semPLS: Structural Equation Modeling Using Partial Least Squares

Armin Monecke
Ludwig–Maximilians–Universität
München

Friedrich Leisch
Universität für Bodenkultur
Wien

Abstract

This introduction to the R package **semPLS** is a (slightly) modified version of Monecke and Leisch (2012), published in the *Journal of Statistical Software*.

Structural equation models (SEM) are very popular in many disciplines. The partial least squares (PLS) approach to SEM offers an alternative to covariance based SEM, which is especially suited for situations when data is not normally distributed. PLS path modelling is referred to as *soft–modeling–technique* with minimum demands regarding measurement scales, sample sizes and residual distributions. The **semPLS** package provides the capability to estimate PLS path models within the R programming environment. Different setups for the estimation of factor scores can be used. Furthermore it contains modular methods for computation of bootstrap confidence intervals, model parameters and several quality indices. Various plot functions help to evaluate the model. The well known mobile phone dataset from marketing research is used to demonstrate the features of the package.

Keywords: structural equation model, path model,partial least squares, R.

1. Introduction

Within the academic literature of many fields, Rigdon (1998) remarks, structural equation modeling (SEM) has taken up a prominent role. Whenever researchers deal with relations between constructs such as satisfaction, role ambiguity, or attitude, SEM is likely to be the methodology of choice. Since SEM is designed for working with multiple related equations simultaneously, it offers a number of advantages over some more familiar methods and therefore provides a general framework for linear modeling. SEM allows great flexibility on how the equations are specified. The development of an evocative graphical language (McArdle 1980; McArdle and McDonald 1984) has accompanied the development of SEM as a statistical method. Due to this language, complex relationships can be presented in a convenient and powerful way to others not familiar with SEM.

The partial least squares approach to SEM (or PLS path modeling), originally developed by Wold (1966, 1982, 1985) and Lohmöller (1989), offers an alternative to the more prominent covariance-based (CBSEM, Jöreskog 1978). Whereas CBSEM estimates model parameters so that the discrepancy between the estimated and sample covariance matrices is minimized, in PLS path models the explained variance of the endogenous latent variables is maximized by estimating partial model relationships in an iterative sequence of ordinary least squares

(OLS) regressions (e.g., Hair, Ringle, and Sarstedt 2011b). It is worth mentioning that in PLS path modeling latent variable (LV) scores are estimated as exact linear combinations of their associated manifest variables (MVs) and treats them as error free substitutes for the manifest variables. Whereas CBSEM requires *hard* distributional assumptions, PLS path modeling is a *soft-modeling-technique* with less rigid distributional assumptions on the data. At this point it should be mentioned, that PLS path modeling is not to be confused with PLS regression. According to Chin (1998) it can be argued, that depending on the researcher's objectives and epimistic view of data to theory, properties of the data at hand or level of theoretical knowledge and measurement development, PLS path modeling is more suitable. Additionally, great interest in applying PLS path models has been stimulated by the increasing need in modeling so called formative constructs, especially in marketing and management/organizational research (e.g., Diamantopoulos and Winklhofer 2001; Jarvis, MacKenzie, and Podsakoff 2003; MacKenzie, Podsakoff, and Jarvis 2005). The application of PLS path models in marketing is discussed in depth by Henseler, Ringle, and Sinkovics (2009) and Hair, Sarstedt, Ringle, and Mena (2011a). For a related discussion in the field of management information systems, see Ringle, Sarstedt, and Straub (2012).

The **semPLS** is a package for structural equation modeling (SEM) with partial least squares (PLS) in R (R Development Core Team 2012). It is available from the Comprehensive R Archive Network at `http://CRAN.R-project.org/package=semPLS`. One of the major design goals is to provide a comprehensive open-source reference implementation. The package offers

- modular methods for model fitting, calculation of quality indices, etc.,
- plotting features for better understanding of the multivariate model data,
- a convenient user interface for specifying, manipulating, importing and exporting model specifications,
- and an easily extensible infrastructure.

Within the package there are two central methods. The first is `plsm` which is used to create valid model specifications. The second is `sempls` which fits the model, specified with `plsm`. Factor scores can be estimated by using three different weighting schemes: centroid, factorial and path weighting. For the calculation of the outer weights, correlations can be calculated by using Pearson-correlations for continuous data or Spearman- or Kendall-correlations when the scale of the data has rather ordinal character. If the data contains missing values it is possible to use pairwise correlations to compute outer weights. In addition to the estimated factor scores and outer weights, `sempls` computes loadings, path coefficients and total effects, as those are the parameters of interest. For the outer loadings/weights and path coefficients different types of bootstrap confidence intervals and standard errors are available. Calculation of quality indices (R^2, Q^2, Dillon-Goldstein's ρ, etc.) is done via specific methods. PLS path models specified with `plsm` can be easily manipulated by a variety of utility methods. Models specified in **SmartPLS** can be imported. Several plot types (e.g. pairs plots of MV blocks, convergence diagnostic of outer weights, kernel density estimates of residuals/bootstrap parameters, parallel coordinates of bootstrap parameters, etc.) support the researcher in evaluating their models. Finally a graphical representation of the model including outer loadings and path coefficients can be written to a DOT file which can be rendered and plotted by `dot` (Gansner, Koutsofios, and North 2006), a layout program contained in

Graphviz (AT&T Research 2009). **Graphviz** is an open-source graph visualization software. When it is intended to also estimate the model by the covariance-based approach (CBSEM), the model can be exported to an object of class `semmod` and fitted with `sem` (Fox 2006; Fox, Nie, and Byrnes 2012), see Section 5.

In the development process of the **semPLS** package we checked the results for model parameters against those obtained by a list of other PLS path modeling software. This list includes **SmartPLS** (Ringle, Wende, and Will 2005), **XLSTAT-PLSPM** (Esposito Vinzi, Fahmy, Chatelin, and Tenenhaus 2007, in cooperation with Addinsoft France, see `http://www.xlstat.com/en/products/xlstat-plspm/`) and the **plspm** package (Sanchez and Trinchera 2012). Note, that **SmartPLS** and **XLSTAT-PLSPM** are closed source and **plspm** is licensed under the General Public License (GPL ≥ 2). All differences in model parameters due to the used software were in line with the predefined tolerance for the outer weights.

SmartPLS: **SmartPLS** is a stand alone software specialized for PLS path models. It is built on a Java **Eclipse** platform making it operating system independent. The model is specified via drag & drop by drawing the structural model for the latent variables and by assigning the indicators to the latent variables. Data files of various formats can be loaded. After fitting a model, coefficients are added to the plot. More detailed output is provided in plain text, LaTeX and HTML format. The graph representing the model can be exported to PNG. Besides bootstrapping and blindfolding methods it supports the specification of interaction effects. A special feature of **SmartPLS** is the finite mixture routine (FIMIX), a method to deal with unobserved heterogeneity (e.g., Ringle, Wende, and Will 2010; Sarstedt and Ringle 2010; Sarstedt, Becker, M., and Schwaiger 2011).

XLSTAT-PLSPM: **XLSTAT** (Addinsoft 2011) is a modular statistical software relying on Microsoft **Excel** for the input of data and the display of results, but the computations are done using autonomous software components. **XLSTAT-PLSPM** is integrated in XLSTAT as a module for the estimation of PLS path models. It is developed by a research team from the Department of Mathematics and Statistics of the University of Naples in Italy and Addinsoft in France and implements all methodological features and most recent findings of the PLEASURE (Partial LEAst Squares strUctural Relationship Estimation) technology by Esposito Vinzi *et al.* (2007). Special features of **XLSTAT-PLSPM** are multi-group comparisons (Chin and Dibbern 2010) and the REBUS segmentation approach (Esposito Vinzi, Trinchera, and Amato 2010) for treatment of unobserved heterogeneity.

plspm in R: The **plspm** package implements PLS methods with emphasis on structural equation models in R. The fitting method `plspm.fit` returns a list including all the estimated parameters and almost all statistics associated with PLS path models. The `print` method gives an overview of the following list elements: outer model, inner model, scaled LVs, LVs for `scaled = FALSE`, outer weights, loadings, path coefficients matrix, R^2, outer correlations, summary inner model, total effects, unidimensionality, goodness-of-fit, bootstrap results (only if activated) and the data matrix. A `summary` is available, which basically returns the latter list including some formatting. A plot method creates a graphical representation of the model including estimated parameters. For treatment of observed heterogeneity **pathmox** (Sanchez and Aluja 2012) is provided as companion package.

For a long time **LVPLS** 1.8 (Lohmöller 1987) was the only available software for PLS path modeling. The DOS-based program includes two different modules for estimating path models. The `LVPLSC` method analyzes the covariance matrix of the observed variables, whereas the `LVPLSX` module is able to process raw data. In order to specify the input file an external editor is necessary. The input specification requires that the program parameters are defined at specific positions in the file. Results are reported in a plain text file. The program offers blindfolding and jackknifing as resampling methods in case raw data has been analyzed. When analyzing covariance/correlation matrices, resampling techniques cannot be applied.

A comparison of PLS Software available in August 2006 is provided by Temme, Kreis, and Hildebrandt (2010): **LVPLS**, **VisualPLS** (Fu 2006), **PLS-Graph** (Chin 2003), **SPAD** (Test&Go 2006) and **SmartPLS**. **XLSTAT-PLSPM** and the **plspm** package were released later. For users who want a graphical user interface (GUI), **SmartPLS** or **XLSTAT-PLSPM** may be convenient choices. **SmartPLS** can be obtained free of charge whereas **XLSTAT-PLSPM** is distributed commercially. Concerning the open-source implementations **semPLS** and **plspm**, there may not exist a specific reason for many users to prefer one over the other, though the modular design of **semPLS** makes it more flexible and easier to extend. In general it is of benefit to have independent open-source implementations, e.g., for benchmarking.

The remainder of this paper is organized as follows: In Section 2 we sketch the theoretical background of PLS path modeling exemplary for the ECSI model (Tenenhaus, Esposito Vinzi, Chatelin, and Lauro 2005), a customer satisfaction index for the mobile phone industry. The basic usage of `plsm` and `sempls` is illustrated in Section 3. Section 4 explains how to get bootstrap confidence intervals for the model parameters and how to visualize bootstrapped parameters. In Section 5 other topics such as manipulation of the model specification, export for fitting with `sem` and importing of model specifications from **SmartPLS** are addressed. Finally, we close with a summary and outlook in Section 6.

2. Theoretical background: PLS path models in a nutshell

PLS path models consists of three components: the structural model, the measurement model and the weighting scheme. Whereas structural and measurement model are components in all kinds of SEMs with latent constructs, the weighting scheme is specific to the PLS approach. As in Tenenhaus *et al.* (2005) we introduce the theory by the example of European customer satisfaction index (ECSI) and the measurement instrument for the mobile phone industry. The description of the measurement instrument is available from the help page `help("ECSImobi")` in the **semPLS** package. In Figure 1 all relations between latent variables (LVs) and manifest variables (MVs), the nomological network, are shown. Nodes representing LVs are coded as ellipses and those representing MVs as boxes. Contrary to the CBSEM approach, in the PLS context each MV is only allowed to be connected to one LV. Furthermore all arrows connecting a LV with its block of MVs must point in the same direction. The connections between LVs and MVs is referred to as measurement or outer model. A model with all arrows pointing outwards is called a Mode A model – all LVs have reflective measurements. A model with all arrows pointing inwards is called a Mode B model – all LVs have formative measurements. A model containing both, formative and reflective LVs is referred to as MIMIC or a mode C model.

