# systemfit: A Package for Estimating Systems of Simultaneous Equations in R

**Arne Henningsen**
University of Copenhagen

**Jeff D. Hamann**
Forest Informatics, Inc.

**Abstract**

This introduction to the R package **systemfit** is a slightly modified version of Henningsen and Hamann (2007), published in the *Journal of Statistical Software*.

Many statistical analyses (e.g., in econometrics, biostatistics and experimental design) are based on models containing systems of structurally related equations. The **systemfit** package provides the capability to estimate systems of linear equations within the R programming environment. For instance, this package can be used for "ordinary least squares" (OLS), "seemingly unrelated regression" (SUR), and the instrumental variable (IV) methods "two-stage least squares" (2SLS) and "three-stage least squares" (3SLS), where SUR and 3SLS estimations can optionally be iterated. Furthermore, the **systemfit** package provides tools for several statistical tests. It has been tested on a variety of datasets and its reliability is demonstrated.

*Keywords*: R, system of simultaneous equations, seemingly unrelated regression, two-stage least squares, three-stage least squares, instrumental variables.

## 1. Introduction

Many theoretical models that are econometrically estimated consist of more than one equation. The disturbance terms of these equations are likely to be contemporaneously correlated, because unconsidered factors that influence the disturbance term in one equation probably influence the disturbance terms in other equations, too. Ignoring this contemporaneous correlation and estimating these equations separately leads to inefficient estimates of the coefficients. However, estimating all equations simultaneously with a "generalized least squares" (GLS) estimator, which takes the covariance structure of the residuals into account, leads to efficient estimates. This estimation procedure is generally called "seemingly unrelated regression" (SUR, Zellner 1962). Another reason to estimate a system of equations simultaneously are cross-equation restrictions on the coefficients.[1] Estimating the coefficients under cross-equation restrictions and testing these restrictions requires a simultaneous estimation approach.

Furthermore, these models can contain variables that appear on the left-hand side in one equation and on the right-hand side of another equation. Ignoring the endogeneity of these variables can lead to inconsistent estimates. This simultaneity bias can be corrected for by

[1] Especially the economic theory suggests many cross-equation restrictions on the coefficients (e.g., the symmetry restriction in demand models).

applying a "two-stage least squares" (2SLS) estimation to each equation. Combining this estimation method with the SUR method results in a simultaneous estimation of the system of equations by the "three-stage least squares" (3SLS) method (Zellner and Theil 1962).

The **systemfit** package provides the capability to estimate systems of linear equations in R (R Development Core Team 2007). Currently, the estimation methods "ordinary least squares" (OLS), "weighted least squares" (WLS), "seemingly unrelated regression" (SUR), "two-stage least squares" (2SLS), "weighted two-stage least squares" (W2SLS), and "three-stage least squares" (3SLS) are implemented.[2] The WLS, SUR, W2SLS, and 3SLS estimates can be based either on one-step (OLS or 2SLS) (co)variances or these estimations can be iterated, where the (co)variances are calculated from the estimates of the previous step. Furthermore, the **systemfit** package provides statistical tests for restrictions on the coefficients and for testing the consistency of the 3SLS estimation.

Although systems of linear equations can be estimated with several other statistical and econometric software packages (e.g., SAS, EViews, TSP), **systemfit** has several advantages. First, all estimation procedures are publicly available in the source code. Second, the estimation algorithms can be easily modified to meet specific requirements. Third, the (advanced) user can control estimation details generally not available in other software packages by overriding reasonable defaults.

In Section 2 we introduce the statistical background of estimating equation systems. The implementation of the statistical procedures in R is briefly explained in Section 3. Section 4 demonstrates how to run **systemfit** and how some of the features presented in the second section can be used. In Section 5 we replicate several textbook results with the **systemfit** package. Finally, a summary and outlook are presented in Section 6.

## 2. Statistical background

In this section we give a short overview of the statistical background that the **systemfit** package is based on. More detailed descriptions of simultaneous equations systems are available for instance in Theil (1971, Chapter 7), Judge, Hill, Griffiths, Lütkepohl, and Lee (1982, Part 4), Judge, Griffiths, Hill, Lütkepohl, and Lee (1985, Part 5), Srivastava and Giles (1987), Greene (2003, Chapters 14–15), and Zivot and Wang (2006, Chapter 10).

After introducing notations and assumptions, we provide the formulas to estimate systems of linear equations. We then demonstrate how to estimate coefficients under linear restrictions. Finally, we present additional relevant issues about estimation of equation systems.

Consider a system of $G$ equations, where the $i$th equation is of the form

$$y_i = X_i\beta_i + u_i, \quad i = 1, 2, \ldots, G, \tag{1}$$

where $y_i$ is a vector of the dependent variable, $X_i$ is a matrix of the exogenous variables, $\beta_i$ is the coefficient vector and $u_i$ is a vector of the disturbance terms of the $i$th equation.

[2] In this context, the term "weighted" in "weighted least squares" (WLS) and "weighted two-stage least squares" (W2SLS) means that the *equations* might have different weights and *not* that the *observations* have different weights.

We can write the "stacked" system as

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_G \end{bmatrix} = \begin{bmatrix} X_1 & 0 & \cdots & 0 \\ 0 & X_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & X_G \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_G \end{bmatrix} + \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_G \end{bmatrix} \tag{2}$$

or more simply as

$$y = X\beta + u. \tag{3}$$

We assume that there is no correlation of the disturbance terms across observations, so that

$$\mathsf{E}\left[u_{it}\, u_{js}\right] = 0 \;\forall\; t \neq s, \tag{4}$$

where $i$ and $j$ indicate the equation number and $t$ and $s$ denote the observation number, where the number of observations is the same for all equations.

However, we explicitly allow for contemporaneous correlation, i.e.,

$$\mathsf{E}\left[u_{it}\, u_{jt}\right] = \sigma_{ij}. \tag{5}$$

Thus, the covariance matrix of all disturbances is

$$\mathsf{E}\left[u\, u^\top\right] = \Omega = \Sigma \otimes I_T, \tag{6}$$

where $\Sigma = [\sigma_{ij}]$ is the (contemporaneous) disturbance covariance matrix, $\otimes$ is the Kronecker product, $I_T$ is an identity matrix of dimension $T$, and $T$ is the number of observations in each equation.

## 2.1. Estimation with only exogenous regressors

If all regressors are exogenous, the system of equations (Equation 1) can be consistently estimated by ordinary least squares (OLS), weighted least squares (WLS), and seemingly unrelated regression (SUR). These estimators can be obtained by

$$\hat{\beta} = \left(X^\top \widehat{\Omega}^{-1} X\right)^{-1} X^\top \widehat{\Omega}^{-1} y. \tag{7}$$

The covariance matrix of these estimators can be estimated by

$$\widehat{\mathsf{COV}}\left[\hat{\beta}\right] = \left(X^\top \widehat{\Omega}^{-1} X\right)^{-1}. \tag{8}$$

*Ordinary least squares (OLS)*

The ordinary least squares (OLS) estimator is based on the assumption that the disturbance terms are not contemporaneously correlated ($\sigma_{ij} = 0 \;\forall\; i \neq j$) and have the same variance in each equation ($\sigma_i^2 = \sigma_j^2 \;\forall\, i, j$). In this case, $\widehat{\Omega}$ in Equation 7 is equal to $I_{G.T}$ and thus, cancels out. The OLS estimator is efficient, as long as the disturbances are not contemporaneously correlated.

If the whole system is treated as one single equation, $\widehat{\Omega}$ in Equation 8 is $\hat{\sigma}^2 I_{G \cdot T}$, where $\hat{\sigma}^2$ is an estimator for the variance of all disturbances ($\sigma^2 = \mathsf{E}[u_{it}^2]$). If the disturbance terms of the individual equations are allowed to have different variances, $\widehat{\Omega}$ in Equation 8 is $\widehat{\Sigma} \otimes I_T$, where $\hat{\sigma}_{ij} = 0 \; \forall \; i \neq j$ and $\hat{\sigma}_{ii} = \hat{\sigma}_i^2$ is the estimated variance of the disturbance term in the $i$th equation.

If the estimated coefficients are not constrained by cross-equation restrictions, the simultaneous OLS estimation of the system leads to the same estimated coefficients as an equation-wise OLS estimation. The covariance matrix of the coefficients from an equation-wise OLS estimation is equal to the covariance matrix obtained by Equation 8 with $\widehat{\Omega}$ equal to $\widehat{\Sigma} \otimes I_T$.

*Weighted least squares (WLS)*

The weighted least squares (WLS) estimator allows for different variances of the disturbance terms in the different equations ($\sigma_i^2 \neq \sigma_j^2 \, \forall \, i \neq j$), but assumes that the disturbance terms are not contemporaneously correlated. In this case, $\widehat{\Omega}$ in Equations 7 and 8 is $\widehat{\Sigma} \otimes I_T$, where $\hat{\sigma}_{ij} = 0 \; \forall \; i \neq j$ and $\hat{\sigma}_{ii} = \hat{\sigma}_i^2$ is the estimated variance of the disturbance terms in the $i$th equation. Theoretically, $\hat{\sigma}_{ii}$ should be the variance of the (true) disturbances ($\sigma_{ii}$). However, they are not known in most empirical applications. Therefore, true variances are generally replaced by estimated variances ($\hat{\sigma}_{ii}$) that are calculated from the residuals of a first-step OLS estimation (see Section 2.4).[3]

The WLS estimator is (asymptotically) efficient only if the disturbance terms are not contemporaneously correlated. If the estimated coefficients are not constrained by cross-equation restrictions, they are equal to OLS estimates.

*Seemingly unrelated regression (SUR)*

If the disturbances are contemporaneously correlated, a generalized least squares (GLS) estimation leads to an efficient estimator for the coefficients. In this case, the GLS estimator is generally called "seemingly unrelated regression" (SUR) estimator (Zellner 1962). However, the true covariance matrix of the disturbance terms is generally unknown. The textbook solution for this problem is a feasible generalized least squares (FGLS) estimation. As the FGLS estimator is based on an estimated covariance matrix of the disturbance terms, it is only asymptotically efficient. In case of a SUR estimator, $\widehat{\Omega}$ in Equations 7 and 8 is $\widehat{\Sigma} \otimes I_T$, where $\widehat{\Sigma}$ is the estimated covariance matrix of the disturbance terms.

It should be noted that while an unbiased OLS or WLS estimation requires only that the regressors and the disturbance terms of each single equation are uncorrelated ($\mathsf{E}\left[u_i^\top X_i\right] = 0 \; \forall \; i$), a consistent SUR estimation requires that all disturbance terms and all regressors are uncorrelated ($\mathsf{E}\left[u_i^\top X_j\right] = 0 \; \forall \; i, j$).

## 2.2. Estimation with endogenous regressors

If the regressors of one or more equations are correlated with the disturbances ($\mathsf{E}\left[u_i^\top X_i\right] \neq 0$), OLS, WLS, and SUR estimates are biased. This can be circumvented by a two-stage least

[3]Note that $\widehat{\Omega}$ in Equation 7 is not the same $\widehat{\Omega}$ as in Equation 8. The first is calculated from the residuals of a first-step OLS estimation; the second is calculated from the residuals of this WLS estimation. The same applies to the SUR, W2SLS, and 3SLS estimations described in the following sections.

squares (2SLS), weighted two-stage least squares (W2SLS), or a three-stage least squares (3SLS) estimation with instrumental variables (IV). The instrumental variables for each equation $Z_i$ can be either different or identical for all equations. They must not be correlated with the disturbance terms of the corresponding equation ($\mathsf{E}\left[u_i^\top Z_i\right] = 0$).

At the first stage new ("fitted") regressors are obtained by

$$\widehat{X}_i = Z_i \left(Z_i^\top Z_i\right)^{-1} Z_i^\top X_i. \tag{9}$$

Then, these "fitted" regressors are substituted for the original regressors in Equation 7 to obtain unbiased 2SLS, W2SLS, or 3SLS estimates of $\beta$ by

$$\hat{\beta} = \left(\widehat{X}^\top \widehat{\Omega}^{-1} \widehat{X}\right)^{-1} \widehat{X}^\top \widehat{\Omega}^{-1} y, \tag{10}$$

where

$$\widehat{X} = \begin{bmatrix} \widehat{X}_1 & 0 & \cdots & 0 \\ 0 & \widehat{X}_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \widehat{X}_G \end{bmatrix}. \tag{11}$$

An estimator of the covariance matrix of the estimated coefficients can be obtained from Equation 8 analogously. Hence, we get

$$\widehat{\mathsf{COV}}\left[\hat{\beta}\right] = \left(\widehat{X}^\top \widehat{\Omega}^{-1} \widehat{X}\right)^{-1}. \tag{12}$$

*Two-stage least squares (2SLS)*

The two-stage least squares (2SLS) estimator is based on the same assumptions about the disturbance terms as the OLS estimator. Accordingly, $\widehat{\Omega}$ in Equation 10 is equal to $I_{G \cdot T}$ and thus, cancels out. Like for the OLS estimator, the whole system can be treated either as one single equation with $\widehat{\Omega}$ in Equation 12 equal to $\hat{\sigma}^2 I_{G \cdot T}$, or the disturbance terms of the individual equations are allowed to have different variances with $\widehat{\Omega}$ in Equation 12 equal to $\widehat{\Sigma} \otimes I_T$, where $\hat{\sigma}_{ij} = 0 \; \forall \; i \neq j$ and $\hat{\sigma}_{ii} = \hat{\sigma}_i^2$.

