

Package ‘networktools’

April 18, 2017

Title Assorted Tools for Identifying Important Nodes in Networks (Impact, Expected Influence)

Version 1.0.0

Description Includes assorted tools for network analysis. Specifically, includes functions for calculating impact statistics, which aim to identify how each node impacts the overall network structure (global strength impact, network structure impact, edge impact), and for calculating and visualizing expected influence.

Depends R (>= 3.0.0)

License GPL-3

Encoding UTF-8

LazyData true

Imports

qgraph,igraph,IsingFit,reshape2,nnet,ggplot2,gridExtra,stats,graphics,utils,NetworkComparisonTest,devtools

RoxxygenNote 6.0.1

Suggests knitr,
rmarkdown

VignetteBuilder knitr

URL <http://github.com/paytonjjones/networktools>

BugReports <http://github.com/paytonjjones/networktools/issues>

R topics documented:

depression	2
edge.impact	2
expectedInf	4
global.impact	5
impact	6
impact.NCT	8
plot.all.impact	9
plot.edge.impact	10
plot.expectedInf	11
plot.global.impact	12
plot.structure.impact	13
social	14
structure.impact	14
Index	17

depression

Simulated Depression Profiles

Description

This simulated dataset contains severity ratings for 9 symptoms of major depressive disorder in 1000 individuals. Symptom ratings are assumed to be self-reported on a 100 point sliding scale.

Usage

```
depression
```

Format

a dataframe. Columns represent symptoms and rows represent individuals

Examples

```
head(depression)

out1 <- impact(depression)
summary(out1)
plot(out1)

out2 <- edge.impact(depression, gamma=0.75, nodes=c("sleep_disturbance", "psychomotor_retardation"))
summary(out2)
plot(out2)

# Visualize depression networks for "low" psychomotor retardation vs. "high" psychomotor retardation
par(mfrow=c(1,2))
qgraph::qgraph(out2$lo$psychomotor_retardation, title="Low Psychomotor Retardation")
qgraph::qgraph(out2$hi$psychomotor_retardation, title="High Psychomotor Retardation")
```

edge.impact

Edge Impact

Description

Generates a matrix of edge impacts for each specified node. Each scalar in a given matrix represents the degree to which the level of a node impacts the strength of a specified edge in the network

Usage

```
edge.impact(input, gamma, nodes = c("all"), binary.data = FALSE,
  weighted = TRUE, split = c("median", "mean", "forceEqual", "quartiles"))
```

Arguments

input	a matrix or data frame of observations (not a network/edgelist). See included example datasets depression and social .
gamma	the sparsity parameter used in generating networks. Defaults to 0.5 for interval data and 0.25 for binary data
nodes	indicates which nodes should be tested. Can be given as a character string of desired nodes (e.g., <code>c("node1", "node2")</code>) or as a numeric vector of column numbers (e.g., <code>c(1,2)</code>).
binary.data	logical. Indicates whether the input data is binary
weighted	logical. Indicates whether resultant networks preserve edge weights or binarize edges.
split	method by which to split network given non-binary data. "median": median split (excluding the median), "mean": mean split, "forceEqual": creates equally sized groups by partitioning median observations to the smaller group, "quartile": uses the top and bottom quartile as groups

Details

For an explanation of impact functions in general, see [impact](#).

Edge impact is the change in an edge's value as a function of a given node. A separate edge impact value is calculated for each edge in the network.

It is highly useful to plot the edge impacts as if they were a network. Positive edges in the resultant graph can be interpreted as edges that were made more positive by the given node, and negative edges can be interpreted as edges that were made more negative by the given node.

The `$hi` and `$lo` output of `edge.impact` can also be used to quickly visualize the difference in network structure depending on node level (see examples).

Value

`edge.impact()` returns a list of class "edge.impact" which contains:

impact	a list of matrices. Each symmetric matrix contains the edge impacts for the given node
lo	a list of matrices. Each symmetric matrix contains the edge estimates for the given node's lower half
hi	a list of matrices. Each symmetric matrix contains the edge estimates for the given node's upper half
edgelist	a list of dataframes. Each dataframe contains an edgelist of edge impacts

