Package ‘hergm’

June 3, 2018

Version 3.2-1
Date 2018-06-03
Title Hierarchical Exponential-Family Random Graph Models
Author Michael Schweinberger <michael.schweinberger@rice.edu> [aut, cre], Mark S. Handcock <handcock@ucla.edu> [ctb], Pamela Luna <pamela.luna@rice.edu> [ctb]
Maintainer Michael Schweinberger <michael.schweinberger@rice.edu>
Depends ergm, latentnet, mcgibbsit, network, sna
Imports methods
Description Hierarchical exponential-family random graph models with local dependence.
License GPL-3
NeedsCompilation yes

BugReports http://www.stat.rice.edu/~ms88/hergm/BugReports/BugReports.html
Repository CRAN
Date/Publication 2018-06-03 21:12:58 UTC

R topics documented:

bali .................................................. 2
bunt .................................................. 2
example ............................................. 3
gof.hergrm ........................................... 4
hergms .............................................. 5
hergm-terms ....................................... 11
hergm.mcmc.diagnostics .......................... 13
hergm.postprocess ................................. 14
kapferer ............................................ 17
plot.hergrm ....................................... 17
print.hergrm ...................................... 18
simulate.hergrm .................................. 19
summary.hergrm ................................... 21

Index 22
bali  

*Bal*ti *terrorist network*

**Description**

The network corresponds to the contacts between the 17 terrorists who carried out the bombing in Bali, Indonesia in 2002. The network is taken from Koschade (2006).

**Usage**

```r
data(bali)
```

**Value**

Undirected network.

**References**


**See Also**

network, hergm, ergm.terms, hergm.terms

**Examples**

```r
## Not run: data(bali)
hergm(bali ~ edges_ij + triangle_iijk)
## End(Not run)
```

---

bunt  

*Van de Bunt* friendship network

**Description**

Van de Bunt (1999) and Van de Bunt et al. (1999) collected data on friendships between 32 freshmen at a European university at 7 time points. Here, the last time point is used. A directed edge from student i to j indicates that student i considers student j to be a “friend” or “best friend”.

**Usage**

```r
data(bunt)
```
Value

Directed network.

References


See Also

network, hergm, ergm.terms, hergm.terms

Examples

```r
## not run: data(bunt)
hergm(bunt ~ edges_ij + ttriple_ijk)
## end(not run)
```

---

**Example network**

Description

Example data set: synthetic, undirected network with 15 nodes.

Usage

```r
data(example)
```

Value

Undirected network.

See Also

network, hergm, ergm.terms, hergm.terms
Examples

```r
## Not run: data(example)
hergm(d ~ edges_i)
hergm(d ~ edges_ij + triangle_ijk)
## End(Not run)
```

gof.hergm

### Goodness-of-fit

The function `gof.hergm` accepts an object of class `hergm` as argument and assesses the goodness-of-fit of the model estimated by function `hergm`.

#### Usage

```r
## S3 method for class 'hergm'
gof(object, ...)
```

#### Arguments

- `object` object of class `hergm`; objects of class `hergm` can be generated by function `hergm`.
- `...` additional arguments, to be passed to lower-level functions in the future.

#### Value

The function `gof.hergm` returns a list with components:

- `component.number` number of components.
- `max.component.size` size of largest component.
- `distance` geodesic distance of pairs of nodes.
- `degree` degree of nodes.
- `edges` number of edges.
- `stars` number of 2-stars.
- `triangle` number of triangles.

#### References

hrgm

See Also

hrgm, simulate.hrgm

Examples

```r
## not run: data(example)
object <- hrgm(d ~ edges_ij + triangle_ijk)
gof(object)
## End(not run)
```

hrgm

Hierarchical exponential-family random graph models with local dependence

Description

The function hrgm estimates and simulates three classes of hierarchical exponential-family random graph models:

1. The \( p_1 \) model of Holland and Leinhardt (1981) in exponential-family form and extensions by Vu, Hunter, and Schweinberger (2013) and Schweinberger, Petrescu-Prahova, and Vu (2014) to both directed and undirected random graphs with additional model terms, with and without covariates, and with parametric and nonparametric priors (see \( \text{arcs}_i \), \( \text{arcs}_j \), \( \text{edges}_i \), \( \text{edges}_{ij} \), \( \text{mutual}_i \), \( \text{mutual}_{ij} \)).

