Package ‘netmap’

October 13, 2022

Title Represent Network Objects on a Map
Version 0.1.1
Description Represent ‘network’ or ‘igraph’ objects whose vertices can be represented by features in an ‘sf’ object as a network graph surmising a ‘sf’ plot. Fits into ‘ggplot2’ grammar.
License GPL (>= 3)
Encoding UTF-8
RoxygenNote 7.1.2
URL https://github.com/artod83/netmap
BugReports https://github.com/artod83/netmap/issues
Imports ggnetwork, igraph, network, rlang, sf, sna
Suggests rmarkdown, knitr, testthat (>= 3.0.0)
Config/testthat/edition 3
VignetteBuilder knitr
Depends R (>= 2.10)
LazyData true
NeedsCompilation no
Author Matteo Dimai [aut, cre] (<https://orcid.org/0000-0003-1126-5234>)
Maintainer Matteo Dimai <matteo.dimai@phd.units.it>
Repository CRAN
Date/Publication 2022-03-18 16:00:02 UTC

R topics documented:

  check_network_sf ......................................................... 2
  fvgmap ................................................................. 3
  ggcentrality ............................................................ 3
  ggconn_area ............................................................. 5
  ggnetmap ............................................................... 6
  is_lookup_table ......................................................... 7
check_network_sf

Internal checks before ggnetmap and ggcentrality

Description

Checks whether the proper packages are installed, whether the parameters are of the proper classes, whether the network-map link is possible, then performs the link.

Usage

check_network_sf(n, m, lkp = NULL, m_name = NULL, n_name = "vertex.names")

Arguments

n  A network or igraph object.

m  A sf object.

lkp  An optional lookup table.

m_name  Optional character, name of field in m and of column in lkp.

n_name  Optional character, name of vertex attribute in n and of column in lkp.

Value

A list with a network or igraph object with only the vertices present in the sf object as the first element and a list with two vectors, one of features in m present both in the lookup table and in n, the other of nodes in n present both in the lookup table and in m.
**fvmap**

*Map of municipality borders in the Friuli Venezia Giulia region, Italy*

**Description**

An sf object containing the ISTAT municipality codes, geometry and the municipality names in the Friuli Venezia Giulia region in northeastern Italy, based on official ISTAT shapefiles.

**Usage**

fvmap

**Format**

An sf object with 215 features and 6 fields:

- **Cod_reg** region code, always =6 (Friuli Venezia Giulia)
- **Cod_pro** province code (93=Pordenone, 30=Udine, 31=Gorizia, 32=Trieste)
- **Pro_com** municipality code, consists of province code + progressive code of the municipality within the province
- **Shape_leng** length of municipality perimeter
- **Shape_area** municipality area
- **geometry** a MULTIPOLYGON

**Source**

https://www.istat.it/it/archivio/104317

---

**ggcentrality**

*Calculate centrality indices for vertices linked to a sf object*

**Description**

Given a sf object with features that can be linked to a network or igraph object, obtain centrality indices for linked features.
Usage

```
rgcentrality(
    n,
    m,
    lkp = NULL,
    m_name = NULL,
    n_name = "vertex.names",
    par.deg = NULL,
    par.bet = NULL,
    par.clo = NULL
)
```

Arguments

- `n`: A network or igraph object.
- `m`: A sf object.
- `lkp`: An optional lookup table.
- `m_name`: Optional character, name of field in `m` and of column in `lkp`.
- `n_name`: Optional character, name of vertex attribute in `n` and of column in `lkp`.
- `par.deg`: List with additional optional parameters to functions `degree` or `degree`.
- `par.bet`: List with additional optional parameters to functions `betweenness` or `betweenness`.
- `par.clo`: List with additional optional parameters to functions `closeness` or `closeness`.

Value

An sf object, input `m` with added columns for centrality indices (degree, betweenness, closeness; existing columns with the same name will be overwritten) and with only the features linked to vertices in input `n`.

