Package ‘epicontacts’

November 21, 2017

Type Package

Title Handling, Visualisation and Analysis of Epidemiological Contacts

Version 1.1.0

Date 2017-05-11

Description A collection of tools for representing epidemiological contact data, composed of case line lists and contacts between cases. Also contains procedures for data handling, interactive graphics, and statistics.

License GPL (>= 2)

LazyData TRUE

RoxygenNote 6.0.1.9000

Imports grDevices, dplyr, igraph, magrittr, visNetwork, threejs, colorspace

Suggests outbreaks, testthat, covr, shiny, readr, knitr, rmarkdown

VignetteBuilder knitr

URL http://www.repidemicsconsortium.org/epicontacts/

BugReports http://github.com/reconhub/epicontacts/issues

NeedsCompilation no

Author VP Nagraj [aut, cre],
    Thibaut Jombart [aut],
    Nistara Randhawa [aut],
    Bertrand Sudre [aut],
    Finlay Campbell [aut],
    Thomas Crellen [aut]

Maintainer VP Nagraj <vpnagraj@virginia.edu>

Repository CRAN

Date/Publication 2017-11-21 08:24:53 UTC
as.igraph.epicontacts

Create igraph object from contact data

Description

This function creates an igraph object from a given epicontacts object containing a 'contacts' dataframe.

Usage

```r
## S3 method for class 'epicontacts'
as.igraph(x)
```

Arguments

- `x` An epicontacts object.

Value

An igraph object (from the igraph package). Note: any column called "name" in the original linelist will be stored as a new vertex attribute in the igraph object named 'epicontacts_name'. This is due to the inherent behaviour of igraph creating its own 'name' vertex attribute.
Author(s)

Nistara Randhawa (<nrandhawa@ucdavis.edu>)

Examples

```r
if (require(outbreaks) && require(igraph)) {

  ## build data
  x <- make_epicontacts(ebola_sim$linelist, ebola_sim$contacts,
    id = "case_id", to = "case_id", from = "infector",
    directed = TRUE)

  ## subset data - keep 50 cases from linelist with contacts
  ids <- get_id(x, "common")[1:50]
  x <- x[ids, ids]

  ## make igraph object with associated attributes from epicontacts object
  net <- as.igraph(x)
  plot(net, vertex.label = ",", vertex.size = 10,
    vertex.color = cases_pal(50))
}
```

Description

The object `codeawesome` is a character vector of fontawesome codes, named after their aliases.

Usage

codeawesome

Format

An object of class character of length 519.

Author(s)

Thibaut Jombart
get_clusters

Assign cluster IDs to epicontacts data

Description

This function identifies transitive clusters (i.e. connected components) as well as the number of members in each cluster, and adds this information to the linelist data.

Usage

get_clusters(x, output = c("epicontacts", "data.frame"),
               member_col = "cluster_member", size_col = "cluster_size",
               override = FALSE)

Arguments

x
An epicontacts object.

output
A character string indicating the type of output: either an epicontacts object (default) or a data.frame containing cluster memberships to which members of epicontacts linelist belong to.

member_col
Name of column to which cluster membership is assigned to in the linelist. Default name is 'cluster_member'.

size_col
Name of column to which cluster sizes are assigned to in the linelist. Default name is 'cluster_size'.

override
Logical value indicating whether cluster member and size columns should be overwritten if they already exist in the linelist. Default is 'FALSE'.

Value

An epicontacts object whose 'linelist' dataframe contains new columns corresponding to cluster membership and size, or a data.frame containing member ids, cluster memberships as factors, and associated cluster sizes. All ids that were originally in the 'contacts' dataframe but not in the linelist will also be added to the linelist.

Author(s)

Nistara Randhawa (<nrandhawa@ucdavis.edu>)

Examples

if (require(outbreaks)) {
  ## build data
  x <- make_epicontacts(ebola_sim$linelist, ebola_sim$contacts,
                         id = "case_id",
                         to = "case_id",
                         from = "infector",
                         directed = TRUE)
get_degree

## add cluster membership and sizes to epicontacts 'linelist'
y <- get_clusters(x, output = "epicontacts")
y
## return a data.frame with linelist member ids and cluster memberships as
## factors
z <- get_clusters(x, output = "data.frame")
head(z)

get_degree Find node degree for epicontacts objects

Description

This function computes the number of contacts per cases in a epicontacts dataset. Whenever
contacts are directed, the argument 'type' can be used to specify which kind of contact should be
considered: 'in' (towards the case), 'out' (from the case), or 'both'.

