Package ‘ergm.count’

May 15, 2019

Version 3.4.0

Date 2019-05-15

Title Fit, Simulate and Diagnose Exponential-Family Models for Networks with Count Edges

Depends ergm (>= 3.10.1), network (>= 1.15)

Imports statnet.common (>= 4.2.0)

LinkingTo ergm

Description
A set of extensions for the 'ergm' package to fit weighted networks whose edge weights are counts.

License GPL-3 + file LICENSE

URL `http://statnet.org`

BugReports `https://github.com/statnet/ergm.count/issues`

NeedsCompilation yes

Author Pavel N. Krivitsky [aut, cre] (<https://orcid.org/0000-0002-9101-3362>),
Mark S. Handcock [ctb],
David R. Hunter [ctb]

Maintainer Pavel N. Krivitsky <`pavel@uow.edu.au`>

Repository CRAN

Date/Publication 2019-05-15 07:42:59 UTC

R topics documented:

ergm.count-package *Fit, Simulate and Diagnose Exponential-Family Models for Networks with Count Edges*

Description

ergm.count is a set of extensions to R package ergm to fit and simulate from exponential-family random graph models for networks whose edge weights are counts. For a list of functions type `help(package='ergm')` and `help(package='ergm.count')`

Mainly, it implements Poisson, binomial, geometric, and discrete uniform dyadwise reference measures for valued ERGMs (documented here), and provides some count-specific change statistics (documented here).

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

When publishing results obtained using this package, please cite the original authors as described in `citation(package="ergm.count")`.

All programs derived from this package must cite it.

Details

This package contains functions specific to using ergm to model networks whose dyad values are counts. Examples include counts of conversations, messages, and other interactions.

In particular, this package implements the Poisson, geometric, binomial, and discrete uniform reference measures (documented in ergm-references) for use by ergm and simulate.ergm to fit models from this family, as well as statistics specific to modeling counts, such as the CMP for the Conway-Maxwell-Poisson Distribution.

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

Author(s)

Pavel N. Krivitsky <pavel@uow.edu.au>

References

Handcock MS, Hunter DR, Butts CT, Goodreau SG, Krivitsky PN and Morris M (2012). _Fit, Simulate and Diagnose Exponential-Family Models for Networks_. Version 3.1. Project home page at <URL: http://www.statnet.org>, <URL: CRAN.R-project.org/package=ergm>.

Krivitsky PN (2012). Exponential-Family Random Graph Models for Valued Networks. *Electronic Journal of Statistics*, 2012, 6, 1100-1128. doi:10.1214/12-EJS696

See Also

ergm-terms, ergm-references

ergm-references *Reference Measures for Exponential-Family Random Graph Models for Counts*

Description

This page describes the possible reference measures (baseline distributions) for modeling count data found in the `ergm.count` package.

Each of these is specified on the RHS of a one-sided formula passed as the `reference` argument to `ergm`. See the `ergm` documentation for a complete description of how reference measures are specified.

Known issues

Parameter space constraints:

Poisson- and geometric-reference ERGMs have an unbouded sample space. This means that the parameter space may be constrained in complex ways that depend on the terms used in the model. At this time `ergm` has no way to detect when a parameter configuration had strayed outside of the parameter space, but it may be noticeable on a runtime trace plot (activated via `MCMC.runtime.traceplot` control parameter), when the simulated values keep climbing upwards. (See Krivitsky (2012) for a further discussion.)

A possible remedy if this appears to occur is to try lowering the control parameter `MCMLE.steplength`.

`MCMLE.trustregion`:

Because Monte Carlo MLE's approximation to the likelihood becomes less accurate as the estimate moves away from the one used for the sample, `ergm` limits how far the optimization can move the estimate for every iteration: the log-likelihood may not change by more than `MCMLE.trustregion` control parameter, which defaults to 20. This is an adequate value for binary ERGMs, but because each dyad in a valued ERGM contains more information, this number may be too small, resulting in unnecessarily many iterations needed to find the MLE.

Automatically setting `MCMLE.trustregion` is work in progress, but, in the meantime, you may want to set it to a high number (e.g., 1000).

