Package 'SemNeT'

September 12, 2026

Title Methods and Measures for Semantic Network Analysis

Version 2.0.0

Date 2026-09-12

Maintainer Alexander P. Christensen <alexpaulchristensen@gmail.com>

Description Implements several functions for the analysis of semantic networks including different network estimation algorithms, partial node bootstrapping (Kenett, Anaki, & Faust, 2014 <doi:10.3389/fnhum.2014.00407>), random walk simulation (Kenett & Austerweil, 2016), and a function to compute global network measures. Significance tests and plotting features are also implemented.

Depends R (>= 3.6.0)

License GPL (>= 3.0)

Encoding UTF-8

LazyData true

Imports methods, utils, stats, graphics, future, future.apply, progressr, igraph, qgraph, car, broom, effects

Suggests testthat (>= 3.0.0), shiny (>= 1.8.0), shinyjs, shinyalert, DT, promises, spreadr, shinyMatrix, shinyBS, readxl, foreign, R.matlab, tools, ggplot2

URL https://github.com/AlexChristensen/SemNeT

BugReports https://github.com/AlexChristensen/SemNeT/issues

NeedsCompilation yes

Config/testthat/edition 3

Config/roxygen2/version 8.1.0

Author Alexander P. Christensen [aut, cre] (ORCID: <https://orcid.org/0000-0002-9798-7037>), Yoed N. Kenett [aut, ctb] (ORCID: <https://orcid.org/0000-0003-3872-7689>)

Repository CRAN

Date/Publication 2026-09-12 15:10:08 UTC

Contents

SemNeT-package . . . 3
animals.freq . . . 3
aspl . . . 4
bootSemNeT . . . 5
bootstrap_SemNeT . . . 6
bootstrap_test_SemNeT . . . 8
cc . . . 9
cn . . . 10
compare_nets . . . 12
compare_networks . . . 13
convert2cytoscape . . . 14
convert2igraph . . . 15
equate . . . 16
finalize . . . 17
forward_flow . . . 18
net.high . . . 20
net.low . . . 20
NRW . . . 21
one.result . . . 22
open.binary . . . 22
open.clean . . . 23
open.group . . . 23
permutation_SemNeT . . . 24
PF . . . 26
plot.bootSemNeT . . . 27
plot.bootstrap_SemNeT . . . 27
Q . . . 28
randnet.test . . . 30
random_network_test . . . 30
random_walk . . . 31
randwalk . . . 33
read_uploaded_file . . . 34
response.analysis . . . 35
responses_to_binary . . . 35
response_analysis . . . 36
semantic_network_measures . . . 37
semnetmeas . . . 38
SemNeTShiny . . . 39
sim.fluency . . . 40
similarity . . . 40
simulate_fluency . . . 41
test.bootSemNeT . . . 42
TMFG . . . 43
two.result . . . 44
vignette.plots . . . 44

Index **45**

SemNeT-package *SemNeT–package*

Description

Implements several functions for the analysis of semantic networks including partial node bootstrapping (Kenett, Anaki, & Faust, 2014), random walk simulation (Kenett & Austerweil, 2016), and a function to compute global network measures. Significance tests and plotting features are also implemented.

Author(s)

Alexander P. Christensen <alexpaulchristensen@gmail.com> & Yoed N. Kenett <yoedkenett@gmail.com>

References

Christensen, A. P., Kenett, Y. N., Cotter, K. N., Beaty, R. E., & Silvia, P. J. (2018). Remotely close associations: Openness to experience and semantic memory structure. *European Journal of Personality*, *32*, 480-492.

Kenett, Y. N., Anaki, D., & Faust, M. (2014). Investigating the structure of semantic networks in low and high creative persons. *Frontiers in Human Neuroscience*, *8*, 407.

Kenett, Y. N., & Austerweil, J. L. (2016). Examining search processes in low and high creative individuals with random walks. In *Paper presented at the proceedings of the 38th annual meeting of the cognitive science society*. Austin, TX.

See Also

Useful links:

- https://github.com/AlexChristensen/SemNeT
- Report bugs at https://github.com/AlexChristensen/SemNeT/issues

---

animals.freq *Frequency of Animal Responses*

Description

Frequency of animal responses from Christensen & Kenett (2019). These frequencies are used to generate data in the sim.fluency function.

Usage

```
data(animals.freq)
```

Format

animals.freq (vector, length = 367)

Examples

```
data("animals.freq")
```

`aspl` *Average Shortest Path Length*

Description

Computes the global average shortest path length of a network

Usage

```
aspl(A, weighted = FALSE)

ASPL(A, weighted = FALSE)
```

Arguments

`A`	Matrix or data frame. An adjacency matrix of network data
`weighted`	Boolean. Is the network weighted? Defaults to `FALSE`. Set to `TRUE` for weighted measures

Value

Returns the network's ASPL. The per-node average shortest path length and eccentricity are attached as the `"node"` and `"eccentricity"` attributes, rather than being computed and discarded

Author(s)

Alexander Christensen <alexpaulchristensen@gmail.com>

References

Rubinov, M., & Sporns, O. (2010). Complex network measures of brain connectivity: Uses and interpretations. *NeuroImage*, *52*, 1059-1069.

Examples

```
# Pearson's correlation only for CRAN checks
fluency_data <- matrix(sample(0:1, 1000, replace = TRUE), nrow = 100, ncol = 10)
A <- suppressWarnings(tmfg(similarity(fluency_data, method = "cor")))

# Unweighted
network_aspl <- aspl(A)
```

bootSemNeT *Bootstrapped Semantic Network Analysis (Legacy)*

Description

Backward-compatible wrapper for bootstrap_SemNeT – bootSemNeT() was renamed bootstrap_SemNeT() in SemNeT 2.0.0. The returned object gets both the old class ("bootSemNeT", so plot() still dispatches to plot.bootSemNeT) and the new one ("bootstrap_SemNeT", so it can also be passed straight into bootstrap_test_SemNeT or bootstrap_test_SemNeT's own legacy wrapper, test.bootSemNeT)

Usage

```
bootSemNeT(
  ...,
  method = c("CN", "NRW", "PF", "TMFG"),
  methodArgs = list(),
  type = c("case", "node"),
  prop = 0.5,
  sim,
  weighted = FALSE,
  iter = 1000,
  cores
)
```

Arguments

...	Matrices or data frames. Response matrices, one per group, e.g. bootSemNeT(low, high) – names are captured from the given expressions when not passed as named arguments, matching the original's own substitute()-based name capture
method	Character. See bootstrap_SemNeT. Unlike the original – which had no working default for this argument at all (switch() on an unresolved multi-value default silently required a real value every call) – omitting it now falls through to bootstrap_SemNeT()'s own default, "tmfg"
methodArgs	List. See bootstrap_SemNeT's method_args. minCase inside methodArgs is pulled out into minimum_cases, matching where the original actually used it
type	Character. See bootstrap_SemNeT. Same no-working-default situation as method above; omitting it now defaults to "case"
prop	Numeric. See bootstrap_SemNeT
sim	Character. See bootstrap_SemNeT
weighted	Boolean. Unused – was already dead code in the original bootSemNeT()
iter	Numeric. See bootstrap_SemNeT
cores	Numeric. See bootstrap_SemNeT. Unlike the original, defaults to sequential processing rather than auto-detecting cores (see randwalk's equivalent note)

Value

See bootstrap_SemNeT

Author(s)