*Weighted two-stage least squares (W2SLS)*

The weighted two-stage least squares (W2SLS) estimator allows for different variances of the disturbance terms in the different equations. Hence, $\widehat{\Omega}$ in Equations 10 and 12 is $\widehat{\Sigma} \otimes I_T$, where $\hat{\sigma}_{ij} = 0 \; \forall \; i \neq j$ and $\hat{\sigma}_{ii} = \hat{\sigma}_i^2$. If the estimated coefficients are not constrained by cross-equation restrictions, they are equal to 2SLS estimates.

*Three-stage least squares (3SLS)*

If the disturbances are contemporaneously correlated, a feasible generalized least squares (FGLS) version of the two-stage least squares estimation leads to consistent and asymptotically more efficient estimates. This estimation procedure is generally called "three-stage least squares" (3SLS, Zellner and Theil 1962). The standard 3SLS estimator and its covariance matrix are obtained by Equations 10 and 12 with $\widehat{\Omega}$ equal to $\widehat{\Sigma} \otimes I_T$, where $\widehat{\Sigma}$ is the estimated covariance matrix of the disturbance terms.

While an unbiased 2SLS or W2SLS estimation requires only that the instrumental variables and the disturbance terms of each single equation are uncorrelated ($\mathsf{E}\left[u_i^\top Z_i\right]) = 0 \; \forall \; i$), Schmidt (1990) points out that the 3SLS estimator is only consistent if all disturbance terms and all instrumental variables are uncorrelated ($\mathsf{E}\left[u_i^\top Z_j\right]) = 0 \; \forall \; i, j$). Since there might be occasions where this cannot be avoided, Schmidt (1990) analyses other approaches to obtain 3SLS estimators.

One of these approaches based on instrumental variable estimation (3SLS-IV) is

$$\hat{\beta}_{\text{3SLS-IV}} = \left( \widehat{X}^\top \widehat{\Omega}^{-1} X \right)^{-1} \widehat{X}^\top \widehat{\Omega}^{-1} y. \tag{13}$$

An estimator of the covariance matrix of the estimated 3SLS-IV coefficients is

$$\widehat{\mathsf{COV}}\left[\hat{\beta}_{\text{3SLS-IV}}\right] = \left( \widehat{X}^\top \widehat{\Omega}^{-1} X \right)^{-1}. \tag{14}$$

Another approach based on the generalized method of moments (GMM) estimator (3SLS-GMM) is

$$\hat{\beta}_{\text{3SLS-GMM}} = \left( X^\top Z \left( Z^\top \widehat{\Omega} Z \right)^{-1} Z^\top X \right)^{-1} X^\top Z \left( Z^\top \widehat{\Omega} Z \right)^{-1} Z^\top y \tag{15}$$

with

$$Z = \begin{bmatrix} Z_1 & 0 & \cdots & 0 \\ 0 & Z_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & Z_G \end{bmatrix}. \tag{16}$$

An estimator of the covariance matrix of the estimated 3SLS-GMM coefficients is

$$\widehat{\mathsf{COV}}\left[\hat{\beta}_{\text{3SLS-GMM}}\right] = \left( X^\top Z \left( Z^\top \widehat{\Omega} Z \right)^{-1} Z^\top X \right)^{-1}. \tag{17}$$

A fourth approach developed by Schmidt (1990) himself is

$$\hat{\beta}_{\text{3SLS-Schmidt}} = \left( \widehat{X}^\top \widehat{\Omega}^{-1} \widehat{X} \right)^{-1} \widehat{X}^\top \widehat{\Omega}^{-1} Z \left( Z^\top Z \right)^{-1} Z^\top y. \tag{18}$$

An estimator of the covariance matrix of these estimated coefficients is

$$\begin{aligned} \widehat{\mathsf{COV}}\left[\hat{\beta}_{\text{3SLS-Schmidt}}\right] = & \left( \widehat{X}^\top \widehat{\Omega}^{-1} \widehat{X} \right)^{-1} \widehat{X}^\top \widehat{\Omega}^{-1} Z \left( Z^\top Z \right)^{-1} Z^\top \widehat{\Omega} Z \\ & \left( Z^\top Z \right)^{-1} Z^\top \widehat{\Omega}^{-1} \widehat{X} \left( \widehat{X}^\top \widehat{\Omega}^{-1} \widehat{X} \right)^{-1}. \end{aligned} \tag{19}$$

The econometrics software EViews uses

$$\hat{\beta}_{\text{3SLS-EViews}} = \hat{\beta}_{\text{2SLS}} + \left( \widehat{X}^\top \widehat{\Omega}^{-1} \widehat{X} \right)^{-1} \widehat{X}^\top \widehat{\Omega}^{-1} \left( y - X \hat{\beta}_{\text{2SLS}} \right), \tag{20}$$

where $\hat{\beta}_{\text{2SLS}}$ is the two-stage least squares estimator as defined above. EViews uses the standard 3SLS formula (Equation 12) to calculate an estimator of the covariance matrix of the estimated coefficients.

If the same instrumental variables are used in all equations ($Z_1 = Z_2 = \ldots = Z_G$), all the above mentioned approaches lead to identical estimates. However, if this is not the case, the results depend on the method used (Schmidt 1990). The only reason to use different instruments for different equations is a correlation of the instruments of one equation with the disturbance terms of another equation. Otherwise, one could simply use all instruments in every equation (Schmidt 1990). In this case, only the 3SLS-GMM (Equation 15) and the 3SLS estimator developed by Schmidt (1990) (Equation 18) are consistent.

### 2.3. Estimation under linear restrictions on the coefficients

In many empirical applications, it is desirable to estimate the coefficients under linear restrictions. For instance, in econometric demand and production analysis, it is common to estimate the coefficients under homogeneity and symmetry restrictions that are derived from the underlying theoretical model.

There are two different methods to estimate the coefficients under linear restrictions. First, a matrix $M$ can be specified that

$$\beta = M \cdot \beta^{\mathrm{M}}, \tag{21}$$

where $\beta^{\mathrm{M}}$ is a vector of restricted (linear independent) coefficients, and $M$ is a matrix with the number of rows equal to the number of unrestricted coefficients ($\beta$) and the number of columns equal to the number of restricted coefficients ($\beta^{\mathrm{M}}$). $M$ can be used to map each unrestricted coefficient to one or more restricted coefficients.

The second method to estimate the coefficients under linear restrictions constrains the coefficients by

$$R\beta^{\mathrm{R}} = q, \tag{22}$$

where $\beta^{\mathrm{R}}$ is the vector of the restricted coefficients, and $R$ and $q$ are a matrix and vector, respectively, that specify the restrictions (see Greene 2003, p. 100). Each linear independent restriction is represented by one row of $R$ and the corresponding element of $q$.

The first method is less flexible than the second[4], but is preferable if the coefficients are estimated under many equality constraints across different equations of the system. Of course, these restrictions can be also specified using the latter method. However, while the latter method increases the dimension of the matrices to be inverted during estimation, the first reduces it. Thus, in some cases the latter way leads to estimation problems (e.g., (near) singularity of the matrices to be inverted), while the first does not.

These two methods can be combined. In this case, the restrictions specified using the latter method are imposed on the linear independent coefficients that are restricted by the first method, so that

$$R\beta^{\mathrm{MR}} = q, \tag{23}$$

where $\beta^{\mathrm{MR}}$ is the vector of the restricted $\beta^{\mathrm{M}}$ coefficients.

[4]While restrictions like $\beta_1 = 2\beta_2$ can be specified by both methods, restrictions like $\beta_1 + \beta_2 = 4$ can be specified only by the second method.

*Calculation of restricted estimators*

If the first method (Equation 21) is chosen to estimate the coefficients under these restrictions, the matrix of regressors $X$ is (post-)multiplied by the $M$ matrix, so that

$$X^{\mathrm{M}} = X \cdot M. \tag{24}$$

Then, $X^{\mathrm{M}}$ is substituted for $X$ and a standard estimation as described in the previous section is done (Equations 7–20). This results in the linear independent coefficient estimates $\hat{\beta}^{\mathrm{M}}$ and their covariance matrix. The original coefficients can be obtained by Equation 21 and the estimated covariance matrix of the original coefficients can be obtained by

$$\widehat{\mathsf{COV}}\left[\hat{\beta}\right] = M \cdot \widehat{\mathsf{COV}}\left[\hat{\beta}^{\mathrm{M}}\right] \cdot M^{\top}. \tag{25}$$

The implementation of the second method to estimate the coefficients under linear restrictions (Equation 22) is described for each estimation method in the following sections.

*Restricted OLS, WLS, and SUR estimation*

The OLS, WLS, and SUR estimators restricted by $R\beta^{\mathrm{R}} = q$ can be obtained by

$$\begin{bmatrix} \hat{\beta}^{\mathrm{R}} \\ \hat{\lambda} \end{bmatrix} = \begin{bmatrix} X^{\top}\widehat{\Omega}^{-1}X & R^{\top} \\ R & 0 \end{bmatrix}^{-1} \cdot \begin{bmatrix} X^{\top}\widehat{\Omega}^{-1}y \\ q \end{bmatrix}, \tag{26}$$

where $\lambda$ is a vector of the Lagrangean multipliers of the restrictions and $\widehat{\Omega}$ is defined as in Section 2.1. An estimator of the covariance matrix of the estimated coefficients is

$$\widehat{\mathsf{COV}}\begin{bmatrix} \hat{\beta}^{\mathrm{R}} \\ \hat{\lambda} \end{bmatrix} = \begin{bmatrix} X^{\top}\widehat{\Omega}^{-1}X & R^{\top} \\ R & 0 \end{bmatrix}^{-1}. \tag{27}$$

*Restricted 2SLS, W2SLS, and 3SLS estimation*

The 2SLS, W2SLS, and standard 3SLS estimators restricted by $R\beta^{\mathrm{R}} = q$ can be obtained by

$$\begin{bmatrix} \hat{\beta}^{\mathrm{R}} \\ \hat{\lambda} \end{bmatrix} = \begin{bmatrix} \widehat{X}^{\top}\widehat{\Omega}^{-1}\widehat{X} & R^{\top} \\ R & 0 \end{bmatrix}^{-1} \cdot \begin{bmatrix} \widehat{X}^{\top}\widehat{\Omega}^{-1}y \\ q \end{bmatrix}, \tag{28}$$

where $\widehat{\Omega}$ is defined as in Section 2.2. An estimator of the covariance matrix of the estimated coefficients is

$$\widehat{\mathsf{COV}}\begin{bmatrix} \hat{\beta}^{\mathrm{R}} \\ \hat{\lambda} \end{bmatrix} = \begin{bmatrix} \widehat{X}^{\top}\widehat{\Omega}^{-1}\widehat{X} & R^{\top} \\ R & 0 \end{bmatrix}^{-1}. \tag{29}$$

The 3SLS-IV estimator restricted by $R\beta^{\mathrm{R}} = q$ can be obtained by

$$\begin{bmatrix} \hat{\beta}^{\mathrm{R}}_{\text{3SLS-IV}} \\ \hat{\lambda} \end{bmatrix} = \begin{bmatrix} \widehat{X}^{\top}\widehat{\Omega}^{-1}X & R^{\top} \\ R & 0 \end{bmatrix}^{-1} \cdot \begin{bmatrix} \widehat{X}^{\top}\widehat{\Omega}^{-1}y \\ q \end{bmatrix}, \tag{30}$$

where

$$\widehat{\mathsf{COV}}\begin{bmatrix} \hat{\beta}^{\mathrm{R}}_{\text{3SLS-IV}} \\ \hat{\lambda} \end{bmatrix} = \begin{bmatrix} \widehat{X}^{\top}\widehat{\Omega}^{-1}X & R^{\top} \\ R & 0 \end{bmatrix}^{-1}. \tag{31}$$