PLS path models only permit recursive relationships and can be expressed as simple connected

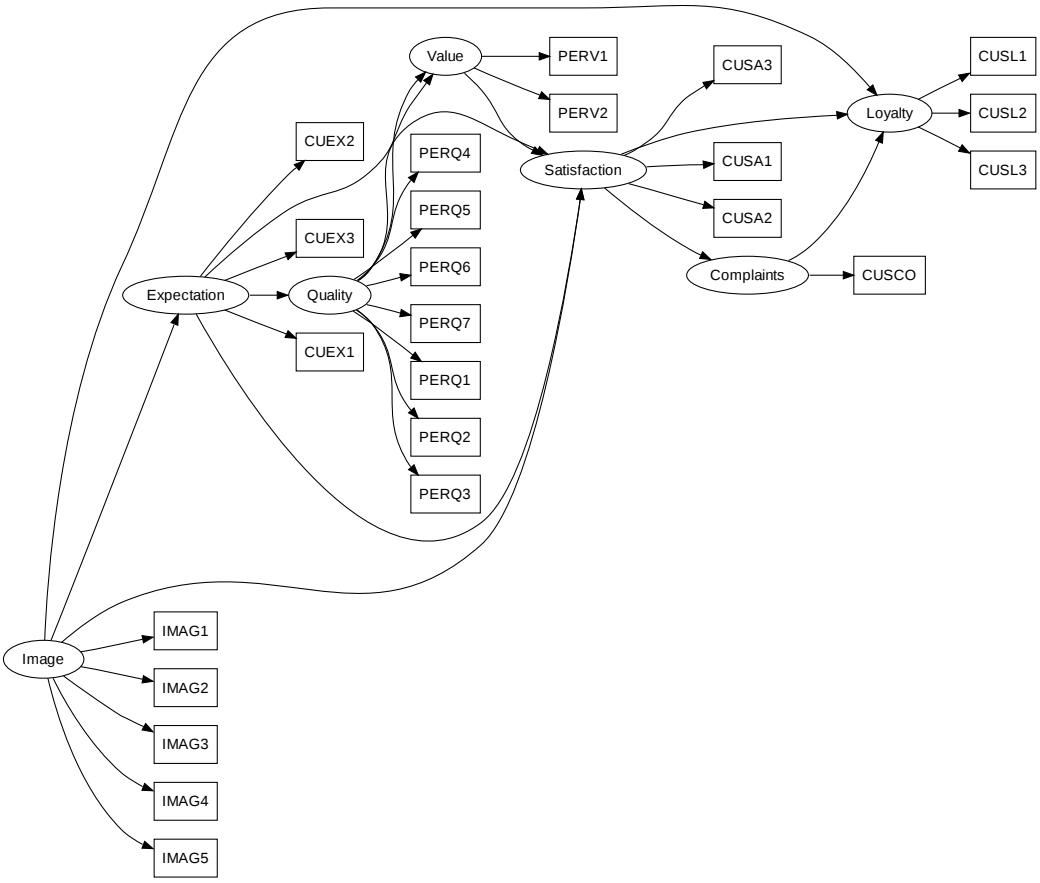

Figure 1: The graph represents the nomological network of the ECSI model for mobile phone provider (Tenenhaus *et al.* 2005). LVs are displayed in ellipses and MVs are displayed in boxes.

digraphs. A digraph is called simple if it has no loops and at most one arc between any pair of nodes. A digraph is connected if an undirected path between any two nodes exits; consequently no node is isolated from the rest.

2.1. The structural model

In the *structural model*, also called inner model, the LVs are related with each other according to substantive theory. LVs are divided into two classes, exogenous and endogenous. Exogenous LVs do not have any predecessor in the structural model, all others are endogenous. The structural model for the ECSI model is depicted by Figure 2. The only exogenous LV in the ECSI model is Image. The graph can be described by an adjacency matrix $\boldsymbol{D}$ as displayed in Table 1.

For the benefit simplicity the notation we use for the structural model dismisses the difference between exogenous and endogenous variables and we start with the compact form

$$\boldsymbol{Y} \quad = \quad \boldsymbol{Y}\boldsymbol{B} + \boldsymbol{Z} \tag{1}$$

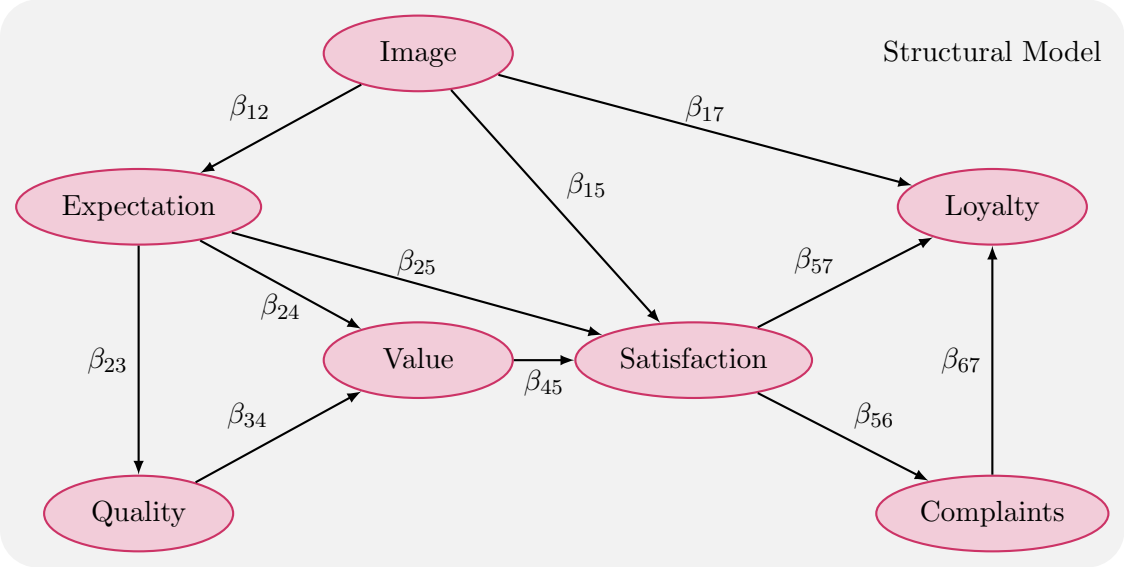

Figure 2: Causality model describing causes and consequences of customer satisfaction.

	Image	Expectation	Quality	Value	Satisfaction	Complaints	Loyalty
Image	**1**	**1**	0	0	**1**	0	**1**
Expectation	0	0	**1**	**1**	**1**	0	0
Quality	0	0	0	**1**	**1**	0	0
Value	0	0	0	0	**1**	0	0
Satisfaction	0	0	0	0	0	**1**	**1**
Complaints	0	0	0	0	0	0	**1**
Loyalty	0	0	0	0	0	0	0

Table 1: The table displays the adjacency matrix $\boldsymbol{D}$ for the ECSI model. If the entry $d_{ij} = 1$ the LV i is a predecessor of LV j. The matrix $\boldsymbol{D}$ can always be structured as a triangular matrix.

where $\boldsymbol{Y}$ denotes the matrix for the latent variables, both exogenous and endogenous. The error terms $\boldsymbol{Z}$ are assumed to be centred, i.e., $\mathsf{E}[\boldsymbol{Z}] = \boldsymbol{0}$. Elements of the coefficients matrix $\boldsymbol{B}$ are restricted to zero where the elements of the adjacency matrix $\boldsymbol{D}$ are zero.

Formally we can write the equations for the ECSI model as follows:

$$
\begin{aligned}
\text{Image} &= \text{Image} + 0 \quad \textit{(Note: exogenous variable)} \\
\text{Expectation} &= \beta_{12}\,\text{Image} + \boldsymbol{z}_2 \\
\text{Quality} &= \beta_{23}\,\text{Expectation} + \boldsymbol{z}_3 \\
\text{Value} &= \beta_{24}\,\text{Expectation} + \beta_{34}\,\text{Quality} + \boldsymbol{z}_4 \\
\text{Satisfaction} &= \beta_{15}\,\text{Image} + \beta_{25}\,\text{Expectation} + \beta_{35}\text{Quality} + \beta_{45}\,\text{Value} + \boldsymbol{z}_5 \\
\text{Complaints} &= \beta_{56}\,\text{Satisfaction} + \boldsymbol{z}_6 \\
\text{Loyalty} &= \beta_{17}\,\text{Image} + \beta_{57}\,\text{Satisfaction} + \beta_{67}\,\text{Complaints} + \boldsymbol{z}_7.
\end{aligned}
$$

2.2. The measurement model

The *measurement model* or outer model relates observed variables (MVs) to their latent variables (LVs). Often observed variables are referred to as manifest variables or indicators, latent variables as factors. Within the PLS framework one manifest variable can only be related to one LV. All manifest variables related to one LV are called a block. So each LV has its own block of observed variables. A block must contain at least one MV. The way a block can be related to an LV can be either reflective (see Figure 3) or formative (see Figure 4).

Without loss of generality we can make the following assumptions:

1. All MVs contained in the data matrix $\boldsymbol{X}$ are scaled to have zero mean and unit variance.
2. Each block of MVs $\boldsymbol{X}_g$ is already transformed to be positively correlated for all LVs $\boldsymbol{y}_g$, $g = 1, \ldots, G$.

As we will see, when the PLS algorithm (Section 2.3) is described, all the LVs values (factor scores) are constructed in a way to also have zero mean and unit variance. Table 4 in the appendix gives an overview of the notation used.

Reflective measurement: In the *reflective* way (Mode A) each block of MVs reflects its LV and can be written as the multivariate regression:

$$\boldsymbol{X}_g = \boldsymbol{y}_g \boldsymbol{w}_g^\top + \boldsymbol{F}_g, \qquad \mathsf{E}\left[\boldsymbol{F}_g | \boldsymbol{y}_g\right] = \boldsymbol{0}.$$

So $\boldsymbol{w}_g^\top$ can be estimated by least squares as

$$\begin{aligned} \hat{\boldsymbol{w}_g^\top} &= (\boldsymbol{y}_g^\top \boldsymbol{y}_g)^{-1} \boldsymbol{y}_g^\top \boldsymbol{X}_g \\ &= \mathsf{VAR}(\boldsymbol{y}_g)^{-1} \mathsf{COV}(\boldsymbol{y}_g, \boldsymbol{X}_g) \\ &= \mathsf{COV}(\boldsymbol{y}_g, \boldsymbol{X}_g) \\ &= \mathsf{COR}(\boldsymbol{y}_g, \boldsymbol{X}_g). \end{aligned} \tag{2}$$

Note, that the PLS algorithm (see Section 2.3) estimates all the LVs $\boldsymbol{y}_g$, $g = 1, \ldots, G$, as linear combination of their MVs under the constraint to have unit variance. At the beginning of this chapter we assumed all the MVs to be scaled to zero mean and unit variance. Consequently, the equality above is valid. Figure 3 depicts a path diagram for a reflectively measured LV.

Formative measurement: For the *formative* way (Mode B) the LV is considered to be formed by its MVs following a multiple regression:

$$\boldsymbol{y}_g = \boldsymbol{X}_g \boldsymbol{w}_g + \boldsymbol{\delta}_g \qquad , \ \mathsf{E}\left[\boldsymbol{\delta}_g | \boldsymbol{X}_g\right] = \boldsymbol{0}.$$

Again $\boldsymbol{w}_g$ is estimated by least squares:

$$\begin{aligned} \hat{\boldsymbol{w}_g} &= (\boldsymbol{X}_g^\top \boldsymbol{X}_g)^{-1} \boldsymbol{X}_g^\top \boldsymbol{y}_g \\ &= \mathsf{VAR}(\boldsymbol{X}_g)^{-1} \mathsf{COV}(\boldsymbol{X}_g, \boldsymbol{y}_g) \\ &= \mathsf{COR}(\boldsymbol{X}_g)^{-1} \mathsf{COR}(\boldsymbol{X}_g, \boldsymbol{y}_g). \end{aligned} \tag{3}$$

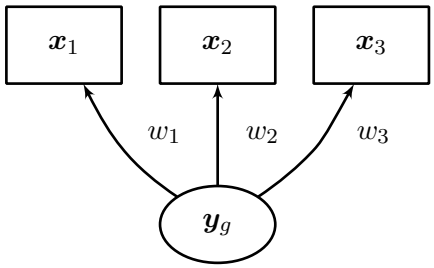

Figure 3: The latent variable $\boldsymbol{y}_g$ is measured by the block $\boldsymbol{X}_g$ consisting of three observed variables, $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_3$, in a reflective way (mode A).

