Package 'seminr'

July 23, 2020

Type Package

Title Domain-Specific Language for Building and Estimating Structural Equation Models

Version 1.1.0

Date 2020-07-23

Description A powerful, easy to syntax for specifying and estimating complex Structural Equation Models. Models can be estimated using Partial Least Squares Path Modeling or Covariance-Based Structural Equation Modeling or covariance based Confirmatory Factor Analysis.

Imports parallel, lavaan, MASS

License GPL-3

Depends R (>= 3.1.0)

LazyData TRUE

RoxygenNote 7.1.0

Suggests knitr, testthat, rmarkdown

VignetteBuilder knitr

Encoding UTF-8

NeedsCompilation no

Author Soumya Ray [aut, ths],
Nicholas Patrick Danks [aut, cre],
Juan Manuel Velasquez Estrada [aut],
James Uanhoro [ctr],
Arturo Heynar Cano Bejar [ctr]

Maintainer Nicholas Patrick Danks <nicholasdanks@hotmail.com>

Repository CRAN

Date/Publication 2020-07-23 16:00:02 UTC

R topics documented:

`as.reflective` *Converts all contructs of a measurement model, or just a single construct into reflective factors.*

Description

Converts all contructs of a measurement model, or just a single construct into reflective factors.

Usage

```
as.reflective(x, ...)
```

Arguments

x	A measurement model defined by constructs or a single composite construct defined by composite
...	Any further parameters for the specific construct.

See Also

as.reflective.measurement_model, as.reflective.construct

Examples

```
mobi_mm <- constructs(
  composite("Image",        multi_items("IMAG", 1:5)),
  composite("Expectation",  multi_items("CUEX", 1:3)),
  composite("Value",        multi_items("PERV", 1:2))
)

new_mm <- as.reflective(mobi_mm)
```

---

as.reflective.construct

Converts a contruct of a measurement model into a reflective factor.

Description

Converts a contruct of a measurement model into a reflective factor.

Usage

```
## S3 method for class 'construct'
as.reflective(x, ...)
```

Arguments

x	A measurement model defined by constructs or a single composite construct defined by composite
...	Any further parameters for the specific construct.

See Also

as.reflective.measurement_model

Examples

```
mobi_mm <- constructs(
  composite("Image",        multi_items("IMAG", 1:5)),
  composite("Expectation",  multi_items("CUEX", 1:3)),
  composite("Value",        multi_items("PERV", 1:2))
)

new_mm <- as.reflective(mobi_mm)
```

`as.reflective.interaction` *Converts interaction of a measurement model into a reflective factors.*

Description

Converts interaction of a measurement model into a reflective factors.

Usage

```
## S3 method for class 'interaction'
as.reflective(x, ...)
```

Arguments

`x`	A measurement model defined by `constructs` or a single composite construct defined by `composite`
`...`	Any further parameters for the specific construct.

See Also

`as.reflective.measurement_model`

Examples

```
mobi_mm <- constructs(
  composite("Image",        multi_items("IMAG", 1:5)),
  composite("Expectation",  multi_items("CUEX", 1:3)),
  composite("Value",        multi_items("PERV", 1:2))
)

new_mm <- as.reflective(mobi_mm)
```

as.reflective.measurement_model

Converts all contructs of a measurement model, or just a single construct into reflective factors.

Description

Converts all contructs of a measurement model, or just a single construct into reflective factors.

Usage

```
## S3 method for class 'measurement_model'
as.reflective(x, ...)
```

Arguments

x	A measurement model defined by constructs or a single composite construct defined by composite
...	Any further parameters for the specific construct.

See Also

as.reflective.construct

Examples

```
mobi_mm <- constructs(
  composite("Image",        multi_items("IMAG", 1:5)),
  composite("Expectation",  multi_items("CUEX", 1:3)),
  composite("Value",        multi_items("PERV", 1:2))
)

new_mm <- as.reflective(mobi_mm)
```

associations

Specifies inter-item covariances that should be supplied to CBSEM estimation (estimate_cbsem) or CFA estimation (estimate_cfa)

Description

Specifies inter-item covariances that should be supplied to CBSEM estimation (estimate_cbsem) or CFA estimation (estimate_cfa)

Usage

```
associations(...)
```

Arguments

...	One or more associations defined by item_errors

Examples

```
covaries <- associations(
  item_errors(c("a1", "a2"), c("b1", "b2")),
  item_errors("a3", "c3")
)
```

---

bootstrap_model *seminr bootstrap_model Function*

Description

The seminr package provides a natural syntax for researchers to describe PLS structural equation models. bootstrap_model provides the verb for bootstrapping a pls model from the model parameters and data.

Usage

```
bootstrap_model(seminr_model, nboot = 500, cores = NULL, seed = NULL, ...)
```

Arguments

seminr_model	A fully estimated model with associated data, measurement model and structural model
nboot	A parameter specifying the number of bootstrap iterations to perform, default value is 500. If 0 then no bootstrapping is performed.
cores	A parameter specifying the maximum number of cores to use in the parallelization.
seed	A parameter to specify the seed for reproducibility of results. Default is NULL.
...	A list of parameters passed on to the estimation method.

References

Hair, J. F., Hult, G. T. M., Ringle, C. M., and Sarstedt, M. (2017). A Primer on Partial Least Squares Structural Equation Modeling (PLS-SEM), 2nd Ed., Sage: Thousand Oaks.

