Package 'ergm.ego'

November 20, 2020

Version 0.6.1

Date 2020-11-19

Title Fit, Simulate and Diagnose Exponential-Family Random Graph
Models to Egocentrically Sampled Network Data

Depends ergm (>= 3.11.0), network (>= 1.15)

Imports statnet.common (>= 4.4.0), coda (>= 0.19.2), RColorBrewer (>=
1.1.2), purrr (>= 0.3.2), tibble (>= 2.1.1), stats, methods

Suggests testthat (>= 2.1.1), covr (>= 3.2.1)

Description Utilities for managing egocentrically sampled network data and a wrap-
per around the 'ergm' package to facilitate ERGM inference and simulation from such data.

License GPL-3 + file LICENSE

URL <https://statnet.org>

BugReports <https://github.com/statnet/ergm.ego/issues>

RoxygenNote 7.1.1

Encoding UTF-8

NeedsCompilation yes

Author Pavel N. Krivitsky [aut, cre] (<https://orcid.org/0000-0002-9101-3362>),
Steven M. Goodreau [ctb],
Martina Morris [ctb],
Kirk Li [ctb],
Emily N. Beylerian [ctb],
Michał Bojanowski [ctb] (<https://orcid.org/0000-0001-7503-852X>),
Chad Klumb [ctb]

Maintainer Pavel N. Krivitsky <pavel@statnet.org>

R **topics documented:**

---

as.egodata.network   *Construct an Egocentric View of a Network*

Description

Given a [network](#) object, construct an [egodata](#) object representing a census of all the actors in the network. Used mainly for testing.

Usage

```
## S3 method for class 'network'
as.egodata(
  object,
  special.cols = c("na", "vertex.names"),
  ...,
  egoIDcol = "vertex.names"
)
```

Arguments

object	A [network](#) object.
special.cols	Vertex attributes that should not be copied to the egos and alters tables. Defaults to attributes special to the [network](#) objects.
...	Additional arguments, currently unused.
egoIDcol	The name of the vertex attribute containing unique ego IDs. Defaults to "vertex.names".

Value

An [egodata](#) object.

Author(s)

Pavel N. Krivitsky

See Also

[as.network.egodata](#), which performs the inverse operation (though drops the ties).

Examples

```
# See example(ergm.ego) and example(as.network.egodata).
```

---

as.network.egodata	*Construct an Empty "Template" Network Consistent with an Egocentric Sample*

Description

Taking a [egodata](#) object, constructs a [network](#) object with no edges whose vertices have the attributes of the egos in the dataset, replicating the egos as needed, and taking into accounts their sampling weights.

Usage

```
## S3 method for class 'egodata'
as.network(x, N, scaling = c("round", "sample"), ...)
```

Arguments

x	A [egodata](#) object.
N	The target number of vertices the output network should have.
scaling	If [egodata](#) contains weights or N is not a multiple of number of egos in the sample, it may not be possible, for a finite N to represent each ego exactly according to its relative weight, and scaling controls how the fractional egos are allocated:
	"round" (the default) Rather than treating N as a hard setting, calculate $Nw_i/w.$ for each ego i and round it to the nearest integer. Then, the N actually used will be the sum of these rounded freqencies.
	"sample" Resample in proportion to w_i.
...	Additional arguments, currently unused.

Value

A [network](#) object. It has an additional vertex attribute .ego.ind giving the *row index* (not ego ID) of the ego from which a particular node was generated.

Author(s)

Pavel N. Krivitsky

See Also

[as.egodata.network](#), which performs the inverse operation.

Examples

```
data(faux.mesa.high)
summary(faux.mesa.high, print.adj = FALSE)

fmh.ego <- as.egodata(faux.mesa.high)

# Same actor attributes
fmh.template <- as.network(fmh.ego, N=network.size(faux.mesa.high))
summary(fmh.template, print.adj = FALSE)

# Twice the actors, same distribution
fmh2.template <- as.network(fmh.ego, N=2*network.size(faux.mesa.high))
summary(fmh2.template, print.adj = FALSE)
```

---

control.ergm.ego   *Control parameters for* [ergm.ego](#).

Description

Constructs and checks the list of control parameters for estimation by [ergm.ego](#).