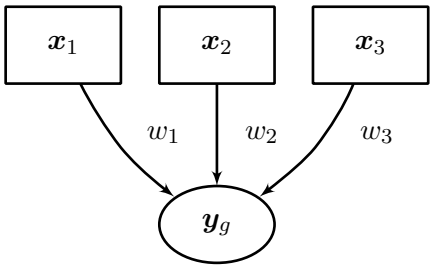

Figure 4: The latent variable $\boldsymbol{y}_g$ is measured by the block $\boldsymbol{X}_g$ consisting of three observed variables, $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_3$, in a formative way (mode B).

As for the reflective measurement, the equality results from the scaling of LVs and MVs. Let us keep in mind, that $\boldsymbol{X}_g$ is a matrix, when the LV $\boldsymbol{y}_g$ is measured by a block of more than one MV. In that case $\mathsf{VAR}(\boldsymbol{X}_g)$ refers to covariance matrix. Figure 4 depicts a path diagram for a formatively measured LV. E.g., Diamantopoulos and Winklhofer (2001) discuss formative constructs in detail.

When all latents in a model are measured reflectively, it is called a reflective model. If all of them are measured formatively, the model is formative. A mixture of both measurement modes is referred to as MIMIC (Tenenhaus *et al.* 2005) or multi-block model (Chin 1998).

Let $\kappa_g = \{k \in \{1, \ldots, K\} \mid \boldsymbol{x}_k \sim \boldsymbol{y}_g\}$ be a set of indices for MVs related to LV $\boldsymbol{y}_g$ then $\boldsymbol{w}_g$, $g = 1, \ldots, g$, is a column vector of length $|\kappa_g|$. We can write down the matrix of outer weights $\boldsymbol{W}$ as

$$\boldsymbol{W} = \begin{pmatrix} \boldsymbol{w}_1 & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \boldsymbol{w}_2 & \mathbf{0} & \ddots & \vdots \\ \mathbf{0} & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \boldsymbol{w}_G \end{pmatrix}.$$

Table 2 depicts the adjacency matrix $\boldsymbol{M}$ for the ECSI model. It has the same structure as the matrix of outer weights $\boldsymbol{W}$ and it is used for the initialization, as we will see, when the PLS algorithm (Section 2.3) is described. If the entry $m_{kg} = 1$, MV $\boldsymbol{x}_k$ is one of indicators of LV $\boldsymbol{y}_g$. The MVs `CUEX1` , `CUEX2` and `CUEX3` for example are indicators of the LV `Expectation`. Note, that the matrix $\boldsymbol{M}$ includes no information about the direction. So it does not tell us anything about the measurement mode of the blocks.

2.3. The partial least squares (PLS) algorithm

Now let us have a look at the partial least squares (PLS) algorithm (Wold 1982; Lohmöller 1989). The PLS algorithm aims at estimating the values for LVs (factor scores) by an iterative procedure. Figure 5 depicts the flowchart of the algorithm. The idea is to first construct each LV by the sum of its MVs. Then in the inner approximation we try to reconstruct each LV by means of its neighbouring LVs. In the outer approximation we try to find the best linear

	Image	Expectation	Quality	Value	Satisfaction	Complaints	Loyalty
IMAG1	**1**	0	0	0	0	0	0
IMAG2	**1**	0	0	0	0	0	0
IMAG3	**1**	0	0	0	0	0	0
IMAG4	**1**	0	0	0	0	0	0
IMAG5	**1**	0	0	0	0	0	0
CUEX1	0	**1**	0	0	0	0	0
CUEX2	0	**1**	0	0	0	0	0
CUEX3	0	**1**	0	0	0	0	0
PERQ1	0	0	**1**	0	0	0	0
PERQ2	0	0	**1**	0	0	0	0
PERQ3	0	0	**1**	0	0	0	0
PERQ4	0	0	**1**	0	0	0	0
PERQ5	0	0	**1**	0	0	0	0
PERQ6	0	0	**1**	0	0	0	0
PERQ7	0	0	**1**	0	0	0	0
PERV1	0	0	0	**1**	0	0	0
PERV2	0	0	0	**1**	0	0	0
CUSA1	0	0	0	0	**1**	0	0
CUSA2	0	0	0	0	**1**	0	0
CUSA3	0	0	0	0	**1**	0	0
CUSCO	0	0	0	0	0	**1**	0
CUSL1	0	0	0	0	0	0	**1**
CUSL2	0	0	0	0	0	0	**1**
CUSL3	0	0	0	0	0	0	**1**

Table 2: The table shows the adjacency matrix $\boldsymbol{M}$ for the measurement model. If the entry $m_{kg} = 1$ the MV k is one of the indicators of the LV g. The zeros are shaded out to better perceive the block structure.

combination to express each LV by means of its MVs; the coefficients are referred to as outer weights. Finally, in step 4, each LV is constructed as weighted sum or linear combination of its MVs. After each step the LVs are scaled to have zero mean and unit variance. The algorithm stops if the relative change for all the outer weights is smaller than a predefined tolerance.

Step 1 (initialization): We are constructing each LV as a weighted sum of their MVs. Remember that we are assuming all the MVs, $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_K$, to be scaled ($\text{mean}(\boldsymbol{X}_i) = 0$ and $\mathsf{VAR}(\boldsymbol{X}_i) = 1$). We have already seen the matrix $\boldsymbol{M}$ for the ECSI model (Table 2).

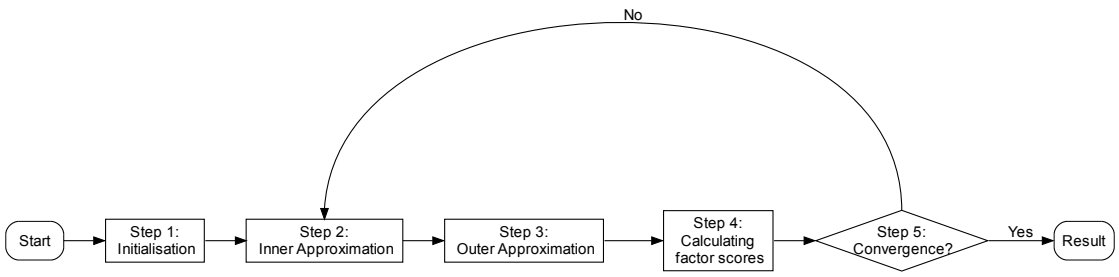

Figure 5: The diagram depicts the flowchart for the PLS algorithm.

In the initialization all the weights equal one. As a sum of centred variables all the LVs are also centred (mean = 0). But we still have to scale them to have unit variance (var = 1).

$$\hat{\boldsymbol{Y}} = \boldsymbol{X}\boldsymbol{M} \tag{4}$$

$$\hat{\boldsymbol{y}}_g = \frac{\hat{\boldsymbol{y}}_g}{\sqrt{\mathsf{VAR}(\hat{\boldsymbol{y}}_g)}}, \qquad g = 1, \ldots, G \tag{5}$$

The LVs are initialized as: $\hat{\boldsymbol{Y}} = (\hat{\boldsymbol{y}}_1, \ldots, \hat{\boldsymbol{y}}_G)$.

Step 2 (inner approximation): In the inner approximation we estimate each LV as a weighted sum of its neighbouring LVs. The weighting depends on the used scheme (see Section 2.3.1). Again we are scaling the recomputed LVs to have unit variance.

$$\tilde{\boldsymbol{Y}} = \hat{\boldsymbol{Y}}\boldsymbol{E}$$

$$\tilde{\boldsymbol{y}}_g = \frac{\tilde{\boldsymbol{y}}_g}{\sqrt{\mathsf{VAR}(\tilde{\boldsymbol{y}}_g)}}, \qquad g = 1, \ldots, G \tag{6}$$

We obtain the inner estimation: $\tilde{\boldsymbol{Y}} = (\tilde{\boldsymbol{y}}_1, \ldots, \tilde{\boldsymbol{y}}_G)$.

Step 3 (outer approximation): For the initialization all weights were one, now we are recalculating the weights on the basis of the LV values from the inner approximation (Step 2). According to the measurement mode (see Section 2.2) of the LV in focus, the weights can be estimated as,

Mode A a multivariate regression coefficient with the block of MVs as response and the LV as regressor:

$$\begin{aligned} \hat{\boldsymbol{w}}_g^\top &= (\tilde{\boldsymbol{y}}_g^\top \tilde{\boldsymbol{y}}_g)^{-1} \tilde{\boldsymbol{y}}_g^\top \boldsymbol{X}_g \\ &= \mathsf{COR}(\tilde{\boldsymbol{y}}_g, \boldsymbol{X}_g), \end{aligned} \tag{7}$$

Mode B or a multiple regression coefficient with the LV as response and its block of MVs as regressors:

$$\begin{aligned} \hat{\boldsymbol{w}}_g &= (\boldsymbol{X}_g^\top \boldsymbol{X}_g)^{-1} \boldsymbol{X}_g^t \tilde{\boldsymbol{y}}_g \\ &= \mathsf{VAR}(\boldsymbol{X}_g)^{-1} \mathsf{COR}(\boldsymbol{X}_g, \tilde{\boldsymbol{y}}_g) \end{aligned} \tag{8}$$

Step 4: In Section 2.2 we have seen how to arrange the outer weights vectors, $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_G$, in an outer weights matrix $\boldsymbol{W}$, which we are using now to estimate the factor scores by means of the MVs:

$$\hat{\boldsymbol{Y}} = \boldsymbol{X}\boldsymbol{W} \tag{9}$$

$$\hat{\boldsymbol{Y}}_g = \frac{\hat{\boldsymbol{Y}}_g}{\sqrt{\mathsf{VAR}(\hat{\boldsymbol{Y}}_g)}}, \qquad g = 1, \ldots, G, \tag{10}$$

resulting in the outer estimation: $\hat{\boldsymbol{Y}} = (\hat{\boldsymbol{y}}_1, \ldots, \hat{\boldsymbol{y}}_G)$.

Step 5: If the relative change of all the outer weights from one iteration to the next are smaller than a predefined tolerance,

$$\left| \frac{\hat{w}_{kg}^{old} - \hat{w}_{kg}^{new}}{\hat{w}_{kg}^{new}} \right| \; < \; \text{tolerance} \qquad \forall \; k = 1, \ldots, K \;\; \wedge \;\; g = 1, \ldots, G, \tag{11}$$

the estimation of factor scores done in Step 4 is taken to be final. Otherwise go back to Step 2.

Weighting schemes

The weighting scheme is used for the estimation of the inner weights in Step 2 (2.3) of the PLS algorithm. Originally Wold (1982) proposed the centroid weighting scheme. Later Lohmöller (1989) introduced two other schemes, factorial and path weighting. Table 1 shows the adjacency matrix $\boldsymbol{D}$ for the LVs in the ECSI model. This matrix is representing the structural part of the model we have already seen in Figure 2. Contrary to the matrix $\boldsymbol{M}$ for the measurement model, $\boldsymbol{D}$ accounts for the directionality. For every $d_{ij} = 1$, there is an arc from node i, the head of the arc, to node j, the tail of the arc. We could also say, the columns indicate the successor, whereas the rows indicate the predecessors. As we will see, the adjacency matrix $\boldsymbol{D}$ facilitates the calculation of the inner weights. For all the weighting schemes, each LV is constructed as a weighted sum of the LVs it is *related* with. The weighting schemes differ in the way the *relation* is defined. Generally we can express the inner estimate $\tilde{\boldsymbol{Y}}$ as matrix product of the outer estimate $\hat{\boldsymbol{Y}}$ and the matrix of inner weights $\boldsymbol{E}$:

$$\tilde{\boldsymbol{Y}} \;\; = \;\; \hat{\boldsymbol{Y}} \boldsymbol{E}. \tag{12}$$

Furthermore let us denote $\boldsymbol{R} = \mathsf{COR}(\hat{\boldsymbol{Y}})$, the empirical correlation matrix for the LVs resulting from the outer estimation, and $\boldsymbol{C} = \boldsymbol{D} + \boldsymbol{D}^\top$ a symmetrical matrix indicating whether two LVs are neighbours.

