Package 'egor'

October 8, 2019

Type Package

Title Import and Analyse Ego-Centered Network Data

Version 0.19.10

Date 2019-10-06

Description Tools for importing, analyzing and visualizing ego-centered network data. Supports several data formats, including the export formats of 'EgoNet', 'EgoWeb 2.0' and 'openeddi'. An interactive (shiny) app for the intuitive visualization of ego-centered networks is provided. Also included are procedures for creating and visualizing Clustered Graphs (Lerner 2008 <DOI:10.1109/PACIFICVIS.2008.4475458>).

URL https://github.com/tilltnet/egor, https://tilltnet.github.io/egor/

BugReports https://github.com/tilltnet/egor/issues

License AGPL-3

Depends R (>= 2.10), dplyr, tibble

Imports tidygraph, igraph, network, shiny, plyr, survey, tidyr, methods, utils, purrr, rlang

Suggests knitr, testthat, rmarkdown

VignetteBuilder knitr

RoxygenNote 6.1.1.9000

LazyData true

Encoding UTF-8

NeedsCompilation no

Author Till Krenz [aut, cre], Pavel N. Krivitsky [aut], Raffaele Vacca [aut], Michal Bojanowski [aut], Markus Gamper [ctb], Andreas Herz [aut], Christopher McCarty [ctb]

Maintainer Till Krenz <egor@tillt.net>

Repository CRAN

Date/Publication 2019-10-07 22:10:06 UTC

R topics documented:

aaties32 *32 sets of randomly created alter-alter ties belonging to ego-centered networks*

Description

32 sets of randomly created alter-alter ties belonging to ego-centered networks

Usage

```
aaties32
```

Format

A data frame with 32 sets of alter-alter relations and 4 variables:

.EGOID ego identifier

.SRCID source alter ID

.TGTID target alter ID

weight weight of relation

activate.egor *Activate ego, alter or alter-alter tie data llevel of an egor .dataect*

Description

This function activates one of the data levels of an egor .dataect, so that the dplyr verbs know which level to execute on.

Usage

```
## S3 method for class 'egor'
activate(.data, what)
```

Arguments

.data	The egor .dataect.
what	Character naming the level to activate, this can be "ego", "alter" or "aatie".

alters32 *32 sets of randomly created alters belonging to ego-centered networks*

Description

32 sets of randomly created alters belonging to ego-centered networks

Usage

```
alters32
```

Format

A data frame with 32 sets of up to 32 alters per egoID and 7 variables:

.ALTID alter identifier

.EGOID ego identifier

age age in categories

age.years age in years

country country

income income

sex gender

alter_design *Set and query the alter nomination design*

Description

Extract, set, or update the alter nomination design associated with an ego-centered dataset.

Usage

```
alter_design(x, ...)

## S3 method for class 'egor'
alter_design(x, which, ...)

alter_design(x, ...) <- value

## S3 replacement method for class 'egor'
alter_design(x, which, ...) <- value
```

Arguments

x	an egor object.
...	arguments to be passed to methods
which	name of the alter design setting to query or replace
value	if which is specified, the new value of the attribute; if not, a named list of settings that replace their old values.

---

alts_diversity_count *Calculate diversity measures on an* egor *object.*

Description

alts_diversity_count() counts the categories of a variable present in the networks of an egor object. alts_diversity_entropy() calculates the Shannon entropy as a measurement for diversity of an alter attribute.

Usage

```
alts_diversity_count(object, alt.attr)

alts_diversity_entropy(object, alt.attr, base = 2)
```

Arguments

object	An egor object.
alt.attr	A character naming the variable containing the alter-attribute.
base	Numeric, base value of logarithm for entropy calculation.

Value

A numeric vector.

Author(s)

Michał Bojanowski, <m.bojanowski@uw.edu.pl>

Till Krenz, <public@tillt.net>

Examples

```
data("egor32")
alts_diversity_count(egor32, "age")
alts_diversity_entropy(egor32, "age")
```

append_egor *Append rows/columns to ego, alter or aatie data*

Description

These work like dplyr's bind_cols() and bind_rows(). The first argument has to be an egor object. Additional rows/columns are added bottom/RHS of the active data level (ego, alter, aatie).

Usage

```
append_rows(.egor, ..., .id = NULL)

append_cols(.egor, ...)
```

Arguments

.egor	An egor object.
...	Data frames to combine.
.id	Data frame identifier.

as_alters_df *Create global alters and alter-alter relations dataframes from an* egor *object*

Description

Provided an egor-object, these functions create a 'global' data.frame, containing alter attributes, or alter-alter relations. The resulting dataframes are useful for advanced analysis procedures, e.g. multi-level regressions.

Usage

```
as_alters_df(object, include.ego.vars = FALSE)

as_aaties_df(object, include.ego.vars = FALSE,
  include.alt.vars = FALSE)
```

Arguments

object	An egor object. a new variable with the specified name is created.
include.ego.vars	Logical, specifying if ego variables should be included in the result.
include.alt.vars	Logical, specifying if alter variables should be included in the result.

Examples

```
# Load example data
data(egor32)

# Create global alters dataframes
as_alters_df(egor32)

# Create global alter-alter relaions dataframes
as_aaties_df(egor32)

# ... adding alter variables
as_aaties_df(egor32, include.alt.vars = TRUE)
```

---

as_igraph *Convert* egor *object to* network *or* igraph *objects*

Description

These functions convert an egor object into a list of network or igraph objects. By default ego itself is not included in the created objects, there is a parameter (**include.egor**) that allows for including ego.

