance matrix. One can either assign the names of the observed variables to the rows and columns of the input covariance matrix, as done here, or pass these names directly to `sem`; in either event, variables in the model specification that do not appear in the input covariance matrix are assumed by `sem` to be latent variables. One must therefore be careful in typing these names, because the misspelled name of an observed variable is interpreted as a latent variable, producing an erroneous model.

To estimate the model:

```
> sem.wh.1 <- sem(mod.wh.1, S.wh, N=932)
> summary(sem.wh.1)
```

```
Model Chisquare = 71.47 Df = 6 Pr(>Chisq) = 2.0417e-13
Goodness-of-fit index = 0.97517
Adjusted goodness-of-fit index = 0.91309
RMSEA index = 0.10826 90 % CI: (0.086585, 0.13145)
BIC = 19.695
```

Normalized Residuals

```
Min. 1st Qu. Median Mean 3rd Qu. Max.
-1.26e+00 -2.12e-01 -8.35e-05 -1.53e-02 2.44e-01 1.33e+00
```

Parameter Estimates

```
Estimate Std Error z value Pr(>|z|)
lam1 0.88854 0.043196 20.5699 0.0000e+00 Powerless67 <-- Alienation67
lam2 0.84872 0.041560 20.4215 0.0000e+00 Powerless71 <-- Alienation71
lam3 5.32898 0.430955 12.3655 0.0000e+00 SEI <-- SES
beta 0.70471 0.053393 13.1985 0.0000e+00 Alienation71 <-- Alienation67
gam1 -0.61382 0.056270 -10.9084 0.0000e+00 Alienation67 <-- SES
gam2 -0.17419 0.054244 -3.2113 1.3216e-03 Alienation71 <-- SES
phi 6.66585 0.642394 10.3766 0.0000e+00 SES <--> SES
```

```

psi1    5.30697  0.484105  10.9624  0.0000e+00  Alienation67 <-> Alienation67
psi2    3.74127  0.388844   9.6215  0.0000e+00  Alienation71 <-> Alienation71
the11   4.01554  0.358989  11.1857  0.0000e+00  Anomia67 <-> Anomia67
the22   3.19131  0.283900  11.2410  0.0000e+00  Powerless67 <-> Powerless67
the33   3.70111  0.391894   9.4441  0.0000e+00  Anomia71 <-> Anomia71
the44   3.62481  0.304365  11.9094  0.0000e+00  Powerless71 <-> Powerless71
thd1    2.94419  0.501395   5.8720  4.3057e-09  Education <-> Education
thd2    260.99237 18.278663  14.2785  0.0000e+00  SEI <-> SEI
Iterations = 83

```

- The first argument to `sem` is the model-specification object returned by `specify.model`.
- The second argument, `S.wh`, is the input covariance matrix.
- The third argument is the number of observations on which the covariances are based.
- There are other optional arguments, which are explained in the `sem` help page (type `?sem` to see it). One other argument is worth mentioning here: `fixed.x` takes a vector of quoted names of fixed exogenous variables, obviating the tedious necessity of enumerating the variances and covariances among these variables in the model specification. In this case, there are no fixed exogenous variables.

As is typical of R programs, `sem` returns an object rather than a printed report. The summary method for `sem` objects produces the printout shown previously. One can perform additional computations on `sem` objects, for example, producing various kinds of residual covariances or modification indexes⁴ (e.g., Sörbom, 1989):

```
> mod.indices(sem.wh.1)
```

```

5 largest modification indices, A matrix:
  Anomia71:Anomia67   Anomia67:Anomia71  Powerless71:Anomia67
           10.378667             9.086581             9.024746
Anomia67:Powerless71  Powerless67:Anomia71
           7.712109             7.285830

5 largest modification indices, P matrix:
  Anomia71:Anomia67  Powerless71:Anomia67  Anomia71:Powerless67
           40.944815             35.377847             32.051155
Powerless71:Powerless67  Education:Powerless67
           26.512979             5.878679

```

⁴The modification indexes returned by `mod.indices` are slightly in error because of an apparent bug that has to this point been elusive.

Simply allow the object returned by `mod.indices` to be printed, which produces a brief report; the `summary` method for these objects produces a more complete report, showing all modification indexes along with approximations to the estimates that would result were each omitted parameter included in the model.