Centroid weighting scheme: Following the centroid weighting scheme, the matrix of inner weights $\boldsymbol{E}$ takes the form

$$e_{ij} \;\; = \;\; \begin{cases} \text{sign}\,(r_{ij}) & \text{, for } c_{ij} = 1, \\ 0 & \text{, else} \end{cases} \qquad , \; i, j = 1, \ldots, G. \tag{13}$$

Factorial weighting scheme: The factorial weighting scheme,

$$e_{ij} \;\; = \;\; \begin{cases} r_{ij} & \text{, for } c_{ij} = 1, \\ 0 & \text{, else} \end{cases} \qquad , \; i, j = 1, \ldots, G, \tag{14}$$

is quite similar to the centroid weighting scheme, except for the sign of the correlation between two neighbouring LVs, the correlation is used directly. This might be quite reasonable, when there are pairs of neighbouring LVs with correlations close to zero.

Path weighting scheme: For the path weighting scheme (or structural scheme) the predecessors and successor of a LV play a different role in the *relation*. Let us define the out-neighbourhood, or successor set of a node i as the set of tails of arcs going from i. Likewise, an in-neighbourhood, or predecessor set of a node i is the set of heads of

arcs going into i. A head is representing the start/initial node of an arch, a tail its end/terminal node.

The *relation* for one specific LV $\boldsymbol{y}_i$ with its successor is determined by their correlation, for the predecessors it is determined by a multiple regression

$$\boldsymbol{y}_i = \boldsymbol{y}_i^{pred}\boldsymbol{\gamma} + \boldsymbol{z}_i \qquad , \mathsf{E}\left[\boldsymbol{z}_i\right] = 0, \quad i = 1, \ldots, G$$

with $\boldsymbol{y}_i^{pred}$ the predecessor set of the LV $\boldsymbol{y}_i$. Denoting $\boldsymbol{y}_i^{succ}$ the successor set of the LV $\boldsymbol{y}_i$ the elements of the inner weight matrix $\boldsymbol{E}$ are

$$e_{ij} = \begin{cases} \gamma_j & , \text{for } j \in \boldsymbol{y}_i^{pred}, \\ \mathsf{COR}(\boldsymbol{y}_i, \boldsymbol{y}_j) & , \text{for } j \in \boldsymbol{y}_i^{succ}, \\ 0 & , \text{else.} \end{cases} \tag{15}$$

2.4. Calculation of path coefficients, total effects and loadings

Once the factor scores are estimated by PLS algorithm, the *path coefficients* can be estimated by ordinary least squares (OLS), according to the structural model (Section 2.1). For each LV $\hat{\boldsymbol{y}}_g$, $g = 1, \ldots, G$, the path coefficient is the regression coefficient on its predecessor set $\hat{\boldsymbol{y}}_g^{pred}$:

$$\begin{aligned} \hat{\boldsymbol{\beta}}_g &= (\hat{\boldsymbol{y}}_g^{pred\,\top}\hat{\boldsymbol{y}}_g^{pred})^{-1}\hat{\boldsymbol{y}}_g^{pred\,\top}\hat{\boldsymbol{y}}_g \\ &= \mathsf{COR}(\hat{\boldsymbol{y}}_g^{pred}, \hat{\boldsymbol{y}}_g^{pred})^{-1}\mathsf{COR}(\hat{\boldsymbol{y}}_g^{pred}, \hat{\boldsymbol{y}}_g) \end{aligned} \tag{16}$$

We obtain the elements $\hat{b}_{ij}$, $i, j = 1, \ldots, G$, of the estimated matrix of path coefficients $\hat{\boldsymbol{B}}$:

$$\hat{\beta}_{ij} = \begin{cases} \hat{\boldsymbol{\beta}}_{g_j} & , \text{for } j \in \boldsymbol{y}_i^{pred}, \\ 0 & , \text{else.} \end{cases} \tag{17}$$

The matrix $\hat{\boldsymbol{B}}$ can be interpreted as transition matrix for the structural model. We can calculate the matrix of *total effects* $\hat{\boldsymbol{T}}$ as the sum of the 1 to G step transition matrices:

$$\hat{\boldsymbol{T}} = \sum_{g=1}^{G} \hat{\boldsymbol{B}}^g. \tag{18}$$

Note, that $\hat{\boldsymbol{B}}^g$ expands to $\overbrace{\hat{\boldsymbol{B}} \cdot \hat{\boldsymbol{B}} \cdot \ldots \cdot \hat{\boldsymbol{B}}}^{g\text{-times}}$, e.g., $\hat{\boldsymbol{B}}^2$ contains all the indirect effects mediated by only one LV. The *cross* and *outer loadings* are estimated as:

$$\hat{\boldsymbol{\Lambda}}^{cross} = \mathsf{COR}(\boldsymbol{X}, \hat{\boldsymbol{Y}}) \tag{19}$$

$$\hat{\boldsymbol{\lambda}}_{kg}^{outer} = \begin{cases} \hat{\boldsymbol{\lambda}}_{kg}^{cross} & , \text{if } m_{kg} = 1, \\ 0 & , \text{else.} \end{cases} \tag{20}$$

Remember, $\boldsymbol{M}$ is the adjacency matrix for the measurement model. Table 2 shows the respective matrix for the ECSI model.

3. Getting started: how to fit a model with `sempls()`

For illustration, we continue with the ECSI model introduced in the previous Section. The first step, of course, is to attach the **semPLS** package.

```
R> library("semPLS")
```

Starting from scratch we have to create two so-called from-to-matrices that are used for constructing the adjacency matrix $\boldsymbol{D}$ of the structural model, the other for the adjacency matrix $\boldsymbol{M}$ of the measurement model. A from-to-matrix is a two column matrix with each row representing a directed edge in a graph. The first column of a row contains the name of the node where the tail of an arrow starts, the second must contain the name of the node where the head of the arrow is connected. For the ECSI model the according matrices are already pre-built, so we just have to load them. The matrices `ECSIsm` and `ECSImm` represent structural and measurement model.

```
R> data("ECSIsm")
R> ECSIsm
```

```
      source         target
 [1,] "Image"        "Expectation"
 [2,] "Expectation"  "Quality"
 [3,] "Expectation"  "Value"
 [4,] "Quality"      "Value"
 [5,] "Image"        "Satisfaction"
 [6,] "Expectation"  "Satisfaction"
 [7,] "Quality"      "Satisfaction"
 [8,] "Value"        "Satisfaction"
 [9,] "Satisfaction" "Complaints"
[10,] "Image"        "Loyalty"
[11,] "Satisfaction" "Loyalty"
[12,] "Complaints"   "Loyalty"
```

```
R> data("ECSImm")
R> ECSImm
```

```
      source         target
 [1,] "Image"        "IMAG1"
 [2,] "Image"        "IMAG2"
 [3,] "Image"        "IMAG3"
 [4,] "Image"        "IMAG4"
 [5,] "Image"        "IMAG5"
 [6,] "Expectation"  "CUEX1"
 [7,] "Expectation"  "CUEX2"
 [8,] "Expectation"  "CUEX3"
 [9,] "Quality"      "PERQ1"
[10,] "Quality"      "PERQ2"
```

```
[11,] "Quality"      "PERQ3"
[12,] "Quality"      "PERQ4"
[13,] "Quality"      "PERQ5"
[14,] "Quality"      "PERQ6"
[15,] "Quality"      "PERQ7"
[16,] "Value"        "PERV1"
[17,] "Value"        "PERV2"
[18,] "Satisfaction" "CUSA1"
[19,] "Satisfaction" "CUSA2"
[20,] "Satisfaction" "CUSA3"
[21,] "Complaints"   "CUSCO"
[22,] "Loyalty"      "CUSL1"
[23,] "Loyalty"      "CUSL2"
[24,] "Loyalty"      "CUSL3"
```

As mentioned before, all LVs of the ECSI model are measured reflectively, thus the MVs of a block are all found in the second column. In a graph arrows would be drawn from `Expectation` to `CUEX1`, `CUEX2` and `CUEX3`, etc..

```
R> ECSImm[ECSImm[, 1] == "Expectation", ]
```

```
     source        target
[1,] "Expectation" "CUEX1"
[2,] "Expectation" "CUEX2"
[3,] "Expectation" "CUEX3"
```

If Expectation would have formative measurements, first and second column of the matrix must be swapped.

```
R> ECSImm[ECSImm[, 1] == "Expectation", 2:1]
```

```
     target  source
[1,] "CUEX1" "Expectation"
[2,] "CUEX2" "Expectation"
[3,] "CUEX3" "Expectation"
```

The last prerequisite we need before we can finally setup our model is a dataset containing the MVs. In our example we use the `mobi` dataset which is included in the package.

```
R> data("mobi")
```

Now we use `plsm` function to create an object suited for use with the fitting function `sempls`. The method needs the arguments

`data` the name of the dataset containing the observed variables,

`strucmod` a from-to-matrix representing the structural model and

`measuremod` a from-to-matrix representing the measurement model.

Matrices as shown above can be created by `matrix()`. For convenience one can use a spreadsheet to quickly enter the from-to-matrices by setting `interactive=TRUE`. For reproducibility reasons the corresponding R expression is printed and should be saved. Alternatively csv-files can be used to specify structural and measurement models, see the example section in `help("plsm")`. Furthermore models already specified in SmartPLS (Ringle *et al.* 2005), see Section 5, can be imported.

```
R> ECSI <- plsm(data = mobi, strucmod = ECSIsm, measuremod = ECSImm)
```

Objects of class `plsm` provide a structure such that the block structure of the data can be reflected, see Figure 6.

```
R> mvpairs(model = ECSI, data = mobi, LVs = "Expectation")
```

Once the model is setup by `plsm`, model parameters can be estimated by the `sempls` function. By specifying the argument `wscheme="centroid"`, the centroid weighting scheme is used for the inner estimation, for other weighting schemes consult the help page of `sempls`. The `print` method for `sempls` objects assures that only the estimates of special interest, path coefficients and loadings (weights in case of formative measures), are printed. Additional values of the `sempls` object can be accessed either explicitly or with specific getter methods.

```
R> ecsi <- sempls(model = ECSI, data = mobi, wscheme = "centroid")

All 250 observations are valid.
Converged after 6 iterations.
Tolerance: 1e-07
Scheme: centroid
```

For graphical representation of the results, `pathDiagram()` creates a graph in the DOT Language (Gansner *et al.* 2006). If Graphviz (AT&T Research 2009) is available on the system, the DOT code can be directly rendered to a graphics format such as PDF (vector graphic) or PNG (bitmap) or various others. By specifying `edge.labels = "both"`, names of the parameters and values are both printed. By setting `full = FALSE`, only the structural model is processed. As Graphviz uses internal rendering algorithms for the layout of the graph, this function is especially useful for models with a large number variables. Note, that for people unfamiliar with Graphviz some aspects of the resulting path diagram may be hard to change.

```
R> pathDiagram(ecsi, file = "ecsi-structure", full = FALSE, edge.labels = "both",
+      output.type = "graphics", digits = 2, graphics.fmt = "pdf")
```

Figure 7 depicts the resulting PDF file.

```
R> ecsi

                                Path Estimate
lam_1_1               Image -> IMAG1    0.743
```

