

Package ‘seminr’

January 27, 2026

Type Package

Title Building and Estimating Structural Equation Models

Version 2.4.0

Date 2026-01-26

Description A powerful, easy to use 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 (Ray, Danks, and Valdez 2021 <[doi:10.2139/ssrn.3900621](https://doi.org/10.2139/ssrn.3900621)>).

Imports parallel, lavaan, glue, knitr, testthat, rmarkdown, DiagrammeR (>= 1.0.6), DiagrammeRsvg (>= 0.1), webp, stats

License GPL-3

Depends R (>= 4.1.0)

LazyData TRUE

URL <https://github.com/sem-in-r/seminr>

BugReports <https://github.com/sem-in-r/seminr/issues>

RoxygenNote 7.3.2

Enhances rsvg (>= 2.1), semPlot, vdiff, seminrExtras

VignetteBuilder knitr

Encoding UTF-8

NeedsCompilation no

Author Soumya Ray [aut, ths],
Nicholas Patrick Danks [aut, cre],
André Calero Valdez [aut],
Juan Manuel Velasquez Estrada [ctb],
James Uanhoro [ctb],
Johannes Nakayama [ctb],
Lilian Koyan [ctb],
Laura Burbach [ctb],
Arturo Heynar Cano Bejar [ctb],
Susanne Adler [ctb]

Maintainer Nicholas Patrick Danks <nicholasdanks@hotmail.com>

Repository CRAN

Date/Publication 2026-01-27 13:00:02 UTC

Contents

as.reflexive	4
as.reflexive.construct	5
as.reflexive.interaction	6
as.reflexive.measurement_model	7
associations	8
bootstrap_model	8
boot_paths_df	10
browse_plot	11
check_test_plot	12
composite	12
compute_itcriteria_weights	13
constructs	14
corp_rep_data	15
corp_rep_data2	17
cor_rsq	19
csem2seminr	19
df_xtab_matrix	20
dot_component_mm	21
dot_graph	21
dot_graph_htmt	24
dot_subcomponent_mm	25
edge_template_default	26
edge_template_minimal	26
esc_node	26
estimate_cbsem	27
estimate_cfa	29
estimate_lavaan_ten_berge	31
estimate_pls	32
estimate_pls_mga	34
extract_bootstrapped_values	35
extract_htmt_nodes	36
extract_mm_coding	36
extract_mm_edges	37
extract_mm_edge_value	37
extract_mm_nodes	38
extract_sm_nodes	38
format_endo_node_label	39
format_exo_node_label	39
fSquared	40
get_construct_element_size	41
get_construct_type	41

get_manifest_element_size	42
get_mm_edge_style	42
get_mm_node_shape	43
get_mm_node_style	43
get_sm_node_shape	44
get_value_dependent_mm_edge_style	44
get_value_dependent_sm_edge_style	45
higher_composite	45
higher_reflective	46
influencer_data	47
interaction_term	48
is_sink	49
item_errors	50
mean_replacement	50
mobi	51
mode_A	52
mode_B	53
mode_plsc	54
multi_items	54
node_endo_template_default	55
node_exo_template_default	55
orthogonal	56
path_factorial	57
path_weighting	58
plot.reliability_table	58
plot.seminr_model	59
plot_htmt	60
plot_interaction	61
PLSc	62
predict_DA	63
predict_EA	63
predict_pls	64
print.seminr_pls_mga	66
product_indicator	66
reflective	68
relationships	69
report_missing	70
report_paths	70
rerun	71
rerun.pls_model	72
rhoC_AVE	73
rho_A	73
save_plot	74
seminr_theme_academic	76
seminr_theme_create	76
seminr_theme_dark	81
seminr_theme_get	82
seminr_theme_smart	83

simplePLS	84
single_item	86
slope_analysis	87
specific_effect_significance	88
specify_model	89
standardize_safely	90
total_indirect_ci	91
two_stage	92
unit_weights	93
Index	95

as.reflective	<i>Converts all constructs of a measurement model, or just a single construct into reflective factors.</i>
----------------------	--

Description

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

Usage

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

Arguments

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

Value

A list of reflective constructs.

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 contract of a measurement model into a reflective factor.

Description

Converts a contract 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.

Value

A list of reflective constructs.

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.

Value

A list of reflective constructs.

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 constructs of a measurement model, or just a single construct into reflective factors.

Description

Converts all constructs 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.

Value

A list of reflective constructs.

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

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
-----	---

Value

A matrix of items that covary.

Examples

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

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

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.

Value

A list of the estimated parameters for the bootstrapped model including:

boot_paths	An array of the 'nboot' estimated bootstrap sample path coefficient matrices.
boot_loadings	An array of the 'nboot' estimated bootstrap sample item loadings matrices.
boot_weights	An array of the 'nboot' estimated bootstrap sample item weights matrices.
boot_HTMT	An array of the 'nboot' estimated bootstrap sample model HTMT matrices.
boot_total_paths	An array of the 'nboot' estimated bootstrap sample model total paths matrices.
paths_descriptives	A matrix of the bootstrap path coefficients and standard deviations.
loadings_descriptives	A matrix of the bootstrap item loadings and standard deviations.
weights_descriptives	A matrix of the bootstrap item weights and standard deviations.
HTMT_descriptives	A matrix of the bootstrap model HTMT and standard deviations.
total_paths_descriptives	A matrix of the bootstrap model total paths and standard deviations.

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)

```

boot_paths_df

Return all path bootstraps as a long dataframe. Columns of the dataframes are specified paths and rows are the estimated coefficients for the paths at each bootstrap iteration.

Description

Return all path bootstraps as a long dataframe. Columns of the dataframes are specified paths and rows are the estimated coefficients for the paths at each bootstrap iteration.

Usage

```
boot_paths_df(pls_boot)
```

Arguments

pls_boot	bootstrapped PLS model
----------	------------------------

Examples

```

data(mobi)

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

```

```
composite("Satisfaction", multi_items("CUSA", 1:3))
)

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

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

pls_boot <- bootstrap_model(seminr_model = pls_model,
                            nboot = 50, cores = 2, seed = NULL)

boot_paths_df(pls_boot)
```

browse_plot*Open Edotor graphViz Website with the preloaded in the Browser*

Description

Open Edotor graphViz Website with the preloaded in the Browser

Usage

```
browse_plot(model, theme = seminr_theme_get())
```

Arguments

model	A SEMinR Model
theme	An optional SEMinR theme

Examples

```
## Not run:
browse_plot(model)

## End(Not run)
```

check_test_plot	<i>A function to create regression plots (maybe not needed?)</i>
-----------------	--

Description

A function to create regression plots (maybe not needed?)

Usage

```
check_test_plot(plot, title, plot_dir = "regression_plots", refresh = FALSE)
```

Arguments

plot	the plot
title	a unique title
plot_dir	optional directory name
refresh	whether to refresh all test cases

Value

TRUE if plots were the same, FALSE if they did not exist, or failed

composite	<i>Composite construct measurement model specification</i>
-----------	--

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 <code>multi_items</code> or <code>single_item</code> functions
weights	is the relationship between the construct and its items. This can be specified as <code>correlation_weights</code> or <code>mode_A</code> for correlation weights (Mode A) or as <code>regression_weights</code> or <code>mode_B</code> 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.

Value

A vector of the indicators for a composite.

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("PERO", 1:7), weights = regression_weights),
  composite("Value",          multi_items("PERV", 1:2), weights = mode_B)
)
```

compute_itcriteria_weights

Function to calculate Akaike weights for IT Criteria

Description

Function to calculate Akaike weights for IT Criteria

Usage

```
compute_itcriteria_weights(vector_of_itcriteria)
```

Arguments

`vector_of_itcriteria`

This argument is a vector consisting of the IT criterion estimated value for each model.

constructs	<i>Measurement functions</i>
------------	------------------------------

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

Value

A list of constructs, their indicators and estimation technique (SEMinR measurement model).

