

Package ‘psychnets’

July 30, 2026

Title Tidy Clean-Room Psychological Network Modeling

Version 0.4.3

Description Provides clean-room implementations for estimating psychometric network models, including correlation and partial-correlation networks, Gaussian graphical models with extended Bayesian information criterion (EBIC) regularization, nonparanormal and stepwise selection variants, information-filtering networks (the triangulated maximally filtered graph and the local-global inverse covariance), relative-importance networks, and Ising and mixed graphical models
<doi:10.3758/s13428-017-0862-1> <doi:10.1007/978-3-031-54464-4_19>. All methods are implemented from first principles in base R without compiled dependencies and return consistent, tidy outputs. Functions are designed to be transparent and report optimization diagnostics where applicable. For Gaussian graphical models, the graphical lasso stationarity (Karush-Kuhn-Tucker) residual quantifies the deviation of the estimated solution from the optimum of the corresponding convex optimization problem.

License GPL-3

URL <https://pak.dynasite.org/psychnets>,
<https://github.com/mohsaqr/psychnets>

BugReports <https://github.com/mohsaqr/psychnets/issues>

Encoding UTF-8

LazyData true

RoxygenNote 7.3.3

Depends R (>= 3.5.0)

Imports grDevices, graphics, parallel, stats

Suggests cocor, cograph, glasso, glmnet, igraph, IsingFit, knitr, mgm, mvtnorm, networktools, psych, qgraph, rmarkdown, testthat (>= 3.0.0), tna

VignetteBuilder knitr

Config/testthat/edition 3

NeedsCompilation no

Author Mohammed Saqr [aut, cre, cph],
 Sonsoles López-Pernas [aut]
Maintainer Mohammed Saqr <saqr@saqr.me>
Repository CRAN
Date/Publication 2026-07-30 17:20:02 UTC

Contents

as.data.frame.psychnet	3
as.data.frame.psychnet_bootstrap	4
casedrop_reliability	5
certificate	6
condition	7
cor_auto	7
cor_network	8
dichotomize	9
difference_test	10
ebic_glasso	11
event_frequencies	12
ggm_modselect	14
ggm_support_kkt	15
glasso_kkt	16
glm_lasso_kkt	17
huge_network	18
ising_fit	19
ising_sampler	20
lmg_certificate	21
logo_network	22
mgm_fit	23
network_reliability	25
net_aggregate	26
net_boot	27
net_bridge	28
net_centralities	29
net_clustering	30
net_compare	31
net_crosswalk	32
net_edge_betweenness	33
net_predict	33
net_smallworld	34
net_stability	35
node_predictability	36
pcor_network	36
plot.psychnet	38
plot.psychnet_bootstrap	38
plot.psychnet_bridge	39
plot.psychnet_casedrop	40

plot.psychnet_centrality	41
plot.psychnet_difference	42
plot.psychnet_nct	43
plot.psychnet_reliability	43
plot.psychnet_stability	44
print.psychnet	45
print.psychnet_bootstrap	45
print.psychnet_casedrop	46
print.psychnet_group	46
print.psychnet_moderated	47
print.psychnet_multilevel	48
print.psychnet_nct	48
print.psychnet_redundancy	49
print.psychnet_reliability	49
print.psychnet_result_group	50
print.psychnet_stability	50
psychnet	51
redundancy	53
relimp_network	54
SRL	55
summary.psychnet	57
tmfg_certificate	57
tmfg_network	58
\$.psychnet	59
Index	60

as.data.frame.psychnet
<i>Tidy edge list for a psychnet network</i>

Description

Tidy edge list for a psychnet network

Usage

```
## S3 method for class 'psychnet'  
as.data.frame(x, row.names = NULL, optional = FALSE, ..., include_zero = FALSE)
```

Arguments

- x A psychnet object.
- row.names, optional Ignored (for S3 consistency).
- ... Unused.
- include_zero If TRUE, keep zero-weight (absent) edges. Default FALSE.

Value

A one-row-per-edge data.frame with columns from, to, weight.

Examples

```
fit <- pcor_network(SRL_GPT)
as.data.frame(fit)
as.data.frame(fit, include_zero = TRUE)
```

```
as.data.frame.psychnet_bootstrap
```

Tidy a network bootstrap

Description

Tidy a network bootstrap

Usage

```
## S3 method for class 'psychnet_bootstrap'
as.data.frame(x, row.names = NULL, optional = FALSE, ..., significant = FALSE)
```

Arguments

x	A psychnet_bootstrap object.
row.names, optional	Ignored (S3 consistency).
...	Unused.
significant	If TRUE, return only the edges whose confidence interval excludes zero. Default FALSE (all edges).

Value

The tidy \$edges data frame (one row per edge, with its percentile interval, inclusion proportion, and significant flag).

`casedrop_reliability` *Edge-weight stability coefficient (case-dropping subset bootstrap)*

Description

The edge-vector complement to `net_stability()`. For each drop proportion the network is re-estimated on random case-dropped subsets and the subset edge-weight vector is compared with the full-sample one. The edge-weight CS-coefficient is the largest drop proportion at which the edge-vector correlation stays $\geq$ threshold with probability $\geq$ certainty (Epskamp, Borsboom & Fried 2018).

Usage

```
casedrop_reliability(
  data,
  method = "glasso",
  drop_prop = seq(0.1, 0.9, by = 0.1),
  iter = 100L,
  threshold = 0.7,
  certainty = 0.95,
  cor_method = c("spearman", "pearson", "kendall"),
  labels = NULL,
  ...
)
```

Arguments

<code>data</code>	Numeric data frame or matrix (rows = observations), or a <code>psychnet_group</code> (case-dropped per level).
<code>method</code>	Estimator (see <code>psychnet()</code>). Default "glasso".
<code>drop_prop</code>	Proportions of cases to drop. Default <code>seq(0.1, 0.9, 0.1)</code> .
<code>iter</code>	Subsets per proportion. Default 100.
<code>threshold</code>	Minimum acceptable edge-vector correlation. Default 0.7.
<code>certainty</code>	Probability the correlation must exceed threshold. Default 0.95.
<code>cor_method</code>	Correlation method for the edge-vector comparison: "spearman" (default, robust to the wide range of edge weights), "pearson", or "kendall".
<code>labels</code>	Optional node labels.
<code>...</code>	Passed to the estimator.

Value

A tidy data.frame (class `psychnet_casedrop`), one row per metric per drop proportion, with columns `metric`, `drop_prop`, `mean`, `sd`. The edge-weight CS-coefficient is carried in `attr(x, "cs")` and shown when the result is printed. Visualise it with `plot.psychnet_casedrop()`.

References

Epskamp, S., Borsboom, D., & Fried, E. I. (2018). Estimating psychological networks and their accuracy. *Behavior Research Methods*, 50(1), 195-212.

Examples

```
# `iter` and `drop_prop` are kept small here so the example runs quickly;
# the defaults (iter = 100, drop_prop = seq(0.1, 0.9, 0.1)) are what a real
# reliability assessment should use.
casedrop_reliability(SRL_Claude, iter = 5, drop_prop = c(0.25, 0.5))
```

certificate	<i>Correctness certificate of a fitted network</i>
-------------	--

Description

Every regularized or constrained estimator in psychnet self-certifies: it reports how far the returned network sits from the unique optimum of its own convex objective (a KKT / stationarity residual), or – for the structural methods – whether the graph satisfies the identity that defines it. This verb returns that certificate as a tidy one-row `data.frame`, so correctness is read the same way for every method.

Usage

```
certificate(x, tol = 1e-06)
```

Arguments

x	A psychnet object.
tol	Tolerance below which the fit is flagged certified = TRUE. Default 1e-6.

Details

The residual is near machine zero for a correctly solved problem. cor and pcor have no optimization to certify and report NA.

Value

A one-row `data.frame` with columns `method`, `certificate` (the residual; smaller is better), `kind` ("kkt" for the optimization certificates, "structural" for TMFG/relimp, "none" for cor/pcor), and `certified` (logical: residual at or below tol).

Examples

```
S <- 0.4^abs(outer(1:6, 1:6, "-"))
certificate(ebic_glasso(cor_matrix = S, n = 250))
certificate(tmfg_network(cor_matrix = S))
```

condition	<i>Condition a moderated network at a moderator value</i>
-----------	---

Description

Extracts the effective pairwise network implied by a moderated MGM (`mgm_fit()` with moderators) at a given value of the moderator, mirroring `mgm::condition()`: it applies the AND-rule pre-filter, absorbs the moderator value into the main-effect coefficients, and re-aggregates the pairwise edges.

Usage

```
condition(object, value, rule = NULL)
```

Arguments

object	A psychnet_moderated object from <code>mgm_fit(..., moderators=)</code> .
value	Moderator value to condition on (e.g. 0 or 1 for a binary moderator, or any numeric for a continuous one).
rule	Symmetrization rule; defaults to the rule used at fit time.

Value

A psychnet network object (the moderator node carries no edges).

Examples

```
set.seed(1)
x1 <- stats::rnorm(400); x2 <- stats::rnorm(400)
mod <- rep(0:1, each = 200)
y <- x1 * (mod == 1) + stats::rnorm(400) # x1-y edge only when mod == 1
d <- data.frame(x1 = x1, x2 = x2, y = y, mod = mod)
fit <- mgm_fit(d, types = c("g", "g", "g", "c"), moderators = 4)
condition(fit, value = 0) # no x1-y edge
condition(fit, value = 1) # the x1-y edge appears
```

cor_auto	<i>Automatic correlation matrix (polychoric / polyserial / Pearson)</i>
----------	---

Description

Detects ordinal variables (integer-valued with at most `ordinal_max_levels` levels) and returns the correlation matrix using a polychoric correlation for ordinal-ordinal pairs, a polyserial correlation for ordinal-continuous pairs, and Pearson otherwise, projected to the nearest positive-definite matrix. The base-R counterpart of `qgraph::cor_auto()`; this is the correlation bootnet/qgraph use by default for Likert data.

Usage

```
cor_auto(data, ordinal_max_levels = 7L, na_method = c("pairwise", "listwise"))
```

Arguments

`data` Numeric data frame or matrix (rows = observations).

`ordinal_max_levels` Maximum distinct values for a variable to count as ordinal. Default 7.

`na_method` "pairwise" (default) or "listwise".

Value

A correlation matrix with the variable names as dimnames.

Examples

```
set.seed(1)
z <- matrix(stats::rnorm(300 * 4), 300, 4) %%% chol(0.5^abs(outer(1:4, 1:4, "-")))
x <- apply(z, 2, function(col) as.integer(cut(col, 5))) # 5-level Likert
cor_auto(x)
```

cor_network

Correlation network

Description

Marginal (zero-order) association network: the Pearson correlation matrix with the diagonal removed. Equivalent to bootnet's "cor" default.

Usage

```
cor_network(
  data = NULL,
  cor_matrix = NULL,
  n = NULL,
  cor_method = c("pearson", "spearman", "kendall", "auto"),
  threshold = 0,
  alpha = NULL,
  adjust = "none",
  na_method = c("pairwise", "listwise"),
  labels = NULL
)
```


Arguments

data	Numeric data frame or matrix (rows = observations). Optional if <code>cor_matrix</code> is supplied.
cor_matrix	Optional precomputed correlation matrix; if given, data is ignored and <code>n</code> is required when <code>alpha</code> is used.
n	Sample size (needed for significance testing when <code>cor_matrix</code> is supplied).
cor_method	Correlation method: "pearson" (default), "spearman", "kendall", or "auto" (polychoric/polyserial for ordinal items, the <code>qgraph::cor_auto</code> default; see cor_auto()).
threshold	Correlations with absolute value below this are set to zero. Default 0.
alpha	Significance level; if set, correlations not significant at <code>alpha</code> are zeroed. NULL (default) keeps every edge.
adjust	Multiple-comparison adjustment for the edge p-values (any stats::p.adjust method). Default "none".
na_method	Missing-data handling: "pairwise" (default) uses pairwise-complete correlations projected to the nearest positive-definite matrix; "listwise" drops rows with any NA. Identical when data is complete.
labels	Optional node labels.