Alexander Christensen <alexpaulchristensen@gmail.com>

---

bootstrap_SemNeT *Bootstrapped Semantic Network Analysis*

Description

Repeatedly re-estimates each group's semantic network from resampled data (dropping a proportion of nodes, or resampling participants with replacement) to build a bootstrap distribution of aspl, cc, and q for every group

Usage

```
bootstrap_SemNeT(
  ...,
  method = c("tmfg", "cn", "nrw", "pf"),
  method_args = list(),
  type = c("case", "node"),
  prop = 0.5,
  sim = "cosine",
  minimum_cases = 2,
  iter = 1000,
  cores = 1,
  keep_networks = FALSE
)
```

Arguments

...	Named matrices or data frames. At least one response matrix, one per group, e.g. bootstrap_SemNeT(low = low_data, high = high_data)
method	Character. Network estimation method to use: "tmfg", "cn", "nrw", or "pf" – see tmfg, cn, nrw, and pf. Defaults to "tmfg"
method_args	List. Additional arguments passed on to the chosen method's own function. Defaults to list() (every method's own defaults apply)
type	Character. Type of bootstrap to perform: "case" resamples participants with replacement; "node" keeps every participant but randomly drops nodes (see prop) – this requires every group to already share the same nodes. Defaults to "case"
prop	Numeric. **Only** for type = "node". Proportion of nodes to keep in each resampled network. Defaults to .50

sim	Character. Similarity measure to use, method = "tmfg" only. Defaults to "cosine". See similarity for other options
minimum_cases	Numeric. **Only** for method = "tmfg" and type = "case". Minimum number of participants required to have given a response for that response to be kept, via finalize, before equating every group's vocabulary via equate. Defaults to 2
iter	Numeric. Number of bootstrapped samples to generate per group. Defaults to 1000
cores	Numeric. Number of processing cores to use. Defaults to 1 (sequential)
keep_networks	Boolean. Whether every bootstrapped network itself should be kept in the result, in addition to its measures. A full set of iter dense networks per group can be large (an N-node network is N x N numbers, iter times, per group); most uses only need measures/summary/edges. Defaults to FALSE

Value

Returns a list of class "bootstrap_SemNeT":

groups	A named list, one entry per group in ..., each a list of: networks (the iter bootstrapped networks, or NULL unless keep_networks = TRUE), measures (a 3 x iter matrix of aspl/cc/q values), edges (each bootstrapped network's edge count, length iter – always present, regardless of keep_networks), and summary (a data frame of each measure's mean, standard deviation, standard error, and 95% confidence interval across the iter samples)
meta	A list of prop (only for type = "node"), iter, type, and method

Author(s)

Alexander Christensen <alexpaulchristensen@gmail.com>

Examples

```
# Simulate datasets
one <- simulate_fluency(20)
two <- simulate_fluency(20)


# Bootstrap node-drop (partial) networks
boot_result <- bootstrap_SemNeT(
  one = one, two = two, prop = .50, iter = 100, cores = 2,
  method = "tmfg", type = "node"
)

# Bootstrap case-resampled networks -- equating (see `equate()`) requires
# named responses, so this uses real word responses instead
low <- open.clean[which(open.group == "Low"), ]
high <- open.clean[which(open.group == "High"), ]
boot_result <- bootstrap_SemNeT(
  low = low, high = high, iter = 100, cores = 2,
  method = "tmfg", type = "case", minimum_cases = 2
)
```

bootstrap_test_SemNeT *Statistical Tests for* bootstrap_SemNeT

Description

Computes statistical tests comparing groups' bootstrapped network measures from one or more bootstrap_SemNeT results

Usage

```
bootstrap_test_SemNeT(
  ...,
  test = c("t-test", "ANOVA", "ANCOVA"),
  measures = c("aspl", "cc", "q"),
  formula = NULL,
  groups = NULL
)
```

Arguments

...	One or more bootstrap_SemNeT objects from bootstrap_SemNeT, all describing the same groups. More than one is only useful for type = "node" results at different values of prop, compared side by side
test	Character. Type of statistical test: "t-test" (every pairwise comparison, single-factor designs only), "ANOVA", or "ANCOVA" (controls for each bootstrapped network's number of edges as a covariate, via Anova type II/III sums of squares). Defaults to "t-test"
measures	Character. One or more of "aspl", "cc", "q". Defaults to all three
formula	Character. **Only** for test = "ANOVA"/"ANCOVA" with a multi-factor groups design. A formula with the outcome written as "y", e.g. formula = "y ~ gf * caq" for a two-way ANOVA. Defaults to NULL (every column of groups, additively or as the single factor's own levels)
groups	Matrix or data frame. One row per group (matched to bootstrap_SemNeT's group names by row name if present, otherwise by the order groups were originally entered into bootstrap_SemNeT), one column per grouping/design variable. Defaults to NULL (a single factor using each group's own name as its level)

Value

Returns a list of class "bootstrap_test_SemNeT":

test	The test that was run
measures	The measures that were tested

groups	The (possibly defaulted) groups design matrix used
results	A named list, one entry per bootstrap_SemNeT object in ... (named "Case", or "Proportion (<prop>)" for type = "node" results), each a named list with one entry per measures, each a list of: table (the test's own statistics table), post_hoc (a Tukey HSD table, only for "ANOVA"/ "ANCOVA" with a single factor of more than two levels, otherwise NULL), and adjusted_means (only for "ANOVA"/ "ANCOVA"; a single data frame for a one-factor design or a user-supplied interaction formula, otherwise – an additive, multi-column groups design with no formula – a named list of one data frame per factor; NULL for "t-test")

Author(s)

Alexander Christensen <alexpaulchristensen@gmail.com>

Examples

```
# Get openness data
low <- open.clean[which(open.group == "Low"), ]
high <- open.clean[which(open.group == "High"), ]


# Bootstrap networks
boot_result <- bootstrap_SemNeT(
  low = low, high = high, iter = 100, cores = 2,
  type = "case", method = "tmfg"
)

# Compute tests
bootstrap_test_SemNeT(boot_result)
```

---

cc *Clustering Coefficient*

Description

Computes the global clustering coefficient (CC) of a network

Usage

```
cc(A, weighted = FALSE)

CC(A, weighted = FALSE)
```

Arguments

A	Matrix or data frame. An adjacency matrix of network data
weighted	Boolean. Is the network weighted? Defaults to FALSE. Set to TRUE for weighted measures of CC

Value

Returns the network's CC. The per-node clustering coefficient is attached as the "node" attribute, rather than being computed and discarded

Author(s)

Alexander Christensen <alexpaulchristensen@gmail.com>

References

Rubinov, M., & Sporns, O. (2010). Complex network measures of brain connectivity: Uses and interpretations. *NeuroImage*, *52*, 1059-1069.

Examples

```
# Pearson's correlation only for CRAN checks
fluency_data <- matrix(sample(0:1, 1000, replace = TRUE), nrow = 100, ncol = 10)
A <- suppressWarnings(tmfg(similarity(fluency_data, method = "cor")))

# Unweighted
network_cc <- cc(A)
```

---

cn *Community Network Estimation*

Description

Estimates a semantic network using the Community Network method described in Goni et al. (2011)

Usage

```
cn(data, window = 2, alpha = 0.05, enrich = FALSE)

CN(data, window = 2, alpha = 0.05, enrich = FALSE)
```

Arguments

`data`	Matrix or data frame. A preprocessed verbal fluency matrix where rows are participants and columns are the order of their verbal fluency responses. Content may be the raw word responses or an ordered numeric response matrix (see `responses_to_binary()`)
`window`	Numeric. Size of window to look for co-occurrences in. Defaults to `2`
`alpha`	Numeric. Significance value. Defaults to `.05`
`enrich`	Boolean. Should the network be enriched by connecting all nodes within their respective modules? Defaults to `FALSE`

Value

Returns an undirected semantic network

Author(s)

Alexander Christensen <alexpaulchristensen@gmail.com>

References

Goni, J., Arrondo, G., Sepulcre, J., Martincorena, I., de Mendizabal, N. V., Corominas-Murtra, B., ... & Villoslada, P. (2011). The semantic organization of the animal category: Evidence from semantic verbal fluency and network theory. *Cognitive Processing*, *12*, 183-196.