Recall that the A matrix contains regression coefficients whereas the P matrix contains covariances. The modification indexes suggest that a better fit to the data would be achieved by freeing one or more of the covariances among the measurement errors of the latent endogenous variables; the largest modification index is for the two anomia measures, corresponding to the broken arrow labeled `the13` in Figure 3. Adding this parameter to the model produces a much better fit (but subsequently adding a parameter for the measurement error covariance `the24`, not shown, yields little additional improvement):

```
> mod.wh.2 <- specify.model()
1:  Alienation67  ->  Anomia67,      NA,      1
2:  Alienation67  ->  Powerless67, lam1,   NA
3:  Alienation71  ->  Anomia71,      NA,      1
4:  Alienation71  ->  Powerless71, lam2,   NA
5:  SES           ->  Education,   NA,      1
6:  SES           ->  SEI,         lam3,   NA
7:  Alienation67  ->  Alienation71, beta,   NA
8:  SES           ->  Alienation67, gam1,   NA
9:  SES           ->  Alienation71, gam2,   NA
10: SES           <-> SES,         phi,    NA
11: Alienation67  <-> Alienation67, psi1,   NA
12: Alienation71  <-> Alienation71, psi2,   NA
13: Anomia67      <-> Anomia67,    the11,  NA
14: Powerless67   <-> Powerless67, the22,  NA
15: Anomia71      <-> Anomia71,    the33,  NA
16: Powerless71   <-> Powerless71, the44,  NA
17: Anomia67      <-> Anomia71,    the13,  NA
18: Education     <-> Education,   thd1,   NA
19: SEI           <-> SEI,         thd2,   NA
20:
Read 19 records

> sem.wh.2 <- sem(mod.wh.2, S.wh, 932)
> summary(sem.wh.2)

Model Chisquare = 6.3307 Df = 5 Pr(>Chisq) = 0.27536
Goodness-of-fit index = 0.99773
Adjusted goodness-of-fit index = 0.99047
RMSEA index = 0.016908 90 % CI: (NA, 0.050905)
BIC = -36.815
```



```

Normalized Residuals
  Min.   1st Qu.   Median     Mean   3rd Qu.    Max.
-9.57e-01 -1.34e-01 -4.26e-02 -9.17e-02  1.82e-05  5.46e-01

```

```

Parameter Estimates
      Estimate Std Error z value Pr(>|z|)
lam1  1.02653  0.053421  19.2159 0.0000e+00 Powerless67 <-- Alienation67
lam2  0.97092  0.049611  19.5708 0.0000e+00 Powerless71 <-- Alienation71
lam3  5.16275  0.422382  12.2229 0.0000e+00 SEI <-- SES
beta  0.61734  0.049483  12.4759 0.0000e+00 Alienation71 <-- Alienation67
gam1  -0.54975  0.054290 -10.1262 0.0000e+00 Alienation67 <-- SES
gam2  -0.21143  0.049850  -4.2413 2.2220e-05 Alienation71 <-- SES
phi   6.88047  0.659207  10.4375 0.0000e+00 SES <--> SES
psii  4.70519  0.427546  11.0051 0.0000e+00 Alienation67 <--> Alienation67
psi2  3.86633  0.343950  11.2410 0.0000e+00 Alienation71 <--> Alienation71
the11 5.06528  0.373450  13.5635 0.0000e+00 Anomia67 <--> Anomia67
the22 2.21457  0.319728   6.9264 4.3163e-12 Powerless67 <--> Powerless67
the33 4.81202  0.397357  12.1101 0.0000e+00 Anomia71 <--> Anomia71
the44 2.68302  0.331280   8.0989 4.4409e-16 Powerless71 <--> Powerless71
the13 1.88740  0.241630   7.8111 5.7732e-15 Anomia71 <--> Anomia67
thd1  2.72956  0.517741   5.2721 1.3490e-07 Education <--> Education
thd2 266.89704 18.225397  14.6442 0.0000e+00 SEI <--> SEI

```

```
Iterations = 87
```

Bootstrapping a Simple One-Factor Model for Ordinal Variables

An advantage of working in an extensive system of statistical software is that one can leverage other capabilities of the software. In this example, facilities in the `boot` package (which is associated with Davidson & Hinkley, 1997, and is part of the standard R distribution) are used to bootstrap a model fit by `sem`, and in my `polycor` package (a contributed package on CRAN) to compute polychoric correlations among ordinal variables.⁵ Indeed, the integration of the `sem` package with R is more generally advantageous: For example, there are several packages available for multiple imputation of missing data, and it would be a simple matter to use these with `sem`.