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))
)
```

Description

The data set is used as measurement instrument for corporate reputation.

Usage

corp_rep_data

Format

A data frame with 344 rows and 41 variables:

serviceprovider A categorical variable for the service provider: 1, 2, 3, or 4.

servicetype A categorical variable for the service type: 1=Prepaid plan (n=125); 2=Contract plan (n=219).

csor_1 The company behaves in a socially conscious way.

csor_2 The company is forthright in giving information to the public.

csor_3 The company has a fair attitude toward competitors.

csor_4 The company is concerned about the preservation of the environment.

csor_5 The company is not only concerned about the profits.

csor_global Please assess the extent to which the company acts in socially conscious ways 0 (not at all) to 7 (definitely).

attr_1 The company is successful in attracting high-quality employees.

attr_2 I could see myself working at the company.

attr_3 I like the physical appearance of the company/buildings/shops, etc.

attr_global Please assess the company's overall attractiveness; 0=very low; 7=very high.

perf_1 The company is a very well managed company.

perf_2 The company is an economically stable company.

perf_3 The business risk for the company is modest compared to its competitors.

perf_4 The company has growth potential.

perf_5 The company has a clear vision about the future of the company.

perf_global Please assess the company's overall performance; 0=very low; 7=very high.

qual_1 The products/services offered by the company are of high quality.

qual_2 The company is an innovator, rather than an imitator with respect to industry.

qual_3 The company's services/products offer good quality for money.

qual_4 The services the company offered are good.

qual_5 Customer concerns are held in high regard at the company.

qual_6 The company is a reliable partner for customers.

qual_7 The company is a trustworthy company.

qual_8 I have a lot of respect for the company.

qual_global Please assess the overall quality of the company's activities; 0=very low; 7=very high.

like_1 The company is a company that I can better identify with than other companies.

like_2 The company is a company that I would regret more not having if it no longer existed than other companies.

like_3 I regard the company as a likeable company.

comp_1 The company is a top competitor in its market.

comp_2 As far as I know, the company is recognized worldwide.

comp_3 I believe that the company performs at a premium level.

cusl_1 I would recommend the company to friends and relatives.

cusl_2 If I had to choose again, I would choose the company as my mobile phone services provider.

cusl_3 I will remain a customer of the company in the future.

cusa I am satisfied with company.

switch_1 It takes me a great deal of time to switch to another company.

switch_2 It costs me too much to switch to another company.

switch_3 It takes a lot of effort to get used to a new company with its specific "rules" and practices.

switch_4 In general, it would be a hassle switching to another company.

Details

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

References

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

Examples

```
data("corp_rep_data")
```

Description

The data set is used as measurement instrument for corporate reputation.

Usage

```
corp_rep_data2
```

Format

A data frame with 347 rows and 41 variables:

servicetype A categorical variable for the service type: 1=Postpaid plan; 2=Prepaid plan.

serviceprovider A categorical variable for the service provider: 1, 2, 3, or 4.

cusa If you consider your experiences with "company", how satisfied are you with "company"?

cusl_1 I would recommend "the company" to friends and relatives.

cusl_2 If I had to choose again, I would choose "the company" as my mobile phone services provider.

cusl_3 I will remain a customer of "the company" in the future.

qual_1 The products/services offered by "the company" are of high quality.

qual_2 "The company" is an innovator, rather than an imitator with respect to the mobile phone service industry.

qual_3 "The company's" services/products offer good quality for money.

qual_4 The services "the company" offers are good.

qual_5 Customer concerns are held in high regard at "the company".

qual_6 "The company" is a reliable partner for customers.

qual_7 "The company" is a trustworthy company.

qual_8 I have a lot of respect for "the company".

perf_1 "The company" is a very well managed company.

perf_2 "The company" is an economically stable company.

perf_3 The business risk of "the company" is reasonable compared to its competitors.

perf_4 The growth of "the company" is promising.

perf_5 "The company" has a clear vision about the future of the company.

csor_1 "The company" behaves in a socially conscious way.

csor_2 "The company" is honest in giving information to the public.

csor_3 "The company" competes fairly in the industry.

csor_4 "The company" cares for the preservation of the environment.

csor_5 "The company" is doing more than just making profits.

attr_1 "The company" is successful in attracting high-quality employees.

attr_2 I could see myself working at "the company".

attr_3 I like the physical appearance of "the company" (company/buildings/shops, etc.).

comp_1 "The company" is a top competitor in its market.

comp_2 As far as I know, "the company" is recognized worldwide.

comp_3 I believe that "the company" performs at a premium level.

like_1 "The company" is a company that I can better identify with than other companies.

like_2 When comparing with other companies, "The company" is the company I would regret more if it no longer existed.

like_3 I regard "the company" as a likeable company.

qual_global Please assess the general quality of "the company".

perf_global Please assess the general performance of "the company".

csor_global Please assess the extent to which "the company" acts in socially conscious ways.

attr_global Please assess the attractiveness of "the company".

switch_1 It takes me a great deal of time to switch to another mobile phone services provider.

switch_2 It costs me too much to switch to another mobile phone services provider.

switch_3 It takes a lot of effort to get used to a new mobile phone services provider with its specific "rules" and practices.

switch_4 In general, it would be a hassle switching to another mobile phone services provider.

Details

The data frame `mobi` contains the observed data for the model specified by Corporate Reputation.

References

Sarstedt, M., Hair Jr, J. F., Cheah, J. H., Becker, J. M., & Ringle, C. M. (2019). How to specify, estimate, and validate higher-order constructs in PLS-SEM. *Australasian Marketing Journal (AMJ)*, 27(3), 197-211.

Examples

```
data("corp_rep_data2")
```

`cor_rsq`

Returns R-sq of a dv given correlation matrix of ivs, dv cors <- cbsem_summary\$descriptives\$correlations\$constructs cor_rsq(cors, dv_name = "Value", iv_names = c("Image", "Quality"))

Description

Returns R-sq of a dv given correlation matrix of ivs, dv cors <- cbsem_summary\$descriptives\$correlations\$constructs cor_rsq(cors, dv_name = "Value", iv_names = c("Image", "Quality"))

Usage

```
cor_rsq(cor_matrix, dv_name, iv_names)
```

Arguments

<code>cor_matrix</code>	A correlation matrix that includes ivs and dv
<code>dv_name</code>	Character string of dependent variable
<code>iv_names</code>	Vector of character strings for independent variables

`csem2seminr`

seminr csem2seminr() function

Description

Converts lavaan syntax for composite models used by cSEM package to SEMinR model specifications

Usage

```
csem2seminr(lav_syntax)
```

Arguments

<code>lav_syntax</code>	A string specifying the composite model measurement and structure using lavaan syntax
-------------------------	---

Value

A SEMinR model.

See Also

[estimate_pls](#)

Examples

```

lav_syntax <- '
# Composite model
Image <~ IMAG1 + IMAG2 + IMAG3 + IMAG4 + IMAG5
Expectation <~ CUEX1 + CUEX2 + CUEX3
Value <~ PERV1 + PERV2
Satisfaction <~ CUSA1 + CUSA2 + CUSA3

# Structural model
Satisfaction ~ Image + Expectation + Value '

csem_model <- estimate_pls(mobi, model = csem2seminr(lav_syntax))

```

df_xtab_matrix	<i>Cross-tabulates columns of a dataframe into a matrix with NAs for unspecified pairs</i>
----------------	--

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 <code>data.frame</code> 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

Value

A cross-tabulated matrix matrix with NAs for unspecified pairs.

dot_component_mm	<i>Generates the dot code for the measurement model</i>
------------------	---

Description

Generates the dot code for the measurement model

Usage

```
dot_component_mm(model, theme)
```

Arguments

model	the model to use
theme	the theme to use

dot_graph	<i>Generate a dot graph from various SEMinR models</i>
-----------	--

Description

With the help of the DiagrammeR package this dot graph can then be plotted in various in RMark-down, shiny, and other contexts. Depending on the type of model, different parameters can be used.

For a full description of parameters for lavaan models see `semPaths` method in the `semPlot` package.

Usage

```
dot_graph(model, title = "", theme = NULL, ...)

## S3 method for class 'cfa_model'
dot_graph(
  model,
  title = "",
  theme = NULL,
  what = "std",
  whatLabels = "std",
  ...
)

## S3 method for class 'cbsem_model'
dot_graph(
  model,
  title = "",
  theme = NULL,
```

```

what = "std",
whatLabels = "std",
...
)

## S3 method for class 'measurement_model'
dot_graph(model, title = "", theme = NULL, ...)

## S3 method for class 'structural_model'
dot_graph(model, title = "", theme = NULL, ...)

## S3 method for class 'specified_model'
dot_graph(
  model,
  title = "",
  theme = NULL,
  measurement_only = FALSE,
  structure_only = FALSE,
  ...
)

## S3 method for class 'boot_seminr_model'
dot_graph(
  model,
  title = "",
  theme = NULL,
  measurement_only = FALSE,
  structure_only = FALSE,
  ...
)

## S3 method for class 'pls_model'
dot_graph(
  model,
  title = "",
  theme = NULL,
  measurement_only = FALSE,
  structure_only = FALSE,
  ...
)

```

Arguments

model	Model created with <code>seminr</code> .
title	An optional title for the plot
theme	Theme created with <code>seminr_theme_create</code> .
...	Unused
what	The metric to use for edges ("path", "est", "std", "eq", "col")

```

whatLabels      The metric to use for edge labels
measurement_only
                  Plot only measurement part
structure_only   Plot only structure part

```

Details

Current limitations: - Only plots PLS Models - no higher order constructs

Value

The path model as a formatted string in dot language.

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)

# adapt nboot for better results
mobi_boot <- bootstrap_model(mobi_pls, nboot = 20, cores = 1)
# generate dot-Notation
res <- dot_graph(mobi_pls, title = "PLS-Model plot")

## Not run:
DiagrammeR::grViz(res)
## End(Not run)

# generate dot-Notation
res <- dot_graph(mobi_boot, title = "Bootstrapped PLS-Model plot")

```

```

## Not run:
DiagrammeR::grViz(res)
## End(Not run)

# -----
# Example for plotting a 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))
)
dot_graph(mobi_mm, title = "Preview measurement model")
# -----
# Example for plotting a 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")
)
res <- dot_graph(mobi_sm, title = "Preview structural model")

## Not run:
DiagrammeR::grViz(res)

## End(Not run)

```

dot_graph_htmt

Creates a dot string with a network graph of constructs based on HTMT measures

Description

Using a bootstrapped model this functions shows which constructs show insufficient discriminant validity.

