Package ‘plssem’

March 23, 2026

Type Package

Title Complex Partial Least Squares Structural Equation Modeling

Version 0.1.0

Maintainer Kjell Solem Slupphaug <`slupphaugkjell@gmail.com`>

Description Estimate complex Structural Equation Models (SEMs) by fitting Partial Least Squares Structural Equation Modeling (PLS-SEM) and Partial Least Squares consistent Structural Equation Modeling (PLSc-SEM) specifications that handle categorical data, non-linear relations, and multilevel structures. The implementation follows Lohmöller (1989) for the classic PLS-SEM algorithm, Dijkstra and Henseler (2015) for consistent PLSc-SEM, Dijkstra et al., (2014) for nonlinear PLSc-SEM, and Schuberth, Henseler, Dijkstra (2018) for ordinal PLS-SEM and PLSc-SEM. Additional extensions are under development.
References:
Lohmöller, J.-B. (1989, ISBN:9783790803002).
``Latent Variable Path Modeling with Partial Least Squares."
Dijkstra, T. K., & Henseler, J. (2015).
<doi:10.1016/j.jmva.2015.06.002>.
``Consistent partial least squares path modeling."
Dijkstra, T. K., & Schermelleh-Engel, K. (2014).
<doi:10.1016/j.csda.2014.07.008>.
``Consistent partial least squares for nonlinear structural equation models."
Schuberth, F., Henseler, J., & Dijkstra, T. K. (2018).
<doi:10.1007/s11135-018-0767-9>.
``Partial least squares path modeling using ordinal categorical indicators."

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.3.3

Imports stats, modsem (>= 1.0.17), lme4, lavaan, cli, stringr, purrr, matrixStats, SimDesign, Rfast, collapse, mvnfast, reformulas

Depends R (>= 4.1.0)

URL `https://github.com/kss2k/plssem`

Suggests knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Author Kjell Solem Slupphaug [aut, cre] (ORCID:
<https://orcid.org/0009-0005-8324-2834>)

Repository CRAN

Date/Publication 2026-03-23 09:40:06 UTC

Contents

oneIntOrdered *oneIntOrdered*

Description

A simulated dataset.

Examples

```
m <- '
  X =~ x1 + x2 + x3
  Z =~ z1 + z2 + z3
  Y =~ y1 + y2 + y3

  Y ~ X + Z + X:Z
'

fit <- pls(m, oneIntOrdered)
summary(fit)
```

`parameter_estimates` *Generic accessor for model parameter estimates*

Description

Generic accessor for model parameter estimates

Usage

```
parameter_estimates(object, ...)
```

Arguments

`object`	A fitted model object.
`...`	Additional arguments passed to methods.

Value

A parameter table describing the fitted model.

`parameter_estimates.plssem` *Parameter estimates for ‘plssem‘ objects*

Description

Parameter estimates for ‘plssem‘ objects

Usage

```
## S3 method for class 'plssem'
parameter_estimates(object, colon.pi = TRUE, label.renamed.prod = FALSE, ...)
```

Arguments

`object`	An object of class ‘plssem‘.
`colon.pi`	Logical; whether to replace labels for interaction terms with colon notation.
`label.renamed.prod`	Logical; whether renamed product labels should be retained when colon expansion occurs.
`...`	Additional arguments (not used).

Value

A parameter table (data frame) describing the fitted model.

pls *Fit Partial Least Squares Structural Equation Models*

Description

‘pls()‘ estimates Partial Least Squares Structural Equation Models (PLS-SEM) and their consistent (PLSc) variants. The function accepts ‘lavaan‘-style syntax, handles ordered indicators through polychoric correlations and probit factor scores, and supports multilevel specifications expressed with ‘lme4‘-style random effects terms inside the structural model.

Usage

```
pls(
  syntax,
  data,
  standardize = TRUE,
  consistent = TRUE,
  bootstrap = FALSE,
  sample = 50L,
  ordered = NULL,
  mcpls = NULL,
  probit = NULL,
  tolerance = 1e-05,
  max.iter.0_5 = 100L,
  mc.min.iter = 5L,
  mc.max.iter = 250L,
  mc.reps = 20000L,
  mc.tol = 0.001,
  mc.fixed.seed = FALSE,
  mc.polyak.juditsky = FALSE,
  mc.fn.args = list(),
  verbose = interactive(),
  ...
)
```

Arguments

`syntax` Character string with ‘lavaan‘-style model syntax describing both measurement (‘=~‘) and structural (‘~‘) relations. Random effects are specified with ‘(term | cluster)‘ statements.