Examples

```
out <- edge.impact(depression[450:550,1:3], nodes="anhedonia")

out1 <- edge.impact(depression)
out2 <- edge.impact(depression, gamma=0.65,
  nodes=c("sleep_disturbance", "psychomotor_retardation"))
out3 <- edge.impact(social, nodes=c(1:6, 9), binary.data=TRUE)

summary(out1)
plot(out1, nodes="concentration_problems")
```

```
# Visualize edge impacts of psychomotor_retardation
# as a single network
plot(out1, nodes="psychomotor_retardation", type.edgeplot="single")

# Visualize the edge impacts of psychomotor_retardation
# as contrast between high and low
plot(out1, nodes="psychomotor_retardation", type.edgeplot="contrast")

# Extract the impact of psychomotor_retardation on a single edge
out1$impact[["psychomotor_retardation"]][["worthlessness", "fatigue"]]

# Extract edge impacts of node Dan in edgelist format
out3$edgelist$Dan
```

expectedInf	<i>Expected Influence</i>
-------------	---------------------------

Description

Calculates the one-step and two-step expected influence of each node.

Usage

```
expectedInf(network, step = c("both", 1, 2), directed = FALSE)
```

Arguments

network	an object of type qgraph, igraph, or an adjacency matrix representing a network. Adjacency matrices should be complete (e.g., not only upper or lower half)
step	compute 1-step expected influence, 2-step expected influence, or both
directed	logical. Specifies if edges are directed, defaults to FALSE

Details

When a network contains both positive and negative edges, traditional centrality measures such as strength centrality may not accurately predict node influence on the network. Robinaugh, Millner, & McNally (2016) showed that in these cases, expected influence is a more appropriate measure.

One-step expected influence is defined as the sum of all edges extending from a given node (where the sign of each edge is maintained).

Two-step expected influence, as the name implies, measures connectivity up to two edges away from the node. It is defined as the sum of the (weighted) expected influences of each node connected to the initial node plus the one-step expected influence of the initial node. Weights are determined by the edge strength between the initial node and each "second step" node.

See citations in the references section for further details.

References

Robinaugh, D. J., Millner, A. J., & McNally, R. J. (2016). Identifying highly influential nodes in the complicated grief network. *Journal of abnormal psychology*, 125, 747.

Examples

```
out1 <- expectedInf(cor(depression[,1:5]))

out1$step1
out1$step2
plot(out1)
plot(out1, order="value", zscore=TRUE)

igraph_obj <- igraph::graph_from_adjacency_matrix(cor(depression))
out_igraph <- expectedInf(igraph_obj)

qgraph_obj <- qgraph::qgraph(cor(depression), DoNotPlot=TRUE)
out_qgraph <- expectedInf(qgraph_obj)

Ising_adj_mat <- IsingFit::IsingFit(social, plot=FALSE)$weiadj
out_Ising <- expectedInf(Ising_adj_mat)
plot(out_Ising)
```

global.impact

Global Strength Impact

Description

Generates the global strength impact of each specified node. Global strength impact can be interpreted as the degree to which the level of a node impacts the overall connectivity of the network

Usage

```
global.impact(input, gamma, nodes = c("all"), binary.data = FALSE,
  weighted = TRUE, split = c("median", "mean", "forceEqual", "quartiles"))
```

Arguments

input	a matrix or data frame of observations (not a network/edgelist). See included example datasets depression and social .
gamma	the sparsity parameter used in generating networks. Defaults to 0.5 for interval data and 0.25 for binary data
nodes	indicates which nodes should be tested. Can be given as a character string of desired nodes (e.g., c("node1", "node2")) or as a numeric vector of column numbers (e.g., c(1,2)).
binary.data	logical. Indicates whether the input data is binary
weighted	logical. Indicates whether resultant networks preserve edge weights or binarize edges.
split	method by which to split network given non-binary data. "median": median split (excluding the median), "mean": mean split, "forceEqual": creates equally sized groups by partitioning median observations to the smaller group, "quartile": uses the top and bottom quartile as groups

Details

For an explanation of impact functions in general, see [impact](#).

Global strength is defined as the sum of the absolute value of all edges in the network, and is closely related to the concept of density (density is the sum of edges not accounting for absolute values). Global strength impact measures to what degree the global strength varies as a function of each node.