The CNES data frame distributed with the `sem` package includes four variables from the 1997 Canadian National Election Study (CNES) meant to tap attitude toward “traditional values.” These variables are four-category Likert-type items, and appear in the data as factors (the S representation of categorical variables), with levels `StronglyDisagree`, `Disagree`, `Agree`, and `StronglyAgree`.

⁵I am the author of the `polycor` package, which is distributed on the same basis as the `sem` package, and as R itself; that is, as free, open-source software.

These variables originated as responses to the following statements on the mail-back component of the election study:

MBSA2: “We should be more tolerant of people who choose to live according to their own standards, even if they are very different from our own.”

MBSA7: “Newer lifestyles are contributing to the breakdown of our society.”

MBSA8: “The world is always changing and we should adapt our view of moral behaviour to these changes.”

MBSA9: “This country would have many fewer problems if there were more emphasis on traditional family values.”

The `hetcor` function in the `polycor` package computes heterogeneous correlation matrices among ordinal and numeric variables: the product–moment correlation between two numeric variables, the polychoric correlation between two factors (assumed to be properly ordered), and the point–polyserial correlation between a factor and a numeric variable. For the CNES data, for example:

```
> data(CNES)
> CNES[1:5,] # first 5 observations
      MBSA2      MBSA7      MBSA8      MBSA9
1 StronglyAgree      Agree      Disagree      Disagree
2      Agree      StronglyAgree StronglyDisagree StronglyAgree
3      Agree      Disagree      Disagree      Agree
4 StronglyAgree      Agree StronglyDisagree StronglyAgree
5      Agree StronglyDisagree      Agree      Disagree

> library(polycor)
> hetcor(CNES, ML=TRUE)
```

Maximum-Likelihood Estimates

Correlations/Type of Correlation:

	MBSA2	MBSA7	MBSA8	MBSA9
MBSA2	1	Polychoric	Polychoric	Polychoric
MBSA7	-0.3028	1	Polychoric	Polychoric
MBSA8	0.2826	-0.344	1	Polychoric
MBSA9	-0.2229	0.5469	-0.3213	1

Standard Errors:

	MBSA2	MBSA7	MBSA8
MBSA2			
MBSA7	0.02737		
MBSA8	0.02773	0.02642	
MBSA9	0.02901	0.02193	0.02742

n = 1529

P-values for Tests of Bivariate Normality:

	MBSA2	MBSA7	MBSA8
MBSA2			
MBSA7	1.277e-07		
MBSA8	1.852e-07	2.631e-23	
MBSA9	5.085e-09	2.356e-10	1.500e-19

By default, the `hetcor` function computes polychoric and polyserial correlations by a relatively quick two-step procedure (see Drasgow, 1986; Olsson, 1979); specifying the argument `ML=TRUE` causes `hetcor` to compute pairwise ML estimates instead; in this instance (and as is typically the case), the two procedures produce very similar results (see later), so the faster procedure for bootstrapping was used. The tests of bivariate normality, applied to the contingency table for each pair of variables, are highly statistically significant, indicating departures from binormality. Even though a nonparametric bootstrap is employed later, non-normality suggests that it might not be appropriate to summarize the relations between the variables with correlations; on the other hand, the sample size ($N = 1,529$) is fairly large, making these tests quite sensitive.

The `hetcor` function returns an object with correlations and other information, but for fitting and bootstrapping a structural equation model, only the correlation matrix is desired. The following simple function (from the `boot.sem` help page), entered at the command prompt, does the trick:

```
> hcor <- function(data) hetcor(data, std.err=FALSE)$correlations
> R.CNES <- hcor(CNES)
> R.CNES
```

	MBSA2	MBSA7	MBSA8	MBSA9
MBSA2	1.0000000	-0.3017953	0.2820608	-0.2230010
MBSA7	-0.3017953	1.0000000	-0.3422176	0.5449886
MBSA8	0.2820608	-0.3422176	1.0000000	-0.3206524
MBSA9	-0.2230010	0.5449886	-0.3206524	1.0000000