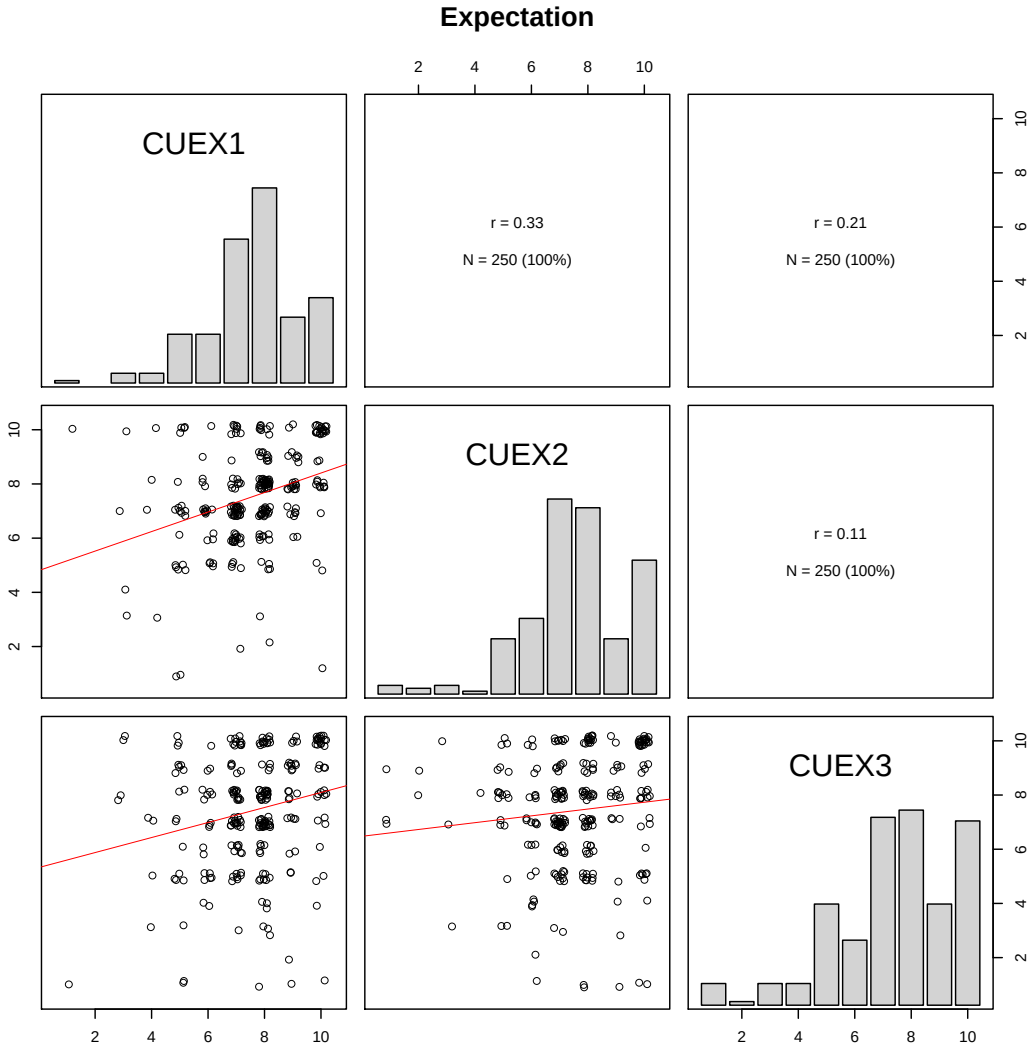

Figure 6: The pairs plot for LV Expectation reveals association structures for its block of MVs, CUEX1, CUEX2 and CUEX3. In the lower triagonal the scatterplots of the jittered observations including a linear regression line are plotted against each other. The diagonal elements contain univariate barcharts of the MVs. The upper triagonal shows pairwise Bravais-Pearson correlation coefficients and the percentage of pairwise complete observations.

```
lam_1_2             Image -> IMAG2    0.601
lam_1_3             Image -> IMAG3    0.578
lam_1_4             Image -> IMAG4    0.768
lam_1_5             Image -> IMAG5    0.744
lam_2_1       Expectation -> CUEX1    0.771
lam_2_2       Expectation -> CUEX2    0.687
lam_2_3       Expectation -> CUEX3    0.612
lam_3_1           Quality -> PERQ1    0.803
```

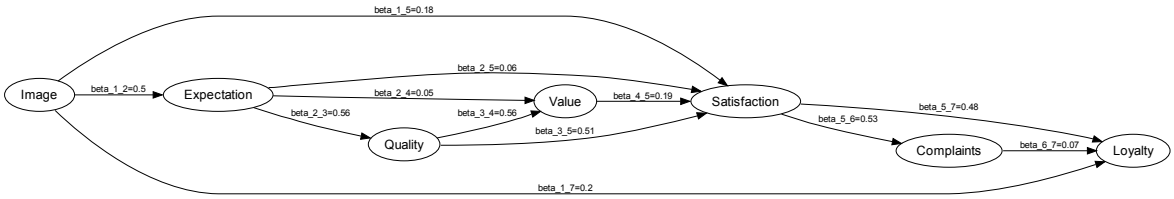

Figure 7: A path diagram for the structural part of the fitted ECSI model can be created with pathDiagram().

```
lam_3_2             Quality -> PERQ2    0.637
lam_3_3             Quality -> PERQ3    0.784
lam_3_4             Quality -> PERQ4    0.769
lam_3_5             Quality -> PERQ5    0.756
lam_3_6             Quality -> PERQ6    0.775
lam_3_7             Quality -> PERQ7    0.779
lam_4_1               Value -> PERV1    0.904
lam_4_2               Value -> PERV2    0.938
lam_5_1        Satisfaction -> CUSA1    0.799
lam_5_2        Satisfaction -> CUSA2    0.846
lam_5_3        Satisfaction -> CUSA3    0.852
lam_6_1          Complaints -> CUSCO    1.000
lam_7_1             Loyalty -> CUSL1    0.814
lam_7_2             Loyalty -> CUSL2    0.219
lam_7_3             Loyalty -> CUSL3    0.917
beta_1_2         Image -> Expectation    0.505
beta_2_3      Expectation -> Quality    0.557
beta_2_4        Expectation -> Value    0.051
beta_3_4             Quality -> Value    0.557
beta_1_5       Image -> Satisfaction    0.179
beta_2_5 Expectation -> Satisfaction    0.064
beta_3_5     Quality -> Satisfaction    0.513
beta_4_5       Value -> Satisfaction    0.192
beta_5_6  Satisfaction -> Complaints    0.526
beta_1_7             Image -> Loyalty    0.195
beta_5_7     Satisfaction -> Loyalty    0.483
beta_6_7       Complaints -> Loyalty    0.071
```

Values returned by `sempls`:

```
R> names(ecsi)
```

```
 [1] "coefficients"      "path_coefficients" "outer_loadings"
 [4] "cross_loadings"    "total_effects"     "inner_weights"
 [7] "outer_weights"     "blocks"            "factor_scores"
[10] "data"              "scaled"            "model"
[13] "weighting_scheme"  "weights_evolution" "sum1"
```

Function	Model criteria
`rSquared()`	coefficients of determination, R^2 values, for each endogenous LV
`qSquared()`	Stone-Geisser's Q^2 for assessment of predictive relevance
`dgrho()`	Dillon-Goldstein's rho, also referred to as composite reliability
`communality()`	communality indices for reflectively measured LVs with more than one MV
`redundancy()`	redundancy indices for endogenous LVs
`gof()`	GoF index (geometric mean of average communality and average determination coefficient)

Table 3: A list of criteria for model validation which are already available in the semPLS package.

```
[16] "pairwise"          "method"            "iterations"
[19] "convCrit"          "verbose"           "tolerance"
[22] "maxit"             "N"                 "incomplete"
[25] "Hanafi"
```

Since there is no well identified global optimisation criterion for PLS path models, each part of the model needs to be validated. For this task several indices are known from literature, see e.g., Tenenhaus *et al.* (2005, p. 172–176) or Esposito Vinzi *et al.* (2010, p. 56–62). Table 3 lists the model criteria currently implemented in **semPLS**.

Path coefficients and total effects are extracted by `pathCoeff` and `totalEffects`. As we see in the example `dimnames` can be abbreviated.

```
R> pC <- pathCoeff(ecsi)
R> print(pC, abbreviate = TRUE, minlength = 3)

    Img   Exp   Qlt   Val   Sts   Cmp   Lyl
Img   . 0.505     .     . 0.179     . 0.195
Exp   .     . 0.557 0.051 0.064     .     .
Qlt   .     .     . 0.557 0.513     .     .
Val   .     .     .     . 0.192     .     .
Sts   .     .     .     .     . 0.526 0.483
Cmp   .     .     .     .     .     . 0.071
Lyl   .     .     .     .     .     .     .

R> tE <- totalEffects(ecsi)
R> print(tE, abbreviate = TRUE, minlength = 3)

    Img   Exp   Qlt   Val   Sts   Cmp   Lyl
Img   . 0.505 0.281 0.182 0.390 0.205 0.399
Exp   .     . 0.557 0.361 0.419 0.221 0.218
Qlt   .     .     . 0.557 0.619 0.326 0.323
Val   .     .     .     . 0.192 0.101 0.100
Sts   .     .     .     .     . 0.526 0.521
Cmp   .     .     .     .     .     . 0.071
Lyl   .     .     .     .     .     .     .
```

Outer weights are extracted by `plsWeights`.

```
R> plsWeights(ecsi)
```

```
      Image Expectation Quality Value Satisfaction Complaints Loyalty
IMAG1  0.30           .       .     .            .          .       .
IMAG2  0.26           .       .     .            .          .       .
IMAG3  0.22           .       .     .            .          .       .
IMAG4  0.33           .       .     .            .          .       .
IMAG5  0.32           .       .     .            .          .       .
CUEX1     .        0.52       .     .            .          .       .
CUEX2     .        0.47       .     .            .          .       .
CUEX3     .        0.45       .     .            .          .       .
PERQ1     .           .    0.21     .            .          .       .
PERQ2     .           .    0.14     .            .          .       .
PERQ3     .           .    0.20     .            .          .       .
PERQ4     .           .    0.18     .            .          .       .
PERQ5     .           .    0.18     .            .          .       .
PERQ6     .           .    0.18     .            .          .       .
PERQ7     .           .    0.21     .            .          .       .
PERV1     .           .       .  0.49            .          .       .
PERV2     .           .       .  0.60            .          .       .
CUSA1     .           .       .     .         0.38          .       .
CUSA2     .           .       .     .         0.38          .       .
CUSA3     .           .       .     .         0.44          .       .
CUSCO     .           .       .     .            .       1.00       .
CUSL1     .           .       .     .            .          .    0.45
CUSL2     .           .       .     .            .          .    0.13
CUSL3     .           .       .     .            .          .    0.66
```

Loadings are extracted by `plsLoadings`. Since loadings can be used to check for discriminant validity, the default for the `print` method of `plsLoadings` objects is to print numeric values only for the row maxima and loading relatively close to them. The MV `IMAG2` for example loads relatively high on the LVs `Image` and `Quality`. To print outer or cross loadings, the `print` method has to be called explicitly with its `type` argument specified. Another argument, `reldiff`, can be used to check for discriminant validity. The default is 0.2, which means that all crossloadings bigger than $(1 - 0.2)$ times the maximum crossloading for a MV are printed.

```
R> plsLoadings(ecsi)
```

```
      Image Expectation Quality Value Satisfaction Complaints Loyalty
IMAG1  0.74           .       .     .            .          .       .
IMAG2  0.60           .    0.50     .            .          .       .
IMAG3  0.58           .       .     .            .          .       .
IMAG4  0.77           .       .     .            .          .       .
IMAG5  0.74           .       .     .            .          .       .
CUEX1     .        0.77       .     .            .          .       .
```

```
CUEX2     .        0.69       .     .            .          .       .
CUEX3     .        0.61       .     .            .          .       .
PERQ1     .           .    0.80     .         0.68          .       .
PERQ2     .           .    0.64     .            .          .       .
PERQ3  0.63           .    0.78     .         0.64          .       .
PERQ4     .           .    0.77     .            .          .       .
PERQ5  0.61           .    0.76     .            .          .       .
PERQ6     .           .    0.78     .            .          .       .
PERQ7     .           .    0.78     .         0.70          .       .
PERV1     .           .       .  0.90            .          .       .
PERV2     .           .       .  0.94            .          .       .
CUSA1     .           .    0.64     .         0.80          .       .
CUSA2     .           .       .     .         0.85          .       .
CUSA3     .           .       .     .         0.85          .       .
CUSCO     .           .       .     .            .       1.00       .
CUSL1     .           .       .     .            .          .    0.81
CUSL2     .           .       .     .            .          .    0.22
CUSL3     .           .       .     .            .          .    0.92
```

By calling `plot` on a `sempls` object, a plot of the evolution of outer weights until convergence for all blocks of MVs is created. Figure 8 depicts the result of `plot(ecsi)`, using **lattice** (Sarkar 2008).