Usage

get_degree(x, type = c("in", "out", "both"), only_linelist = FALSE)

Arguments

x an epicontacts object
type the type of degree to be computed (see description); if contacts are not directed,
this will be forced to 'both'	only_linelist a logical indicating if cases whose degree is computed should be from the linelist
exclusively

Author(s)

Thibaut Jombart (<thibautjombart@gmail.com>)

Examples

## make epicontacts object
if (require(outbreaks)) {
x <- make_epicontacts(ebola_sim$linelist, ebola_sim$contacts,
  id="case_id", to="case_id", from="infector",
  directed=TRUE)
x

## compute in-degree
deg_in <- get_degree(x)
get_id

Access unique identifiers in epicontacts objects

description

This accessor is used to extract unique identifiers from epicontacts objects. The argument 'which' can be used to specify if IDs should include: linelist only ('linelist'), contacts only ('contacts'), the union of both ('all'), or the intersection of both ('common'); two additional options are 'from' (ID 'giving' contacts) and 'to' (ID 'receiving' contacts).

Usage

get_id(x, which = c("linelist", "contacts", "all", "common", "from", "to"))

Arguments

x an epicontacts object
which the type of ID to return (see description); value can be 'linelist', 'contacts', 'all', 'common', 'from' or 'to'.

Value

x a character vector of unique identifiers

Author(s)

Thibaut Jombart (<thibautjombart@gmail.com>)

Examples

if (require(outbreaks)) {
  ## build data
  x <- make_epicontacts(ebola_sim$linelist, ebola_sim$contacts,
                        id="case_id", to="case_id", from="infector",
                        directed=TRUE)

  ## get identifiers
  id1 <- get_id(x, "linelist")
  table(deg_in)

  ## compute out-degree
  deg_out <- get_degree(xL "out"
  barplot(table(deg_out), main = "Reproduction number distribution")
  mtext(side = 3, "(based on case out-degree)")

}
get_pairwise

Characterise contacts by comparing case attributes

Description

This function extract attributes of cases involved in contacts using case information provided in the linelist of an epicontacts dataset. If not provided, the function used to process attributes will adjust to the type of attribute selected (see details).

Usage

get_pairwise(x, attribute, f = NULL, hard NA = TRUE)

Arguments

x an epicontacts object
attribute the attribute to be examined between contact pairs
f a function processing the attributes of ‘from’ and ‘to’
hard NA a logical indicating if the output should be NA whenever one of the paired values is NA (TRUE, default); otherwise, ‘NA’ may be treated as another character (e.g. when pasting paired values)

Author(s)

Thibaut Jombart (<thibautjombart@gmail.com>) Tom Crellen (<tomcrellen@gmail.com>)

Examples

if (require(outbreaks)) {
  ## example using mers outbreak in Korea, 2014
  head(mers_korea_2015[[1]])
  head(mers_korea_2015[[2]])

  x <- make_epicontacts(linelist=mers_korea_2015[[1]],
                        contacts=mers_korea_2015[[2]], directed=TRUE)

  ## estimate serial interval (onset->onset)
SI <- get_pairwise(x, "dt_onset")
SI
summary(SI)

hist(SI, col="grey", border="white", xlab="Days after symptoms",
     main="MERS Korea 2014 - Serial Interval")

## check gender mixing:
get_pairwise(x, "sex") # not good, we want 2-way table

generate pairwise(x, "sex", f=table) # use custom function
fisher.test(get_pairwise(x, "sex", f=table)) # test association

---

**graph3D**

*Interactive 3D Force-directed graph from epicontacts object*

**Description**

This function creates a 3D graph from an epicontacts object.

**Usage**

```r
graph3D(x, node_color = "id", annot = TRUE, col_pal = cases_pal,
       NA_col = "lightgrey", g_title = "", bg_col = "white",
       label_col = "darkgrey", node_size = 1, edge_size = 0.5)
```

**Arguments**

- `x`: An epicontacts object.
- `node_color`: An index or character string indicating which field of the linelist should be used to color the nodes. Default is `id`.
- `annot`: An index, logical, or character string indicating which fields of the linelist should be used for annotating the nodes upon mouseover. The default `TRUE` shows the 'id' and 'node_color' (if the grouping column is different from 'id').
- `col_pal`: A color palette for the node_colors.
- `NA_col`: The color used for unknown node_color.
- `g_title`: The title of the graph.
- `bg_col`: The background color of graph.
- `label_col`: The color of the graph title and labels of groups.
- `node_size`: The sizes of graph nodes.
- `edge_size`: The width of graph edges.