Value

A psychnet object whose `$weights` is the thresholded correlation matrix, with `$cor_matrix`, `$n_eff`, `$na_method` (and `$p_values` when `alpha` is used).

Examples

```
x <- matrix(stats::rnorm(200 * 4), 200, 4)
cor_network(x)
cor_network(x, alpha = 0.05, adjust = "BH")
```

dichotomize

Dichotomize numeric columns to 0/1

Description

Splits each column of a numeric matrix or data frame into a binary 0/1 variable. This is the usual preprocessing step before fitting an Ising network ([ising_fit\(\)](#), [ising_sampler\(\)](#)) to Likert or other ordinal/continuous data, which require binary input.

Usage

```
dichotomize(data, method = c("median", "mean", "rank"))
```

Arguments

data	Numeric matrix or data frame (rows = observations).
method	Split rule, applied independently to each column: "median" (default) 1 if the value is $\geq$ the column median. "mean" 1 if the value is $>$ the column mean. "rank" 1 for the upper half of the column by rank, giving a balanced (~50/50) split that is robust to ties (useful for coarse Likert items where a median split is badly unbalanced).

Value

An integer matrix of 0/1 values with the same dimensions and dimnames as data.

Examples

```
b <- dichotomize(SRL_GPT, method = "median")
table(b) # values are 0/1 only
```

difference_test

Bootstrapped difference test for edges or centralities

Description

Tests, within a single network, whether two edge weights or two node centralities differ. For every pair it forms the per-resample difference from the stored bootstrap draws, takes the percentile interval of that difference, and flags the pair significant when the interval excludes zero; it also reports the two-sided bootstrap p-value (Epskamp, Borsboom & Fried 2018). This is the within-network counterpart to the edge accuracy intervals reported by [net_boot\(\)](#).

Usage

```
difference_test(boot, type = "edge", ci = NULL, p_adjust = "none")
```

Arguments

boot	A psychnet_bootstrap object from net_boot() .
type	Quantity to compare: "edge" (default), or any centrality measure bootstrapped by net_boot() (e.g. "strength", "expected_influence").
ci	Confidence level for the difference interval. Defaults to the level used by the bootstrap object.
p_adjust	Multiple-comparison adjustment for the pairwise p-values (any stats::p.adjust method). Default "none".

Value

A tidy data frame, one row per pair, with `item1`, `item2`, the two observed values, their observed difference, the percentile interval of the bootstrap difference (lower, upper), the two-sided `p_value`, and a logical significant.

Examples

```
set.seed(1)
x <- matrix(stats::rnorm(150 * 5), 150, 5) %% chol(0.4^abs(outer(1:5, 1:5, "-")))
colnames(x) <- paste0("V", 1:5)
bs <- net_boot(x, n_boot = 50, cores = 1) # n_boot >= 1000 for real use
difference_test(bs, type = "strength")
```

 ebic_glasso

EBIC-regularized Gaussian graphical model (graphical lasso)

Description

Selects an L1 penalty by the extended BIC (Foygel & Drton 2010) over a log-spaced path, then refits the chosen penalty to machine precision so the returned network is the certified global optimum of the convex objective. Equivalent in purpose to `qgraph::EBICglasso()` / bootnet's "EBICglasso" default, but pure base R and self-certified (see [glasso_kkt\(\)](#)).

Usage

```
ebic_glasso(
  data = NULL,
  cor_matrix = NULL,
  n = NULL,
  gamma = 0.5,
  nlambda = 100L,
  lambda_min_ratio = 0.01,
  threshold = 0,
  cor_method = c("pearson", "spearman", "kendall", "auto"),
  na_method = c("pairwise", "listwise"),
  native = TRUE,
  labels = NULL
)
```

Arguments

<code>data</code>	Numeric data frame or matrix (rows = observations). Optional if <code>cor_matrix</code> is supplied.
<code>cor_matrix</code>	Optional correlation matrix; if given, <code>n</code> is required and <code>data</code> is ignored.
<code>n</code>	Sample size (required when <code>cor_matrix</code> is supplied).
<code>gamma</code>	EBIC hyperparameter. Default 0.5.

nlambda	Number of penalties on the path. Default 100.
lambda_min_ratio	Smallest penalty as a fraction of the largest. Default 0.01.
threshold	Partial correlations with absolute value below this are set to zero. Default 0.
cor_method	Correlation used when data is supplied: "pearson" (default), "spearman", "kendall", or "auto" (polychoric/polyserial for ordinal items, the qgraph::cor_auto / bootnet default). See cor_auto() .
na_method	Missing-data handling when data is supplied: "pairwise" (default, pairwise-complete correlations + nearest-PD projection) or "listwise" (drop incomplete rows). Identical for complete data.
native	Solver switch. TRUE (default) uses psychnet's own pure-R, dependency-free, self-certified solver. FALSE delegates each fixed-penalty solve to the established glasso Fortran package (in Suggests) for speed and byte-identical glasso/qgraph output, at its looser convergence (the reported \$kkt then shows glasso's tolerance rather than $\sim 1e-11$).
labels	Optional node labels.

Value

A psychnet object whose \$weights is the partial-correlation matrix, with \$precision, \$lambda, \$gamma, \$cor_matrix, \$ebic, \$native, and \$kkt (the stationarity residual of the returned network).

Examples

```
S <- 0.4^abs(outer(1:6, 1:6, "-"))
fit <- ebic_glasso(cor_matrix = S, n = 250)
fit
as.data.frame(fit)
```

event_frequencies	<i>Action frequencies from an event log</i>
-------------------	---

Description

Converts a long event log – one row per event, with an actor, an action, and optionally a session and time – into a frequency table: one row per occasion, one column per distinct action, holding that action's count. The conversion mirrors the "frequency" format of the Nestimate event pipeline. An occasion is one (actor, session) cell; with compute_sessions = TRUE the time column is used to split each actor into sessions wherever the gap between consecutive events exceeds time_threshold. This is the frequency input that [psychnet\(\)](#) builds internally for event data (source = "eventdata").

Usage

```
event_frequencies(
  data,
  actor = "Actor",
  action = "Action",
  session = NULL,
  time = NULL,
  compute_sessions = TRUE,
  time_threshold = 900
)
```

Arguments

<code>data</code>	A long event log (data frame), one row per event.
<code>actor</code>	Column(s) naming the actor / subject. Default "Actor".
<code>action</code>	Column naming the action / state. Default "Action".
<code>session</code>	Optional column(s) naming an explicit session within an actor.
<code>time</code>	Optional column used to compute sessions from gaps (see <code>compute_sessions</code>); it is not used for any temporal model.
<code>compute_sessions</code>	If TRUE, split each actor into sessions from the time gaps (a new session starts when the gap exceeds <code>time_threshold</code>). Default TRUE.
<code>time_threshold</code>	Maximum gap (in the units of time, seconds for a timestamp) between consecutive events before a new session begins. Default 900 (15 minutes), as in Nestimate.

Value

A data.frame with an actor column, a session index, and one integer count column per action (one row per occasion).

Examples

```
ev <- data.frame(
  Actor = rep(c("a", "b"), each = 6),
  Session = rep(rep(1:2, each = 3), 2),
  Action = c("read", "quiz", "read", "quiz", "read", "note",
             "note", "note", "read", "read", "quiz", "quiz"))
event_frequencies(ev, session = "Session")
```

ggm_modselect

*Stepwise Gaussian graphical model selection (ggmModSelect)***Description**

Selects a GGM by extended-BIC model search over edge sets generated from the glasso path, refitting the *unregularized* maximum-likelihood precision on each candidate graph, with an optional stepwise add/drop search. Unlike the graphical lasso, retained edges are not shrunk. Equivalent in purpose to `qgraph::ggmModSelect()`, but pure base R and self-certified via `ggm_support_kkt()`.

Usage

```
ggm_modselect(
  data = NULL,
  cor_matrix = NULL,
  n = NULL,
  gamma = 0,
  stepwise = TRUE,
  nlambdas = 100L,
  lambda_min_ratio = 0.01,
  threshold = 0,
  cor_method = c("pearson", "spearman", "kendall", "auto"),
  na_method = c("pairwise", "listwise"),
  native = TRUE,
  labels = NULL
)
```

Arguments

<code>data</code>	Numeric data frame or matrix (rows = observations). Optional if <code>cor_matrix</code> is supplied.
<code>cor_matrix</code>	Optional correlation matrix; if given, <code>n</code> is required.
<code>n</code>	Sample size (required when <code>cor_matrix</code> is supplied).
<code>gamma</code>	EBIC hyperparameter. Default 0, matching <code>qgraph::ggmModSelect()</code> : because the selected graph is refit with the <i>unregularized</i> MLE (no edge shrinkage), the extra EBIC penalty is not needed, so gamma 0 (plain BIC) is the method's intended setting. (The regularized GGMs <code>ebic_glasso()</code> and <code>huge_network()</code> keep gamma 0.5.)
<code>stepwise</code>	If TRUE (default), refine the best glasso-path graph by a greedy single-edge add/drop search.
<code>nlambdas</code>	Number of glasso penalties scanned for candidate graphs.
<code>lambda_min_ratio</code>	Smallest penalty as a fraction of the largest.
<code>threshold</code>	Partial correlations with absolute value below this are zeroed. Default 0.

cor_method	Correlation used when data is supplied: "pearson" (default), "spearman", "kendall", or "auto" (polychoric/polyserial, the qgraph/bootnet default for ordinal items). See cor_auto() .
na_method	Missing-data handling when data is supplied: "pairwise" (default) or "listwise". See ebic_glasso() .
native	Solver switch for generating candidate supports: TRUE (default) uses the pure-R solver; FALSE delegates to the glasso Fortran package (in Suggests). The reported precision is the unregularized refit either way. See ebic_glasso() .
labels	Optional node labels.

Value

A psychnet object whose \$weights is the partial-correlation matrix, with \$precision, \$support (the selected graph), \$gamma, \$ebic, \$cor_matrix, and \$kkt.

Examples

```
S <- 0.5^abs(outer(1:6, 1:6, "-"))
ggm_modselect(cor_matrix = S, n = 250)
```

ggm_support_kkt	<i>Constrained Gaussian-MRF (graph-restricted MLE) stationarity residual</i>
-----------------	--

Description

Certificate for an *unregularized* Gaussian graphical model whose precision is constrained to a fixed graph (the estimator behind [ggm_modselect\(\)](#) and [logo_network\(\)](#)). The maximum-likelihood / maximum-entropy conditions for a Gaussian Markov random field on a graph G are exact: $W_{ij} = S_{ij}$ for every (i, j) on the graph and on the diagonal ($W = \Theta^{-1}$), and $\Theta_{ij} = 0$ for every (i, j) not on the graph. A near-zero return certifies the constrained optimum with no reference solver.

Usage

```
ggm_support_kkt(theta, cor_matrix, support, active_tol = 1e-08)
```

Arguments

theta	Precision matrix to test.
cor_matrix	Correlation / covariance the model was fit to.
support	Logical p x p matrix; TRUE where an edge is allowed.
active_tol	Magnitude above which an off-support entry counts as a nonzero violation.