The restricted 3SLS-GMM estimator can be obtained by

$$\begin{bmatrix} \hat{\beta}^{\mathrm{R}}_{\text{3SLS-GMM}} \\ \hat{\lambda} \end{bmatrix} = \begin{bmatrix} X^\top Z \left( Z^\top \widehat{\Omega} Z \right)^{-1} Z^\top X & R^\top \\ R & 0 \end{bmatrix}^{-1} \cdot \begin{bmatrix} X^\top Z \left( Z \widehat{\Omega} Z \right)^{-1} Z^\top y \\ q \end{bmatrix}, \tag{32}$$

where

$$\widehat{\mathsf{COV}} \begin{bmatrix} \hat{\beta}^{\mathrm{R}}_{\text{3SLS-GMM}} \\ \hat{\lambda} \end{bmatrix} = \begin{bmatrix} X^\top Z \left( Z^\top \widehat{\Omega} Z \right)^{-1} Z^\top X & R^\top \\ R & 0 \end{bmatrix}^{-1}. \tag{33}$$

The restricted 3SLS estimator based on the suggestion of Schmidt (1990) is

$$\begin{bmatrix} \hat{\beta}^{\mathrm{R}}_{\text{3SLS-Schmidt}} \\ \hat{\lambda} \end{bmatrix} = \begin{bmatrix} \widehat{X}^\top \widehat{\Omega}^{-1} \widehat{X} & R^\top \\ R & 0 \end{bmatrix}^{-1} \cdot \begin{bmatrix} \widehat{X}^\top \widehat{\Omega}^{-1} Z \left( Z^\top Z \right)^{-1} Z^\top y \\ q \end{bmatrix}, \tag{34}$$

where

$$\begin{aligned} \widehat{\mathsf{COV}} \begin{bmatrix} \hat{\beta}^{\mathrm{R}}_{\text{3SLS-Schmidt}} \\ \hat{\lambda} \end{bmatrix} = & \begin{bmatrix} \widehat{X}^\top \widehat{\Omega}^{-1} \widehat{X} & R^\top \\ R & 0 \end{bmatrix}^{-1} \\ & \cdot \begin{bmatrix} \widehat{X}^\top \widehat{\Omega}^{-1} Z \left( Z^\top Z \right)^{-1} Z^\top \widehat{\Omega} Z \left( Z^\top Z \right)^{-1} Z^\top \widehat{\Omega}^{-1} \widehat{X} & 0^\top \\ 0 & 0 \end{bmatrix} \\ & \cdot \begin{bmatrix} \widehat{X}^\top \widehat{\Omega}^{-1} \widehat{X} & R^\top \\ R & 0 \end{bmatrix}^{-1}. \end{aligned} \tag{35}$$

The econometrics software EViews calculates the restricted 3SLS estimator by

$$\begin{bmatrix} \hat{\beta}^{\mathrm{R}}_{\text{3SLS-EViews}} \\ \hat{\lambda} \end{bmatrix} = \begin{bmatrix} \widehat{X}^\top \widehat{\Omega}^{-1} \widehat{X} & R^\top \\ R & 0 \end{bmatrix}^{-1} \cdot \begin{bmatrix} \widehat{X}^\top \widehat{\Omega}^{-1} \left( y - X \hat{\beta}^{\mathrm{R}}_{\text{2SLS}} \right) \\ q \end{bmatrix}, \tag{36}$$

where $\hat{\beta}^{\mathrm{R}}_{\text{2SLS}}$ is the restricted 2SLS estimator calculated by Equation 28. EViews uses the standard formula of the restricted 3SLS estimator (Equation 29) to calculate an estimator for the covariance matrix of the estimated coefficients.

If the same instrumental variables are used in all equations ($Z_1 = Z_2 = \ldots = Z_G$), all the above mentioned approaches lead to identical coefficient estimates and identical covariance matrices of the estimated coefficients.

## 2.4. Residual covariance matrix

Since the (true) disturbances ($u_i$) of the estimated equations are generally not known, their covariance matrix cannot be determined. Therefore, this covariance matrix is generally calculated from estimated residuals ($\hat{u}_i$) that are obtained from a first-step OLS or 2SLS estimation. Then, in a second step, the estimated residual covariance matrix can be employed for a WLS, SUR, W2SLS, or 3SLS estimation. In many cases, the residual covariance matrix is calculated by

$$\hat{\sigma}_{ij} = \frac{\hat{u}_i^\top \hat{u}_j}{T}, \tag{37}$$

where $T$ is the number of observations in each equation. However, in finite samples this estimator is biased, because it is not corrected for degrees of freedom. The usual single-equation procedure to correct for degrees of freedom cannot always be applied, because the number of regressors in each equation might differ. Two alternative approaches to calculate the residual covariance matrix are

$$\hat{\sigma}_{ij} = \frac{\hat{u}_i^\top \hat{u}_j}{\sqrt{(T - K_i) \cdot (T - K_j)}} \tag{38}$$

and

$$\hat{\sigma}_{ij} = \frac{\hat{u}_i^\top \hat{u}_j}{T - \max\left(K_i, K_j\right)} \, , \tag{39}$$

where $K_i$ and $K_j$ are the number of regressors in equation $i$ and $j$, respectively. However, these formulas yield unbiased estimators only if $K_i = K_j$ (Judge *et al.* 1985, p. 469).

A further approach to obtain a residual covariance matrix is

$$\hat{\sigma}_{ij} = \frac{\hat{u}_i^\top \hat{u}_j}{T - K_i - K_j + \operatorname{tr}\left[X_i \left(X_i^\top X_i\right)^{-1} X_i^\top X_j \left(X_j^\top X_j\right)^{-1} X_j^\top\right]} \tag{40}$$

$$= \frac{\hat{u}_i^\top \hat{u}_j}{T - K_i - K_j + \operatorname{tr}\left[\left(X_i^\top X_i\right)^{-1} X_i^\top X_j \left(X_j^\top X_j\right)^{-1} X_j^\top X_i\right]} \tag{41}$$

(Zellner and Huang 1962, p. 309). This yields an unbiased estimator for all elements of $\Sigma$, but even if $\widehat{\Sigma}$ is an unbiased estimator of $\Sigma$, its inverse $\widehat{\Sigma}^{-1}$ is not an unbiased estimator of $\Sigma^{-1}$ (Theil 1971, p. 322). Furthermore, the covariance matrix calculated by Equation 40 is not necessarily positive semidefinite (Theil 1971, p. 322). Hence, "it is doubtful whether [this formula] is really superior to [Equation 37]" (Theil 1971, p. 322).

The WLS, SUR, W2SLS and 3SLS coefficient estimates are consistent if the residual covariance matrix is calculated using the residuals from a first-step OLS or 2SLS estimation. There exists also an alternative slightly different approach that consists of three steps.[5] In a first step, an OLS or 2SLS estimation is applied to obtain residuals to calculate a (first-step) residual covariance matrix. In a second step, the first-step residual covariance matrix is used to estimate the model by WLS or W2SLS and new residuals are obtained to calculate a (second-step) residual covariance matrix. Finally, in the third step, the second-step residual covariance matrix is used to estimate the model by SUR or 3SLS. If the estimated coefficients are not constrained by cross-equation restrictions, OLS and WLS estimates as well as 2SLS and W2SLS estimates are identical. Hence, in this case both approaches generate the same results.

It is also possible to iterate WLS, SUR, W2SLS and 3SLS estimations. At each iteration the residual covariance matrix is calculated from the residuals of the previous iteration. If Equation 37 is applied to calculate the residual covariance matrix, an iterated SUR estimation converges to maximum likelihood (Greene 2003, p. 345).

In some uncommon cases, for instance in pooled estimations, where the coefficients are restricted to be equal in all equations, the means of the residuals of each equation are not equal

[5] For instance, this approach is applied by the command `TSCS` of the software LIMDEP that carries out SUR estimations in which all coefficient vectors are constrained to be equal (Greene 2006b).

to zero ($\overline{\hat{u}}_i \neq 0$). Therefore, it might be argued that the residual covariance matrix should be calculated by subtracting the means from the residuals and substituting $\hat{u}_i - \overline{\hat{u}}_i$ for $\hat{u}_i$ in Equations 37–40.

If the coefficients are estimated under any restrictions, the residual covariance matrix for a WLS, SUR, W2SLS, or 3SLS estimation can be obtained either from a restricted or from an unrestricted first-step estimation.

## 2.5. Degrees of freedom

To our knowledge the question about how to determine the degrees of freedom for single-coefficient $t$ tests is not comprehensively discussed in the literature. While sometimes the degrees of freedom of the entire system (total number of observations in all equations minus total number of estimated coefficients) are applied, in other cases the degrees of freedom of each single equation (number of observations in the equations minus number of estimated coefficients in the equation) are used. Asymptotically, this distinction does not make a difference. However, in many empirical applications, the number of observations of each equation is rather small, and therefore, it matters.

If a system of equations is estimated by an unrestricted OLS and the covariance matrix of the coefficients is calculated with $\widehat{\Omega}$ in Equation 8 equal to $\widehat{\Sigma} \otimes I_T$, the estimated coefficients and their standard errors are identical to an equation-wise OLS estimation. In this case, it is reasonable to use the degrees of freedom of each single equation, because this yields the same $P$ values as the equation-wise OLS estimation.

In contrast, if a system of equations is estimated with many cross-equation restrictions and the covariance matrix of an OLS estimation is calculated with $\widehat{\Omega}$ in Equation 8 equal to $\hat{\sigma}^2 I_{G \cdot T}$, the system estimation is similar to a single equation estimation. Therefore, in this case, it seems to be reasonable to use the degrees of freedom of the entire system.

## 2.6. Goodness of fit

The goodness of fit of each single equation can be measured by the traditional $R^2$ values

$$R_i^2 = 1 - \frac{\hat{u}_i^\top \hat{u}_i}{(y_i - \overline{y_i})^\top (y_i - \overline{y_i})} \,, \tag{42}$$

where $R_i^2$ is the $R^2$ value of the $i$th equation and $\overline{y_i}$ is the mean value of $y_i$.

The goodness of fit of the whole system can be measured by the McElroy's $R^2$ value

$$R_*^2 = 1 - \frac{\hat{u}^\top \widehat{\Omega}^{-1} \hat{u}}{y^\top \left( \widehat{\Sigma}^{-1} \otimes \left( I_T - \frac{\iota \iota^\top}{T} \right) \right) y}, \tag{43}$$

where $\iota$ is a column vector of $T$ ones (McElroy 1977).

## 2.7. Testing linear restrictions

Linear restrictions can be tested by an $F$ test, two Wald tests and a likelihood ratio (LR) test.

The $F$ statistic for systems of equations is

$$F = \frac{(R\hat{\beta} - q)^\top (R(X^\top(\Sigma \otimes I)^{-1}X)^{-1}R^\top)^{-1}(R\hat{\beta} - q)/j}{\hat{u}^\top(\Sigma \otimes I)^{-1}\hat{u}/(G \cdot T - K)}\,, \tag{44}$$

where $j$ is the number of restrictions, $K$ is the total number of estimated coefficients, and all other variables are as defined before (Theil 1971, p. 314). Under the null hypothesis, $F$ is $F$ distributed with $j$ and $G \cdot T - K$ degrees of freedom.

However, $F$ in Equation 44 cannot be computed, because $\Sigma$ is generally unknown. As a solution, Theil (1971, p. 314) proposes to replace the unknown $\Sigma$ in Equation 44 by the estimated covariance matrix $\widehat{\Sigma}$.

$$\hat{F} = \frac{(R\hat{\beta} - q)^\top (R(X^\top(\widehat{\Sigma} \otimes I)^{-1}X)^{-1}R^\top)^{-1}(R\hat{\beta} - q)/j}{\hat{u}^\top(\widehat{\Sigma} \otimes I)^{-1}\hat{u}/(G \cdot T - K)} \tag{45}$$

Asymptotically, $\hat{F}$ has the same distribution as $F$ in Equation 44, because the numerator of Equation 45 converges in probability to the numerator of Equation 44 and the denominator of Equation 45 converges in probability to the denominator of Equation 44 (Theil 1971, p. 402).

Furthermore, the denominators of both Equations 44 and 45 converge in probability to 1. Taking this into account and applying Equation 8, we obtain the usual $F$ statistic of the Wald test.

$$\hat{\hat{F}} = \frac{(R\hat{\beta} - q)^\top (R\,\widehat{\mathsf{COV}}\left[\hat{\beta}\right]R^\top)^{-1}(R\hat{\beta} - q)}{j} \tag{46}$$

Under the null hypotheses, also $\hat{\hat{F}}$ is asymptotically $F$ distributed with $j$ and $G \cdot T - K$ degrees of freedom.