See Also

relationships constructs paths interaction_term

Examples

```
data(mobi)
# seminr syntax for creating measurement model
mobi_mm <- constructs(
  composite("Image",        multi_items("IMAG", 1:5)),
  composite("Expectation",  multi_items("CUEX", 1:3)),
  composite("Value",        multi_items("PERV", 1:2)),
  composite("Satisfaction", multi_items("CUSA", 1:3)),
  interaction_term(iv = "Image", moderator = "Expectation", method = orthogonal),
  interaction_term(iv = "Image", moderator = "Value", method = orthogonal)
)

# structural model: note that name of the interactions construct should be
#  the names of its two main constructs joined by a '.' in between.
mobi_sm <- relationships(
  paths(to = "Satisfaction",
        from = c("Image", "Expectation", "Value",
                 "Image*Expectation", "Image*Value"))
)

seminr_model <- estimate_pls(data = mobi,
                             measurement_model = mobi_mm,
                             structural_model = mobi_sm)

# Load data, assemble model, and bootstrap
boot_seminr_model <- bootstrap_model(seminr_model = seminr_model,
                                     nboot = 50, cores = 2, seed = NULL)

summary(boot_seminr_model)
```

`composite` *Composite construct measurement model specification*

Description

`composite` creates the composite measurement model matrix for a specific construct, specifying the relevant items of the construct and assigning the relationship of either correlation weights (Mode A) or regression weights (Mode B).

Usage

```
composite(construct_name, item_names,weights = correlation_weights)
```

Arguments

`construct_name`	of construct
`item_names`	returned by the `multi_items` or `single_item` functions

`weights` is the relationship between the construct and its items. This can be specified as `correlation_weights` or `mode_A` for correlation weights (Mode A) or as `regression_weights` or `mode_B` for regression weights (Mode B). Default is correlation weights.

Details

This function conveniently maps composite defined measurement items to a construct and is estimated using PLS.

See Also

See `constructs`, `reflective`

Examples

```
mobi_mm <- constructs(
  composite("Image",        multi_items("IMAG", 1:5), weights = correlation_weights),
  composite("Expectation",  multi_items("CUEX", 1:3), weights = mode_A),
  composite("Quality",      multi_items("PERQ", 1:7), weights = regression_weights),
  composite("Value",        multi_items("PERV", 1:2), weights = mode_B)
)
```

---

confidence_interval *seminr confidence intervals function*

Description

The `seminr` package provides a natural syntax for researchers to describe PLS structural equation models. `confidence_interval` provides the verb for calculating the confidence intervals of a direct or mediated path in a bootstrapped SEMinR model.

Usage

```
confidence_interval(boot_seminr_model, from, to, through, alpha)
```

Arguments

`boot_seminr_model`	A bootstrapped model returned by the `bootstrap_model` function.
`from`	A parameter specifying the antecedent composite for the path.
`to`	A parameter specifying the outcome composite for the path.
`through`	A parameter to specify the mediator for the path. Default is NULL.
`alpha`	A parameter for specifying the alpha for the confidence interval. Default is 0.05.

References

Zhao, X., Lynch Jr, J. G., & Chen, Q. (2010). Reconsidering Baron and Kenny: Myths and truths about mediation analysis. Journal of consumer research, 37(2), 197-206.

See Also

bootstrap_model

Examples

```
mobi_mm <- constructs(
composite("Image",        multi_items("IMAG", 1:5)),
composite("Expectation",  multi_items("CUEX", 1:3)),
composite("Quality",      multi_items("PERQ", 1:7)),
composite("Value",        multi_items("PERV", 1:2)),
composite("Satisfaction", multi_items("CUSA", 1:3)),
composite("Complaints",   single_item("CUSCO")),
composite("Loyalty",      multi_items("CUSL", 1:3))
)

# Creating structural model
mobi_sm <- relationships(
  paths(from = "Image",        to = c("Expectation", "Satisfaction", "Loyalty")),
  paths(from = "Expectation",  to = c("Quality", "Value", "Satisfaction")),
  paths(from = "Quality",      to = c("Value", "Satisfaction")),
  paths(from = "Value",        to = c("Satisfaction")),
  paths(from = "Satisfaction", to = c("Complaints", "Loyalty")),
  paths(from = "Complaints",   to = "Loyalty")
)

# Estimating the model
mobi_pls <- estimate_pls(data = mobi,
                         measurement_model = mobi_mm,
                         structural_model = mobi_sm)

# Load data, assemble model, and bootstrap
boot_seminr_model <- bootstrap_model(seminr_model = mobi_pls,
                                     nboot = 50, cores = 2, seed = NULL)

confidence_interval(boot_seminr_model = boot_seminr_model,
                    from = "Image",
                    through = "Expectation",
                    to = "Satisfaction",
                    alpha = 0.05)
```

`constructs` *Measurement functions*

Description

`constructs` creates the constructs from measurement items by assigning the relevant items to each construct and specifying reflective or formative (composite/causal) measurement models

Usage

```
constructs(...)
```

Arguments

`...`	Comma separated list of the construct variable measurement specifications, as generated by the `reflective()`, or `composite()` methods.

Details

This function conveniently maps measurement items to constructs using root name, numbers, and affixes with explicit definition of formative or reflective relationships

See Also

See `composite`, `reflective`

Examples

```
mobi_mm <- constructs(
  reflective("Image",        multi_items("IMAG", 1:5)),
  reflective("Expectation",  multi_items("CUEX", 1:3)),
  reflective("Quality",      multi_items("PERQ", 1:7)),
  reflective("Value",        multi_items("PERV", 1:2)),
  reflective("Satisfaction", multi_items("CUSA", 1:3)),
  reflective("Complaints",   single_item("CUSCO")),
  reflective("Loyalty",      multi_items("CUSL", 1:3))
)
```

---

df_xtab_matrix *Cross-tabulates columns of a dataframe into a matrix with NAs for unspecified pairs*

Description

Cross-tabulates columns of a dataframe into a matrix with NAs for unspecified pairs

Usage

```
df_xtab_matrix(model, df, rows, columns)
```

Arguments

model	A formula indicating relevant columns from data frame
df	A data.frame of columns to cross-tabulate
rows	A vector of row names for the matrix to sort by
columns	A vector of column names for the matrix to sort by

---

estimate_cbsem *seminr estimate_cbsem() function*

Description

The seminr package provides a natural syntax for researchers to describe structural equation models.