**Value**

An htmlwidget object that is displayed using the object’s show or print method. (If you don’t see your widget plot, try printing it with the print function.)
Note

All colors must be specified as color names like "red", "blue", etc. or as hexadecimal color values without opacity channel, for example "#FF0000", "#0a3e55" (upper or lower case hex digits are allowed).

Double-click or tap on the plot to reset the view.

Author(s)

Nistara Randhawa (<nrandhawa@ucdavis.edu>) Thibaut Jombart (<thibautjombart@gmail.com>)
VP Nagraj (<vpnagraj@virginia.edu>)

References


Examples

```r
if (require(outbreaks)) {

## example using MERS outbreak in Korea, 2014
head(mers_korea_2015[['1']])
head(mers_korea_2015[['2']])

x <- make_epicontacts(linelist = mers_korea_2015$linelist,
                        contacts = mers_korea_2015$contacts,
                        directed = FALSE)

## Not run:
graph3D(x)
graph3D(x, annot = FALSE)
graph3D(x, node_color = "sex", g_title = "MERS Korea 2014")
graph3D(x, node_color = "sex", annot = c("sex", "age"),
        g_title = "MERS Korea 2014")

## End(Not run)
}
```

make_epicontacts  
Read linelist and contact data

Description

This function reads data stored as data.frame containing linelist (case information, where each row corresponds to a unique patient), and contacts between patients. Common identifiers should be used in the two data sources for matching to be achieved.
Usage

make_epicontacts(linelist, contacts, id = 1L, from = 1L, to = 2L, directed = FALSE)

Arguments

linelist: a data.frame with at least one column providing unique patient identifiers
contacts: a data.frame that needs at least two columns indicating patients between which cases take place; these need not be referenced in the linelist
id: an index or name indicating which column in linelist contains unique identifiers; default is first column in linelist data frame
from: an index or name indicating which column in contacts contains the first case of a contact
to: an index or name indicating which column in contacts contains the second case of a contact
directed: a logical indicating if contacts are directed or not; default is FALSE but note that contacts will be indicated as 'from' and 'to' even in non-directed contacts

Details

An epicontacts object can be created from two components:

• a linelist provided as a data.frame where columns are different variables describing cases, and where each row is a different case, and a contact list.

• a contact list provided as a data.frame where each row contains unique pairs of contacts with unique features of contact in columns. The line list and contact list should share an identification scheme for individuals.

Value

An epicontacts object in list format with three elements:

• linelist: data.frame of cases with first column 'id' containing character vector of unique identifiers

• contacts: data.frame of contacts with first two columns named 'from' and 'to' indicating unique pairs of contact between individuals

• directed: indicator as to whether or not the contacts are to be considered directed or not

Author(s)

Thibaut Jombart (<thibautjombart@gmail.com>)

References

http://foodborne.unl.edu/public/role/epidemiologist/lineLists.html


Examples

```r
if (require(outbreaks)) {
  ## make epicontacts object from simulated Ebola data
  x <- make_epicontacts(ebola_sim$linelist, ebola_sim$contacts)

  ## test reordering of columns
  linelist <- ebola_sim$linelist[,rev(seq_len(ncol(ebola_sim$linelist)))]
  contacts <- ebola_sim$contacts[,rev(seq_len(ncol(ebola_sim$contacts)))]
  head(linelist)
  head(contacts)

  ## make object
  x <- make_epicontacts(linelist, contacts, id = "case_id",
                        to = "case_id", from = "infector")
  head(x$linelist)
  head(x$contacts)
}
```

plot.epicontacts  

Plot epicontacts objects

Description

This function plots epicontacts objects using various approaches. The default method uses vis_epicontacts.