Multiplying Equation 46 with $j$, we obtain the usual $\chi^2$ statistic for the Wald test

$$W = (R\hat{\beta} - q)^\top (R\,\widehat{\mathsf{COV}}[\hat{\beta}]\,R^\top)^{-1}(R\hat{\beta} - q). \tag{47}$$

Asymptotically, $W$ has a $\chi^2$ distribution with $j$ degrees of freedom under the null hypothesis (Greene 2003, p. 347).

The likelihood-ratio (LR) statistic for systems of equations is

$$LR = T \cdot \left(\log\left|\widehat{\Sigma}_r\right| - \log\left|\widehat{\Sigma}_u\right|\right), \tag{48}$$

where $T$ is the number of observations per equation, and $\widehat{\Sigma}_r$ and $\widehat{\Sigma}_u$ are the residual covariance matrices calculated by Equation 37 of the restricted and unrestricted estimation, respectively. Asymptotically, $LR$ has a $\chi^2$ distribution with $j$ degrees of freedom under the null hypothesis (Greene 2003, p. 349).

## 2.8. Hausman test

Hausman (1978) developed a test for misspecification. The null hypothesis of the test is that the instrumental variables of each equation are uncorrelated with the disturbance terms of all other equations ($\mathsf{E}\left[u_i^\top Z_j\right] = 0 \,\forall\, i \neq j$). Under this null hypothesis, both the 2SLS and the 3SLS estimator are consistent, but the 3SLS estimator is (asymptotically) more efficient.

Under the alternative hypothesis, the 2SLS estimator is consistent but the 3SLS estimator is inconsistent, i.e., the instrumental variables of each equation are uncorrelated with the disturbances of the same equation ($\mathsf{E}\left[u_i^\top Z_i\right] = 0 \,\forall\, i$), but the instrumental variables of at least one equation are correlated with the disturbances of another equation ($\mathsf{E}\left[u_i^\top Z_j\right] \neq 0 \,\exists\, i \neq j$). The Hausman test statistic is

$$m = \left(\hat{\beta}_{\text{2SLS}} - \hat{\beta}_{\text{3SLS}}\right)^\top \left(\widehat{\mathsf{COV}}\left[\hat{\beta}_{\text{2SLS}}\right] - \widehat{\mathsf{COV}}\left[\hat{\beta}_{\text{3SLS}}\right]\right) \left(\hat{\beta}_{\text{2SLS}} - \hat{\beta}_{\text{3SLS}}\right), \tag{49}$$

where $\hat{\beta}_{\text{2SLS}}$ and $\widehat{\mathsf{COV}}\left[\hat{\beta}_{\text{2SLS}}\right]$ are the estimated coefficient and covariance matrix from a 2SLS estimation, and $\hat{\beta}_{\text{3SLS}}$ and $\widehat{\mathsf{COV}}\left[\hat{\beta}_{\text{3SLS}}\right]$ are the estimated coefficients and covariance matrix from a 3SLS estimation. Under the null hypothesis, this test statistic has a $\chi^2$ distribution with degrees of freedom equal to the number of estimated coefficients.

# 3. Source code

The source code of the **systemfit** package is publicly available for download from the Comprehensive R Archive Network (CRAN, http://CRAN.R-project.org/). The basic functionality of this package is provided by the function `systemfit`. Moreover, this package provides tools for statistical tests, functions (methods) to show, extract or calculate results, some convenience functions, and internal helper functions.

## 3.1. The basic function `systemfit`

The `systemfit` function estimates systems of linear equations by different estimation methods. Where possible, the user interface and the returned object of this function follow the function `lm` — the basic tool for linear regressions in R — to make the usage of `systemfit` as easy as possible for R users.

The econometric estimation is done by applying the formulas in Sections 2.1 and 2.2 or — if the coefficients are estimate under linear restrictions — by the formulas in Section 2.3. If the restrictions on the coefficients are specified symbolically, function `makeHypothesis` of the **car** package (Fox 2006, 2002) is used to create the restriction matrix.

The `systemfit` function returns a list of class `systemfit` that contains the results that belong to the entire system of equations. One special element of this list is called `eq`, which is a list that contains one object for each estimated equation. These objects are lists of class `systemfit.equation` and contain the results that belong only to the regarding equation. A complete description is available in the documentation of this function that is included in the package. A comparison of the elements returned by `lm` and by `systemfit` is available in appendix A.

## 3.2. Statistical tests

The `linearHypothesis` and `lrtest` methods for `systemfit` objects as well as the function `hausman.systemfit` apply the statistical tests described in Sections 2.7 and 2.8.

The `linearHypothesis` method for systemfit objects can be used to test linear restrictions on the estimated coefficients by Theil's $F$ test or by usual Wald tests. Internally, Theil's

$F$ statistic is computed by the hidden function `.ftest.systemfit` and the Wald tests are computed by the default `linearHypothesis` method of the **car** package (Fox 2006, 2002). The `lrtest` method for `systemfit` objects is a wrapper function to the default `lrtest` method of the **lmtest** package (Zeileis and Hothorn 2002), which computes the likelihood-ratio (LR) test statistic. All these functions return an object of class `anova` that contains — amongst others — the empirical test statistic, the degrees of freedom, and the corresponding $P$ value.

The function `hausman.systemfit` tests the consistency of the 3SLS estimator. It returns an object of class `htest`, which contains — amongst others — the empirical test statistic, the degrees of freedom, and the $P$ value.

### 3.3. Other methods and functions

The **systemfit** package provides several methods for objects both of classes `systemfit` and `systemfit.equation`: `print` methods print the estimation results, `summary` methods calculate summary results, `confint` methods compute confidence intervals for the coefficients, `predict` methods calculate predicted values, `coef` methods extract the estimated coefficients, `vcov` methods extract their covariance matrix, `fitted` methods extract the fitted values, `residuals` methods extract the residuals, `formula` methods extract the formula(s), `terms` methods extract the model terms, `model.frame` methods extract the model frame, and `model.matrix` methods extract the model matrix. Some methods can be applied to objects of class `systemfit` only: a `correlation` method calculates the correlations between the predictions of two equations, an `se.ratio` method computes the ratios of the standard errors of the predictions between two models, and a `logLik` method extracts the log likelihood value. The package provides `print` methods to print objects of classes `summary.systemfit`, `summary.systemfit.equation`, and `confint.systemfit` that are returned by the above mentioned `summary` and `confint` methods. There exist also two `coef` methods to extract the estimated coefficients, their standard errors, $t$ values, and $P$ values from objects of classes `summary.systemfit` and `summary.systemfit.equation`.[6]

The convenience function `createSystemfitModel` creates a model for `systemfit` by random numbers; `systemfit.control` sets suitable default values for the technical control parameters for `systemfit`.

Finally, the package includes some internal (hidden) helper functions: `.prepareData.systemfit`, `.stackMatList`, and `.prepareWmatrix` for preparing the data matrices; `.calcXtOmegaInv` and `.calcGLS` for calculating the GLS estimator; `.calcResidCov` and `.calcSigma2` for calculating the (co)variances of the residuals; and `.ftest.systemfit` for calculating Theil's $F$ statistic. If `systemfit` is applied to a (classical) "seemingly unrelated regression" analysis with panel data, it calls the hidden internal function `.systemfitPanel`, which reshapes the data, creates the formulas to be estimated, and — if requested — specifies restrictions to ensure that the coefficients of all individuals are equal.

### 3.4. Efficiency of computations

We have followed Bates (2004) to make the code of **systemfit** faster and more stable. First, if a formula contains an inverse of a matrix that is post-multiplied by a vector or ma-

[6] There does not exist a special method to extract the degrees of freedom of the residuals from `systemfit` objects, because the default method of `df.residual` works for these objects.

trix, we use `solve(A,b)` instead of `solve(A) %*% b`. Second, we calculate crossproducts by `crossprod(X)` or `crossprod(X,y)` instead of `t(X) %*% X` or `t(X) %*% y`, respectively.

The matrix $\Omega^{-1}$ that is used to compute the estimated coefficients and their covariance matrix is of size $(G \cdot T) \times (G \cdot T)$ (see Sections 2.1, 2.2, and 2.3). In case of large data sets, $\Omega^{-1}$ becomes computationally infeasible. Therefore, we use the following transformation and compute $X^\top \Omega^{-1}$ by dividing the $X$ matrix into submatrices, doing some calculations with these submatrices, adding up some of these submatrices, and finally putting the submatrices together, so that

$$X^\top \Omega^{-1} = \sum_{i=1} \begin{bmatrix} \sigma^{1i} X^1 \\ \sigma^{2i} X^2 \\ \vdots \\ \sigma^{Gi} X^G \end{bmatrix}^\top , \tag{50}$$

where $\sigma^{ij}$ are the elements of the matrix $\Sigma^{-1}$, and $X^i$ is a submatrix of $X$ that contains the rows that belong to the $i$'s equation. This computation is done inside the internal (hidden) function `.calcXtOmegaInv`.

Since version 1.0, the `systemfit` function by default uses the **Matrix** package (Bates and Maechler 2007) for all computations where matrices are involved. The **Matrix** package provides classes for different types of matrices. For instance, we choose class `dgeMatrix` ("real matrices in general storage mode"), for matrices $X_i$ in Equation 2, class `dgCMatrix` ("general, numeric, sparse matrices in the (sorted) compressed sparse column format") for matrix $X$ in Equation 3, and class `dspMatrix` ("symmetric real matrices in packed storage (one triangle only)") for the residual covariance matrix $\widehat{\Sigma}$. If the **Matrix** package is used, the possibly huge matrix $\Omega^{-1}$ is no longer a problem, because it is a sparse matrix that can be stored in a compressed format (class `dgCMatrix`). Hence, we no longer need the algorithm in Equation 50. We have tested different ways to calculate a GLS estimator like in Equation 7 and we found that the following code is the fastest:

```
sigmaInv <- solve( residCov )
xtOmegaInv <- crossprod( xMat, kronecker( sigmaInv, Diagonal( nObs ) ) )
coef <- solve( xtOmegaInv %*% xMat, xtOmegaInv %*%  yVec )
```

In this code snippet, `residCov` is the residual covariance matrix $\widehat{\Sigma}$, `nObs` is the number of observations in each equation $T$, `xMat` is the matrix $X$ and `yVec` is the vector $y$ in Equation 7.

By default, the `systemfit` function uses the **Matrix** package to perform GLS estimations, because using this package considerably decreases the computation time for many common models. However, the estimation of small models with small data sets gets slower by using the **Matrix** package (see appendix B). While this increase in computation time is often imperceptible to human beings, it might matter in some cases such as iterated estimations or Monte Carlo studies. Therefore, the user can opt for not using the **Matrix** package, but Equation 50 with standard R matrices.

### 3.5. Overlap with other functions and packages in R

Single-equation models can be fitted in R by OLS with function `lm` (package **stats**) and by 2SLS with function `tsls` (package **sem**, Fox 2011). This is also possible with the `systemfit`

function, but `systemfit` is specialized in estimating systems of equation, i.e., more than one equation. Its capability to estimate single-equation models is just a side-effect.

Function `sem` (package **sem**, Fox 2011) can be used to estimate structural equation models in R by limited information maximum likelihood (LIML) and full information maximum likelihood (FIML) assuming normal or multinormal errors, respectively. A special feature of this function is the estimation of models with unobserved ("latent") variables, which is not possible with `systemfit`. While `sem` cannot be used to consistently estimate systems of simultaneous equations with some endogenous regressors, it can be used to estimate systems of equations, where all regressors are exogenous. However, the latter is rather cumbersome (see appendix C). Hence, `systemfit` is the only function in R that can be used to estimate systems of simultaneous equations and it is the most convenient function to estimate systems of equations with purely exogenous regressors.

# 4. Using systemfit

In this section we demonstrate how to use the **systemfit** package. First, we show the standard usage of `systemfit` by a simple example. Second, several options that can be specified by the user are presented. Then, the usage of `systemfit` for a (classical) "seemingly unrelated regression" analysis with panel data is described. Finally, we demonstrate how to apply some statistical tests.

## 4.1. Standard usage of `systemfit`

As described in the previous section, systems of equations can be econometrically estimated with the function `systemfit`. The only mandatory argument is `formula`. Typically, it is a list of formulas to be estimated, but it may also be a single formula for estimating a single-equation model. Each formula is a standard regression formula in R (see documentation of `formula`).