Usage

```
estimate_cbsem(data, measurement_model, structural_model, item_associations=NULL,
               estimator="MLR", ...)
```

Arguments

data	A dataframe containing the indicator measurement data.
measurement_model	A list represention of how constructs are measured by their items, generated using constructs. Note that only reflective constructs are supported for CB-SEM models, though a composite measurement model can be converted into a reflective one using as.reflective.
structural_model	A source-to-target matrix representing the structural model, generated by relationships.
item_associations	An item-to-item matrix representing error covariances that are freed for estimation. This matrix is created by associations(), or defaults to NULL (no associations).
estimator	A character string indicating which estimation method to use in Lavaan. It defaults to "MLR" for robust estimation. See the Lavaan documentation for other supported estimators.
...	Any other parameters to pass to lavaan::sem during estimation.

References

Joreskog, K. G. (1973). A general method for estimating a linear structural equation system In: Goldberger AS, Duncan OD, editors. Structural Equation Models in the Social Sciences. New York: Seminar Press.

See Also

as.reflective relationships constructs paths associations item_errors

Examples

```
mobi <- mobi

#seminr syntax for creating measurement model
mobi_mm <- constructs(
  reflective("Image",        multi_items("IMAG", 1:5)),
  reflective("Quality",      multi_items("PERQ", 1:7)),
  reflective("Value",        multi_items("PERV", 1:2)),
  reflective("Satisfaction", multi_items("CUSA", 1:3)),
  reflective("Complaints",   single_item("CUSCO")),
  reflective("Loyalty",      multi_items("CUSL", 1:3))
)

#seminr syntax for freeing up item-item covariances
mobi_am <- associations(
  item_errors(c("PERQ1", "PERQ2"), "IMAG1")
)

#seminr syntax for creating structural model
mobi_sm <- relationships(
  paths(from = c("Image", "Quality"), to = c("Value", "Satisfaction")),
  paths(from = c("Value", "Satisfaction"), to = c("Complaints", "Loyalty")),
  paths(from = "Complaints",   to = "Loyalty")
)

# Estimate model and get results
mobi_cbsem <- estimate_cbsem(mobi, mobi_mm, mobi_sm, mobi_am)

# Use or capture the summary object for more results and metrics
summary(mobi_cbsem)

cbsem_summary <- summary(mobi_cbsem)
cbsem_summary$descriptives$correlations$constructs
```

---

estimate_cfa *seminr estimate_cfa() function*

Description

Estimates a Confirmatory Factor Analysis (CFA) model

Usage

```
estimate_cfa(data, measurement_model, item_associations=NULL, estimator="MLR", ...)
```

Arguments

`data`	A `dataframe` containing the indicator measurement data.
`measurement_model`	A list represention of how constructs are measured by their items, generated using `constructs`. Note that only reflective constructs are supported for CB-SEM models, though a composite measurement model can be converted into a reflective one using `as.reflective`.
`item_associations`	An item-to-item matrix representing error covariances that are freed for estimation. This matrix is created by `associations()`, or defaults to NULL (no associations).
`estimator`	A character string indicating which estimation method to use in Lavaan. It defaults to "MLR" for robust estimation. See the Lavaan documentation for other supported estimators.
`...`	Any other parameters to pass to `lavaan::sem` during estimation.

References

Jöreskog, K.G. (1969) A general approach to confirmatory maximum likelihood factor analysis. Psychometrika, 34, 183-202.

See Also

`constructs` `reflective` `associations` `item_errors` `as.reflective`

#' @examples mobi <- mobi

#seminr syntax for creating measurement model mobi_mm <- constructs(reflective("Image", multi_items("IMAG", 1:5)), reflective("Expectation", multi_items("CUEX", 1:3)), reflective("Quality", multi_items("PERQ", 1:7)))

#seminr syntax for freeing up item-item covariances mobi_am <- associations(item_errors(c("PERQ1", "PERQ2"), "CUEX3"), item_errors("IMAG1", "CUEX2"))

mobi_cfa <- estimate_cfa(mobi, mobi_mm, mobi_am)

`estimate_lavaan_ten_berge` *seminr estimate_lavaan_ten_berge() function*

Description

Estimates factor scores using ten Berge method for a fitted Lavaan model

Usage

```
estimate_lavaan_ten_berge(fit)
```

Arguments

fit	A fitted lavaan object – can be extracted from cbesem estimation or from using Lavaan directly.

Value

A list with two elements: ten berge scores; weights for calculating scores

Examples

```
#' #seminr syntax for creating measurement model
mobi_mm <- constructs(
  reflective("Image",        multi_items("IMAG", 1:5)),
  reflective("Quality",      multi_items("PERQ", 1:7)),
  reflective("Value",        multi_items("PERV", 1:2)),
  reflective("Satisfaction", multi_items("CUSA", 1:3)),
  reflective("Complaints",   single_item("CUSCO")),
  reflective("Loyalty",      multi_items("CUSL", 1:3))
)

#seminr syntax for freeing up item-item covariances
mobi_am <- associations(
  item_errors(c("PERQ1", "PERQ2"), "IMAG1")
)

#seminr syntax for creating structural model
mobi_sm <- relationships(
  paths(from = c("Image", "Quality"), to = c("Value", "Satisfaction")),
  paths(from = c("Value", "Satisfaction"), to = c("Complaints", "Loyalty")),
  paths(from = "Complaints",   to = "Loyalty")
)

# Estimate model and get results
cbsem <- estimate_cbsem(mobi, mobi_mm, mobi_sm, mobi_am)
tb <- estimate_lavaan_ten_berge(cbsem$lavaan_model)
tb$scores
tb$weights
```

---

estimate_pls *seminr estimate_pls() function*

Description

The seminr package provides a natural syntax for researchers to describe structural equation models.