Usage

```
control.ergm.ego(
  ppopsize = c("auto", "samp", "pop"),
  ppopsize.mul = 1,
  ppop.wt = c("round", "sample"),
  stats.wt = c("data", "ppop"),
  stats.est = c("asymptotic", "bootstrap", "jackknife", "naive"),
  boot.R = 10000,
  ergm.control = control.ergm(),
  ...
)
```

Arguments

ppopsize, ppopsize.mul

Parameters to determine the size $|N'|$ of the pseudopopulation network. ppopsize can be

"auto" If the popsize ($|N|$) argument is specified and is different from 1, as if "pop"; otherwise, as "samp".

"samp" set $|N'|$ based on the sample size: $|N'| = |S| \times$ popsize.mul

"pop" set $|N'|$ based on the population size: $|N'| = |N| \times$ popsize.mul

a number set $|N'|$ directly (popsize.mul ignored)

a network **object** use the specified network as the pseudo-population network directly; use at your own risk

a data frame use the specified data frame as the pseudo-population; use at your own risk

The default is to use the same pseudopopulation size as the sample size, but, particularly if there are sampling weights in the data, it should be bigger.

Note that depending on ppop.wt, this may only be an approximate target specification, with the actual constructed pseudopopulation network being slightly bigger or smaller.

ppop.wt

Because each ego must be represented in the pseuodopopulation network an integral number of times, if the sample is weighted (or the target $|N'|$ calculated from ppopsize and ppopsize.mul is not a multiple of the sample size), it may not be possible, for a finite $|N'|$ to represent each ego exactly according to its relative weight, and ppop.wt controls how the fractional egos are allocated:

"round" (default) Rather than treating ppopsize as a hard setting, calculate $|N'|w_i/w.$ for each ego i and round it to the nearest integer. Then, the $|N'|$ actually used will be the sum of these rounded freqencies.

"sample" Resample in proportion to w_i.

stats.wt

Weight assigned to each ego's contribution to the ERGM's sufficient statistic:

"data" (default) Use weights $|N'|w_i/w.$ for each ego i as in the data.

"ppop" Use weights ultimately used in the pseudopopulation network.

stats.est, boot.R

Method to be used to estimate the ERGM's sufficient statistics and their variance:

"asymptotic" Delta method, as derived by Krivitsky and Morris (2015), assuming the ego weights are sampled alongside the egos.

(default) Delta method, as derived by Krivitsky and Morris (2015), assuming the ego weights are sampled alongside the egos.

"bootstrap" Nonparametric bootstrap with bias correction, resampling egos, using R replications.

"jackknife" Jackknife with bias correction.

"naive" "Naive" estimator, assuming that weights are fixed.

ergm.control   Control parameters for the ergm call to fit the model, constructed by control.ergm.

...   Not used at this time.

Value

A list with arguments as components.

Author(s)

Pavel N. Krivitsky

References

Pavel N. Krivitsky and Martina Morris. Inference for Social Network Models from Egocentrically-Sampled Data, with Application to Understanding Persistent Racial Disparities in HIV Prevalence in the US. Thechnical Report. National Institute for Applied Statistics Research Australia, University of Wollongong, 2015(05-15). http://niasra.uow.edu.au/publications/UOW190187.html

See Also

control.ergm

---

control.simulate.ergm.ego

Control parameters for `simulate.ergm.ego`.

Description

Constructs and checks the list of control parameters for simulation by `simulate.ergm.ego`.

Usage

```
control.simulate.ergm.ego(
  ppop.wt = c("round", "sample"),
  SAN.control = control.san(),
  simulate.control = control.simulate(),
  ...
)
```

Arguments

ppop.wt   Because each ego must be represented in the pseuodopopulation network an integral number of times, if the sample is weighted (or the target $|N'|$ calculated from ppopsize and ppopsize.mul is not a multiple of the sample size), it may not be possible, for a finite $|N'|$ to represent each ego exactly according to its relative weight, and ppop.wt controls how the fractional egos are allocated:

"round" (default) Rather than treating ppopsize as a hard setting, calculate $|N'|w_i/w.$ for each ego i and round it to the nearest integer. Then, the $|N'|$ actually used will be the sum of these rounded freqencies.
"sample" Resample in proportion to w_i.