2. The stochastic block model of Snijders and Nowicki (1997) and Nowicki and Snijders (2001) in exponential-family form and extensions by Vu, Hunter, and Schweinberger (2013) and Schweinberger, Petrescu-Prahova, and Vu (2014) with additional model terms, with and without covariates, and with parametric and nonparametric priors (see \( \text{arcs}_i \), \( \text{arcs}_j \), \( \text{edges}_i \), \( \text{edges}_{ij} \), \( \text{mutual}_i \), \( \text{mutual}_{ij} \)).

3. The exponential-family random graph models with local dependence of Schweinberger and Handcock (2015), with and without covariates, and with parametric and nonparametric priors (see \( \text{arcs}_i \), \( \text{arcs}_j \), \( \text{edges}_i \), \( \text{edges}_{ij} \), \( \text{mutual}_i \), \( \text{mutual}_{ij} \), \( \text{twostar}_{ijk} \), \( \text{triangle}_{ijk} \), \( \text{ttriple}_{ijk} \), \( \text{ctriple}_{ijk} \)). The exponential-family random graph models with local dependence replace the long-range dependence of conventional exponential-family random graph models by short-range dependence. Therefore, the exponential-family random graph models with local dependence replace the strong dependence of conventional exponential-family random graph models by weak dependence, reducing the problem of model degeneracy (Handcock, 2003; Schweinberger, 2011) and improving goodness-of-fit (Schweinberger and Handcock, 2015).

Usage

```r
hrgm(formula,
    max_number = NULL,
    hierarchical = TRUE,
    parametric = FALSE,
```
initialize = FALSE,
perturb = FALSE,
scaling = NULL,
alpha = NULL,
alpha_shape = NULL,
alpha_rate = NULL,
etta = NULL,
etta_mean = NULL,
etta_sd = NULL,
etta_mean_mean = NULL,
etta_mean_sd = NULL,
etta_precision_shape = NULL,
etta_precision_rate = NULL,
mean_between = NULL,
indicator = NULL,
parallel = 1,
simulate = FALSE,
seeds = NULL,
sample_size = 1e+5,
interval = 1024,
burnin = 16*interval,
mh.scale = 0.25,
vvaritional = FALSE,
temperature = c(1,100),
predictions = FALSE,
posterior.burnin = 2000,
prioror.thinning = 1,
relabel = 1,
number_runs = 1,
verbose = 0,
...)

Arguments

formula formula of the form network ~ terms. network is an object of class network and can be created by calling the function network. Possible terms can be found in ergm.terms and hergm.terms.

max_number maximum number of blocks.

hierarchical hierarchical prior; if hierarchical == TRUE, prior is hierarchical (i.e., the means and variances of block parameters are governed by a hyper-prior), otherwise non-hierarchical (i.e., the means and variances of block parameters are fixed).

parametric parametric prior; if parametric == FALSE, prior is truncated Dirichlet process prior, otherwise parametric Dirichlet prior.

initialize if initialize == TRUE, initialize block memberships of nodes by spectral clustering.
perturb if initialize == TRUE and perturb == TRUE, initialize block memberships
of nodes by spectral clustering and perturb.

scaling if scaling == TRUE, use size-dependent parameterizations which ensure that
the scaling of between- and within-neighborhood terms is consistent with sparse
edge terms.

alpha concentration parameter of truncated Dirichlet process prior of natural param-
ters of exponential-family model.

alpha_shape, alpha_rate
shape and rate parameter of Gamma prior of concentration parameter.

eta the parameters of ergm.terms and hergm.terms; the parameters of hergm.terms
must consist of max_number within-neighborhood parameters and one between-
neighborhood parameter.

eta_mean, eta_sd
means and standard deviations of Gaussian baseline distribution of Dirichlet
process prior of natural parameters.

eta_mean_mean, eta_mean_sd
means and standard deviations of Gaussian prior of mean of Gaussian baseline
distribution of Dirichlet process prior.

eta_precision_shape, eta_precision_rate
shape and rate (inverse scale) parameter of Gamma prior of precision parameter
of Gaussian baseline distribution of Dirichlet process prior.