Usage

```r
## S3 method for class 'epicontacts'
plot(x, node_color = "id", method = c("visNetwork", "graph3D"), thin = TRUE, ...)
```

Arguments

- `x`  An epicontacts object
- `node_color`  An integer or a character string indicating which attribute column in the linelist should be used to color the nodes.
- `method`  A character string indicating the plotting method to be used; available values are "visNetwork" and "graph3D"; see details.
- `thin`  A logical indicating if the data should be thinned so that only cases with contacts should be plotted.
- `...`  Further arguments passed to the plotting methods.
print.epicontacts

Details

This function is merely a wrapper for other plotting functions in the package, depending on the value of method:

- visNetwork: calls the function vis.epicontacts
- graph3D: calls the function graph3D

Author(s)

Thibaut Jombart (<thibautjombart@gmail.com>)

See Also

vis.epicontacts, which uses the package visNetwork, and codeawesome for icon codes.

Examples

```r
if (require(outbreaks)) {
  ## example using MERS outbreak in Korea, 2014
  head(mers_korea_2015[[1]])
  head(mers_korea_2015[[2]])

  x <- make_epicontacts(linelist = mers_korea_2015[[1]],
                        contacts = mers_korea_2015[[2]], directed=TRUE)

  ## Not run:
  plot(x)
  plot(x, "place_infect")
  plot(x, "loc_hosp", legend_max = 20, annot = TRUE)
  plot(x, "place_infect", node_shape = "sex",
       shapes = c(M = "male", F = "female"))
  plot(x, 4)
  plot(x, 4, method = "graph3D")

  ## End(Not run)
}
```

print.epicontacts  

Print method for epicontacts objects

Description

This method prints the content of epicontacts objects, giving a brief summary of the reported cases and contacts.

Usage

```r
## S3 method for class 'epicontacts'
print(x, ...)
```
print.summary_epicontacts

Arguments

x               an epicontacts object
...             further parameters to be passed to other methods (currently not used)

Author(s)

Thibaut Jombart (<thibautjombart@gmail.com>)

print.summary_epicontacts

Print method for summary_epicontacts objects

Description

This method outputs a printed summary of the content of summary_epicontacts objects.

Usage

## S3 method for class 'summary_epicontacts'
print(x, ...)

Arguments

x               a summary_epicontacts object
...             further parameters to be passed to other methods (currently not used)

Author(s)

VP Nagraj (<vpnagraj@virginia.edu>)

subset.epicontacts

Subset an epicontact object by factors

Description

This function subsets an epicontacts object based on node, edge and/or cluster attributes. Specifying node attributes will return an epicontacts object containing only individuals with these given attributes in the linelist. Specifying edge attributes will return contacts with the attributes provided. Specifying cluster attributes will return clusters of connected cases, and can be defined by ids (returning clusters of cases connected to specified cases) or cluster sizes (returning cluster of a specific, minimum or maximum size).
Usage

```r
## S3 method for class 'epicontacts'
subset(x, node_attribute = NULL,  
    edge_attribute = NULL, cluster_id = NULL, cs = NULL, cs_min = NULL, 
    cs_max = NULL, ...)  
```

Arguments

- `x`: an epi_contact object to be subsetted
- `node_attribute`: a named list defining the node attribute name and node attribute value (as a single value or vector of values). Dates must be provided as a vector of date objects, defining the range of dates included in the subset. If only one date is provided, only node attributes with that date will be returned.
- `edge_attribute`: a named list defining the edge attribute name and edge attribute value (as a single value or vector of values). Dates must be provided as a vector of date objects, defining the range of dates included in the subset. If only one date is provided, only edge attributes with that date will be returned.
- `cluster_id`: a character vector of case identifiers; the connected components attached to these cases will be retained in the output object.
- `cs`: cluster size to be used for subsetting
- `cs_min`: minimum cluster size for subsetting
- `cs_max`: maximum cluster size for subsetting
- `...`: further arguments passed on to other methods

Author(s)

Finlay Campbell (<f.campbell15@imperial.ac.uk>), Nistara Randhawa (<nrandhawa@ucdavis.edu>)

Examples

```r
if (require(outbreaks)) {
  ## build data
  x <- make_epicontacts( 
    ebola_sim$linelist, ebola_sim$contacts, 
    id = "case_id", to = "case_id", 
    from = "infector", directed = FALSE)

  ## subset based on node and edge attributes
  x_subset <- subset(x, node_attribute = list("gender" = "f"), 
                     edge_attribute = list("source" = "funeral"))

  ## subset a cluster connected to a given id
  ## (can be a vector of ids as well)
  ## here picking node with highest out-degree

  id <- names(which.max(get_degree(x, "out")))
  x_subset <- thin(subset(x, cluster_id = id), 2)
}  
```
## subset_clusters_by_id

```r
subset_clusters_by_id
```

### Description

This function subsets an `epicontacts` object by identifying clusters of cases connected to specified cases.