SAN.control	A list of control parameters for [san](#) constructed by [control.ergm](#), called to construct a pseudopopulation network consistent with the data.
simulate.control	
	A list of control parameters for [simulate.formula](#) constructed by [control.simulate](#), called to simulate from the model fit.
...	Not used at this time.

Value

A list with arguments as components.

Author(s)

Pavel N. Krivitsky

See Also

control.simulate, control.san

---

degreedist.egodata   *Plotting the degree distribution of an egocentric dataset*

Description

A [degreedist()](#) method for [egodata](#) objects: plot a histogram of the degree distribution of actors in the egocentric dataset, optionally broken down by group and/or compared with a Bernoulli graph.

Usage

```
## S3 method for class 'egodata'
degreedist(
  object,
  freq = FALSE,
  prob = !freq,
  by = NULL,
  brgmod = FALSE,
  main = NULL,
  plot = TRUE,
  weight = TRUE,
  ...
)
```

Arguments

`object`	A [egodata](#) object.
`freq, prob`	Whether to plot the raw frequencies or the conditional proportions of the degree values. Defaults to the latter.
`by`	A character vector giving the name of a vertex attribute; if given, plots the frequences broken down by that attribute.
`brgmod`	Plot the range of predicted frequencies/probabilities according to a Bernoulli graph having the same expected density as the observed.
`main`	Main title of the plot.
`plot`	Whether to plot the histogram; if `FALSE`, graphical parameters and bgrmod have no effect.
`weight`	Whether sampling weights should be incorporated into the calculation (`TRUE`, the default) or ignored (`FALSE`).
`...`	Additional arguments to [simulate.ergm.ego()](#).

Value

Returns either a vector of degree frequencies/proportions if by=NULL or a matrix with a row for each category if not. If plot==TRUE returns invisibly.

See Also

[degreedist](#), [summary](#)

Examples

```
data(faux.mesa.high)
fmh.ego <- as.egodata(faux.mesa.high)

degreedist(fmh.ego,by="Grade",brgmod=TRUE)
# Compare:
degreedist(faux.mesa.high)
```

---

egodata   *Convert to or Construct* [egodata](#) *Objects*

Description

[as.egodata](#) is a generic function to construct [egodata](#) objects from a variety of sources. [egodata](#) function is the standard constructor, taking two data frames. For other methods for this class, see the Miscellaneous Methods section.

Usage

```
egodata(egos, alters, egoWt = 1, ..., egoIDcol = "egoID")

as.egodata(object, ..., egoIDcol = "egoID")

## S3 method for class 'data.frame'
as.egodata(
  object,
  alters,
  egoWt = 1,
  ...,
  egoIDcol = "egoID",
  alterIcol = "alterInd",
  alterIDcol = "alterID"
)
```

Arguments

alters   The [data.frame](#) containing at least the column named in egoIDcol, whose
                values do not have to be unique, and not every ego has to be represented. Other
                columns contain information about the alters.

                For the [data.frame](#) method in which the object argument also contains alter
                information in 'wide' format, a [list](#) with the following information:

                columns A character, integer, or logical vector identifying which columns con-
                      tain the alters' information.

                count The name of the column containing the number of alters nominated by
                      that ego.

                name.sep A one-character string or an empty string (defaulting to ".") speci-
                      fying the character, if any, used to separate alter attribute name from alter's
                      index within the ego. If an empty string (""), attribute name is assumed to
                      be made of letters, with any numbers being the alter index.

                egoIDcol, whose values do not have to be unique, and not every ego must be
                represented. Other columns contain information about the alters.

egoWt   A vector of the same length as number of rows in egos or object, containing
                the relative sampling weight of each ego.

...   Additional arguments; currently unused.

egoIDcol   Name of the column in the ego table containing the unique ego identifier.

object, egos   The object from which the egocentric data should be constructed. For the [data.frame](#)
                methods and [egodata](#) itself, a data frame containing at least the column named
                in egoIDcol, whose values must all be unique. Other columns contain informa-
                tion about the egos.

                For the [data.frame](#) method, it may also contain the information about the
                alters in a 'wide' format, in the form of additional columns with names like
                ATTRNAME1, ATTRNAME2, etc. for attribute ATTRNAME of alter 1, 2, etc., as well as
                a column containing the number of alters nominated by that ego.

alterIcol   Column name to use for the within-ego index of the alter.

alterIDcol   Column name to use for the unique ID of each alter, constructed by concatenat-
                    ing the ID of the ego that nominated them and their index within that ego.

