Package 'hergm'

July 2, 2014

Version 1.3-11

Date 2013-12-01

Title Hierarchical Exponential-Family Random Graph Models with Local Dependence

Author Michael Schweinberger <michael.schweinberger@rice.edu>, Mark S. Handcock <handcock@ucla.edu>

Maintainer Michael Schweinberger <michael.schweinberger@rice.edu>

Depends ergm (>= 3.1-1), parallel

Description The R package 'hergm' implements Hierarchical Exponential-Family Random Graph Models (HERGMs), which can be used to model a wide range of relational data (networks). 'hergm' implements both simulation and Bayesian inference.

License GPL-3

URL http://cran.r-project.org/web/packages/hergm/index.html

BugReports

NeedsCompilation yes

Repository CRAN

Date/Publication 2013-12-01 18:52:26

R topics documented:

bali *Terrorist network behind Bali bombing in 2002*

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

```
data(bali)
```

Value

Undirected network.

References

Koschade, S. (2006). A social network analysis of Jemaah Islamiyah: The applications to counter-terrorism and intelligence. Studies in Conflict and Terrorism, 29, 559–575.

See Also

network, hergm, ergm.terms, hergm.terms, hergm.postprocess

Examples

```
## Not run: # Load undirected network with 17 nodes
data(bali)

# Hierarchical exponential-family model with stick-breaking prior
hergm(bali ~ edges_ij(5) + triangle_ijk(5))


## End(Not run)
```

example *Example data set*

Description

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

Usage

```
data(example)
```

Value

Undirected network.

See Also

network, hergm, ergm.terms, hergm.terms, hergm.postprocess

Examples

```
## Not run: # Load undirected network with 15 nodes
data(example)

# p_1 model for undirected network with Dirichlet process prior
hergm(d ~ edges_i)

# Hierarchical exponential-family model with stick-breaking prior
hergm(d ~ edges + triangle_ijk)

## End(Not run)
```

hergm *Hierarchical Exponential-Family Random Graph Models: Simulation and Bayesian inference*

Description

The package hergm implements exponential-family random graph models with Dirichlet process / stick-breaking priors, including

- the p_1 model for directed networks of Holland and Leinhardt (1981) and its extension to undirected random graph models with Dirichlet process priors (see arcs_i, arcs_j, mutual_i, and edges_i). While the p_1 model for undirected and directed networks with parametric priors contains O(n) parameters (n = number of nodes) and therefore is not parsimonious, the non-parametric Dirichlet

process prior encourages a small number of unique parameters and therefore represents an elegant alternative to parametric priors.

- the stochastic block model of Snijders and Nowicki (1997) and Nowicki and Snijders (2001) with natural parameterization (restricted between-block parameters) and Dirichlet process priors (see edges_ij).

- hierarchical exponential-family models with stick-breaking priors (see mutual_ij, twostar_ijk, triangle_ijk, ttriple_ijk, ctriple_ijk). Hierarchical exponential-family models replace the strong dependence of simple exponential-family models by weak dependence with an eye to solving the near-degeneracy problem of simple exponential-family model.

The package `hergm` implements simulation and Bayesian inference for the mentioned models.

Usage

```
hergm(formula,
      alpha = NULL,
      alpha_shape = NULL,
      alpha_rate = NULL,
      eta = NULL,
      eta_mean = NULL,
      eta_sd = NULL,
      eta_mean_mean = NULL,
      eta_mean_sd = NULL,
      eta_precision_shape = NULL,
      eta_precision_rate = NULL,
      indicator = NULL,
      parallel = 1,
      simulate = FALSE,
      seeds = NULL,
      samplesize = 1e+5,
      burnin = 1e+4,
      interval = 1e+2,
      mh_scale = NULL,
      temperature = c(1,10),
      output = TRUE,
      verbose = -1,
      name = NULL,
      ...)
```

Arguments

`formula`	formula of the form `network ~ terms`. Networks can be created by calling the function `network`. Possible terms can be found in `ergm.terms` and `hergm.terms`.
`alpha`	scaling parameter of truncated Dirichlet process / stick-breaking prior of natural parameters of exponential-family model.

`alpha_shape, alpha_rate`
: shape and rate (inverse scale) parameter of Gamma prior of scaling parameter.

`eta`
: natural parameters of exponential-family model.

`eta_mean, eta_sd`
: means and standard deviations of Gaussian baseline distribution of Dirichlet process / stick-breaking 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 / stick-breaking 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 / stick-breaking prior.

`indicator`
: if `simulate == TRUE` and indicator of block membership is specified, condition on specified indicator (of block membership).

`parallel`
: number of processors; if more than one, computing is parallel.

`simulate`
: if `TRUE`, simulation of networks, otherwise Bayesian inference.

`seeds`
: seed of pseudo-random number generator; if computing is parallel, number of seeds must equal number of processors.

`samplesize`
: if `simulate == TRUE`, number of networks to be sampled, otherwise number of draws from posterior; if computing is parallel, number of draws per processors.