Usage

```
estimate_pls(data, measurement_model, structural_model,
             inner_weights = path_weighting)
```

Arguments

data	A dataframe containing the indicator measurement data.
measurement_model	A source-to-target matrix representing the outer/measurement model, generated by constructs.
structural_model	A source-to-target matrix representing the inner/structural model, generated by relationships.
inner_weights	A parameter declaring which inner weighting scheme should be used path_weighting is default, alternately path_factorial can be used.

See Also

relationships constructs paths interaction_term bootstrap_model

Examples

```
mobi <- mobi

#seminr syntax for creating measurement model
mobi_mm <- constructs(
             reflective("Image",        multi_items("IMAG", 1:5)),
             reflective("Expectation",  multi_items("CUEX", 1:3)),
             reflective("Quality",      multi_items("PERQ", 1:7)),
             reflective("Value",        multi_items("PERV", 1:2)),
             reflective("Satisfaction", multi_items("CUSA", 1:3)),
             reflective("Complaints",   single_item("CUSCO")),
             reflective("Loyalty",      multi_items("CUSL", 1:3))
           )
#seminr syntax for creating structural model
mobi_sm <- relationships(
  paths(from = "Image",        to = c("Expectation", "Satisfaction", "Loyalty")),
  paths(from = "Expectation",  to = c("Quality", "Value", "Satisfaction")),
  paths(from = "Quality",      to = c("Value", "Satisfaction")),
  paths(from = "Value",        to = c("Satisfaction")),
  paths(from = "Satisfaction", to = c("Complaints", "Loyalty")),
  paths(from = "Complaints",   to = "Loyalty")
)

mobi_pls <- estimate_pls(data = mobi,
                         measurement_model = mobi_mm,
                         structural_model = mobi_sm)

summary(mobi_pls)
plot_scores(mobi_pls)
```

fSquared *seminr fSquared Function*

Description

The fSquared function calculates f^2 effect size for a given IV and DV

Usage

```
fSquared(seminr_model, iv, dv)
```

Arguments

seminr_model	A seminr_model containing the estimated seminr model.
iv	An independent variable in the model.
dv	A dependent variable in the model.

References

Cohen, J. (2013). Statistical power analysis for the behavioral sciences. Routledge.

Examples

```
mobi_mm <- constructs(
             reflective("Image",        multi_items("IMAG", 1:5)),
             reflective("Expectation",  multi_items("CUEX", 1:3)),
             reflective("Quality",      multi_items("PERQ", 1:7)),
             reflective("Value",        multi_items("PERV", 1:2)),
             reflective("Satisfaction", multi_items("CUSA", 1:3)),
             reflective("Complaints",   single_item("CUSCO")),
             reflective("Loyalty",      multi_items("CUSL", 1:3))
           )

mobi_sm <- relationships(
  paths(from = "Image",        to = c("Expectation", "Satisfaction", "Loyalty")),
  paths(from = "Expectation",  to = c("Quality", "Value", "Satisfaction")),
  paths(from = "Quality",      to = c("Value", "Satisfaction")),
  paths(from = "Value",        to = c("Satisfaction")),
  paths(from = "Satisfaction", to = c("Complaints", "Loyalty")),
  paths(from = "Complaints",   to = "Loyalty")
)

mobi_pls <- estimate_pls(data = mobi,
                         measurement_model = mobi_mm,
                         structural_model = mobi_sm)

fSquared(mobi_pls, "Image", "Satisfaction")
```

higher_composite *higher_composite*

Description

higher_composite creates a higher order construct from first order constructs using the two-stage method (Becker et al., 2012).

Usage

```
higher_composite(construct_name, dimensions, method, weights)
```

Arguments

construct_name	of second order construct
dimensions	the first order constructs
method	is the estimation method, default is two_stage
weights	is the relationship between the second order construct and first order constructs. This can be specified as correlation_weights or mode_A for correlation weights (Mode A) or as regression_weights or mode_B for regression weights (Mode B). Default is correlation weights.

Details

This function conveniently maps first order constructs onto second order constructs using construct names.

See Also

See constructs, reflective

Examples

```
mobi_mm <- constructs(
  composite("Image",        multi_items("IMAG", 1:5), weights = correlation_weights),
  composite("Expectation",  multi_items("CUEX", 1:3), weights = mode_A),
  higher_composite("Quality",  c("Image","Expectation"), method = two_stage),
  composite("Value",        multi_items("PERV", 1:2), weights = mode_B)
)
```

`interaction_term` *Interaction function*

Description

`interaction_term` creates interaction measurement items by applying product indicator, two stage, or orthogonal approaches to creating new interaction constructs.

Usage

```
interaction_term(iv, moderator, method, weights)
```

Arguments

`iv`	The independent variable that is subject to moderation.
`moderator`	The moderator variable.
`method`	The method to generate the estimated interaction term with a default of 'two_stage'.
`weights`	The weighting mode for interaction items in a PLS model (only) with default of 'modeA'.
	Interaction Combinations as generated by the `interaction` or `interaction_term` methods.

Details

This function automatically generates interaction measurement items for a PLS or a CBSEM model.