Usage

```
dot_graph_htmt(
  model,
  title = "HTMT Plot",
  theme = seminr::seminr_theme_get(),
  htmt_threshold = 1,
```

```

  omit_threshold_edges = TRUE,
  use_ci = FALSE
)

```

Arguments

model	A bootstrapped PLS-Model
title	Optional title over the plot.
theme	Optional theme to use for plotting
htmt_threshold	The threshold to use under which constructs are highlighted (default = 1.0)
omit_threshold_edges	Whether or not to omit constructs that have low HTMT values (default = TRUE)
use_ci	Whether or not to rely on the upper threshold of the CI instead of the bootstrapped mean (default = FALSE)

Value

Returns a dot string of the plot

dot_subcomponent_mm *generates the dot code for a subgraph (per construct)*

Description

generates the dot code for a subgraph (per construct)

Usage

```
dot_subcomponent_mm(index, model, theme)
```

Arguments

index	the index of the construct
model	the model to use
theme	the theme to use

edge_template_default *The default template for labeling bootstrapped edges*

Description

The default template for labeling bootstrapped edges

Usage

```
edge_template_default()
```

Value

The template string

edge_template_minimal *A minimal template for labeling bootstrapped edges that only shows the bootstrapped mean value*

Description

A minimal template for labeling bootstrapped edges that only shows the bootstrapped mean value

Usage

```
edge_template_minimal()
```

Value

The template string

esc_node *Wrap a text in single quotes*

Description

Wrap a text in single quotes

Usage

```
esc_node(x)
```

Arguments

x	a character string
---	--------------------

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

Description

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

Usage

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

Arguments

<code>data</code>	A dataframe containing the indicator measurement data. The entire CBSEM model can be specified in one of three ways: The pair of measurement and structural models, along associated items, can optionally be specified as separate model components
<code>measurement_model</code>	An optional <code>measurement_model</code> object representing the outer/measurement model, as generated by <code>constructs</code> . Note that only reflective constructs are supported for CBSEM models, though a composite measurement model can be converted into a reflective one using <code>as.reflective</code> .
<code>structural_model</code>	An optional <code>smMatrix</code> object representing the inner/structural model, as generated by <code>relationships</code> .
<code>item_associations</code>	An item-to-item matrix representing error covariances that are freed for estimation. This matrix is created by <code>associations()</code> , or defaults to <code>NULL</code> (no inter-item associations). The combination of measurement and structural models and inter-item associations can also be specified as a single <code>specified_model</code> object. Note that any given model components (<code>measurement_model</code> , <code>structural_model</code> , <code>item_associations</code>) will override components in the fully specified model
<code>model</code>	An optional <code>specified_model</code> object containing both the the outer/measurement and inner/structural models, along with any inter-item associations, as generated by <code>specify_model</code> . The entire model can also be specified in Lavaan syntax (this overrides any other specifications)
<code>lavaan_model</code>	Optionally, a single character string containing the relevant model specification in lavaan syntax. Any further optional parameters to alter the estimation method:

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.

Value

A list of the estimated parameters for the CB-SEM model including:

data	A matrix of the data upon which the model was estimated.
measurement_model	The SEMinR measurement model specification.
factor_loadings	The matrix of estimated factor loadings.
associations	A matrix of model variable associations.
mmMatrix	A Matrix of the measurement model relations.
smMatrix	A Matrix of the structural model relations.
constructs	A vector of the construct names.
construct_scores	A matrix of the estimated construct scores for the CB-SEM model.
item_weights	A matrix of the estimated CFA item weights.
lavaan_model	The lavaan model syntax equivalent of the SEMinR model.
lavaan_output	The raw lavaan output generated after model 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 = NULL, item_associations=NULL,
             model = NULL, lavaan_model = NULL, estimator="MLR", ...)
```

Arguments

data	A data frame containing the indicator measurement data. The entire CBSEM model can be specified in one of three ways: The pair of measurement and structural models, along associated items, can optionally be specified as separate model components
measurement_model	An optional <code>measurement_model</code> object representing the outer/measurement model, as generated by <code>constructs</code> . Note that only reflective constructs are supported for CBSEM models, though a composite measurement model can be converted into a reflective one using <code>as.reflective</code> .
item_associations	An item-to-item matrix representing error covariances that are freed for esti- mation. This matrix is created by <code>associations()</code> , or defaults to NULL (no inter-item associations).

	The combination of measurement and structural models and inter-item associations can also be specified as a single <code>specified_model</code> object. Note that any given model components (<code>measurement_model</code> , <code>structural_model</code> , <code>item_associations</code>) will override components in the fully specified model
<code>model</code>	An optional <code>specified_model</code> object containing both the outer/measurement and inner/structural models, along with any inter-item associations, as generated by <code>specify_model</code> . The entire model can also be specified in Lavaan syntax (this overrides any other specifications)
<code>lavaan_model</code>	Optionally, a single character string containing the relevant model specification in lavaan syntax.
<code>estimator</code>	Any further optional parameters to alter the estimation method: 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 <code>lavaan::sem</code> during estimation.

Value

A list of the estimated parameters for the CFA model including:

<code>data</code>	A matrix of the data upon which the model was estimated.
<code>measurement_model</code>	The SEMinR measurement model specification.
<code>construct_scores</code>	A matrix of the estimated construct scores for the CB-SEM model.
<code>item_weights</code>	A matrix of the estimated CFA item weights.
<code>lavaan_model</code>	The lavaan model syntax equivalent of the SEMinR model.
<code>lavaan_output</code>	The raw lavaan output generated after model 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 cbsem 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_output)
tb$scores
tb$weights
```

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

Description

Estimates a pair of measurement and structural models using PLS-SEM, with optional estimation methods

Usage

```
estimate_pls(data,
             measurement_model = NULL, structural_model = NULL, model = NULL,
             inner_weights = path_weighting,
             missing = mean_replacement,
             missing_value = NA,
             maxIt = 300,
             stopCriterion = 7,
             assess_syntax = FALSE)
```

Arguments

data	A data frame containing the manifest measurement items in named columns. The pair of measurement and structural models can optionally be specified as separate model objects
measurement_model	An optional measurement_model object representing the outer/measurement model, as generated by constructs.
structural_model	An optional smMatrix object representing the inner/structural model, as generated by relationships. The pair of measurement and structural models can also be specified as a single specified_model object
model	An optional specified_model object containing both the outer/measurement and inner/structural models, as generated by specify_model.
inner_weights	Function that implements inner weighting scheme: path_weighting (default) or path_factorial can be used.
missing	Function that replaces missing values. mean_replacement is default. na.omit removes incomplete cases.
missing_value	Value in dataset that indicates missing values. NA is used by default.
maxIt	A parameter that specifies that maximum number of iterations when estimating the PLS model. Default value is 300.
stopCriterion	A parameter specifying the stop criterion for estimating the PLS model. Default value is 7.
assess_syntax	A parameter that specifies whether the measurement and structural model should be assessed for errors. Default value is FALSE.

Value

A list of the estimated parameters for the SEMinR model including:

meanData	A vector of the indicator means.
sdData	A vector of the indicator standard deviations
mmMatrix	A Matrix of the measurement model relations.
smMatrix	A Matrix of the structural model relations.
constructs	A vector of the construct names.
mmVariables	A vector of the indicator names.
outer_loadings	The matrix of estimated indicator loadings.
outer_weights	The matrix of estimated indicator weights.
path_coef	The matrix of estimated structural model relationships.
iterations	A numeric indicating the number of iterations required before the algorithm converged.
weightDiff	A numeric indicating the minimum weight difference between iterations of the algorithm.
construct_scores	A matrix of the estimated construct scores for the PLS model.
rSquared	A matrix of the estimated R Squared for each construct.
inner_weights	The inner weight estimation function.
data	A matrix of the data upon which the model was estimated (INcluding interactions).
rawdata	A matrix of the data upon which the model was estimated (EXcluding interactions).
measurement_model	The SEMinR measurement model specification.