Value

Maximum absolute stationarity violation (scalar); 0 = exact optimum.

Examples

```
S <- 0.4^abs(outer(1:6, 1:6, "-"))
fit <- ggm_modselect(cor_matrix = S, n = 250)
ggm_support_kkt(fit$precision, S, fit$support)
```

glasso_kkt

*Graphical-lasso stationarity (KKT) residual***Description**

A dependency-free correctness certificate for a fitted Gaussian graphical model. For the convex objective

$$\min_{\Theta \succ 0} -\log \det \Theta + \text{tr}(S\Theta) + \rho \sum_{i \neq j} |\Theta_{ij}|$$

(off-diagonal penalty), let $W = \Theta^{-1}$. The subgradient optimality conditions are $W_{ii} = S_{ii}$; $W_{ij} - S_{ij} = \rho \text{sign}(\Theta_{ij})$ where $\Theta_{ij} \neq 0$; and $|W_{ij} - S_{ij}| \leq \rho$ otherwise. By strict convexity, a precision matrix with zero violation is the unique global optimum, so a near-zero return certifies correctness independently of any reference solver.

Usage

```
glasso_kkt(theta, cor_matrix, rho, active_tol = 1e-08)
```

Arguments

theta	Precision matrix to test.
cor_matrix	Correlation / covariance the model was fit to.
rho	Scalar penalty.
active_tol	Magnitude above which an off-diagonal entry is "active".

Value

Maximum absolute stationarity violation (scalar); 0 = exact optimum.

Examples

```
S <- 0.5^abs(outer(1:5, 1:5, "-"))
fit <- ebic_glasso(cor_matrix = S, n = 200)
glasso_kkt(fit$precision, S, fit$lambda)
```


glm_lasso_kkt

*Stationarity (KKT) residual of an L1-penalized GLM fit***Description**

Dependency-free correctness certificate for a nodewise lasso, analogous to `glasso_kkt()` for the graphical lasso. With standardized predictors X and fitted mean μ (identity link for gaussian, logistic for binomial), the subgradient conditions are $n^{-1}X_j^\top(y - \mu) = \lambda \text{sign}(\beta_j)$ for active coordinates and $|n^{-1}X_j^\top(y - \mu)| \leq \lambda$ otherwise. Near-zero certifies the penalized-likelihood optimum.

Usage

```
glm_lasso_kkt(
  X,
  y,
  b0,
  beta,
  lambda,
  family = "gaussian",
  weights = NULL,
  active_tol = 1e-08
)
```

Arguments

<code>X</code>	Standardized predictor matrix (mean 0, unit variance columns).
<code>y</code>	Response.
<code>b0</code>	Fitted intercept.
<code>beta</code>	Fitted (standardized) coefficients.
<code>lambda</code>	Penalty.
<code>family</code>	"gaussian" or "binomial".
<code>weights</code>	Optional observation weights (NULL = unweighted).
<code>active_tol</code>	Magnitude above which a coefficient is "active".

Value

Maximum absolute stationarity violation (scalar). Near-zero certifies the fit is at the penalized-likelihood optimum.

Examples

```
set.seed(1)
x <- scale(matrix(stats::rnorm(200 * 3), 200, 3))
y <- as.numeric(x %*% c(0.5, 0, -0.3) + stats::rnorm(200))
fit <- stats::lm.fit(cbind(1, x), y)
glm_lasso_kkt(x, y, fit$coefficients[1], fit$coefficients[-1], lambda = 0)
```

huge_network	<i>Nonparanormal graphical model (huge)</i>
--------------	---

Description

Estimates a Gaussian graphical model after a rank-based nonparanormal transform that relaxes the multivariate-normal assumption, then selects the L1 penalty by EBIC and refits to the certified optimum. Equivalent in purpose to `huge::huge()` (nonparanormal) / bootnet's "huge" default, but pure base R and self-certified via [glasso_kkt\(\)](#) on the transformed correlation.

Usage

```
huge_network(
  data = NULL,
  cor_matrix = NULL,
  n = NULL,
  npn = c("shrinkage", "truncation", "skeptical"),
  gamma = 0.5,
  nlambdas = 100L,
  lambda_min_ratio = 0.01,
  threshold = 0,
  na_method = c("pairwise", "listwise"),
  native = TRUE,
  labels = NULL
)
```

Arguments

<code>data</code>	Numeric data frame or matrix (rows = observations). Optional if <code>cor_matrix</code> is supplied (then the transform is skipped).
<code>cor_matrix</code>	Optional pre-transformed correlation matrix; if given, <code>n</code> is required, <code>data</code> and <code>nnp</code> are ignored.
<code>n</code>	Sample size (required when <code>cor_matrix</code> is supplied).
<code>nnp</code>	Nonparanormal transform: "shrinkage" (default), "truncation", or "skeptical" (Spearman).
<code>gamma</code>	EBIC hyperparameter. Default 0.5.
<code>nlambdas</code>	Number of penalties on the path. Default 100.
<code>lambda_min_ratio</code>	Smallest penalty as a fraction of the largest.
<code>threshold</code>	Partial correlations with absolute value below this are zeroed. Default 0.
<code>na_method</code>	Missing-data handling when data is supplied: "pairwise" (default, with the nonparanormal transform applied per column over observed values) or "listwise". See ebic_glasso() .
<code>native</code>	Solver switch for the glasso path: TRUE (default) uses the pure-R solver; FALSE delegates to the glasso Fortran package (in Suggests). See ebic_glasso() .
<code>labels</code>	Optional node labels.

Value

A psychnet object whose `$weights` is the partial-correlation matrix, with `$precision`, `$lambda`, `$gamma`, `$cor_matrix` (the transformed correlation), `$npn`, `$ebic`, and `$kkt`.

Examples

```
set.seed(1)
x <- matrix(stats::rnorm(300 * 5), 300, 5)
x <- exp(x %*% chol(0.4^abs(outer(1:5, 1:5, "-")))) # break normality
huge_network(x)
```

ising_fit

*Ising network for binary data***Description**

Estimates an Ising model by nodewise L1-penalized logistic regression with EBIC selection, combined by the AND (default) or OR rule. Equivalent in purpose to `IsingFit::IsingFit()`, but pure base R and self-certified: each node's regression reports its stationarity (KKT) residual (see [glm_lasso_kkt\(\)](#)).

Usage

```
ising_fit(
  data,
  gamma = 0.25,
  rule = c("AND", "OR"),
  nlambdas = 100L,
  lambda_min_ratio = 0.01,
  min_sum = NULL,
  weights = NULL,
  na_method = c("pairwise", "listwise"),
  native = TRUE,
  labels = NULL
)
```

Arguments

<code>data</code>	Binary (0/1) data frame or matrix (rows = observations).
<code>gamma</code>	EBIC hyperparameter. Default 0.25.
<code>rule</code>	Edge-combination rule: "AND" (default) or "OR".
<code>nlambdas</code>	Number of penalties per nodewise path. Default 100.
<code>lambda_min_ratio</code>	Smallest penalty as a fraction of the largest.
<code>min_sum</code>	Minimum row sum-score (number of endorsed items); rows below it are dropped before fitting. NULL (default) keeps every row.

weights	Optional non-negative observation weights, one per retained row. NULL (default) is unweighted.
na_method	Missing-data handling: "pairwise" (default) single-imputes each column over its observed values (mode for binary), keeping the full sample; "listwise" drops incomplete rows. Identical for complete data.
native	Solver switch. TRUE (default) uses psychnet's own pure-R, dependency-free, self-certified L1 logistic path (KKT $\sim 1e-9$). FALSE delegates each per-node fit to the glmnet package with the IsingFit EBIC path, so the returned \$weights/\$thresholds byte-match IsingFit::IsingFit() (to $\sim 1e-16$) at the cost of glmnet's looser self-certificate. native = FALSE needs the optional glmnet package (Suggests); weights/min_sum are supported with native = TRUE only.
labels	Optional node labels.

Value

A psychnet object whose \$weights is the symmetric weight matrix, with \$thresholds (node intercepts) and \$kkt (the worst nodewise stationarity residual).

Examples

```
set.seed(1)
z <- matrix(stats::rnorm(400 * 2), 400, 2)
x <- cbind(z[, 1], z[, 1], z[, 2], z[, 2]) + matrix(stats::rnorm(400 * 4), 400)
b <- (x > 0) * 1L
colnames(b) <- paste0("V", 1:4)
ising_fit(b)
```

ising_sampler

Unregularized Ising network for binary data

Description

Estimates an Ising model by *unpenalized* nodewise logistic regression, with optional Wald p-value edge pruning, combined by the AND (default) or OR rule. The unregularized counterpart of [ising_fit\(\)](#); self-certified by the maximum-likelihood score residual (see [glm_lasso_kkt\(\)](#) at $\lambda = 0$).

Usage

```
ising_sampler(
  data,
  rule = c("AND", "OR"),
  alpha = NULL,
  adjust = "none",
  min_sum = NULL,
  weights = NULL,
  na_method = c("pairwise", "listwise"),
  labels = NULL
)
```

Arguments

data	Binary (0/1) data frame or matrix (rows = observations).
rule	Edge-combination rule: "AND" (default) or "OR".
alpha	Significance level for Wald edge pruning; NULL (default) keeps every edge.
adjust	Multiple-comparison adjustment for the edge p-values (any stats::p.adjust method). Default "none".
min_sum	Minimum row sum-score; rows below it are dropped before fitting. NULL (default) keeps every row.
weights	Optional non-negative observation weights, one per retained row. NULL (default) is unweighted.
na_method	Missing-data handling: "pairwise" (default, mode-impute) or "listwise". See ising_fit() .
labels	Optional node labels.

Value

A psychnet object whose \$weights is the symmetric weight matrix, with \$thresholds (node intercepts), \$rule, \$p_values, \$nodewise (for [net_predict\(\)](#)), and \$kkt (worst nodewise score residual).

Examples

```
set.seed(1)
z <- matrix(stats::rnorm(500 * 2), 500, 2)
x <- cbind(z[, 1], z[, 1], z[, 2], z[, 2]) + matrix(stats::rnorm(500 * 4), 500)
b <- (x > 0) * 1L
colnames(b) <- paste0("V", 1:4)
ising_sampler(b)
```

lmg_certificate	<i>Relative-importance (LMG / Shapley) certificate</i>
-----------------	--

Description

By Shapley efficiency, the importance shares a node receives from its predictors sum exactly to that node's full-model R-squared. Returns the maximum absolute deviation from that identity; near zero certifies the decomposition.

Usage

```
lmg_certificate(x)
```

Arguments

x A [psychnet](#) object produced by [relimp_network\(\)](#).

Value

Maximum absolute deviation of incoming-share sums from the per-node R-squared (scalar); 0 = exact decomposition.

Examples

```
S <- 0.4^abs(outer(1:5, 1:5, "-"))
lmg_certificate(relimp_network(cor_matrix = S))
```

logo_network	<i>Local-Global sparse inverse covariance (LoGo)</i>
--------------	--

Description

Estimates a sparse Gaussian graphical model whose conditional-independence structure is the chordal TMFG: the precision is the closed-form Gaussian Markov random field on that graph (Barfuss et al. 2016). Equivalent in purpose to `NetworkToolbox::LoGo()` / bootnet's "LoGo" default, pure base R and self-certified via [ggm_support_kkt\(\)](#) (the precision reproduces S exactly on the TMFG support).