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

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

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

`temperature`
: 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 minimum and maximum temperature are user-defined lower and upper bounds on the temperature.

`output`
: if `TRUE`, full output, otherwise limited output.

`name`
: name of project; if `output == TRUE`, name of project is used to name output files.

`verbose`
: console output: -1: no output; 0: short output; +1: long output.

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

Value

If called with the option `simulate = TRUE`, the function `hergm` returns a sample of networks, otherwise a raw MCMC sample from the posterior. To postprocess the sample, call the function `hergm.postprocess`.

References

Holland, P. W. and S. Leinhardt (1981). An exponential family of probability distributions for directed graphs. Journal of the American Statistical Association 76 (373), 33–65.

Nowicki, K. and T. A. B. Snijders (2001). Estimation and prediction for stochastic blockstructures. Journal of the American Statistical Association 96 (455), 1077–1087.

Snijders, T. A. B. and K. Nowicki (1997). Estimation and prediction for stochastic blockmodels for graphs with latent block structure. Journal of Classification 14, 75–100.

See Also

network, ergm.terms, hergm.terms, hergm.postprocess

Examples

```
## Not run: # Load undirected network with 15 nodes (see ?example)
data(example)

# p_1 model for undirected network with Dirichlet process prior
hergm(d ~ edges_i)

# Stochastic block model for undirected network
# with natural parameterization and Dirichlet process prior
hergm(d ~ edges_ij)

# Hierarchical exponential-family model with stick-breaking prior
hergm(d ~ edges + triangle_ijk)

## End(Not run)
```

---

hergm-dirichlet *Hierarchical Exponential-Family Random Graph Models: Sample Truncated Dirichlet Process Prior*

Description

Sample truncated Dirichlet process prior.

Usage

```
hergm.dirichlet(n,
                number,
                alpha,
                eta_mean,
                eta_sd)
```

Arguments

n	number of nodes.
number	number of blocks.
alpha	scaling parameter.
eta_mean	mean of Gaussian base distribution of Dirichlet process prior.
eta_sd	standard deviation of Gaussian base distribution of Dirichlet process prior.

Value

Indicators of block memberships of nodes `indicator` and block parameters `eta`.

Examples

```
## Not run: # Load undirected network with 15 nodes
hergm.dirichlet(n=100, number=100, alpha=1, eta_mean=-1, eta_sd=1)

## End(Not run)
```

---

hergm-postprocess *Hierarchical Exponential-Family Random Graph Models: Postprocessing Samples*

Description

If called with the option `simulate = TRUE`, the function `hergm` returns a sample of networks, otherwise a raw MCMC sample from the posterior. The function `hergm.postprocess` postprocesses samples: if called with the `relabel = FALSE`, `hergm.postprocess` extracts information of interest, otherwise it solves, in addition, 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 the raw MCMC sample cannot be used to infer to block-dependent entities. The label-switching problem can be solved in a Bayesian decision-theoretic framework: by defining a loss function and minimizing the posterior expected loss. Calling `hergm.process` minimizes the posterior expected loss using a simple and convenient loss function. The required computations can be time-consuming when the number of blocks `k` is large.

Usage

```
hergm.postprocess(sample = NULL,
                  burnin = 0,
                  thinning = 1,
                  relabel = FALSE,
                  name = "",
                  ...)
```

Arguments

sample	MCMC sample generated by function hergm.
burnin	number of burn-in iterations; if computing is parallel, number of burn-in iterations per processor.
thinning	if thinning > 1, every thinning-th sample point is used while all others discarded.
relabel	if TRUE, relabel MCMC sample.
name	name of project; if output == TRUE, name of project is used to name output files.
...	additional arguments, to be passed to lower-level functions in the future.

Value

ergm_theta	parameters of ergm-terms.
alpha	scaling parameter of truncated Dirichlet process / stick-breaking prior of parameters of hergm-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.
hergm_theta	parameters of hergm-terms.
loss	if relabel == TRUE, local minimum of loss function.
p_k	probabilities of membership to blocks.
indicator	indicators of memberships of nodes.
p_i_k	probabilities of membership of nodes to blocks.
prediction	posterior predictions of statistics.

See Also

network, hergm.example, hergm, ergm.terms, hergm.terms

Examples

```
## Not run: # Load undirected network with 15 nodes (see ?example)
data(example)

# Generate MCMC sample of size 1,000
mcmc <- hergm(d ~ edges_i, parallel = 1, samplesize = 1000)
# Postprocess MCMC sample of size 1,000
processed_mcmc <- hergm.postprocess(sample = mcmc, burnin = 200)

# Generate MCMC sample of size 20 * 100 = 2,000
mcmc <- hergm(d ~ edges + triangle_ijk, parallel = 20, samplesize = 100)
# Postprocess MCMC sample of size 20 * 100 = 2,000
processed_mcmc <- hergm.postprocess(sample = mcmc, burnin = 20)

## End(Not run)
```

hergm-terms *Hierarchical Exponential-Family Random Graph Models: Terms*

Description

Hierarchical Exponential-Family Random Graph Models can be specified by calling the function `hergm(formula)`, where formula is a formula of the form `network ~ terms`.

