

# Package ‘plssem’

June 1, 2026

**Type** Package

**Title** Complex Partial Least Squares Structural Equation Modeling

**Version** 0.1.2

**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. The MC-OrdPLSc algorithm, used to handle ordinal interaction models is detailed in Slupphaug et al., (2026).

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.csa.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."

Slupphaug, K. Mehmetoglu, M. & Mittner, M. (2026).

<doi:10.31234/osf.io/fwzj6\_v1>.

``Consistent Estimates from Biased Estimators: Monte-

Carlo Consistent Partial Least Squares for Latent Interaction Models with Ordinal Indicators."

**License** GPL-3

**Encoding** UTF-8

**LazyData** true

**Imports** methods, stats, modsem ( $\geq 1.0.20$ ), lme4, lavaan, stringr, purrr, matrixStats, Rfast, collapse, mvnfast, reformulas, future, future.apply, progressr, FNN, MASS

**Depends** R ( $\geq 4.1.0$ )

**URL** <https://github.com/kss2k/plssem>, <https://kss2k.github.io/plssem/>

**Suggests** knitr, rmarkdown, ggplot2, mice, mvtnorm, pkgload

**VignetteBuilder** knitr

**Config/roxygen2/version** 8.0.0

**NeedsCompilation** yes

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

**Repository** CRAN

**Date/Publication** 2026-06-01 19:20:02 UTC

## Contents

boot . . . . .	3
coef,PlsModel-method . . . . .	3
fit_measures . . . . .	4
implied_construct_corr . . . . .	5
implied_indicator_corr . . . . .	5
is_admissible . . . . .	6
is_mcpls . . . . .	7
oneIntOrdered . . . . .	7
parameter_estimates . . . . .	8
parameter_estimates,PlsModel-method . . . . .	8
pls . . . . .	9
pls_construct_scores . . . . .	12
pls_predict . . . . .	13
predict,PlsModel-method . . . . .	14
print.PlsSemPredict . . . . .	14
print.SummaryPlsSem . . . . .	15
randomIntercepts . . . . .	15
randomInterceptsOrdered . . . . .	16
randomSlopes . . . . .	16
randomSlopesOrdered . . . . .	16
show,PlsModel-method . . . . .	17
summary,PlsModel-method . . . . .	17
titanic . . . . .	18
TPB_Ordered . . . . .	19
vcov,PlsModel-method . . . . .	19

**Index**

**20**

---

boot	<i>Retrieve bootstrap coefficient matrix</i>
------	--

---

**Description**

Retrieve bootstrap coefficient matrix

**Usage**

```
boot(object)

## S4 method for signature 'PlsModel'
boot(object)
```

**Arguments**

object            A fitted model object.

**Value**

A `PlsSemMatrix` of bootstrap replicate parameter vectors (rows = replicates, cols = parameters).

**Examples**

```
library(modsem)
library(plssem)

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

fit <- pls(m, oneInt, bootstrap = TRUE, boot.R = 50)
boot(fit)
```

---

coef, PlsModel-method	<i>Extract coefficients from a PlsModel model</i>
-----------------------	---

---

**Description**

Extract coefficients from a `PlsModel` model

**Usage**

```
## S4 method for signature 'PlsModel'
coef(object, ...)
```

```
## S4 method for signature 'PlsModel'
coefficients(object, ...)
```

**Arguments**

object	A PlsModel object.
...	Currently unused.

**Value**

A named PlsSemVector of parameter estimates.

---

fit\_measures

*Fit Measures*


---

**Description**

Computes global fit measures (e.g., chi-square, SRMR, RMSEA) for a fitted model.

**Usage**

```
fit_measures(object, saturated = FALSE, mc.reps = 1e+06, ...)
```

```
## S4 method for signature 'PlsModel'
fit_measures(object, saturated = FALSE, mc.reps = 1e+06, ...)
```

**Arguments**

object	A fitted [PlsModel] object.
saturated	Logical; if 'TRUE', compute the saturated fit.
mc.reps	Integer; number of Monte Carlo resamples used for MC-PLSc fit.
...	Reserved for future extensions.

**Value**

A named list with fit statistics.