`data` A ‘data.frame‘ or coercible object containing the manifest indicators referenced in ‘syntax‘. Ordered factors are automatically detected, but can also be supplied explicitly through ‘ordered‘.

`standardize` Logical; if ‘TRUE‘, indicators are standardized before estimation so that factor scores have comparable scales.

`consistent` Logical; ‘TRUE‘ requests PLSc corrections, whereas ‘FALSE‘ fits the traditional PLS model.

`bootstrap` Logical; if ‘TRUE‘, nonparametric bootstrap standard errors are computed with ‘sample‘ resamples.

`sample` Integer giving the number of bootstrap resamples drawn when ‘bootstrap = TRUE‘.

`ordered` Optional character vector naming manifest indicators that should be treated as ordered when computing polychoric correlations.

`mcpls` Should a Monte-Carlo consistency correction be applied?

`probit` Logical; overrides the automatic choice of probit factor scores that is based on whether ordered indicators are present.

`tolerance` Numeric; Convergence criteria/tolerance.

`max.iter.0_5` Maximum number of PLS iterations performed when estimating the measurement and structural models.

`mc.min.iter` Minimum number of iterations in MC-PLS algorithm.

`mc.max.iter` Maximum number of iterations in MC-PLS algorithm.

`mc.reps` Monte-Carlo sample size in MC-PLS algorithm.

`mc.tol` Tolerance in MC-PLS algorithm.

`mc.fixed.seed` Should a fixed seed be used in the MC-PLS algorithm?

`mc.polyak.juditsky`
Should the polyak.juditsky running average method be applied in the MC-PLS algorithm?

`mc.fn.args` Additional arguments to MC-PLS algorithm, mainly for controling the step size.

`verbose` Should verbose output be printed?

`...` Currently unused, reserved for future extensions.

Value

An object of class ‘plssem‘ containing the estimated parameters, fit measures, factor scores, and any bootstrap results. Methods such as ‘summary()‘, ‘print()‘, and ‘coef()‘ can be applied to inspect the fit.

See Also

[summary.plssem()], [print.plssem()]

Examples

```
# Linear Model with Continuous Data


library(plssem)
library(modsem)

tpb <- '
```

```
# Outer Model (Based on Hagger et al., 2007)
  ATT =~ att1 + att2 + att3 + att4 + att5
  SN =~ sn1 + sn2
  PBC =~ pbc1 + pbc2 + pbc3
  INT =~ int1 + int2 + int3
  BEH =~ b1 + b2

# Inner Model (Based on Steinmetz et al., 2011)
  INT ~ ATT + SN + PBC
  BEH ~ INT + PBC
'

fit <- pls(tpb, TPB, bootstrap = TRUE)
summary(fit)

# Linear Model with Ordered Data
tpb <- '
# Outer Model (Based on Hagger et al., 2007)
  ATT =~ att1 + att2 + att3 + att4 + att5
  SN =~ sn1 + sn2
  PBC =~ pbc1 + pbc2 + pbc3
  INT =~ int1 + int2 + int3
  BEH =~ b1 + b2

# Inner Model (Based on Steinmetz et al., 2011)
  INT ~ ATT + SN + PBC
  BEH ~ INT + PBC
'

fit <- pls(tpb, TPB_Ordered, bootstrap = TRUE)
summary(fit)

# Multilevel Random Slopes Model with Continuous Data
syntax <- "
  X =~ x1 + x2 + x3
  Z =~ z1 + z2 + z3
  Y =~ y1 + y2 + y3
  W =~ w1 + w2 + w3
  Y ~ X + Z + (1 + X + Z | cluster)
  W ~ X + Z + (1 + X + Z | cluster)
"

fit <- pls(syntax, data = randomSlopes, bootstrap = TRUE)
summary(fit)

# Multilevel Random Slopes Model with Ordered Data
syntax <- "
  X =~ x1 + x2 + x3
  Z =~ z1 + z2 + z3
  Y =~ y1 + y2 + y3
  W =~ w1 + w2 + w3
  Y ~ X + Z + (1 + X + Z | cluster)
  W ~ X + Z + (1 + X + Z | cluster)
```

```
"

fit <- pls(syntax, data = randomSlopesOrdered, bootstrap = TRUE)
summary(fit)

# Multilevel Random Intercepts Model with Continuous Data
syntax <- '
  f =~ y1 + y2 + y3
  f ~ x1 + x2 + x3 + w1 + w2 + (1 | cluster)
'

fit <- pls(syntax, data = randomIntercepts, bootstrap = TRUE)
summary(fit)

# Multilevel Random Intercepts Model with Ordered Data
syntax <- '
  f =~ y1 + y2 + y3
  f ~ x1 + x2 + x3 + w1 + w2 + (1 | cluster)
'

fit <- pls(syntax, data = randomInterceptsOrdered, bootstrap = TRUE)
summary(fit)

# Interaction Model with Continuous Data
m <- '
  X =~ x1 + x2 + x3
  Z =~ z1 + z2 + z3
  Y =~ y1 + y2 + y3

  Y ~ X + Z + X:Z
'

fit <- pls(m, modsem::oneInt, bootstrap = TRUE)
summary(fit)

# Interaction Model with Ordered Data
m <- '
  X =~ x1 + x2 + x3
  Z =~ z1 + z2 + z3
  Y =~ y1 + y2 + y3

  Y ~ X + Z + X:Z
'

fit <- pls(m, oneIntOrdered, bootstrap = TRUE)
summary(fit)
```