By using suitable terms, it is possible to specify

- the p_1 model for directed networks of Holland and Leinhardt (1981) and its extension to undirected random graph models with Dirichlet process priors (see arcs_i, arcs_j, mutual_i, and edges_i). While the p_1 model for undirected and directed networks with parametric priors contains O(n) parameters (n = number of nodes) and therefore is not parsimonious, the non-parametric Dirichlet process prior encourages a small number of unique parameters and therefore represents an elegant alternative to parametric priors.

- the stochastic block model of Snijders and Nowicki (1997) and Nowicki and Snijders (2001) with natural parameterization (restricted between-block parameters) and Dirichlet process priors (see edges_ij).

- hierarchical exponential-family models with stick-breaking priors (see mutual_ij, twostar_ijk, triangle_ijk, ttriple_ijk, ctriple_ijk). Hierarchical exponential-family models replace the strong dependence of simple exponential-family models by weak dependence with an eye to solving the near-degeneracy problem of simple exponential-family model.

`hergm.terms` can be found here. Additional terms, e.g. covariate-dependent terms, can be found in `ergm.terms`.

Arguments

`edges_i(k) (undirected network)`
: adding the term `edges_i` to the model adds node-dependent edge terms to the model; the optional argument `k` is the maximum number of blocks (default: the number of nodes).

`arcs_i(k) (directed network)`
: adding the term `arcs_i` to the model adds node-dependent outdegree terms to the model; the optional argument `k` is the maximum number of blocks (default: the number of nodes).

`arcs_j(k) (directed network)`
: adding the term `arcs_j` to the model adds node-dependent indegree terms to the model; the optional argument `k` is the maximum number of blocks (default: the number of nodes).

`edges_ij(k) (undirected, directed network)`
: adding the term `edges_ij` to the model adds block-dependent edge terms to the model; the optional argument `k` is the maximum number of blocks (default: the number of nodes).

`mutual_i(k) (directed network)`
: adding the term `mutual_i` to the model adds additive, block-dependent mutual edge terms to the model; the optional argument `k` is the maximum number of blocks (default: the number of nodes).

`mutual_ij(k) (directed network)`
: adding the term `mutual_ij` to the model adds block-dependent mutual edge terms to the model; the optional argument `k` is the maximum number of blocks (default: the number of nodes).

`transedges (undirected network)`
: adding the term `transedges` to the model adds the number of transitive edges to the model.

`twostar_ijk(k) (undirected network)`
: adding the term `twostar_ijk` to the model adds block-dependent two-star terms to the model; the optional argument `k` is the maximum number of blocks (default: the number of nodes).

`triangle_ijk(k) (undirected, directed network)`
: adding the term `triangle_ijk` to the model adds block-dependent triangle terms to the model; the optional argument `k` is the maximum number of blocks (default: the number of nodes).

`ttriple_ijk(k) (directed network)`
: adding the term `ttriple_ijk` to the model adds block-dependent transitive triple terms to the model; the optional argument `k` is the maximum number of blocks (default: the number of nodes).

`ctriple_ijk(k) (directed network)`
: adding the term `ctriple_ijk` to the model adds block-dependent cyclic triple terms to the model; the optional argument `k` is the maximum number of blocks (default: the number of nodes).

References

Holland, P. W. and S. Leinhardt (1981). An exponential family of probability distributions for directed graphs. Journal of the American Statistical Association 76 (373), 33–65.

Nowicki, K. and T. A. B. Snijders (2001). Estimation and prediction for stochastic blockstructures. Journal of the American Statistical Association 96 (455), 1077–1087.

Snijders, T. A. B. and K. Nowicki (1997). Estimation and prediction for stochastic blockmodels for graphs with latent block structure. Journal of Classification 14, 75–100.

See Also

network, hergm, ergm.terms, hergm.postprocess

Examples

```
## Not run: # Load undirected network with 15 nodes (see ?example)
data(example)
# p_1 model for undirected network with Dirichlet process prior
hergm(d ~ edges_i)

# Load directed network with 18 nodes (see ?sampson)
data(sampson)
# p_1 model for directed network with Dirichlet process prior
hergm(samplike ~ arcs_i + arcs_j + mutual)
```

```
# Load undirected network with 15 nodes (see ?example)
data(example)
# Stochastic block model for undirected network
# with natural parameterization and Dirichlet process prior
hergm(d ~ edges_ij)

# Load directed network with 18 nodes (see ?sampson)
data(sampson)
# Stochastic block model for directed network
# with natural parameterization and Dirichlet process prior
hergm(samplike ~ edges_ij + mutual)

# Load undirected network with 15 nodes (see ?example)
data(example)
# Hierarchical exponential-family model with stick-breaking prior
hergm(d ~ edges + mutual + ttriple_ijk)

# Load directed network with 18 nodes (see ?sampson)
data(sampson)
# Hierarchical exponential-family model with stick-breaking prior
hergm(samplike ~ edges + mutual + ttriple_ijk)

## End(Not run)
```

Index