---

`implied_construct_corr`*Implied Construct Correlation Matrix*

---

**Description**

Returns the implied construct correlation matrix for a fitted model.

**Usage**

```
implied_construct_corr(object, saturated = FALSE, mc.reps = 1e+06, ...)
```

```
## S4 method for signature 'PlsModel'
```

```
implied_construct_corr(object, saturated = FALSE, mc.reps = 1e+06, ...)
```

**Arguments**

<code>object</code>	A fitted [PlsModel] object.
<code>saturated</code>	Logical; if 'TRUE', return the saturated implied matrix.
<code>mc.reps</code>	Integer; number of Monte Carlo resamples used for MC-PLSc.
<code>...</code>	Reserved for future extensions.

**Details**

For higher-order models, this is computed for the combined model returned by [combinedModel()].

**Value**

A [PlsSemMatrix].

---

`implied_indicator_corr`*Implied Indicator Correlation Matrix*

---

**Description**

Returns the implied indicator correlation matrix for a fitted model.

**Usage**

```
implied_indicator_corr(object, saturated = FALSE, mc.reps = 1e+06, ...)
```

```
## S4 method for signature 'PlsModel'
```

```
implied_indicator_corr(object, saturated = FALSE, mc.reps = 1e+06, ...)
```

**Arguments**

object	A fitted [PlsModel] object.
saturated	Logical; if 'TRUE', return the saturated implied matrix.
mc.reps	Integer; number of Monte Carlo resamples used for MC-PLSc.
...	Reserved for future extensions.

**Details**

For higher-order models, this is computed for the combined model returned by [combinedModel()].

**Value**

A numeric matrix.

---

is_admissible	<i>Check whether a fitted model has admissible parameter estimates</i>
---------------	--

---

**Description**

Check whether a fitted model has admissible parameter estimates

**Usage**

```
is_admissible(object)

## S4 method for signature 'PlsModel'
is_admissible(object)
```

**Arguments**

object	A fitted PlsModel object.
--------	---------------------------

**Value**

A single logical value.

---

`is_mcpls`*Check whether an object uses the MC-OrdPLSc estimator*

---

**Description**

Check whether an object uses the MC-OrdPLSc estimator

**Usage**

```
is_mcpls(object)
```

```
## S4 method for signature 'PlsModel'  
is_mcpls(object)
```

**Arguments**

`object`            A fitted model object.

**Value**

TRUE or FALSE.

---

`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,PlsModel-method  
*Parameter estimates for PlsModel objects*

---

**Description**

Parameter estimates for PlsModel objects

**Usage**

```
## S4 method for signature 'PlsModel'
parameter_estimates(object, colon.pi = TRUE, label.renamed.prod = FALSE, ...)
```

**Arguments**

object            A PlsModel object.  
 colon.pi         Logical; replace product-indicator labels with colon notation (X:Z).  
 label.renamed.prod    Logical; retain renamed product labels when colon expansion occurs.  
 ...              Currently unused.

**Value**

A PlsSemParTable data frame.

**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,  
  ordered = NULL,  
  missing = c("listwise", "mean", "kNN"),  
  knn.k = 5,  
  mcpls = NULL,  
  mc.fast.lmer = mcpls,  
  probit = NULL,  
  tolerance = 1e-05,  
  max.iter.0_5 = 100L,  
  boot.ncores = 1L,  
  boot.ncpus = NULL,  
  boot.parallel = c("no", "multicore", "multisession", "snow"),  
  boot.R = 50L,  
  boot.iseed = NULL,  
  sample = NULL,  
  mc.min.iter = 50L,  
  mc.max.iter = 500L,  
  mc.reps = 20000L,  
  mc.fixed.seed = FALSE,  
  mc.polyak.juditsky = TRUE,  
  mc.pj.extrapolate = TRUE,  
  mc.tol = if (mc.polyak.juditsky) 1e-04 else 0.001,  
  mc.delta.se = TRUE,  
  mc.delta.jacobian.k = max(floor(boot.R/100L), 1),  
  mc.fn.args = list(),  
  verbose = interactive(),  
  boot.optimize = TRUE,  
  mc.boot.control = list(min.iter = mc.min.iter, max.iter = mc.max.iter, mc.reps =  
    floor(0.5 * mc.reps), tol = mc.tol, polyak.juditsky = mc.polyak.juditsky,  
    pj.extrapolate = FALSE, verbose = FALSE, fixed.seed = TRUE, reuse.p.start = TRUE),
```