See Also

[specify_model](#) [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,
                           missing = mean_replacement,
                           missing_value = NA)

summary(mobi_pls)
plot_scores(mobi_pls)

```

estimate_pls_mga	<i>Performs PLS-MGA to report significance of path differences between two subgroups of data</i>
------------------	--

Description

Performs PLS-MGA to report significance of path differences between two subgroups of data

Usage

```
estimate_pls_mga(pls_model, condition, nboot = 2000, ...)
```

Arguments

pls_model	SEMinR PLS model estimated on the full sample
condition	logical vector of TRUE/FALSE indicating which rows of sample data are in group 1
nboot	number of bootstrap resamples to use in PLS-MGA
...	any further parameters for bootstrapping (e.g., cores)

References

Henseler, J., Ringle, C. M. & Sinkovics, R. R. New Challenges to International Marketing. *Adv Int Marketing* 277–319 (2009) doi:10.1108/s1474-7979(2009)0000020014

Examples

```

mobi <- 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("Quality",    multi_items("PERQ", 1:7)),
  composite("Value",      multi_items("PERV", 1:2)),
  composite("Satisfaction", multi_items("CUSQ", 1:3)),
  composite("Complaints",  single_item("CUSCO")),
  composite("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,
                           missing = mean_replacement,
                           missing_value = NA)

# Should usually use nboot ~2000 and don't specify cores for full parallel processing

mobi_mga <- estimate_pls_mga(mobi_pls, mobi$CUEX1 < 8, nboot=50, cores = 2)

```

extract_bootstrapped_values

extract bootstrapped statistics from an edge using a row_index

Description

extract bootstrapped statistics from an edge using a row_index

Usage

```
extract_bootstrapped_values(ltbl, row_index, model, theme)
```

Arguments

tbl	a table of bootstrapped values (weights, loadings, path coefficients)
row_index	the index for the specific edge to extract
model	the model to use
theme	the theme to use

`extract_htmt_nodes` *Helper function that applies formatting to each construct*

Description

Helper function that applies formatting to each construct

Usage

`extract_htmt_nodes(model, theme)`

Arguments

model	the model to use
theme	the theme to use

Value

Returns a string of the structural model in dot notation.

`extract_mm_coding` *extracts the constructs and their types from the model*

Description

extracts the constructs and their types from the model

Usage

`extract_mm_coding(model)`

Arguments

model	the model to use
-------	------------------

extract_mm_edges*extract mm edges from model for a given index of all constructs*

Description

extract mm edges from model for a given index of all constructs

Usage

```
extract_mm_edges(index, model, theme, weights = 1000)
```

Arguments

index	the index of the construct
model	the model to use
theme	the theme to use
weights	a default weight for measurement models (high values suggested)

extract_mm_edge_value *gets the mm_edge value (loading, weight) for bootstrapped and regular models*

Description

gets the mm_edge value (loading, weight) for bootstrapped and regular models

Usage

```
extract_mm_edge_value(model, theme, indicator, construct)
```

Arguments

model	the model to use
theme	the theme to use
indicator	the indicator to use
construct	the construct to use

extract_mm_nodes	<i>gets the individual nodes and applies formatting</i>
------------------	---

Description

gets the individual nodes and applies formatting

Usage

```
extract_mm_nodes(index, model, theme)
```

Arguments

index	the index of the construct
model	the model to use
theme	the theme to use

extract_sm_nodes	<i>Helper function that applies formatting to each construct</i>
------------------	--

Description

Helper function that applies formatting to each construct

Usage

```
extract_sm_nodes(model, theme, structure_only = FALSE)
```

Arguments

model	the model to use
theme	the theme to use
structure_only	is this called in a structure_only model

Value

Returns a string of the structural model in dot notation.

```
format_endo_node_label
```

Helps to render a node label for endogenous variables

Description

Helps to render a node label for endogenous variables

Usage

```
format_endo_node_label(theme, name, rstring)
```

Arguments

theme	the theme to use
name	the content of the name placeholder
rstring	the content of the rstring placeholder

Value

Returns the formatted string

```
format_exo_node_label  Helps to render a node label for exogenous variables
```

Description

Helps to render a node label for exogenous variables

Usage

```
format_exo_node_label(theme, name)
```

Arguments

theme	the theme to use
name	the content of the name placeholder

Value

Returns the formatted string

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 <code>seminr_model</code> containing the estimated seminr model.
iv	An independent variable in the model.
dv	A dependent variable in the model.

Value

A matrix of the estimated F Square metric for each construct.

References

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

Examples

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

get_construct_element_size

Gets the optimal size for construct elements in the plot

Description

Currently orients on reflective theme settings

Usage

```
get_construct_element_size(model, theme)
```

Arguments

model	the model to use
theme	the theme to use

Value

Returns a two-element vector with c(width, height)

get_construct_type

Returns the type of a construct from a model

Description

Returns the type of a construct from a model

Usage

```
get_construct_type(model, construct)
```

Arguments

model	the model to get the type from
construct	the character string name of the construct

Value

Returns a character string

get_manifest_element_size

Gets the optimal size for manifest elements in the plot

Description

Currently orients on reflective theme settings

Usage

```
get_manifest_element_size(model, theme)
```

Arguments

model	the model to use
theme	the theme to use

Value

Returns a two-element vector with c(width, height)

get_mm_edge_style

individual styles for measurement model edges

Description

individual styles for measurement model edges

Usage

```
get_mm_edge_style(theme, construct_type, flip = FALSE)
```

Arguments

theme	the theme to use
construct_type	Forward direction?
flip	invert the arrow direction because of sink?

get_mm_node_shape *Get a string to insert into a node specification using the themed shape*

Description

Get a string to insert into a node specification using the themed shape

Usage

```
get_mm_node_shape(model, construct, theme)
```

Arguments

model	the model to use
construct	the construct to use
theme	the theme to use

Value

Returns a string that determines the shape of a node

get_mm_node_style *get global measurement model node style*

Description

get global measurement model node style

Usage

```
get_mm_node_style(theme)
```

Arguments

theme	the theme to use
-------	------------------

get_sm_node_shape	<i>Get a string to insert into a node specification using the themed shape</i>
-------------------	--

Description

Get a string to insert into a node specification using the themed shape

Usage

```
get_sm_node_shape(model, construct, theme)
```

Arguments

model	the model to use
construct	the construct to use
theme	the theme to use

Value

Returns a string that determines the shape of a node

get_value_dependent_mm_edge_style	<i>Formats the style of the structural model edges</i>
-----------------------------------	--

Description

Formats the style of the structural model edges

Usage

```
get_value_dependent_mm_edge_style(value, theme)
```

Arguments

value	value to compare for negativity
theme	the theme to use

Value

Returns the style for the edge (both style and color)

```
get_value_dependent_sm_edge_style
```

Formats the style of the structural model edges

Description

Formats the style of the structural model edges

Usage

```
get_value_dependent_sm_edge_style(value, theme)
```

Arguments

value	value to compare for negativity
theme	the theme to use

Value

Returns the style for the edge (both style and color)

```
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.

Value

A vector of the indicators for a higher-order-composite.

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)
)
```

higher_reflective *higher_reflective*

Description

higher_reflective creates a higher-order reflective construct

Usage

```
higher_reflective(construct_name, dimensions)
```

Arguments

construct_name of second-order construct
 dimensions the first-order constructs

Details

This function maps first-order constructs onto second-order reflective constructs using construct names. It is currently only suitable for CB-SEM and not PLS

Value

A vector of the indicators for a higher-order-factor.

See Also

See [constructs](#), [reflective](#)

Examples

```
mobi_mm <- constructs(
  reflective("Image",           multi_items("IMAG", 1:5)),
  reflective("Expectation",     multi_items("CUEX", 1:3)),
  higher_reflective("Quality",  c("Image", "Expectation"))
)
```

influencer_data	<i>Measurement Instrument for the Influencer Model</i>
-----------------	--

Description

The data set is used as measurement instrument for the Influencer Model which is used in Partial Least Squares Structural Equation Modeling (PLS-SEM) Using R - A Workbook (2021) Hair, J.F. (Jr), Hult, T.M., Ringle, C.M., Sarstedt, M., Danks, N.P., and Ray, S.

Usage

influencer_data

Format

A data frame with 250 rows and 24 variables:

- sic_1** The influencer reflects who I am.
- sic_2** I can identify with the influencer.
- sic_3** I feel a personal connection to the influencer.
- sic_4** I (can) use the influencer to communicate who I am to other people.
- sic_5** I think the influencer (could) help(s) me become the type of person I want to be.
- sic_6** I consider the influencer to be "me".
- sic_7** The influencer suits me well.
- sic_global** My personality and the personality of the influencer relate accordingly to one another.
- pq_1** The product has excellent quality.
- pq_2** The product looks to be reliable and durable.
- pq_3** The product will have fewer problems.
- pq_4** The product has excellent quality features.
- pl_1** I dislike the product (reverse coded).
- pl_2** The product is appealing to me.
- pl_3** The presented product raises a positive feeling in me.
- pl_4** The product is interesting to me.
- pi_1** It is very likely that I will purchase this product.
- pi_2** I will purchase this product the next time I need it.

pi_3 I would definitely try the product out.

pi_4 I would recommend this product to my friends.

pi_5 I am willing to purchase this product.

pic_1 The influencer is qualified.

pic_2 The influencer is competent.

pic_3 The influencer is an expert.

pic_4 The influencer is experienced.

pic_5 The influencer is knowledgeable.

wtp Please state your willingness to pay (in Euro) for the presented product.

influencer_group A binary variable indicating which group the influencer belongs to.

Details

The data frame influencer_data contains the observed data for the model specified in the Influencer Model.

Examples