Examples

```
# Get data
data <- open.clean

# Organize group data
group <- open.group
low <- data[which(group == "Low"), ]
high <- data[which(group == "High"), ]

## Not run:
# Compute networks
low_net <- cn(low)
high_net <- cn(high)

## End(Not run)
```

compare_nets *Compare Graphs (Legacy)*

Description

Backward-compatible wrapper for compare_networks – compare_nets() was renamed compare_networks() in SemNeT 2.0.0

Usage

```
compare_nets(
  ...,
  title,
  config,
  placement = c("match", "default"),
  weighted = FALSE,
  qgraph.args = list()
)
```

Arguments

...	Matrices or data frames. Networks to plot, e.g. compare_nets(low, high) – names are captured from the given expressions when not passed as named arguments, matching the original's own substitute()-based name capture
title	List. See compare_networks
config	Character. See compare_networks
placement	Character. See compare_networks
weighted	Boolean. See compare_networks
qgraph.args	List. See compare_networks's qgraph_args

Value

See compare_networks

Author(s)

Alexander Christensen <alexpaulchristensen@gmail.com>

compare_networks *Plot Networks for Comparison*

Description

Uses qgraph to plot networks side by side. Accepts any number of networks and arranges the plots using `floor(sqrt(length(...)))` side-by-side columns (e.g. 3 networks: 1 row of 3; 6 networks: 2 rows of 3; 9 networks: 3 rows of 3)

Usage

```
compare_networks(
  ...,
  title,
  config = "spring",
  placement = c("default", "match"),
  weighted = FALSE,
  qgraph_args = list()
)
```

Arguments

`...`	Named matrices or data frames. At least two network adjacency matrices, e.g. `compare_networks(low = low_net, high = high_net)`
`title`	List. Titles for each plot, in the same order as `...`. Defaults to the names given in `...`
`config`	Character. Defaults to `"spring"`. See qgraph for more options
`placement`	Character. How should nodes be placed when comparing networks? • `"default"` — places nodes in the default `config` positions • `"match"` — places nodes in the same position across every network (requires every network to have the same nodes) Defaults to `"default"`
`weighted`	Boolean. Should networks be plotted with weights? Defaults to `FALSE`. Unweighted networks are often more aesthetically representative of the network's structure
`qgraph_args`	List. Additional arguments passed on to qgraph. Defaults to `list()`

Value

Plots networks side by side using qgraph; returns `NULL` invisibly

Author(s)

Alexander Christensen <alexpaulchristensen@gmail.com>

References

Epskamp, S., Cramer, A. O. J., Waldorp, L. J., Schmittmann, V. D., & Borsboom, D. (2012). qgraph: Network visualizations of relationships in psychometric data. *Journal of Statistical Software*, *48*, 1-18.

Examples

```
# Simulate datasets
one <- simulate_fluency(10)
two <- simulate_fluency(10)

# Compute similarity matrices
cos1 <- similarity(one, method = "cosine")
cos2 <- similarity(two, method = "cosine")

# Compute networks
net1 <- suppressWarnings(tmfg(cos1))
net2 <- suppressWarnings(tmfg(cos2))

# Compare networks
compare_networks(one = net1, two = net2, config = "spring")

# Change edge colors
compare_networks(
  one = net1, two = net2, config = "spring",
  qgraph_args = list(edge.color = "blue")
)
```

---

convert2cytoscape *Convert an Adjacency Matrix to Cytoscape Format*

Description

Converts an adjacency matrix to Cytoscape's sparse edge-list format

Usage

```
convert2cytoscape(A)
```

Arguments

A	Matrix or data frame. A cleaned, finalized network matrix ready to be visualized

Value

A sparse matrix formatted for Cytoscape, with columns `nodeTO`, `nodeFROM`, and `nodeWEIGHT`

Author(s)

Alexander Christensen <alexpaulchristensen@gmail.com>

References

Shannon, P., Markiel, A., Ozier, O., Baliga, N. S., Wang, J. T., Ramage, D., ... & Ideker, T. (2003). Cytoscape: A software environment for integrated models of biomolecular interaction networks. *Genome Research*, *13*, 2498-2504.

Examples

```
# Simulate toy binarized datasets
one <- matrix(sample(0:1, 100, replace = TRUE), nrow = 10, ncol = 10)
two <- matrix(sample(0:1, 100, replace = TRUE), nrow = 10, ncol = 10)

# Compute similarity matrices
cos1 <- similarity(one, method = "cosine")
cos2 <- similarity(two, method = "cosine")

# Compute networks
net1 <- suppressWarnings(tmfg(cos1))
net2 <- suppressWarnings(tmfg(cos2))

# Convert to Cytoscape format
cyto1 <- convert2cytoscape(net1)
cyto2 <- convert2cytoscape(net2)

# Write to .csv
write.csv(cyto1, file.path(tempdir(), "cyto1.csv"), row.names = FALSE)
write.csv(cyto2, file.path(tempdir(), "cyto2.csv"), row.names = FALSE)
```

---

convert2igraph *Convert a Network to igraph's Format*

Description

Converts an adjacency matrix into `igraph`'s network format

Usage

```
convert2igraph(A)
```

Arguments

`A`	Matrix or data frame. An adjacency (network) matrix

Value

Returns a network in `igraph`'s format

Author(s)

Alexander Christensen <alexpaulchristensen@gmail.com>

Examples

```
# Pearson's correlation only for CRAN checks
fluency_data <- matrix(sample(0:1, 500, replace = TRUE), nrow = 50, ncol = 10)
A <- suppressWarnings(tmfg(similarity(fluency_data, method = "cor")))

igraph_network <- convert2igraph(A)
```

---

equate *Equate Groups*

Description

Equates multiple binary response matrices to one another, so that every group retains exactly the responses common to *all* of them

Usage

```
equate(...)
```

Arguments

`...`	Named binary response matrices or data frames. At least two, e.g. `equate(low = low_data, high = high_data)`

Value

A named list of the equated binary response matrices, one per input, each containing the same responses (columns) in the same order

Author(s)

Alexander Christensen <alexpaulchristensen@gmail.com>

Examples

```
# Obtain binary data
bin <- open.binary

# Finalize each group
mat1 <- finalize(bin[c(1:5), ])
mat2 <- finalize(bin[c(6:10), ])

# Equate the groups
eq <- equate(mat1 = mat1, mat2 = mat2)
```

```
# Obtain respective equated response matrices
eq_mat1 <- eq$mat1
eq_mat2 <- eq$mat2
```

`finalize` *Finalize a Response Matrix*

Description

Finalizes a binary response matrix by keeping only responses given by at least a minimum number of participants

Usage

```
finalize(data, minimum_cases = 2)
```

Arguments

`data`	Matrix or data frame. A binary response matrix (participants x responses)
`minimum_cases`	Numeric. Minimum number of participants required to have given a response for that response to be kept. Defaults to 2

Value

A binary response matrix containing only responses given by at least `minimum_cases` participants

Author(s)

Alexander Christensen <alexpaulchristensen@gmail.com>

Examples

```
# Obtain binary data
bin <- open.binary

# Finalize
final <- finalize(bin)
```

`forward_flow` *Forward Flow*

Description

Computes the "forward flow" of a sequence of verbal fluency or free association responses (Gray et al., 2019): for each response after the first, the mean semantic distance (1 - cosine similarity) between that response and every response that preceded it, using word vectors from a user-supplied embedding matrix.