```

    reliabilities = NULL,
    ...
)

```

### Arguments

<code>syntax</code>	Character string with lavaan-style model syntax describing both measurement (=~) and structural (~) relations. Random effects are specified with (term   cluster) statements.
<code>data</code>	A <code>data.frame</code> or coercible object containing the manifest indicators referenced in <code>syntax</code> . Ordered factors are automatically detected, but can also be supplied explicitly through <code>ordered</code> .
<code>standardize</code>	Logical; if TRUE, indicators are standardized before estimation so that factor scores have comparable scales.
<code>consistent</code>	Logical; TRUE requests PLSc corrections, whereas FALSE fits the traditional PLS model.
<code>bootstrap</code>	Logical; if TRUE, nonparametric bootstrap standard errors are computed with <code>boot.R</code> resamples.
<code>ordered</code>	Optional character vector naming manifest indicators that should be treated as ordered when computing polychoric correlations.
<code>missing</code>	Character string specifying how to handle missing indicator data. "listwise" removes rows with missing values (listwise deletion). "mean" imputes missing indicator values using simple univariate imputation: the mean for continuous variables, the median for ordered variables with more than two categories, and the mode for binary ordered variables (two categories) or nominal variables. "kNN" (or "knn") imputes missing indicator values using k-nearest neighbors imputation (kNN). When <code>missing = "kNN"</code> , rows with all indicators missing are removed prior to imputation, and rows with missing <code>cluster</code> values are removed for multilevel models.
<code>knn.k</code>	Integer specifying the number of neighbors (k) used when <code>missing = "kNN"</code> .
<code>mcpls</code>	Should the model be estimated using the Monte-Carlo Consistent Partial Least Squares (MC-PLSc) algorithm?
<code>mc.fast.lmer</code>	Should a faster (biased) GLS based estimator of the Mixed-Effects model be used in conjunction with the MC-PLS algorithm?
<code>probit</code>	Logical; overrides the automatic choice of probit factor scores that is based on whether ordered indicators are present.
<code>tolerance</code>	Numeric; Convergence criteria/tolerance.
<code>max.iter.0_5</code>	Maximum number of PLS iterations performed when estimating the measurement and structural models.
<code>boot.ncores</code>	Integer: number of workers to be used for parallel bootstrapping. Parallel bootstrapping is enabled when <code>boot.ncores &gt; 1</code> .
<code>boot.ncpus</code>	Deprecated alias for <code>boot.ncores</code> .

<code>boot.parallel</code>	The type of parallel operation to be used (if any). The default is "no". "multisession" runs the bootstrap using multiple background R sessions (works on all platforms), while "multicore" uses forked processes (not available on Windows). "snow" is kept for backwards compatibility and is treated as an alias for "multisession". Internally this is implemented using the future package
<code>boot.R</code>	Integer giving the number of bootstrap resamples drawn when <code>bootstrap = TRUE</code> .
<code>boot.iseed</code>	An integer to set the bootstrap seed. Or NULL if no reproducible results are needed. This works for both serial and parallel settings. When <code>boot.iseed</code> is not NULL, <code>.Random.seed</code> (if it exists) in the global environment is left untouched.
<code>sample</code>	DEPRECATED. Integer giving the number of bootstrap resamples drawn when <code>bootstrap = TRUE</code> .
<code>mc.min.iter</code>	Minimum number of iterations in MC-PLS algorithm.
<code>mc.max.iter</code>	Maximum number of iterations in MC-PLS algorithm.
<code>mc.reps</code>	Monte-Carlo sample size in MC-PLS algorithm.
<code>mc.fixed.seed</code>	Should a fixed seed be used in the MC-PLS algorithm? Setting a fixed seed will likely yield less accurate estimates, but can substantially improve the stability and computational efficiency of the algorithm.
<code>mc.polyak.juditsky</code>	Should the polyak.juditsky running average method be applied in the MC-PLS algorithm?
<code>mc.pj.extrapolate</code>	Logical; if TRUE (the default), the Polyak-Juditsky convergence point is estimated via NLS exponential extrapolation (with Aitken $\delta^2$ as a fallback). If FALSE a warm start is performed instead, and the plain Polyak-Juditsky average is used. Only relevant when <code>mc.polyak.juditsky = TRUE</code> .
<code>mc.tol</code>	Tolerance in MC-PLS algorithm.
<code>mc.delta.se</code>	Should delta-method standard errors be computed for MC-PLS estimates?
<code>mc.delta.jacobian.k</code>	Integer number of Monte-Carlo Jacobians to average when computing delta-method standard errors. Defaults to one per 100 bootstrap resamples, with a minimum of 1.
<code>mc.fn.args</code>	Additional arguments to MC-PLS algorithm, mainly for controlling the step size.
<code>verbose</code>	Should verbose output be printed?
<code>boot.optimize</code>	Logical; if TRUE and <code>bootstrap = TRUE</code> , applies the settings in <code>mc.boot.control</code> inside each bootstrap replicate (MC-PLS only). In general it will lead to slightly larger and less accurate standard errors.
<code>mc.boot.control</code>	List of control parameters passed to the MC-PLS algorithm inside each bootstrap replicate when <code>boot.optimize = TRUE</code> .
<code>reliabilities</code>	Optional named numeric vector of user-supplied reliabilities used for the PLS consistency correction.
<code>...</code>	Internal arguments. For advanced users only.

**Value**

A Plssem object containing the estimated parameters, fit measures, factor scores, and any bootstrap results. Methods such as `summary()`, `coef()`, and `parameter_estimates()` can be applied to inspect the fit.

**See Also**

[summary](#), [show](#)

**Examples**