```
data("influencer_data")
```

interaction_term	<i>Interaction function</i>
-------------------------	-----------------------------

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

<code>iv</code>	The independent variable that is subject to moderation.
<code>moderator</code>	The moderator variable.
<code>method</code>	The method to generate the estimated interaction term with a default of 'two_stage'.
<code>weights</code>	The weighting mode for interaction items in a PLS model (only) with default of 'modeA'.

Details

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

Value

An un-evaluated function (promise) for generating a vector of interaction terms.

Interaction Combinations as generated by the `interaction` or `interaction_term` methods.

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)
```

is_sink

Tests whether the i _th construct is endogenous or not

Description

Tests whether the i _th construct is endogenous or not

Usage

```
is_sink(model, index)
```

Arguments

model	the model object
index	the index of the construct to test

Value

whether the construct is endogenous or not

item_errors	<i>Specifies pair of items, or sets of items, that should covary. Used to specify error covariances for associations.</i>
-------------	---

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

Value

A vector of items that covary.

Examples

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

mean_replacement	<i>Function to clean data of omitted values by mean replacement</i>
------------------	---

Description

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

Usage

```
mean_replacement(data)
```

Arguments

data	A dataset to be used for estimating a SEMinR model
------	--

Details

`mean_replacement` provides the verb for replacing all omitted values (NA only) in the dataset with the mean of the variable. It gives a warning if more than 5

Value

A dataset with all missing values replaced with column means

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.

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

`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`.

Value

A matrix of estimated measurement model relations.

`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`.

Value

A matrix of estimated measurement model relations.

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

Description

`mode_plsc` specifies outer weighting as mode A and then applies PLSc disattenuation to mimic common-factor model estimates.

Usage

```
mode_plsc(mmMatrix, j, 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>j</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> .

Value

A matrix of estimated measurement model relations.

multi_items*Multi-items measurement model specification*

Description

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

Usage

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

Arguments

<code>item_name</code>	Prefix name of items
<code>item_numbers</code>	The range of number suffixes for the items
<code>...</code>	Additional Item names and numbers

Value

A vector of numbered indicators.

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)  
)
```

node_endo_template_default

The default template for labeling endogenous construct nodes

Description

The default template for labeling endogenous construct nodes

Usage

```
node_endo_template_default()
```

Value

The template string

node_exo_template_default

The default template for labeling exogenous construct nodes

Description

The default template for labeling exogenous construct nodes

Usage

```
node_exo_template_default()
```

Value

The template string

orthogonal	<i>orthogonal creates interaction measurement items by using the orthogonalized approach wherein</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):  
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 <code>correlation_weights</code> or <code>mode_A</code> for correlation weights (Mode A) or as <code>regression_weights</code> or <code>mode_B</code> for regression weights (Mode B). Default is correlation weights.

Value

An un-evaluated function (promise) for estimating an orthogonal interaction effect.

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.

Value

A matrix of estimated structural relations.

References

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

path_weighting	<i>Inner weighting scheme functions to estimate inner paths matrix</i>
----------------	--

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.

Value

A matrix of estimated structural relations.

References

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

plot.reliability_table	<i>Function for plotting the measurement model reliability metrics of a PLS model</i>
------------------------	---

Description

plot.reliability_table generates an easy to read visualization of the rhoA, Cronbachs alpha, and Composite Reliability for all constructs. The plot visualizes the metrics in such a way as to draw meaning from not only the absolute values, but their relative values too.

Usage

```
## S3 method for class 'reliability_table'
plot(x, ...)
```

Arguments

- x A reliability_table object from a SEMinR PLS model. This can be accessed as the reliability element of the PLS model summary object.
- ... All other arguments inherited from plot.

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)
plot(summary(mobi_pls)$reliability)
```

plot.seminr_model *Plot various SEMinR models*

Description

With the help of the DiagrammeR package this dot graph can then be plotted in various in RMarkdown, shiny, and other contexts. Depending on the type of model, different parameters can be used. Please check the [dot_graph](#) function for additional parameters.

Usage

```
## S3 method for class 'seminr_model'
plot(x, title = "", theme = NULL, ...)
```

Arguments

- x The model description
- title An optional title for the plot
- theme Theme created with [seminr_theme_create](#).
- ... Please check the [dot_graph](#) for the additional parameters

Value

Returns the plot.

plot_htmt

Plots a network graph of constructs based on HTMT measures

Description

Using a bootstrapped model this functions shows which constructs show insufficient discriminant validity.

Usage

```
plot_htmt(
  model,
  title = "HTMT Plot",
  theme = seminr::seminr_theme_get(),
  htmt_threshold = 1,
  omit_threshold_edges = TRUE,
  use_ci = FALSE
)
```

Arguments

<code>model</code>	A bootstrapped PLS-Model
<code>title</code>	Optional title over the plot.
<code>theme</code>	Optional theme to use for plotting
<code>htmt_threshold</code>	The threshold to use under which constructs are highlighted (default = 1.0)
<code>omit_threshold_edges</code>	Whether or not to omit constructs that have low HTMT values (default = TRUE)
<code>use_ci</code>	Whether or not to rely on the upper threshold of the CI instead of the bootstrapped mean (default = FALSE)

Value

Returns a dot string of the plot

plot_interaction	<i>Function for plotting interaction plot for moderated PLS or CBSEM model</i>
------------------	--

Description

plot_interaction generates an interaction plot for the effect of an antecedent on an outcome given a mediator variable.

Usage

```
plot_interaction(moderated_model, intxn, dv, legend)
```

Arguments

moderated_model	SEMinR model that contains an interaction.
intxn	Name (character) of the interaction term in the structural model. Must look like a product of independent variable and moderator (e.g., "ABC*XYZ")
dv	Name (character) of the dependant construct affected by the moderator.
legend	Location (character) of the legend on the plot; must be a combination of bottomtop and leftlright (e.g., "bottomright").

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 = c("Expectation"), method = orthogonal))

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

# Load data, assemble model, and estimate
mobi_pls <- estimate_pls(data = mobi,
  measurement_model = mobi_mm,
  structural_model = mobi_sm)

plot_interaction(mobi_pls, "Image*Expectation", "Satisfaction", "bottomright")
```

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.

Value

A SEMinR model object which has been adjusted according to PLSc.

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)

```

predict_DA*Predictive Scheme*

Description

`predict_EA` and `predict_DA` specify the predictive scheme to be used in the generation of the predictions. EA refers to Earliest Antecedents nad DA to Direct Antecedents.

Usage

```
predict_DA(smMatrix, path_coef, construct_scores)
```

Arguments

<code>smMatrix</code>	is the <code>structural_model</code> - a source-to-target matrix representing the inner/structural model, generated by relationships generated by SEMinR.
<code>path_coef</code>	is the Path Coefficients matrix from a SEMinR model.
<code>construct_scores</code>	is the matrix of construct scores generated by SEMinR.

predict_EA*Predictive Scheme*

Description

`predict_EA` and `predict_DA` specify the predictive scheme to be used in the generation of the predictions. EA refers to Earliest Antecedents nad DA to Direct Antecedents.

Usage

```
predict_EA(smMatrix, path_coef, construct_scores)
```

Arguments

smMatrix is the `structural_model` - a source-to-target matrix representing the inner/structural model, generated by `relationships` generated by SEMinR.

path_coef is the Path Coefficients matrix from a SEMinR model.

construct_scores is the matrix of construct scores generated by SEMinR.

predict_pls *Predict_pls performs either k-fold or LOOCV on a SEMinR PLS model and generates predictions*

Description

`predict_pls` uses cross-validation to generate in-sample and out-sample predictions for PLS models generated by SEMinR.

Usage

