Package: grand (via r-universe)

September 12, 2024

Title Guidelines for Reporting About Network Data

Version 0.9.0

Description Interactively applies the Guidelines for Reporting About Network Data (GRAND) to an 'igraph' object, and generates a uniform narrative or tabular description of the object.

License GPL-3

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.1.2

Depends R (>= 2.10)

Imports graphics, igraph, methods, tools, utils

Suggests knitr, rmarkdown

VignetteBuilder knitr

URL https://github.com/zpneal/grand

BugReports https://github.com/zpneal/grand/issues

LazyData true

Repository https://zpneal.r-universe.dev

RemoteUrl https://github.com/zpneal/grand

RemoteRef HEAD

RemoteSha 07d868f4b32177a87f07a97b51f32a4169339cb9

Contents

`airport` *US Air Traffic Network*

Description

A weighted and directed network of passenger air traffic in the United States in 2019. Each edge represents a single takeoff and landing, and therefore does not consider possible layovers, connecting flights, round trips, etc. This is the directed version of the undirected air traffic network used by Neal (2022) to illustrate `backbone::disparity()`. GRAND attributes have already been added using `grand()`.

Usage

```
airport
```

Format

igraph object

References

Neal, Z. P. (2022). backbone: An R Package to Extract Network Backbones. *PLOS ONE, 17*, e0269137. doi:10.1371/journal.pone.0269137

`cosponsor` *US Senate Co-Sponsorship Network*

Description

A bipartite network representing US Senators' (co-)sponsorship of Senate Bills during the 116th session (2019-2020). It was obtained using `incidentally::incidence.from.congress()` following the procedure described by Neal (2022). GRAND attributes have already been added using `grand()`.

Usage

```
cosponsor
```

Format

igraph object

References

Neal, Z. P. (2022). Constructing legislative networks in R using incidentally and backbone. *Connections, 42*, 1-9. doi:10.2478/connections2019.026

grand *Apply Guidelines for Reporting About Network Data (GRAND) to an igraph object*

Description

The grand function stores characteristics about the graph as graph attributes that can be summarized in a narrative using the grand.text() or a table using grand.table().

Usage

```
grand(
  G,
  interactive = TRUE,
  name = NA,
  doi = NA,
  url = NA,
  vertex1 = NULL,
  vertex2 = NULL,
  vertex1.total = 0,
  vertex2.total = 0,
  edge.pos = NULL,
  edge.neg = NULL,
  weight = NULL,
  measure = NULL,
  mode = NULL,
  year = NULL,
  topology = character()
)
```

Arguments

G	An igraph object, with weights/signs (if present) stored in E(G)$weight
interactive	boolean: Should GRAND run interactively?
name	string: Name of the network
doi	string: DOI associated with the data
url	string: Link to data
vertex1	string: Entity represented by vertices
vertex2	string: Entity represented by vertices
vertex1.total	numeric: Number of entities in the network's boundary
vertex2.total	numeric: Number of entities in the network's boundary
edge.pos	string: Relationship represented by (positive) edges
edge.neg	string: Relationship represented by negative edges
weight	string: What the edge weights represent

`measure`	string: Scale on which edge weights are measured
`mode`	string: Mode of data collection
`year`	numeric: Year in which data was collected
`topology`	string: Vector of topological metrics to be computed in GRAND summaries

Details

The interactive mode (default) asks the user a series of questions based on the igraph object, while non-interactive mode allows the user to directly supply the relevant attributes.

Data

The first set of interactive questions ask about the data as a whole:

- *name* - This should usually be specified ending with the word "network" or "data" (e.g. "Florentine Families Network" or "Airline Traffic Data").
- *doi* - DOI for a manuscript describing the data.
- *url* - Link to a copy of the data.
- Data collection *mode* - This describes how the data was collected or generated. Chose one of the available options (Survey, Interview, Sensor, Observation, Archival, or Simulation) or choose `Other` to enter something else.
- *year* - In what year were the data collected?

Nodes

The second set of interactive questions ask about the nodes/vertices:

- *vertex1* (and in bipartite graphs, *vertex2*) - What type of entity do the nodes/vertices represent? This should be specified as a plural noun (e.g., "People").
- *vertex1.total* (and in bipartite graphs, *vertex2.total*) - Networks often have an externally-defined boundary that determines which nodes/vertices should be included, even if some are missing from the network. These ask about the total number of nodes/vertices inside the boundary (if one exists) and are used to compute rates of missingness.

Edges

The third set of interactive questions ask about the edges:

- *edge.pos* (and in signed graphs, *edge.neg*) - What type of relationship do the edges represent? This should be specified as a plural noun (e.g., "Friendships").
- *weight* - What do the edge weights represent? Choose one of the available options (Frequency, Intensity, Multiplexity, or Valence) or choose `Other` to enter something else.
- *measure* - How are the edge weights measured? Choose one of the available options (Continuous, Count, Ordinal, or Categorical) or choose `Other` to enter something else.