```
library(plssem)
library(modsem)

tpb <- '
ATT =~ att1 + att2 + att3 + att4 + att5
SN =~ sn1 + sn2
PBC =~ pbc1 + pbc2 + pbc3
INT =~ int1 + int2 + int3
BEH =~ b1 + b2
INT ~ ATT + SN + PBC
BEH ~ INT + PBC
'

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

---

pls\_construct\_scores    *Construct latent variable scores*

---

**Description**

Convenience wrapper around `[pls_predict()]` returning only the predicted latent scores matrix.

**Usage**

```
pls_construct_scores(object, ...)
```

**Arguments**

<code>object</code>	A fitted Plssem model.
<code>...</code>	Passed to <code>[pls_predict()]</code> .

**Value**

A `PlsSemMatrix` of predicted latent scores.

---

pls_predict	<i>Predict from a fitted PLS-SEM model</i>
-------------	--

---

### Description

Predict from a fitted PLS-SEM model

### Usage

```
pls_predict(object, ...)

## S4 method for signature 'PlsModel'
pls_predict(
  object,
  approach = c("earliest", "direct"),
  newdata = NULL,
  std.ord.exp = FALSE,
  benchmark = "R2",
  benchmark.vars = c("endog", "exog", "all"),
  ...
)
```

### Arguments

object	A fitted PlsModel object.
...	Additional arguments passed to internal helpers.
approach	Prediction approach. If approach = "earliest" (default), then only indicators of exogenous benchmark.vars are used for prediction. If approach = "direct", then all indicators are used.
newdata	Optional new data matrix/data frame.
std.ord.exp	Logical; standardize ordinal expectation scores.
benchmark	Benchmark type(s). Either length 1 (recycled) or one entry per indicator (optionally named). Supported: "r2", "rmse", "mae", "q2_predict", "acc", "ord_mae".
benchmark.vars	What predictions should be benchmarked? If benchmark.vars = "endog" (default), prediction benchmarks are applied to indicators of endogenous benchmark.vars. If benchmark.vars = "exog", prediction benchmarks are applied to indicators of exogenous benchmark.vars. If benchmark.vars = "all", prediction benchmarks are applied to all of the indicators in the model.

### Value

A PlsSemPredict object with matrices and benchmark results.