Usage

```
as_igraph(x, directed = FALSE, include.ego = FALSE, ego.attrs = NULL,
  ego.alter.weights = NULL)

## S3 method for class 'nested_egor'
as_igraph(x, directed = FALSE,
  include.ego = FALSE, ego.attrs = NULL, ego.alter.weights = NULL)

as.igraph.egor(x, directed = FALSE, include.ego = FALSE,
  ego.attrs = NULL, ego.alter.weights = NULL)
```

Arguments

x	An egor object.
directed	Logical, indicating if alter-alter relations are directed.
include.ego	Logical. Should ego be included?
ego.attrs	Vector of names (character) or indices (numeric) of ego variables that should be carried over to the network/ igraph objects.
ego.alter.weights	Vector of names (character) or indices (numeric) of alter variables that should be carried over to the the network/ igraph objects, as edge attributes of the ego-alter relations.

Details

The names of the variables specified in ego.attr and ego.alter.attr need to be the same as the names of corresponding alter attributes, in order for those variables to be complete in the resulting network/ igraph object (see example).

`as_network` *Creates a list of statnet's network objects, from an* `egor` *object.*

Description

Creates a list of statnet's network objects, from an `egor` object.

Usage

```
as_network(x, directed = FALSE, include.ego = FALSE,
  ego.attrs = NULL, ego.alter.weights = NULL)

## S3 method for class 'egor'
as.network(x, directed = FALSE, include.ego = FALSE,
  ego.attrs = NULL, ego.alter.weights = NULL)
```

Arguments

`x`	An egor Object.
`directed`	Logical.
`include.ego`	Logical.
`ego.attrs`	Names of ego variables.
`ego.alter.weights`	Name of ego alter weight variable.

`clustered_graphs` *Cluster ego-centered networks by a grouping factor*

Description

The idea of clustered graphs is to reduce the complexity of an ego-centered network graph by visualizing alters in clusters defined by a categorical variable (Lerner et al. 2008). `clustered_graphs()` calculates group sizes, inter and intra group tie densities and returns these informations in a `list` of `igraph` objects.

Usage

```
clustered_graphs(object, ..., clust.groups)

## S3 method for class 'list'
clustered_graphs(object, aaties, clust.groups, ...)

## S3 method for class 'egor'
clustered_graphs(object, clust.groups, ...)

## S3 method for class 'data.frame'
clustered_graphs(object, aaties, clust.groups,
  egoID = ".egoID", ...)
```

Arguments

object	Either an egor object or a data.frame/ list containg alter attributes.
...	arguments to be passed to methods
clust.groups	A character naming the factor variable defining the groups.
aaties	data.frame/ list containg alter-alter relations as a 'global edge list' or as a list of 'edge lists'. (not needed if object is an egor object).
egoID	Character. Name of the variable identifying egos (default: "egoID").

Value

clustered_graphs returns a list of graph objects representing the clustered ego-centered network data;

References

Brandes, U., Lerner, J., Lubbers, M. J., McCarty, C., & Molina, J. L. (2008). Visual Statistics for Collections of Clustered Graphs. 2008 IEEE Pacific Visualization Symposium, 47-54.

See Also

vis_clustered_graphs for visualizing clustered graphs

Examples

```
data("egor32")

# Simplify networks to clustered graphs, stored as igraph objects
graphs <- clustered_graphs(egor32, "country")

# Visualise
vis_clustered_graphs(graphs,
                     node.size.multiplier = 5,
                     edge.width.multiplier = 25,
                     labels = TRUE)
```

composition *Calculate the composition of alter attributes in an* egor *object*

Description

composition() calculates the proportional or absolute composition of alters for a given attribute/variable.

Usage

```
composition(object, alt.attr, absolute = FALSE)
```

Arguments

object	An egor object.
alt.attr	A character naming the variable containing the alter-attribute.
absolute	Logical indicating if the results should be absolute.

Value

A tibble with the values per category in the columns.

Examples

```
data("egor32")
composition(egor32, "sex")
```

comp_ei *Calculate the EI-Indices of an* egor *object as a measurement of ego-alter homophily*

Description

comp_ei() calculates the EI-Index values as a measurement for ego-alter homophily.

Usage

```
comp_ei(object, alt.attr, ego.attr)
```

Arguments

object	An egor object.
alt.attr	A character naming the variable containing the alter-attribute.
ego.attr	A character naming an ego attribute.

Value

A numeric vector.

Examples

```
data("egor32")
comp_ei(egor32, "age", "age")
```

---

comp_ply *Calculate third-party compositional measures on an* egor *object*

Description

comp_ply() applies a function, that uses an alter attribute to calculate a compositional measurement, on all networks in an egor object and returns a numeric vector.

Usage

```
comp_ply(object, alt.attr, .f, ..., ego.attr = NULL)
```

Arguments

object	An egor object.
alt.attr	A character naming the variable containing the alter-attribute.
.f	A function that returns a numeric.
...	Optional arguments to .f.
ego.attr	Optional character naming an ego attribute.

Details

When an ego attribute is used the .f is called like this: .f(alt.attr,ego.attr,...). .f must return a single numeric value.

Value

A numeric vector.

Author(s)

Michał Bojanowski, <m.bojanowski@uw.edu.pl>

Till Krenz, <public@tillt.net>

Examples

```
df <- make_egor(10, 32)
comp_ply(df, "age.years", sd, na.rm = TRUE)
```

egor *egor - a data class for ego-centered network data.*

Description

The function `egor()` is used to create an egor object from ego-centered network data.