Topology

The final set of interactive questions ask about relevant topological characteristics. You may choose to (1) use the defaults for this network type, (2) choose characteristics from a list, (3) compute all available characteristics, or (4) compute no characteristics. For comparability and to ensure they are well-defined, all characteristics are computed on an undirected and unweighted version of `G` using existing `igraph` functions. Available topological characteristics include:

- *clustering coefficient* - Computed using `transitivity(G, type = "localaverage")`
- *degree centralization* - Computed using `centr_degree(G)$centralization`
- *degree distribution* - Computed using `fit_power_law(degree(G), implementation = "plfit")`
- *density* - Computed using `edge_density(G)`
- *diameter* - Computed using `diameter(G)`
- *efficiency* - Computed using `global_efficiency(G)`
- *mean degree* - Computed using `mean(degree(G))`
- *modularity* - Computed from a partition generated by `cluster_leiden(G, objective_function = "modularity")`
- *number of communities* - Computed from a partition generated by `cluster_leiden(G, objective_function = "modularity")`
- *number of components* - Computed using `count_components(G)`
- *transitivity* - Computed using `transitivity(G, type = "global")`
- *structural balance* - Computed using the triangle index

Value

An `igraph` object

Examples

```
data(airport)  #Load example data
airport <- grand(airport)  #Apply GRAND interactively
airport <- grand(airport, interactive = FALSE, #Apply GRAND non-interactively
                 vertex1 = "Airports",
                 vertex1.total = 382,
                 edge.pos = "Routes",
                 weight = "Passengers",
                 measure = "Count",
                 mode = "Archival",
                 year = "2019",
           topology = c("clustering coefficient", "mean path length", "degree distribution"))
```

---

grand.table — *Generate a Guidelines for Reporting About Network Data (GRAND) summary table*

Description

The `grand.table` function plots a tabular summary of GRAND attributes that were added to an `igraph` object using `grand()`.

Usage

```
grand.table(G, digits = 3)
```

Arguments

G	An igraph object with GRAND attributed
digits	numeric: number of decimal places to report

Value

A plot

Examples

```
#A weighted, directed network
data(airport)  #Load example data
grand.table(airport)  #Generate narrative

#A bipartite network
data(cosponsor)  #Load example data
grand.table(cosponsor)  #Generate narrative

#A signed network
data(senate)  #Load example data
grand.table(senate)  #Generate narrative
```

---

grand.text *Generate a Guidelines for Reporting About Network Data (GRAND) narrative summary*

Description

The grand.text function writes a narrative summary of GRAND attributes that were added to an igraph object using grand().

Usage

```
grand.text(G, digits = 3)
```

Arguments

G	An igraph object with GRAND attributed
digits	numeric: number of decimal places to report

Value

string: Narrative summary of G

Examples

```
#A weighted, directed network
data(airport)  #Load example data
narrative <- grand.text(airport)  #Generate narrative

#A bipartite network
data(cosponsor)  #Load example data
narrative <- grand.text(cosponsor)  #Generate narrative

#A signed network
data(senate)  #Load example data
narrative <- grand.text(senate)  #Generate narrative
```

---

menu2 *Returns* menu() *response as choice text*

Description

Returns menu() response as choice text

Usage

```
menu2(choices, title, loop = FALSE)
```

Arguments

choices	a character vector of choices
title	a character string to be used as the title of the menu. NULL is also accepted.
loop	boolean: should the menu loop to allow multiple choices?

Value

string: the chosen option

Examples

```
choice <- menu2(choices = c("A", "B", "C"), title = "Choose an option", loop = TRUE)
```

scan2 *Restricts* `scan()` *input to a specified format*

Description

Restricts `scan()` input to a specified format

Usage

```
scan2(prompt, type)
```

Arguments

`prompt`	string: prompt for user input
`type`	string: required format for input

Value

user input in specified format

Examples

```
character <- scan2(prompt = "Type any character", type = "character")
numeric <- scan2(prompt = "Type any number", type = "numeric")
integer <- scan2(prompt = "Type any number", type = "integer")
custom <- scan2(prompt = "Yes or No?", type = c("Y","N"))
```

senate *US Senate Network*

Description

A signed network representing US Senators' alliances and antagonisms, inferred from `cosponsor()` using `backbone::sdsm()` following the procedure described by Neal (2022). GRAND attributes have already been added using `grand()`.

Usage

```
senate
```

Format

igraph object

References

Neal, Z. P. (2022). Constructing legislative networks in R using incidentally and backbone. *Connections, 42*, 1-9. doi:10.2478/connections2019.026

Index