Examples

```r
net=network::network(matrix(c(0, 1, 1, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0), nrow=4, byrow=TRUE))
network::set.vertex.attribute(net, "name", value=c("a", "b", "c", "d"))
wkb = structure(list("010100002040710000000000000001A06410000000000AC5C1641",
                      "010100002040710000000000000001A06410000000000AC5C1441",
                      "010100002040710000000000000001A06410000000000AC5C1241",
                      "010100002040710000000000000001A02410000000000AC5C1841"), class = "WKB")
map=sf::st_sf(id=c("a1", "b2", "c3", "d4"), sf::st_as_sfc(wkb, EWKB=TRUE))
lkptbl=data.frame(id=c("a1", "b2", "c3", "d4"), name=c("a", "b", "c", "d"))
netmap::ggcentrality(net, map, lkptbl, "id", "name")
```
**ggconn_area**

*Calculate connectedness to a specific vertex for vertices linked to a sf object*

**Description**

Given a sf object with features that can be linked to a network or igraph object and given a node with id `id` in said graph that can be linked to the sf object, obtain an indicator variable denoting, for each node, a connection to id.

**Usage**

```r
ggconn_area(n, m, id, lkp = NULL, m_name = NULL, n_name = "vertex.names")
```

**Arguments**

- `n` A network or igraph object.
- `m` A sf object.
- `id` The identifier (as vertex attribute `n_name` of object `n`) of the feature that needs to be checked for connections.
- `lkp` An optional lookup table.
- `m_name` Optional character, name of field in `m` and of column in `lkp`.
- `n_name` Optional character, name of vertex attribute in `n` and of column in `lkp`.

**Value**

An sf object, input `m` with an added column `conn_area` with an indicator variable set to 1 if the feature is connected to the feature with vertex id `id`, 0 otherwise. In directed graphs, only outgoing links are considered a connection. Any existing column with the same name will be overwritten, the result will contain only the features linked to vertices in input. If the vertex `id` is not present in object `n`, `conn_area` will be set to 0 for all vertices.

**Examples**

```r
net=network::network(matrix(c(0, 1, 1, 1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0), nrow=4, byrow=TRUE))
network::set.vertex.attribute(net, "name", value=c("a", "b", "c", "d"))
wkb = structure(list("01010000204071000000000000000001064100000000AC5C1641",  
"01010000204071000000000000000001064100000000AC5C1441",  
"01010000204071000000000000000001064100000000AC5C1241",  
"01010000204071000000000000000001064100000000AC5C1841"), class = "WKB")
map=sf::st_sf(id=c("a1", "b2", "c3", "d4"), sf::st_as_sfc(wkb, EWKB=TRUE))
lkptbl=data.frame(id=c("a1", "b2", "c3", "d4"), name=c("a", "b", "c", "d"))
ggconn_area(net, m, id, lkp=lkptbl, n_name = "name")
```
**ggnetmap**  
*Fortify a network over a map*

**Description**

Link a network or igraph and a sf object in a data.frame for subsequent representation on a plot using ggplot2.

**Usage**

```r
ggnetmap(
  n,  
  m,  
  lkp = NULL,  
  m_name = NULL,  
  n_name = "vertex.names",  
  scale = FALSE,  
  ...  
)
```

**Arguments**

- `n` A network or igraph object.
- `m` A sf object.
- `lkp` An optional lookup table.
- `m_name` Optional character, name of field in `m` and of column in `lkp`.
- `n_name` Optional character, name of vertex attribute in `n` and of column in `lkp`.
- `scale` Whether coordinates should be scaled (defaults to FALSE since the network should be overlayed with the non-scaled sf object).
- `...` Additional parameters passed to `fortify`.

**Details**

Using a network or igraph and a sf object as inputs, with an optional lookup table (a data.frame) in case the IDs don’t match, produces a data.frame that can be used with ggnetwork’s `geom_edges` and `geom_nodes` functions to represent the network as overlayed on a sf object in a ggplot2 graph. Only vertices with a corresponding feature in the sf object are included.