Examples

```
data(mobi)

# seminr syntax for creating measurement model
mobi_mm <- constructs(
  composite("Image",        multi_items("IMAG", 1:5)),
  composite("Expectation",  multi_items("CUEX", 1:3)),
  composite("Value",        multi_items("PERV", 1:2)),
  composite("Satisfaction", multi_items("CUSA", 1:3)),
  interaction_term(iv = "Image", moderator = "Expectation", method = orthogonal),
  interaction_term(iv = "Image", moderator = "Value", method = product_indicator)
)

#  structural model: note that name of the interactions construct should be
#  the names of its two main constructs joined by a '*' in between.
mobi_sm <- relationships(
  paths(to = "Satisfaction",
        from = c("Image", "Expectation", "Value",
                 "Image*Expectation", "Image*Value"))
)
```

```
mobi_pls <- estimate_pls(mobi, mobi_mm, mobi_sm)
summary(mobi_pls)
```

`item_errors` *Specifies pair of items, or sets of items, that should covary. Used to specify error covariances for* `associations`.

Description

Specifies pair of items, or sets of items, that should covary. Used to specify error covariances for `associations`.

Usage

```
item_errors(items_a, items_b)
```

Arguments

`items_a`	One or more items that should covary
`items_b`	One or more items that should covary

Examples

```
item_errors(c("a1", "a2"), c("b1", "b2"))
```

`mobi` *Measurement Instrument for the Mobile Phone Industry*

Description

The data set is used as measurement instrument for the european customer satisfaction index (ECSI) adapted to the mobile phone market, see Tenenhaus et al. (2005).

Usage

```
mobi
```

Format

A data frame with 250 rows and 24 variables:

CUEX1 Expectations for the overall quality of "your mobile phone provider" at the moment you became customer of this provider

CUEX2 Expectations for "your mobile phone provider" to provide products and services to meet your personal need

CUEX3 How often did you expect that things could go wrong at "your mobile phone provider

CUSA1 Overall satisfaction

CUSA2 Fulfillment of expectations

CUSA3 How well do you think "your mobile phone provider" compares with your ideal mobile phone provider?

CUSCO You complained about "your mobile phone provider" last year. How well, or poorly, was your most recent complaint handled or You did not complain about "your mobile phone provider" last year. Imagine you have to complain to "your mobile phone rovider" because of a bad quality of service or product. To what extent do you think that "your mobile phone provider" will care about your complaint?

CUSL1 If you would need to choose a new mobile phone provider how likely is it that you would choose "your provider" again?

CUSL2 Let us now suppose that other mobile phone providers decide to lower their fees and prices, but "your mobile phone provider" stays at the same level as today. At which level of difference (in percentage) would you choose another mobile phone provider?

CUSL3 If a friend or colleague asks you for advice, how likely is it that you would recommend "your mobile phone provider"?

IMAG1 It can be trusted what it says and does

IMAG2 It is stable and firmly established

IMAG3 It has a social contribution to society

IMAG4 It is concerned with customers

IMAG5 It is innovative and forward looking

PERQ1 Overall perceived quality

PERQ2 Technical quality of the network

PERQ3 Customer service and personal advice offered

PERQ4 Quality of the services you use

PERQ5 Range of services and products offered

PERQ6 Reliability and accuracy of the products and services provided

PERQ7 Clarity and transparency of information provided

PERV1 Given the quality of the products and services offered by "your mobile phone provider" how would you rate the fees and prices that you pay for them?

PERV2 Given the fees and prices that you pay for "your mobile phone provider" how would you rate the quality of the products and services offered by "your mobile phone provider"?

Details

The data frame mobi contains the observed data for the model specified by ECSImobi.

References

Tenenhaus, M., V. E. Vinzi, Y.-M. Chatelin, and C. Lauro (2005) PLS path modeling. Computational Statistics & Data Analysis 48, 159-205.

Examples

```
data("mobi")
```

`mode_A` *Outer weighting scheme functions to estimate construct weighting.*

Description

`mode_A`, `correlation_weights` and `mode_B`, `regression_weights` specify the outer weighting scheme to be used in the estimation of the construct weights and score.

Usage

```
mode_A(mmMatrix, i, normData, construct_scores)
```

Arguments

`mmMatrix`	is the `measurement_model` - a source-to-target matrix representing the measurement model, generated by `constructs`.
`i`	is the name of the construct to be estimated.
`normData`	is the dataframe of the normalized item data.
`construct_scores`	is the matrix of construct scores generated by `estimate_model`.

mode_B *Outer weighting scheme functions to estimate construct weighting.*

Description

mode_A, correlation_weights and mode_B, regression_weights specify the outer weighting scheme to be used in the estimation of the construct weights and score.

Usage

```
mode_B(mmMatrix, i, normData, construct_scores)
```

Arguments

mmMatrix	is the measurement_model - a source-to-target matrix representing the measurement model, generated by constructs.
i	is the name of the construct to be estimated.
normData	is the dataframe of the normalized item data.
construct_scores	is the matrix of construct scores generated by estimate_model.

multi_items *Multi-items measurement model specification*

Description

multi_items creates a vector of measurment names given the item prefix and number range.

Usage

```
multi_items(item_name, item_numbers, ...)
```

Arguments

item_name	Prefix name of items
item_numbers	The range of number suffixews for the items
...	Additional Item names and nubers

See Also

See single_item

Examples

```
mobi_mm <- constructs(
  composite("Image",        multi_items("IMAG", 1:5), weights = correlation_weights),
  composite("Expectation",  multi_items("CUEX", 1:3), weights = mode_A),
  composite("Quality",      multi_items("PERQ", 1:7), weights = regression_weights),
  composite("Value",        multi_items("PERV", 1:2), weights = mode_B)
)
```

`orthogonal` `orthogonal` *creates interaction measurement items by using the orthogonalized approach wherein*

Description

This function automatically generates interaction measurement items for a PLS SEM using the orthogonalized approach..

