Package 'seminr'

October 18, 2018

Type Package

Title Domain-Specific Language for Building PLS Structural Equation Models

Version 0.5.3

Date 2018-10-18

Description A powerful, easy to write and easy to modify syntax for specifying and estimating Partial Least Squares (PLS) path models allowing for the latest estimation methods for Consistent PLS as per Dijkstra & Henseler (2015, MISQ 39(2): 297-316), adjusted interactions as per Henseler & Chin (2010) <doi:10.1080/10705510903439003> and bootstrapping utilizing parallel processing as per Hair et al. (2017, ISBN:978-1483377445).

Imports parallel

License GPL-3

Depends R (>= 3.1.0)

LazyData TRUE

RoxygenNote 6.0.1

Suggests knitr, testthat, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Author Soumya Ray [aut, ths], Nicholas Danks [aut, cre], Juan Manuel Velasquez Estrada [aut]

Maintainer Nicholas Danks <nicholasdanks@hotmail.com>

Repository CRAN

Date/Publication 2018-10-18 09:10:02 UTC

R topics documented:

bootstrap_model 2

composite . . . 3
constructs . . . 4
estimate_pls . . . 5
interactions . . . 6
interaction_ortho . . . 7
interaction_scaled . . . 8
mobi . . . 10
mode_A . . . 11
mode_B . . . 12
multi_items . . . 12
path_factorial . . . 13
path_weighting . . . 13
PLSc . . . 14
reflective . . . 15
relationships . . . 16
report_paths . . . 17
rho_A . . . 18
simplePLS . . . 19
single_item . . . 20
SRMR . . . 21

Index **22**

`bootstrap_model` *seminr bootstrap_model Function*

Description

The `seminr` package provides a natural syntax for researchers to describe PLS structural equation models. `seminr` is compatible with simplePLS. `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 (e.g., `simplePLS`).

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 interactions

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 constructs must be created after the measurement model is defined
mobi_xm <- interactions(
  interaction_ortho("Image", "Expectation"),
  interaction_ortho("Image", "Value")
)

# 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,
                             interactions = mobi_xm,
                             structural_model = mobi_sm)

# Load data, assemble model, and bootstrap using simplePLS
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)
)
```

---

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))
)
```

`estimate_pls` *seminr estimate_pls() function*

Description

The `seminr` package provides a natural syntax for researchers to describe PLS structural equation models. `seminr` is compatible with semPLS, simplePLS and matrixPLS, meaning that once a model is specified, it can be used across models for comparison.

Usage

```
estimate_pls(data, measurement_model, interactions=NULL, 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`.
`interactions`	An object of type `interactions` as generated by the `interactions` method. Default setting is `NULL` and can be excluded for models with no interactions.

`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` `interactions` `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)
```

---

interactions *Interaction Functions*

Description

`interactions` creates interaction measurement items by multipying all combination of construct items.

Usage

```
interactions(...)
```

Arguments

...	Interaction Combinations as generated by the `interaction_scaled` or `interaction_ortho` methods.

Details

This function automatically generates interaction measurement items for a PLS SEM.

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))
)
mobi_xm <- interactions(
  interaction_ortho("Image", "Expectation"),
  interaction_ortho("Image", "Value")
)

#  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_xm, mobi_sm)
summary(mobi_pls)
```

`interaction_ortho` `interaction_ortho` *creates interaction measurement items by using the orthogonalized approach..*

Description

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

Usage

```
# orthogonalization approach as per Henseler & CHin (2010):
 interaction_ortho(construct1, construct2)
```

Arguments

construct1	The first construct which is subject to the interaction.
construct2	The second construct which is subject to the interaction.

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))
)
mobi_xm <- interactions(
  interaction_ortho("Image", "Expectation"),
  interaction_ortho("Image", "Value")
)

#  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_xm, mobi_sm)
summary(mobi_pls)
```

`interaction_scaled` `interaction_scaled` *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):
 interaction_scaled("construct1", "construct2")
```

Arguments

construct1	The first construct which is subject to the interaction.
construct2	The second construct which is subject to the interaction.

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))
)
mobi_xm <- interactions(
  interaction_scaled("Image", "Expectation"),
  interaction_scaled("Image", "Value")
)

#  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_xm, mobi_sm)
summary(mobi_pls)
```

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 `simplePLS`.

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 simplePLS.

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)
)
```

`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 `simplePLS`.
`dependant`	is the vector of dependant constructs in the model.
`paths_matrix`	is the matrix of estimated path coefficients estimated by `simplePLS`.

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 simplePLS.
dependant	is the vector of dependant constructs in the model.
paths_matrix	is the matrix of estimated path coefficients estimated by simplePLS.

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 interactions 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)
```

`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 for use by PLS modelling packages such as semPLS and simplePLS.

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 simplePLS 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, 39(X).

See Also

relationships constructs paths interactions 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 interactions 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"), weights = mode_A)
)
```

SRMR *Function to calculate the SRMR of a model*

Description

SRMR calculate the saturated and estimated SRMR of a estimated SEMinR model.

Usage

```
SRMR(seminr_model)
```

Arguments

seminr_model	is the PLS model estimated by SEMinR to calculate SRMR for.

References

Henseler, J., Hubona, G., & Ray, P. A. (2016). Using PLS path modeling in new technology research: updated guidelines. Industrial management & data systems, 116(1), 2-20.

Index

∗Topic **datasets**
mobi, 10

bootstrap_model, 2, *6*, *14*, *18, 19*

composite, 3, *5*, *15*
constructs, *3, 4*, 4, *6*, *14, 15*, *18, 19*
correlation_weights *(*mode_A*)*, 11

estimate_pls, 5, *19*

interaction_ortho, 7
interaction_scaled, 8
interactions, *3*, *6*, 6, *14*, *18, 19*

mobi, 10
mode_A, 11
mode_A, *(*mode_A*)*, 11
mode_B, 12
mode_B, *(*mode_B*)*, 12
multi_items, 12, *20*

path_factorial, 13
path_weighting, 13
paths, *3*, *6*, *14*, *18, 19*
paths *(*relationships*)*, 16
plot_scores *(*report_paths*)*, 17
PLSc, 14

reflective, *4, 5*, 15
regression_weights *(*mode_B*)*, 12
relationships, *3*, *6*, *14*, 16, *18, 19*
report_paths, 17
rho_A, 18

simplePLS, 19
single_item, *12*, 20
SRMR, 21