Package ‘networksis’

April 16, 2015

Version 2.1-3

Date 2015-04-07

Title Simulate Bipartite Graphs with Fixed Marginals Through Sequential Importance Sampling

Author Ryan Admiraal <`R.Admiraal@murdoch.edu.au`> and Mark S. Handcock <`handcock@ucla.edu`>

Maintainer Ryan Admiraal <`R.Admiraal@murdoch.edu.au`>

Depends network

Description Tools to simulate bipartite networks/graphs with the degrees of the nodes fixed and specified. 'networksis' is part of the 'statnet' suite of packages for network analysis.

License GPL (>= 2)

URL `http://statnet.org`

NeedsCompilation yes

Repository CRAN

Date/Publication 2015-04-16 08:44:42

R topics documented:

`networksis-package` *Simulate Bipartite Graphs with Fixed Marginals Through Sequential Importance Sampling*

Description

The `networksis` package is a collection of functions to simulate bipartite graphs with fixed marginals. For a list of functions, type: `help(package="networksis")`

For a complete list of the functions, use `library(help="networksis")` or read the rest of the manual.

This package is compatible with the **statnet** suite of packages, a collection of functions to plot, fit, diagnose, and simulate from random graph models. When publishing results obtained using this package, the original authors are to be cited as:

Admiraal, Ryan and Mark S. Handcock (2008). *networksis: Simulate bipartite graphs with fixed marginals through sequential importance sampling*.
`statnet.org`.

Admiraal, Ryan and Mark S. Handcock (2008). networksis: A package to simulate bipartite graphs with fixed marginals through sequential importance sampling, *Journal of Statistical Software*, 24(8).

You should also cite the developers of the 'statnet' suite of packages:

Mark S. Handcock, David R. Hunter, Carter T. Butts, Steven M. Goodreau, and Martina Morris (2003) *statnet: Software tools for the Statistical Modeling of Network Data*.
`statnet.org`.

All programs derived from this package must cite it. For complete citation information, use `citation(package="networksis")`.

Details

Sequential importance sampling provides a means to simulate matrices with fixed marginals and do so independently (Chen *et al*., 2005). Importance weights corresponding to simulated matrices allow for the estimation of the number of matrices consistent with these marginals, and they can also be used to estimate the null distribution of statistics based on these matrices.

In social network analysis, networks are represented through sociomatrices, so sequential importance sampling can naturally be extended to simulating independent networks consistent with fixed degree distributions, and **networksis** provides a means to do this for bipartite networks. The package relies on the `network` package which allows networks to be represented in R.

For detailed information on how to download and install the software, go to the `networksis` website: `statnet.org`. A tutorial, support newsgroup, references and links to further resources are provided there.

Author(s)

Ryan Admiraal <`R.Admiraal@murdoch.edu.au`>,
Mark S. Handcock <`handcock@ucla.edu`>

Maintainer: Ryan Admiraal <`R.Admiraal@murdoch.edu.au`>

References

Admiraal, Ryan and Mark S. Handcock (2008). **networksis**: Simulate bipartite graphs with fixed marginals through sequential importance sampling. Statnet Project, Seattle, WA. Version 2.1-3, `statnet.org`.

Admiraal, Ryan and Mark S. Handcock (2008). networksis: A package to simulate bipartite graphs with fixed marginals through sequential importance sampling, *Journal of Statistical Software*, 24(8).

Chen, Yuguo, Persi Diaconis, Susan P. Holmes, and Jun S. Liu (2005). Sequential Monte Carlo methods for statistical analysis of tables, *Journal of the American Statistical Association*, 100, 109-120.

`finch` *Co-location of Darwin's finches as a bipartite graph*

Description

Data on co-location of finches noted during Charles Darwin's visit to the Galapagos Islands.

Usage

```
data(finch)
```

Details

Charles Darwin compiled these data for thirteen finch species on a visit to the Galapagos Islands. For each finch type, he recorded on which of seventeen islands that finch could be found. Sanderson (2000) argues that, in examining island biogeography, it is important to condition on the number of islands and species in order to sample from the appropriate null space, so graphs sampled from the null distribution of the observed graph should have the same marginals. Chen *et al.* (2005) report the number of graphs matching the marginal constraints of Darwin's finch data to be 67,149,106,137,567,626.

References

Chen, Yuguo, Persi Diaconis, Susan P. Holmes, and Jun S. Liu (2005). Sequential Monte Carlo methods for statistical analysis of tables. *Journal of the American Statistical Association*, 100, 109-120.

Sanderson, James G. (2000). Testing Ecological Patterns, *American Scientist*, 88, 332-339.

Examples

```
data(finch)

# Plot the network
plot(finch)

# Network summary
summary(finch)
```

simulate.sisnetwork *Simulate a bipartite network using sequential importance sampling*

Description

The method `simulate.sisnetwork` simulates graphs with the same marginals as the passed network or as the rows and columns specified in a `sisnetwork` object through sequential importance sampling. That is, the degrees of the nodes are fixed and specified.

Usage

```
## S3 method for class 'sisnetwork'
simulate(object, nsim = 1, seed = NULL, save.networks = FALSE, ...)
```

Arguments

`object`	Either a `network` object or `sisnetwork` object. If a `sisnetwork` object, this should be a list with components `row` and `col` to specify the row and column degrees. These are the degrees of the type 1 and type 2 nodes, respectively.
`nsim`	Number of networks to be randomly drawn from the set of all networks.
`seed`	Seed for random number generator.
`save.networks`	If this is `TRUE`, the sampled networks are returned. Otherwise only the last network is returned.
`...`	Further arguments passed to or used by methods.

Details

A sample of networks is randomly drawn from the space of networks with the same degrees for each node.

Value

`simulate.sisnetwork` returns an object of class `network.series`, that is a list consisting of the following elements:

`networks`	The vector of simulated networks.
`log.prob`	The vector of the logarithm of the probability of being sampled.
`log.graphspace.size`	The logarithm of the mean estimate of the number of graphs in the graph space.
`log.graphspace.SE`	The logarithm of the standard error of the mean estimate of the number of graphs in the graph space.
`log.graphspace.size.lne`	The logarithm of the lognormal-based estimate of the number of graphs in the graph space.

`log.graphspace.SE.lne`
: The logarithm of the standard error of the lognormal-based estimate of the number of graphs in the graph space.

See Also

network

Examples

```
bipartite.graph <- matrix(c(1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0), nrow = 3, byrow = TRUE)
example.net <- network(bipartite.graph)

# Specify the set to which each node belongs
example.net %v% "set" <- c(rep(1, 3),rep(2, 4))

# Simulate 100 graphs with the same marginals as 'example.net'
sim <- simulate.sisnetwork(example.net, nsim = 100)

# Estimated graph space size and SE
exp(sim$log.graphspace.size)
exp(sim$log.graphspace.SE)

# Darwin's finches example
data(finch)

sim <- simulate.sisnetwork(finch, nsim = 100, save.networks = TRUE)

# Calculate importance weights from the graph probabilities
importance.weights <- 1 / exp(sim$log.prob)
hist(importance.weights, breaks = 25, xlab = "Inverse Graph Probability", main="")

# Calculate Sanderson's \bar{S}^2
s.bar.squared.vec <- rep(0, 100)

for(i in 1 : 100)
{
   # Extract simulated bipartite graphs
   new.graph <- as.matrix.network(sim$networks[[i]])

   # Calculate custom graph statistic
   s.bar.squared.vec[i] <- (sum((new.graph %*% t(new.graph)) ^ 2) -
   sum(diag((new.graph %*% t(new.graph)) ^ 2))) / (13 * 12)
}
```

Index