Kernel density estimates can provide hints on the adequacy of the model, see Figure 9.

```
R> densityplot(ecsi, use = "residuals")
```

4. Bootstrapping `sempls` objects

Finally we can bootstrap the estimations for outer loadings and path coefficients, leveraging the **boot** package (Canty and Ripley 2012; Davison and Hinkley 1997). The `summary` method also calculates confidence intervals based on the percentile method. We use 500 bootstrap samples and we use ones to initialise the outer weights. For the outer loading `lam_6_1` no confidence interval can be computed, because it relates the LV `Complaints` to its only MV `CUSCO`, which is always estimated as 1.

```
R> set.seed(123)
R> ecsiBoot <- bootsempls(ecsi, nboot = 500, start = "ones", verbose = FALSE)
R> ecsiBoot

Call: bootsempls(object = ecsi, nboot = 500, start = "ones", verbose = FALSE)

                          Estimate      Bias Std.Error
Image -> IMAG1              0.7434 -0.004463  4.31e-02
Image -> IMAG2              0.6007 -0.001859  5.80e-02
Image -> IMAG3              0.5776 -0.004403  6.31e-02
Image -> IMAG4              0.7684 -0.002188  4.47e-02
```

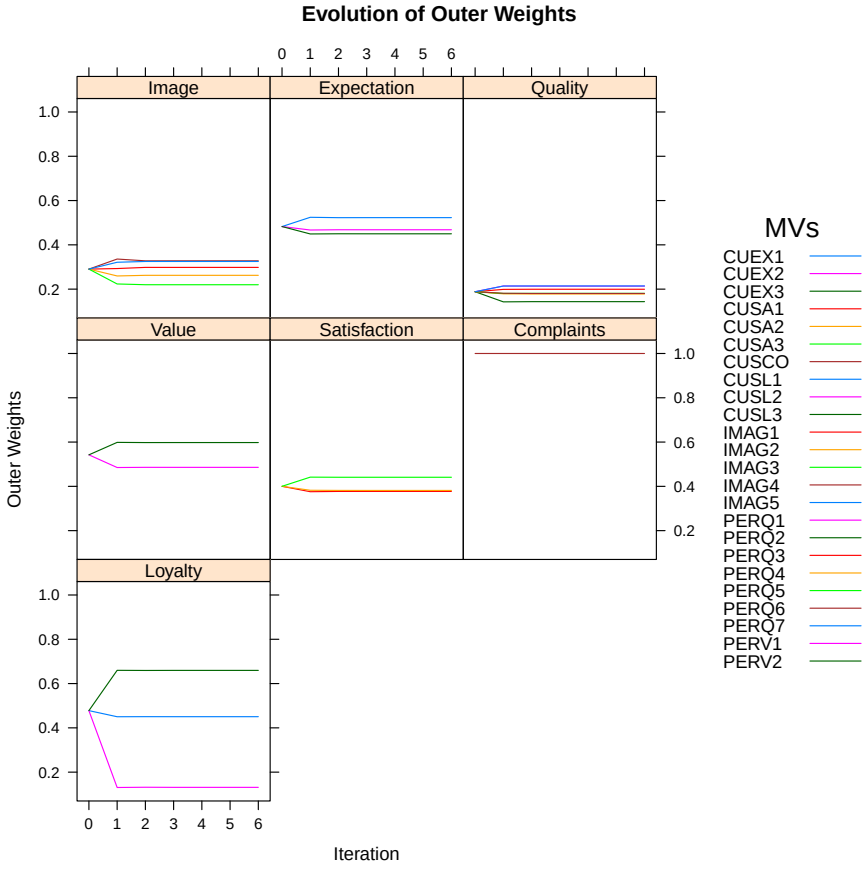

Figure 8: The plot depicts the evolution of outer weights until convergence.

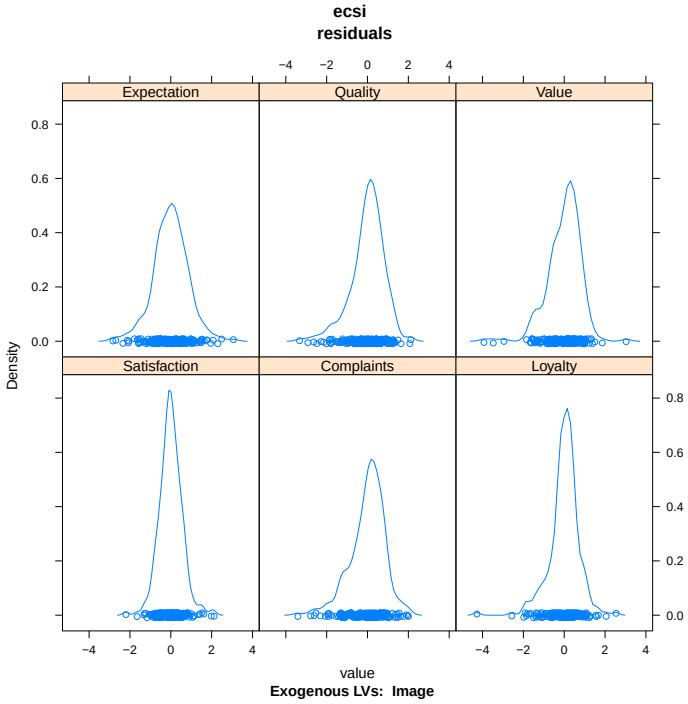

Figure 9: The plot depicts kernel density estimates of the residuals of estimated endogenous LVs.

```
Image -> IMAG5                0.7445  0.001691  3.01e-02
Expectation -> CUEX1          0.7715 -0.006157  5.23e-02
Expectation -> CUEX2          0.6866 -0.002187  8.52e-02
Expectation -> CUEX3          0.6118 -0.000381  7.59e-02
Quality -> PERQ1              0.8033  0.001853  2.34e-02
Quality -> PERQ2              0.6374 -0.004122  5.19e-02
Quality -> PERQ3              0.7835  0.001389  2.97e-02
Quality -> PERQ4              0.7691 -0.002988  4.52e-02
Quality -> PERQ5              0.7558 -0.001085  3.92e-02
Quality -> PERQ6              0.7752 -0.002527  5.75e-02
Quality -> PERQ7              0.7794  0.001324  3.12e-02
Value -> PERV1                0.9043 -0.002026  2.22e-02
Value -> PERV2                0.9379  0.000318  8.07e-03
Satisfaction -> CUSA1         0.7990 -0.001989  3.04e-02
Satisfaction -> CUSA2         0.8462 -0.001261  2.28e-02
Satisfaction -> CUSA3         0.8519 -0.000138  1.75e-02
Complaints -> CUSCO           1.0000  0.000000  5.17e-17
Loyalty -> CUSL1              0.8138 -0.002456  4.03e-02
Loyalty -> CUSL2              0.2191  0.001473  9.65e-02
Loyalty -> CUSL3              0.9168 -0.001029  1.19e-02
Image -> Expectation          0.5047  0.007683  5.87e-02
Expectation -> Quality        0.5572  0.004379  5.35e-02
Expectation -> Value          0.0508  0.011914  8.46e-02
Quality -> Value              0.5572 -0.007401  8.41e-02
Image -> Satisfaction         0.1788  0.008076  5.08e-02
Expectation -> Satisfaction   0.0644 -0.004120  4.83e-02
Quality -> Satisfaction       0.5125 -0.005184  6.49e-02
Value -> Satisfaction         0.1918  0.001316  5.87e-02
Satisfaction -> Complaints    0.5261  0.001322  5.20e-02
Image -> Loyalty              0.1954  0.006120  7.65e-02
Satisfaction -> Loyalty       0.4835 -0.004741  8.57e-02
Complaints -> Loyalty         0.0712  0.000369  5.55e-02
```

```
R> ecsiBootsummary <- summary(ecsiBoot, type = "bca", level = 0.9)
R> ecsiBootsummary

Call: bootsempls(object = ecsi, nboot = 500, start = "ones", verbose = FALSE)

Lower and upper limits are for the 90 percent bca confidence interval

         Estimate      Bias Std.Error    Lower Upper
lam_1_1    0.7434 -0.004463  4.31e-02  0.65788 0.799
lam_1_2    0.6007 -0.001859  5.80e-02  0.49794 0.689
lam_1_3    0.5776 -0.004403  6.31e-02  0.44906 0.660
lam_1_4    0.7684 -0.002188  4.47e-02  0.67570 0.825
lam_1_5    0.7445  0.001691  3.01e-02  0.68441 0.787
lam_2_1    0.7715 -0.006157  5.23e-02  0.65638 0.831
```

```
lam_2_2    0.6866 -0.002187  8.52e-02  0.51959 0.798
lam_2_3    0.6118 -0.000381  7.59e-02  0.44677 0.715
lam_3_1    0.8033  0.001853  2.34e-02  0.75399 0.838
lam_3_2    0.6374 -0.004122  5.19e-02  0.54288 0.714
lam_3_3    0.7835  0.001389  2.97e-02  0.72151 0.821
lam_3_4    0.7691 -0.002988  4.52e-02  0.68072 0.837
lam_3_5    0.7558 -0.001085  3.92e-02  0.67852 0.813
lam_3_6    0.7752 -0.002527  5.75e-02  0.65748 0.853
lam_3_7    0.7794  0.001324  3.12e-02  0.70761 0.817
lam_4_1    0.9043 -0.002026  2.22e-02  0.85289 0.930
lam_4_2    0.9379  0.000318  8.07e-03  0.92111 0.949
lam_5_1    0.7990 -0.001989  3.04e-02  0.74262 0.843
lam_5_2    0.8462 -0.001261  2.28e-02  0.80460 0.878
lam_5_3    0.8519 -0.000138  1.75e-02  0.81982 0.878
lam_6_1    1.0000  0.000000  5.17e-17         .     .
lam_7_1    0.8138 -0.002456  4.03e-02  0.72931 0.866
lam_7_2    0.2191  0.001473  9.65e-02  0.04211 0.365
lam_7_3    0.9168 -0.001029  1.19e-02  0.89681 0.934
beta_1_2   0.5047  0.007683  5.87e-02  0.38627 0.579
beta_2_3   0.5572  0.004379  5.35e-02  0.46267 0.638
beta_2_4   0.0508  0.011914  8.46e-02 -0.08127 0.192
beta_3_4   0.5572 -0.007401  8.41e-02  0.41834 0.692
beta_1_5   0.1788  0.008076  5.08e-02  0.09167 0.248
beta_2_5   0.0644 -0.004120  4.83e-02 -0.00649 0.152
beta_3_5   0.5125 -0.005184  6.49e-02  0.40264 0.615
beta_4_5   0.1918  0.001316  5.87e-02  0.09251 0.288
beta_5_6   0.5261  0.001322  5.20e-02  0.43602 0.612
beta_1_7   0.1954  0.006120  7.65e-02  0.04689 0.313
beta_5_7   0.4835 -0.004741  8.57e-02  0.35094 0.622
beta_6_7   0.0712  0.000369  5.55e-02 -0.01587 0.169
```

The results of the bootstrap samples for the path coefficients can be visualised by plotting kernel density estimates (Figure 10) and parallel coordinates (Figure 11).

```
R> densityplot(ecsiBoot, pattern = "beta")
R> parallelplot(ecsiBoot, pattern = "beta", reflinesAt = c(0, 0.5),
+         alpha = 0.3, type = "bca",
+         main = "Path Coefficients\nof 500 bootstrap samples")
```

5. The `plsm` class: importing, manipulating, exporting

5.1. Manipulating an existing model

Once we are working with a model, we might want to add or remove a path, bring in or take out variables, both LVs and MVs, or even to invert the measurement model from reflective to