### Usage

```r
subset_clusters_by_id(x, id)
```

### Arguments

- **x**: an `epicontacts` object
- **id**: a character vector of case identifiers; the connected components attached to these cases will be retained in the output object.

### Value

An `epicontacts` object whose contact dataframe corresponds to all clusters containing specified case id.
Author(s)
Nistara Randhawa (<nrandhawa@ucdavis.edu>)

Examples

```r
if (require(outbreaks)) {
  ## build data
  x <- make_epicontacts(ebola_sim$linelist, ebola_sim$contacts,
                        id="case_id", to="case_id", from="infector",
                        directed=TRUE)

  id <- "cac51e" ## it can be a vector of id as well

  ## subset based on cluster to which "cac51e" belongs
  x_subset <- subset_clusters_by_id(x, id)
}
```

subset_clusters_by_size

*Subset clusters from epicontacts object by cluster size*

Description

This function subsets an *epicontacts* object based on defined size(s) of clusters (clusters being groups of connected individuals/nodes). Subsetting may be done by specifying a particular cluster size of interest, minimum cluster size, maximum cluster size, or a range (minimum and maximum) of cluster sizes.

Usage

```r
subset_clusters_by_size(x, cs = NULL, cs_min = NULL, cs_max = NULL)
```

Arguments

- `x`: an *epicontacts* object
- `cs`: cluster size to be used for subsetting
- `cs_min`: minimum cluster size for subsetting
- `cs_max`: maximum cluster size for subsetting

Value

An *epicontacts* object whose contact dataframe corresponds to all clusters of specified cluster sizes.


**summary.epicontacts**

**Author(s)**

Nistara Randhawa (<nrandhawa@ucdavis.edu>)

**Examples**

```r
if (require(outbreaks)) {
  ## build data
  x <- make_epicontacts( 
    ebola_sim$linelist, ebola_sim$contacts, 
    id="case_id", to="case_id", from="infector", 
    directed=TRUE)

  ## subset based on cluster size range
  x_subset <- subset_clusters_by_size(x, cs_min = 12, cs_max = 15)

  ## subset based on single cluster size
  x_subset <- subset_clusters_by_size(x, cs = 12)

  ## subset based on minimum cluster size
  x_subset <- subset_clusters_by_size(x, cs_min = 10)

  ## subset based on maximum cluster size
  x_subset <- subset_clusters_by_size(x, cs_max = 9)
}
```

---

**summary.epicontacts**  
*Summary method for epicontacts objects*

**Description**

This method outputs a summary of the content of epicontacts objects.

**Usage**

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

**Arguments**

- `object` an epicontacts object
- `...` further parameters to be passed to other methods (currently not used)

**Author(s)**

VP Nagraj (<vpnagraj@virginia.edu>)
Thin data to retain matching linelist / contacts

Description

This function can be used to remove ('thin') data from epicontacts objects to ensure stricter matching of linelists and contacts. It has two behaviours, triggered by the argument what: either it thins data from linelist, keeping only cases that are in contacts (thin = "linelist", default), or the converse, i.e. removing contacts which are not fully documented in the linelist.

Usage

thin(x, what = "linelist")

Arguments

x
An epicontacts object.

what
A character string or integer determining which type of data is removed ('thinned'). "linelist" / 1 indicates that only cases appearing in contacts are kept in linelist. "contacts / 2" indicates that only cases appearing in linelist are kept in contacts.