Usage

```
logo_network(
  data = NULL,
  cor_matrix = NULL,
  n = NULL,
  cor_method = c("pearson", "spearman", "kendall", "auto"),
  threshold = 0,
  na_method = c("pairwise", "listwise"),
  labels = NULL
)
```

Arguments

data	Numeric data frame or matrix (rows = observations). Optional if <code>cor_matrix</code> is supplied.
cor_matrix	Optional correlation matrix; if given, <code>n</code> is required.
n	Sample size (recorded on the result; required with <code>cor_matrix</code>).
cor_method	Correlation when data is supplied: "pearson" (default), "spearman", "kendall", or "auto" (polychoric/polyserial; see cor_auto()).
threshold	Partial correlations with absolute value below this are zeroed. Default 0.
na_method	Missing-data handling when data is supplied: "pairwise" (default) or "listwise". See ebic_glasso() .
labels	Optional node labels.

Value

A psychnet object whose `$weights` is the partial-correlation matrix, with `$precision`, `$support` (the TMFG graph), `$cor_matrix`, and `$kkt`.

Examples

```
set.seed(1)
x <- matrix(stats::rnorm(300 * 6), 300, 6)
logo_network(x)
```

mgm_fit	<i>Mixed graphical model</i>
---------	------------------------------

Description

Estimates a mixed graphical model by nodewise L1-penalized regression – a gaussian (linear) lasso for continuous nodes and a logistic lasso for binary nodes – with per-node EBIC selection, combined by the AND rule. Equivalent in purpose to `mgm::mgm()`, but pure base R and self-certified: each node's regression reports its stationarity (KKT) residual (see [glm_lasso_kkt\(\)](#)).

Usage

```
mgm_fit(
  data,
  gamma = 0.25,
  types = NULL,
  nlambda = 100L,
  lambda_min_ratio = 0.01,
  threshold = c("LW", "HW", "none"),
  rule = c("AND", "OR"),
  moderators = NULL,
  weights = NULL,
  na_method = c("pairwise", "listwise"),
  native = TRUE,
  labels = NULL
)
```

Arguments

<code>data</code>	Numeric data frame or matrix (rows = observations); columns are continuous or binary (0/1).
<code>gamma</code>	EBIC hyperparameter. Default 0.25.
<code>types</code>	Optional character vector of node types ("g" gaussian, "c" binary); auto-detected if NULL.
<code>nlambda</code>	Number of penalties per nodewise path. Default 100.

<code>lambda_min_ratio</code>	Smallest penalty as a fraction of the largest.
<code>threshold</code>	Post-selection coefficient threshold: "LW" (default), "HW", or "none", matching <code>mgm::mgm()</code> .
<code>rule</code>	Edge-combination rule: "AND" (default) or "OR".
<code>moderators</code>	Optional single column index of a moderator variable. When supplied, fits a <i>moderated</i> MGM (that variable moderates every pairwise edge) and returns a <code>psychnet_moderated</code> object to be read with <code>condition()</code> . Honours native like the unmoderated path: the default base kernel is pure R and KKT-certified, and covers gaussian and binary nodes; <code>native = FALSE</code> uses <code>glmnet</code> and additionally allows multi-level categorical nodes. weights are not supported in this mode.
<code>weights</code>	Optional non-negative observation weights, one per row of the (NA-prepared) data. NULL (default) is unweighted.
<code>na_method</code>	Missing-data handling: "pairwise" (default) single-imputes each column over its observed values (mean for continuous, mode for binary), keeping the full sample; "listwise" drops incomplete rows.
<code>native</code>	Solver switch. TRUE (default) uses <code>psychnet</code> 's own pure-R, dependency-free, self-certified L1 path (KKT $\sim 1e-9$). FALSE delegates each per-node fit to the <code>glmnet</code> package with <code>mgm</code> 's exact EBIC/LW path (gaussian lasso for continuous nodes, 2-class multinomial lasso for binary nodes), so the returned edge magnitudes byte-match <code>abs(mgm::mgm()\$pairwise\$wadj)</code> (to $\sim 1e-6$) at the cost of <code>glmnet</code> 's looser self-certificate. <code>native = FALSE</code> needs the optional <code>glmnet</code> package (Suggests); weights are supported with <code>native = TRUE</code> only.
<code>labels</code>	Optional node labels.

Value

A `psychnet` object whose `$weights` is the symmetric standardized weight matrix, with `$types` and `$kkt` (the worst nodewise residual). A binary-binary edge carries the sign of its nodewise-logistic coefficient; `mgm::mgm()` reports the same edge as a magnitude only (its sign is undefined for a categorical-categorical interaction), so compare such edges on `abs()`. Continuous columns are standardized internally, binary predictors enter the graph on their 0/1 dummy scale, and binary-response logit coefficients are converted to `mgm`'s two-class multinomial scale before edge aggregation. With these conventions the edge magnitudes match `mgm::mgm` closely for gaussian-gaussian, gaussian-binary, and binary-binary edges alike; weak edges near the EBIC/threshold boundary can still differ in support because the penalty is selected on an independent base-R path.

Examples

```
set.seed(1)
f <- stats::rnorm(400)
g1 <- f + stats::rnorm(400); g2 <- f + stats::rnorm(400)
b1 <- (f + stats::rnorm(400) > 0) * 1L
d <- data.frame(g1 = g1, g2 = g2, b1 = b1, n = stats::rnorm(400))
mgm_fit(d)
```

network_reliability	<i>Split-half reliability of the network edge structure</i>
---------------------	---

Description

Repeatedly splits the sample into two halves, estimates a network on each, and compares their edge-weight vectors. Reports, across splits, the edge-weight correlation between halves plus the mean/median/maximum absolute edge deviation - a psychometric reliability view of the estimated structure.

Usage

```
network_reliability(
  data,
  method = "glasso",
  iter = 100L,
  split = 0.5,
  cor_method = c("pearson", "spearman", "kendall"),
  labels = NULL,
  ...
)
```

Arguments

data	Numeric data frame or matrix (rows = observations), or a psychnet_group (split-half per level).
method	Estimator (see psychnet()). Default "glasso".
iter	Number of split-half iterations. Default 100.
split	Fraction of rows in the first half. Default 0.5.
cor_method	Correlation method for the between-halves edge comparison: "pearson" (default), "spearman", or "kendall".
labels	Optional node labels.
...	Passed to the estimator.

Value

A tidy data.frame (class psychnet_reliability), one row per metric with columns metric, mean, sd, lower, upper. The per-split draws are carried in attr(x, "iterations") for [plot.psychnet_reliability\(\)](#).

Examples

```
# `iter` is kept small here so the example runs quickly; the default
# (iter = 100) is what a real reliability assessment should use.
network_reliability(SRL_Claude, iter = 10)
```

net_aggregate

Aggregate a network's communities into super-nodes

Description

Collapses each community of items into a single super-node. score methods build a composite column per community from the raw data and return a reduced data.frame (re-estimate the macro network with [psychnet\(\)](#) on it). assoc methods summarise each community pair's multivariate association directly and return the macro network as a psychnet.

Usage

```
net_aggregate(
  data,
  communities,
  method = "mean",
  estimator = "glasso",
  scale = TRUE,
  labels = NULL,
  ...
)
```

Arguments

data	A numeric data frame or matrix (rows = observations, columns = items).
communities	Community membership, one entry per item (column): a vector aligned to the columns, or a named vector / list keyed by item label.
method	Aggregation method. Score (return reduced data): "mean" (default), "median", "sum", "pca" (first principal component), "factor" (1-factor score, falls back to PCA for communities of < 3 items), "loadings" (within-community connectivity-weighted mean). Association (return macro network): "average" (mean signed edge weight between communities), "rv" (Escoufier RV coefficient), "canonical" (first canonical correlation).
estimator	Estimator used for the node-level network that "loadings" and "average" need (see psychnet()). Default "glasso".
scale	Standardise each item before forming score composites. Default TRUE.
labels	Optional item labels (used when data has no column names).
...	Passed to the estimator.

Value

For a score method, a data.frame with one column per community (one row per observation). For an association method, a psychnet macro network among communities.

Examples

```
net_aggregate(SRL_Claude, communities = c(1, 1, 2, 2, 2))      # reduced data
net_aggregate(SRL_Claude, communities = c(1, 1, 2, 2, 2), method = "rv") # macro net
```

net_boot

Bootstrap a psychometric network

Description

Resamples observations with replacement, re-estimates the network on each resample, and summarizes the sampling distribution of every edge weight and node centrality (mean, percentile confidence interval, and edge inclusion proportion). An edge is flagged significant when its percentile interval excludes zero. The raw per-resample draws are stored on the returned object for use by [difference_test\(\)](#).

Usage

```
net_boot(
  data,
  method = "glasso",
  n_boot = 1000L,
  ci = 0.95,
  measures = c("strength", "expected_influence"),
  centrality_fn = NULL,
  predictability = FALSE,
  threshold = FALSE,
  diff_test = FALSE,
  p_adjust = "none",
  labels = NULL,
  cores = NULL,
  engine = NULL,
  ...
)
```

Arguments

data	Numeric data frame or matrix (rows = observations).
method	Estimator (see psychnet()). Default "glasso".
n_boot	Number of bootstrap resamples. Default 1000.
ci	Confidence level for percentile intervals. Default 0.95.
measures	Centrality measures to bootstrap. Defaults to the two recommended for psychometric networks ("strength", "expected_influence"); "betweenness"/"closeness" and custom measures (via centrality_fn) are also accepted. See net_centralities() .
centrality_fn	Optional function supplying any non-built-in measures (see net_centralities()).

predictability	Logical; if TRUE and the estimator returns a precision matrix (GGM family), bootstrap node predictability (R^2) and report its interval. Default FALSE.
threshold	Logical; if TRUE, also return the observed network with every edge whose bootstrap interval includes zero set to zero (<code>\$thresholded</code>). Default FALSE.
diff_test	Logical; if TRUE, also return two-sided bootstrap difference p-value matrices for edges (<code>\$edge_diff_p</code> , NULL past 500 edges) and for each centrality measure (<code>\$centrality_diff_p</code>). Default FALSE.
p_adjust	Multiple-comparison adjustment applied to the difference p-value matrices (any <code>stats::p.adjust</code> method). Default "none".
labels	Optional node labels.
cores	Number of CPU cores for the resample loop. NULL (default) uses two thirds of the detected cores; 1 forces a serial run. Parallelism uses forking (<code>parallel::mclapply</code>) and falls back to serial on Windows. Because every resample index is drawn in the parent process before any fitting, the result is identical for any number of cores and reproducible from <code>set.seed()</code> .
engine	Optional estimator engine forwarded to each resample fit (e.g. "base"/"glasso" for glasso, "base"/"glmnet" for ising/mgm). NULL (default) uses the estimator's own default.
...	Passed to the estimator.

Value

An object of class `psychnet_bootstrap`: tidy `$edges` (with a significant flag) and `$centrality` data frames, the observed network in `$observed`, raw resample draws in `$edge_boot`, `$str_boot`, `$ei_boot`, and the general `$centrality_boot` (named list, one matrix per measure). Optional `$predictability`, `$thresholded`, `$edge_diff_p`, `$centrality_diff_p`, plus `$lambda_path`/`$lambda_selected` when the estimator reports them.

Examples

```
set.seed(1)
x <- matrix(stats::rnorm(150 * 5), 150, 5) %%% chol(0.4^abs(outer(1:5, 1:5, "-")))
colnames(x) <- paste0("V", 1:5)
bs <- net_boot(x, n_boot = 50, cores = 1)
as.data.frame(bs)
```

net_bridge

Bridge centrality

Description

Computes bridge centrality for an undirected weighted network: how strongly each node connects to communities other than its own. You supply the community membership; psychnets does not detect it.

