Package ‘egor’

January 27, 2020

Type Package

Title Import and Analyse Ego-Centered Network Data

Version 0.20.01

Date 2020-01-26

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>).


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 (>= 2.1.0), rmarkdown, haven

VignetteBuilder knitr

RoxygenNote 7.0.2

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 2020-01-27 15:30:08 UTC

R topics documented:

aaties32 ................................. 3
activate.egor ....................................... 3
allbus_2010_simulated .................. 4
alters32 ..................................... 4
alter_design ................................. 5
alts_diversity_count .................... 5
append_egor ................................. 6
as_alters_df ................................. 7
as_igraph ..................................... 7
as_network .................................... 9
clustered_graphs .......................... 9
composition .................................. 11
comp_ei ...................................... 11
comp_ply ..................................... 12
count_dyads .................................. 13
egor ........................................... 14
egor-package-doc .......................... 15
egor32 ........................................ 16
egor_vis_app ................................. 17
egos32 ....................................... 17
ego_constraint .............................. 18
ego_density .................................. 19
ego_design ................................... 19
EI ............................................. 20
gss2004 ....................................... 21
helper ......................................... 22
make_egor ................................. 23
onefile_to_egor ............................ 23
plot_egograms ............................. 25
read_egonet .................................. 27
rowlist ........................................ 28
subset.egor ................................... 29
summary.egor .................................. 30
threefiles_to_egor ........................ 31
trim_aaties ................................. 33
trim_alters ................................... 34	
twofiles_to_egor ............................ 34
vis_clustered_graphs .................... 35
weights.egor .................................. 37

Index 38
**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 level of an ego .dataect

**Description**

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

**Usage**

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

**Arguments**

- **.data** The ego .dataect.
- **what** Character naming the level to activate, this can be "ego", "alter" or "aatie".
**allbus\_2010\_simulated  Simulated Allbus 2010 Data**

**Description**

A dataset simulated based on the original Allbus 2010 SPSS data. The dataset simulates 100 respondents and does not resemble any actual Allbus respondents. Each variable is randomly generated based on the range of the original variables, co-variances between variables are disregarded. The data’s purpose is purely to demonstrate how to technically work with the Allbus data using egor and R - no analytical assumptions should be made based on this data!

**Usage**

allbus\_2010\_simulated

**Format**

A tibble/data.frame of 100 simulated respondents/rows and 981 variables/columns. Each variable is a labelled dbl.

**Details**

The dataset contains two ego-centered name generators.

---

**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

```r
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

```r
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

```r
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

```r
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.alter.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.alter.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.alter.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

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

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

```r
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,
  egoattrs = 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 An egor object.
...
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


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

### 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 ego 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

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

## comp_ei

### 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 ego 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

```r
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

```r
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

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

**Description**

`count_dyads()` counts the attribute combinations of alter-alter ties/dyads in ego-centered networks. The results can be returned as a wide or long tibble/data.frame.

**Usage**

```r
count_dyads(
  object,
  alter_var_name,
  return_as = c("wide", "long"),
  prefix = NULL
)
```

**Arguments**

- **object** An egor object.
- **alter_var_name** Character, naming the alter variable to use as attribute.
- **return_as** Character, either "wide" (default) or "long".
- **prefix** Character, added in front of variables. Only used if `return_as` is "wide". If NULL (default) prefix is automatically generated.

**Value**

Wide or long tibble/data.frame.

**Examples**

```r
data(egor32)
count_dyads(object = egor32,
  alter_var_name = "country")

# Return result as long tibble.
count_dyads(object = egor32,
  alter_var_name = "country",
  return_as = "long")
```
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

```r
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 `NULLs` corresponding to egos with no nominees.

- `egos` a `data.frame` containing the egos.

- `aaties` a `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 if `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

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

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

Description

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

Further Information or GitHub
Thanks to: Martina Morris

Author(s)

Till Krenz, <egor@tillt.net>
Pavel Krivitsky, <pavel@uow.edu.au>
Michal Bojanowski <m.bojanowski at icm.edu.pl>
Andreas Herz, <herzand@uni-hildesheim.de>
Raffaele Vacca, <r.vacca@ufl.edu>
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_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**

```r
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**

```r
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**

```r
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_constraint

Calculate Burt constraint for the egos of ego-centered networks

Description

This calculates Burt’s network constraint for all egos in an egor object. It iterates over each network and applies `igraph::constraint`. A weight variable can be specified.

Usage