Author(s)

Thibaut Jombart (<thibautjombart@gmail.com>)

Examples

if (require(outbreaks)) {
  ## build data
  x <- make_epicontacts(ebola_sim$linelist, ebola_sim$contacts,
                       id = "case_id", to = "case_id", from = "infector",
                       directed = TRUE)

  ## keep contacts from a specific case '916d0a'
  x <- x[j = "916d0a", contacts = "from"]

}
transp

---

transp

*Color tools and palettes for epicontacts*

---

**Description**

These functions are used for defining palettes or colors in the *epicontacts* package. They include:

**Usage**

```r
transp(col, alpha = 0.5)
edges_pal(n)
cases_pal(n)
spectral(n)
fac2col(x, pal = cases_pal, NA_col = "lightgrey", legend = FALSE)
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>col</td>
<td>A color vector to which transparency should be added.</td>
</tr>
<tr>
<td>alpha</td>
<td>The threshold to be used for transparency: 0 for full transparency, and 1 for full opacity.</td>
</tr>
<tr>
<td>n</td>
<td>An integer indicating the number of colors.</td>
</tr>
<tr>
<td>x</td>
<td>A character or a factor to be converted to colors.</td>
</tr>
<tr>
<td>pal</td>
<td>A color palette.</td>
</tr>
<tr>
<td>NA_col</td>
<td>The color to be used for NA values.</td>
</tr>
<tr>
<td>legend</td>
<td>A logical indicating if legend info should be added to the output. If TRUE, the output will be a list, with colors in the $color component.</td>
</tr>
</tbody>
</table>

**Details**

- *cases_pal*: discrete color palette used for cases (comes from the *dibbler* package)
- *spectral*: continuous color palette (comes from the *adegenet* package)
- *transp*: makes colors transparent (comes from the *adegenet* package)
- *fac2col*: translates a character or a factor to a color using a palette (comes from the *adegenet* package)

**Author(s)**

Thibaut Jombart <thibautjombart@gmail.com>
Examples

```r
barplot(1:5, col = cases_pal(5))
barplot(1:50, col = cases_pal(50))
```

---

**vis_epicontacts**  
*Plot epicontacts objects using visNetwork*

### Description

This function plots `epicontacts` objects using the `visNetwork` package. The produced object is an `htmlwidget` which will need rendering within a web browser.

### Usage

```r
vis_epicontacts(x, node_color = "id", label = "id", annot = TRUE,
               node_shape = NULL, shapes = NULL, edge_label = NULL,
               edge_color = NULL, legend = TRUE, legend_max = 10,
               col_pal = cases_pal, NA_col = "lightgrey", edge_col_pal = edges_pal,
               width = "90E", height = "700px", selector = TRUE, editor = FALSE,
               edge_width = 3, ...)
```

### Arguments

- **x**  
  An `epicontacts` object.
- **node_color**  
  An index or character string indicating which field of the linelist should be used to color the nodes.
- **label**  
  An index, logical, or character string indicating which fields of the linelist should be used for labelling the nodes. Logical will be recycled if necessary, so that the default `TRUE` effectively uses all columns of the linelist.
- **annot**  
  An index, logical, or character string indicating which fields of the linelist should be used for annotating the nodes. Logical will be recycled if necessary, so that the default `TRUE` effectively uses all columns of the linelist.
- **node_shape**  
  An index or character string indicating which field of the linelist should be used to determine the shapes of the nodes.
- **shapes**  
  A named vector of characters indicating which icon code should be used for each value `node_shape`, e.g. `c(m = "male", f = "female")` if 'm' and 'f' are values from `node_shape`. See `codeawesome` for all available codes.
- **edge_label**  
  An index or character string indicating which field of the contacts data should be used to label the edges of the graph.
- **edge_color**  
  An index or character string indicating which field of the contacts data should be used to color the edges of the graph.
- **legend**  
  A logical indicating whether a legend should be added to the plot.
legend_max  The maximum number of groups for a legend to be displayed.
col_pal    A color palette for the nodes.
NA_col     The color used for unknown group.
edge_col_pal A color palette for the edges.
width      The width of the output, in html compatible format (e.g. ‘90%’ or ‘800px’).
height     The height of the output, in html compatible format (e.g. ‘800px’).
selector   A logical indicating if the selector tool should be used; defaults to TRUE.
editor     A logical indicating if the editor tool should be used; defaults to FALSE.
edge_width An integer indicating the width of the edges. Defaults to 3.
...        Further arguments to be passed to visNetwork.

Value
The same output as visNetwork.

Author(s)
Thibaut Jombart (<thibautjombart@gmail.com>) VP Nagraj (<vpnagraj@virginia.edu>)

See Also
visNetwork in the package visNetwork.

Examples
if (require(outbreaks)) {

## example using MERS outbreak in Korea, 2014
head(mers_korea_2015[[1]])
head(mers_korea_2015[[2]])

x <- make_epicontacts(linelist=mers_korea_2015[[1]],
                      contacts = mers_korea_2015[[2]],
                      directed=TRUE)

## Not run:
plot(x)
pplot(x, node_color = "place_infect")
pplot(x, node_color = "loc_hosp", legend_max=20, annot=TRUE)
pplot(x, "place_infect", node_shape = "sex",
       shapes = c(M = "male", F = "female"))
pplot(x, "sex", node_shape = "sex", shapes = c(F = "female", M = "male"),
       edge_label = "exposure", edge_color = "exposure")

## End(Not run)
}
Subset epicontacts objects based on case identifiers

Description

The "[" operator can be used to subset epicontacts objects, retaining a specified set of case identifiers (i for the linelist, j for contacts). Note that unlike most classical R objects, there is no replacement method for epicontacts objects, i.e. no operations such as foo[i] <- bar.

Usage

## S3 method for class 'epicontacts'
x[i, j, k = TRUE, l = TRUE, contacts = c("both", "either", "from", "to"), ...]

Arguments

- x: An epicontacts object
- i: A character vector containing case ID to be retained in the linelist; alternatively, an integer or logical vector used to subset the rows of the $linelist component.
- j: A character vector containing case ID to be retained in the contacts; alternatively, an integer or logical vector used to subset the rows of the $contacts component.
- k: An integer, logical, or character vector subsetting the supplementary columns of x$linelist, i.e. the columns after 'id'; i.e. k=1 refers to the column immediately after 'id'.
- l: An integer, logical, or character vector subsetting the supplementary columns of x$contacts, i.e. the columns after 'from' and 'to'; i.e. l=1 refers to the column immediately after 'to'.
- contacts: A character string indicating the rules for retaining contacts when j indicates case IDs (see details).
- ...: Not used (there for compatibility with generic).

Details

Details on the 'contacts' argument; possible values are:

- 'both': contacts are retained only if both cases are in j
- 'either': contacts are retained if at least one of the cases is in j
- 'from': contacts are retained only if the source ('from') is in j
- 'to': contacts are retained only if the recipient ('to') is in j

Author(s)

Thibaut Jombart (<thibautjombart@gmail.com>)
See Also

`thin` to retain matching cases in linelist or contacts.

Examples