Unlike the pre-2.0.0 version of this function, no semantic space is downloaded automatically – CRAN policy prohibits a package silently authenticating with and downloading files from an external service at runtime. Bring your own pre-trained (e.g. GloVe, word2vec) or custom word embeddings instead

Usage

```
forward_flow(
  response_matrix,
  embeddings,
  min_response = 3,
  max_response = NULL,
  type = c("fluency", "free"),
  cores = 1
)
```

Arguments

`response_matrix`	Matrix or data frame. For `type = "fluency"`: participants (rows) by responses (columns), one word per cell, in the order each participant gave them. Row names, if present, are used as participant IDs. For `type = "free"`: long-format data with columns named `"ID"`, `"Cue"`, and `"Response"`, one response per row, in the order each participant gave them for each cue
`embeddings`	Matrix or data frame. A word-embedding matrix: one row per vocabulary term (row names are the terms themselves), one column per embedding dimension. Not bundled or downloaded by this package – supply your own pre-trained or custom vectors. A response with no matching row is treated as missing (excluded from the distances it would have contributed to, rather than corrupting every flow value it touches)
`min_response`	Numeric. Minimum number of responses a participant (`type = "fluency"`) or participant-cue pair (`type = "free"`) must have to compute forward flow for it. Defaults to `3`
`max_response`	Numeric. Maximum number of responses to use, keeping only the first `n`. Useful for decoupling forward flow from fluency (participants who simply produced more responses). Defaults to `NULL`, which uses every response
`type`	Character. `"fluency"` or `"free"`. Defaults to `"fluency"`
`cores`	Numeric. Number of computer processing cores to use. Defaults to `1`

Value

For `type = "fluency"`, a list with:

`mean_flow`	A data frame of each participant's average forward flow
`response_flow`	A list, one element per participant, of that participant's response-by-response forward flow

For `type = "free"`, a list with:

`mean_flow`	A data frame of each participant's average forward flow, across all of their cues
`mean_response_flow`	A data frame of each participant-cue pair's average forward flow
`response_flow`	A list, one element per participant (itself a list with one element per cue), of that cue's response-by-response forward flow

Author(s)

Alexander Christensen <alexpaulchristensen@gmail.com> & Brendan Baker <bsb5477@psu.edu>

References

Gray, K., Anderson, S., Chen, E. E., Kelly, J. M., Christian, M. S., Patrick, J., ... & Lewis, K. (2019). "Forward flow": A new measure to quantify free thought and predict creativity. *American Psychologist*, *74*(5), 539-554.

Beaty, R. E., Zeitlen, D. C., Baker, B. S., & Kenett, Y. N. (2021). Forward flow and creative thought: Assessing associative cognition and its role in divergent thinking. *Thinking Skills and Creativity*, 100859.

Examples

```
# A tiny toy embedding space and toy fluency data -- illustrates the
# interface only; use real pre-trained vectors (GloVe, word2vec, etc.) in
# practice
set.seed(1)
words <- c("dog", "cat", "wolf", "car", "truck", "bus")
embeddings <- matrix(rnorm(length(words) * 5), nrow = length(words))
row.names(embeddings) <- words

responses <- matrix(
  c("dog", "wolf", "cat", "car", "truck", "bus"),
  nrow = 2, byrow = TRUE, dimnames = list(c("P1", "P2"), NULL)
)

result <- forward_flow(responses, embeddings, min_response = 2)
result$mean_flow
```

net.high *High Openness to Experience Network*

Description

High openness to experience network from Christensen & Kenett (2019)

Usage

```
data(net.high)
```

Format

net.high (matrix, 160 x 160)

References

Christensen, A. P., & Kenett, Y. N. (2019) Semantic network analysis (SemNA): A tutorial on preprocessing, estimating, and analyzing semantic networks. *PsyArXiv*.

Examples

```
data("net.high")
```

net.low *Low Openness to Experience Network*

Description

Low openness to experience network from Christensen & Kenett (2019)

Usage

```
data(net.low)
```

Format

net.low (matrix, 160 x 160)

References

Christensen, A. P., & Kenett, Y. N. (2019) Semantic network analysis (SemNA): A tutorial on preprocessing, estimating, and analyzing semantic networks. *PsyArXiv*.

Examples

```
data("net.low")
```

NRW *Naive Random Walk Network Estimation*

Description

Estimates a semantic network using the Naive Random Walk method described in Lerner, Ogrocki, and Thomas (2009)

Usage

```
NRW(data, type = c("num", "prop"), threshold = 0)

nrw(data, type = c("num", "prop"), threshold = 0)
```

Arguments

`data`	Matrix or data frame. A preprocessed verbal fluency matrix where rows are participants and columns are the order of their verbal fluency responses. Content may be the raw word responses or an ordered numeric response matrix (see `responses_to_binary()`)
`type`	Character. Type of `threshold` to apply. One of: • `"num"` — Minimum number of co-occurrences • `"prop"` — Minimum proportion of co-occurrences Defaults to `"num"`
`threshold`	Numeric. Value of the minimum number or proportion of co-occurrences. Defaults to `0` for both `"num"` and `"prop"`

Value

Returns an undirected semantic network

Author(s)

Alexander Christensen <alexpaulchristensen@gmail.com>

References

Lerner, A. J., Ogrocki, P. K., & Thomas, P. J. (2009). Network graph analysis of category fluency testing. *Cognitive and Behavioral Neurology*, *22*, 45-52.

Examples

```
# Get data
data <- open.clean

# Organize group data
group <- open.group
```

```
low <- data[which(group == "Low"), ]
high <- data[which(group == "High"), ]

# Compute networks
low_net <- nrw(low)
high_net <- nrw(high)
```

`one.result` *Simulated Result for Dataset One*

Description

A result of `bootSemNeT` from a simulated dataset

Usage

```
data(one.result)
```

Format

one.result (list, length = 4)

Examples

```
data("one.result")
```

`open.binary` *Binary response Matrices (Openness and Verbal Fluency)*

Description

Binary response matrices for the Animals verbal fluency data (n = 516) from Christensen et al. (2018).

Usage

```
data(open.binary)
```

Format

open.binary (matrix, 516 x 367)

References

Christensen, A. P., Kenett, Y. N., Cotter, K. N., Beaty, R. E., & Silvia, P. J. (2018). Remotely close associations: Openness to experience and semantic memory structure. *European Journal of Personality*, *32*, 480-492.

Examples

```
data("open.binary")
```

`open.clean` *Cleaned response Matrices (Openness and Verbal Fluency)*

Description

Cleaned response matrices for the Animals verbal fluency data (n = 516) from Christensen et al. (2018).

Usage

```
data(open.clean)
```

Format

open.clean (matrix, 516 x 35)

References

Christensen, A. P., Kenett, Y. N., Cotter, K. N., Beaty, R. E., & Silvia, P. J. (2018). Remotely close associations: Openness to experience and semantic memory structure. *European Journal of Personality*, *32*, 480-492.