Usage

```
egor(alters, egos = NULL, aaties = NULL, ID.vars = list(ego =
  "egoID", alter = "alterID", source = "Source", target = "Target"),
  ego_design = list(~1), alter_design = list(max = Inf))

as.egor(x, ...)
```

Arguments

`alters`	either a `data.frame` containing the alters (whose nominator is identified by the column specified by `egoID` or a list of data frames with the same columns, one for each ego, with empty data frames or `NULL`s corresponding to egos with no nominees.
`egos`	`data.frame` containing the egos.
`aaties`	`data.frame` containing the alter-alter relations in the style of an edge list, or a list of data frames similar to `alters.df`.
`ID.vars`	A named list containing column names of the relevant input columns: `ego` unique identifier associated with each ego, defaulting to `"egoID"`; has no effect if `alters.df` and `aaties.df` are both lists of data frames. `alter` unique-within-ego identifier associated with each alter, defaulting to `"alterID"`; optional `aaties.df` are not provided. `source` if `aaties.df` is provided, the column given the alter identifier of the origin of a relation. `target` if `aaties.df` is provided, the column given the alter identifier of the destination of a relation.
`ego_design`	A `list` of arguments to `survey::svydesign()` specifying the sampling design for the egos. If formulas, they can refer to columns of `egos.df`.
`alter_design`	A `list` of arguments specifying nomination information. Currently, the following elements are supported: `"max"` Maximum number of alters that an ego can nominate.
`x`	an object to be coerced to `egor`.
`...`	arguments to be passed to methods

Details

If parameters `alters.df`, `egos.df`, and `aaties.df` are data frames, they need to share a common ego ID variable, with corresponding values. If `alters.df` and `aaties.df` are lists of data frames, `egoID` is ignored and they are matched positionally with the rows of `egos.df`. Of the three parameters only `alters.df` is necessary to create an `egor` object, and `egos.df` and `aaties.df` are optional.

Value

Returns an `egor` object. An `egor` object is a `tibble` whose top-level columns store the ego attributes, and which has two special nested columns: `.alts`, containing, for each row (ego) a table of that ego's alter attributes and `.aaties`, a table containing that ego's alter–alter ties, if observed.

If alter-alter ties are observed, `.alts` also has a column `.altID` giving a unique (within each ego) ID of the alter, by which the alter can be identified in the `.aaties` table for that ego. `.aaties`, in turn, has columns `.srcID` and `.tgtID` that contain the source and the target of the alter-alter relation.

In addition, `egor` has two attributes: `ego_design`, containing an object returned by `survey::svydesign()` specifying the sampling design by which the egos were selected and `alter_design`, a `list` containing specification of how the alters were nominated. See the argument above for currently implemented settings.

Note

Column names `.alts`, `.aaties`, and `.egoRow` are reserved for internal use of `egor` and should not be used to store persistent data. Other `.`-led column names may be reserved in the future.

Examples

```
data("egos32")
data("alters32")
data("aaties32")

egor(alters32,
     egos32,
     aaties32,
     ID.vars = list(ego = ".EGOID",
                    alter = ".ALTID",
                    source = ".SRCID",
                    target =  ".TGTID"))
```

`egor-package-doc` `egor`

Description

R Package for importing and analyzing ego-centered-network data.

Details

Further Information or GitHub

Thanks to: Martina Morris, Michał Bojanowski

Author(s)

Till Krenz, <egor@tillt.net>

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

Raffaele Vacca, <r.vacca@ufl.edu>

Andreas Herz, <herzand@uni-hildesheim.de>

Christopher McCarty, <ufchris@ufl.edu>

Markus Gamper, <m.gamper@uni-koeln.de>

`egor32` *32 randomly created ego-centered networks stored as an egor object*

Description

32 randomly created ego-centered networks stored as an egor object

Usage

```
egor32
```

Format

An egor object with 32 ego-centered networks (5 variables):

egoID ego identifier

sex ego's gender

age ego's age

.alts nested column/list containing alters

.aaties nested column/list containing alter-alter relations

egor_vis_app — egor *Network Visualization App*

Description

Launches an interactive Shiny Web App, that creates a list of igraph objects from an 'egor' object and offers the user several graphical means of interacting with the visualization parameters for all networks in the egor object.

Usage

```
egor_vis_app(object = NULL, shiny_opts = list(launch.browser = TRUE))
```

Arguments

object	An egor object.
shiny_opts	List of arguments to be passed to shinyApp()'s options argument.

Examples

```
if(interactive()){
  data("egor32")
  egor_vis_app(egor32)
}
```

egos32 — *32 randomly created egos belonging to ego-centered networks*

Description

32 randomly created egos belonging to ego-centered networks

Usage

```
egos32
```

Format

A data frame with 32 sets of alter-alter relations and 4 variables:

.EGOID ego identifier

age age in categories

age.years age in years

country country

income income

sex gender

ego_density *Calculate the relationship density in ego-centered networks*

Description

This function uses an egor object and calculates the density of all the ego-centered networks listed in the 'egor' object. Instead of an egor object, alter and alter-alter data can be provided as lists or data.frames.

Usage

```
ego_density(object, ...)

## S3 method for class 'egor'
ego_density(object, weight = NULL, max.netsize = NULL,
  directed = FALSE, ...)
```

Arguments

object	Either an egor object or a data.frame/ list containg alter attributes.
...	arguments to be passed to methods
weight	Character naming a variable containing the weight values of relations. Weights should range from 0 to 1.
max.netsize	Optional parameter. Constant value used if the number of alters whose relations were collected is limited.
directed	logical indicating if the alter-alter relation data/ edges are directed or undirected.