Using `sem` to fit a one-factor confirmatory factor analysis model to the polychoric correlations produces these results:

```
> model.CNES <- specify.model()
1: F -> MBSA2, lam1, NA
2: F -> MBSA7, lam2, NA
3: F -> MBSA8, lam3, NA
4: F -> MBSA9, lam4, NA
5: F <-> F, NA, 1
6: MBSA2 <-> MBSA2, the1, NA
7: MBSA7 <-> MBSA7, the2, NA
8: MBSA8 <-> MBSA8, the3, NA
9: MBSA9 <-> MBSA9, the4, NA
10:
Read 9 records
```

```
> sem.CNES <- sem(model.CNES, R.CNES, N=1529)
> summary(sem.CNES)
```

```
Model Chisquare = 33.211 Df = 2 Pr(>Chisq) = 6.1407e-08
Goodness-of-fit index = 0.98934
Adjusted goodness-of-fit index = 0.94668
RMSEA index = 0.10106 90 % CI: (0.07261, 0.13261)
BIC = 15.774
```

```
Normalized Residuals
  Min. 1st Qu. Median Mean 3rd Qu. Max.
-1.76e-05 3.00e-02 2.08e-01 8.48e-01 1.04e+00 3.83e+00
```

```
Parameter Estimates
  Estimate Std Error z value Pr(>|z|)
lam1 -0.38933 0.028901 -13.471 0 MBSA2 <--- F
lam2 0.77792 0.029357 26.498 0 MBSA7 <--- F
lam3 -0.46868 0.028845 -16.248 0 MBSA8 <--- F
lam4 0.68680 0.028409 24.176 0 MBSA9 <--- F
the1 0.84842 0.032900 25.788 0 MBSA2 <--> MBSA2
the2 0.39485 0.034436 11.466 0 MBSA7 <--> MBSA7
the3 0.78033 0.031887 24.472 0 MBSA8 <--> MBSA8
the4 0.52831 0.030737 17.188 0 MBSA9 <--> MBSA9
Iterations = 12
```

The model fit here is

$$x_i = \lambda_i \xi + \delta_i, i = 1, \dots, 4$$

$$V(\xi) = 1$$

$$V(\delta_i) = \theta_{ii}, i = 1, \dots, 4$$

with ξ represented by F (for factor) in the `sem` specification of the model.

Using the ML fitting criterion with polychoric correlations produces consistent estimators of the parameters of the model (e.g., Bollen, 1989, p. 443), but the standard errors of the estimators cannot be trusted. Consequently, the `boot.sem` function is used to compute bootstrap standard errors⁶:

```
> system.time(boot.CNES <- boot.sem(CNES, sem.CNES, R=100, cov=hcov),
+ gcFirst=TRUE)
Loading required package: boot
[1] 113.16 0.08 113.77 NA NA

> summary(boot.CNES, type="norm")
Call: boot.sem(data = CNES, model = sem.CNES, R = 100, cov = hcov)
```

⁶`boot.sem` implements the nonparametric bootstrap for an independent random sample; for more complex sampling schemes or parametric bootstrapping, use the facilities of the `boot` package directly.

Lower and upper limits are for the 95 percent norm confidence interval

	Estimate	Bias	Std.Error	Lower	Upper
lam1	-0.3893278	0.003473333	0.03203950	-0.4555973	-0.3300048
lam2	0.7779153	0.008113908	0.03728210	0.6967299	0.8428730
lam3	-0.4686838	0.007162961	0.03258827	-0.5397186	-0.4119749
lam4	0.6867992	-0.001208898	0.03160963	0.6260544	0.7499619
the1	0.8484245	0.001675460	0.02456932	0.7985941	0.8949041
the2	0.3948479	-0.014065971	0.05897530	0.2933244	0.5245034
the3	0.7803349	0.005612135	0.03004934	0.7158271	0.8336184
the4	0.5283057	0.000671089	0.04377695	0.4418334	0.6134359

The `system.time` function was used to time the computation, which took 114 sec on a 3 GHz Windows XP machine; the argument `gcFirst=TRUE` specifies that “garbage collection” take place just before the command is executed, producing a more accurate timing. Notice that `boot.sem` automatically loads the `boot` package. The arguments to `boot.sem` include the data set to be resampled (CNES), the `sem` object for the model (`sem.CNES`), the number of bootstrap replications ($R=100$, which should be sufficient for standard errors and normal-theory confidence intervals), and the function to be used in computing a covariance matrix from the resampled data (here, the `hcor` function). The bootstrap standard errors are mostly somewhat larger than the standard errors assuming multinormal numeric variables computed by `sem`.

FURTHER DEVELOPMENT OF THE `sem` PACKAGE

The latent variable modeling facility provided by the `sem` function is relatively basic compared to special-purpose structural equation software such as AMOS, EQS, LISREL, or *Mplus*. One possible future direction for the `sem` package, therefore, would be to expand capabilities in areas such as multiple-group models and alternative fitting functions. Some enhancements—for example, multiple-group model—should be relatively straightforward, whereas others—for example, Browne’s (1984) asymptotically distribution-free estimator—would likely require implementation in compiled code to achieve acceptable levels of performance. At present, the `sem` package is coded entirely in R, but R makes provisions for the incorporation of portable compiled code in C and Fortran.