Value

An [egodata](#) object. The object is a list containing the following elements:

egos   A data frame with one row for each ego, containing at least the column named
                    in egoIDcol, and other columns containing attributes of the egos.

alters   A data frame containing at least the column named in egoIDcol, and other
                    columns containing attributes of the alters.

egoWt   A vector of the same length as the number of egos, containing the relative sam-
                    pling weight of each ego.

egoIDcol   Name of the column in the ego table containing the unique ego identifier.

Miscellaneous Methods

The following "standard" methods have also been implemented for [egodata](#):

"dim.egodata" A vector with three elements containing the "dimensions" of the [egodata](#) object:
                    number of egos, number of columns in the egos table, and number of columns in the alters
                    table, inclsive of the ego identifier column. As a corollary, [nrow](#) returns the number of egos in
                    the dataset.

"dimnames.egodata" A list with three elements containing the "dimension names" of the [egodata](#)
                    object: ego IDs, column names of the egos table, and column names of the alters table, in-
                    clsive of the ego identifier column.

"sample.egodata" As [sample](#), but takes and returns a simulated [egodata](#) dataset by resampling
                    egos, adjusting ego weights as necessary, if weighted sampling was used.

"head.egodata" As [head](#), but returns the first n rows of egos, alters, and weights.

"na.omit.egodata" As [na.omit.data.frame](#), but takes and returns an [egodata](#) dataset, with egos
                    with NA in their rows or in their alters' rows. An optional argument relevant, defaulting to
                    all columns, can be used to select (by index or name) based on which columns an ego may be
                    dropped. (I.e., NAs in those not "relevant" are ignored.)

Author(s)

Pavel N. Krivitsky

See Also

[ergm.ego](#) for examples, [as.network.egodata](#), [as.egodata.network](#), [subset.egodata](#), [\[.egodata](#)

---

ergm.ego   *Inference for Exponential-Family Random Graph Models based on
                                   Egocentrically Sampled Data*

Description

A wrapper around the [ergm](#) to fit an ERGM to an [egodata](#) object.

Usage

```
ergm.ego(
  formula,
  popsize = 1,
  offset.coef = NULL,
  ...,
  control = control.ergm.ego(),
  na.action = na.fail,
  do.fit = TRUE
)
```

Arguments

formula	An [formula](#) object, of the form e ~ <model terms>, where e is a [egodata](#) object. See [ergm](#) for details and examples.
	For a list of currently implemented egocentric terms for the RHS, see [ergm.ego-terms](#).
popsize	The size $|N|$ of the finite population network from which the egocentric sample was taken; only affects the shift in the coefficients of the terms modeling the overall propensity to have ties. Setting it to 1 (the default) essentially uses the $-\log|N'|$ offset on the edges term.
offset.coef	A vector of coefficients for the offset terms.
...	Additional arguments passed to [ergm](#).
control	A [control.ergm.ego](#) control list.
na.action	How to handle missing actor attributes in egos or alters, when the terms need them.
do.fit	Whether to actually call [ergm](#)

Value

An object of class ergm.ego inheriting from [ergm](#), with the following additional or overridden elements:

"v"	Variance-covariance matrix of the estimate of the sufficient statistics
"m"	Estimate of the sufficient statistics
"egodata"	The egodata object passed
"popsize"	Population network size used

"ppopsize"	Pseudopopulation size used, see control.ergm.ego
"coef"	The coefficients, along with the network size adjustment netsize.adj coefficient.
"covar"	Pseudo-MLE estimate of the variance-covariance matrix of the parameter estimates under repeated egocentric sampling
"ergm.covar"	The variance-covariance matrix of parameter estimates under the ERGM superpopulation process (without incorporating sampling).
"DtDe"	Estimated Jacobian of the expectation of the sufficient statistics with respect to the model parameters

Author(s)

Pavel N. Krivitsky

References

Pavel N. Krivitsky and Martina Morris. Inference for Social Network Models from Egocentrically-Sampled Data, with Application to Understanding Persistent Racial Disparities in HIV Prevalence in the US. Thechnical Report. National Institute for Applied Statistics Research Australia, University of Wollongong, 2015(05-15). http://niasra.uow.edu.au/publications/UOW190187.html