```r
data(egor32)
ego_constraint(egor32)
```

Arguments

- `object` An egor object.
- `weights` Character, naming the alter-alter tie weight variable.
- `ego.alter.weights` Character, naming the ego-alter weight tie weight variable. This defaults to the same value as `weights`, only specify if the name of the `ego.alter.weights` is different from `weights`.

Details

The calculation of weighted network constraint only works, if the alter-alter tie weights are complemented by a alter level variable specifying the same weight for the ego-alter ties.

Value

Numeric vector with a constraint value for each ego.

References

ego_density  

Description

This function uses an ego object and calculates the density of all the ego-centered networks listed in the 'egor' object. Instead of an ego 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: An ego object.
- ...: 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 homophily/heterophily - 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


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.
make_eogor

- 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_eogor**

*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**

```r
make_eogor(net.count, max.alters, netsize_fixed = FALSE, plot = FALSE)
```

**Arguments**

- `net.count`: Number of networks/ egos to generate.
- `max.alters`: Maximum size of networks.
- `netsize_fixed`: Logical, if TRUE all networks will have max.alters as network size.
- `plot`: Whether to plot the network size distribution.

---

**onfile_to_eogor**

*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

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

Arguments

egos Data frame containing ego data (egos as cases)
netsize Numeric, network size values are used to filter out empty alter entries. If the alter data is not structured in a way, where valid alters are stored before the invalid alters, pass NULL here and filter out invalid alters afterwards.
ID.vars Character. For onefile_to_egor only the name of the ego ID needs to be provided.
attr.start.col Index or name of the first column containing alter attributes.
attr.end.col Index or name of the last column 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.
var.wise Logical value indicating if the alter attributes are sorted variable wise (defaults to FALSE).
... additional arguments to egor().

Value

An egor object is returned. It is a list of three data frames: (1) ego: dataframe of all egos and their attributes; (2) alter: dataframe of all alters; (3) aatie: dataframe of alter alter ties/edges

References

plot_egograms

Description

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

Usage

plot_egograms(
  x,
  ego_no = 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 = vertex_color_var,
  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",
  resDisp_vars = NULL,
  vertex_zoom = 1,
  edge_zoom = 2,
  font_size = 1,
  venn_colors = NULL,
  show_venn_labels = TRUE,
  ...
)

plot_ego_graphs(
  x,
  ego_no = 1,
  x_dim = 1,
  y_dim = 1,
  vertex_size_var = NULL,
  vertex_color_var = NULL,
  vertex_color_palette = "Heat Colors",
  vertex_color_legend_label = vertex_color_var,
  vertex_label_var = NULL,
  edge_width_var = NULL,
plot_egograms

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

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

Arguments

x  An egor object.
ego_no  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 forwarded to plot.igraph.
include_ego
    Logical.
type
    Character. Either "egograph" or "egogram".

Details

For type equals "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
ego 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**

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

**Value**

An egor object is returned. It is a list of three data frames: (1) ego: dataframe of all egos and their attributes; (2) alter: dataframe of all alters; (3) aatie: dataframe of alter alter ties/ edges

**Examples**

```r
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 = ";")
```

---

**Description**

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

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 \texttt{.altRow} contains the index (counting from 1) of the row number in the alter table.

Alter–alter indices \texttt{.srcRow} and \texttt{.tgtRow} contain the index (counting from 1) of the row of the alter being referenced by \texttt{.srcID} and \texttt{.tgtID}. (This can be used to quickly access the attributes of the alters in question.)

... extra arguments to \texttt{subset} if \texttt{subset} is a function; otherwise unused.

\texttt{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 \texttt{activate.egor()}. 

\texttt{i} numeric or logical vector indexing the appropriate unit.

\texttt{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 \texttt{.egoID}, \texttt{.altID}, \texttt{.srcID}, \texttt{.tgtID} are not indexed by \texttt{j}.

Details

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

Value

An \texttt{egor()} object.

Examples

\begin{verbatim}
# 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)
\end{verbatim}
### Usage

```r
## 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, an alters file and one file containing the edge data. EgoWeb2.0 and openeddi use variations of this format.

#### Usage

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

```r
read_egoweb(
alter.file,
edges.file,
egos.file = NULL,
eggo vars = NULL,
... )
```

```r
read_openeddi(
eggos.file = NULL,
... )
```
threefiles_to_egor

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

**Arguments**

- **egos**
  Data frame containing ego data (egos as cases)

- **alters.df**
  Dataframe containing alters data (alters as cases), alters are separated by a variable containing 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 aatties.df are both lists of data frames.
  - **alter**
    unique-within-ego identifier associated with each alter, defaulting to "alterID"; optional aatties.df are not provided.
  - **source**
    if aatties.df is provided, the column given the alter identifier of the origin of a relation.
  - **target**
    if aatties.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 specifying the filename of the alters data.

- **edges.file**
  A character specifying the filename of the edge data.

- **egos.file**
  A character specifying the filename of the ego data.

- **alters.file**
  Character name of the alters data file.