mean_between if simulate == TRUE and eta == NULL, then mean_between specifies the
mean-value parameter of edges between blocks.

indicator if the indicators of block memberships of nodes are specified as integers between
1 and max_number, the specified indicators are fixed, which is useful when indi-
cators of block memberships are observed (e.g., in multilevel networks).

parallel number of computing nodes; if parallel > 1, hergm is run on parallel
computing nodes.

simulate if simulate == TRUE, simulation of networks, otherwise Bayesian inference.

seeds seed of pseudo-random number generator; if parallel > 1, number of seeds
must equal number of computing nodes.

sample_size if simulate == TRUE, number of network draws, otherwise number of posterior
draws; if parallel > 1, number of draws on each computing node.

interval if simulate == TRUE, number of proposals between sampled networks.

burnin if simulate == TRUE, number of burn-in iterations.

mh_scale if simulate == FALSE, scale factor of candidate-generating distribution of
Metropolis-Hastings algorithm.

variational if simulate == FALSE and variational == TRUE, variational methods are
used to construct the proposal distributions of block memberships of nodes; lim-
ited to selected models.

temperature if simulate == FALSE and variational == TRUE, minimum and maximum
temperature; the temperature is used to melt down the proposal distributions of
indicators, which are based on the full conditional distributions of indicators
but can have low entropy, resulting in slow mixing of the Markov chain; the
temperature is a function of the entropy of the full conditional distributions and
is designed to increase the entropy of the proposal distributions, and the mini-
mum and maximum temperature are user-defined lower and upper bounds on
the temperature.

predictions if predictions == TRUE and simulate == FALSE, returns posterior predic-
tions of statistics in the model.

posterior.burnin

number of posterior burn-in iterations; if computing is parallel, posterior.burnin
is applied to the sample generated by each processor; please note that hergm re-
turns min(sample_size, 10000) sample points and the burn-in is applied to the
sample of size min(sample_size, 10000), therefore posterior.burnin should
be smaller than min(sample_size, 10000).

posterior.thinning

if posterior.thinning > 1, every posterior.thinning-th sample point is
used while all others discarded; if computing is parallel, posterior.thinning
is applied to the sample generated by each processor; please note that hergm
returns min(sample_size, 10000) sample points and the thinning is applied to the
sample of size min(sample_size, 10000) - posterior.burnin, there-
fore posterior.thinning should be smaller than min(sample_size, 10000) -
posterior.burnin.

relabel

if relabel > 0, relabel MCMC sample by minimizing the posterior expected
loss of Schweinberger and Handcock (2015) (relabel == 1) or Peng and Car-

number_runs

if relabel == 1, number of runs of relabeling algorithm.

verbose

if verbose == -1, no console output; if verbose == 0, short console output;
if verbose == +1, long console output. If, e.g., simulate == FALSE and
verbose == 1, then hergm reports the following console output:

Progress: 50.00% of 1000000

... means of block parameters: -0.2838 1.3323
precisions of block parameters: 0.9234 1.4682
block parameters:
-0.2544 -0.2560 -0.1176 -0.0310 -0.1915 -1.9626
0.4022 1.8887 1.9719 0.6499 1.7265 0.0000
block indicators: 1 3 1 1 1 3 1 1 2 2 2 1 1 1
block sizes: 10 5 2 0 0
block probabilities: 0.5396 0.2742 0.1419 0.0423 0.0020
block probabilities prior parameter: 0.4256
posterior prediction of statistics: 66 123

where ... indicates additional information about the Markov chain Monte Carlo
algorithm that is omitted here. The console output corresponds to:

- "means of block parameters" correspond to the mean parameters of the Gaus-
sian base distribution of parameters of hergm-terms.
"precisions of block parameters" correspond to the precision parameters of the Gaussian base distribution of parameters of hergm-terms.
- "block parameters" correspond to the parameters of hergm-terms.
- "block indicators" correspond to the indicators of block memberships of nodes.
- "block sizes" correspond to the block sizes.
- "block probabilities" correspond to the prior probabilities of block memberships of nodes.
- "block probabilities prior parameter" corresponds to the concentration parameter of truncated Dirichlet process prior of parameters of hergm-terms.
- if predictions == TRUE, "posterior prediction of statistics" correspond to posterior predictions of sufficient statistics.