Usage

```
# orthogonalization approach as per Henseler & Chin (2010):
 orthogonal(iv, moderator, weights)
```

Arguments

`iv`	The independent variable that is subject to moderation.
`moderator`	The moderator variable.
`weights`	is the relationship between the items and the interaction terms. This can be specified as `correlation_weights` or `mode_A` for correlation weights (Mode A) or as `regression_weights` or `mode_B` for regression weights (Mode B). Default is correlation weights.

References

Henseler & Chin (2010), A comparison of approaches for the analysis of interaction effects between latent variables using partial least squares path modeling. Structural Equation Modeling, 17(1),82-109.

Examples

```
data(mobi)

# seminr syntax for creating measurement model
mobi_mm <- constructs(
  composite("Image",        multi_items("IMAG", 1:5)),
  composite("Expectation",  multi_items("CUEX", 1:3)),
  composite("Value",        multi_items("PERV", 1:2)),
  composite("Satisfaction", multi_items("CUSA", 1:3)),
  interaction_term(iv = "Image", moderator = "Expectation", method = orthogonal),
```

```
  interaction_term(iv = "Image", moderator = "Value", method = orthogonal)
)

#  structural model: note that name of the interactions construct should be
#  the names of its two main constructs joined by a '*' in between.
mobi_sm <- relationships(
  paths(to = "Satisfaction",
        from = c("Image", "Expectation", "Value",
                 "Image*Expectation", "Image*Value"))
)

mobi_pls <- estimate_pls(mobi, mobi_mm, mobi_sm)
summary(mobi_pls)
```

`path_factorial` *Inner weighting scheme functions to estimate inner paths matrix*

Description

`path_factorial` and `path_weighting` specify the inner weighting scheme to be used in the estimation of the inner paths matrix

Usage

```
path_factorial(smMatrix,construct_scores, dependant, paths_matrix)
```

Arguments

`smMatrix`	is the `structural_model` - a source-to-target matrix representing the inner/structural model, generated by `relationships`.
`construct_scores`	is the matrix of construct scores generated by `estimate_model`.
`dependant`	is the vector of dependant constructs in the model.
`paths_matrix`	is the matrix of estimated path coefficients estimated by `estimate_model`.

References

Lohmoller, J.-B. (1989). Latent variables path modeling with partial least squares. Heidelberg, Germany: Physica Verlag.

path_weighting *Inner weighting scheme functions to estimate inner paths matrix*

Description

path_factorial and path_weighting specify the inner weighting scheme to be used in the estimation of the inner paths matrix

Usage

```
path_weighting(smMatrix,construct_scores, dependant, paths_matrix)
```

Arguments

smMatrix	is the structural_model - a source-to-target matrix representing the inner/structural model, generated by relationships.
construct_scores	is the matrix of construct scores generated by estimate_model.
dependant	is the vector of dependant constructs in the model.
paths_matrix	is the matrix of estimated path coefficients estimated by estimate_model.

References

Lohmoller, J.B. (1989). Latent variables path modeling with partial least squares. Heidelberg, Germany: Physica-Verlag.

PLSc *seminr PLSc Function*

Description

The PLSc function calculates the consistent PLS path coefficients and loadings for a common-factor model. It returns a seminr_model containing the adjusted and consistent path coefficients and loadings for common-factor models and composite models.

Usage

```
PLSc(seminr_model)
```

Arguments

seminr_model	A seminr_model containing the estimated seminr model.

References

Dijkstra, T. K., & Henseler, J. (2015). Consistent Partial Least Squares Path Modeling, 39(X).

See Also

relationships constructs paths interaction_term bootstrap_model

Examples

```
mobi <- mobi

#seminr syntax for creating measurement model
mobi_mm <- constructs(
             reflective("Image",        multi_items("IMAG", 1:5)),
             reflective("Expectation",  multi_items("CUEX", 1:3)),
             reflective("Quality",      multi_items("PERQ", 1:7)),
             reflective("Value",        multi_items("PERV", 1:2)),
             reflective("Satisfaction", multi_items("CUSA", 1:3)),
             reflective("Complaints",   single_item("CUSCO")),
             reflective("Loyalty",      multi_items("CUSL", 1:3))
           )
#seminr syntax for creating structural model
mobi_sm <- relationships(
  paths(from = "Image",        to = c("Expectation", "Satisfaction", "Loyalty")),
  paths(from = "Expectation",  to = c("Quality", "Value", "Satisfaction")),
  paths(from = "Quality",      to = c("Value", "Satisfaction")),
  paths(from = "Value",        to = c("Satisfaction")),
  paths(from = "Satisfaction", to = c("Complaints", "Loyalty")),
  paths(from = "Complaints",   to = "Loyalty")
)

seminr_model <- estimate_pls(data = mobi,
                             measurement_model = mobi_mm,
                             structural_model = mobi_sm)

PLSc(seminr_model)
```

`product_indicator` `product_indicator` *creates interaction measurement items by scaled product indicator approach.*

Description

This function automatically generates interaction measurement items for a PLS SEM using scaled product indicator approach.

Usage

```
# standardized product indicator approach as per Henseler & Chin (2010):
 product_indicator(iv, moderator, weights)
```

Arguments

iv	The independent variable that is subject to moderation.
moderator	The moderator variable.
weights	is the relationship between the items and the interaction terms. This can be specified as correlation_weights or mode_A for correlation weights (Mode A) or as regression_weights or mode_B for regression weights (Mode B). Default is correlation weights.

References

Henseler & Chin (2010), A comparison of approaches for the analysis of interaction effects between latent variables using partial least squares path modeling. Structural Equation Modeling, 17(1),82-109.