Examples

```
data("open.clean")
```

`open.group` *Groups for Openness and Verbal Fluency*

Description

Groups for the Animals verbal fluency data (n = 516) from Christensen et al. (2018; see also `open.clean`).

Usage

```
data(open.group)
```

Format

open.group (vector, length = 516)

References

Christensen, A. P., Kenett, Y. N., Cotter, K. N., Beaty, R. E., & Silvia, P. J. (2018). Remotely close associations: Openness to experience and semantic memory structure. *European Journal of Personality*, *32*, 480-492.

Examples

```
data("open.group")
```

`permutation_SemNeT` *Permutation Test for Network Measures*

Description

Computes a permutation test to determine whether there's a difference in a global network measure between two samples

Usage

```
permutation_SemNeT(
  sample_1 = NULL,
  sample_2 = NULL,
  method = c("tmfg", "cn", "nrw", "pf"),
  method_args = list(),
  sim = "cosine",
  minimum_cases = 2,
  measure = c("aspl", "cc", "q"),
  alternative = c("two_tailed", "less", "greater"),
  iter = 1000,
  cores = 1,
  previous = NULL,
  keep_networks = FALSE
)
```

Arguments

`sample_1`	Matrix or data frame. Response matrix to be compared with `sample_2`
`sample_2`	Matrix or data frame. Response matrix to be compared with `sample_1`
`method`	Character. Network estimation method to use: `"tmfg"`, `"cn"`, `"nrw"`, or `"pf"` – see `tmfg`, `cn`, `nrw`, and `pf`
`method_args`	List. Additional arguments passed on to the chosen `method`'s own function. Defaults to `list()` (every method's own defaults apply)
`sim`	Character. Similarity measure to use, `method = "tmfg"` only. Defaults to `"cosine"`. See `similarity` for other options

minimum_cases	Numeric. **Only** for method = "tmfg". Minimum number of participants required to have given a response for that response to be kept, via finalize, before equating the two samples' vocabularies via equate. Defaults to 2
measure	Character. Global network measure to compare: one of "aspl", "cc", or "q"
alternative	Character. Alternative hypothesis: "two_tailed", "less", or "greater". Defaults to "two_tailed"
iter	Numeric. Number of permuted samples to generate. Defaults to 1000
cores	Numeric. Number of processing cores to use. Defaults to 1 (sequential)
previous	A permutation_SemNeT object from a previous call. Reuses its already-permuted networks to test a different measure without regenerating and re-estimating every permuted sample from scratch. Requires that original call to have used keep_networks = TRUE. Defaults to NULL
keep_networks	Boolean. Whether every permuted sample's estimated networks should be kept in the result, in addition to the measure values already computed from them. Only needed to test a second measure later via previous – a full set of iter dense networks per sample can be large, and no other use needs them. Defaults to FALSE

Value

Returns a list of class "permutation_SemNeT":

result	A data frame with sample_1's and sample_2's observed measure, the permutation *p*-value, and the direction of the (in)equality tested
networks	A list of network_1/network_2, the networks estimated for every permuted sample (its first entry is each original, unpermuted sample), or NULL unless keep_networks = TRUE – reusable via previous only when kept
method	The method that was used
differences	The observed measure difference for every permutation

Author(s)

Alexander Christensen <alexpaulchristensen@gmail.com>

Examples

```
# Get openness data -- equating (see `equate()`) requires named responses
low <- open.clean[which(open.group == "Low"), ]
high <- open.clean[which(open.group == "High"), ]


# aspl
perm_aspl <- permutation_SemNeT(
  low, high, method = "tmfg", measure = "aspl", iter = 100, cores = 2,
  keep_networks = TRUE
)

# cc, reusing the same permuted networks
```

```
perm_cc <- permutation_SemNeT(previous = perm_aspl, measure = "cc", cores = 2)
```

PF *Pathfinder Network*

Description

Estimates a pathfinder network using the MST-Pathfinder Network method from Quirin et al. (2008; see also Schvaneveldt, 1990)

Usage

```
PF(data)

pf(data)
```

Arguments

`data`	Matrix or data frame. A binary response matrix, or the original raw word responses

Value

An adjacency matrix. The pre-binarization, Chebyshev-distance-weighted network is attached as the `"weighted"` attribute, rather than being computed and discarded

Author(s)

Alexander Christensen <alexpaulchristensen@gmail.com>

References

Quirin, A., Cordon, O., Guerrero-Bote, V. P., Vargas-Quesada, B., & Moya-Aneon, F. (2008) A quick MST-based algorithm to obtain Pathfinder networks (Inf, n-1). *Journal of the American Society for Information Science and Technology*, *59*, 1912-1924.

Schvaneveldt, R. W. (1990). *Pathfinder associative networks: Studies in knowledge organization*. Norwood, NJ: Ablex Publishing.

Examples

```
# Obtain data
data <- open.binary

# Estimate network
pf_net <- pf(data)
```

plot.bootSemNeT *Plot for* bootSemNeT *(Legacy)*

Description

Backward-compatible wrapper for plot.bootstrap_SemNeT – dispatches for objects returned by the legacy bootSemNeT, which carry the old class alongside the new one

Usage

```
## S3 method for class 'bootSemNeT'
plot(..., groups = NULL, measures = c("ASPL", "CC", "Q"))
```

Arguments

...	An object from bootSemNeT (only the first element is used, matching the original)
groups	Character. Labels for groups in the order they were entered into bootSemNeT. Defaults to NULL (keep the names already on the object)
measures	Character. One or more of "ASPL", "CC", "Q". Defaults to all three

Value

See plot.bootstrap_SemNeT

Author(s)

Alexander Christensen <alexpaulchristensen@gmail.com>

plot.bootstrap_SemNeT *Plot for* bootstrap_SemNeT

Description

Plots the bootstrapped distribution of each requested measure as a raincloud plot (Allen, Poggiali, Whitaker, Marshall, & Kievit, 2018) – a half (flat) violin showing the full distribution's shape, paired with a boxplot and the raw jittered bootstrap values – one panel per measure, grouped side by side

Usage

```
## S3 method for class 'bootstrap_SemNeT'
plot(x, measures = c("aspl", "cc", "q"), ...)
```

Arguments

x	A bootstrap_SemNeT object from bootstrap_SemNeT
measures	Character. One or more of "aspl", "cc", "q". Defaults to all three
...	Currently unused

Value

A ggplot object, returned visibly (so it prints automatically at the console, and renders correctly inside shiny::renderPlot())

Author(s)

Alexander Christensen <alexpaulchristensen@gmail.com>

References

Allen, M., Poggiali, D., Whitaker, K., Marshall, T. R., & Kievit, R. A. (2018). Raincloud plots: a multi-platform tool for robust data visualization. *PeerJ Preprints*, *6*, e27137v1. doi:10.7287/peerj.preprints.27137v1

Examples

```
# Get openness data
low <- open.clean[which(open.group == "Low"), ]
high <- open.clean[which(open.group == "High"), ]


# Bootstrap networks
boot_result <- bootstrap_SemNeT(
  low = low, high = high, iter = 100, cores = 2,
  type = "case", method = "tmfg"
)

# Plot
plot(boot_result)
```

Q *Modularity*

Description

Computes a global modularity measure (Q) using the Louvain community detection algorithm

Usage

```
Q(A)

q(A, resolution = 1, seed = NULL, signed = FALSE)
```

Arguments

A	Matrix or data frame. An adjacency matrix of network data
resolution	Numeric. Adjusts modularity to prefer smaller (resolution > 1) or larger (0 < resolution < 1) communities. Defaults to 1 (standard modularity)
seed	Numeric. A seed for the Louvain algorithm's internal random number generator, for reproducible community detection. Defaults to NULL (not reproducible). Note this is independent of R's own set.seed() – see Details
signed	Boolean. Whether modularity should be computed on the signed network (Gomez et al., 2009), discounting negative edges instead of folding them in as positive. Defaults to FALSE (the original, abs()-based behavior)

Details

Community detection runs via a compiled Louvain implementation (Blondel et al., 2008) whose internal randomness depends only on seed, not on R's global random number generator – unlike igraph::cluster_louvain(), calling set.seed() beforehand has no effect on q()'s reproducibility; pass seed directly instead.