**Value**

The function hergm returns an object of class hergm with components:

- network
  - network is an object of class network and can be created by calling the function network.
- formula
  - formula of the form network ~ terms. network is an object of class network and can be created by calling the function network. Possible terms can be found in ergm.terms and hergm.terms.
- n
  - number of nodes.
- hyper_prior
  - indicator of whether hyper prior has been specified, i.e., whether the parameters alpha, eta_mean, and eta_precision are estimated.
- alpha
  - concentration parameter of truncated Dirichlet process prior of parameters of hergm-terms.
- ergm_theta
  - parameters of ergm-terms.
- eta_mean
  - mean parameters of Gaussian base distribution of parameters of hergm-terms.
- eta_precision
  - precision parameters of Gaussian base distribution of parameters of hergm-terms.
- d1
  - total number of parameters of ergm terms.
- d2
  - total number of parameters of hergm terms.
- hergm_theta
  - parameters of hergm-terms.
- relabeled.hergm_theta
  - relabeled parameters of hergm-terms by using relabel = 1 or relabel = 2.
- number_fixed
  - number of fixed indicators of block memberships of nodes.
- indicator
  - indicators of block memberships of nodes.
- relabel
  - if relabel > 0, relabel MCMC sample by minimizing the posterior expected loss of Schweinberger and Handcock (2015) (relabel == 1) or Peng and Carvalho (2015) (relabel == 2).
- relabeled.indicator
  - relabeled indicators of block memberships of nodes by using relabel = 1 or relabel = 2.
size the size of the blocks, i.e., the number of nodes of blocks.

parallel number of computing nodes; if parallel > 1, hergm is run on parallel computing nodes.

p_i_k posterior probabilities of block membership of nodes.

p_k probabilities of block memberships of nodes.

predictions if predictions == TRUE and simulate == FALSE, returns posterior predictions of statistics in the model.

simulate if simulate == TRUE, simulation of networks, otherwise Bayesian inference.

prediction posterior predictions of statistics.

edgelist edge list of simulated network.

sample_size if simulate == TRUE, number of network draws, otherwise number of posterior draws minus number of burn-in iterations; if parallel > 1, number of draws on each computing node.

extract indicator of whether function hergm.postprocess has postprocessed the object of class hergm generated by function hergm and thus whether the MCMC sample generated by function hergm has been extracted from the object of class hergm.

convergence.diagnostics
MCMC diagnostics generated by function mcmc.diagnostics, which in turn relies on function mcgibbsit in R package mcgibbsit; see ?mcgibbsit.

verbose if verbose == -1, no console output; if verbose == 0, short console output; if verbose == +1, long console output.

References


See Also

network, ergm.terms, hergm.terms, hergm.postprocess, mcmc.diagnostics.hergm, summary.hergm, print.hergm, plot.hergm, gof.hergm, simulate.hergm

Examples

```r
## not run: data(example)
hergm(d ~ edges_i)
hergm(d ~ edges_ij)
hergm(d ~ edges_ij + triangle_ijk)
data(sampson)
hergm(samplike ~ arcs_i + arcs_j)
hergm(samplike ~ edges_ij + mutual_ij)
hergm(samplike ~ edges_ij + mutual_ij + ttriple_ijk)
## end(not run)
```

<table>
<thead>
<tr>
<th>hergm-terms</th>
<th>Model terms</th>
</tr>
</thead>
</table>

Description

Hierarchical exponential-family random graph models with local dependence can be specified by calling the function `hergm(formula)`, where formula is a formula of the form `network ~ terms`. By specifying suitable terms, it is possible to specify a wide range of models: see `hergm`. `hergm.terms` can be found here. In addition, `ergm.terms` can be used to include covariates.