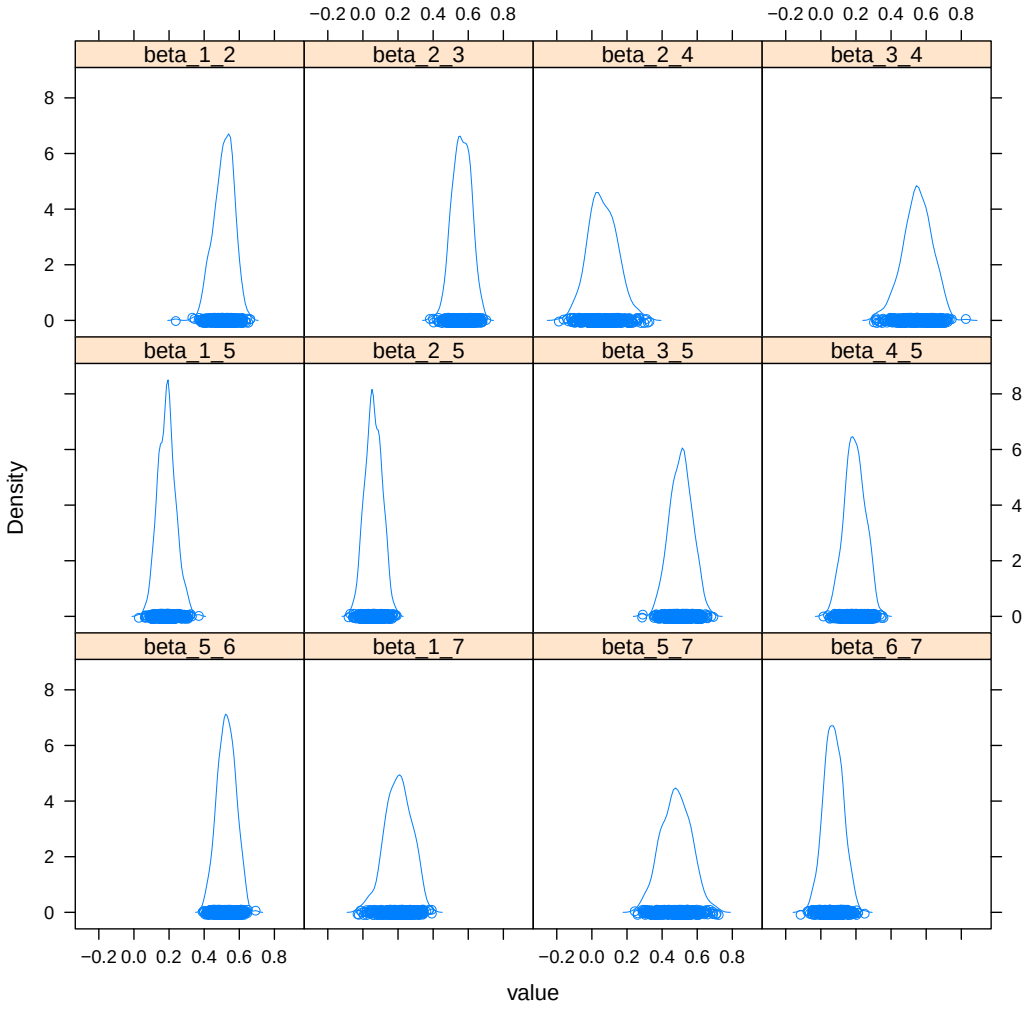

Figure 10: The figure depicts the bootstrap distribution of the path coefficients based on 500 resamples.

formative and vice versa. The **semPLS** package provides a list of methods to perform those tasks. All of them are found in `help("plsmUtils")`. With `plsmEdit` the from-to-matrices for structural and measurement model can be edited in a spreadsheet. When the spreadsheets are saved, the method checks whether the model is still valid. Valid means all MVs are available in the data, names of MVs and LVs are not allowed to coincide. All MVs of a block must be in the same column, this is because a block of MVs can either belong to a reflective or formative LV. And the structural model must be recursive – an acyclic graph.

To better understand models of class `plsm`, we will check for changes made by the utility methods in the respective elements of the `plsm` object. We continue with the `ECSI` model and invert the measurement model of the LV `Expectation`. This does not result in changes in the *adjacency* matrix M, `ECSI[["M"]]`, as it does not include the direction. The measurement model is coded in the element `ECSI[["blocks"]]`, a list with elements named by the LVs and character vectors naming the MVs as elements. Each Element has an attribute `"mode"` with

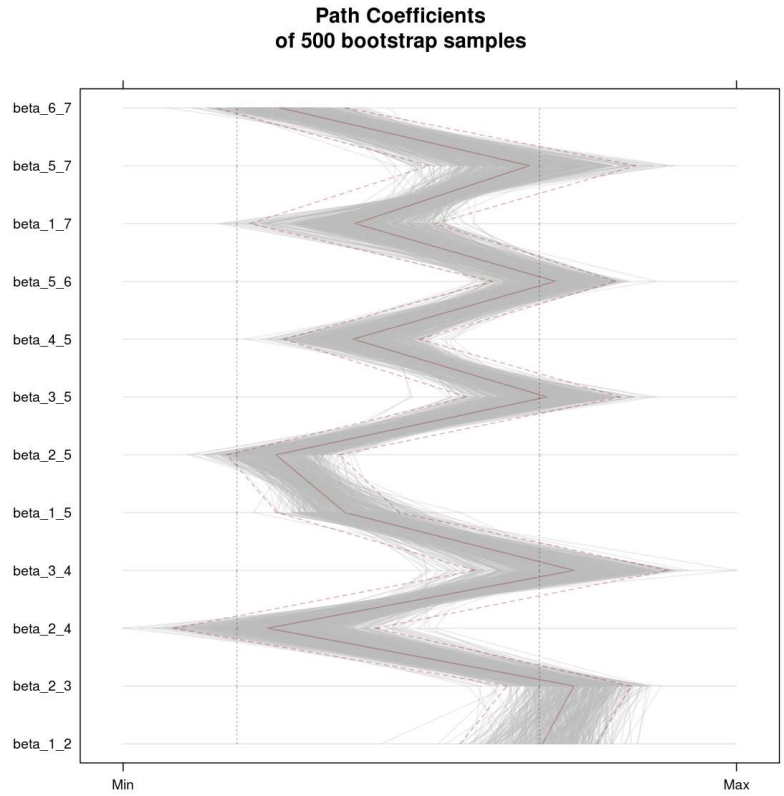

Figure 11: The figure depicts parallel coordinates for the path coefficients of 500 bootstrap samples (solid light-gray lines), the sample path coefficients (solid dark-red line), 90% bootstrap bca confidence intervals (dashed dark-red lines) and two reference lines at 0 and 0.5 (dotted black lines).

supported values `"A"`, reflective, and `"B"`, formative.

```
R> ECSI[["blocks"]]["Expectation"]

$Expectation
[1] "CUEX1" "CUEX2" "CUEX3"
attr(,"mode")
[1] "A"

R> invertLVs(model = ECSI, LVs = c("Expectation"))[["blocks"]]["Expectation"]

$Expectation
[1] "CUEX1" "CUEX2" "CUEX3"
attr(,"mode")
[1] "B"
```

Now we want to add a path from `Quality` to `Loyalty`.

```
R> ECSI[["D"]]
```

```
             Image Expectation Quality Value Satisfaction Complaints Loyalty
Image            0           1       0     0            1          0       1
Expectation      0           0       1     1            1          0       0
Quality          0           0       0     1            1          0       0
Value            0           0       0     0            1          0       0
Satisfaction     0           0       0     0            0          1       1
Complaints       0           0       0     0            0          0       1
Loyalty          0           0       0     0            0          0       0
```

```
R> addPath(model = ECSI, from = "Quality", to = "Loyalty")[["D"]]
```

```
             Image Expectation Quality Value Satisfaction Complaints Loyalty
Image            0           1       0     0            1          0       1
Expectation      0           0       1     1            1          0       0
Quality          0           0       0     1            1          0       1
Value            0           0       0     0            1          0       0
Satisfaction     0           0       0     0            0          1       1
Complaints       0           0       0     0            0          0       1
Loyalty          0           0       0     0            0          0       0
```

Two paths can be removed simultaneously. The same applies to adding paths.

```
R> removePath(model = ECSI, from = "Image", to = c("Satisfaction",
+      "Loyalty"))[["D"]]
```

```
             Image Expectation Quality Value Satisfaction Complaints Loyalty
Image            0           1       0     0            0          0       0
Expectation      0           0       1     1            1          0       0
Quality          0           0       0     1            1          0       0
Value            0           0       0     0            1          0       0
Satisfaction     0           0       0     0            0          1       1
Complaints       0           0       0     0            0          0       1
Loyalty          0           0       0     0            0          0       0
```

5.2. Exporting `plsm` objects for use with sem

If we are interested in a covariance based estimation of the path coefficients, we can for example use the **sem** package (Fox *et al.* 2012). We convert the model `ECSI` to a model representation, `semmod`, as used by the fitting method `sem`. For the scaling, we fix the LVs variances and the first loadings of an LVs instrument to one — thus comparing loadings of the two approaches is not fair. So we focus only on the estimated path coefficients. Though not necessary in this example, the variances for the MVs are fixed to one, too.

```
R> library("matrixcalc", pos = which(search()=="package:semPLS") + 1)
R> library("sem", pos = which(search()=="package:semPLS") + 1)
R> semmodECSI <- plsm2sem(model = ECSI,
+       fixedLoad = c(names(mobi)[grep("1", names(mobi))], "CUSCO"),
+       fixedVarMV=TRUE, fixedVarLV=TRUE)
R> ecsiSEM <- sem(model = semmodECSI, S = cor(mobi), N = nrow(mobi))
R> betaIndx <- grep("beta*", names(ecsiSEM$coeff))
R> cbind(ecsi$coefficients[names(ecsiSEM$coeff)[betaIndx], ],
+       CBSEM = ecsiSEM$coeff[betaIndx])

                               Path   Estimate      CBSEM
beta_1_2        Image -> Expectation 0.50470564 0.66437251
beta_2_3      Expectation -> Quality 0.55724786 0.69297541
beta_2_4        Expectation -> Value 0.05078755 0.11857742
beta_3_4            Quality -> Value 0.55721686 0.48088261
beta_1_5       Image -> Satisfaction 0.17883348 0.26916413
beta_2_5 Expectation -> Satisfaction 0.06442534 0.06259237
beta_3_5     Quality -> Satisfaction 0.51254524 0.50358818
beta_4_5       Value -> Satisfaction 0.19181566 0.16180132
beta_5_6  Satisfaction -> Complaints 0.52609731 0.44085882
beta_1_7            Image -> Loyalty 0.19535970 0.25719296
beta_5_7     Satisfaction -> Loyalty 0.48347472 0.37182098
beta_6_7       Complaints -> Loyalty 0.07123241 0.06378005
```

5.3. Importing model specification created with SmartPLS

The ECSI model, including the data for the mobile phone industry, is shipped with SmartPLS. After loading the project file `ecsi.splsp`, a directory `ecsi` is created, which contains the XML representation of the model, `ECSI.splsm` and the data file `mobi_250.txt`. The mentioned directory is located in the SmartPLS workspace. We can use the method `read.splsm` to create a `splsm` object in R. The argument `order="generic"` ensures to arrange the structural model according to the causal chain it implies. The data file is read as usual by `read.table()`.

The **semPLS** contains a SmartPLS workspace in the `/inst` directory. This workspace contains the SmartPLS model description `ECSI_Tenenhaus.splsm`. To get the system path to the file, use `system.file()`. We can see, all returned values with identical names to the result from (plsm) are equal. The `splsm` object is inheriting from class `plsm` and contains some SmartPLS specific additional values, e.g., node descriptions and positions of the graphical representation of the model. For now, these additional values are not used by **semPLS**.

```
R> ptf <- system.file("SmartPLS", "workspace", "ecsi", "ECSI_Tenenhaus.splsm",
+      package = "semPLS")
R> ECSIimported <- read.splsm(file = ptf, order = "generic")
R> for (i in names(ECSI)) print(all.equal(ECSI[i], ECSIimported[i]))

[1] TRUE
[1] TRUE
```

```
[1] TRUE
[1] TRUE
[1] TRUE
[1] TRUE
[1] TRUE
[1] TRUE
```

6. Summary and outlook

In this article, we have described some of the basic features of the **semPLS** package for working with PLS path models in R. While illustrating the usage of the different functions for model specification, model fitting, bootstrapping and computation of quality indices, we have focused at showing the modularity of the package. Due to this modularity the **semPLS** package can be extended easily.