**Value**

An `egor` object is returned. It is a list of three data frames: (1) ego: dataframe of all egos and their attributes; (2) alter: dataframe of all alters; (3) aattie: dataframe of alter alter ties/edges

**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/qualinitiative/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
**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 ego object.

**Value**

An ego 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**

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

- **egos**: Data frame containing ego data (egos as cases)
- **alters**: Data frame containing alters data (alters as cases), alters are separated by a variable containing an egoID.
- **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.
- **selection**: Character naming numeric variable indicating alters selection with zeros and ones.
- ...: additional arguments to `egor()`.

### Value

An `egor` object is returned. It is a list of three data frames: (1) `ego`: data frame of all egos and their attributes; (2) `alter`: data frame of all alters; (3) `aaties`: data frame of alter alter ties/edges.

---

### Description

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

### Usage

```r
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,
```
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 preceded 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


See Also

- `clustered_graphs` for creating clustered graphs objects

Examples

```r
data("egor32")

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

# Visualise
vis_clustered_graphs(graphs,
```
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**

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

**Arguments**

- **object**: an *egor* object.
- **...**: arguments to be passed to methods

**See Also**

*weights.survey.design*
Index

*Topic analysis
  alts_diversity_count, 5
  clustered_graphs, 9
  comp_ei, 11
  comp_ply, 12
  composition, 11
  ego_density, 19
  egor, 14
  egor_vis_app, 17
  vis_clustered_graphs, 35

*Topic datasets
  aaties32, 3
  allbus_2010_simulated, 4
  alters32, 4
  egor32, 16
  egos32, 17

*Topic ego-centered
  activate.egor, 3
  alts_diversity_count, 5
  clustered_graphs, 9
  comp_ei, 11
  comp_ply, 12
  composition, 11
  ego_density, 19
  egor, 14
  egor_vis_app, 17
  EI, 20
  make_egor, 23
  read_egonet, 27
  vis_clustered_graphs, 35

*Topic import
  onefile_to_egor, 23
  read_egonet, 27
  twofiles_to_egor, 34

*Topic network
  activate.egor, 3
  alts_diversity_count, 5
  clustered_graphs, 9
  comp_ei, 11
  comp_ply, 12
  composition, 11
  ego_density, 19
  egor, 14
  egor_vis_app, 17
  EI, 20
  make_egor, 23
  vis_clustered_graphs, 35

*Topic random
  make_egor, 23

*Topic sna
  EI, 20
  [.egor (subset.egor), 29
  aaties32, 3
  activate.egor, 3
  activate.egor(), 30
  allbus_2010_simulated, 4
  alter_design, 5
  alter_design<-(alter_design), 5
  alters32, 4
  alts_diversity_count, 5
  alts_diversity_entropy
    (alts_diversity_count), 5
  append_cols (append_egor), 6
  append_egor, 6
  append_rows (append_egor), 6
  as.egor (egor), 14
  as.igraph.egor (as_igraph), 7
  as.network.egor (as_network), 9
  as_aaties_df (as_alterns_df), 7
  as_alterns_df, 7
  as_igraph, 7
  as_nested_egor (helper), 22
  as_network, 9
  clustered_graphs, 9, 35, 36
  comp_ei, 11
  comp_ply, 12
  composition, 11
count_dyads, 13
create_edge_names_wide (helper), 22
data.frame(), 28, 29
din_page_dist (helper), 22
dyad.poss (helper), 22
dyads_possible_between_groups (helper), 22
ego_constraint, 18
ego_density, 19
ego_design, 19
ego_design <- (ego_design), 19
egor, 14, 14, 15, 20, 30, 31, 37
egor(), 14, 24, 28–30, 32, 35
egor-package-doc, 15
egor32, 16
egor_vis_app, 17
egos32, 17
EI, 20
gss2004, 21
helper, 22
igraph::constraint, 18
list, 14, 15, 20
list(), 29
make_egor, 23
onefile_to_egor, 23
plot.egor (plot_egograms), 25
plot_ego_graphs (plot_egograms), 25
plot_egograms, 25
plot_egor (plot_egograms), 25
print.egor (summary.egor), 30
read_egonet, 27
read_egoweb (threefiles_to_egor), 31
read_openeddi (threefiles_to_egor), 31
regex, 24
rowlist, 28
sanitize.wide.edges (helper), 22
subset(), 29
subset.egor, 29
summary.egor, 30
survey::svydesign(), 14, 15
svydesign, 19
svydesign(), 20
threefiles_to_egor, 31
tibble, 15
tibble(), 28, 29
trim_aaties, 33
trim_alters, 34
twofiles_to_egor, 34
vis_clustered_graphs, 10, 35
weights.egor, 37
weights.egor(), 37
weights.survey.design, 37
weights.egor()