```
predict_pls(model, technique, noFolds, reps, cores)
```

Arguments

model A SEMinR model that has been estimated on the FULL dataset.

technique The predictive technique to be employed, Earliest Antecedents (EA) `predict_EA` or Direct Antecedents (DA) `predict_DA`

noFolds The required number of folds to use in k-fold cross validation. If NULL, then parallel LOOCV will be executed. Default is NULL.

reps The number of times the cross-validation will be repeated. Default is NULL.

cores The number of cores to use for parallel LOOCV processing. If k-fold is used, the process will not be parallelized.

Details

This function generates cross-validated in-sample and out-sample predictions for PLS models generated by SEMinR. The cross validation technique can be k-fold if a number of folds are specified, or leave-one-out-cross-validation (LOOCV) if no folds are specified. LOOCV is recommended for small datasets.

Value

A list of the estimated PLS and LM prediction results:

PLS_out_of_sample

A matrix of the out-of-sample indicator predictions generated by the SEMinR model.

PLS_in_sample A matrix of the in-sample indicator predictions generated by the SEMinR model.
lm_out_of_sample A matrix of the out-of-sample indicator predictions generated by a linear regression model.
lm_in_sample A matrix of the in-sample indicator predictions generated by a linear regression model.
item_actualls A matrix of the actual indicator scores.
PLS_out_of_sample_residuals A matrix of the out-of-sample indicator PLS prediction residuals.
PLS_in_sample_residuals A matrix of the in-sample indicator PLS prediction residuals.
lm_out_of_sample_residuals A matrix of the out-of-sample LM indicator prediction residuals.
lm_in_sample_residuals A matrix of the in-sample LM indicator prediction residuals.
mmMatrix A Matrix of the measurement model relations.
smMatrix A Matrix of the structural model relations.
constructs A vector of the construct names.
mmVariables A vector of the indicator names.
outer_loadings The matrix of estimated indicator loadings.
outer_weights The matrix of estimated indicator weights.
path_coef The matrix of estimated structural model relationships.
iterations A numeric indicating the number of iterations required before the algorithm converged.
weightDiff A numeric indicating the minimum weight difference between iterations of the algorithm.
construct_scores A matrix of the estimated construct scores for the PLS model.
rSquared A matrix of the estimated R Squared for each construct.
inner_weights The inner weight estimation function.
data A matrix of the data upon which the model was estimated (INcluding interactions).
rawdata A matrix of the data upon which the model was estimated (EXcluding interactions).
measurement_model The SEMinR measurement model specification.

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_sm <- relationships(
  paths(to = "Satisfaction",
        from = c("Image", "Expectation", "Value"))
)

mobi_pls <- estimate_pls(mobi, mobi_mm, mobi_sm)
cross_validated_predictions <- predict_pls(model = mobi_pls,
                                             technique = predict_DA,
                                             noFolds = 10,
                                             cores = NULL)

```

print.seminr_pls_mga *Summary function for PLS-MGA*

Description

Summary function for PLS-MGA

Usage

```
## S3 method for class 'seminr_pls_mga'
print(x, digits = 3, ...)
```

Arguments

x	estimated seminr_pls_mga object
digits	number of digits to print
...	any further parameters for printing

product_indicator	product_indicator <i>creates interaction measurement items by scaled product indicator approach.</i>
-------------------	--

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 <code>correlation_weights</code> or <code>mode_A</code> for correlation weights (Mode A) or as <code>regression_weights</code> or <code>mode_B</code> for regression weights (Mode B). Default is correlation weights.

Value

An un-evaluated function (promise) for estimating a product-indicator interaction effect.

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 <code>multi_items</code> or <code>single_item</code> functions

Details

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

Value

A vector of the indicators for a reflective construct.

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))
)
```

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.

Value

A vector of construct names and structural relationships.

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_missing	<i>Function to report how missing data was handled and how much was missing.</i>
----------------	--

Description

Function to report how missing data was handled and how much was missing.

Usage

```
report_missing(seminr_model)
```

Arguments

seminr_model An estimated seminr PLS model.

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.

Value

A matrix of structural paths.

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)
```

rerun*Reruns a previously specified seminr model/analysis*

Description

Reruns a previously specified seminr model/analysis

Usage

```
rerun(x, ...)
```

Arguments

x	An estimated seminr_model object - refer to specific rerun methods
...	Any parameters to change during the rerun.

Value

A re-estimated model of the same class

See Also

[rerun.pls_model](#)

<code>rerun.pls_model</code>	<i>Reruns a previously specified seminr PLS model</i>
------------------------------	---

Description

Reruns a previously specified seminr PLS model

Usage

```
## S3 method for class 'pls_model'
rerun(x, ...)
```

Arguments

<code>x</code>	An estimated <code>pls_model</code> object produced by estimate_pls
<code>...</code>	Any parameters to change during the re-estimation (e.g., <code>data</code> , <code>measurement_model</code> , etc.)

Value

A re-estimated `pls_model` object

Examples

```
mobi <- mobi

mobi_mm <- constructs(
  composite("Image",           multi_items("IMAG", 1:5)),
  composite("Loyalty",         multi_items("CUSL", 1:3))
)

mobi_sm <- relationships(
  paths(from = "Image",        to = c("Loyalty"))
)

mobi_pls <- estimate_pls(data = mobi,
                           measurement_model = mobi_mm,
                           structural_model = mobi_sm,
                           missing = mean_replacement,
                           missing_value = NA)

# Re-estimate model faithfully
mobi_pls2 <- rerun(mobi_pls)

# Re-estimated model with altered measurement model
mobi_pls3 <- rerun(mobi_pls, measurement_model=as.reflective(mobi_mm))
```

<code>rhoC_AVE</code>	<i>seminr rhoC_AVE() function</i>
-----------------------	-----------------------------------

Description

Get rhoC and AVE for a CFA model estimated with `estimate_pls`, `estimate_cbsem` or `estimate_cfa`. Dillon-Goldstein's Rho as per: Dillon, W. R, and M. Goldstein. 1987. Multivariate Analysis: Methods and Applications. Biometrical Journal 29 (6). Average Variance Extracted as per: Fornell, C. and D. F. Larcker (February 1981). Evaluating structural equation models with unobservable variables and measurement error, Journal of Marketing Research, 18, pp. 39-5

Usage

```
rhoC_AVE(x, constructs = NULL)
```

Arguments

<code>x</code>	Estimated <code>seminr_model</code> object.
<code>constructs</code>	Vector containing the names of the constructs to calculate rhoC and AVE for; if <code>NULL</code> , all constructs are used.

Value

A matrix containing the rhoC and AVE metrics for each construct.

<code>rho_A</code>	<i>seminr rho_A Function</i>
--------------------	------------------------------

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, constructs)
```

Arguments

<code>seminr_model</code>	A <code>seminr_model</code> containing the estimated <code>seminr</code> model.
<code>constructs</code>	A vector containing the names of the constructs to calculate rhoA for.

Value

A matrix containing the rhoA metric for each construct.

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, mobi_pls$constructs)
```

save_plot

Saves a SEMinR model plot to file

Description

Saves a SEMinR model plot to a graphical file. Default output is RPlots.pdf.

Usage

```
save_plot(
  filename = "RPlot.pdf",
  plot = last_seminr_plot(),
  width = NULL,
```

```
  height = NULL
)
```

Arguments

filename	The name of the file output (can be png, pdf, webp, ps, or svg.)
plot	A plot that is created from the <code>plot</code> function. By default it uses the last plot created.
width	An optional parameter for width in pixels.
height	An optional parameter for height in pixels.

Value

Does not return a value

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")
)

# estimate the model
mobi_pls <- estimate_pls(data = mobi,
                           measurement_model = mobi_mm,
                           structural_model = mobi_sm)
## Not run:
# generate the plot
plot(mobi_pls)
# save to file
save_plot("myplot.pdf")

## End(Not run)
```

seminr_theme_academic *A theme function for a basic b/w theme*

Description

A theme function for a basic b/w theme

Usage

```
seminr_theme_academic(
  plot.title.fontsize = 24,
  mm.node.label.fontsize = 8,
  sm.node.label.fontsize = 12,
  mm.edge.label.fontsize = 7,
  sm.edge.label.fontsize = 9
)
```

Arguments

plot.title.fontsize	Title font size
mm.node.label.fontsize	Font size for measurement variables
sm.node.label.fontsize	Font size for constructs
mm.edge.label.fontsize	Font size for measurement model edges
sm.edge.label.fontsize	Font size for path edges

Value

a theme object

seminr_theme_create *Create a theme for a seminr graph visualization*

Description

All customizable options are parameters of this function. See the details all the way down for more information.

Usage

```
seminr_theme_create(  
  plot.title.fontsize = 24,  
  plot.title.fontcolor = "black",  
  plot.fontname = "helvetica",  
  plot.splines = TRUE,  
  plot.rounding = 3,  
  plot.adj = FALSE,  
  plot.specialcharacters = TRUE,  
  plot.randomizedweights = FALSE,  
  plot.bgcolor = "transparent",  
  mm.node.color = "dimgrey",  
  mm.node.fill = "white",  
  mm.node.label.fontsize = 8,  
  mm.node.label.fontcolor = "black",  
  mm.edge.positive.color = "dimgrey",  
  mm.edge.negative.color = "dimgrey",  
  mm.edge.positive.style = "solid",  
  mm.edge.negative.style = "dashed",  
  mm.edge.label.fontsize = 7,  
  mm.edge.label.fontcolor = "black",  
  mm.edge.label.show = TRUE,  
  mm.edge.minlen = 1,  
  mm.edge.width_multiplier = 3,  
  mm.edge.width_offset = 0.5,  
  mm.edge.use_outer_weights = TRUE,  
  mm.edge.boot.show_t_value = FALSE,  
  mm.edge.boot.show_p_value = FALSE,  
  mm.edge.boot.show_p_stars = TRUE,  
  mm.edge.boot.show_ci = FALSE,  
  mm.edge.boot.template = edge_template_minimal(),  
  sm.node.color = "black",  
  sm.node.fill = "white",  
  sm.node.label.fontsize = 12,  
  sm.node.label.fontcolor = "black",  
  sm.node.endo.template = node_endo_template_default(),  
  sm.node.exo.template = node_exo_template_default(),  
  sm.edge.boot.show_t_value = FALSE,  
  sm.edge.boot.show_p_value = FALSE,  
  sm.edge.boot.show_p_stars = TRUE,  
  sm.edge.boot.show_ci = TRUE,  
  sm.edge.boot.template = edge_template_default(),  
  sm.edge.positive.color = "black",  
  sm.edge.negative.color = "black",  
  sm.edge.positive.style = "solid",  
  sm.edge.negative.style = "dashed",  
  sm.edge.label.fontsize = 9,  
  sm.edge.label.fontcolor = "black",
```