Usage

```
net_bridge(x, communities, normalize = FALSE, labels = NULL)
```

Arguments

x	A psychnet object or a square weighted adjacency matrix.
communities	Community membership, one entry per node: a vector aligned to the node order, or a named vector / list keyed by node label.
normalize	If TRUE, divide each metric by the number of available other-community nodes (comparable across differently sized networks). Default FALSE.
labels	Optional node labels (used when x is a bare matrix).

Value

A tidy data.frame (class psychnet_bridge), one row per node, with columns node, community, bridge_strength, bridge_betweenness, bridge_closeness, bridge_ei1, bridge_ei2. Visualise with [plot.psychnet_bridge\(\)](#).

References

Jones, P. J., Ma, R., & McNally, R. J. (2021). Bridge centrality. *Multivariate Behavioral Research*, 56(2), 353-367.

Examples

```
S <- 0.3^abs(outer(1:6, 1:6, "-"))
fit <- ebic_glasso(cor_matrix = S, n = 400)
net_bridge(fit, communities = c(1, 1, 1, 2, 2, 2))
```

net_centralities	<i>Node centrality</i>
------------------	------------------------

Description

Node centrality

Usage

```
net_centralities(
  x,
  measures = c("strength", "expected_influence"),
  centrality_fn = NULL,
  ...
)
```

Arguments

<code>x</code>	A psychnet object or a weighted adjacency matrix.
<code>measures</code>	Character vector of measures to return. Any of "strength", "expected_influence" (the defaults, recommended for psychometric networks), "betweenness", "closeness", plus any names supplied via <code>centrality_fn</code> . Betweenness and closeness are computed on the absolute, inverted-weight graph and are not generally meaningful on signed networks – request them only when a downstream comparison needs them.
<code>centrality_fn</code>	Optional function taking the weighted adjacency matrix and returning a named list of node-centrality vectors, used to supply any measures not built in.
<code>...</code>	Unused.

Value

A tidy data.frame, one row per node, with a node column and one column per requested measure (strength = sum of absolute edge weights, expected_influence = sum of signed edge weights, by default).

Examples

```
S <- 0.4^abs(outer(1:6, 1:6, "-"))
net_centralities(ebic_glasso(cor_matrix = S, n = 250))
```

net_clustering	<i>Weighted clustering coefficients</i>
----------------	---

Description

Per-node clustering coefficients for a weighted (optionally signed) network: Watts-Strogatz, Zhang-Horvath, Onnela, and Barrat. For signed networks the signed variants are also returned (a closed triangle of three negative or one negative edge lowers the signed coefficient). Ported from `qgraph::clustcoef_auto`.

Usage

```
net_clustering(x, labels = NULL)
```

Arguments

<code>x</code>	A psychnet object or a square weighted adjacency matrix.
<code>labels</code>	Optional node labels (used when <code>x</code> is a bare matrix).

Value

A tidy data.frame (class `psychnet_clustering`), one row per node: node, clustWS, signed_clustWS, clustZhang, signed_clustZhang, clustOnnela, signed_clustOnnela, clustBarrat.

References

Costantini, G., & Perugini, M. (2014); Watts & Strogatz (1998); Zhang & Horvath (2005); Onnela et al. (2005); Barrat et al. (2004).

Examples

```
S <- 0.4^abs(outer(1:6, 1:6, "-"))
net_clustering(ebic_glasso(cor_matrix = S, n = 400))
```

net_compare	<i>Network Comparison Test</i>
-------------	--------------------------------

Description

Permutation test for whether two groups' Gaussian graphical models differ, on three invariants: global strength (M), maximum edge difference (S), and per-edge differences (E). Networks are EBIC graphical lassos (clean-room pure R). Equivalent in purpose to `NetworkComparisonTest::NCT()`.

Usage

```
net_compare(
  data1,
  data2 = NULL,
  iter = 1000L,
  gamma = 0.5,
  paired = FALSE,
  abs = TRUE,
  weighted = TRUE,
  p_adjust = "none"
)
```

Arguments

data1, data2	Numeric data frames/matrices with the same columns.
iter	Number of permutations. Default 1000.
gamma	EBIC hyperparameter. Default 0.5.
paired	Logical; within-row swapping for paired designs. Default FALSE.
abs	Logical; compare absolute edge weights. Default TRUE.
weighted	Logical; if FALSE, binarize networks first. Default TRUE.
p_adjust	Multiple-comparison adjustment for per-edge p-values (any stats::p.adjust method). Default "none".

Value

An object of class `psychnet_nct` with `$nw1`, `$nw2`, and `$M`, `$S`, `$E` (each observed, perm, p_value); `$E` also carries `edge_names`, a from/to data frame aligned to the per-edge vector.

Examples

```
set.seed(1)
a <- matrix(stats::rnorm(150 * 5), 150, 5)
b <- matrix(stats::rnorm(150 * 5), 150, 5)
colnames(a) <- colnames(b) <- paste0("V", 1:5)
fit <- net_compare(a, b, iter = 25)
fit
```

net_crosswalk	<i>Argument crosswalk: psychnet as a substitute for qgraph / IsingFit / mgm</i>
---------------	---

Description

A tidy, one-row-per-argument map from each reference package's estimator to its psychnet equivalent, so users migrating from qgraph::EBICglasso, qgraph::cor_auto, qgraph::ggmModSelect, IsingFit::IsingFit, or mgm::mgm can find the matching argument and see what psychnet changes by default. Cross-sectional estimators only (temporal models are out of scope).

Usage

```
net_crosswalk(
  reference = c("all", "EBICglasso", "cor_auto", "ggmModSelect", "IsingFit", "mgm")
)
```

Arguments

reference	Which reference function to show: "all" (default), "EBICglasso", "cor_auto", "ggmModSelect", "IsingFit", or "mgm".
-----------	--

Value

A tidy data.frame, one row per argument, with columns reference (the pkg::fn being substituted), psychnet (the psychnet verb), ref_arg, psychnet_arg ("-" when there is no counterpart), status (identical / renamed / default differs / semantics differ / reference only / psychnet only), and a short note.

Examples

```
net_crosswalk("EBICglasso")
net_crosswalk("IsingFit")
```

net_edge_betweenness	<i>Edge betweenness centrality</i>
----------------------	------------------------------------

Description

For each edge, the share of weighted shortest paths (across all node pairs) that pass through it - a high value marks an edge that bridges otherwise distant parts of the network. Geodesics are computed on inverse absolute weights, so strong edges count as short, matching `net_centralities()`'s node betweenness/closeness.

Usage

```
net_edge_betweenness(x, invert = TRUE, labels = NULL)
```

Arguments

<code>x</code>	A psychnet object or a square weighted adjacency matrix.
<code>invert</code>	If TRUE (default) edge weights are inverted to distances (strong association = short path). Set FALSE to treat weights as distances.
<code>labels</code>	Optional node labels (used when <code>x</code> is a bare matrix).

Value

A tidy data.frame, one row per edge: from, to, edge_betweenness. Undirected networks give one row per unordered edge.

Examples

```
S <- 0.4^abs(outer(1:6, 1:6, "-"))
net_edge_betweenness(ebic_glasso(cor_matrix = S, n = 400))
```

net_predict	<i>Node predictability</i>
-------------	----------------------------

Description

Reports how well each node is predicted by the others in a fitted network. For Gaussian graphical models this is the closed-form variance explained (R-squared) from the precision matrix and needs no data. For the nodewise models (`ising_fit()`, `ising_sampler()`, `mgm_fit()`) it requires the data and reports R-squared for Gaussian nodes and classification accuracy (CC) plus normalized accuracy (nCC) for binary nodes.

Usage

```
net_predict(x, data = NULL, ...)
```

Arguments

x	A psychnet object.
data	The data the network was estimated from; required for the nodewise models (ising / IsingSampler / mgm), ignored for the GGMs.
...	Unused.

Value

A tidy data.frame, one row per node, with columns node, type ("gaussian" or "binary"), metric ("R2" or "nCC"), predictability, and accuracy (classification accuracy for binary nodes, NA for Gaussian).

Examples

```
S <- 0.4^abs(outer(1:6, 1:6, "-"))
net_predict(ebic_glasso(cor_matrix = S, n = 250))
```

net_smallworld	<i>Small-world index</i>
----------------	--------------------------

Description

The Humphries & Gurney (2008) small-world index σ : observed transitivity and average shortest-path length compared with degree-preserving random graphs ($\sigma = (C/C_{\text{rand}}) / (L/L_{\text{rand}})$; $\sigma > 1$ indicates small-worldness). Computed on the binarised network. Ported from `qgraph::smallworldness`.

Usage

```
net_smallworld(x, n_rand = 100L, seed = NULL)
```

Arguments

x	A psychnet object or a square weighted adjacency matrix.
n_rand	Number of degree-preserving random graphs. Default 100.
seed	Optional integer for reproducibility of the random graphs.

Value

A tidy one-row data.frame (class psychnet_smallworld): smallworldness, transitivity, aspl, transitivity_rand, aspl_rand.

References

Humphries, M. D., & Gurney, K. (2008). *PLoS ONE*, 3(4), e0002051.

Examples

```
S <- 0.4^abs(outer(1:8, 1:8, "-"))
net_smallworld(ebic_glasso(cor_matrix = S, n = 400), n_rand = 50, seed = 1)
```

net_stability

*Centrality-stability coefficient (case-dropping subset bootstrap)***Description**

Centrality-stability coefficient (case-dropping subset bootstrap)

Usage

```
net_stability(
  data,
  method = "glasso",
  measures = c("strength", "expected_influence"),
  centrality_fn = NULL,
  drop_prop = seq(0.1, 0.9, by = 0.1),
  iter = 100L,
  threshold = 0.7,
  certainty = 0.95,
  labels = NULL,
  ...
)
```

Arguments

data	Numeric data frame or matrix (rows = observations).
method	Estimator (see psychnet()). Default "glasso".
measures	Centrality measures to assess. Defaults to the two recommended for psychometric networks (c("strength", "expected_influence")); "betweenness"/"closeness" and custom measures (via centrality_fn) are also accepted. See net_centralities() .
centrality_fn	Optional function supplying any non-built-in measures (see net_centralities()).
drop_prop	Proportions of cases to drop. Default seq(0.1, 0.9, 0.1).
iter	Subsets per proportion. Default 100.
threshold	Minimum acceptable rank correlation. Default 0.7.
certainty	Probability the correlation must exceed threshold. Default 0.95.
labels	Optional node labels.
...	Passed to the estimator.

Value

An object of class `psychnet_stability` with `$cs` (CS-coefficient per measure) and a tidy `$table` (columns `measure`, `drop_prop`, `mean_cor`, `sd_cor`, `prop_above`) of the case-dropping correlations by drop proportion. Visualise it with [plot.psychnet_stability\(\)](#).

Examples

```
set.seed(1)
x <- matrix(stats::rnorm(200 * 5), 200, 5) %%% chol(0.4^abs(outer(1:5, 1:5, "-")))
colnames(x) <- paste0("V", 1:5)
cs <- net_stability(x, drop_prop = c(0.3, 0.6), iter = 10)
cs$cs
```

node_predictability	<i>Node predictability as a plotting vector</i>
---------------------	---

Description

A thin companion to `net_predict()` that returns predictability as a plain numeric vector in node order, clamped to $[0, 1]$ – the form `cograph::splot()` expects for `pie_values` (the predictability ring drawn around each node). Use `net_predict()` when you want the full tidy table.