```r
if (require(outbreaks)) {
## build data
x <- make_epicontacts(ebola_sim$linelist, ebola_sim$contacts,
  id = "case_id", to = "case_id", from = "infector",
  directed = TRUE)

## subset first 10 linelist cases
x[1:10]

## same, remove contacts
x[1:10, j = FALSE]

## subset first 10 contacts
x[j = 1:10]

## remove the metadata
x[k = FALSE, j = FALSE]

## keep contacts where both cases are in linelist
x[j = get_id(x, "linelist"), contacts = "both"]

## keep contacts from a specific case '916d0a'
x[j = "916d0a", contacts = "from"]

## more complex: keep all cases and contacts with > 4 secondary contacts
## i) find cases to keep
temp <- table(x$contacts$from)
temp[temp > 4]
to.keep <- names(temp)[temp > 4]
to.keep

## subset the contacts
y <- x[j = to.keep, contacts = "either"]
y

## keep only relevant entries in the linelist
y <- thin(y)

## visualise result
plot(y)
}
```
Index

*Topic **datasets**
  codeawesome, 3
  [.epicontacts, 22
  as.igraph.epicontacts, 2
  cases_pal (transp), 19
codeawesome, 3, 12, 20
data.frame, 4, 10
dataset_pal (transp), 19
epiprints, 2, 4–8, 11, 13, 15–18, 20, 22
epiprints (make_epicontacts), 9
fac2col (transp), 19
get_clusters, 4
get_degree, 5
get_id, 6
get_pairwise, 7
graph3D, 8, 12
make_epicontacts, 9
plot.epicontacts, 11
print.epicontacts, 12
print.summary.epicontacts, 13
spectral (transp), 19
subset.epicontacts, 13
subset_clusters_by_id, 15
subset_clusters_by_size, 16
summary.epicontacts, 17
summary.epicontacts, 13
summary.epicontacts
  (summary.epicontacts), 17
thin, 18, 23
transp, 19
vis.epicontacts, 11, 12, 20
visNetwork, 21

24