Examples

```
data(faux.mesa.high)
fmh.ego <- as.egodata(faux.mesa.high)

head(fmh.ego)

egofit <- ergm.ego(fmh.ego~edges+degree(0:3)+nodefactor("Race")+nodematch("Race")
                        +nodefactor("Sex")+nodematch("Sex")+absdiff("Grade"),
                         popsize=network.size(faux.mesa.high))

# Run convergence diagnostics
mcmc.diagnostics(egofit)

# Estimates and standard errors
summary(egofit)

# Note that we recover the ergm() parameters
## Not run:
coef(ergm(faux.mesa.high~edges+degree(0:3)+nodefactor("Race")+nodematch("Race")
                        +nodefactor("Sex")+nodematch("Sex")+absdiff("Grade"),
           eval.loglik=FALSE))

## End(Not run)
rbind(c(0, -0.8407, 2.3393, 1.4686, 0.6323, 0.5287, -1.3603, -1.0454,
        -2.4998, -0.7207, 0.833, -0.1823, 0.6357, -1.3513),
      coef(egofit))
```

---

ergm.ego-terms   *ergm Terms Implemented for* egodata

Description

This page describes the ergm terms (and hence network statistics) for which inference based on egocentrically sampled data is implemented in ergm.ego package. Other packages may add their own terms.

Details

The current recommendation for any package implementing additional egocentric calculator terms is to create a help file with a name or alias ergm.egodata-terms, so that help("ergm.egodata-terms") will list egocentric ERGM terms available from all loaded packages.

Currently implemented egocentric statistics

For each of these, please see their respective package's ergm-terms help for meaning and parameters. The simplest way to do this is usually via ? TERM.

Special-purpose terms: netsize.adj A special-purpose term equivalent to edges, to house the network-size adjustment offset. This term is added to the model automatically and should not be used in the model formula directly.

ergm:
- offset
- edges
- nodecov
- nodefactor
- nodematch
- nodemix
- absdiff
- degree
- degrange
- concurrent
- concurrentties
- degree1.5
- mm
- meandeg*

tergm:
- mean.age*

Starred terms are *nonscaling*, in that while they can be evaluated, some inferential results and standard error calculation methods may not be applicable.

See Also

ergm-terms

gof.ergm.ego   *Conduct Goodness-of-Fit Diagnostics on a Exponential Family Random Graph Model fit to Egocentrically Sampled Data*

Description

`gof.ergm.ego` implements the `gof` method for `ergm.ego` fit objects.

Usage

```
## S3 method for class 'ergm.ego'
gof(
  object,
  ...,
  GOF = c("model", "degree"),
  control = control.gof.ergm(),
  verbose = FALSE
)
```

Arguments

object	An `ergm.ego` fit.
...	Additional arguments, currently unused.
GOF	A string specifying the statistics whose goodness of fit is to be evaluated. Currently, only "degree" and "model" are implemented; see `gof` documentation for details.
control	A list to control parameters, constructed using `control.gof.formula` or `control.gof.ergm` (which have different defaults).
verbose	Provide verbose information on the progress of the simulation.

Value

An object of class `gofobject`.

Author(s)

Pavel N. Krivitsky

See Also

For examples, see `ergm.ego`.

Examples

```
data(faux.mesa.high)
fmh.ego <- as.egodata(faux.mesa.high)

head(fmh.ego)

egofit <- ergm.ego(fmh.ego~edges+degree(0:3)+nodefactor("Race")+nodematch("Race")
                        +nodefactor("Sex")+nodematch("Sex")+absdiff("Grade"),
                        popsize=network.size(faux.mesa.high))

# Check whether the model "converged":
(modelgof <- gof(egofit, GOF="model"))
plot(modelgof)

# Check whether the model reconstructs the degree distribution:
(deggof <- gof(egofit, GOF="degree"))
plot(deggof)
```

---

mixingmatrix.egodata   *Summarizing the mixing among groups in an egocentric dataset*

Description

A mixingmatrix method for egodata objects, to return counts of how often a ego of each group nominates an alter of each group.