Value

returns a vector of network density values.

Examples

```
data("egor32")
ego_density(egor32)
```

ego_design *Set and query the ego sampling design*

Description

Extract, set, or update the svydesign associated with an ego-centered dataset.

Usage

```
ego_design(x, ...)

## S3 method for class 'egor'
ego_design(x, ...)

ego_design(x, ...) <- value

## S3 replacement method for class 'egor'
ego_design(x, ...) <- value
```

Arguments

x	an egor object.
...	arguments to be passed to methods
value	either survey.design object (like one constructed by svydesign()) or a list of arguments to svydesign() specifying the sampling design for the egos. If the arguments are formulas, they can refer to columns (ego attributes) of x.

Note

This can be useful for adjusting or reinitializing the ego design information after the underlying ego attributes had been modified.

EI *Calculate the EI-Index for the alter-alter ties of an ego object*

Description

The EI-Index is the division of the intra-group edge density and the outer-group edge density. It is calculated for the whole network and for subgroups. The whole network EI is a metric indicating the tendency of a network to be clustered by the categories of a given factor variable. The EI value of a groups describes the tendency of a group to be connected or not connected to other groups. Additionally, the EI index can be employed as a measurement for egos tendency to homo-/heteorphily - use the comp_ei() command for that version of EI-Index.

Usage

```
EI(object, alt.attr)
```

Arguments

object	An egor object.
alt.attr	Character naming grouping variable.

References

Krackhardt, D., Stern, R.N., 1988. Informal networks and organizational crises: an experimental simulation. Social Psychology Quarterly 51 (2), 123-140.

Everett, M. G., & Borgatti, S. P. (2012). Categorical attribute based centrality: E-I and G-F centrality. Social Networks, 34(4), 562-569.

Examples

```
data("egor32")
EI(egor32, "sex")
```

---

gss2004 *A selective subset of GSS 2004 data*

Description

This is a selective subset of General Social Survey 2004 data containing variables from network questions. See Details for description how this particular subset was selected. The data has a near 0 research value, it is provided to illustrate the functions in **egor** package.

Format

A tibble with 499 rows and the variables listed below. Data was imported from SPSS file and are labelled. Functions in the **labelled** package can be used to handle them.

Variables:

id Case ID

vpsu, vstrat, wtssall Design variables and weight

age Ego's age in years

race Ego's race. 1=white, 2=black, 3=other

sex Ego's sex. 1=male, 2=female

marital Ego's marital status. 1=married, 2=widowed, 3=divorced, 4=separated, 5=never married

numgiven Number of alters mentioned

age[1-5] Alter's age in years

race[1-5] Alter's race. 1=asian, 2=black, 3=hispanic, 4=white, 5=other

sex[1-5] Alter's sex. 1=male, 2=female

spouse[1-5] Whether alter is a spouse of ego. 1=mentioned, 2=not mentioned

close[1-4 [2-5]] How close are the two alters according to ego. 1=especially close, 2=know each other, 3=total strangers

Details

This dataset was created from original GSS 2004 data for illustrative purposes such that (1) it is small and (2) contains just enough variation in respondent's personal networks to illustrate various functions in the package. It is essentially a stratified sample from original data (1472 cases). Strata correspond to groups of cases created from unique combinations of values on the following ego variables: `age` (3 categories), `race`, `sex`, `marital`, `numgiven`. At most 2 cases were sampled from each stratum via simple random sampling with replacement.

Source

General Social Survey data at NORC: http://gss.norc.org/get-the-data

---

`helper` *General helper functions*

Description

Helper functions for ego centered network analysis

Usage

```
as_nested_egor(x)

dyad.poss(max.alters, directed = FALSE)

sanitize.wide.edges(max.alters)

create_edge_names_wide(x)

dyads_possible_between_groups(x, y)

din_page_dist(x)
```

Arguments

`x`	`Numeric`.
`max.alters`	A `numeric` giving the maximum number of alters.
`directed`	A `logical` value indicating directedness of alter-alter data.
`y`	`Numeric`.

Functions

- as_nested_egor: Converts an egor object to a "legacy" egor object with nested .alts and .aaties columns.
- dyad.poss: Returns the count of possible edges in an undirected or directed, ego-centered network, based on the number of alters.
- sanitize.wide.edges: Generates a data.frame marking possible dyads in a wide alter-alter relation data.frame. Row names corresponds to the network size. This is useful for sanitizing alter-alter relations in the wide format.
- create_edge_names_wide: Creates a vector of names for variables containing data on alter-alter relations/ dyads in ego-centered networks.
- dyads_possible_between_groups: Calculates the possible edges between members of different groups in an ego-centered network.
- din_page_dist: Calculates the optimal distribution of a number of equally sized objects on a DIN-Norm DIN 476 (i.e. DIN A4) page in landscape view.

---

make_egor *Generate random ego-centered-network data.*

Description

This function generates random ego-centered-network data for a specified number of networks with a maximum network size. The network size of the generated networks is a normal distribution with sd=5.

Usage

```
make_egor(net.count, max.alters, netsize = NULL, plot = FALSE)
```

Arguments

net.count	Number of networks/ egos to generate.
max.alters	Maximum size of networks.
netsize	Numeric for fixed network sizes.
plot	whether to plot the network size distribution.

onefile_to_egor *Import ego-centered network data from 'one file format'*

Description

This function imports ego-centered network data, stored in a single file, providing ego, alter and edge data. This data format is used by the Allbus 2010 (GESIS) and similar social surveys.