Examples

```
data(mobi)

# seminr syntax for creating measurement model
mobi_mm <- constructs(
 composite("Image",        multi_items("IMAG", 1:5),weights = mode_A),
 composite("Expectation",  multi_items("CUEX", 1:3),weights = mode_A),
 composite("Value",        multi_items("PERV", 1:2),weights = mode_A),
 composite("Satisfaction", multi_items("CUSA", 1:3),weights = mode_A),
 interaction_term(iv = "Image",
                  moderator = "Expectation",
                  method = product_indicator,
                  weights = mode_A),
 interaction_term(iv = "Image",
                  moderator = "Value",
                  method = product_indicator,
                  weights = mode_A)
)

# structural model: note that name of the interactions construct should be
#  the names of its two main constructs joined by a '*' in between.
mobi_sm <- relationships(
  paths(to = "Satisfaction",
        from = c("Image", "Expectation", "Value",
                 "Image*Expectation", "Image*Value"))
)

# Load data, assemble model, and estimate using semPLS
mobi <- mobi
seminr_model <- estimate_pls(mobi, mobi_mm, mobi_sm, inner_weights = path_factorial)
```

`reflective` *Reflective construct measurement model specification*

Description

`reflective` creates the reflective measurement model matrix for a specific common-factor, specifying the relevant items of the construct and assigning the relationship of reflective. By definition this construct will be estimated by PLS consistent.

Usage

```
reflective(construct_name, item_names)
```

Arguments

`construct_name`	of construct
`item_names`	returned by the `multi_items` or `single_item` functions

Details

This function conveniently maps reflectively defined measurement items to a construct and is estimated using PLS consistent.

See Also

See `composite`, `constructs`

Examples

```
mobi_mm <- constructs(
  reflective("Image",        multi_items("IMAG", 1:5)),
  reflective("Expectation",  multi_items("CUEX", 1:3)),
  reflective("Quality",      multi_items("PERQ", 1:7)),
  reflective("Value",        multi_items("PERV", 1:2)),
  reflective("Satisfaction", multi_items("CUSA", 1:3)),
  reflective("Complaints",   single_item("CUSCO")),
  reflective("Loyalty",      multi_items("CUSL", 1:3))
)
```

relationships *Structural specification functions for seminr package*

Description

paths creates the structural paths of a PLS SEM model and relationships generates the matrix of paths.

Usage

```
relationships(...)

paths(from,to)
```

Arguments

...	A comma separated list of all the structural relationships in the the model. These paths take the form (from = c(construct_name), to = c(construct_name)).
to	The destination construct of a structural path
from	The source construct of a structural path
paths	The function paths that specifies the source and destination constructs for each of the model's structural paths.

Examples

```
mobi_sm <- relationships(
  paths(from = "Image",        to = c("Expectation", "Satisfaction", "Loyalty")),
  paths(from = "Expectation",  to = c("Quality", "Value", "Satisfaction")),
  paths(from = "Quality",      to = c("Value", "Satisfaction")),
  paths(from = "Value",        to = c("Satisfaction")),
  paths(from = "Satisfaction", to = c("Complaints", "Loyalty")),
  paths(from = "Complaints",   to = "Loyalty")
)
```

report_paths *Functions for reporting the Path Coefficients and R2 of endogenous constructs and for generating a scatterplot matrix of construct scores.*

Description

report_paths generates an easy to read table reporting path coefficients and R2 values for endogenous constructs.plot_scores generates a scatterplot matrix of each construct's scores against every other construct's scores.

Usage

```
report_paths(seminr_model, digits=3)

plot_scores(seminr_model, constructs=NULL)
```

Arguments

seminr_model	The PLS model estimated by seminr. The estimated model returned by the estimate_pls or bootstrap_model methods.
digits	A numeric minimum number of significant digits. If not specified, default is "2".
constructs	a list indicating which constructs to report. If not specified, all constructs are graphed and returned.

Details

These functions generate an easy to read table reporting path coefficients and R2 values for endogenous constructs or a scatterplot matrix of construct scores.

Examples

```
data(mobi)

# seminr syntax for creating measurement model
mobi_mm <- constructs(
  composite("Image",        multi_items("IMAG", 1:5)),
  composite("Expectation",  multi_items("CUEX", 1:3)),
  composite("Value",        multi_items("PERV", 1:2)),
  composite("Satisfaction", multi_items("CUSA", 1:3))
)

#  structural model: note that name of the interactions construct should be
#  the names of its two main constructs joined by a '.' in between.
mobi_sm <- relationships(
  paths(to = "Satisfaction",
        from = c("Image", "Expectation", "Value"))
)

mobi_pls <- estimate_pls(mobi, measurement_model = mobi_mm, structural_model = mobi_sm)
report_paths(mobi_pls)
plot_scores(mobi_pls)
```

---

rho_A *seminr rho_A Function*

Description

The rho_A function calculates the rho_A reliability indices for each construct. For formative constructs, the index is set to 1.

Usage

```
rho_A(seminr_model)
```

Arguments

seminr_model	A seminr_model containing the estimated seminr model.

References

Dijkstra, T. K., & Henseler, J. (2015). Consistent partial least squares path modeling. MIS quarterly, 39(2).