Usage

```
## S3 method for class 'egodata'
mixingmatrix(object, attrname, rowprob = FALSE, weight = TRUE, ...)
```

Arguments

object	A egodata object.
attrname	A character vector containing the name of the network attribute whose mixing matrix is wanted.
rowprob	Whether the counts should be normalized by row sums. That is, whether they should be proportions conditional on the ego's group.
weight	Whether sampling weights should be incorporated into the calculation (TRUE, the default) or ignored (FALSE).
...	Additional arguments, currently unused.

Value

A matrix with a row and a column for each level of attrname.

Note that, unlike mixingmatrix, what is counted are *nominations*, not ties. This means that under an egocentric census, the diagonal of mixingmatrix.egodata will be twice that returned by mixingmatrix for the original undirected network.

See Also

mixingmatrix, nodemix, summary method for egocentric data

Examples

```
data(faux.mesa.high)
fmh.ego <- as.egodata(faux.mesa.high)

(mm <- mixingmatrix(faux.mesa.high,"Grade"))
(mm.ego <- mixingmatrix(fmh.ego,"Grade"))

stopifnot(isTRUE(all.equal({tmp<-unclass(mm$matrix); diag(tmp) <- diag(tmp)*2;
tmp}, mm.ego, check.attributes=FALSE)))
```

---

nodal_attributes-API   *Helper functions for specifying nodal attribute levels*

Description

These functions are meant to be used in EgoStat and other implementations to provide the user with a way to extract nodal attributes and select their levels in standardized and flexible ways. They are intended to parallel ergm::nodal_attributes-API of ergm package.

ergm.ego_get_vattr extracts and processes the specified nodal attribute vector. It is strongly recommended that check.ErgmTerm()'s corresponding vartype="function,formula,character" (using the ERGM_VATTR_SPEC constant).

ergm.ego_attr_levels filters the levels of the attribute. It is strongly recommended that check.ErgmTerm()'s corresponding vartype="function,formula,character,numeric,logical,AsIs,NULL" (using the ERGM_LEVELS_SPEC constant).

Usage

```
ergm.ego_get_vattr(
  object,
  df,
  accept = "character",
  multiple = if (accept == "character") "paste" else "stop",
  ...
)
```

```
## S3 method for class 'character'
ergm.ego_get_vattr(
  object,
  df,
  accept = "character",
  multiple = if (accept == "character") "paste" else "stop",
  ...
)

## S3 method for class '`function`'
ergm.ego_get_vattr(
  object,
  df,
  accept = "character",
  multiple = if (accept == "character") "paste" else "stop",
  ...
)

## S3 method for class 'formula'
ergm.ego_get_vattr(
  object,
  df,
  accept = "character",
  multiple = if (accept == "character") "paste" else "stop",
  ...
)

ergm.ego_attr_levels(object, attr, egodata, levels = sort(unique(attr)), ...)

## S3 method for class 'numeric'
ergm.ego_attr_levels(object, attr, egodata, levels = sort(unique(attr)), ...)

## S3 method for class 'logical'
ergm.ego_attr_levels(object, attr, egodata, levels = sort(unique(attr)), ...)

## S3 method for class 'AsIs'
ergm.ego_attr_levels(object, attr, egodata, levels = sort(unique(attr)), ...)

## S3 method for class 'character'
ergm.ego_attr_levels(object, attr, egodata, levels = sort(unique(attr)), ...)

## S3 method for class '`NULL`'
ergm.ego_attr_levels(object, attr, egodata, levels = sort(unique(attr)), ...)

## S3 method for class '`function`'
ergm.ego_attr_levels(object, attr, egodata, levels = sort(unique(attr)), ...)
```

```
## S3 method for class 'formula'
ergm.ego_attr_levels(object, attr, egodata, levels = sort(unique(attr)), ...)
```

Arguments

`object`	An argument specifying the nodal attribute to select or which levels to include.
`df`	Table of egos or of alters.
`accept`	A character vector listing permitted data types for the output. See the Details section for the specification.
`multiple`	Handling of multiple attributes or matrix or data frame output. See the Details section for the specification.
`...`	Additional argument to the functions of network or to the formula's environment.
`attr`	A vector of length equal to the number of nodes, specifying the attribute vector.
`egodata`	An [egodata](#) object.
`levels`	Starting set of levels to use; defaults to the sorted list of unique attributes.