Possible reference measures to represent baseline distributions

Reference measures currently available are:

`Poisson` *Poisson-reference ERGM:* Specifies each dyad's baseline distribution to be Poisson with mean 1: $h(y) = \prod_{i,j} 1/y_{i,j}!$, with the support of $y_{i,j}$ being natural numbers (and 0). Using `valued ERGM terms` that are "generalized" from their binary counterparts, with form `"sum"` (see previous link for the list) produces Poisson regression. Using `CMP` induces a Conway-Maxwell-Poisson distribution that is Poisson when its coefficient is 0 and geometric when its coefficient is 1.

Three proposal functions are currently implemented, two of them designed to improve mixing for sparse networks. They can can be selected via the `MCMC.prop.weights=` control parameter. The sparse proposals work by proposing a jump to 0. Both of them take an optional proposal argument `p0` (i.e., `MCMC.prop.args=list(p0=...)`) specifying the probability of such a jump. However, the way in which they implement it are different:

`"random"` Select a dyad (i, j)at random, and draw the proposal $y^\star_{i,j} \sim \text{Poisson}_{\neq y_{i,j}}(y_{i,j} + 0.5)$ (a Poisson distribution with mean slightly higher than the current value and conditional on not proposing the current value).

`"0inflated"` As `"random"` but, with probability `p0`, propose a jump to 0 instead of a Poisson jump (if not already at 0). If `p0` is not given, defaults to the "surplus" of 0s in the observed network, relative to Poisson.

`"TNT"` **(the default)** As `"0inflated"` but instead of selecting a dyad at random, select a tie with probability `p0`, and a random dyad otherwise, as with the binary TNT. Currently, `p0` defaults to 0.2.

`Geometric` *Geometric-reference ERGM:* Specifies each dyad's baseline distribution to be uniform on the natural numbers (and 0): $h(y) = 1$. In itself, this "distribution" is improper, but in the presence of `sum`, a geometric distribution is induced. Using `CMP` (in addition to `sum`) induces a Conway-Maxwell-Poisson distribution that is geometric when its coefficient is 0 and Poisson when its coefficient is -1.

`Binomial(trials)` *Binomial-reference ERGM:* Specifies each dyad's baseline distribution to be binomial with `trials` trials and success probability of 0.5: $h(y) = \prod_{i,j} \binom{\texttt{trials}}{y_{i,j}}$. Using `valued ERGM terms` that are "generalized" from their binary counterparts, with form `"sum"` (see previous link for the list) produces logistic regression.

References

Krivitsky PN (2012). Exponential-Family Random Graph Models for Valued Networks. *Electronic Journal of Statistics*, 2012, 6, 1100-1128. doi:10.1214/12-EJS696

Shmueli G, Minka TP, Kadane JB, Borle S, and Boatwright P (2005). A Useful Distribution for Fitting Discrete Data: Revival of the Conway–Maxwell–Poisson Distribution. *Journal of the Royal Statistical Society: Series C*, 54(1): 127-142.

See Also

ergm, network, %v%, %n%, sna, summary.ergm, print.ergm

`ergm-terms` *Terms used in Exponential Family Random Graph Models Specific to Counts*

Description

This page describes the possible terms (and hence network statistics) included in the `ergm.count` package.

See the `ergm-terms` documentation in the `ergm` package for a complete description of what ERGM terms are and how they are used.

Terms to represent network statistics included in the `ergm.count` pacakge

All terms listed are valued.

`CMP` *Conway-Maxwell-Poisson Distribution:* This term adds one statistic to the model, of the form $\sum_{i,j} \log(y_{i,j}!)$. This turns a Poisson- or a geometric-reference ERGM into a Conway-Maxwell-Poisson-reference ERGM, allowing it to represent a broad range of disperson values. In particular, combined with a Poisson-reference ERGM, a negative coefficient on this term induces underdispersion and a positive coefficient induces overdispersion. (This behavior is different from 3.1.1, when the negation of this value was used.)

Note that its current implementation may not perform well if the data are overdispersed relative to geometric.