A second area in which the `sem` package could be improved is the user interface: It would be desirable to provide a graphical interface in which the user specifies a model via its path diagram. Currently, the `path.diagram` function in `sem` provides only the inverse facility, producing a description of the path diagram from a fitted model, which subsequently can be rendered by the graph-drawing program *dot* (Gansner, Koutsofios, & North, 2002). Although R provides tools for the con-

struction of graphical interfaces, making a path-drawing interface feasible, implementing such an interface would be a substantial undertaking.

The specific future trajectory of the `sem` package, and the rapidity with which it is developed, will depend on user interest.

REFERENCES

- Becker, R. A., Chambers, J. M., & Wilks, A. R. (1988). *The new S language: A programming environment for data analysis and graphics*. Pacific Grove, CA: Wadsworth.
- Bollen, K. A. (1989). *Structural equations with latent variables*. New York: Wiley.
- Browne, M. W. (1984). Asymptotically distribution-free methods for the analysis of covariance structures. *British Journal of Mathematical and Statistical Psychology*, 37, 62–83.
- Chambers, J. M. (1998). *Programming with data: A guide to the S language*. New York: Springer.
- Chambers, J. M., & Hastie, T. J. (Eds.). (1992). *Statistical models in S*. Pacific Grove, CA: Wadsworth.
- Davison, A. C., & Hinkley, D. V. (1997). *Bootstrap methods and their application*. Cambridge, UK: Cambridge University Press.
- Drasgow, F. (1986). Polychoric and polyserial correlations. In S. Kotz & N. Johnson (Eds.), *The encyclopedia of statistics* (Vol. 7, pp. 68–74). New York: Wiley.
- Fox, J. (2002). *An R and S-PLUS companion to applied regression*. Thousand Oaks, CA: Sage.
- Gansner, E., Koutsofios, E., & North, S. (2002). *Drawing graphs with dot*. Retrieved June 13, 2005, from <http://www.graphviz.org/Documentation/dotguide.pdf>
- Greene, W. H. (1993). *Econometric analysis* (2nd ed.). New York: Macmillan.
- Ihaka, R., & Gentleman, R. (1996). R: A language for data analysis and graphics. *Journal of Computational and Graphical Statistics*, 5, 299–314.
- Jöreskog, K. G., & Sörbom, D. (1989). *LISREL 7: A guide to the program and applications* (2nd ed.). Chicago: SPSS.
- Klein, L. (1950). *Economic fluctuations in the United States 1921–1941*. New York: Wiley.
- McArdle, J. J. (1980). Causal modeling applied to psychonomic systems simulation. *Behavior Research Methods and Instrumentation*, 12, 193–209.
- McArdle, J. J., & Epstein, D. (1987). Latent growth curves within developmental structural equation models. *Child Development*, 58, 110–133.
- McArdle, J. J., & McDonald, R. P. (1984). Some algebraic properties of the reticular action model. *British Journal of Mathematical and Statistical Psychology*, 37, 234–251.
- McDonald, R. P., & Hartmann, W. M. (1992). A procedure for obtaining initial values of parameters in the RAM model. *Multivariate Behavioral Research*, 27, 57–76.
- Olsson, U. (1979). Maximum likelihood estimation of the polychoric correlation coefficient. *Psychometrika*, 44, 443–460.
- R Development Core Team. (2005). *R: A language and environment for statistical computing*. Vienna, Austria: R Foundation for Statistical Computing.
- SAS Institute. (2004). *SAS 9.1.3 help and documentation*. Cary, NC: Author.
- Sörbom, D. (1989). Model modification. *Psychometrika*, 54, 371–384.
- Venables, W. N., & Ripley, B. D. (2002). *Modern applied statistics with S* (4th ed.). New York: Springer.
- Wheaton, B., Muthén, B., Alwin, D. F., & Summers, G. F. (1977). Assessing reliability and stability in panel models. In D. R. Heise (Ed.), *Sociological methodology 1977* (pp. 84–136). San Francisco: Jossey-Bass.
- Wilkinson, G. N., & Rogers, C. E. (1973). Symbolic description of factorial models for analysis of variance. *Applied Statistics*, 22, 392–399.