plsEXTpm

Quick reference

Jorge M. Mendes
Comprehensive Health Research Centre (CHRC)
NOVA Medical School|Faculdade de Ciências Médicas
Universidade Nova de Lisboa
Campo dos Mártires da Pátria, 1159-056 Lisbon, Portugal

Pedro S. Coelho
NOVA Information Management School (NOVAIMS)
Universidade Nova de Lisboa
Campus de Campolide, 1070-312 Lisbon, Portugal

May 14, 2025

1 The `plsExtpm` software

The model represented in Figure 1 is used here to illustrate the `plsExtpm` software.

The `plsExtpm` main function is:

```
plsExtpm(data, strucmod, measuremod, maxit=200, tol=1e-7, scaled=TRUE,
imputed = FALSE, wscheme = 'centroid', sum1 = FALSE, ngrid = 300,verbose
= TRUE, convCrit=c('relative','absolute')
```

This function contains three mandatory arguments: the *data*, the *structural model* and the *measurement model*. The data should be a `data frame` containing the manifest variables used in the measurement model. By default, the function does not impute the missing data (`imputed = FALSE`; when `TRUE` missing data imputed using mean imputation), scales the manifest variables to zero mean and unit standard deviation (`scaled = TRUE`), and runs the algorithm for 200 iterations using by default a `relative` convergence criterion (`convCrit`) with a `tolerance` of 1×10^{-7}. The default inner weighting scheme is `wscheme = "centroid` (the other alternatives are `"factorial"`, `"pathWeigthing"`, and `"smoothWeighting"`). If the

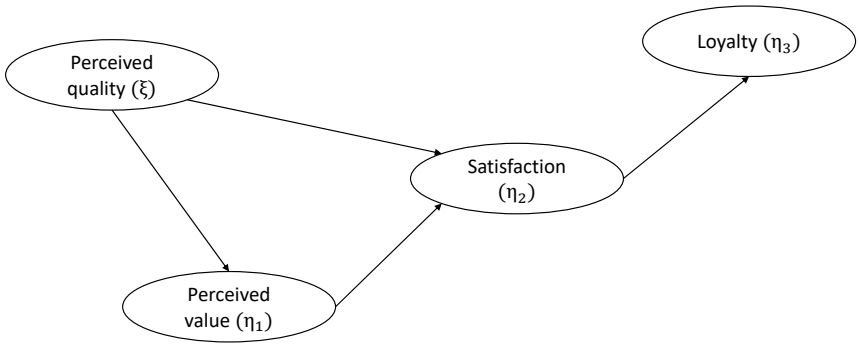

Figure 1: Structural equation model representing the causes and consequences of customer satisfaction (reduced version of the European Customer Satisfaction Index (ECSI) model)

default method is provided or any other of the traditional schemes, *factorial* and *pathWeighting*, but any of the partial structural relationships is nonlinear, it is automatically changed to `smoothWeighting`. If `verbose = TRUE`, informative messages regarding data checks are printed out. Finally, the estimated partial structural relationships are predicted on a `grid` of 300 points evenly distributed between the minimum and the maximum of the independent latent variable estimated factor scores.

The structural model is a four-column data frame. The columns should be named `source`, `target`, `type` and `K`, respectively. Whether a partial relationship between the `source` and the `target` latent variables is linear (`ln`) or nonlinear (`nln`) is indicated in the column `type`. If the partial relationship is linear, the column `K` should me set to `NA`, otherwise the dimension of the basis of the cubic regression spline is indicated in column `K`. To achieve the objectives referred to above, we show how to model the data in incremental way. Therefore, we start with a structural model assuming all partial relationships are nonlinear and approximated by a cubic regression spline with $K = 10$ knots (see Section **??** about the trade-off between dimension of the basis and penalisation term). This structural model is represented as follows:

```
ECSIsm_all_nlinear
```

```
##          source       target type  K
## 4       Quality        Value  nln 10
## 7       Quality Satisfaction  nln 10
## 8         Value Satisfaction  nln 10
## 10 Satisfaction      Loyalty  nln 10
```