References

- Handcock M. S., Hunter D. R., Butts C. T., Goodreau S. G., Krivitsky P. N. and Morris M. (2012). _Fit, Simulate and Diagnose Exponential-Family Models for Networks_. Version 3.1. Project home page at <URL: http://www.statnet.org>, <URL: CRAN.R-project.org/package=ergm>.
- Krivitsky P. N. (2012). Exponential-Family Random Graph Models for Valued Networks. *Electronic Journal of Statistics*, 2012, 6, 1100-1128. `doi:10.1214/12-EJS696`
- Shmueli G., Minka T. P., Kadane J. B., Borle S., and Boatwright P. (2005). A Useful Distribution for Fitting Discrete Data: Revival of the Conway–Maxwell–Poisson Distribution. *Journal of the Royal Statistical Society: Series C*, 54(1): 127-142.

See Also

`ergm-terms` (from the `ergm` package), `ergm`, `network`, `%v%`, `%n%`

`zach` *Karate club social network of Zachary (1977)*

Description

Zachary (1977) reported observations of social relations in a university karate club, with membership that varied between 50 and 100, of whom 34 individuals: 32 ordinary club members and officers, the club president ("John A."), and the part-time instructor ("Mr. Hi"); consistently interacted outside of the club. Over the course of the study, the club divided into two factions, and, ultimately, split into two clubs, one led by Hi and the other by John and the original club's officers. The split was driven by a disagreement over whether Hi could unilaterally change the level of compensation for his services.

Zachary identifies the faction with which each of the 34 actors was aligned and how strongly and reports, for each pair of actors, the count of social contexts in which they interacted. The 8 contexts recorded were

- academic classes at the university;
- Hi's private karate studio in his night classes;

- Hi's private karate studio where he taught on weekends;
- student-teaching at Hi's studio;
- the university rathskeller (bar) located near the karate club;
- a bar located near the university campus;
- open karate tournaments in the area; and
- intercollegiate karate tournaments.

The highest number of contexts of interaction for a pair of individuals that was observed was 7.

Usage

```
data(zach)
```

Format

The data are represented as a `network` object, with an edge attribute `contexts`, giving the number of contexts of interaction for that pair of actors. In addition, the following vertex attributes are provided:

`club`**:** the club in which the actor ended up;

`faction`**:** faction alignment of the actor as recorded by Zachary

`faction.id` faction alignment coded numerically, as -2 (strongly Mr. Hi's), -1 (weakly Mr. Hi's), 0 (neutral), $+1$ (weakly John's), and $+2$ (strongly John's);

`role` role of the actor in the network (Instructor, Member, or President)

Source

Zachary, WW (1977). An Information Flow Model for Conflict and Fission in Small Groups. *Journal of Anthropological Research*, 33(4), 452-473.

Sociomatrix in machine-readable format was retrieved from `http://vlado.fmf.uni-lj.si/pub/networks/data/ucinet/ucidata.htm`.

References

Zachary, WW (1977). An Information Flow Model for Conflict and Fission in Small Groups. *Journal of Anthropological Research*, 33(4), 452-473.

Examples

```
data(zach)

oldpal <- palette()
palette(gray((1:8)/8))
plot(zach, vertex.col="role", displaylabels=TRUE, edge.col="contexts")
palette(oldpal)


# Fit a binomial-reference ERGM.
```

```
zach.fit1 <- ergm(zach~nonzero+sum+nodefactor("role",base=2)+absdiffcat("faction.id"),
                  response="contexts", reference=~Binomial(8),
                  control=control.ergm(MCMLE.trustregion=1000))

mcmc.diagnostics(zach.fit1)

summary(zach.fit1)

## Not run:
# This is much slower.
zach.fit2 <- ergm(zach~nonzero+sum+nodefactor("role",base=2)+transitiveties,
                  response="contexts", reference=~Binomial(8),
                  control=control.ergm(MCMLE.trustregion=1000),
                  eval.loglik=FALSE)

mcmc.diagnostics(zach.fit2)

summary(zach.fit2)

## End(Not run)
```

Index