Value

Returns Q, a measure of how well the network partitions into distinct, densely-connected communities

Author(s)

Alexander Christensen <alexpaulchristensen@gmail.com>

References

Blondel, V. D., Guillaume, J. L., Lambiotte, R., & Lefebvre, E. (2008). Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, *2008*, P10008.

Gomez, S., Jensen, P., & Arenas, A. (2009). Analysis of community structure in networks of correlated data. *Physical Review E*, *80*, 016114.

Rubinov, M., & Sporns, O. (2010). Complex network measures of brain connectivity: Uses and interpretations. *NeuroImage*, *52*, 1059-1069.

Examples

```
# Pearson's correlation only for CRAN checks
fluency_data <- matrix(sample(0:1, 1000, replace = TRUE), nrow = 100, ncol = 10)
A <- suppressWarnings(tmfg(similarity(fluency_data, method = "cor")))
```

```
modularity <- q(A, seed = 1)
```

`randnet.test` *Random Network Test (Legacy)*

Description

Backward-compatible wrapper for `random_network_test` – `randnet.test()` was renamed `random_network_test()` in SemNeT 2.0.0

Usage

```
randnet.test(..., iter, cores)
```

Arguments

`...`	Matrices or data frames. Networks to test, e.g. `randnet.test(low, high)` – names are captured from the given expressions when not passed as named arguments, matching the original's own `substitute()`-based name capture
`iter`	Numeric. See `random_network_test`
`cores`	Numeric. See `random_network_test`. Unlike the original, defaults to sequential processing rather than auto-detecting cores (see `randwalk`'s equivalent note)

Value

See `random_network_test`

Author(s)

Alexander Christensen <alexpaulchristensen@gmail.com>

`random_network_test` *Test Against Random Networks*

Description

Performs significance tests for global measures of semantic networks against the global measures of equivalent size (and density) random networks

Usage

```
random_network_test(..., iter = 1000, cores = 1)
```

Arguments

`...`	Named matrices or data frames. Semantic networks to be compared against random networks, e.g. `random_network_test(low = low_net, high = high_net)`
`iter`	Numeric. Number of random networks to generate per input network. Defaults to `1000`
`cores`	Numeric. Number of computer processing cores to use. Defaults to `1`

Value

Returns a named list (one entry per input network) of data frames with the *p*-value, and the random-network distribution's mean and standard deviation, for each of `aspl`, `cc`, and `q`

Author(s)

Alexander Christensen <alexpaulchristensen@gmail.com>

References

Viger, F., & Latapy, M. (2016). Efficient and simple generation of random simple connected graphs with prescribed degree sequence. *Journal of Complex Networks*, *4*, 15-37.

Examples

```
# Get openness data
one <- open.clean[which(open.group == "Low"), ]
two <- open.clean[which(open.group == "High"), ]


# Compute networks
net_one <- cn(one)
net_two <- cn(two)

# Perform random networks test
random_network_test(low = net_one, high = net_two, iter = 100, cores = 2)
```

`random_walk` *Random Walk Simulation*

Description

Simulates random walks over two networks to examine the characteristics of spontaneous spreading activation (see Kenett & Austerweil, 2016)

Usage

```
random_walk(
  A,
  B,
  name_a = "A",
  name_b = "B",
  reps = 20,
  steps = 10,
  iter = 10000,
  cores = 1
)
```

Arguments

A	Matrix or data frame. Adjacency matrix of a semantic network
B	Matrix or data frame. A comparison adjacency matrix of a semantic network
name_a	Character. Label for network A in the output. Defaults to "A"
name_b	Character. Label for network B in the output. Defaults to "B"
reps	Numeric. Number of repetitions of increments in 10 steps. Defaults to 20
steps	Numeric. Number of random steps to begin with. Defaults to 10
iter	Numeric. Number of iterations for each random walk. Defaults to 10000
cores	Numeric. Number of computer processing cores to use. Defaults to 1

Value

A list with long (every simulated summary statistic, all numeric) and short (a compact, correctly-typed summary with Mann-Whitney *p*-values and effect directions)

Author(s)

Alexander Christensen <alexpaulchristensen@gmail.com> and Yoed Kenett <yoedkenett@gmail.com>

References

Kenett, Y. N., & Austerweil, J. L. (2016). Examining search processes in low and high creative individuals with random walks. In *Paper presented at the proceedings of the 38th annual meeting of the cognitive science society*. Austin, TX.

Examples

```
# Simulate toy binarized datasets
one <- matrix(sample(0:1, 100, replace = TRUE), nrow = 10, ncol = 10)
two <- matrix(sample(0:1, 100, replace = TRUE), nrow = 10, ncol = 10)

# Compute similarity matrices
cos1 <- similarity(one, method = "cosine")
cos2 <- similarity(two, method = "cosine")
```

```
# Compute networks
net1 <- suppressWarnings(tmfg(cos1))
net2 <- suppressWarnings(tmfg(cos2))



# Run random walk analysis
rw_results <- random_walk(net1, net2, iter = 100, cores = 2)
```

`randwalk` *Random Walk Simulation (Legacy)*

Description

Backward-compatible wrapper for `random_walk` – `randwalk()` was renamed `random_walk()` in SemNeT 2.0.0

Usage

```
randwalk(A, B, reps = 20, steps = 10, iter = 10000, cores)
```

Arguments

`A, B`	Matrix or data frame. Two networks to compare – names are captured from the given expressions, matching the original's own `substitute()`-based name capture
`reps`	Numeric. See `random_walk`
`steps`	Numeric. See `random_walk`
`iter`	Numeric. See `random_walk`
`cores`	Numeric. See `random_walk`. Unlike the original, defaults to sequential processing rather than auto-detecting cores – matches `random_walk()`'s own default and this rebuild's dependency trim (no more direct `parallel::detectCores()` calls outside `parallel_process()`)

Value

See `random_walk`

Author(s)

Alexander Christensen <alexpaulchristensen@gmail.com>

`read_uploaded_file` *Read a Response or Group File in Any Supported Format*

Description

Multi-format file reader dispatching on a file's extension: `.rds`, `.RData`, `.csv`, `.txt`, `.xls`/`.xlsx` (via `readxl`), `.sav` (via `foreign`), `.mat` (via `R.matlab`), or no/unrecognized extension (read as plain delimited text). Used by the Shiny app's upload widgets, and usable directly for the same purpose outside Shiny

Usage

```
read_uploaded_file(datapath, filename, header = TRUE, sep = ",")
```

Arguments

`datapath`	Character. Path to the file's actual content on disk (e.g. Shiny's `input$file$datapath`)
`filename`	Character. The file's original name, used only to determine its format from its extension – kept separate from `datapath` since Shiny's own upload temp files don't necessarily carry a matching extension
`header`	Boolean. Does the file have a header row (for delimited-text and spreadsheet formats)? Defaults to `TRUE`
`sep`	Character. Field delimiter (for delimited-text formats). Defaults to `","`