---

predict, PlsModel-method

*Predict from a fitted PlsModel (alias for [pls\\_predict](#))*

---

### Description

Predict from a fitted PlsModel (alias for [pls\\_predict](#))

### Usage

```
## S4 method for signature 'PlsModel'
predict(object, newdata = NULL, ...)
```

### Arguments

object	A fitted PlsModel object.
newdata	Optional new data matrix/data frame.
...	Further arguments passed to <a href="#">pls_predict</a> .

### Value

A PlsSemPredict object.

---

print.PlsSemPredict *Print a PlsSemPredict object*

---

### Description

Print a PlsSemPredict object

### Usage

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

### Arguments

x	A PlsSemPredict object.
...	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\(\)](#).  
...                   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)
```

---

show,PlsModel-method    *Show a PlsModel object*

---

### Description

Called automatically when an object is printed at the prompt. Displays the package version, iteration count, and the parameter table.

### Usage

```
## S4 method for signature 'PlsModel'
show(object)
```

### Arguments

object            A PlsModel object.

### Value

object, invisibly.

---

summary,PlsModel-method  
                           *Summarize a fitted PlsModel model*

---

### Description

Summarize a fitted PlsModel model

### Usage

```
## S4 method for signature 'PlsModel'
summary(object, fit = TRUE, ...)
```

### Arguments

object            A PlsModel object.  
 fit               Logical; whether to compute fit measures.  
 ...               Currently unused.

### Value

A SummaryPlsSem list with formatted results.

---

titanic

*Titanic Passenger Survival Data Set.*

---

### Description

This dataset has been re-packaged for convenience from <https://github.com/paulhendricks/titanic>

**PassengerId** Passenger ID

**Survived** Passenger Survival Indicator

**Pclass** Passenger Class

**Name** Name

**Sex** Sex

**Age** Age

**SibSp** Number of Siblings/Spouses Aboard

**Parch** Number of Parents/Children Aboard

**Ticket** Ticket Number

**Fare** Passenger Fare

**Cabin** Cabin

**Embarked** Port of Embarkation

**Female** Dummy variable for Sex="female"

### Format

A data frame with 1309 rows and 12 variables:

### Source

<https://www.kaggle.com/c/titanic/data>

### Examples

```
fit <- pls("Survived ~ Age + Female + Age:Female",
          data = titanic, ordered = "Survived")
pls_predict(fit, benchmark = "acc")
```

---

 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)
```

---

 vcov,PlsModel-method *Extract the variance-covariance matrix from a PlsModel model*


---

### Description

Extract the variance-covariance matrix from a PlsModel model

### Usage

```
## S4 method for signature 'PlsModel'
vcov(object, ...)
```

### Arguments

object	A PlsModel object.
...	Currently unused.

### Value

A PlsSemMatrix (bootstrap-based vcov, or NULL).

# Index

boot, 3  
boot, PlsModel-method (boot), 3

coef, PlsModel-method, 3  
coefficients, PlsModel-method  
    (coef, PlsModel-method), 3

fit\_measures, 4  
fit\_measures, PlsModel-method  
    (fit\_measures), 4

implied\_construct\_corr, 5  
implied\_construct\_corr, PlsModel-method  
    (implied\_construct\_corr), 5  
implied\_indicator\_corr, 5  
implied\_indicator\_corr, PlsModel-method  
    (implied\_indicator\_corr), 5

is\_admissible, 6  
is\_admissible, PlsModel-method  
    (is\_admissible), 6

is\_mcpls, 7  
is\_mcpls, PlsModel-method (is\_mcpls), 7

oneIntOrdered, 7

parameter\_estimates, 8  
parameter\_estimates, PlsModel-method, 8

pls, 9  
pls\_construct\_scores, 12  
pls\_predict, 13, 14  
pls\_predict, PlsModel-method  
    (pls\_predict), 13  
predict, PlsModel-method, 14  
print.PlsSemPredict, 14  
print.SummaryPlsSem, 15

randomIntercepts, 15  
randomInterceptsOrdered, 16  
randomSlopes, 16  
randomSlopesOrdered, 16

show, 12  
show, PlsModel-method, 17  
summary, 12, 15  
summary, PlsModel-method, 17

titanic, 18  
TPB\_Ordered, 19

vcov, PlsModel-method, 19