As we have demonstrated, a variety of graphical tools support the researcher in exploring their model data. Parallel coordinates of bootstrap coefficients can be useful to detect unobserved heterogeneity. With the help of `mvpairs` plots, ceiling or floor effects and dubious observations are spotted quickly. By means of plots for the evolution of outer weights, convergence problems can be discovered.

Currently the **semPLS** does not support moderating effects in an object oriented way, though they can be specified manually. The `plpm` class will be extended to also support moderating effects. Further development plans are

- to enhance visualization methods by making them more dynamic and better accessible by the user, e.g., to add grouping variables post-hoc,
- to integrate a simulator function to draw samples from hypothetical models, thus opening the door to large scale Monte Carlo experiments, and
- to develop new methods for dealing with unobserved heterogeneity.

Computational details

With exception of the path diagrams, all computations and graphics in this paper have been done using R version 2.15.2 with the packages **XML** 3.95-0.1, **MASS** 7.3-23, **sem** 3.1-0, **semPLS** 1.0-10, **matrixcalc** 1.0-3, **lattice** 0.20-11, **grid** 2.15.2, **tools** 2.15.2. The path diagrams have been rendered with **dot - Graphviz** version 2.20.2 (Tue Mar 2 21:46:26 UTC 2010).

References

Addinsoft (2011). "**XLSTAT** – Statistics Package for **Excel**." URL http://www.xlstat.com/.

AT&T Research (2009). "**Graphviz** – Graph Visualization Software." URL http://www.graphviz.org/.

Canty A, Ripley BD (2012). ***boot**: Bootstrap R (S-PLUS) Functions*. R package version 1.3-4, URL http://CRAN.R-project.org/package=boot.

Chin WW (1998). "The Partial Least Squares Approach for Structural Equation Modeling." In GA Marcoulides (ed.), *Modern Methods for Business Research*, pp. 295–336. Lawrence Erlbaum Associates, London.

Chin WW (2003). ***PLS Graph** – Version 3.0*. Soft Modeling Inc. URL http://www.plsgraph.com/.

Chin WW, Dibbern J (2010). "An Introduction to a Permutation Based Procedure for Multi-Group PLS Analysis: Results of Tests of Differences on Simulated Data and a Cross Cultural Analysis of the Sourcing of Information System Services between Germany and the USA." In V Esposito Vinzi, WW Chin, J Henseler, HF Wang (eds.), *Handbook of Partial Least Squares: Concepts, Methods and Applications in Marketing and Related Fields*, chapter 7, pp. 171–193. Springer-Verlag, Berlin.

Davison AC, Hinkley DV (1997). *Bootstrap Methods and Their Applications*. Cambridge University Press, Cambridge. URL http://statwww.epfl.ch/davison/BMA/.

Diamantopoulos A, Winklhofer H (2001). "Index Construction with Formative Indicators: An Alternative to Scale Development." *Journal of Marketing Research*, **38**(2), 269–277.

Esposito Vinzi V, Fahmy T, Chatelin YM, Tenenhaus M (2007). *PLS Path Modeling: Some Recent Methodological Developments, a Software Integrated in **XLSTAT** and Its Application to Customer Satisfaction Studies*. Proceedings of the Academy of Marketing Science Conference "Marketing Theory and Practice in an Inter-Functional World", Verona, Italy, 11–14 July.

Esposito Vinzi V, Trinchera L, Amato S (2010). "PLS Path Modeling: From Foundations to Recent Developments and Open Issues for Model Assessment and Improvement." In V Esposito Vinzi, WW Chin, J Henseler, HF Wang (eds.), *Handbook of Partial Least Squares: Concepts, Methods and Applications in Marketing and Related Fields*, chapter 2, pp. 47–82. Springer-Verlag, Berlin.

Fox J (2006). "Structural Equation Modeling with the **sem** Package in R." *Structural Equation Modeling*, **13**(3), 465–486.

Fox J, Nie Z, Byrnes J (2012). ***sem**: Structural Equation Models*. R package version 3.0-0, URL http://CRAN.R-project.org/package=sem.

Fu JR (2006). ***VisualPLS** – Partial Least Square (PLS) Regression – An Enhanced GUI for **LVPLS** (PLS 1.8 PC) Version 1.04*. National Kaohsiung University of Applied Sciences, Taiwan. URL http://www2.kuas.edu.tw/prof/fred/vpls/.

Gansner E, Koutsofios E, North S (2006). "Drawing Graphs with DOT." *Technical report*, AT&T Research. URL http://www.graphviz.org/Documentation/dotguide.pdf.

Hair J, Sarstedt M, Ringle C, Mena J (2011a). "An Assessment of the Use of Partial Least Squares Structural Equation Modeling in Marketing Research." *Journal of the Academy of Marketing Science*, pp. 1–20.

Hair JF, Ringle CM, Sarstedt M (2011b). "PLS-SEM: Indeed a Silver Bullet." *Journal of Marketing Theory and Practice*, **19**(2), 139–151.

Henseler J, Ringle CM, Sinkovics RR (2009). "The Use of Partial Least Squares Path Modeling in International Marketing." *Advances in International Marketing*, **20**, 277–319.

Jarvis CB, MacKenzie SB, Podsakoff PM (2003). "A Critical Review of Construct Indicators and Measurement Model Misspecification in Marketing and Consumer research." *Journal of Consumer Research*, **30**, 199–218.

Jöreskog KG (1978). "Structural Analysis of Covariance and Correlation Matrices." *Psychometrika*, **43**(4), 443–477.

Lohmöller JB (1987). *PLS-PC: Latent Variables Path Analysis with Partial Least Squares – Version 1.8 for PCs under MS-Dos.*

Lohmöller JB (1989). *Latent Variable Path Modeling with Partial Least Squares.* Physica, Heidelberg.

MacKenzie SB, Podsakoff PM, Jarvis CB (2005). "The Problem of Measurement Model Misspecification in Behavioral and Organizational Research and Some Recommended Solutions." *Journal of Applied Psychology*, **90**(4), 710–730.

McArdle JJ (1980). "Causal Modeling Applied to Psychonomic Systems Simulation." *Behavior Research Methods and Instrumentation*, **12**, 193–209.

McArdle JJ, McDonald RP (1984). "Some Algebraic Properties of the Rectingular Action Model." *British Journal of Mathematical and Statistical Psychology.*

Monecke A, Leisch F (2012). "semPLS: Structural Equation Modeling Using Partial Least Squares." *Journal of Statistical Software*, **48**(3), 1–32. URL http://www.jstatsoft.org/v48/i03/.

R Development Core Team (2012). *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/.

Rigdon EE (1998). "Structural Equation Modeling." In GA Marcoulides (ed.), *Modern Methods for Business Research*, pp. 251–294. Lawrence Erlbaum Association, London.

Ringle CM, Sarstedt M, Straub D (2012). "A Criticle Look at the Use of PLS-SEM in MIS." *MIS Quaterly*, **36**(1), iii–xiv.

Ringle CM, Wende S, Will A (2005). "**SmartPLS** 2.0 (beta)." University of Hamburg, URL http://www.smartpls.de/.

Ringle CM, Wende S, Will A (2010). "Finite Mixture Partial Least Squares Analysis: Methodology and Numeric Examples." In V Esposito Vinzi, WW Chin, J Henseler, HF Wang (eds.), *Handbook of Partial Least Squares: Concepts, Methods and Applications in Marketing and Related Fields*, chapter 8, pp. 195–218. Springer-Verlag, Berlin.

Sanchez G, Aluja T (2012). ***pathmox**: Segmentation Trees in Partial Least Squares Path Modeling.* R package version 0.1-1, URL http://CRAN.R-project.org/package=pathmox.

Sanchez G, Trinchera L (2012). ***plspm**: Partial Least Squares Data Analysis Methods*. R package version 0.2-2, URL http://CRAN.R-project.org/package=plspm.

Sarkar D (2008). ***lattice**: Multivariate Data Visualization with R*. Springer-Verlag, New York.

Sarstedt M, Becker JM, M RC, Schwaiger M (2011). "Uncovering and Treating Unobserved Heterogeneity with FIMIX-PLS: Which Model Selection Criterion Provides an Appropriate Number of Segments?" *Schmalenbach Business Review*, **63**(1), 34–62.

Sarstedt M, Ringle CM (2010). "Treating Unobserved Heterogeneity in PLS Path Modelling: A Comparison of FIMIX-PLS with Different Data Analysis Strategies." *Applied Statistics*, **37**(8), 1299–1318.

Temme D, Kreis H, Hildebrandt L (2010). "A Comparison of Current PLS Path Modeling Software: Features, Ease-of-Use, and Performance." In V Esposito Vinzi, WW Chin, J Henseler, HF Wang (eds.), *Handbook of Partial Least Squares: Concepts, Methods and Applications in Marketing and Related Fields*, chapter 31. Springer-Verlag, Berlin.

Tenenhaus M, Esposito Vinzi V, Chatelin YM, Lauro C (2005). "PLS Path Modeling." *Computational Statistics & Data Analysis*, **48**, 159–205.

Test&Go (2006). ***SPAD** Version 6.0*. Paris, France.

Wold H (1966). "Estimation of Principal Components and Related Models by Iterative Least Squares." In PR Krishnaiah (ed.), *Multivariate Analysis*, pp. 391–420. Academic Press, New York.

Wold H (1982). "Soft Modeling: Intermediate between Traditional Model Building and Data Analysis." *Mathematical Statistics*, **6**, 333–346.

Wold H (1985). "Partial Least Squares." In S Kotz, NL Johnson (eds.), *Encyclopedia of Statistical Sciences*, volume 6, pp. 581–591. John Wiley & Sons, New York.

A. Notation

N	number of observations
K	number of observed variables
G	number of latent variables
$\boldsymbol{X}$	N x K matrix of observed variables (MVs); each variable standardized
$\boldsymbol{X}_g,\ g = 1, \ldots, G$	block of observed variables (MVs) belonging to latent $\boldsymbol{y}_g$
$\boldsymbol{F}_g$	N x K matrix measurement error for reflective block $\boldsymbol{X}_g$
$\boldsymbol{\delta}_g$	measurement error vector of length N for a formative LV $\boldsymbol{y}_g$
$\boldsymbol{Y}$	N x G matrix of for the latent variables (LVs)
$\boldsymbol{Z}$	N x G matrix of the structural model error terms
$\tilde{\boldsymbol{Y}}$	N x G matrix: inner approximation/estimation of factor scores
$\hat{\boldsymbol{Y}}$	N x G matrix: outer approximation/estimation of factor scores
$\boldsymbol{M}$	adjacency matrix (K x G) for the measurement model
$\boldsymbol{D}$	adjacency matrix (G x G) for the structural model
$\boldsymbol{W}$	K x G matrix of outer weights
$\boldsymbol{E}$	G x G matrix of inner weights
$\boldsymbol{B}$	G x G matrix of path coefficients
$\boldsymbol{T}$	G x G matrix of the total effects
$\boldsymbol{\Lambda}^{cross}$	K x G matrix of cross loadings
$\boldsymbol{\Lambda}^{outer}$	K x G matrix of outer loadings

Table 4: Notation used.

Affiliation:

Armin Monecke
Institut für Statistik
Ludwig–Maximilians–Universität München
Ludwigstr. 33
D-80539 München, Germany
E-mail: Armin.Monecke@stat.uni-muenchen.de
URL: http://www.statistik.lmu.de/~monecke/

Friedrich Leisch
Institut für angewandte Statistik und EDV
Universität für Bodenkultur Wien
Gregor-Mendel-Str. 33
A-1180 Wien, Austria
E-mail: Friedrich.Leisch@R-project.org
URL: http://www.rali.boku.ac.at/friedrichleisch.html