Value

The file's contents as a data frame (or the single object it contains, for `.rds`/`.RData`)

Author(s)

Alexander Christensen <alexpaulchristensen@gmail.com>

Examples

```
path <- tempfile(fileext = ".csv")
write.csv(data.frame(x = 1:3, y = c("a", "b", "c")), path, row.names = FALSE)

read_uploaded_file(path, "responses.csv")
```

response.analysis *Response Pattern Analysis (Legacy)*

Description

Backward-compatible wrapper for response_analysis – response.analysis() was renamed response_analysis() in SemNeT 2.0.0

Usage

```
response.analysis(...)
```

Arguments

...	Matrices or data frames. Response matrices, e.g. response.analysis(low, high) – names are captured from the given expressions when not passed as named arguments, matching the original's own substitute()-based name capture

Value

See response_analysis

Author(s)

Alexander Christensen <alexpaulchristensen@gmail.com>

responses_to_binary *Convert Raw Responses to a Binary Response Matrix*

Description

Converts a raw word-response matrix (participants by response order, one word per cell) into a binary (participant by vocabulary) response matrix, alongside a matching matrix recording response order

Usage

```
responses_to_binary(responses)
```

Arguments

responses	Matrix or data frame. A raw word-response matrix: participants (rows) by response order (columns), one word per cell (e.g. verbal fluency data). Empty, missing, and placeholder cells (NA, "", whitespace-only) are treated as no response

Value

A list of class `"responses_to_binary"` with:

`binary`	A participant by vocabulary matrix of 0s (never said) and 1s (said)
`order`	A participant by vocabulary matrix of 0s (never said) and the response's 1-based order (first said, second said, ...) otherwise

Author(s)

Alexander Christensen <alexpaulchristensen@gmail.com>

Examples

```
# Raw fluency responses, 2 participants, up to 3 responses each
responses <- matrix(
  c("dog", "cat", NA, "cat", "wolf", "dog"),
  nrow = 2, byrow = TRUE
)

result <- responses_to_binary(responses)
result$binary
result$order
```

`response_analysis` *Response Analysis*

Description

Computes the difference in the total and unique number of responses between groups (follows Christensen et al., 2018). With more than two groups, every pairwise comparison is returned, since both underlying tests (a two-sample *t*-test and McNemar's test) are inherently pairwise

Usage

```
response_analysis(...)
```

Arguments

`...`	Named matrices or data frames. At least two response matrices (raw word responses or already-binary), one per group, e.g. `response_analysis(low = low_data, high = high_data)`

Value

With exactly two groups, a list with:

`total`	A named vector comparing the total responses given by each participant via a two-sample *t*-test and Cohen's *d*
`unique`	A named vector comparing the number of unique responses provided by each group via McNemar's test

With more than two groups, a named list of such lists, one per pair of groups (named "<group1>_vs_<group2>")

Author(s)

Alexander Christensen <alexpaulchristensen@gmail.com>

References

Christensen, A. P., Kenett, Y. N., Cotter, K. N., Beaty, R. E., & Silvia, P. J. (2018). Remotely close associations: Openness to experience and semantic memory structure. *European Journal of Personality*, *32*, 480-492.

Examples

```
# Obtain data
low <- open.clean[which(open.group == "Low"), ]
high <- open.clean[which(open.group == "High"), ]

# Perform analysis
response_analysis(low = low, high = high)
```

`semantic_network_measures`

Semantic Network Measures

Description

Computes the average shortest path length (`aspl`), clustering coefficient (`cc`), and modularity (`q`) of a network

Usage

```
semantic_network_measures(
  A,
  measures = c("aspl", "cc", "q"),
  weighted = FALSE,
  seed = NULL
)
```

Arguments

A	Matrix or data frame. An adjacency matrix of a network
measures	Character. Which global network measures to compute: any of "aspl", "cc", and "q". Defaults to all three
weighted	Boolean. Should weighted measures be computed? Defaults to FALSE. Set to TRUE for weighted measures
seed	Numeric. Passed on to q for reproducible community detection. Defaults to NULL (not reproducible) – see q's Details

Value

Returns a named numeric vector with one value per requested measure

Author(s)

Alexander Christensen <alexpaulchristensen@gmail.com>

Examples

```
# Simulate a toy binarized dataset
one <- matrix(sample(0:1, 100, replace = TRUE), nrow = 10, ncol = 10)

# Compute a similarity matrix
cos <- similarity(one, method = "cosine")

# Compute a network
net <- suppressWarnings(tmfg(cos))

# Compute global network measures
global_measures <- semantic_network_measures(net)
```

---

semnetmeas *Semantic Network Measures (Legacy)*

Description

Backward-compatible wrapper for semantic_network_measures – semnetmeas() was renamed semantic_network_measures() in SemNeT 2.0.0

Usage

```
semnetmeas(A, meas = c("ASPL", "CC", "Q"), weighted = FALSE)
```

Arguments

A	Matrix or data frame. See semantic_network_measures
meas	Character. One or more of "ASPL", "CC", "Q". Defaults to all three
weighted	Boolean. See semantic_network_measures

Value

A named numeric vector, named to match meas (uppercase), as in the original

Author(s)

Alexander Christensen <alexpaulchristensen@gmail.com>

SemNeTShiny *Shiny App for* SemNeT

Description

Launches an interactive Shiny application for semantic network analysis, covering network estimation, comparison, bootstrap and permutation testing, random walk simulation, and spreading activation. Results are retrieved via the app's own download buttons (as .rds/ .csv/image files), not returned to the R session that launched it – each browser session keeps its own results independently, so the app can be run locally or shared with multiple simultaneous users

Usage

```
SemNeTShiny()
```

Value

Invisibly returns NULL. Launches the app as a side effect; see the app's own "Export" tab for how to retrieve results

Author(s)

Alexander Christensen <alexpaulchristensen@gmail.com>

Examples

```
if(interactive()){
  SemNeTShiny()
}
```

sim.fluency *Simulate Verbal Fluency Data (Legacy)*

Description

Backward-compatible wrapper for simulate_fluency – sim.fluency() was renamed simulate_fluency() in SemNeT 2.0.0

Usage

```
sim.fluency(nodes, cases, random = FALSE)
```

Arguments

nodes	Numeric. See simulate_fluency. Unlike simulate_fluency(), has no default – matches the original, which always required this argument
cases	Numeric. See simulate_fluency. Unlike simulate_fluency(), has no default – matches the original, which always required this argument
random	Boolean. See simulate_fluency

Value

See simulate_fluency

Author(s)

Alexander Christensen <alexpaulchristensen@gmail.com>

similarity *Measures of Similarity*

Description

Computes a similarity matrix between the columns of a binarized verbal fluency or linguistic dataset (see Choi, Cha, & Tappert, 2010 for additional measures)

Usage

```
similarity(
  data,
 method = c("cosine", "angular", "cor", "euclid", "faith", "jaccard", "phi", "rr")
)
```

Arguments

`data`	Matrix or data frame. A binarized dataset of verbal fluency or linguistic data
`method`	Character. Type of similarity measure to compute. One of `"cosine"` (default), `"angular"`, `"cor"`, `"euclid"`, `"faith"`, `"jaccard"`, `"phi"`, or `"rr"`. Defaults to `"cosine"`

Value

A symmetric similarity matrix

Author(s)

Alexander Christensen <alexpaulchristensen@gmail.com>

References

Choi, S. S., Cha, S. H., & Tappert, C. C. (2010). A survey of binary similarity and distance measures. *Journal of Systemics, Cybernetics and Informatics*, *8*, 43-48.