Usage

```
node_predictability(x, data = NULL)
```

Arguments

<code>x</code>	A psychnet object.
<code>data</code>	The data the network was estimated from; required for the nodewise models (ising / ising_sampler / mgm), ignored for the GGMs.

Value

A named numeric vector, one value per node (node order), each in $[0, 1]$.

Examples

```
S <- 0.4^abs(outer(1:6, 1:6, "-"))
node_predictability(ebic_glasso(cor_matrix = S, n = 250))
```

pcor_network	<i>Partial correlation network</i>
--------------	------------------------------------

Description

Conditional (full-order) association network: each edge is the correlation between two variables with all others partialled out, obtained from the inverse correlation matrix. Equivalent to `bootnet`'s "pcor" default.

Usage

```
pcor_network(
  data = NULL,
  cor_matrix = NULL,
  n = NULL,
  cor_method = c("pearson", "spearman", "kendall", "auto"),
  threshold = 0,
  alpha = NULL,
  adjust = "none",
  na_method = c("pairwise", "listwise"),
  labels = NULL
)
```

Arguments

data	Numeric data frame or matrix (rows = observations). Optional if <code>cor_matrix</code> is supplied.
cor_matrix	Optional precomputed correlation matrix; if given, data is ignored and <code>n</code> is required when <code>alpha</code> is used.
n	Sample size (needed for significance testing when <code>cor_matrix</code> is supplied).
cor_method	Correlation method: "pearson" (default), "spearman", "kendall", or "auto" (polychoric/polyserial for ordinal items, the <code>qgraph::cor_auto</code> default; see cor_auto()).
threshold	Correlations with absolute value below this are set to zero. Default 0.
alpha	Significance level; if set, correlations not significant at <code>alpha</code> are zeroed. NULL (default) keeps every edge.
adjust	Multiple-comparison adjustment for the edge p-values (any stats::p.adjust method). Default "none".
na_method	Missing-data handling: "pairwise" (default) uses pairwise-complete correlations projected to the nearest positive-definite matrix; "listwise" drops rows with any NA. Identical when data is complete.
labels	Optional node labels.

Value

A `psychnet` object whose `$weights` is the thresholded partial-correlation matrix, with `$precision`, `$cor_matrix` (and `$p_values` when `alpha` is used).

Examples

```
x <- matrix(stats::rnorm(200 * 4), 200, 4)
pcor_network(x)
pcor_network(x, alpha = 0.05, adjust = "holm")
```

plot.psychnet	<i>Plot a psychnet network</i>
---------------	--------------------------------

Description

Renders the estimated network with `cograph::splot()` (a Suggested package); psychnet objects inherit from `cograph_network` for exactly this purpose. For the bootstrap / centrality / difference / stability diagnostics use the dedicated `plot()` methods for those result objects instead.

Usage

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

Arguments

x	A psychnet object.
...	Passed to <code>cograph::splot()</code> .

Value

The value of `cograph::splot()`, invisibly.

Examples

```
S <- 0.4^abs(outer(1:6, 1:6, "-"))
fit <- ebic_glasso(cor_matrix = S, n = 300)
if (requireNamespace("cograph", quietly = TRUE)) {
  plot(fit)
}
```

plot.psychnet_bootstrap	<i>Plot a network bootstrap</i>
-------------------------	---------------------------------

Description

Visualises a `net_boot()` result. `type = "edges"` (default) draws the bootstrapped edge-weight confidence intervals sorted by the observed weight (bootnet's edge-accuracy plot); `type = "centrality"` draws the bootstrapped centrality intervals, one sorted panel per measure; `type = "edge_diff"` and `type = "centrality_diff"` draw the bootstrapped difference "significance box" matrix for edges or for one centrality; `type = "predictability"` draws the node predictability intervals (only when `net_boot()` was run with `predictability = TRUE`).

Usage

```
## S3 method for class 'psychnet_bootstrap'
plot(
  x,
  type = c("edges", "centrality", "edge_diff", "centrality_diff", "predictability"),
  measure = NULL,
  ...
)
```

Arguments

x	A psychnet_bootstrap object from net_boot() .
type	One of "edges", "centrality", "edge_diff", "centrality_diff", "predictability".
measure	For "centrality"/"centrality_diff", which measure(s) to draw. Default: all bootstrapped measures ("centrality") or the first ("centrality_diff").
...	Unused.

Value

x, invisibly. Called for the plot it draws.

Examples

```
set.seed(1)
x <- matrix(stats::rnorm(150 * 5), 150, 5) %%% chol(0.4^abs(outer(1:5, 1:5, "-")))
colnames(x) <- paste0("V", 1:5)
bs <- net_boot(x, n_boot = 50, cores = 1) # n_boot >= 1000 for real use
plot(bs) # edge-weight CIs
plot(bs, type = "centrality") # centrality CIs
```

plot.psychnet_bridge *Plot bridge centrality*

Description

One sorted horizontal panel per bridge measure (nodes coloured by community).

Usage

```
## S3 method for class 'psychnet_bridge'
plot(x, measures = NULL, ...)
```

Arguments

x	A psychnet_bridge data frame from net_bridge() .
measures	Which bridge columns to draw. Default: all five.
...	Unused.

Value

`x`, invisibly. Called for the plot it draws.

Examples

```
S <- 0.3^abs(outer(1:6, 1:6, "-"))
fit <- ebic_glasso(cor_matrix = S, n = 400)
plot(net_bridge(fit, communities = c(1, 1, 1, 2, 2, 2)))
```

plot.psychnet_casedrop

Plot edge-weight case-dropping stability

Description

Draws the four similarity metrics (correlation, mean/median/max absolute deviation) against the proportion of cases dropped, each with a ± 1 SD band; the correlation panel carries the acceptance threshold and CS-coefficient.

Usage

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

Arguments

<code>x</code>	A psychnet_casedrop object.
<code>...</code>	Unused.

Value

`x`, invisibly. Called for the plot it draws.

Examples

```
# Small `iter` / `drop_prop` for a fast example; see [casedrop_reliability()]
# for the defaults a real assessment should use.
plot(casedrop_reliability(SRL_Claude, iter = 5, drop_prop = c(0.25, 0.5)))
```

```
plot.psychnet_centrality
```

Plot node centralities

Description

Draws the centrality table returned by `net_centralities()`. `type = "bar"` gives one sorted horizontal lollipop panel per measure; `type = "line"` gives the qgraph/bootnet centrality plot — one faceted panel per measure, nodes on a shared vertical axis, a line-and-marker series within each panel.

Usage

```
## S3 method for class 'psychnet_centrality'
plot(
  x,
  type = c("bar", "line"),
  scale = c("raw", "z", "relative"),
  measures = NULL,
  ...
)
```

Arguments

<code>x</code>	A psychnet_centrality data frame from <code>net_centralities()</code> .
<code>type</code>	"bar" (default) for one sorted lollipop panel per measure, or "line" for the faceted qgraph-style centrality plot.
<code>scale</code>	For <code>type = "line"</code> , the per-measure transform: "raw" (default — each faceted panel keeps its own axis, so no rescaling is needed), "z" (z-score per measure, centred at 0), or "relative" (each measure min-max scaled to [0, 1]). <code>type = "bar"</code> always shows raw values.
<code>measures</code>	Which measure columns to draw. Default: all of them.
<code>...</code>	Unused.

Value

`x`, invisibly. Called for the plot it draws.

Examples

```
S <- 0.4^abs(outer(1:6, 1:6, "-"))
fit <- ebic_glasso(cor_matrix = S, n = 300)
plot(net_centralities(fit))
plot(net_centralities(fit), type = "line")
```

plot.psychnet_difference

Plot a bootstrapped difference test

Description

Draws the bootnet-style "significance box" matrix for the pairwise difference test returned by `difference_test()`: items on both axes ordered by their observed value, the diagonal showing the observed value, and each off-diagonal cell filled when that pair differs significantly (red when the row item is larger, blue when smaller). `style = "forest"` instead draws a forest plot: one row per pair, the bootstrapped difference as a point with its confidence interval, a reference line at zero, and significant pairs (interval excluding zero) emphasised.

Usage

```
## S3 method for class 'psychnet_difference'
plot(x, style = c("box", "forest"), ...)
```

Arguments

<code>x</code>	A psychnet_difference data frame from <code>difference_test()</code> .
<code>style</code>	"box" (default) for the significance-box matrix, or "forest" for a forest plot of the pairwise differences with their CIs.
<code>...</code>	Unused.

Value

`x`, invisibly. Called for the plot it draws.

Examples

```
set.seed(1)
x <- matrix(stats::rnorm(150 * 5), 150, 5) %*% chol(0.4^abs(outer(1:5, 1:5, "-")))
colnames(x) <- paste0("V", 1:5)
bs <- net_boot(x, n_boot = 50, cores = 1) # n_boot >= 1000 for real use
plot(difference_test(bs, type = "strength")) # box matrix
plot(difference_test(bs, type = "strength"), style = "forest") # forest plot
```

plot.psychnet_nct	<i>Plot a Network Comparison Test</i>
-------------------	---------------------------------------

Description

Visualises a `net_compare()` result. `type = "strength"` (default) and `type = "structure"` draw the permutation null distribution for the global strength invariance (M) and the maximum edge-difference (S) statistics, with the observed value and p-value marked; `type = "edges"` draws the observed per-edge absolute differences, coloured by whether each edge differs significantly.

Usage

```
## S3 method for class 'psychnet_nct'
plot(x, type = c("strength", "structure", "edges"), alpha = 0.05, ...)
```

Arguments

x	A psychnet_nct object from <code>net_compare()</code> .
type	One of "strength", "structure", "edges".
alpha	Significance level for colouring per-edge differences. Default 0.05.
...	Unused.

Value

x, invisibly. Called for the plot it draws.

Examples

```
set.seed(1)
mk <- function(s) { set.seed(s)
  matrix(stats::rnorm(120 * 4), 120, 4) %*% chol(0.3^abs(outer(1:4, 1:4, "-"))) }
cmp <- net_compare(mk(1), mk(2), iter = 25)
plot(cmp)                # global strength permutation null
plot(cmp, type = "edges") # per-edge differences
```

plot.psychnet_reliability	<i>Plot split-half reliability</i>
---------------------------	------------------------------------

Description

Histograms of the four between-halves edge metrics across split-half iterations, with each observed mean marked.

Usage

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

Arguments

```
x          A psychnet_reliability object.
...        Unused.
```

Value

x, invisibly. Called for the plot it draws.

Examples

```
# Small `iter` for a fast example; see [network_reliability()] for the
# default a real assessment should use.
plot(network_reliability(SRL_Claude, iter = 10))
```

```
plot.psychnet_stability
```

```
Plot centrality stability (case-dropping)
```

Description

Draws the case-dropping stability curves from `net_stability()`: mean rank correlation with the full-sample centrality against the proportion of cases dropped, one line per measure, with a +/- 1 SD band, the acceptance threshold, and the CS-coefficient annotated in the legend.

Usage

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

Arguments

```
x          A psychnet_stability object from net_stability().
...        Unused.
```

Value

x, invisibly. Called for the plot it draws.

Examples

```
set.seed(1)
d <- matrix(stats::rnorm(200 * 5), 200, 5) %*% chol(0.4^abs(outer(1:5, 1:5, "-")))
s <- net_stability(d, drop_prop = c(0.3, 0.6), iter = 10)
plot(s)
```

print.psychnet	<i>Print a psychnet network</i>
----------------	---------------------------------

Description

Print a psychnet network

Usage

```
## S3 method for class 'psychnet'  
print(x, ...)
```

Arguments

x	A psychnet object.
...	Unused.