`print.plssem` *Print a ‘plssem‘ object*

Description

Print a ‘plssem‘ object

Usage

```
## S3 method for class 'plssem'
print(x, ...)
```

Arguments

`x`	An object of class ‘plssem‘.
`...`	Additional arguments for compatibility with the generic.

Value

The input object, invisibly.

`print.SummaryPlsSem` *Print a ‘SummaryPlsSem‘ object*

Description

Print a ‘SummaryPlsSem‘ object

Usage

```
## S3 method for class 'SummaryPlsSem'
print(x, ...)
```

Arguments

`x`	A ‘SummaryPlsSem‘ object as returned by [summary.plssem()].
`...`	Additional arguments for compatibility with the generic.

Value

The input object, invisibly.

`randomIntercepts` *randomIntercepts*

Description

A simulated dataset.

Examples

```
syntax <- '
  f =~ y1 + y2 + y3
  f ~ x1 + x2 + x3 + w1 + w2 + (1 | cluster)
'

fit <- pls(syntax, data = randomIntercepts)
summary(fit)
```

`randomInterceptsOrdered` *randomInterceptsOrdered*

Description

A simulated dataset.

Examples

```
syntax <- '
  f =~ y1 + y2 + y3
  f ~ x1 + x2 + x3 + w1 + w2 + (1 | cluster)
'

fit <- pls(syntax, data = randomInterceptsOrdered)
summary(fit)
```

`randomSlopes` *randomSlopes*

Description

A simulated dataset. syntax <- " X =~ x1 + x2 + x3 Z =~ z1 + z2 + z3 Y =~ y1 + y2 + y3 W =~ w1 + w2 + w3 Y ~ X + Z + (1 + X + Z | cluster) W ~ X + Z + (1 + X + Z | cluster) "

fit <- pls(syntax, data = randomSlopes) fit

`randomSlopesOrdered` *randomSlopesOrdered*

Description

A simulated dataset.

Examples

```
syntax <- "
  X =~ x1 + x2 + x3
  Z =~ z1 + z2 + z3
  Y =~ y1 + y2 + y3
  W =~ w1 + w2 + w3
  Y ~ X + Z + (1 + X + Z | cluster)
  W ~ X + Z + (1 + X + Z | cluster)
"

fit <- pls(syntax, data = randomSlopesOrdered)
fit
summary(fit)
```

`summary.plssem` *Summarize a fitted ‘plssem‘ model*

Description

Summarize a fitted ‘plssem‘ model

Usage

```
## S3 method for class 'plssem'
summary(object, ...)
```

Arguments

`object`	An object of class ‘plssem‘.
`...`	Additional arguments passed to or from methods.

Value

A ‘SummaryPlsSem‘ object containing formatted parameter estimates.

TPB_Ordered *TPB_Ordered*

Description

A simulated dataset.

Examples

```
tpb <- '
# Outer Model (Based on Hagger et al., 2007)
  ATT =~ att1 + att2 + att3 + att4 + att5
  SN =~ sn1 + sn2
  PBC =~ pbc1 + pbc2 + pbc3
  INT =~ int1 + int2 + int3
  BEH =~ b1 + b2

# Inner Model (Based on Steinmetz et al., 2011)
  INT ~ ATT + SN + PBC
  BEH ~ INT + PBC
'

fit <- pls(tpb, TPB_Ordered)
summary(fit)
```

Index