Details

The `accept` argument is meant to allow the user to quickly check whether the output is of an *acceptable* class or mode. Typically, if a term accepts a character (i.e., categorical) attribute, it will also accept a numeric one, treating each number as a category label. For this reason, the following outputs are defined:

`"character"` Accept any mode or class (since it can beconverted to character).

`"numeric"` Accept real, integer, or logical.

`"logical"` Accept logical.

`"integer"` Accept integer or logical.

`"natural"` Accept a strictly positive integer.

`"0natural"` Accept a nonnegative integer or logical.

`"nonnegative"` Accept a nonnegative number or logical.

`"positive"` Accept a strictly positive number or logical.

`"paste"` Paste together with dot as the separator.

`"stop"` Fail with an error message.

`"matrix"` Construct and/or return a matrix whose rows correspond to vertices.

Value

`ergm.ego_get_vattr` returns a vector of length equal to the number of nodes giving the selected attribute function. It may also have an attribute `"name"`, which controls the suggested name of the attribute combination.

`ergm.ego_attr_levels` returns a vector of levels to use and their order.

Examples

```
data(florentine)
flomego <- as.egodata(flomarriage)
ergm.ego_get_vattr("priorates", flomego$egos)
ergm.ego_get_vattr(~priorates, flomego$alters)
ergm.ego_get_vattr(c("wealth","priorates"), flomego$egos)
ergm.ego_get_vattr(~priorates>30, flomego$alters)
(a <- ergm.ego_get_vattr(~cut(priorates,c(-Inf,0,20,40,60,Inf),label=FALSE)-1, flomego$egos))
ergm.ego_attr_levels(NULL, a, flomego$egos)
ergm.ego_attr_levels(-1, a, flomego$egos)
ergm.ego_attr_levels(1:2, a, flomego$egos)
ergm.ego_attr_levels(I(1:2), a, flomego$egos)
```

---

predict.ergm.ego   *ERGM-based predicted tie probabilities for the pseudo-population network*

Description

ERGM-based predicted tie probabilities for the pseudo-population network

Usage

```
## S3 method for class 'ergm.ego'
predict(object, ...)
```

Arguments

object	model fit as returned by ergm.ego()
...	other arguments passed to/from other methods

Value

See ergm::predict.ergm()

---

simulate.ergm.ego   *Simulate from a* ergm.ego *fit.*

Description

A wrapper around simulate.formula to simulate networks from an ERGM fit using ergm.ego.

Usage

```
## S3 method for class 'ergm.ego'
simulate(
  object,
  nsim = 1,
  seed = NULL,
  constraints = object$constraints,
  popsize = if (object$popsize == 1) object$ppopsize else object$popsize,
  control = control.simulate.ergm.ego(),
 output = c("network", "stats", "edgelist", "pending_update_network", "ergm_state"),
  ...,
  verbose = FALSE
)
```

Arguments

object	An [ergm.ego](#) fit.
nsim	Number of realizations to simulate.
seed	Random seed.
constraints, ...	
	Additional arguments passed to [san](#) and [simulate.formula](#).
popsize	Either network size to which to scale the model for simulation or a [data.frame](#) with at least those ego attributes required to estimate the model, to simulate over a specific set of actors.
control	A [control.simulate.ergm.ego](#) control list.
output	one of "network", "stats", "edgelist", "pending_update_network", or, for future compatibility, "ergm_state". See help for [simulate.ergm()](#) for explanation.
verbose	Verbosity of output.

Value

The ouput has the same format (with the same options) as [simulate.formula](#). If output="stats" is passed, an additional attribute, "ppopsize" is set, giving the actual size of the network recon-structed, when the pop.wt control parameter is set to "round" and "popsize" is not a multiple of the egocentric sample size or the sampling weights.