The measurement model is a two-column matrix. The columns should be named `source` and `target`, respectively. Whether a latent variable is measured in a reflective or formative fashion depends upon the order they are represented in the measurement model. A reflective variable appears in the column `source` and its indicators in the column `target`. The formative indicators are represented the other way around. The reduced ECSI model of Example I is composed of three reflective latent variables, *Perceived Value*, *Customer Satisfaction* and *Customer Loyalty* and one formative latent variable, *Perceived Quality*. The measurement model is represented as follows:

```
ECSImm
```

```
##       source         target
##  [1,] "QUAL1"        "Quality"
##  [2,] "QUAL2"        "Quality"
##  [3,] "QUAL3"        "Quality"
##  [4,] "QUAL4"        "Quality"
##  [5,] "QUAL5"        "Quality"
##  [6,] "QUAL6"        "Quality"
##  [7,] "QUAL7"        "Quality"
##  [8,] "QUAL8"        "Quality"
##  [9,] "QUAL9"        "Quality"
## [10,] "Value"        "VALU1"
## [11,] "Value"        "VALU2"
## [12,] "Satisfaction" "SATI1"
## [13,] "Satisfaction" "SATI2"
## [14,] "Satisfaction" "SATI3"
## [15,] "Loyalty"      "LOYA1"
## [16,] "Loyalty"      "LOYA3"
```

To plot the estimated partial impacts of the structural model and to assess the appropriateness of the used dimension of the basis `plsExtpm` contains the function:

```
plsExtplot(pls.object, boot.object = NULL,x.LV = NULL, y.LV = NULL, verbose
= TRUE)
```

This function has only one mandatory argument, `pls.object`. It is as object of class `plsExtpm` resulting from a `plsExtpm()` run. If no variables are indicated in arguments `x.LV` and `y.LV` the function plots all partial structural relationships, otherwise, it plots only the requested *valid* relationship. Information messages regarding data inputs checks are printed out if `verbose = TRUE`, the default. If the user runs the bootstrap procedure (using the function `plsExtboot` described below), prior to the `plsExtplot` call, the result object of class `plsExtboot` can be provided the partial structural relationships plots contain the bootstrapped credible intervals. This analysis allows the user to decides whether each partial relationship should remain as a nonlinear one or change to the linear form.

Furthermore, taking advantage of the features of package `mgcv`, a function was specifically developed to check the choice of the basis dimension of the penalised regression smoothers, `plsExtcheck()`:

```
plsExtcheck(pls.object)
```

This function has only one mandatory argument, `pls.object`. It is as object of class `plsExtpm` resulting from a `plsExtpm()` run. Recurring to the `mgcv` package function `gam.check`, it takes a fitted `pls.object` object produced by `plsExtpm()` and produces some diagnostic information about the fitting procedure and results. For more information about the diagnostic information and plot produced we refer to @Wood2017.

If the user runs the bootstrap procedure (using the function `plsExtboot` described below), prior to the `plsExtplot` call, the result object of class `plsExtboot` can be provided the partial structural relationships plots contain the bootstrapped credible intervals. This analysis allows the user to decides whether each partial relationship should remain as a nonlinear one or change to the linear form.

The bootstrap procedure runs through function

```
plsExtboot(pls.object, nboot=200, start=c("ones", "old"), conf.level = 0.95,
verbose=TRUE)
```

This function `plsExtboot()` only requires as input an object of class `plsExtpm`. It runs a bootstrap procedure for 200 replicates (the default), with *outer weights* initialised to 1 (`start = 'ones'`; the alternative, which speeds up the procedure, `start = 'old'`, uses the outer weights resulting from the last iteration of the PLS-PM algorithm). Using the default confidence level of 95% (`conf.level = 0.95`) the function computes upper and lower credible limits for the partial structural relationships that might be used in `plsExtplot`. By

default, the function prints out important messages of the bootstrap procedure (`verbose = TRUE`).