The following demonstration uses an example taken from Kmenta (1986, p. 685). We want to estimate a small model of the US food market:

$$\texttt{consump} = \beta_1 + \beta_2 \cdot \texttt{price} + \beta_3 \cdot \texttt{income} \tag{51}$$

$$\texttt{consump} = \beta_4 + \beta_5 \cdot \texttt{price} + \beta_6 \cdot \texttt{farmPrice} + \beta_7 \cdot \texttt{trend} \tag{52}$$

The first equation represents the demand side of the food market. Variable `consump` (food consumption per capita) is the dependent variable. The regressors are `price` (ratio of food prices to general consumer prices) and `income` (disposable income) as well as a constant. The second equation specifies the supply side of the food market. Variable `consump` is the dependent variable of this equation as well. The regressors are again `price` (ratio of food prices to general consumer prices) and a constant as well as `farmPrice` (ratio of preceding year's prices received by farmers to general consumer prices) and `trend` (a time trend in years). These equations can be estimated by OLS in R by

```
library( "systemfit" )
data( "Kmenta" )
attach( Kmenta )
```

```
eqDemand <- consump ~ price + income
eqSupply <- consump ~ price + farmPrice + trend
eqSystem <- list( demand = eqDemand, supply = eqSupply )
fitols <- systemfit( eqSystem )
print( fitols )

##
## systemfit results
## method: OLS
##
## Coefficients:
## demand_(Intercept)       demand_price      demand_income supply_(Intercept)
##          99.895423          -0.316299           0.334636          58.275431
##       supply_price   supply_farmPrice       supply_trend
##           0.160367           0.248133           0.248302
```

The first line loads the **systemfit** package. The second line loads example data that are included with the package. They are attached to the R search path in line three. In the fourth and fifth line, the demand and supply equations are specified, respectively.[7] In the sixth line, these equations are concatenated into a list and are labeled `demand` and `supply`, respectively.[8] Finally, in the last two lines, the regression is performed and the estimation results are printed.

## 4.2. User options of `systemfit`

The user can modify the default estimation method by providing additional optional arguments, e.g., to specify instrumental variables or restrictions on the coefficients. All optional arguments are described in the following:

### *Estimation method*

The optional argument `method` is a string that determines the estimation method. It must be either `"OLS"`, `"WLS"`, `"SUR"`, `"2SLS"`, `"W2SLS"`, or `"3SLS"`. These methods correspond to the estimation methods described in Sections 2.1, 2.2, and 2.3. The following command estimates the model described above as "seemingly unrelated regression".

```
fitsur <- systemfit( eqSystem, method = "SUR" )
```

### *Instrumental variables*

The instruments for a 2SLS, W2SLS or 3SLS estimation can be specified by the argument `inst`. If the same instruments should be used for all equations, `inst` must be a one-sided formula.[9] If different instruments should be used for each equation, `inst` must be a list that

[7] A regression constant is always implied if not explicitly omitted.
[8] If no labels are provided, the equations are numbered consecutively ( `eq1`, `eq2`, ... ).
[9] A one-sided formula is a standard formula in R without a dependent variable.

contains a one-sided formula for each equation. The following example uses instrumental variables to estimate the model described above by "three-stage least squares" (3SLS). While the first command specifies the same instruments for all equations, the second uses different instruments:

```
fit3sls  <- systemfit( eqSystem, method = "3SLS",
   inst = ~ income + farmPrice + trend )
fit3sls2 <- systemfit( eqSystem, method = "3SLS",
   inst = list( ~ farmPrice + trend, ~ income + farmPrice + trend ) )
```

### *Data*

Having all data in the global environment or attached to the search path is often inconvenient. Therefore, `systemfit` has the argument `data` to specify a data frame that contains the variables of the model. In the following example, we use this argument to specify that the data for the estimation should be taken from the data frame `Kmenta`. Hence, we no longer need to attach this data frame before calling `systemfit`:

```
fitsur <- systemfit( eqSystem, method = "SUR", data = Kmenta )
```

### *Restrictions on the coefficients*

As outlined in Section 2.3, restrictions on the coefficients can be specified in two ways. One way is to use the matrix $R$ and the vector $q$ (see Section 2.3). These restrictions can be specified symbolically by argument `restrict.matrix` as in the generic function `linearHypothesis` of the **car** package (Fox 2006, 2002). This argument must be a vector of character strings, where each element represents one linear restriction, and each element must be either a linear combination of coefficients, or a linear equation in the coefficients (see documentation of function `linearHypothesis` in the **car** package, Fox 2006, 2002). We illustrate this by estimating the model under the restriction $\beta_2+\beta_6 = 0$. Since the name of $\beta_2$ (coefficient of variable `price` in equation `demand`) is `demand_price` and the name of $\beta_6$ (coefficient of variable `farmPrice` in equation `supply`) is `supply_farmPrice`, this restriction can be specified by

```
restrict <- "demand_price + supply_farmPrice = 0"
fitsurRmat <- systemfit(eqSystem, method = "SUR",
   restrict.matrix = restrict)
```

Alternatively, the restrictions via matrix $R$ and vector $q$ can be specified numerically. The matrix $R$ can be specified with argument `restrict.matrix` and the vector $q$ with argument `restrict.rhs`.

```
Rmat <- matrix(0, nrow = 1, ncol = 7)
Rmat[1, 2] <- 1
Rmat[1, 6] <- 1
qvec <- c(0)
fitsurRmatNum <- systemfit(eqSystem, method = "SUR",
   restrict.matrix = Rmat, restrict.rhs = qvec)
```

The first line creates a $1 \times 7$ matrix of zeros, where 1 is the number of restrictions and 7 is the number of unrestricted coefficients. The following two lines specify this matrix in a way that the multiplication with the coefficient vector results in $\beta_2 + \beta_6$. The fourth line creates a vector with a single element that contains the right hand side of the restriction, i.e., zero. Finally the coefficients are estimated under the restriction $\beta_2 + \beta_6 = 0$.

The other way to specify restrictions on the coefficients is to modify the regressor matrix by post-multiplying it with a matrix, say $M$ (see Section 2.3). This kind of restriction can be specified by setting argument `restrict.regMat` equal to the matrix $M$. We convert the restriction specified above to $\beta_2 = -\beta_6$, and set $\beta_1 = \beta_1^{\mathrm{M}}$, ..., $\beta_5 = \beta_5^{\mathrm{M}}$, $\beta_6 = -\beta_2^{\mathrm{M}}$, and $\beta_7 = \beta_6^{\mathrm{M}}$. We can do this in R by

```
modRegMat <- matrix(0, nrow = 7, ncol = 6)
modRegMat[1:5, 1:5] <- diag(5)
modRegMat[6, 2] <- -1
modRegMat[7, 6] <- 1
fitsurRegMat <- systemfit(eqSystem, method = "SUR",
   restrict.regMat = modRegMat)
```

The first line creates a $7 \times 6$ matrix of zeros, where 7 is the number of unrestricted coefficients and 6 is the number of restricted coefficients. The following three lines specify the matrix $M$ (`modRegMat`) as described before. Finally the coefficients are estimated under the restriction $\beta_2^{\mathrm{M}} = \beta_2 = -\beta_6$.

Of course, the estimation results do not depend on the method that was used to specify this restriction:

```
all.equal( coef( fitsurRmat ), coef( fitsurRmatNum ) )

## [1] TRUE

all.equal( coef( fitsurRmat ), coef( fitsurRegMat ) )

## [1] TRUE
```

*Iteration control*

The estimation methods WLS, SUR, W2SLS and 3SLS need a covariance matrix of the residuals that can be calculated from a first-step OLS or 2SLS estimation (see Section 2.4). This procedure can be iterated and at each iteration the covariance matrix is calculated from the previous step estimation. This iteration is repeated until the maximum number of iterations is reached or the coefficient estimates have converged. The maximum number of iterations is specified by argument `maxiter`. Its default value is one, which means no iteration. The convergence criterion is

$$\sqrt{\frac{\sum_i (\beta_{i,g} - \beta_{i,g-1})^2}{\sum_i \beta_{i,g-1}^2}} < \texttt{tol}, \tag{53}$$

where $\beta_{i,g}$ is the $i$th coefficient of the $g$th iteration. The default value of the convergence criterion (argument `tol`) is $10^{-5}$.

In the following example, we estimate the model described above by iterated SUR:

```
fitsurit <- systemfit( eqSystem, method = "SUR", maxiter = 500 )
```

*Residual covariance matrix*

It was explained in Section 2.4 that several different methods have been proposed to calculate the residual covariance matrix. The user can specify, which method `systemfit` should use. Possible values of the argument `methodResidCov` are presented in Table 1. By default, `systemfit` uses Equation 38.

| argument `methodResidCov` | equation |
|---|---|
| `"noDfCor"` | 37 |
| `"geomean"` | 38 |
| `"max"` | 39 |
| `"Theil"` | 40 |

Table 1: Possible values of argument `methodResidCov`

Furthermore, the user can specify whether the means should be subtracted from the residuals before Equations 37, 38, 39, or 40 are applied to calculate the residual covariance matrix (see Section 2.4). The corresponding argument is called `centerResiduals`. It must be either `TRUE` (subtract the means) or `FALSE` (take the unmodified residuals). The default value of `centerResiduals` is `FALSE`.

Moreover, if the coefficients are estimated under restrictions, the user can use argument `residCovRestricted` to specify whether the residual covariance matrix for a WLS, SUR, W2SLS, or 3SLS estimation should be obtained from a restricted or from an unrestricted first-step estimation (see Section 2.4). If this argument is `TRUE` (the default), the residual covariance matrix is obtained from a restricted OLS or 2SLS estimation. If it is `FALSE`, the residual covariance matrix is obtained from an unrestricted first-step estimation.

Finally, argument `residCovWeighted` can be used to decide, whether the residual covariance matrix for a SUR (3SLS) estimation should be obtained from a WLS (W2SLS) estimation instead of from an OLS (2SLS) estimation (see Section 2.4). By default, `residCovWeighted` is `FALSE`, which means that the residuals of an OLS (2SLS) estimation are used to compute the residual covariance matrix.

*3SLS formula*

As discussed in Sections 2.2 and 2.3, there exist several different methods to perform a 3SLS estimation. The user can specify the method by argument `method3sls`. Possible values are presented in Table 2. The default value is `"GLS"`.

| argument `method3sls` | equation (unrestricted) | equation (restricted) |
|---|---|---|
| `"GLS"` | 10 | 28 |
| `"IV"` | 13 | 30 |
| `"GMM"` | 15 | 32 |
| `"Schmidt"` | 18 | 34 |
| `"EViews"` | 20 | 36 |

Table 2: Possible values of argument `method3sls`

*Sigma squared*

In case of OLS or 2SLS estimations, argument `singleEqSigma` can be used to specify, whether different $\sigma^2$s for each single equation or the same $\sigma^2$ for all equations should be used. If argument `singleEqSigma` is `TRUE`, $\widehat{\Omega}$ in Equation 8 or 12 is set to $\widehat{\Sigma} \otimes I_T$. In contrast, if argument `singleEqSigma` is `FALSE`, $\widehat{\Omega}$ in Equation 8 or 12 is set to $\hat{\sigma}^2 I_{G \cdot T}$. In case of an unrestricted regression, argument `singleEqSigma` is `TRUE` by default. However, if the coefficients are estimated under restrictions, this argument is `FALSE` by default.

*System options*

Furthermore, two options regarding some internal calculations are available. First, argument `solvetol` specifies the tolerance level for detecting linear dependencies when inverting a matrix or calculating a determinant (using functions `solve` and `det`). The default value depends on the used computer system and is equal to the default tolerance level of `solve` and `det`.

Second, argument `useMatrix` specifies whether the **Matrix** package (Bates and Maechler 2007) should be used for all computations where matrices are involved (see Section 3.4).

*Returned data objects*

Finally, the user can decide whether `systemfit` should return some data objects. Argument `model` indicates whether a data frame with the data of the model should be returned. Its default value is `TRUE`, i.e., the model frame is returned. Arguments `x`, `y`, and `z` specify whether the model matrices ($X_i$), the responses ($y_i$), and the matrices of instrumental variables ($Z_i$), respectively, should be returned. These three arguments are `FALSE` by default, i.e., these data objects are not returned.