Arguments

- **edges_i** (undirected network)
  - adding the term `edges_i` to the model adds node-dependent edge terms to the model.
- **arcs_i** (directed network)
  - adding the term `arcs_i` to the model adds node-dependent outdegree terms to the model.
- **arcs_j** (directed network)
  - adding the term `arcs_j` to the model adds node-dependent indegree terms to the model.
- **edges_ij** (undirected, directed network)
  - adding the term `edges_ij` to the model adds block-dependent edge terms to the model.
- **mutual_i** (directed network)
  - adding the term `mutual_i` to the model adds additive, block-dependent mutual edge terms to the model.
mutual_ij (directed network)
adding the term mutual_ij to the model adds block-dependent mutual edge
terms to the model.
twostar_ijk (undirected network)
adding the term twostar_ijk to the model adds block-dependent two-star terms
to the model;
transitiveties_ijk (directed network)
adding the term transitiveties_ijk to the model adds block-dependent tran-
sitive ties terms to the model.
triangle_ijk (undirected, directed network)
adding the term triangle_ijk to the model adds block-dependent triangle terms
to the model.
ttriple_ijk (directed network)
adding the term ttriple_ijk to the model adds block-dependent transitive
triple terms to the model.
ctriple_ijk (directed network)
adding the term ctriple_ijk to the model adds block-dependent cyclic triple
terms to the model.

References
graphs with latent block structure. Journal of Classification 14, 75–100.
Schweinberger, M. and M. S. Handcock (2015). Local dependence in random graph models: character-
zation, properties, and statistical Inference. Journal of the Royal Statistical Society, Series B
(Statistical Methodology), 7, 647-676.
Schweinberger, M., Petrescu-Prahova, M. and D. Q. Vu (2014). Disaster response on September

See Also
hergm, ergm.terms
Examples

```r
## not run:
data(example)
# p_1 model: undirected network
hergm(d ~ edges_i)

data(sampson)
# p_1 model: directed network
hergm(samplike ~ arcs_i + arcs_j + mutual)

data(example)
# Stochastic block model: undirected network
hergm(d ~ edges_ij)

data(sampson)
# Stochastic block model: directed network
hergm(samplike ~ edges_ij + mutual)

data(example)
# Exponential-family random graph model with local dependence: undirected network
hergm(d ~ edges_ij + triangle_ijk)

data(sampson)
# Exponential-family random graph model with local dependence: directed network
hergm(samplike ~ edges + mutual + ttriple_ijk)

## End(not run)
```

**hergm.mcmc.diagnostics**

MCMC diagnostics of objects of class hergm

Description

The function `mcmc.diagnostics.hergm` helps detect non-convergence of the auxiliary-variable MCMC algorithm implemented in function `hergm`. It reports Markov chain Monte Carlo convergence diagnostics by using the function `mcmcsit` of R package `mcmcsit` along with trace plots. The help function of the function `mcmcsit` provides additional details about the output of the function `mcmcsit`.

Usage

```r
## S3 method for class 'hergm'
mcmc.diagnostics(object, ...)
```
Arguments

object object of class hergm; objects of class hergm can be generated by function hergm.

... additional arguments, to be passed to lower-level functions in the future.

Value

The function mcmc.diagnostics returns a list with the following components:

- mcmc.alpha MCMC diagnostics for the concentration parameter of truncated Dirichlet prior of parameters of hergm-terms.
- mcmc.eta_mean MCMC diagnostics for the mean parameters of Gaussian base distribution of parameters of hergm-terms.
- mcmc.eta_precision MCMC diagnostics for the precision parameters of Gaussian base distribution of parameters of hergm-terms.
- mcmc.ergm_theta MCMC diagnostics for the parameters of ergm-terms.
- mcmc.hergm_theta MCMC diagnostics for the parameters of hergm-terms.

References


See Also

hergm

Examples

```r
## not run: data(example)
object <- hergm(d ~ edges_ij + triangle_ijk)
mcmc.diagnostics(object)

## End(not run)
```
hergm.postprocess

Description

The function hergm.postprocess postprocesses an object of class hergm. Please note that the function hergm calls the function hergm.postprocess with relabel = 0 by default or with other values of relabel specified by the user, therefore users do not need to call the function hergm.postprocess unless it is desired to postprocess an object of class hergm with a value of relabel that was not used by function hergm.

