

Package: netsimhelpers (via r-universe)

October 8, 2026

Type Package
Title Helper Functions for Simulation Studies in Network Psychometrics
Version 0.1.0
Date 2026-08-04
Maintainer Ria H. A. Hoekstra <h.a.hoekstra@uva.nl>
Description Helper functions for setting up simulations in network psychometrics.
License GPL (>= 2)
Language en-US
Encoding UTF-8
RoxygenNote 7.3.3
Imports mvtnorm, stats, bootnet, qgraph
Suggests testthat (>= 3.0.0)
Config/testthat/edition 3
NeedsCompilation no
Author Ria H. A. Hoekstra [aut, cre], Sacha Epskamp [aut]
Repository <https://cran.r-universe.dev>
Date/Publication 2026-08-09 06:50:07 UTC
RemoteUrl <https://github.com/cran/netsimhelpers>
RemoteRef HEAD
RemoteSha 2dd41eb0ff4f1c0852eb2b3d7b6340de3138fe31

Contents

evalNet	2
genGVAR	3
genIsing	4
GGMsim	5
Index	7

evalNet

*Evaluate two networks against each other***Description**

Compute performance metrics for an estimated network relative to a known ground-truth network to assess estimation quality. Calculates how well the estimated network recovers the true network structure (which edges exist) and edge weights (how strong connections are).

Usage

```
evalNet(true, est, metric = NULL, directed = FALSE)
```

Arguments

<code>true</code>	The known ground-truth network A numeric matrix representing the true underlying network structure.
<code>est</code>	Estimated network to evaluate against <code>true</code> . A numeric matrix of the same dimensions as <code>true</code> , typically obtained from a network estimation method.
<code>metric</code>	Character vector specifying which metrics to compute. If <code>NULL</code> (default), all available metrics are computed and returned.
<code>directed</code>	Logical indicating whether the network should be treated as directed. If <code>FALSE</code> (default), only upper triangle is used. Set to <code>TRUE</code> for temporal networks where direction matters.

Value

A named list of evaluation metrics. Higher values generally indicate better recovery, except for bias metrics (lower is better). Returns NA for metrics that cannot be computed (e.g., when no edges are present).

See Also

[genIsing](#), [genGVAR](#) for generating networks

Examples

```
# Generate true GVAR networks
true_nets <- genGVAR(n_node = 6)

# Simulate estimated networks with some errors
est_nets <- true_nets

# Add false positive to contemporaneous network
est_nets$PCC[2, 5] <- est_nets$PCC[5, 2] <- 0.15

# Remove true edge
```

```

true_edge <- which(true_nets$PCC != 0, arr.ind = TRUE)[1, ]
est_nets$PCC[true_edge[1], true_edge[2]] <- 0
est_nets$PCC[true_edge[2], true_edge[1]] <- 0

# Evaluate contemporaneous network recovery
results_contemp <- evalNet(true = true_nets$PCC, est = est_nets$PCC)

# Inspect specific metrics
results_contemp$sensitivity
results_contemp$specificity

# Evaluate temporal network recovery
results_temp <- evalNet(true = true_nets$PDC, est = est_nets$PDC, directed = TRUE)
results_temp$in_strength_correlation

# Evaluate specific metrics only
evalNet(true_nets$PCC, est_nets$PCC,
        metric = c("sensitivity", "precision", "correlation"))

```

genGVAR

Generate a synthetic GVAR network

Description

Generate network structures for Graphical Vector Autoregression (GVAR) model. Generates both contemporaneous and temporal networks.

Usage

```

genGVAR(
  n_node,
  contemp_prop_pos = 0.8,
  temp_weight_mean = 0.3,
  temp_weight_sd = 0.1,
  ...
)

```

Arguments

n_node	Number of nodes in the network.
contemp_prop_pos	Proportion of edges that are positive in the contemporaneous network. Default is set to 0.8.
temp_weight_mean	Mean of the normal distribution used to generate edge weights for the temporal network.
temp_weight_sd	Standard deviation of the normal distribution used to generate edge weights for the temporal network.