### 4.3. Summary results with `summary.systemfit`

The `summary` method can be used to compute and print summary results of objects returned by `systemfit`.

```
summary( fitsur )

##
## systemfit results
```

```
## method: SUR
##
##         N DF     SSR  detRCov   OLS-R2 McElroy-R2
## system 40 33 169.741 0.879285 0.683453   0.788722
##
##         N DF      SSR     MSE    RMSE       R2   Adj R2
## demand 20 17  65.6829 3.86370 1.96563 0.755019 0.726198
## supply 20 16 104.0584 6.50365 2.55023 0.611888 0.539117
##
## The covariance matrix of the residuals used for estimation
##         demand  supply
## demand 3.72539 4.13696
## supply 4.13696 5.78444
##
## The covariance matrix of the residuals
##         demand  supply
## demand 3.86370 4.92431
## supply 4.92431 6.50365
##
## The correlations of the residuals
##          demand   supply
## demand 1.000000 0.982348
## supply 0.982348 1.000000
##
##
## SUR estimates for 'demand' (equation 1)
## Model Formula: consump ~ price + income
##
##               Estimate Std. Error  t value   Pr(>|t|)
## (Intercept) 99.3328942  7.5144525 13.21891 2.2597e-10 ***
## price       -0.2754857  0.0885091 -3.11251  0.0063324 **
## income       0.2985505  0.0419454  7.11760 1.7249e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.96563 on 17 degrees of freedom
## Number of observations: 20 Degrees of Freedom: 17
## SSR: 65.682902 MSE: 3.8637 Root MSE: 1.96563
## Multiple R-Squared: 0.755019 Adjusted R-Squared: 0.726198
##
##
## SUR estimates for 'supply' (equation 2)
## Model Formula: consump ~ price + farmPrice + trend
##
##               Estimate Std. Error t value   Pr(>|t|)
## (Intercept) 61.9661660 11.0807901 5.59222 4.0480e-05 ***
```

```
## price        0.1468841  0.0944351 1.55540 0.13940780
## farmPrice    0.2140040  0.0398684 5.36776 6.2829e-05 ***
## trend        0.3393039  0.0679113 4.99628 0.00013185 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.550226 on 16 degrees of freedom
## Number of observations: 20 Degrees of Freedom: 16
## SSR: 104.05843 MSE: 6.503652 Root MSE: 2.550226
## Multiple R-Squared: 0.611888 Adjusted R-Squared: 0.539117
```

First, the estimation method is reported and a few summary statistics for the entire system and for each equation are given. Then, the covariance matrix used for estimation and the covariance matrix as well as the correlation matrix of the (final) residuals are printed. Finally, the estimation results of each equation are reported: the formula of the estimated equation, the estimated coefficients, their standard errors, $t$ values, $P$ values and codes indicating their statistical significance, as well as some other statistics like the standard error of the residuals and the $R^2$ value of the equation.

*Degrees of freedom for t tests*

The `summary` method for `systemfit` objects has an optional argument `useDfSys`. It selects the approach that is applied by `systemfit` to determine the degrees of freedom of $t$ tests of the estimated coefficients (Section 2.5). If argument `useDfSys` is `TRUE`, the degrees of freedom of the whole system are taken. In contrast, if `useDfSys` is `FALSE`, the degrees of freedom of the single equation are taken. If the coefficients are estimated under restrictions, argument `useDfSys` is `TRUE` by default. However, if no restrictions on the coefficient are specified, the default value of `useDfSys` is `FALSE`.

*Reduce amount of printed output*

The optional arguments `residCov` and `equations` can be used reduce the amount of the printed output. Argument `residCov` specifies whether the covariance matrix and the correlation matrix of the residuals are printed. Argument `equations` specifies whether summary results of each equation are printed. By default, both arguments are `TRUE`. The following command returns a sparse summary output:

```
summary( fitsur, residCov = FALSE, equations = FALSE )

##
## systemfit results
## method: SUR
##
##         N DF     SSR  detRCov   OLS-R2 McElroy-R2
## system 40 33 169.741 0.879285 0.683453   0.788722
##
##         N DF      SSR     MSE    RMSE       R2   Adj R2
```

```
## demand 20 17  65.6829 3.86370 1.96563 0.755019 0.726198
## supply 20 16 104.0584 6.50365 2.55023 0.611888 0.539117
##
##
## Coefficients:
##                       Estimate Std. Error  t value   Pr(>|t|)
## demand_(Intercept) 99.3328942  7.5144525 13.21891 2.2597e-10 ***
## demand_price       -0.2754857  0.0885091 -3.11251 0.00633240 **
## demand_income       0.2985505  0.0419454  7.11760 1.7249e-06 ***
## supply_(Intercept) 61.9661660 11.0807901  5.59222 4.0480e-05 ***
## supply_price        0.1468841  0.0944351  1.55540 0.13940780
## supply_farmPrice    0.2140040  0.0398684  5.36776 6.2829e-05 ***
## supply_trend        0.3393039  0.0679113  4.99628 0.00013185 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

## 4.4. Panel data

The `systemfit` function can also be used for a (classical) "seemingly unrelated regression" analysis with panel data. For this type of analysis, the data must be provided in a transformed data frame of class `pdata.frame`[10] which can be created with the function `pdata.frame` from the R package **plm** (Croissant and Millo 2007). In contrast to the previously described usage of `systemfit`, argument `formula` must be a single equation (object of class `formula`). This formula is estimated for all individuals.

We demonstrate the application of `systemfit` to panel data using an example taken from Greene (2003, p. 340) that is based on Grunfeld (1958). We want to estimate a model for gross investment of 5 US firms in the years 1935–1954:

$$\texttt{invest}_{it} = \beta_1 + \beta_2 \cdot \texttt{value}_{it} + \beta_3 \cdot \texttt{capital}_{it} \tag{54}$$

where `invest` is the gross investment of firm $i$ in year $t$, `value` is the market value of the firm at the end of the previous year, and `capital` is the capital stock of the firm at the end of the previous year.

This model can be estimated by

```
### this code chunk is evaluated only if the 'plm' package is available
data( "GrunfeldGreene" )
library( "plm" )
GGPanel <- pdata.frame( GrunfeldGreene, c( "firm", "year" ) )
greeneSur <- systemfit( invest ~ value + capital, method = "SUR",
   data = GGPanel )
```

[10] Generally, panel data can be either in "long format" (different individuals are arranged below each other) or in "wide format" (different individuals are arranged next to each other). For this analysis, the data must be in "long format".

The first line loads the example data set `GrunfeldGreene` that is included in the **systemfit** package. The second line loads the **plm** package and the following line specifies a data frame of class `pdata.frame`, where the variables `firm` and `year` indicate the individual (cross-section) and time identifier, respectively. Finally, a seemingly unrelated regression is performed.

The optional argument `pooled` is a logical variable indicating whether the coefficients are restricted to be equal for all individuals. By default, this argument is set to `FALSE`. The following command does a seemingly unrelated regression of the same model as before, but with coefficients restricted to be equal for all individuals.

```
### this code chunk is evaluated only if the 'plm' package is available
greeneSurPooled <- systemfit( invest ~ value + capital, method = "SUR",
   data = GGPanel, pooled = TRUE )
```

## 4.5. Testing linear restrictions

As described in Section 2.7, linear restrictions can be tested by an $F$ test, two Wald tests and an LR test. The **systemfit** package provides the method `linearHypothesis` for the $F$ and Wald tests as well as the method `lrtest` for LR tests.

We will now test the restriction $\beta_2 = -\beta_6$ that was specified by the matrix `Rmat` and the vector `qvec` in the example above (p. 18).

```
linearHypothesis( fitsur, Rmat, qvec, test = "FT" )

## Linear hypothesis test (Theil's F test)
##
## Hypothesis:
## demand_price  + supply_farmPrice = 0
##
## Model 1: restricted model
## Model 2: fitsur
##
##   Res.Df Df      F Pr(>F)
## 1     34
## 2     33  1 0.9322 0.3413

linearHypothesis( fitsur, Rmat, qvec, test = "F" )

## Linear hypothesis test (F statistic of a Wald test)
##
## Hypothesis:
## demand_price  + supply_farmPrice = 0
##
## Model 1: restricted model
## Model 2: fitsur
##
```

```
##   Res.Df Df      F Pr(>F)
## 1     34
## 2     33  1 0.6092 0.4407

linearHypothesis( fitsur, Rmat, qvec, test = "Chisq" )

## Linear hypothesis test (Chi^2 statistic of a Wald test)
##
## Hypothesis:
## demand_price  + supply_farmPrice = 0
##
## Model 1: restricted model
## Model 2: fitsur
##
##   Res.Df Df  Chisq Pr(>Chisq)
## 1     34
## 2     33  1 0.6092     0.4351

lrtest( fitsurRmat, fitsur )

## Likelihood ratio test
##
## Model 1: fitsurRmat
## Model 2: fitsur
##   #Df  LogLik Df  Chisq Pr(>Chisq)
## 1   9 -52.117
## 2  10 -51.614  1 1.0043     0.3163
```

The linear restrictions are tested by Theil's $F$ test (Equation 45) first, second by the $F$ statistic of a Wald test (Equation 46), third by the $\chi^2$ statistic of a Wald test (Equation 47), and finally by an LR test (Equation 48).

The first argument of the `linearHypothesis` method for `systemfit` objects must be an unrestricted regression returned by `systemfit`. The second and third arguments are the restriction matrix $R$ and the optional vector $q$, as described in Section 2.3. Analogously to the argument `restrict.matrix` of the `systemfit` function, the restrictions can be specified either in matrix form or symbolically. The optional argument `test` must be a character string, `"FT"`, `"F"`, or `"Chisq"`, specifying whether to compute Theil's finite-sample $F$ test (with approximate $F$ distribution) the finite-sample Wald test (with approximate $F$ distribution) or the large-sample Wald test (with asymptotic $\chi^2$ distribution).

All arguments of the `lrtest` method for `systemfit` objects must be fitted model objects returned by `systemfit`. It consecutively compares all provided fitted model objects.

All tests print a short description of the test and the tested model objects first. Then, a small table is printed, where each row belongs to one (unrestricted or restricted) model. The second row reports (amongst others) the degree(s) of freedom of the test, the empirical test statistic, and the marginal level of significance ($P$ value). Although all tests check the same hypothesis,

there is some variation of the $P$ values. However, all tests suggest the same decision: The null hypothesis $\beta_2 = -\beta_6$ cannot be rejected at any reasonable level of significance.

### 4.6. Hausman test

A Hausman test, which is described in Section 2.8, can be carried out with following commands:

```
fit2sls  <- systemfit( eqSystem, method = "2SLS",
   inst = ~ income + farmPrice + trend, data = Kmenta )
fit3sls <- systemfit( eqSystem, method = "3SLS",
   inst = ~ income + farmPrice + trend, data = Kmenta )
hausman.systemfit( fit2sls, fit3sls )

##
##  Hausman specification test for consistency of the 3SLS estimation
##
## data:  Kmenta
## Hausman = 2.5357, df = 7, p-value = 0.9244
```

First of all, the model is estimated by 2SLS and then by 3SLS. Finally, in the last line the test is carried out by the command `hausman.systemfit`. This function requires two arguments: the result of a 2SLS estimation and the result of a 3SLS estimation. The Hausman test statistic is 2.536, which has a $\chi^2$ distribution with 7 degrees of freedom under the null hypothesis. The corresponding $P$ value is 0.924. This shows that the null hypothesis is not rejected at any reasonable level of significance. Hence, we can assume that the 3SLS estimator is consistent.

## 5. Replication of textbook results

In this section, we reproduce results from several textbook examples using the **systemfit** package for several reasons. First, a comparison of **systemfit**'s results with results published in the literature confirms the reliability of the **systemfit** package. Second, this section helps teachers and students become familiar with using the **systemfit** package. Third, the section encourages reproducible research, which should be a general goal in scientific analysis (Buckheit and Donoho 1995; Schwab, Karrenbach, and Claerbout 2000). For instance, by preparing this section, the exact estimation methods of the replicated analyses have been discovered and a few errors in Greene (2003) have been found (see Greene 2006a).

### 5.1. Kmenta (1986): Example on p. 685 (food market)

First, we reproduce an example taken from Kmenta (1986, p. 685). The data are available from Table 13-1 (p. 687), and the results are presented in Table 13-2 (p. 712) of this book.

Before starting the estimation, we load the data and specify the two formulas of the model as well as the instrumental variables. Then the equation system is estimated by OLS, 2SLS, 3SLS, and iterated 3SLS. After each estimation, we provide the commands to print the estimated coefficients.

```
data( "Kmenta" )
eqDemand <- consump ~ price + income
eqSupply <- consump ~ price + farmPrice + trend
inst <- ~ income + farmPrice + trend
system <- list( demand = eqDemand, supply = eqSupply )
```

OLS estimation:

```
fitOls <- systemfit( system, data = Kmenta )
round( coef( summary( fitOls ) ), digits = 4 )
```

2SLS estimation:

```
fit2sls <- systemfit( system, method = "2SLS", inst = inst, data = Kmenta )
round( coef( summary( fit2sls ) ), digits = 4 )
```

3SLS estimation:

```
fit3sls <- systemfit( system, method = "3SLS", inst = inst, data = Kmenta )
round( coef( summary( fit3sls ) ), digits = 4 )
```

Iterated 3SLS estimation:

```
fitI3sls <- systemfit( system, method = "3SLS", inst = inst, data = Kmenta,
   maxit = 250 )
round( coef( summary( fitI3sls ) ), digits = 4 )
```

The above commands return exactly the same coefficients and standard errors as published in Kmenta (1986, p. 712) except for two minor exceptions: two standard errors of the 2SLS estimation deviate by 0.0001. However, this difference is likely due to rounding errors in `systemfit` or Kmenta (1986) and is so small that it empirically does not matter.

### 5.2. Greene (2003): Example 15.1 (Klein's model I)

Second, we try to replicate Klein's "Model I" (Klein 1950) that is described in Greene (2003, p. 381). The data are available from the online complements to Greene (2003), Table F15.1 (`http://pages.stern.nyu.edu/~wgreene/Text/tables/TableF15-1.txt`), and the estimation results are presented in Table 15.3 (p. 412).

Initially, the data are loaded and three equations as well as the instrumental variables are specified. As in the example before, the equation system is estimated by OLS, 2SLS, 3SLS, and iterated 3SLS, and commands to print the estimated coefficients are presented.