Value

x, invisibly.

Examples

```
fit <- pcor_network(SRL_GPT)  
print(fit)
```

print.psychnet_bootstrap	<i>Print a network bootstrap</i>
--------------------------	----------------------------------

Description

Print a network bootstrap

Usage

```
## S3 method for class 'psychnet_bootstrap'  
print(x, ...)
```

Arguments

x	A psychnet_bootstrap object.
...	Unused.

Value

x, invisibly.

```
print.psychnet_casedrop
```

Print an edge-weight stability result

Description

Print an edge-weight stability result

Usage

```
## S3 method for class 'psychnet_casedrop'
print(x, ...)
```

Arguments

x	A psychnet_casedrop object.
...	Unused.

Value

x, invisibly.

```
print.psychnet_group
```

Per-group psychometric networks

Description

The object returned by `psychnet()` when group is supplied: a named list of `psychnet` networks, one per level of the grouping variable. It plots as a grid with `cograph::splot()` and is consumed per level by the framework verbs (`net_centralities()`, `net_predict()`, `net_boot()`, `net_stability()`, `net_compare()`), each returning a `*_group` result.

Usage

```
## S3 method for class 'psychnet_group'
print(x, ...)

## S3 method for class 'psychnet_group'
as.data.frame(x, row.names = NULL, optional = FALSE, ...)

## S3 method for class 'psychnet_group'
summary(object, ...)
```

Arguments

x	A psychnet_group object.
...	Ignored.
row.names, optional	Ignored (S3 consistency).
object	A psychnet_group object.

Value

x, invisibly (for print).

For as.data.frame, the per-group edge lists stacked with a group column.

For summary, a data frame with one row per group (node/edge counts and mean absolute edge weight).

```
print.psychnet_moderated
```

Print a moderated MGM fit

Description

Print a moderated MGM fit

Usage

```
## S3 method for class 'psychnet_moderated'
print(x, ...)
```

Arguments

x	A psychnet_moderated object.
...	Unused.

Value

x, invisibly.

```
print.psychnet_multilevel
```

Within / between event-data networks

Description

The object returned by `psychnet()` for nested event data with `standardize = FALSE`: a pair of Gaussian graphical networks decomposing the actor-by-action frequency covariance into a within-actor and a between-actor part.

Usage

```
## S3 method for class 'psychnet_multilevel'
print(x, ...)

## S3 method for class 'psychnet_multilevel'
as.data.frame(x, row.names = NULL, optional = FALSE, ...)
```

Arguments

`x` A `psychnet_multilevel` object.
`...` Ignored.
`row.names, optional` Ignored (S3 consistency).

Value

`x`, invisibly (for `print`).
 For `as.data.frame`, the two edge lists stacked with a `level` column.

```
print.psychnet_nct      Print a Network Comparison Test
```

Description

Print a Network Comparison Test

Usage

```
## S3 method for class 'psychnet_nct'
print(x, ...)
```

Arguments

`x` A `psychnet_nct` object.
`...` Unused.

Value

x, invisibly.

```
print.psychnet_redundancy
```

Print a redundancy result

Description

Print a redundancy result

Usage

```
## S3 method for class 'psychnet_redundancy'  
print(x, ...)
```

Arguments

x	A psychnet_redundancy object.
...	Unused.

Value

x, invisibly.

```
print.psychnet_reliability
```

Print a split-half reliability result

Description

Print a split-half reliability result

Usage

```
## S3 method for class 'psychnet_reliability'  
print(x, ...)
```

Arguments

x	A psychnet_reliability object.
...	Unused.

Value

x, invisibly.

```
print.psychnet_result_group
```

Per-group framework results

Description

The list returned by a framework verb (`net_centralities()`, `net_predict()`, `net_boot()`, `net_stability()`) applied to a `psychnet_group`: one result per group level, keyed by level. `as.data.frame()` stacks the per-group tables with a leading group column.

Usage

```
## S3 method for class 'psychnet_result_group'
print(x, ...)

## S3 method for class 'psychnet_result_group'
as.data.frame(x, row.names = NULL, optional = FALSE, ...)
```

Arguments

<code>x</code>	A <code>psychnet_result_group</code> object.
<code>...</code>	Ignored.
<code>row.names, optional</code>	Ignored (S3 consistency).

Value

`x`, invisibly (for `print`).

For `as.data.frame`, the per-group tables stacked with a group column.

```
print.psychnet_stability
```

Print a centrality-stability result

Description

Print a centrality-stability result

Usage

```
## S3 method for class 'psychnet_stability'
print(x, ...)
```

Arguments

`x` A psychnet_stability object.
`...` Unused.

Value

`x`, invisibly.

psychnet	<i>Estimate a psychometric network</i>
----------	--

Description

The package's main entry point. Supply data and a method; everything else is optional fine-grained control. `psychnet()` reads two kinds of input and picks the right one automatically (`source = "auto"`):

Usage

```
psychnet(
  data,
  method = c("glasso", "cor", "pcor", "ising", "mgm", "huge", "ggm", "tmfg", "logo",
    "relimp", "ising_sampler"),
  threshold = 0,
  gamma = NULL,
  labels = NULL,
  vars = NULL,
  group = NULL,
  source = c("auto", "data", "eventdata"),
  actor = NULL,
  action = "Action",
  session = NULL,
  time = NULL,
  compute_sessions = TRUE,
  time_threshold = 900,
  id = NULL,
  standardize = TRUE,
  ...
)
```

Arguments

`data` A numeric data frame / matrix (rows = observations), or a long event log when actor is supplied.

method	Estimator. One of "glasso" (default), "ggm", "tmfg", "logo", "relimp", "ising", "ising_sampler", "huge", "mgm", "cor", "pcor". The qgraph/bootnet names are accepted as aliases (e.g. "EBICglasso" -> "glasso", "ggmModSelect" -> "ggm"). Event data is restricted to the Gaussian graphical methods.
threshold	Absolute-weight threshold below which edges are zeroed (forwarded only to the methods that take it: cor, pcor, glasso, huge, ggm, logo).
gamma	EBIC hyperparameter. NULL (default) keeps each method's own default (0.5 for the regularized Gaussian graphical models, 0 for ggm, 0.25 for ising/mgm); set it to override. Forwarded only to the regularized methods.
labels	Optional node labels.
vars	Which variables to build the network on. Defaults to every variable. Selected the tidy way: a name range (motivation:regulation), a column-index range (3:9), a vector of names (c(joy, fear) or c("joy", "fear")), or a single name – the same grammar as subset()'s select=. (For an event log, actions become the variables, so vars selects feature columns only when data is already a numeric table.)
group	Optional grouping column(s). When supplied, one network is estimated per level of group and a psychnet_group (a named list of networks) is returned, which plots as a grid and is iterated per level by the framework verbs.
source	Input kind: "auto" (default; an event log when actor is given, otherwise a numeric table), "data", or "eventdata".
actor, action, session, time	Event-log columns. actor (and, by default, action = "Action") name the subject and the event; session is an explicit within-actor grouping, and time is used only to compute sessions from gaps. Supplying actor selects source = "eventdata".
compute_sessions, time_threshold	Split each actor into sessions from time gaps (a new session starts when the gap exceeds time_threshold, default 900 s). compute_sessions defaults to TRUE; it is a no-op when no time is supplied.
id	Actor column when an event log has already been reduced to a numeric feature table (one row per occasion) rather than raw events.
standardize	For nested event data (several occasions per actor), TRUE (default) removes the actor clustering by person-centering and fits a single network; FALSE returns the within/between pair instead.
...	Passed to the underlying estimator (e.g. cor_method= for the correlation-based methods, npn= for "huge", rule= for the Ising methods, alpha= for the correlation / "ising_sampler" methods).

Details

- **a numeric table** (source = "data") – the ordinary cross-sectional case: one row per observation, one column per variable;
- **a long event log** (source = "eventdata") – one row per event, with an actor and an action column (and optionally session / time). The log is converted to action frequencies with

`event_frequencies()` and, when actors contribute several occasions, decomposed into within- and between-actor networks. Passing actor switches this on automatically.

method speaks the package's own short vocabulary – "glasso", "ggm", "tmfg", "logo", "relimp", "ising", "ising_sampler", "huge", "mgm", "cor", "pcor". For interoperability it **also** accepts the qgraph/bootnet spellings ("EBICglasso", "ggmModSelect", "TMFG", "LoGo", "IsingFit", "IsingSampler"), which resolve to the same estimators. Whichever you pass in, the stored \$method is the short name.

Value

A psychnet object; a psychnet_group (named list of networks) when group is supplied; or, for nested event data with standardize = FALSE, a psychnet_multilevel object carrying \$within and \$between networks.

Examples

```
x <- matrix(stats::rnorm(200 * 5), 200, 5)
colnames(x) <- c("joy", "fear", "calm", "anger", "trust")
psychnet(x, method = "glasso")
psychnet(x, method = "pcor", vars = joy:anger)      # name range
psychnet(x, method = "glasso", vars = 1:3)          # index range
psychnet(x, method = "EBICglasso")                  # qgraph alias, same result

ev <- data.frame(
  Actor = rep(paste0("s", 1:30), each = 20),
  Action = sample(c("read", "quiz", "note", "watch"), 600, replace = TRUE))
psychnet(ev, actor = "Actor", action = "Action")    # event data, auto-detected

d <- data.frame(g = rep(c("A", "B"), each = 100),
  matrix(stats::rnorm(200 * 4), 200, 4,
    dimnames = list(NULL, paste0("V", 1:4))))
psychnet(d, method = "glasso", group = "g")         # one network per level
```

redundancy

Detect redundant node pairs ("goldbricker")

Description

Flags pairs of items that behave redundantly in a network: their correlations with all other items are mostly statistically indistinguishable (a small proportion of significantly different correlations) and the two items are themselves strongly correlated. Ported from `networktools::goldbricker`.

Usage

```
redundancy(
  data,
  p = 0.05,
  threshold = 0.25,
```

```
cor_min = 0.5,
cor_method = c("auto", "pearson", "spearman", "kendall")
)
```

Arguments

data	A numeric data frame or matrix (rows = observations).
p	Significance level for each pairwise correlation-difference test. Default 0.05.
threshold	Maximum proportion of significantly different correlations for a pair to be flagged redundant. Default 0.25.
cor_min	Minimum correlation between the two items themselves. Default 0.5.
cor_method	Correlation type: "auto" (default, cor_auto() - polychoric/polyserial as appropriate, matching goldbricker), "pearson", "spearman", or "kendall".

Value

A tidy data.frame (class psychnet_redundancy), one row per flagged pair (sorted most-redundant first), with columns item1, item2, proportion (share of significantly different correlations) and correlation. Zero rows when nothing is flagged. The full proportion matrix is in attr(x, "proportion_matrix").

References

Hallquist, M. N., Wright, A. G. C., & Molenaar, P. C. M. (2021). Problems with centrality measures in psychopathology networks. *Multivariate Behavioral Research*, 56(2), 199-223.

Examples

```
redundancy(SRL_Claude)
```

relimp_network	<i>Relative-importance network (LMG / Shapley)</i>
----------------	--

Description

Builds a directed network in which the edge predictor -> outcome is the predictor’s LMG (Shapley) share of the outcome node’s regression R-squared. Equivalent in purpose to relaimpo::calc.relimp(type = "lmg") applied nodewise / bootnet’s "relimp" default, pure base R and self-certified via [lmg_certificate\(\)](#).