Usage

```
onefile_to_egor(egos, netsize, ID.vars = list(ego = "egoID"),
  attr.start.col, attr.end.col, max.alters, aa.first.var,
  aa.regex = NULL, ego.vars = NULL, var.wise = FALSE, ...)
```

Arguments

egos	Data frame containg ego data (egos as cases)
netsize	Vector containing values of network size per ego.
ID.vars	A named list containing column names of the relevant input columns: ego unique identifier associated with each ego, defaulting to "egoID"; has no effect if alters.df and aaties.df are both lists of data frames. alter unique-within-ego identifier associated with each alter, defaulting to "alterID"; optional aaties.df are not provided. source if aaties.df is provided, the column given the alter identifier of the origin of a relation. target if aaties.df is provided, the column given the alter identifier of the destination of a relation.
attr.start.col	Index or name of the first colum containing alter attributes.
attr.end.col	Index or name of the last colum containing alter attributes.
max.alters	Maximum number of alters.
aa.first.var	First column containing alter-alter relations/ edges.
aa.regex	A Perl regular expression with name capture, intended to be run on column names and capturing via named capture the following regex groups: "attr", "src", and "tgt", representing the edge attribute being captured, the source (or the first alter identified), and the target (or the second alter identified) of the edge, respectively. See regex for more information.
ego.vars	A data.frame of alter attributes in the wide format.
var.wise	Logical value indicatin if the alter attributes are sorted variable wise (defaults to FALSE).
...	additional arguments to egor().

Value

A `list` of six objects - the **egoR** object: (1) egos.df: `dataframe` of all egos and their attributes; (2) alters.df: `dataframe` of all alters; (3) alters.list: `list` of `dataframes` of all alters per ego; (4) edges: `list` of `dataframes` with edge lists per network; (5) graphs: `list` of igraph objects; (6) results: a result `dataframe`, pre-populated with the network size of each network

References

Muller, C., Wellman, B., & Marin, A. (1999). How to Use SPSS to Study Ego-Centered Networks. Bulletin de Methodologie Sociologique, 64(1), 83-100.

`plot_egograms` *Plotting* egor *objects*

Description

egor Objects can be plotted as *egographs* or *egograms*. By default networks of the four first egos are plotted.

Usage

```
plot_egograms(x, nnumber = 1, x_dim = 1, y_dim = 1, venn_var,
  pie_var, vertex_size_var = NULL, vertex_color_var = NULL,
  vertex_color_palette = "Heat Colors",
  vertex_color_legend_label = NULL, vertex_label_var = NULL,
  edge_width_var = NULL, edge_color_var = NULL,
  edge_color_palette = "Heat Colors", highlight_box_col_var = NULL,
  highlight_box_col_palette = "Heat Colors", res_disp_vars = NULL,
  vertex_zoom = 1, edge_zoom = 3, font_size = 1,
  venn_colors = NULL, show_venn_labels = TRUE, ...)

plot_ego_graphs(x, nnumber, x_dim = 1, y_dim = 1,
  vertex_size_var = NULL, vertex_color_var = NULL,
  vertex_color_palette = "Heat Colors",
  vertex_color_legend_label = NULL, vertex_label_var = NULL,
  edge_width_var = NULL, edge_color_var = NULL,
  edge_color_palette = "Heat Colors", highlight_box_col_var = NULL,
  highlight_box_col_palette = "Heat Colors", res_disp_vars = NULL,
  vertex_zoom = 1, edge_zoom = 3, font_size = 1,
  include_ego = FALSE, ...)

plot_egor(x, nnumber = 1, x_dim = 2, y_dim = 2, ...,
  type = c("egograph", "egogram"))

## S3 method for class 'egor'
plot(x, ...)
```

Arguments

x	An *egor* object.
nnumber	Ego row number.
x_dim	Number of ego networks to plot horizontally.
y_dim	Number of ego networks to plot vertically
venn_var	Name (character) of alter column.
pie_var	Name (character) of alter column.
vertex_size_var	Name (**character**) of alter column.
vertex_color_var	Name (**character**) of alter column.
vertex_color_palette	Name (**character**) of color palette.
vertex_color_legend_label	Character.
vertex_label_var	Name (**character**) of alter column.
edge_width_var	Name (**character**) of aatie column.
edge_color_var	Name (**character**) of aatie column.
edge_color_palette	Name (**character**) of color palette.
highlight_box_col_var	Name (**character**) of ego column.
highlight_box_col_palette	Name (**character**) of color palette.
res_disp_vars	Name (**character**) of ego column.
vertex_zoom	Numeric.
edge_zoom	Numeric.
font_size	Numeric.
venn_colors	Vector of colors.
show_venn_labels	Logical.
...	Additional arguments forwared to plot.igraph.
include_ego	Logical.
type	Character. Either "egograph" or "egogram".

Details

For type eqals "egograph" ego networks are plotted using

Functions

- plot_egograms: Plots an ego-socio-gram.
- plot_ego_graphs: Plots an ego graph.

read_egonet *Read ego-centered network data exported with EgoNet software as an* egor *object*

Description

This function imports ego-centered network data from folders with separate files for alters-level and edge data. It will run some basic checks upon the completeness of the data and inform the user of potential problems. This function can be used to import data exported from EgoNet (McCarty 2011).