Author(s)

Pavel N. Krivitsky

References

Pavel N. Krivitsky and Martina Morris. Inference for Social Network Models from Egocentrically-Sampled Data, with Application to Understanding Persistent Racial Disparities in HIV Prevalence in the US. Thechnical Report. National Institute for Applied Statistics Research Australia, University of Wollongong, 2015(05-15). doi: [10.1214/16AOAS1010](#)

Pavel N. Krivitsky, Mark S. Handcock, and Martina Morris. Adjusting for Network Size and Composition Effects in Exponential-Family Random Graph Models. *Statistical Methodology*, 2011, 8(4), 319-339. doi: 10.1016/j.stamet.2011.01.005

See Also

simulate.formula, simulate.ergm

Examples

```
data(faux.mesa.high)
fmh.ego <- as.egodata(faux.mesa.high)
egofit <- ergm.ego(fmh.ego~edges+degree(0:3)+nodefactor("Race")+nodematch("Race")
             +nodefactor("Sex")+nodematch("Sex")+absdiff("Grade"),
             popsize=network.size(faux.mesa.high))
colMeans(egosim <- simulate(egofit, popsize=300,nsim=50,
                   output="stats", control=control.simulate.ergm.ego(
                   simulate.control=control.simulate.formula(MCMC.burnin=2e6))))
colMeans(egosim)/attr(egosim,"ppopsize")*network.size(faux.mesa.high)
summary(faux.mesa.high~edges+degree(0:3)+nodefactor("Race")+nodematch("Race")
         +nodefactor("Sex")+nodematch("Sex")+absdiff("Grade"))
```

---

summary_formula.egodata
                              Calculation of ERGM-style summary statistics for egodata *objects.*

Description

Used to calculate the specified network statistics inferred from a egodata object.

Usage

```
## S3 method for class 'egodata'
summary_formula(object, ..., basis = NULL, individual = FALSE, scaleto = NULL)
```

Arguments

object	An ergm-style formula with a egodata object as the LHS.
	For a list of currently implemented egocentric terms for the RHS, see ergm.ego-terms.
...	Not used at this time.
basis	An optional egodata object relative to which the statistics should be calculated.
individual	If FALSE (the default), calculate the estimated per-capita statistics, weighted according to the ego weights, then scale them up to a network of size scaleto.
	If TRUE, calculate each ego's individual contribution to the specified network statistics.
scaleto	Size of a hypothetical network to which to scale the statistics. Defaults to the number of egos in the dataset.

Value

If individual==FALSE, a named vector of statistics. If individual==TRUE, a matrix with a row for each ego, giving that ego's contribution to the network statistic.

Author(s)

Pavel N. Krivitsky

References

Pavel N. Krivitsky and Martina Morris. Inference for Social Network Models from Egocentrically-Sampled Data, with Application to Understanding Persistent Racial Disparities in HIV Prevalence in the US. Thechnical Report. National Institute for Applied Statistics Research Australia, University of Wollongong, 2015(05-15). doi: 10.1214/16AOAS1010

Pavel N. Krivitsky, Mark S. Handcock, and Martina Morris. Adjusting for Network Size and Composition Effects in Exponential-Family Random Graph Models. *Statistical Methodology*, 2011, 8(4), 319-339. doi: 10.1016/j.stamet.2011.01.005

See Also

summary_formula, summary_formula.ergm

Examples

```
data(faux.mesa.high)
fmh.ego <- as.egodata(faux.mesa.high)
(nw.summ <- summary(faux.mesa.high~edges+degree(0:3)+nodematch("Race")+
                    nodematch("Sex")+absdiff("Grade")+nodemix("Grade")))

(ego.summ <- summary(fmh.ego~edges+degree(0:3)+nodematch("Race")+nodematch("Sex")+
                     absdiff("Grade")+nodemix("Grade"),
                     scaleto=network.size(faux.mesa.high)))

stopifnot(isTRUE(all.equal(nw.summ,ego.summ)))
```

[.egodata   *Subsetting egodata Objects*

Description

Returns subsets of egodata objects that meet conditions.

Usage

```
## S3 method for class 'egodata'
x[i, j, ..., dup.action = c("make.unique", "fail", "number")]

## S3 method for class 'egodata'
subset(x, subset, select, ..., dup.action = c("make.unique", "fail", "number"))
```

Arguments

x	An egodata object.
...	Additional arguments, currently unused.
dup.action	What to do when an ego is referenced multiple times:
	"make.unique" Construct new unique ego IDs using the make.unique function
	"fail" Exit with an error.
	"number" Number the egos consecutively in the order they were selected
subset, i	An expression (evaluated in the context of the egos table of x producing a logical, integer, or character vector indicating which egos to select (and, for the latter two, how many times).
select, j	A numeric or character vector specifying the columns of egos and alters to select.

Value

An egodata object.

Author(s)

Pavel N. Krivitsky

See Also

sample.egodata

Index