If hergm.postprocess is called with relabel > 0, it solves the so-called label-switching problem. The label-switching problem is rooted in the invariance of the likelihood function to permutations of the labels of blocks, and implies that raw MCMC samples from the posterior cannot be used to infer to block-dependent entities. The label-switching problem can be solved in a Bayesian decision-theoretic framework: by choosing a loss function and minimizing the posterior expected loss. Two loss functions are implemented in hergm.postprocess, the loss function of Schweinberger and Handcock (2015) (relabel == 1) and the loss function of Peng and Carvalho (2015) (relabel == 2). The first loss function seems to be superior in terms of the reported clustering probabilities, but is more expensive in terms of computing time. A rule of thumb is to use the first loss function when max_number < 15 and use the second loss function otherwise.

Usage

hergm.postprocess(object,
burnin = 2000,
thinning = 1,
relabel = 1,
number_runs = 1,
...)

Arguments

object object of class hergm; objects of class hergm can be generated by function hergm.
burnin number of posterior burn-in iterations; if computing is parallel, burnin is applied to the sample generated by each processor; please note that hergm returns min(sample_size, 10000) sample points and the burn-in is applied to the sample of size min(sample_size, 10000), therefore burnin should be smaller than min(sample_size, 10000).
thinning if thinning > 1, every thinning-th sample point is used while all others discarded; if computing is parallel, thinning is applied to the sample generated by each processor; please note that hergm returns min(sample_size, 10000) sample points and the thinning is applied to the sample of size min(sample_size, 10000) - burnin, therefore thinning should be smaller than min(sample_size, 10000) - burnin.
relabel if relabel > 0, relabel MCMC sample by minimizing the posterior expected loss of Schweinberger and Handcock (2015) (relabel == 1) or Peng and Carvalho (2015) (relabel == 2).
number_runs if relabel == 1, number of runs of relabeling algorithm.
... additional arguments, to be passed to lower-level functions in the future.

**Value**

- `ergm_theta`: parameters of `ergm-terms`.
- `alpha`: concentration parameter of truncated Dirichlet process prior of parameters of `ergm-terms`.
- `eta_mean`: mean parameters of Gaussian base distribution of parameters of `ergm-terms`.
- `hergm_theta`: parameters of `hergm-terms`.
- `loss`: if `relabel == TRUE`, local minimum of loss function.
- `p_k`: probabilities of block memberships of nodes.
- `indicator`: indicators of block memberships of nodes.
- `p_i_k`: posterior probabilities of block memberships of nodes.
- `prediction`: posterior predictions of statistics.

**References**


**See Also**

hergm

**Examples**

```r
## Not run: data(example)
object <- hergm(d ~ edges_ij + triangle_ijk)
hergm.postprocess(object)

## End(Not run)
```
Description

The network corresponds to collaborations between 39 workers in a tailor shop in Africa: an undirected edge between workers \( i \) and \( j \) indicates that the workers collaborated. The network is taken from Kapferer (1972).

Usage

```r
data(kapferer)
```

Value

Undirected network.

References


See Also

network, hrgm, ergm.terms, hrgm.terms

Examples

```r
## Not run: data(kapferer)
hergm(kapferer ~ edges_ij + triangle_ijk)
## End(Not run)
```

Description

The function `plot.hrgm` accepts an object of class `hrgm` as argument and plots a summary of a sample of block memberships of nodes from the posterior. Please note that the function `hrgm` should have been called with `relabel > 0` to solve the so-called label-switching problem, which is done by default. If the function `hrgm` has not been called with option `relabel > 0`, call the function `hrgm.postprocess` with `relabel > 0`. 

```
plot.hrgm
```

Plot summary of object of class `hrgm`
Usage

## S3 method for class 'hergm'
plot(x, threshold = c(.7, .8, .9), ...)

Arguments

x
  object of class hergm; objects of class hergm can be generated by function hergm.

threshold
  if the component relabel of the object of class hergm is relabel = 3, then threshold is a vector of thresholds between 0 and 1, indicating the thresholds at which the same-block-membership posterior probabilities of nodes are to be thresholded to construct the same-block graphs.

... additional arguments, to be passed to lower-level functions in the future.