Usage

```
read_egonet(egos.file, alter.folder, edge.folder, csv.sep = ",",
  ID.vars = list(ego = "egoID", alter = "alterID", source = "Source",
  target = "Target"), first.col.row.names = FALSE, ...)
```

Arguments

egos.file	File name of the .csv file containing the ego data.
alter.folder	Folder name of the folder containing the alter data in separate .csv files for each ego/ network.
edge.folder	Folder name of the folder containing the edge/ tie data in separate .csv files for each ego/ network.
csv.sep	Character indicating the separator used in csv files.
ID.vars	A named list containing column names of the relevant input columns: ego unique identifier associated with each ego, defaulting to "egoID"; has no effect if alters.df and aaties.df are both lists of data frames. alter unique-within-ego identifier associated with each alter, defaulting to "alterID"; optional aaties.df are not provided. source if aaties.df is provided, the column given the alter identifier of the origin of a relation. target if aaties.df is provided, the column given the alter identifier of the destination of a relation.
first.col.row.names	Boolean indicating if first column contains row names, that are to be skipped, default is FALSE.
...	additional arguments to egor().

Value

A list of six objects - the **egoR** object: (1) egos.df: dataframe of all egos and their attributes; (2) alters.df: dataframe of all alters; (3) alters.list: list of dataframes of all alters per ego; (4) edges: list of dataframes with edge lists per network; (5) graphs: list of igraph objects; (6) results: a result dataframe, pre-populated with the network size of each network

Examples

```
egos.file <-  system.file("extdata", "egos_32.csv", package = "egor")
alters.folder <- system.file("extdata", "alters_32", package = "egor")
edge.folder <-  system.file("extdata", "edges_32", package = "egor")

ef <- read_egonet(egos.file = egos.file,
                          alter.folder = alters.folder,
                          edge.folder = edge.folder,
                          csv.sep = ";")
```

`rowlist` *Convert a table to a list of rows*

Description

A convenience function converting a `data.frame()` or a `tibble()`.

Usage

```
rowlist(x)
```

Arguments

`x`	a `data.frame()`, a `tibble()`, or some other table data structure backed by a `list()` of columns.

Value

A `list()` of length `nrow(x)`, with each element itself a named `list()` containing the elements in the corresponding row.

Examples

```
library(tibble)
(df <- tibble(x=2:1, y=list(list(1:3), list(3:4))))
rowlist(df)
```

`subset.egor` *Filter and Subset Ego-centered Datasets*

Description

Functions to index and take subsets of `egor()` objects: manipulate egos, alters, or alter-alter ties.

Usage

```
## S3 method for class 'egor'
subset(x, subset, ..., unit = attr(x, "active"))

## S3 method for class 'egor'
x[i, j, unit = attr(x, "active"), ...]
```

Arguments

`x`	an `egor()` object.
`subset`	either an expression evaluated on each of the rows of the selected unit (as in the eponymous argument of `subset()`) or a function whose first argument is a row, specifying which egos, alters, or alter-alter ties to keep. The expressions can access variables in the calling environment; columns of the active unit, columns of other units with which the active unit shares an ego via `egos$`, `alters$`, and `aaties$` as well as the following "virtual" columns to simplify indexing: **Ego index** `.egoRow` contains the index (counting from 1) of the row being evaluated. (This can be used to access vector variables in the calling environment.) **Alter index** `.altRow` contains the index (counting from 1) of the row number in the alter table. **Alter–alter indices** `.srcRow` **and** `.tgtRow` contain the index (counting from 1) of the row of the alter being refereced by `.srcID` and `.tgtID`. (This can be used to quickly access the attributes of the alters in question.)
`...`	extra arguments to `subset` if `subset` is a function; otherwise unused.
`unit`	a selector of the unit of analysis being affected: the egos, the alters or the (alter-alter) ties. Note that only one type of unit can be affected at a time. Defaults to the current active unit selected by `activate.egor()`.
`i`	numeric or logical vector indexing the appropriate unit.
`j`	either an integer vector specifying which columns of the filtered structure (ego, alters, or ties) to select, or a logical vector specifying which columns to keep. Note that the special columns .egoID, .altID, .srcID, .tgtID are not indexed by `j`.

Details

Removing or duplicating an ego will also remove or duplicate their alters and ties.

Value

An egor() object.

Examples

```
# Generate a small sample dataset
(e <- make_egor(5,4))

# First three egos in the dataset
e[1:3,]

# Using an external vector
# (though normally, we would use e[.keep,] here)
.keep <- rep(c(TRUE, FALSE), length.out=nrow(e$ego))
subset(e, .keep)
```

---

summary.egor *Methods to print and summarize* egor *objects*

Description

Methods to print and summarize egor objects

Usage

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

## S3 method for class 'egor'
print(x, ..., n = 3)
```

Arguments

object, x	an egor object.
...	additional arguments, either unused or passed to lower-level functions.
n	Number of rows to print.

threefiles_to_egor *Read/ import ego-centered network data from the three files format, EgoWeb2.0 or openeddi.*

Description

These functions read ego-centered network data from the three files format, EgoWeb2.0 or openeddi and transform it to an egoR object. The three files format consists of an ego file, on alters file and one file containing the edge data. EgoWeb2.0 and openeddi use variations of this format.

Usage

```
threefiles_to_egor(egos, alters.df, edges, ID.vars = list(ego = "egoID",
  alter = "alterID", source = "Source", target = "Target"),
  ego.vars = NULL, ...)

read_egoweb(alter.file, edges.file, egos.file = NULL,
  ID.vars = list(ego = "EgoID", alter = "Alter.Number", source =
  "Alter.1.Number", target = "Alter.2.Number"), ego.vars = NULL, ...)