```

sm.edge.label.show = TRUE,
sm.edge.label.all_betas = TRUE,
sm.edge.minlen = NA_integer_,
sm.edge.width_offset = 0.5,
sm.edge.width_multiplier = 5,
construct.reflective.shape = "ellipse",
construct.reflective.arrow = "backward",
construct.reflective.use_weights = FALSE,
construct.compositeA.shape = "hexagon",
construct.compositeA.arrow = "backward",
construct.compositeA.use_weights = FALSE,
construct.compositeB.shape = "hexagon",
construct.compositeB.arrow = "forward",
construct.compositeB.use_weights = TRUE,
manifest.reflective.shape = "box",
manifest.compositeA.shape = "box",
manifest.compositeB.shape = "box",
...
)

```

Arguments

plot.title.fontsize	Font size of the title.
plot.title.fontcolor	Fontcolor of the title of the plot.
plot.fontname	Font to be used throughout the plot.
plot.splines	Whether or not to use splines as edges (default = TRUE).
plot.rounding	The amount of decimals to keep for rounding (default = 3).
plot.adj	TRUE or FALSE (default). Whether or not to use adjusted r^2 in constructs.
plot.specialcharacters	Whether or not to use greek UTF-8 symbols in plots.
plot.randomizedweights	TRUE or FALSE (default), decides whether to add. minimal random weights to the measurement model. Can help with determinism in plot outcomes.
plot.bgcolor	The background color of the plot (default = "transparent").
mm.node.color	Color of the measurement model nodes.
mm.node.fill	Fill of the measurement model nodes.
mm.node.label.fontsize	Font size of the measurement model node labels.
mm.node.label.fontcolor	Color of the measurement model node labels.
mm.edge.positive.color	Color of the measurement model edges, when values are positive.
mm.edge.negative.color	Color of the measurement model edges, when values are negative.

```
mm.edge.positive.style
  Style of the measurement model edges, when values are positive.
mm.edge.negative.style
  Style of the measurement model edges, when values are negative.
mm.edge.label.fontsize
  Font size of the measurement model edge labels.
mm.edge.label.fontcolor
  Font color of the measurement model edge labels.
mm.edge.label.show
  Whether or not to show measurement model edge labels.
mm.edge.minlen  Minimum length of the measurement model edges.
mm.edge.width_multiplier
  The multiplier for measurement model edge penwidth (default = 3).
mm.edge.width_offset
  The minimal width of an edge of the measurement model (default = 0.5).
mm.edge.use_outer_weights
  Whether or not to use outer weights as edge labels in the measurement model.
mm.edge.boot.show_t_value
  Should boot-strapped loadings/weights show a t-value
mm.edge.boot.show_p_value
  Should boot-strapped loadings/weights show a p-value
mm.edge.boot.show_p_stars
  Should boot-strapped loadings/weights show significance stars
mm.edge.boot.show_ci
  Should boot-strapped loadings/weights show a 95 percent confidence interval
mm.edge.boot.template
  A template string for HTML formatting of edges for loadings/weights
sm.node.color  Color of the structural model nodes.
sm.node.fill    Fill of the structural model nodes.
sm.node.label.fontsize
  Font size of the structural model node labels.
sm.node.label.fontcolor
  Font color of the structural model node labels.
sm.node.endo.template
  A template string for the nodes of endogenous constructs
sm.node.exo.template
  A template string for the nodes of exogenous constructs
sm.edge.boot.show_t_value
  Should boot-strapped path coefficients show a t-value
sm.edge.boot.show_p_value
  Should boot-strapped path coefficients show a p-value
sm.edge.boot.show_p_stars
  Should boot-strapped path coefficients show significance stars
sm.edge.boot.show_ci
  Should boot-strapped path coefficients show a 95 percent confidence interval
```

```

sm.edge.boot.template
  A template string for HTML formatting of edges
sm.edge.positive.color
  Color of the structural model edges, when values are positive.
sm.edge.negative.color
  Color of the structural model edges, when values are negative.
sm.edge.positive.style
  Style of the structural model edges, when values are positive.
sm.edge.negative.style
  Style of the structural model edges, when values are negative.
sm.edge.label.fontsize
  Font size of the structural model edge labels.
sm.edge.label.fontcolor
  Font color of the structural model edge labels.
sm.edge.label.show
  Whether or not to show edge labels on structural model edges.
sm.edge.label.all_betas
  Whether to label both endogenous and exogenous paths with a beta (default = TRUE).
sm.edge.minlen
  Minimum length of the structural model edges.
sm.edge.width_offset
  The minimal width of an edge of the structural model (default = 0.5).
sm.edge.width_multiplier
  The multiplier for structural model edges (default = 5).
construct.reflective.shape
  Dot shape of reflective constructs
construct.reflective.arrow
  Direction of the arrow for reflective constructs. Can be forward, backward (default), or none.
construct.reflective.use_weights
  Should measurements from reflective constructs show weights (TRUE) or loadings (FALSE: default).
construct.compositeA.shape
  Dot shape of composite constructs using correlation weights
construct.compositeA.arrow
  Direction of the arrow for constructs using correlation weight (default: backward)
construct.compositeA.use_weights
  Should measurements from constructs using correlation weights show weights (TRUE) or loadings (FALSE: default).
construct.compositeB.shape
  Dot shape of composite constructs using regression weights
construct.compositeB.arrow
  Direction of the arrow for constructs using regression weights (default: forward)

```

```

construct.compositeB.use_weights
  Should measurements from constructs using regression weights show weights
  (TRUE: default) or loadings (FALSE).

manifest.reflective.shape
  Dot shape of manifest variables of reflective constructs

manifest.compositeA.shape
  Dot shape of manifest variables of composite constructs using correlation weights

manifest.compositeB.shape
  Dot shape of manifest variables of composite constructs using regression weights

...
  additional parameters (unused)

```

Details

You can use the auto-complete feature of your editor to help you find the right parameter.

General settings start with `plot.*`

Measurement model settings start with `mm.*`

Structural model settings start with `sm.*`

Setting the shape of manifest or construct variables depending on their estimation type can be found under `manifest.*` and `construct.*`

Value

A `seminr.theme` object that can be supplied to `dot_graph`

`seminr_theme_dark` *The theme function for an inverted theme on black background.*

Description

The theme function for an inverted theme on black background.

Usage

```

seminr_theme_dark(
  plot.title.fontsize = 24,
  mm.node.label.fontsize = 8,
  sm.node.label.fontsize = 12,
  mm.edge.label.fontsize = 7,
  sm.edge.label.fontsize = 9
)

```

Arguments

```

plot.title.fontsize
  Title font size
mm.node.label.fontsize
  Font size for measurement variables
sm.node.label.fontsize
  Font size for constructs
mm.edge.label.fontsize
  Font size for measurement model edges
sm.edge.label.fontsize
  Font size for path edges

```

Value

a theme object

seminr_theme_get *Get and set the active theme*

Description

The current/active theme (see [seminr_theme()]) is automatically applied to every graph you draw. Use ‘seminr_theme_get()‘ to get the current theme, and ‘seminr_theme_set()‘ to completely override it.

Usage

```

seminr_theme_get()
seminr_theme_set(new)

```

Arguments

new	new theme (a list of theme elements)
-----	--------------------------------------

seminr_theme_smart *A colored theme*

Description

A colored theme
A theme function for a modern approach of visualizing PLS models in b/w

Usage

```
seminr_theme_smart(  
  plot.title.fontsize = 24,  
  mm.node.label.fontsize = 8,  
  sm.node.label.fontsize = 12,  
  mm.edge.label.fontsize = 7,  
  sm.edge.label.fontsize = 9  
)  
  
seminr_theme_default(  
  plot.title.fontsize = 24,  
  mm.node.label.fontsize = 8,  
  sm.node.label.fontsize = 12,  
  mm.edge.label.fontsize = 7,  
  sm.edge.label.fontsize = 9,  
  construct.reflective.shape = "ellipse",  
  construct.compositeA.shape = "hexagon",  
  construct.compositeB.shape = "hexagon",  
  construct.reflective.arrow = "backward",  
  construct.compositeA.arrow = "backward",  
  construct.compositeB.arrow = "forward",  
  ...  
)
```

Arguments