Usage

```
relimp_network(
  data = NULL,
  cor_matrix = NULL,
  cor_method = c("pearson", "spearman", "kendall", "auto"),
  max_nodes = 21L,
  normalized = FALSE,
```

```
na_method = c("pairwise", "listwise"),
labels = NULL
)
```

Arguments

data	Numeric data frame or matrix (rows = observations). Optional if <code>cor_matrix</code> is supplied.
cor_matrix	Optional correlation matrix.
cor_method	Correlation when data is supplied: "pearson" (default), "spearman", "kendall", or "auto" (polychoric/polyserial; see cor_auto()).
max_nodes	Refuse to run above this many nodes (the cost grows as $2^{(p-1)}$ per node). Default 21.
normalized	If TRUE, rescale each outcome's incoming importance shares to sum to 1, matching bootnet/relaimpo's normalized reporting. The default FALSE keeps raw LMG shares that sum to the outcome R-squared.
na_method	Missing-data handling when data is supplied: "pairwise" (default) or "listwise". See ebic_glasso() .
labels	Optional node labels.

Value

A psychnet object whose `$weights` is the directed importance matrix (`weights[k, j]` = importance of `k` for outcome `j`), with `$r2` (per-node full-model R-squared), `$cor_matrix`, and `$kkt` (the decomposition residual). With `normalized = FALSE` (default) each outcome's incoming shares sum to its R-squared (`colSums($weights) == $r2`); with `normalized = TRUE` they are rescaled to sum to 1, `$raw_importance` holds the unscaled shares, and `$kkt` (like [lmg_certificate\(\)](#)) is computed from those raw shares.

Examples

```
S <- 0.4^abs(outer(1:5, 1:5, "-"))
relimp_network(cor_matrix = S)
```

SRL	<i>Self-regulated-learning (MSLQ) construct scores simulated by large language models</i>
-----	---

Description

Five data sets of Motivated Strategies for Learning Questionnaire (MSLQ) construct scores, each a random sample of 300 cases generated by one large language model in the study *"Delving into the psychology of Machines: Exploring the structure of self-regulated learning via LLM-generated survey responses"* (Computers in Human Behavior, 2025). One data set per model: SRL_GPT, SRL_Gemini, SRL_Claude, SRL_Mistral, and SRL_LLaMa.

Usage

SRL_GPT

SRL_Gemini

SRL_Claude

SRL_Mistral

SRL_LLaMa

Format

Each is a data frame with 300 rows and 5 variables – the MSLQ construct scores (each the mean of that construct's Likert items, range 1–7):

CSU cognitive strategy use

IV intrinsic value

SE self-efficacy

SR self-regulation

TA test anxiety

Details

Only the five models whose responses carry estimable structure are included. Two other generators from the original study (ChatGPT and LeChat) produced near-independent items (mean absolute inter-item correlation ≈ 0.03), so their network is the empty graph; they are omitted. The five retained models span the spectrum the paper describes, from realistically structured (SRL_GPT) to strongly "over-coherent" (SRL_Gemini, SRL_Claude).

Source

Saqr, M. (2025). Delving into the psychology of Machines: Exploring the structure of self-regulated learning via LLM-generated survey responses. *Computers in Human Behavior*, 173, 108769. [doi:10.1016/j.chb.2025.108769](https://doi.org/10.1016/j.chb.2025.108769)

Examples

```
# partial-correlation network of the five constructs for one model
net <- ebic_glasso(SRL_GPT)
net
net_centralities(net)
```

summary.psychnet	<i>Summarize a psychnet network</i>
------------------	-------------------------------------

Description

Summarize a psychnet network

Usage

```
## S3 method for class 'psychnet'
summary(object, ...)
```

Arguments

object	A psychnet object.
...	Unused.

Value

The tidy edge list (invisibly); prints a summary as a side effect.

Examples

```
fit <- pcor_network(SRL_GPT)
summary(fit)
```

tmfg_certificate	<i>Structural certificate for a TMFG network</i>
------------------	--

Description

A TMFG has no convex objective, so its correctness is certified structurally: a valid TMFG on $p \geq 3$ nodes has exactly $3(p - 2)$ edges, is connected, and is chordal (every cycle of length ≥ 4 has a chord). Returns a non-negative score that is 0 for a valid TMFG.

Usage

```
tmfg_certificate(x)
```

Arguments

x	A psychnet object produced by tmfg_network() .
---	--

Value

Scalar; 0 certifies a valid TMFG (correct edge count, connected, chordal), otherwise a positive integer counting the violated invariants.

Examples

```
set.seed(1)
x <- matrix(stats::rnorm(200 * 6), 200, 6)
tmfg_certificate(tmfg_network(x))
```

tmfg_network	<i>Triangulated Maximally Filtered Graph (TMFG)</i>
--------------	---

Description

Builds a sparse, planar, chordal association network by greedily retaining the $3(p - 2)$ most informative edges (Massara et al. 2016). Equivalent in purpose to `NetworkToolbox::TMFG()` / `bootnet`'s "TMFG" default, pure base R; correctness is certified structurally by `tmfg_certificate()`.

Usage

```
tmfg_network(
  data = NULL,
  cor_matrix = NULL,
  n = NULL,
  cor_method = c("pearson", "spearman", "kendall", "auto"),
  na_method = c("pairwise", "listwise"),
  labels = NULL
)
```

Arguments

data	Numeric data frame or matrix (rows = observations). Optional if <code>cor_matrix</code> is supplied.
cor_matrix	Optional correlation matrix.
n	Accepted and ignored. TMFG is a structural filter and needs no sample size; the argument exists only so a uniform (<code>cor_matrix=</code> , <code>n=</code>) call shared with the other estimators does not partial-match <code>na_method</code> .
cor_method	Correlation when data is supplied: "pearson" (default), "spearman", "kendall", or "auto" (polychoric/polyserial; see <code>cor_auto()</code>).
na_method	Missing-data handling when data is supplied: "pairwise" (default) or "listwise". See <code>ebic_glasso()</code> .
labels	Optional node labels.

Value

A psychnet object whose `$weights` is the filtered (signed) correlation matrix on the retained edges, with `$adjacency`, `$cliques`, `$separators` (the chordal decomposition used by `logo_network()`), and `$cor_matrix`.

Examples

```
set.seed(1)
x <- matrix(stats::rnorm(200 * 6), 200, 6)
tmfg_network(x)
```

\$.psychnet

*Back-compatible field access for a psychnet object***Description**

The canonical (str-visible) fields are the lean netobject set; this method adds virtual aliases so older/external accessors keep working without storing redundant fields.

Usage

```
## S3 method for class 'psychnet'
x$name
```

Arguments

x	A psychnet object.
name	Field name. Canonical fields plus the legacy aliases graph (= weights), labels (= nodes\$label), n_nodes, n_edges, and n_obs (= n).

Value

The requested field, or NULL if neither a canonical field nor a known alias.

Examples

```
fit <- pcor_network(SRL_GPT)
fit$method
fit$labels
```

Index

- * **datasets**
 - SRL, [55](#)
- \$.psychnet, [59](#)
- as.data.frame.psychnet, [3](#)
- as.data.frame.psychnet_bootstrap, [4](#)
- as.data.frame.psychnet_group
 - (print.psychnet_group), [46](#)
- as.data.frame.psychnet_multilevel
 - (print.psychnet_multilevel), [48](#)
- as.data.frame.psychnet_result_group
 - (print.psychnet_result_group), [50](#)
- casedrop_reliability, [5](#)
- certificate, [6](#)
- cograph::splot(), [38](#)
- condition, [7](#)
- condition(), [24](#)
- cor_auto, [7](#)
- cor_auto(), [9](#), [12](#), [15](#), [22](#), [37](#), [54](#), [55](#), [58](#)
- cor_network, [8](#)
- dichotomize, [9](#)
- difference_test, [10](#)
- difference_test(), [27](#), [42](#)
- ebic_glasso, [11](#)
- ebic_glasso(), [14](#), [15](#), [18](#), [22](#), [55](#), [58](#)
- event_frequencies, [12](#)
- event_frequencies(), [53](#)
- ggm_modselect, [14](#)
- ggm_modselect(), [15](#)
- ggm_support_kkt, [15](#)
- ggm_support_kkt(), [14](#), [22](#)
- glasso_kkt, [16](#)
- glasso_kkt(), [11](#), [17](#), [18](#)
- glm_lasso_kkt, [17](#)
- glm_lasso_kkt(), [19](#), [20](#), [23](#)
- huge_network, [18](#)
- huge_network(), [14](#)
- ising_fit, [19](#)
- ising_fit(), [9](#), [20](#), [21](#), [33](#)
- ising_sampler, [20](#)
- ising_sampler(), [9](#), [33](#)
- lmg_certificate, [21](#)
- lmg_certificate(), [54](#), [55](#)
- logo_network, [22](#)
- logo_network(), [15](#), [58](#)
- mgm_fit, [23](#)
- mgm_fit(), [7](#), [33](#)
- net_aggregate, [26](#)
- net_boot, [27](#)
- net_boot(), [10](#), [38](#), [39](#), [46](#), [50](#)
- net_bridge, [28](#)
- net_bridge(), [39](#)
- net_centralities, [29](#)
- net_centralities(), [27](#), [33](#), [35](#), [41](#), [46](#), [50](#)
- net_clustering, [30](#)
- net_compare, [31](#)
- net_compare(), [43](#), [46](#)
- net_crosswalk, [32](#)
- net_edge_betweenness, [33](#)
- net_predict, [33](#)
- net_predict(), [21](#), [36](#), [46](#), [50](#)
- net_smallworld, [34](#)
- net_stability, [35](#)
- net_stability(), [5](#), [44](#), [46](#), [50](#)
- network_reliability, [25](#)
- node_predictability, [36](#)
- pcor_network, [36](#)
- plot.psychnet, [38](#)
- plot.psychnet_bootstrap, [38](#)
- plot.psychnet_bridge, [39](#)
- plot.psychnet_bridge(), [29](#)

plot.psychnet_casedrop, 40
plot.psychnet_casedrop(), 5
plot.psychnet_centrality, 41
plot.psychnet_difference, 42
plot.psychnet_nct, 43
plot.psychnet_reliability, 43
plot.psychnet_reliability(), 25
plot.psychnet_stability, 44
plot.psychnet_stability(), 35
print.psychnet, 45
print.psychnet_bootstrap, 45
print.psychnet_casedrop, 46
print.psychnet_group, 46
print.psychnet_moderated, 47
print.psychnet_multilevel, 48
print.psychnet_nct, 48
print.psychnet_redundancy, 49
print.psychnet_reliability, 49
print.psychnet_result_group, 50
print.psychnet_stability, 50
psychnet, 6, 21, 30, 34, 36, 46, 51, 57
psychnet(), 5, 12, 25–27, 35, 46, 48
psychnet_group, 50
psychnet_group(print.psychnet_group),
46

redundancy, 53
relimp_network, 54
relimp_network(), 21

SRL, 55
SRL_Claude (SRL), 55
SRL_Gemini (SRL), 55
SRL_GPT (SRL), 55
SRL_LLaMa (SRL), 55
SRL_Mistral (SRL), 55
stats::p.adjust, 9, 10, 21, 28, 31, 37
summary.psychnet, 57
summary.psychnet_group
(print.psychnet_group), 46

tmfg_certificate, 57
tmfg_certificate(), 58
tmfg_network, 58
tmfg_network(), 57