```
data( "KleinI" )
eqConsump  <- consump ~ corpProf + corpProfLag + wages
eqInvest   <- invest ~ corpProf + corpProfLag + capitalLag
```

```
eqPrivWage <- privWage ~ gnp + gnpLag + trend
inst <- ~ govExp + taxes + govWage + trend + capitalLag + corpProfLag + gnpLag
system <- list( Consumption = eqConsump, Investment = eqInvest,
   PrivateWages = eqPrivWage )
```

OLS estimation:

```
kleinOls <- systemfit( system, data = KleinI )
round( coef( summary( kleinOls ) ), digits = 3 )
```

2SLS estimation:

```
klein2sls <- systemfit( system, method = "2SLS", inst = inst, data = KleinI,
   methodResidCov = "noDfCor" )
round( coef( summary( klein2sls ) ), digits = 3 )
```

3SLS estimation:

```
klein3sls <- systemfit( system, method = "3SLS", inst = inst, data = KleinI,
   methodResidCov = "noDfCor" )
round( coef( summary( klein3sls ) ), digits = 3 )
```

iterated 3SLS estimation:

```
kleinI3sls <- systemfit( system, method = "3SLS", inst = inst, data = KleinI,
   methodResidCov = "noDfCor", maxit = 500 )
round( coef( summary( kleinI3sls ) ), digits = 3 )
```

Again, these commands return almost the same results as published in Greene (2003).[11] There are only two minor deviations, where these values differ merely in the last digit.

### 5.3. Greene (2003): Example 14.1 (Grunfeld's investment data)

Third, we reproduce Example 14.1 of Greene (2003, p. 340) that is based on Grunfeld (1958). The data are available from the online complements to Greene (2003), Table F13.1 (`http://pages.stern.nyu.edu/~wgreene/Text/tables/TableF13-1.txt`). Several different versions of the "Grunfeld" data set can be found, whereas the version of Greene (2003) is considered incorrect (Cummins 2001). However, we use this incorrect version to replicate the results in Greene (2003), Tables 14.1 and 14.2 (p. 351).[12]

First, we load the data and the **plm** package, indicate the individual (cross-section) and time identifiers, and specify the formula to be estimated. Then, the system is estimated by OLS, pooled OLS, SUR, and pooled SUR. After each estimation, we show the commands to print the estimated coefficients, the $\sigma^2$ values of the OLS estimations, and the residual covariance matrix as well as the residual correlation matrix of the SUR estimations.

[11] There are two typos in Table 15.3 (p. 412). Please take a look at the errata (Greene 2006a).

[12] A correct version of this data set with five additional firms is available as data set `Grunfeld` in the **Ecdat** package (Croissant 2006).

```
### this code chunk is evaluated only if the 'plm' package is available
data( "GrunfeldGreene" )
library( "plm" )
GGPanel <- pdata.frame( GrunfeldGreene, c( "firm", "year" ) )
formulaGrunfeld <- invest ~ value + capital
```

OLS estimation (Table 14.2):

```
### this code chunk is evaluated only if the 'plm' package is available
greeneOls <- systemfit( formulaGrunfeld,
   data = GGPanel )
round( coef( summary( greeneOls ) ), digits = 4 )
round( sapply( greeneOls$eq, function(x){return(summary(x)$ssr/20)} ), digits = 3 )
```

pooled OLS (Table 14.2):

```
### this code chunk is evaluated only if the 'plm' package is available
greeneOlsPooled <- systemfit( formulaGrunfeld,
   data = GGPanel, pooled = TRUE )
round( coef( summary( greeneOlsPooled$eq[[1]] ) ), digits = 4 ) #$
sum( sapply( greeneOlsPooled$eq, function(x){return(summary(x)$ssr)}) )/100
```

SUR estimation (Table 14.1):

```
### this code chunk is evaluated only if the 'plm' package is available
greeneSur <- systemfit( formulaGrunfeld, method = "SUR",
   data = GGPanel, methodResidCov = "noDfCor" )
round( coef( summary( greeneSur ) ), digits = 4 )
round( greeneSur$residCov, digits = 3 ) #$
round( summary( greeneSur )$residCor, digits = 3 ) #$
```

pooled SUR estimation (Table 14.1):

```
### this code chunk is evaluated only if the 'plm' package is available
greeneSurPooled <- systemfit( formulaGrunfeld, method = "SUR",
   data = GGPanel, pooled = TRUE, methodResidCov = "noDfCor",
   residCovWeighted = TRUE )
round( coef( summary( greeneSurPooled$eq[[1]] ) ), digits = 4 ) #$

round( greeneSurPooled$residCov, digits = 3 ) #$
round( cov( residuals( greeneSurPooled ) ), digits = 3 )
round( summary( greeneSurPooled )$residCor, digits = 3 ) #$
```

These commands return nearly the same results as published in Greene (2003).[13] We present two different commands to print the residual covariance matrix of the pooled SUR estimation.

[13] There are several typos and errors in Table 14.1 (p. 412). Please take a look at the errata of this book (Greene 2006a).

The first calculates the covariance matrix without centering the residuals (see Section 2.4); the returned values are equal to those published in Greene (2003, p. 351). The second command calculates the residual covariance matrix after centering the residuals; these returned values are equal to those published in the errata (Greene 2006a).

### 5.4. Theil (1971): Example on p. 295ff (Grunfeld's investment data)

Finally, we estimate an example taken from Theil (1971, p. 295ff) that is also based on Grunfeld (1958). The data are available in Table 7.1 of Theil (1971, p. 296). They are a subset of the data set published by Greene (2003) (see Section 5.3).

After extracting the data from the `GrunfeldGreene` data set, the individual (cross-section) and time identifiers are indicated. Then, the formula is specified, and the model is estimated by OLS and SUR. Commands to print the estimated coefficients are reported after each estimation.

```
### this code chunk is evaluated only if the 'plm' package is available
GrunfeldTheil <- subset( GrunfeldGreene,
   firm %in% c( "General Electric", "Westinghouse" ) )
GTPanel <- pdata.frame( GrunfeldTheil, c( "firm", "year" ) )
formulaGrunfeld <- invest ~ value + capital
```

OLS estimation (page 295)

```
### this code chunk is evaluated only if the 'plm' package is available
theilOls <- systemfit( formulaGrunfeld,
   data = GTPanel )
round( coef( summary( theilOls ) ), digits = 3 )
```

SUR estimation (page 300)

```
### this code chunk is evaluated only if the 'plm' package is available
theilSur <- systemfit( formulaGrunfeld, method = "SUR",
   data = GTPanel, methodResidCov = "noDfCor" )
round( coef( summary( theilSur ) ), digits = 3 )
```

These commands return exactly the same results as published in Theil (1971, pp. 295, 300).

Now, we apply an $F$ test to check whether the slope parameters are equal for General Electric and Westinghouse (pages 313–315). Then we re-estimate the model under these restrictions on the coefficients.

$F$ test (page 313–315)[14]

```
### this code chunk is evaluated only if the 'plm' package is available
RMatrix <- rbind( c( 0, 1, 0, 0, -1, 0 ), c( 0, 0, 1, 0, 0, -1 ) )
linearHypothesis( theilSur, RMatrix )
```

[14] The same restriction can be specified also symbolically by `RMatrix <- c("General.Electric_value = Westinghouse_value", "General.Electric_capital = Westinghouse_capital")`

Restricted SUR estimation (page 316)