```
plot.title.fontsize  
          Title font size  
mm.node.label.fontsize  
          Font size for measurement variables  
sm.node.label.fontsize  
          Font size for constructs  
mm.edge.label.fontsize  
          Font size for measurement model edges  
sm.edge.label.fontsize  
          Font size for path edges  
construct.reflective.shape  
          Shape of reflective constructs
```

```

construct.compositeA.shape
  Shape of composite constructs mode A
construct.compositeB.shape
  Shape of composite constructs mode B
construct.reflective.arrow
  Direction of arrows of reflective constructs
construct.compositeA.arrow
  Direction of arrows of composite constructs mode A
construct.compositeB.arrow
  Direction of arrows of composite constructs mode B
...
  Other parameters for the seminr_theme_create function

```

Value

a theme object

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 data frame 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

Value

A list of the estimated parameters for the SimplePLS model including:

meanData	A vector of the indicator means.
sdData	A vector of the indicator standard deviations
mmMatrix	A Matrix of the measurement model relations.
smMatrix	A Matrix of the structural model relations.
constructs	A vector of the construct names.
mmVariables	A vector of the indicator names.
outer_loadings	The matrix of estimated indicator loadings.
outer_weights	The matrix of estimated indicator weights.
path_coef	The matrix of estimated structural model relationships.
iterations	A numeric indicating the number of iterations required before the algorithm converged.
weightDiff	A numeric indicating the minimum weight difference between iterations of the algorithm.
construct_scores	A matrix of the estimated construct scores for the PLS model.
rSquared	A matrix of the estimated R Squared for each construct.
inner_weights	The inner weight estimation function.

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

<code>item</code>	Name of item
-------------------	--------------

Value

A vector of a single indicator for a composite.

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"))
)
```

slope_analysis	<i>Function for plotting a slope analysis for an interaction in a PLS model</i>
----------------	---

Description

`slope_analysis` generates an interaction plot for the effect of an antecedent on an outcome given a mediator variable.

Usage

```
slope_analysis(moderated_model, dv, moderator, iv, leg_place)
```

Arguments

<code>moderated_model</code>	A SEMinR model that contains an interaction.
<code>dv</code>	The name of the dependant construct affected by the moderator (interaction term).
<code>moderator</code>	The name of the moderator construct.
<code>iv</code>	The name of the independant construct affected by the moderator.
<code>leg_place</code>	The location of the legend, in order to make sure the legend does not obscure the plot lines.

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 = c("Expectation"), method = orthogonal))

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

# Load data, assemble model, and estimate
mobi_pls <- estimate_pls(data = mobi,
  measurement_model = mobi_mm,
  structural_model = mobi_sm)
```

```
slope_analysis(mobi_pls, "Satisfaction", "Expectation", "Image", "bottomright")
```

specific_effect_significance

seminr specific effect significance function

Description

The `seminr` package provides a natural syntax for researchers to describe PLS structural equation models. `specific_effect_significance` provides the verb for calculating the bootstrap mean, standard deviation, T value, and confidence intervals for direct or mediated path in a bootstrapped SEMinR model.

Usage

```
specific_effect_significance(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 a vector of mediators 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.

Value

A vector of lower and upper confidence intervals for a path.

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)

specific_effect_significance(boot_seminr_model = boot_seminr_model,
                             from = "Image",
                             through = c("Expectation", "Satisfaction", "Complaints"),
                             to = "Loyalty",
                             alpha = 0.05)

```

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

Description

Combines model components together into a single `specified_model` object for estimation functions

Usage

```
specify_model(
  measurement_model,
  structural_model = NULL,
```

```
  item_associations = NULL
)
```

Arguments

`measurement_model`

An optional `measurement_model` object representing the outer/measurement model, as generated by `constructs`.

`structural_model`

An optional `smMatrix` object representing the inner/structural model, as 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 inter-item associations).

Value

A list containing a SEMinR measurement model, structural model, and item associations.

See Also

[estimate_pls](#) [estimate_cbsem](#) [estimate_cfa](#)

`standardize_safely` *Standardize (scale) a matrix/df and report interpretable errors*

Description

Standardize (scale) a matrix/df and report interpretable errors

Usage

`standardize_safely(x)`

Arguments

`x` vector, data.frame, or matrix

Value

scaled object as returned by `scale` function

total_indirect_ci	<i>seminr total indirect confidence intervals function</i>
-------------------	--

Description

total_indirect_ci provides the verb for calculating the total indirect confidence intervals of a direct or mediated path in a bootstrapped SEMinR model.

Usage

```
total_indirect_ci(boot_seminr_model, from, to, 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.
alpha	A parameter for specifying the alpha for the confidence interval. Default is 0.05.

Value

A vector of lower and upper confidence intervals for a path.

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("Satisfaction", "Loyalty")),
  paths(from = "Satisfaction",   to = c("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)

total_indirect_ci(boot_seminr_model = boot_seminr_model,
                  from = "Image",
                  to = "Loyalty",
                  alpha = 0.05)

```

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.

Value

An un-evaluated function (promise) for estimating a two-stage interaction effect.

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)
```

Description

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

Usage

```
unit_weights(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`.

Value

A matrix of estimated measurement model relations.

Index

* **datasets**
 corp_rep_data, 15
 corp_rep_data2, 17
 influencer_data, 47
 mobi, 51

as.reflective, 4, 27–30
as.reflective.construct, 4, 5, 7
as.reflective.interaction, 6
as.reflective.measurement_model, 4–6, 7
associations, 8, 28, 30, 50

boot_paths_df, 10
bootstrap_model, 8, 33, 62, 74, 85, 88, 91
browse_plot, 11

check_test_plot, 12
composite, 4–7, 12, 14, 68
compute_itcriteria_weights, 13
constructs, 4–7, 9, 13, 14, 28, 30, 33, 46, 62, 68, 74, 85
cor_rsq, 19
corp_rep_data, 15
corp_rep_data2, 17
correlation_weights (mode_A), 52
csem2seminr, 19

df_xtab_matrix, 20
dot_component_mm, 21
dot_graph, 21, 59, 81
dot_graph_htmt, 24
dot_subcomponent_mm, 25

edge_template_default, 26
edge_template_minimal, 26
esc_node, 26
estimate_cbsem, 8, 27, 90
estimate_cfa, 8, 29, 90
estimate_lavaan_ten_berge, 31
estimate_pls, 19, 32, 72, 85, 90
estimate_pls_mga, 34

extract_bootstrapped_values, 35
extract_htmt_nodes, 36
extract_mm_coding, 36
extract_mm_edge_value, 37
extract_mm_edges, 37
extract_mm_nodes, 38
extract_sm_nodes, 38

format_endo_node_label, 39
format_exo_node_label, 39
fSquared, 40

get_construct_element_size, 41
get_construct_type, 41
get_manifest_element_size, 42
get_mm_edge_style, 42
get_mm_node_shape, 43
get_mm_node_style, 43
get_sm_node_shape, 44
get_value_dependent_mm_edge_style, 44
get_value_dependent_sm_edge_style, 45

higher_composite, 45
higher_reflective, 46

influencer_data, 47
interaction, 49
interaction_term, 9, 33, 48, 49, 62, 74, 85
is_sink, 49
item_errors, 8, 28, 30, 50

mean_replacement, 50
mobi, 51
mode_A, 52
mode_A, (mode_A), 52
mode_B, 53
mode_B, (mode_B), 53
mode_plsc, 54
multi_items, 54, 86

node_endo_template_default, 55

node_exo_template_default, 55
orthogonal, 56
path_factorial, 57
path_weighting, 58
paths, 9, 28, 33, 62, 74, 85
paths (relationships), 69
plot, 75
plot.reliability_table, 58
plot.seminr_model, 59
plot_htmt, 60
plot_interaction, 61
plot_scores (report_paths), 70
PLSc, 62
predict_DA, 63
predict_EA, 63
predict_pls, 64
print.seminr_pls_mga, 66
product_indicator, 66
reflective, 13, 14, 30, 46, 68
regression_weights (mode_B), 53
relationships, 9, 28, 33, 62, 69, 74, 85
report_missing, 70
report_paths, 70
rerun, 71
rerun.pls_model, 71, 72
rho_A, 73
rhoC_AVE, 73
save_plot, 74
seminr_theme_academic, 76
seminr_theme_create, 22, 59, 76
seminr_theme_dark, 81
seminr_theme_default
 (seminr_theme_smart), 83
seminr_theme_get, 82
seminr_theme_set (seminr_theme_get), 82
seminr_theme_smart, 83
simplePLS, 84
single_item, 55, 86
slope_analysis, 87
specific_effect_significance, 88
specify_model, 33, 89
standardize_safely, 90
total_indirect_ci, 91
two_stage, 92
unit_weights, 93