read_openeddi(egos.file = NULL, alters.file = NULL,
  edges.file = NULL, ID.vars = list(ego = "puid", alter = "nameid",
  source = "nameid", target = "targetid"), ego.vars = NULL, ...)
```

Arguments

egos	Data frame containg ego data (egos as cases)
alters.df	dataframe containing alters data (alters as cases), alters are separated by a variable containg an egoID.
edges	Dataframe. A global edge list, first column is ego ID variable. egos.
ID.vars	A named list containing column names of the relevant input columns: ego unique identifier associated with each ego, defaulting to "egoID"; has no effect if alters.df and aaties.df are both lists of data frames. alter unique-within-ego identifier associated with each alter, defaulting to "alterID"; optional aaties.df are not provided. source if aaties.df is provided, the column given the alter identifier of the origin of a relation. target if aaties.df is provided, the column given the alter identifier of the destination of a relation.
ego.vars	A data.frame of alter attributes in the wide format.
...	additional arguments to egor().
alter.file	A character specifiying the filename of the alters data.
edges.file	A character specifiying the filename of the edge data.
egos.file	A character specifiying the filename of the ego data.
alters.file	Character name of the alters data file.

Value

A `list` of six objects - the **egoR** object: (1) egos.df: `dataframe` of all egos and their attributes; (2) alters.df: `dataframe` of all alters; (3) alters.list: `list` of `dataframes` of all alters per ego; (4) edges: `list` of `dataframes` with edge lists per network; (5) graphs: `list` of igraph objects; (6) results: a result `dataframe`, pre-populated with the network size of each network

Functions

- `read_egoweb`: This function reads in data from an EgoWeb 2.0 survey and transforms it to an egoR object. If no file name for the egos file is provided ego data is assumed to be merged with alters data and it will be extracted by `read_egoweb`. By default the standard ID variable names of EgoWeb are used, if you need to specify the ID variable names use the ID.vars parameter. Further Information: github.com/qualintitative/egoweb
- `read_openeddi`: This function reads in data created by the openeddi survey software and transforms it to an egoR object. If no parameters are provided `read_openeddi` will try to find the adequate files in the working directory. By default the standard ID variable names of openeddi are used, if you need to specify the ID variable names use the ID.vars parameter. Further Information: www.openeddi.com

Examples

```
# The data for read.egonet.threefiles() needs to be loaded with read.csv(),
# for it to be converted to an egoR object.
egos.file <-  system.file("extdata", "egos_32.csv", package = "egor")
alters.file <- system.file("extdata", "alters_32.csv", package = "egor")
edges.file <-  system.file("extdata", "edges_32.csv", package = "egor")

egos <- read.csv2(egos.file)
alters <- read.csv2(alters.file)
edges <- read.csv2(edges.file)

tf <- threefiles_to_egor(egos = egos, alters.df = alters, edges = edges)

# read_egoweb() and read_openeddi() read the files directly from the disk.

#' # Fetch current working directory
wd <- getwd()

#' setwd(system.file("extdata", "openeddi", package = "egor"))
oe <- read_openeddi()

setwd(system.file("extdata", "egoweb", package = "egor"))
ew <- read_egoweb(alter.file = "alters_32.csv", edges.file = "edges_32.csv",
                  egos.file = "egos_32.csv")

# Restore working directory
setwd(wd)
```

trim_aaties *Trims alter-alter ties of alters that are missing/ deleted from alters data*

Description

Trims alter-alter ties of alters that are missing/ deleted from alters data

Usage

```
trim_aaties(object)
```

Arguments

object	An egor object.

Value

An egor object with trimmed alter-alter ties (.aaties).

trim_alters *Trims alters that are missing/ deleted from ego data*

Description

Trims alters that are missing/ deleted from ego data

Usage

```
trim_alters(object)
```

Arguments

object	An egor object.

Value

An egor object with trimmed alter-alter ties (.aaties).

twofiles_to_egor *Import ego-centered network data from two file format*

Description

This function imports ego-centered network data, stored in two files, where one file contains the ego attributes and the edge information and the other file contains the alters data. This form of data storage for ego-centered network data is proposed by Muller, Wellman and Marin (1999).

Usage

```
twofiles_to_egor(egos, alters, netsize = NULL, ID.vars = list(ego =
  "egoID", alter = "alterID", source = "Source", target = "Target"),
  e.max.alters, e.first.var, ego.vars = NULL, selection = NULL, ...)
```

Arguments

egos	Data frame containg ego data (egos as cases)
alters	Data frame containing alters data (alters as cases), alters are separated by a variable containg an egoID.
netsize	Vector containing values of network size per ego.
ID.vars	A named list containing column names of the relevant input columns: ego unique identifier associated with each ego, defaulting to "egoID"; has no effect if alters.df and aaties.df are both lists of data frames. alter unique-within-ego identifier associated with each alter, defaulting to "alterID"; optional aaties.df are not provided. source if aaties.df is provided, the column given the alter identifier of the origin of a relation. target if aaties.df is provided, the column given the alter identifier of the destination of a relation.
e.max.alters	Maximum number of alters that are included in edge data.
e.first.var	Index or name of the first column in egos containing edge data.
ego.vars	Character vector naming variables in the egos data, in order to copy them in to the long alters dataframe.
selection	Character naming numeric variable indicating alters selection with zeros and ones.
...	additional arguments to egor().

Value

A list of six objects - the **egoR** object: (1) egos.df: dataframe of all egos and their attributes; (2) alters.df: dataframe of all alters; (3) alters.list: list of dataframes of all alters per ego; (4) edges: list of dataframes with edge lists per network; (5) graphs: list of igraph objects; (6) results: a result dataframe, pre-populated with the network size of each network

vert.attr *Vertex attribute to data frame into graph.*

Description

Extracts a vertex variable from a data.frame and returns it as a vector ready to be set as vertex attribute in a graph. May aggregate the variable by node before returning. This function extracts variables at the author level from 'data' and set them as vertex.attributes in 'graph', a network of authors. The function aggregates the variable by author using the aggregating function FUN. The function needs a 'graph' igraph object of authors in which the UFIDs are recorded as a attribute, and a 'data' data.frame in which the same UFIDs are associated to a variable (attribute). The function takes `data` and aggregates FUN(attribute) by dataID in 'data'. It returns the result of the aggregation in the order of dataID given by V(graph)$name. NOTE that if attribute is categorical (e.g. College), the argument FUN is ignored and vert.attr() is just going to take the category of the variable for authorID (if there are more than 1 category, e.g. more than 1 Academic Units, for the same authorID, vert.attr just takes the 1st category associated to authorID in 'data').

Usage