```
### this code chunk is evaluated only if the 'plm' package is available
theilSurRestr <- systemfit(formulaGrunfeld, method = "SUR",
   data = GTPanel, methodResidCov = "noDfCor", restrict.matrix = RMatrix,
   residCovRestricted = FALSE)
round(coef(summary(theilSurRestr)), digits = 3)
```

The method `linearHypothesis` returns the same value of the $F$ statistic as published in Theil (1971, p. 315). Hence, we arrive at the the same conclusion: we accept the null hypothesis (restrictions on the coefficients are true) at the 5 percent significance level.

Also the results of the restricted SUR estimation are identical to the results published in Theil (1971, p. 316).

## 6. Summary and outlook

In this article, we have described some of the basic features of the **systemfit** package for estimating systems of linear equations. Many details of the estimation can be controlled by the user. Furthermore, the package provides some statistical tests for restrictions on the coefficients and consistency of 3SLS estimation. It has been tested on a variety of datasets and has produced satisfactory for a few years. While the **systemfit** package performs the basic fitting methods, more sophisticated tools exist. We hope to implement missing functionalities in the near future.

### *Unbalanced datasets*

Currently, the **systemfit** package requires that all equations have the same number of observations. However, many data sets have unbalanced observations.[15] Simply dropping data points that do not contain observations for all equations may reduce the number of observations considerably, and thus, the information utilized in the estimation. Hence, it is our intention to include the capability for estimations with unbalanced data sets as described in Schmidt (1977) in future releases of **systemfit**.

### *Serial correlation and heteroscedasticity*

For all of the methods developed in the package, the disturbances of the individual equations are assumed to be independent and identically distributed (iid). The package could be enhanced by the inclusion of methods to fit equations with serially correlated and heteroscedastic disturbances (Parks 1967).

### *Estimation methods*

In the future, we wish to include more sophisticated estimation methods such as limited information maximum likelihood (LIML), full information maximum likelihood (FIML), generalized methods of moments (GMM) and spatial econometric methods (Paelinck and Klaassen 1979; Anselin 1988).

[15] For instance, forestry datasets typically contain many observations of inexpensive variables (stem diameter, tree count) and few expensive variables such as stem height or volume.

*Non-linear estimation*

Finally, the **systemfit** package provides a function to estimate systems of non-linear equations. However, the function `nlsystemfit` is currently under development and the results are not yet always reliable due to convergence difficulties.

## Acknowledgments

We thank Achim Zeileis, John Fox, Ott Toomet, William H. Greene, two anonymous referees and several users of **systemfit** for their comments and suggestions that helped us to improve the **systemfit** package as well as this paper. Of course, any remaining errors are the authors'.

## References

Anselin L (1988). *Spatial Econometrics: Methods and Models*. Kluwer Academic, Dordrecht.

Bates D (2004). "Least Squares Calculations in R." *R News*, **4**(1), 17–20. URL http://CRAN.R-project.org/doc/Rnews/.

Bates D, Maechler M (2007). *Matrix: A Matrix Package for R*. R package version 0.99875-2, URL http://CRAN.R-project.org/.

Buckheit J, Donoho DL (1995). "Wavelab and Reproducible Research." In A Antoniadis (ed.), *Wavelets and Statistics*. Springer.

Croissant Y (2006). *Ecdat: Data Sets for Econometrics*. R package version 0.1-5, URL http://CRAN.R-project.org/.

Croissant Y, Millo G (2007). ***plm**: Linear Models for Panel Data*. R package version 0.3-1, URL http://CRAN.R-project.org/.

Cummins C (2001). "Different Versions of Grunfeld Dataset." URL http://www.stanford.edu/~clint/bench/grunfeld.htm.

Fox J (2002). *An R and S-Plus Companion to Applied Regression*. SAGE Publications Ltd, Thousand Oaks.

Fox J (2006). *car: Companion to Applied Regression*. R package version 1.2-1, URL http://CRAN.R-project.org/.

Fox J (2011). *sem: Structural Equation Models*. R package version 2.0, URL http://CRAN.R-project.org/.

Greene WH (2003). *Econometric Analysis*. 5th edition. Prentice Hall.

Greene WH (2006a). "Errata and Discussion to Econometric Analysis, 5th edition." URL http://pages.stern.nyu.edu/~wgreene/Text/Errata/ERRATA5.htm.

Greene WH (2006b). "Information about SUR Estimation in LIMDEP." Personal email on 2006/02/16.

Grunfeld Y (1958). *The Determinants of Corporate Investment*. Ph.D. thesis, University of Chicago.

Hausman JA (1978). "Specification Test in Econometrics." *Econometrica*, **46**(6), 1251–1272.

Henningsen A, Hamann JD (2007). "systemfit: A Package for Estimating Systems of Simultaneous Equations in R." *Journal of Statistical Software*, **23**(4), 1–40. URL http://www.jstatsoft.org/v23/i04/.

Judge GG, Griffiths WE, Hill RC, Lütkepohl H, Lee TC (1985). *The Theory and Practice of Econometrics*. 2nd edition. John Wiley and Sons, New York.

Judge GG, Hill RC, Griffiths W, Lütkepohl H, Lee TC (1982). *Introduction to the Theory and Practice of Econometrics*. John Wiley and Sons, New York.

Klein L (1950). *Economic Fluctuations in the United States, 1921–1941*. John Wiley, New York.

Kmenta J (1986). *Elements of Econometrics*. 2 edition. Macmillan, New York.

McElroy MB (1977). "Goodness of Fit for Seemingly Unrelated Regressions." *Journal of Econometrics*, **6**, 381–387.

Paelinck JHP, Klaassen LH (1979). *Spatial Econometrics*. Saxon House, Farnborough.

Parks RW (1967). "Efficient Estimation of a System of Regression Equations when Disturbances are both Serially and Contemporaneously Correlated." *Journal of the American Statistical Association*, **62**(318), 500–509.

R Development Core Team (2007). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL http://www.R-project.org/.

Schmidt P (1977). "Estimation of Seemingly Unrelated Regressions with Unequal Numbers of Observations." *Journal of Econometrics*, **5**, 365–377.

Schmidt P (1990). "Three-Stage Least Squares with Different Instruments for Different Equations." *Journal of Econometrics*, **43**, 389–394.

Schwab M, Karrenbach M, Claerbout J (2000). "Making Scientific Computations Reproducible." *Computing in Science & Engineering*, **2**(6), 61–67.

Srivastava VK, Giles DEA (1987). *Seemingly Unrelated Regression Equations Models*. Marcel Dekker, Inc., New York.

Theil H (1971). *Principles of Econometrics*. Wiley, New York.

Zeileis A, Hothorn T (2002). "Diagnostic Checking in Regression Relationships." *R News*, **2**(3), 7–10. URL http://CRAN.R-project.org/doc/Rnews/.

Zellner A (1962). "An Efficient Method of Estimating Seemingly Unrelated Regressions and Tests for Aggregation Bias." *Journal of the American Statistical Association*, **57**, 348–368.

Zellner A, Huang DS (1962). "Further Properties of Efficient Estimators for Seemingly Unrelated Regression Equations." *International Economic Review*, **3**(3), 300–313.

Zellner A, Theil H (1962). "Three-Stage Least Squares: Simultaneous Estimation of Simultaneous Equations." *Econometrica*, **30**(1), 54–78.

Zivot E, Wang J (2006). *Modeling Financial Time Series with S-PLUS*. 2nd edition. Springer, New York.

# A. Object returned by `systemfit`

`systemfit` returns a list of class `systemfit` that contains the results that belong to the entire system of equations. One special element of this list is called `eq`. It is a list that contains one object for each estimated equation. These objects are of the class `systemfit.equation` and contain the results that belong only to the regarding equation.

| lm | systemfit | systemfit.equation |
|---|---|---|
| coefficients | coefficients | coefficients |
| | coefCov | coefCov |
| fitted.values | | fitted.values |
| residuals | | residuals |
| | residCov | |
| | residCovEst | |
| rank | rank | rank |
| | | rank.sys |
| | | nCoef.sys |
| df.residual | df.residual | df.residual |
| | | df.residual.sys |
| call | call | |
| terms | | terms |
| | | inst |
| weights | | |
| contrasts | | |
| xlevels | | |
| offset | | |
| model* | | model* |
| x** | | x** |
| y** | | y** |
| | | z** |
| | iter | |
| | eq | |
| | | eqnLabel |
| | | eqnNo |
| | method | method |
| | panelLike | |
| | restrict.matrix | |
| | restrict.rhs | |
| | restrict.regMat | |
| | control | |

Table 3: Elements returned by `systemfit` and `lm` (* if requested by the user with default `TRUE`, ** if requested by the user with default `FALSE`).

The elements returned by `systemfit` are similar to those returned by `lm`, the basic tool for linear regressions in R. While some counterparts of elements returned by `lm` can be found directly in objects of class `systemfit`, other counterparts are available for each equation in objects of class `systemfit.equation`. This is demonstrated in Table 3.

## B. Computation times

Theoretically, one would expect that the calculations with the **Matrix** package are faster and more robust than calculations with the traditional method. To test this hypothesis, we use function `createSystemfitModel` to create a medium-sized multi-equation model with 8 equations, 10 regressors in each equation (without constant), and 750 observations. Then, we estimated this model with and without using the **Matrix** package. Finally, the results are compared.

```
library( "systemfit" )
set.seed( 1 )
systemfitModel <- createSystemfitModel( nEq = 8, nReg = 10, nObs = 750 )
system.time(
   fitMatrix <- systemfit( systemfitModel$formula, method = "SUR",
      data = systemfitModel$data )
)

##    user  system elapsed
##   0.216   0.032   0.247

system.time(
   fitTrad <- systemfit( systemfitModel$formula, method = "SUR",
      data = systemfitModel$data, useMatrix = FALSE )
)

##    user  system elapsed
##   0.501   0.084   0.585

all.equal( fitMatrix, fitTrad )

## [1] "Component \"call\": target, current do not match when deparsed"
## [2] "Component \"control\": Component \"useMatrix\": 1 element mismatch"
```

The returned computation times clearly show that using the **Matrix** package makes the estimation faster. The comparison of the estimation results shows that both methods return the same results. The only differences between the returned objects are — as expected — the `call` and the stored control variable `useMatrix`.

However, the estimation of rather small models is much slower with the **Matrix** package than without this package. Moreover, the differences in computation time accumulate, if the estimation is iterated.

```
smallModel <- createSystemfitModel( nEq = 3, nReg = 4, nObs = 50 )
system.time(
   fitSmallMatrix <- systemfit( smallModel$formula, method = "SUR",
      data = smallModel$data, maxit = 500 )
)
```

```
##    user  system elapsed
##   0.087   0.000   0.087

system.time(
   fitSmallTrad <- systemfit( smallModel$formula, method = "SUR",
      data = smallModel$data, maxit = 500, useMatrix = FALSE )
)

##    user  system elapsed
##   0.007   0.000   0.008

all.equal( fitSmallMatrix, fitSmallTrad )

## [1] "Component \"call\": target, current do not match when deparsed"
## [2] "Component \"control\": Component \"useMatrix\": 1 element mismatch"
```

As mentioned above, the usage of the **Matrix** package clearly increases the computation times for iterated (SUR) estimations of small models with small data sets.

## C. Estimating systems of equations with `sem`

This section compares the commands to estimate a system of equations by `sem` and `systemfit`. This comparison uses Klein's "Model I" (see Section 5.2). Before starting the estimation, we load the **sem** and **systemfit** package as well as the required data set.

```
### this code chunk is evaluated only if the 'sem' package is available
library( "sem" )
library( "systemfit" )
data( "KleinI" )
```

First, we estimate the system by limited information maximum likelihood (LIML) with `sem`:

```
### this code chunk is evaluated only if the 'sem' package is available
limlRam <- matrix(c(
   "consump  <-  corpProf",     "consump_corpProf",     NA,
   "consump  <-  corpProfLag", "consump_corpProfLag", NA,
   "consump  <-  wages",        "consump_wages",        NA,
   "invest   <-  corpProf",     "invest_corpProf",      NA,
   "invest   <-  corpProfLag", "invest_corpProfLag",  NA,
   "invest   <-  capitalLag",  "invest_capitalLag",    NA,
   "privWage <-  gnp",          "privWage_gnp",         NA,
   "privWage <-  gnpLag",       "privWage_gnpLag",      NA,
   "privWage <-  trend",        "privWage_trend",       NA,
   "consump  <-> consump",      "s11", NA,
```

```
   "privWage <-> privWage",    "s22", NA,
   "invest   <-> invest",      "s33", NA),
   ncol = 3, byrow = TRUE)
class(limlRam) <- "mod"
exogVar <- c("corpProf", "corpProfLag", "wages", "capitalLag", "trend",
   "gnp", "gnpLag")
endogVar <- c("consump", "invest", "privWage")
allVar <- c(exogVar, endogVar)

limlResult <- sem(model = limlRam, S = cov(KleinI[ -1, allVar ]),
   N = (nrow(KleinI) - 1), fixed.x = exogVar)
print(limlResult)

##
##  Model Chisquare =  108.5145   Df =  15
##
##    consump_corpProf consump_corpProfLag       consump_wages
##           0.1929344           0.0898849           0.7962187
##     invest_corpProf  invest_corpProfLag   invest_capitalLag
##           0.4796356           0.3330387          -0.1117947
##        privWage_gnp     privWage_gnpLag      privWage_trend
##           0.4394770           0.1460899           0.1302452
##                 s11                 s22                 s33
##           0.8939724           0.5002375           0.8661351
##
##  Iterations =  0
```

Theoretically, the LIML results should be identical to OLS results. Therefore, we re-estimate this model by OLS with `systemfit`.

```
eqConsump <- consump ~ corpProf + corpProfLag + wages
eqInvest <- invest ~ corpProf + corpProfLag + capitalLag
eqPrivWage <- privWage ~ gnp + gnpLag + trend
system <- list(consump = eqConsump, invest = eqInvest,
   privWage = eqPrivWage)
olsResult <- systemfit(system, data = KleinI)
print(olsResult)

##
## systemfit results
## method: OLS
##
## Coefficients:
##  consump_(Intercept)     consump_corpProf  consump_corpProfLag
##            16.2366003           0.1929344           0.0898849
##        consump_wages   invest_(Intercept)      invest_corpProf
```

```
##            0.7962187           10.1257885            0.4796356
##   invest_corpProfLag    invest_capitalLag privWage_(Intercept)
##            0.3330387           -0.1117947            1.4970438
##         privWage_gnp      privWage_gnpLag       privWage_trend
##            0.4394770            0.1460899            0.1302452
```

As expected, the results are identical.

Now, we estimate the system by full information maximum likelihood (FIML) with `sem`:

```
### this code chunk is evaluated only if the 'sem' package is available
fimlRam <- rbind(limlRam,
   c("consump  <-> invest",   "s12", NA),
   c("consump  <-> privWage", "s13", NA),
   c("privWage <-> invest",   "s23", NA))
class(fimlRam) <- "mod"

fimlResult <- sem(model = fimlRam, S = cov(KleinI[ -1, allVar ]),
   N = (nrow(KleinI) - 1), fixed.x = exogVar)
print(fimlResult)

##
##  Model Chisquare =  92.05881   Df =  12
##
##    consump_corpProf consump_corpProfLag       consump_wages
##          0.30160254          0.04239037          0.78017329
##     invest_corpProf  invest_corpProfLag   invest_capitalLag
##          0.38068528          0.41092158         -0.13826099
##        privWage_gnp     privWage_gnpLag      privWage_trend
##          0.37050390          0.20764029          0.18453865
##                 s11                 s22                 s33
##          0.97698124          0.68197216          0.93669694
##                 s12                 s13                 s23
##          0.05836130         -0.59677693          0.31379062
##
##  Iterations =  84
```

Theoretically, results of an iterated SUR estimation should converge to FIML results. Therefore, we re-estimate this model by iterated SUR with `systemfit`.

```
surResult <- systemfit( system, method = "SUR", data = KleinI,
   methodResidCov = "noDfCor", maxit = 500 )
print( surResult )

##
## systemfit results
## method: iterated SUR
```

```
##
## convergence achieved after 18 iterations
##
## Coefficients:
##  consump_(Intercept)     consump_corpProf  consump_corpProfLag
##            15.8445600             0.3015609             0.0424001
##        consump_wages   invest_(Intercept)      invest_corpProf
##            0.7801850            15.8278109             0.3807044
##   invest_corpProfLag    invest_capitalLag privWage_(Intercept)
##            0.4109122            -0.1382606             2.0699937
##         privWage_gnp      privWage_gnpLag       privWage_trend
##            0.3705266             0.2076226             0.1845203
```

As expected, the results are rather similar.

**Affiliation:**

Arne Henningsen
Institute of Food and Resource Economics
University of Copenhagen
Rolighedsvej 25
D-1958 Frederiksberg C, Denmark
E-mail: arne.henningsen@gmail.com
URL: http://www.arne-henningsen.name/

Jeff D. Hamann
Forest Informatics, Inc.
PO Box 1421
Corvallis, Oregon 97339-1421, United States of America
E-mail: jeff.hamann@forestinformatics.com
URL: http://www.forestinformatics.com/