**Value**

A data frame, produced by `fortify`, which can be used as data source in ggplot2 graphs.
is_lookup_table

Examples

```r
net = network::network(matrix(c(0, 1, 1, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0), nrow=4, byrow=TRUE))
network::set.vertex.attribute(net, "name", value=c("a", "b", "c", "d"))
wkb = structure(list("010100002040710000000000000081A064100000000AC5C1641",
"010100002040710000000000000081A084100000000AC5C1441",
"010100002040710000000000000081A044100000000AC5C1241",
"010100002040710000000000000081A024100000000AC5C1841"), class = "WKB")
map = sf::st_sf(id=c("a1", "b2", "c3", "d4"), sf::st_as_sfc(wkb, EWKB=TRUE))
lkptbl = data.frame(id=c("a1", "b2", "c3", "d4"), name=c("a", "b", "c", "d"))
ggnetmap(net, map, lkptbl, "id", "name")
```

**is_lookup_table**  
Is data frame a lookup table?

**Description**

Checks whether a data.frame is a valid lookup table.

**Usage**

```r
is_lookup_table(lkp, m_name = NULL, n_name = NULL)
```

**Arguments**

- `lkp`: A data.frame.
- `m_name`: Optional, a character string with the name of the column in `lkp` to check against `m`.
- `n_name`: Optional, a character string with the name of the column in `lkp` to check against `n`.

**Value**

FALSE on error, a vector with `m_name` and `n_name` if the lookup table is valid.

**is_network**  
Is object a network?

**Description**

Checks whether an object is a network object or an igraph object, returns message if it's not.

**Usage**

```r
is_network(n)
```
Arguments
n
Object of class network or igraph.

Value
TRUE if object of class network, FALSE otherwise.

is_sf
Is object a map?

Description
Checks whether an object is an sf object, returns message if it’s not

Usage
is_sf(m)

Arguments
m
Object of class sf.

Value
TRUE if object of classes sf and data.frame, FALSE otherwise.

link_network_map
Link a network and a map

Description
Checks which vertices of a network object can be represented with features of a sf object.

Usage
link_network_map(m, n, m_name, n_name = "vertex.names")

Arguments
m
Object of class sf.
n
Object of class network or igraph.
m_name
Name of the map field to use for the link.
n_name
Name of the vertex attribute to use for the link, defaults to vertex.names.

Value
On success a list with two vectors, one of features in m present in n, the other of nodes in n present in m, -1 on error.
**link_network_map2**  
*Link a network and a map with a lookup table*

**Description**
Checks which vertices of a network object can be represented with features of a sf object with a lookup table.

**Usage**

```r
link_network_map2(m, n, lkp, m_name = NULL, n_name = NULL)
```

**Arguments**
- `m` Object of class `sf`.
- `n` Object of class `network` or `igraph`.
- `lkp` Lookup table, a `data.frame`.
- `m_name` Optional character, name of field in `m` and of column in `lkp` (first column of `lkp` is used if `NULL`).
- `n_name` Optional character, name of vertex attribute in `n` and of column in `lkp` (second column of `lkp` is used if `NULL`).

**Value**
On success a list with two vectors, one of features in `m` present both in the lookup table and in `n`, the other of nodes in `n` present both in the lookup table and in `m`, -1 on error.

**netmap**
*netmap: Plot network and igraph objects on a sf map using ggplot2*

**Description**
The netmap package extends the ggnetwork package by providing functions to plot networks, with vertices usually representing objects with a spatial attribute (cities, regions, countries, users with location data attached etc.) on a map, provided by a sf object (which in turn is able to represent more or less all spatial data available). Networks and maps need not have the same set of elements: if they don’t, only the intersection will be represented.

**netmap functions**
The main function is `ggnetmap`, which produces a `data.frame` that is then used as data within ggplot2 calls. For those wishing to use the `plot.network` or the `plot.igraph` function to plot the network (without overlaying it on an sf object), both a custom layout function, `network.layout.extract_coordinates`, and a wrapper that provides convenient manipulation of network and sf objects, `netmap_plot`, are available.
netmap_plot

Description

Wrapper for plot.network and plot.igraph using a custom network layout that extracts coordinates of centroids from a sf object. Only vertices with a corresponding feature are plotted.