... Any additional arguments passed to `bootnet::genGGM()` to specify contemporaneous network.

Value

A list with four elements:

kappa Precision matrix (inverse covariance) for contemporaneous network

PCC Partial correlation matrix for contemporaneous network

beta Temporal regression coefficients matrix

PDC Partial directed correlations for temporal network

Examples

```
# Generate a 5-node GVAR network
gvar_net <- genGVAR(n_node = 5)

# Check the networks
gvar_net$PCC # contemporaneous network
gvar_net$PDC # temporal network

# More positive edges in contemporaneous network
gvar_net2 <- genGVAR(n_node = 5, contemp_prop_pos = 0.9)

# Stronger temporal effects
gvar_net3 <- genGVAR(n_node = 5, temp_weight_mean = 0.5, temp_weight_sd = 0.2)
```

genIsing

Generate a synthetic Ising network

Description

Generate undirected Ising network structures and assign edge weights from a normal distribution.

Usage

```
genIsing(
  n_node,
  rewire = 0,
  inclusion = NULL,
  weight_mean = 0.3,
  weight_sd = 0.1,
  ...
)
```

Arguments

<code>n_node</code>	Number of nodes in the network.
<code>rewire</code>	Rewiring probability used when generating the graph structure using <code>bootnet::genGGM()</code> . Will be ignored when <code>inclusion</code> is provided.
<code>inclusion</code>	Probability of edge inclusion. If provided, each possible edge is included independently with probability <code>inclusion</code> .
<code>weight_mean</code>	Mean of the normal distribution used to generate edge weights.
<code>weight_sd</code>	Standard deviation of the normal distribution used to generate edge weights.
<code>...</code>	Additional arguments passed to <code>bootnet::genGGM()</code> when <code>inclusion</code> is <code>NULL</code> .

Value

A symmetric numeric matrix of size `n_node` by `n_node` with zeros on the diagonal. Off-diagonal entries are Ising interaction parameters (0 indicates absence of an edge)

Examples

```
# Generate a 5 node Ising model with default settings
ising1 <- genIsing(n_node = 5)

# Using edge inclusion probability
ising2 <- genIsing(n_node = 5, inclusion = 0.3)

# Change edge weight distribution
ising3 <- genIsing(n_node = 5, inclusion = 0.4,
                  weight_mean = 0.5, weight_sd = 0.2)
```

GGMsim

*Simulate data from a Gaussian Graphical Model (GGM)***Description**

Simulate multivariate normal data from a Gaussian Graphical Model (GGM) specified by a partial correlation matrix. Optionally introduces missing values completely at random (MCAR).

Usage

```
GGMsim(n_obs, omega, missing_prop = 0)
```

Arguments

<code>n_obs</code>	Number of observations (rows) to simulate.
<code>omega</code>	Partial correlation matrix that defines the GGM. Must be symmetric square matrix with zeros on the diagonal.
<code>missing_prop</code>	Proportion of values to set to missing (MCAR) per variable. Default is set to 0 (no missingness).

Value

A matrix with `n_obs` rows and `ncol(omega)` columns. Values are simulated from a multivariate normal distribution implied by the provided partial correlation matrix `omega` with optional NAs.

Examples

```
# Construct a 5 node network
omega <- matrix(0, 5, 5)
omega[1, 2] <- omega[2, 1] <- 0.3
omega[2, 3] <- omega[3, 2] <- 0.4

# Simulate data
data <- GGMsim(n_obs = 100, omega = omega)

# Simulate data with missing values
data_missing <- GGMsim(n_obs = 100, omega = omega, missing_prop = 0.1)
```

Index

evalNet, [2](#)

genGVAR, [2](#), [3](#)

genIsing, [2](#), [4](#)

GGMsim, [5](#)