Examples

```
# Simulate a toy binarized dataset
one <- matrix(sample(0:1, 100, replace = TRUE), nrow = 10, ncol = 10)

# Compute a cosine similarity matrix
cos <- similarity(one, method = "cosine")
```

`simulate_fluency` *Simulate a Verbal Fluency Binary Response Matrix*

Description

Simulates verbal fluency data based on the number of nodes desired in the resulting network. Each response's total endorsement count is simulated from a Poisson distribution (see `rpois`), using frequencies from the `animals.freq` data; participants' individual responses are then simulated with a probability of endorsing each response equal to that response's simulated total over the number of participants

Usage

```
simulate_fluency(nodes = 100, cases = 500, random = FALSE)
```

Arguments

`nodes`	Numeric. Number of nodes (unique responses) to simulate. Defaults to `100`
`cases`	Numeric. Number of participants to simulate. Defaults to `500`
`random`	Boolean. Should response frequencies be randomly sampled (with replacement) from `animals.freq`, rather than taken in their existing order? Defaults to `FALSE`

Value

A binary matrix with `cases` rows (participants) and `nodes` columns (responses)

Author(s)

Alexander Christensen <alexpaulchristensen@gmail.com>

Examples

```
# Simulate data for 50 nodes and 200 participants
simulate_fluency(nodes = 50, cases = 200)
```

`test.bootSemNeT` *Statistical Tests for* `bootSemNeT` *(Legacy)*

Description

Backward-compatible wrapper for `bootstrap_test_SemNeT` – `test.bootSemNeT()` was renamed `bootstrap_test_SemNeT()` in SemNeT 2.0.0. Only accepts objects produced by this package's own `bootSemNeT` / `bootstrap_SemNeT`, not a raw object saved from SemNeT < `2.0.0`

Usage

```
test.bootSemNeT(
  ...,
  test = c("ANCOVA", "ANOVA", "t-test"),
  measures = c("ASPL", "CC", "Q"),
  formula = NULL,
  groups = NULL
)
```

Arguments

`...`	One or more `bootSemNeT`/`bootstrap_SemNeT` objects. See `bootstrap_test_SemNeT`
`test`	Character. See `bootstrap_test_SemNeT`
`measures`	Character. One or more of `"ASPL"`, `"CC"`, `"Q"`. Defaults to all three
`formula`	Character. See `bootstrap_test_SemNeT`
`groups`	Matrix or data frame. See `bootstrap_test_SemNeT`

Value

See `bootstrap_test_SemNeT`

Author(s)

Alexander Christensen <alexpaulchristensen@gmail.com>

TMFG *Triangulated Maximally Filtered Graph*

Description

Applies the Triangulated Maximally Filtered Graph (TMFG) filtering method (**please see and cite Massara et al., 2016**). The TMFG method uses a structural constraint that limits the number of zero-order correlations included in the network (3*n* - 6, where *n* is the number of variables). Construction begins by forming a tetrahedron from the four nodes with the largest sum of correlations, then iteratively adds each remaining node to the connected set of three nodes (a triangle) it correlates with most strongly, until every node is included.

Usage

```
TMFG(data, depend = FALSE)

tmfg(data, depend = FALSE)
```

Arguments

`data`	Matrix or data frame. A square association matrix (e.g. a correlation or similarity matrix)
`depend`	Boolean. Is the network a dependency (directed) network? Defaults to `FALSE`. Set to `TRUE` to generate a TMFG-filtered dependency network, where the direction of each edge follows whichever of the two asymmetric values is larger

Details

The TMFG method applies a structural constraint on the network, which restrains the network to retain a certain number of edges (3*n* - 6). The network is composed of 3- and 4-node cliques (a triangle and a tetrahedron, respectively). Notably, TMFG can use any association measure and does not assume the data is multivariate normal.

Value

Returns a TMFG-filtered adjacency matrix

Author(s)

Alexander Christensen <alexpaulchristensen@gmail.com>

References

Christensen, A. P., Kenett, Y. N., Aste, T., Silvia, P. J., & Kwapil, T. R. (2018). Network structure of the Wisconsin Schizotypy Scales-Short Forms: Examining psychometric network filtering approaches. *Behavior Research Methods*, *50*, 2531-2550.

Massara, G. P., Di Matteo, T., & Aste, T. (2016). Network filtering for big data: Triangulated maximally filtered graph. *Journal of Complex Networks*, *5*, 161-178.

Examples

```
# Pearson's correlation only for CRAN checks
fluency_data <- matrix(sample(0:1, 1000, replace = TRUE), nrow = 100, ncol = 10)
A <- suppressWarnings(tmfg(similarity(fluency_data, method = "cor")))
```

`two.result` *Simulated Result for Dataset One and Two*

Description

A result of `bootSemNeT` from two simulated datasets

Usage

```
data(two.result)
```

Format

two.result (list, length = 6)

Examples

```
data("two.result")
```

`vignette.plots` *Plots for Vignette*

Description

Plots for vignette taken from Christensen & Kenett (2019)

Usage

```
data(vignette.plots)
```

Format

vignette.plots (list, length = 3)

References

Christensen, A. P., & Kenett, Y. N. (2019) Semantic network analysis (SemNA): A tutorial on preprocessing, estimating, and analyzing semantic networks. *PsyArXiv*.

Examples

```
data("vignette.plots")
```

Index

∗ **datasets**
animals.freq, 3
net.high, 20
net.low, 20
one.result, 22
open.binary, 22
open.clean, 23
open.group, 23
two.result, 44
vignette.plots, 44

animals.freq, 3, *41*
Anova, *8*
ASPL *(*aspl*)*, 4
aspl, 4, *6*, *37*

bootSemNeT, 5, *22*, *27*, *42*, *44*
bootstrap_SemNeT, *5, 6*, 6, *8*, *27, 28*, *42*
bootstrap_test_SemNeT, *5*, 8, *42*

CC *(*cc*)*, 9
cc, *6*, 9, *37*
CN *(*cn*)*, 10
cn, *6*, 10, *24*
compare_nets, 12
compare_networks, *12*, 13
convert2cytoscape, 14
convert2igraph, 15

equate, *7*, 16, *25*

finalize, *7*, 17, *25*
forward_flow, 18

net.high, 20
net.low, 20
NRW, 21
nrw, *6*, *24*
nrw *(*NRW*)*, 21

one.result, 22
open.binary, 22
open.clean, *23*, 23
open.group, 23

permutation_SemNeT, 24
PF, 26
pf, *6*, *24*
pf *(*PF*)*, 26
plot.bootSemNeT, *5*, 27
plot.bootstrap_SemNeT, *27*, 27

Q, 28
q, *6*, *37, 38*
q *(*Q*)*, 28
qgraph, *13*

randnet.test, 30
random_network_test, *30*, 30
random_walk, 31, *33*
randwalk, *5*, *30*, 33
read_uploaded_file, 34
response.analysis, 35
response_analysis, *35*, 36
responses_to_binary, 35
rpois, *41*

semantic_network_measures, 37, *38, 39*
SemNeT, *39*
SemNeT *(*SemNeT-package*)*, 3
SemNeT-package, 3
semnetmeas, 38
SemNeTShiny, 39
sim.fluency, *3*, 40
similarity, *7*, *24*, 40
simulate_fluency, *40*, 41

test.bootSemNeT, *5*, 42
TMFG, 43
tmfg, *6*, *24*
tmfg *(*TMFG*)*, 43
two.result, 44

vignette.plots, 44