See Also

relationships constructs paths interaction_term bootstrap_model

Examples

```
#seminr syntax for creating measurement model
mobi_mm <- constructs(
             reflective("Image",        multi_items("IMAG", 1:5)),
             reflective("Expectation",  multi_items("CUEX", 1:3)),
             reflective("Quality",      multi_items("PERQ", 1:7)),
             reflective("Value",        multi_items("PERV", 1:2)),
             reflective("Satisfaction", multi_items("CUSA", 1:3)),
             reflective("Complaints",   single_item("CUSCO")),
             reflective("Loyalty",      multi_items("CUSL", 1:3))
           )
#seminr syntax for creating structural model
mobi_sm <- relationships(
  paths(from = "Image",        to = c("Expectation", "Satisfaction", "Loyalty")),
  paths(from = "Expectation",  to = c("Quality", "Value", "Satisfaction")),
  paths(from = "Quality",      to = c("Value", "Satisfaction")),
  paths(from = "Value",        to = c("Satisfaction")),
  paths(from = "Satisfaction", to = c("Complaints", "Loyalty")),
  paths(from = "Complaints",   to = "Loyalty")
)

mobi_pls <- estimate_pls(data = mobi,
                         measurement_model = mobi_mm,
                         structural_model = mobi_sm)

rho_A(mobi_pls)
```

simplePLS *seminr simplePLS Function*

Description

The seminr package provides a natural syntax for researchers to describe PLS structural equation models. seminr is compatible with simplePLS. simplePLS provides the verb for estimating a pls model.

Usage

```
simplePLS(obsData,smMatrix, mmMatrix,inner_weights = path_weighting,
          maxIt=300, stopCriterion=7,measurement_mode_scheme)
```

Arguments

obsData	A dataframe containing the indicator measurement data.
smMatrix	A source-to-target matrix representing the inner/structural model, generated by relationships.
mmMatrix	A source-to-target matrix representing the outer/measurement model, generated by constructs.
inner_weights	A parameter declaring which inner weighting scheme should be used path_weighting is default, alternately path_factorial can be used.
maxIt	The maximum number of iterations to run (default is 300).
stopCriterion	The criterion to stop iterating (default is 7).
measurement_mode_scheme	A named list of constructs and measurement scheme functions

See Also

relationships constructs paths interaction_term estimate_pls bootstrap_model

Examples

```
#seminr syntax for creating measurement model
mobi_mm <- constructs(
             reflective("Image",        multi_items("IMAG", 1:5)),
             reflective("Expectation",  multi_items("CUEX", 1:3)),
             reflective("Quality",      multi_items("PERQ", 1:7)),
             reflective("Value",        multi_items("PERV", 1:2)),
             reflective("Satisfaction", multi_items("CUSA", 1:3)),
             reflective("Complaints",   single_item("CUSCO")),
             reflective("Loyalty",      multi_items("CUSL", 1:3))
           )
#seminr syntax for creating structural model
mobi_sm <- relationships(
  paths(from = "Image",        to = c("Expectation", "Satisfaction", "Loyalty")),
```

```
  paths(from = "Expectation",  to = c("Quality", "Value", "Satisfaction")),
  paths(from = "Quality",      to = c("Value", "Satisfaction")),
  paths(from = "Value",        to = c("Satisfaction")),
  paths(from = "Satisfaction", to = c("Complaints", "Loyalty")),
  paths(from = "Complaints",   to = "Loyalty")
)

mobi_pls <- estimate_pls(data = mobi,
                         measurement_model = mobi_mm,
                         structural_model = mobi_sm)
```

`single_item` *Single-item measurement model specification*

Description

`single_item` specifies a single item name to be assigned to a construct.

Usage

```
single_item(item)
```

Arguments

`item`	Name of item

See Also

See `multi_items`

Examples

```
mobi_mm <- constructs(
  composite("Image",        multi_items("IMAG", 1:5), weights = correlation_weights),
  composite("Expectation",  multi_items("CUEX", 1:3), weights = mode_A),
  composite("Quality",      multi_items("PERQ", 1:7), weights = regression_weights),
  composite("Value",        single_item("PERV1"))
)
```

two_stage	*Creates an interaction measurement item using a two-stage approach. The two-stage procedure for both PLS and CBSEM models estimates construct scores in the first stage, and uses them to produce a single-item product item for the interaction term in the second stage. For a PLS model, the first stage uses PLS to compute construct scores. For a CBSEM model, the first stage uses a CFA to produce ten Berge construct scores.*

Description

Creates an interaction measurement item using a two-stage approach. The two-stage procedure for both PLS and CBSEM models estimates construct scores in the first stage, and uses them to produce a single-item product item for the interaction term in the second stage. For a PLS model, the first stage uses PLS to compute construct scores. For a CBSEM model, the first stage uses a CFA to produce ten Berge construct scores.

Usage

```
# two stage approach as per Henseler & Chin (2010):
 two_stage(iv, moderator, weights)
```

Arguments

iv	The independent variable that is subject to moderation.
moderator	The moderator variable.
weights	is the relationship between the items and the interaction terms. This can be specified as `correlation_weights` or `mode_A` for correlation weights (Mode A) or as `regression_weights` or `mode_B` for regression weights (Mode B). Default is correlation weights.

References

Henseler & Chin (2010), A comparison of approaches for the analysis of interaction effects between latent variables using partial least squares path modeling. Structural Equation Modeling, 17(1),82-109.

Examples

```
data(mobi)

# seminr syntax for creating measurement model
mobi_mm <- constructs(
  composite("Image",        multi_items("IMAG", 1:5)),
  composite("Expectation",  multi_items("CUEX", 1:3)),
  composite("Value",        multi_items("PERV", 1:2)),
  composite("Satisfaction", multi_items("CUSA", 1:3)),
  interaction_term(iv = "Image", moderator = "Expectation", method = two_stage)
```

```
)

#  structural model: note that name of the interactions construct should be
#  the names of its two main constructs joined by a '*' in between.
mobi_sm <- relationships(
  paths(to = "Satisfaction",
        from = c("Image", "Expectation", "Value",
                 "Image*Expectation"))
)

# PLS example:
mobi_pls <- estimate_pls(mobi, mobi_mm, mobi_sm)
summary(mobi_pls)

# CBSEM example:
mobi_cbsem <- estimate_cbsem(mobi, as.reflective(mobi_mm), mobi_sm)
summary(mobi_cbsem)
```

Index