Usage

netmap_plot(n, m, lkp = NULL, m_name = NULL, n_name = "vertex.names", ...)

Arguments

n  A network or igraph object.
m  A sf object.
lkp  An optional lookup table.
m_name  Optional character, name of field in m and of column in lkp.
n_name  Optional character, name of vertex attribute in n and of column in lkp.
...  Additional parameters passed to plot.network.

Value

A plot of the network.

Examples

net=network::network(matrix(c(0, 1, 1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0), nrow=4, byrow=TRUE))
network::set.vertex.attribute(net, "name", value=c("a", "b", "c", "d"))
wkb = structure(list("01010000204071000000000000000081A064100000000AC5C1641", "01010000204071000000000000000081A064100000000AC5C1441", "01010000204071000000000000000081A044100000000AC5C1241", "01010000204071000000000000000081A024100000000AC5C1841"), class = "WKB")
map=sf::st_sf(id=c("a1", "b2", "c3", "d4"), sf::st_as_sfc(wkb, EWKB=TRUE))
lkptbl=data.frame(id=c("a1", "b2", "c3", "d4"), name=c("a", "b", "c", "d"))
netmap::netmap_plot(net, map, lkptbl, "id", "name")
network.layout.extract_coordinates

*Layout of a network based on a sf object*

**Description**

Custom layout for `plot.network`, extracting coordinates of vertices from a sf object. Its result can be used by `plot.igraph` as well.

**Usage**

```r
network.layout.extract_coordinates(n, layout.par)
```

**Arguments**

- **n**: A network or igraph object. Not used, only for compatibility with `plot.network`.
- **layout.par**: A list of layout parameters (the only one implemented is `layout.par$sf`, an sf object whose rows match the order of vertices in `n`).

**Value**

A matrix whose rows contain the x,y coordinates of the vertices of `n`.

**Examples**

```r
net = network::network(matrix(c(0, 1, 1, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0), nrow=4, byrow=TRUE))
network::set.vertex.attribute(net, "name", value=c("a", "b", "c", "d"))
wkb = structure(list("01010000204071000000000000000001A06410000000000AC5C1641", "01010000204071000000000000000001A08410000000000AC5C1441", "01010000204071000000000000000001A04410000000000AC5C1241", "01010000204071000000000000000001A0410000000000AC5C1841"), class = "WKB")
map = sf::st_sf(id=c("a1", "b2", "c3", "d4"), sf::st_as_sfc(wkb, EWKB=TRUE))
ltkpltb = data.frame(id=c("a1", "b2", "c3", "d4"), name=c("a", "b", "c", "d"))
netmap::network.layout.extract_coordinates(net, list(sf=map))
```

---

reduce_to_map

*Reduces network to vertices present on the map*

**Description**

Removes vertices from a network or igraph object which are not present in the link vector produced by `link_network_map` or `link_network_map2`.

**Usage**

```r
reduce_to_map(n, link, n_name)
```
reduce_to_map

Arguments

- **n**: A network or igraph object.
- **link**: A vector with the identifiers of the vertices to keep.
- **n_name**: Name of the vertex attribute to filter on.

Value

A network or igraph object with only the vertices listed in `link`. 
Index

* datasets
  fvgmap, 3
betweenness, 4
check_network_sf, 2
closeness, 4
degree, 4
fortify, 6
fvgmap, 3
geom_edges, 6
geom_nodes, 6
ggcentrality, 3
ggconn_area, 5
ggnetmap, 6, 9
is_lookup_table, 7
is_network, 7
is_sf, 8
link_network_map, 8, 11
link_network_map2, 9, 11
netmap, 9
netmap_plot, 9, 10
network.layout.extract_coordinates, 9, 11
plot.igraph, 9–11
plot.network, 9–11
reduce_to_map, 11