```
vert.attr(data, attribute, graph, dataID, graphID = "name", FUN = NA,
  unfactor = TRUE)
```

Arguments

`data`	Data frame. The data.frame containing the author attribute variable to be imported in the graph (it must also include a authorID variable).
`attribute`	A variable name in 'data' (as character)
`graph`	Researchers graph where to import the attribute.
`dataID`	Character. The variable in data giving the ID of nodes in the graph (authors)
`graphID`	Character. The vertex attribute of graph that gives node IDs (i.e. IDs of authors) which are the same as dataID.
`FUN`	A function name given as character. The argument is ignored if "data" has one record for each dataID value, i.e. does not need aggregation by dataID. If NA or missing, attribute is aggregated by just taking its 1st non-NA value for each value of dataID.
`unfactor`	Logical. If TRUE, and attribute is character, keeps it from being turned into factor. Notice that this argument has no effect if the attribute is aggregated first. **TODO**: regulate this behavior when attribute is factor (not character) in the first place; regulate this behavior when attribute is aggregated first.

Value

A vector that contains the attribute 'attribute' from data, in the order given by the order of authorID in the graph 'graph'.

See Also

vert.attr.multi

---

vert.attr.multi *Get multiple variables from a data.frame and set them as vertex attributes in a graph. Uses* vert.attr() *in a loop.*

Description

This function repeatedly applies vert.attr() to a set of attributes. The problem that this function solves is that authors (nodes) in the graph and authors (records) in the data.frame data are not necessarily in the same sort order.

Usage

```
vert.attr.multi(data, graph, dataID, graphID = "name", attributes,
  FUN = NA, attr.names = attributes)
```

Arguments

data	The data.frame containing the vertex attribute variables to be imported in the graph (it must also include a authorID variable).
graph	The graph where to import the attribute.
dataID	Character. The ID variable for nodes in data. Must be the name of a variable in data. Will be used as a merge key with graphID.
graphID	Character. The vertex attribute that contains node IDs in graph. Must be the name of a vertex attribute in graph. Will be used as a merge key with dataID.
attributes	Character. Variable names in data.
FUN	Character. Function names given as character vector (the FUN argument to vert.attr()). If the vector is shorter than "attributes", it's recycled. Defaults to NA for all attributes.
attr.names	Character. Names to use for the attributes in the graph. Defaults to attributes.

Value

An igraph object.

Uses

vert.attr

vis_clustered_graphs *Visualize clustered graphs*

Description

vis_clustered_graphs visualizes clustered_graphs using a list of clustered graphs created with clustered_graphs.

Usage

```
vis_clustered_graphs(graphs, node.size.multiplier = 1,
  node.min.size = 0, node.max.size = 200, edge.width.multiplier = 30,
  center = 1, label.size = 0.8, labels = FALSE,
  legend.node.size = 45, pdf.name = NULL, ...)
```

Arguments

graphs	List of graph objects, representing the clustered graphs.
node.size.multiplier	Numeric used to multiply the node diameter of visualized nodes.
node.min.size	Numeric indicating minimum size of plotted nodes
node.max.size	Numeric indicating maximum size of plotted nodes
edge.width.multiplier	Numeric used to multiply the edge width.
center	Numeric indicating the vertex to be plotted in center.
label.size	Numeric.
labels	Boolean. Plots with turned off labels will be preceeded by a 'legend' plot giving the labels of the vertices.
legend.node.size	Numeric used as node diameter of legend graph.
pdf.name	Character giving the name/path of the pdf file to create.
...	Arguments to pass to plot.igraph.

Value

vis_clustered_graphs plots a list of igraph objects created by the clustered_graphs function.

clustered_graphs returns a list of graph objects representing the clustered ego-centered network data;

References

Brandes, U., Lerner, J., Lubbers, M. J., McCarty, C., & Molina, J. L. (2008). Visual Statistics for Collections of Clustered Graphs. 2008 IEEE Pacific Visualization Symposium, 47-54.

See Also

clustered_graphs for creating clustered graphs objects

Examples

```
data("egor32")

# Simplify networks to clustered graphs, stored as igraph objects
graphs <- clustered_graphs(egor32, "country")

# Visualise
vis_clustered_graphs(graphs,
                     node.size.multiplier = 5,
                     edge.width.multiplier = 25,
                     labels = TRUE)
```

---

weights.egor *weights.egor() extracts the (relative) sampling weights of each ego in the dataset.*

Description

weights.egor() extracts the (relative) sampling weights of each ego in the dataset.

Usage

```
## S3 method for class 'egor'
weights(object, ...)
```

Arguments

object	an egor object.
...	arguments to be passed to methods

See Also

weights.survey.design

Index