

Package ‘seminr’

May 7, 2019

Type Package

Title Domain-Specific Language for Building PLS Structural Equation Models

Version 0.7.0

Date 2019-05-07

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 6.1.1

Suggests knitr, testthat, rmarkdown

VignetteBuilder knitr

Encoding UTF-8

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 2019-05-07 08:50:02 UTC

R topics documented:

bootstrap_model	2
composite	4
confidence_interval	5
constructs	6
estimate_pls	7
interactions	8
interaction_ortho	9
interaction_scaled	10
mobi	11
mode_A	13
mode_B	14
multi_items	14
path_factorial	15
path_weighting	15
PLSc	16
reflective	17
relationships	18
report_paths	19
rho_A	20
simplePLS	21
single_item	22
SRMR	23
Index	24

bootstrap_model	<i>seminr bootstrap_model Function</i>
-----------------	--

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

<code>seminr_model</code>	A fully estimated model with associated data, measurement model and structural model
<code>nboot</code>	A parameter specifying the number of bootstrap iterations to perform, default value is 500. If 0 then no bootstrapping is performed.
<code>cores</code>	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 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
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

<code>boot_seminr_model</code>	A bootstrapped model returned by the <code>bootstrap_model</code> function.
<code>from</code>	A parameter specifying the antecedent composite for the path.
<code>to</code>	A parameter specifying the outcome composite for the path.
<code>through</code>	A parameter to specify the mediator for the path. Default is <code>NULL</code> .
<code>alpha</code>	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))
)

```

estimate_pls	<i>seminr estimate_pls() function</i>
--------------	---------------------------------------

Description

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

Usage

```

estimate_pls(data, measurement_model, interactions=NULL, structural_model,
             inner_weights = path_weighting)

```

Arguments

<code>data</code>	A dataframe containing the indicator measurement data.
<code>measurement_model</code>	A source-to-target matrix representing the outer/measurement model, generated by <code>constructs</code> .
<code>interactions</code>	An object of type <code>interactions</code> as generated by the <code>interactions</code> method. Default setting is <code>NULL</code> and can be excluded for models with no interactions.
<code>structural_model</code>	A source-to-target matrix representing the inner/structural model, generated by <code>relationships</code> .
<code>inner_weights</code>	A parameter declaring which inner weighting scheme should be used <code>path_weighting</code> is default, alternately <code>path_factorial</code> can be used.

See Also

[relationships](#) [constructs](#) [paths](#) [interactions](#) [bootstrap_model](#)

Examples

```

mobi <- mobi

#semnr 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))
)

#semnr 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 multiplying 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	<i>interaction_ortho creates interaction measurement items by using the orthogonalized approach..</i>
-------------------	---

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 provider" 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

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

mode_B	<i>Outer weighting scheme functions to estimate construct weighting.</i>
--------	--

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	<i>Multi-items measurement model specification</i>
-------------	--

Description

multi_items creates a vector of measurement 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 suffixes for the items
...	Additional Item names and numbers

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 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](#) [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))
    )
#semirn 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")
)

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

PLSc(semirn_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.

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	<i>Functions for reporting the Path Coefficients and R2 of endogenous constructs and for generating a scatterplot matrix of construct scores.</i>
--------------	---

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

*semnr 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(semnr_model)
```

Arguments

semnr_model A semnr_model containing the estimated semnr 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

```
#semnr 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))
)

#semnr 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

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

See Also

[relationships](#) [constructs](#) [paths](#) [interactions](#) [estimate_pls](#) [bootstrap_model](#)

Examples

```
#semnr 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))
)

#semnr 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"))  
)
```

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, [11](#)

bootstrap_model, [2](#), [5](#), [7](#), [16](#), [20](#), [21](#)

composite, [4](#), [6](#), [17](#)

confidence_interval, [5](#)

constructs, [3](#), [4](#), [6](#), [7](#), [16](#), [17](#), [20](#), [21](#)

correlation_weights (mode_A), [13](#)

estimate_pls, [7](#), [21](#)

interaction_ortho, [9](#)

interaction_scaled, [10](#)

interactions, [3](#), [7](#), [8](#), [16](#), [20](#), [21](#)

mobi, [11](#)

mode_A, [13](#)

mode_A, (mode_A), [13](#)

mode_B, [14](#)

mode_B, (mode_B), [14](#)

multi_items, [14](#), [22](#)

path_factorial, [15](#)

path_weighting, [15](#)

paths, [3](#), [7](#), [16](#), [20](#), [21](#)

paths (relationships), [18](#)

plot_scores (report_paths), [19](#)

PLSc, [16](#)

reflective, [4](#), [6](#), [17](#)

regression_weights (mode_B), [14](#)

relationships, [3](#), [7](#), [16](#), [18](#), [20](#), [21](#)

report_paths, [19](#)

rho_A, [20](#)

simplePLS, [21](#)

single_item, [14](#), [22](#)

SRMR, [23](#)