References


See Also

hergm, hergm.postprocess, print.hergm, summary.hergm

Examples

## Not run: data(example)
object <- hergm(d ~ edges_ij + triangle_ijk)
plot(object)

## End(Not run)

print.hergm  

Print summary of object of class hergm

Description

The function print.hergm accepts an object of class hergm as argument and prints a summary of parameters from the posterior. Please note that the function hergm should have been called with relabel > 0 to solve the so-called label-switching problem, which is done by default. If the function hergm has not been called with option relabel > 0, call the function hergm.postprocess with relabel > 0.
Usage

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

Arguments

x          object of class hergm; objects of class hergm can be generated by function
...         hergm.

References


See Also

hergm, hergm.postprocess, plot.hergm, summary.hergm

Examples

## Not run: data(example)
object <- hergm(d ~ edges_ij + triangle_ijk)
print(object)

## End(Not run)

simulate.hergm Simulate network

Description

The function simulate.hergm accepts an object of class hergm as argument and simulates
networks.

Usage

## S3 method for class 'hergm'
simulate(object,           max_number = NULL,
           indicator = NULL,
           eta = NULL,
           sample_size = 1,
           verbose = 0,
           ...)


simulate.hergm

Arguments

object either object of class hergm or formula of the form network \sim terms; objects of class hergm can be generated by function hergm; network is an object of class network and can be created by calling the function network; possible terms can be found in ergm.terms and hergm.terms.

max_number maximum number of blocks.

indicator indicators of block memberships of nodes.

eta ergm.terms and hergm.terms parameters.

sample_size number of networks to be simulated.

verbose if verbose == -1, no console output; if verbose == 0, short console output; if verbose == +1, long console output.

... additional arguments, to be passed to lower-level functions in the future.

Value

The function simulate.hergm returns the simulated networks in the form of edge lists.

References


See Also

hergm, ergm.terms, hergm.terms, gof.hergm

Examples

## Not run: data(example)

# Simulate network given 'object' of class 'hergm':
object <- hergm(d ~ edges_ij + triangle_ijk)
simulate.hergm(object)

# Simulate network given 'formula':
indicator <- c(rep.int(1, 10), rep.int(2, 10))
eta <- c(-1, -1, -2, 1, 1, 0)
simulate.hergm(d ~ edges_ij + triangle_ijk, max_number = 2, indicator = indicator, eta = eta)

## End(Not run)
Summary of object of class hergm

Description

The function `summary.hergm` generates a summary of an object of class `hergm` by using the functions `print.hergm` and `plot.hergm`. The function `print.hergm` prints a summary of a sample of parameters from the posterior, whereas the function `plot.hergm` plots a summary of a sample of block memberships of nodes from the posterior.

Usage

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

Arguments

- `object` : object of class `hergm`; objects of class `hergm` can be generated by function `hergm`.
- `...` : additional arguments, to be passed to lower-level functions in the future.

References


See Also

`hergm`, `hergm.postprocess`, `print.hergm`, `plot.hergm`

Examples

```r
## Not run: data(example)
object <- hergm(d ~ edges_ij + triangle_ijk)
summary(object)

## End(Not run)
```
Index

arcs_i (hergm-terms), 11
arcs_j (hergm-terms), 11

bali, 2
bunt, 2
ctriple_ijk (hergm-terms), 11
d (example), 3
edges_i (hergm-terms), 11
edges_ij (hergm-terms), 11
example, 3
gof.hergm, 4
hergm, 5
hergm-terms, 11
hergm.gof (gof.hergm), 4
hergm.mcmc.diagnostics, 13
hergm.plot (plot.hergm), 17
hergm.postprocess, 14
hergm.print (print.hergm), 18
hergm.simulate (simulate.hergm), 19
hergm.summary (summary.hergm), 21
hergm.terms (hergm-terms), 11

kapferer, 17
mcmc.diagnostics.hergm

  (hergm.mcmc.diagnostics), 13
mutual_i (hergm-terms), 11
mutual_ij (hergm-terms), 11
plot (plot.hergm), 17
plot.hergm, 17
postprocess.hergm (hergm.postprocess), 14
print.hergm, 18
simulate (simulate.hergm), 19

twostar_ijk (hergm-terms), 11
transitivities_ijk (hergm-terms), 11
triangle_ijk (hergm-terms), 11
ttriple_ijk (hergm-terms), 11

simulate.hergm, 19
summary.hergm, 21

terms-hergm (hergm-terms), 11
terms.hergm (hergm-terms), 11

22