Value

`global.impact()` returns a list of class "global.impact" which contains:

<code>impact</code>	a named vector containing the global strength impact for each node tested
<code>lo</code>	a named vector containing the global strength estimate for the lower half
<code>hi</code>	a named vector containing the global strength estimate for the upper half

Examples

```
out <- global.impact(depression[,1:3])

out1 <- global.impact(depression)
out2 <- global.impact(depression, gamma=0.65,
  nodes=c("sleep_disturbance", "psychomotor_retardation"))
out3 <- global.impact(social, binary.data=TRUE)
out4 <- global.impact(social, nodes=c(1:6, 9), binary.data=TRUE)

summary(out1)
plot(out1)
```

<code>impact</code>	<i>Network Impact (combined function)</i>
---------------------	---

Description

Generates the global strength impact, network structure impact, and edge impact simultaneously for a given set of nodes. See [global.impact](#), [structure.impact](#), and [edge.impact](#) for additional details

Usage

```
impact(input, gamma, nodes = c("all"), binary.data = FALSE,
  weighted = TRUE, split = c("median", "mean", "forceEqual", "quartiles"))
```

Arguments

<code>input</code>	a matrix or data frame of observations (not a network/edgelist). See included example datasets depression and social .
<code>gamma</code>	the sparsity parameter used in generating networks. Defaults to 0.5 for interval data and 0.25 for binary data

<code>nodes</code>	indicates which nodes should be tested. Can be given as a character string of desired nodes (e.g., <code>c("node1", "node2")</code>) or as a numeric vector of column numbers (e.g., <code>c(1,2)</code>).
<code>binary.data</code>	logical. Indicates whether the input data is binary
<code>weighted</code>	logical. Indicates whether resultant networks preserve edge weights or binarize edges.
<code>split</code>	method by which to split network given non-binary data. "median": median split (excluding the median), "mean": mean split, "forceEqual": creates equally sized groups by partitioning median observations to the smaller group, "quartile": uses the top and bottom quartile as groups

Details

The structures of networks sometimes vary as a function of certain external variables. For instance, Pe et al. (2015) found that the structure of mood networks varied as a function of whether or not individuals had been diagnosed with major depression.

The structures of networks may also vary as a function of *internal* variables; that is to say, as a function of each node. ***Impact statistics measure the degree to which node levels impact network structure.*** Impact statistics are similar to centrality statistics in the sense that they are a property of each node in a network.

Three relevant impact statistics are included in the `networktools` package: global strength impact, network structure impact, and edge impact. To ease computational burden, all three statistics are calculated simultaneously in the `impact` function. They can also be calculated separately using `global.impact`, `structure.impact`, and `edge.impact`.

Impact statistics are calculated by temporarily regarding a node as an *external* variable to the network. The remaining data are then divided into two networks according to a median split (default) on the external node. Network invariance measures are then computed on the two networks. While median splits are not advisable when continuous analyses are possible, it is not possible to compute networks in a continuous fashion. The median split excludes observations that fall exactly on the median. In the case of binary data, data are split by level rather than by median.

Value

`impact` returns a list of class `"all.impact"` which contains:

1. A list of class `"global.impact"`
2. A list of class `"structure.impact"`
3. A list of class `"edge.impact"`

See `global.impact`, `structure.impact`, and `edge.impact` for details on each list

Examples

```
out <- impact(depression[,1:3])

out1 <- impact(depression)
out2 <- impact(depression, gamma=0.65, nodes=c("sleep_disturbance", "psychomotor_retardation"))
out3 <- impact(social, binary.data=TRUE)
out4 <- impact(social, nodes=c(1:6, 9), binary.data=TRUE)

summary(out1)
plot(out1)
```

```
# Extract the impact of psychomotor_retardation on the
# edge that runs between worthlessness and fatigue
out1$Edge$impact[["psychomotor_retardation"]][["worthlessness", "fatigue"]]
```

impact.NCT

Network Comparison Test for Impact Statistics

Description

This function wraps the function NCT from the [NetworkComparisonTest](#) package to provide an explicit test for the significance of node impacts.

Usage

```
impact.NCT(input, it, gamma, nodes = c("all"), binary.data = FALSE,
  weighted = TRUE, split = c("median", "mean", "forceEqual", "quartiles"),
  paired = FALSE, AND = TRUE, test.edges = FALSE, edges,
  progressbar = TRUE)
```

Arguments

input	a matrix or data frame of observations (not a network/edgelist). See included example datasets depression and social .
it	the number of iterations (permutations) in each network comparison test
gamma	the sparsity parameter used in generating networks. Defaults to 0.5 for interval data and 0.25 for binary data
nodes	indicates which nodes should be tested. Can be given as a character string of desired nodes (e.g., c("node1", "node2")) or as a numeric vector of column numbers (e.g., c(1,2)).
binary.data	logical. Indicates whether the input data is binary
weighted	logical. Indicates whether resultant networks preserve edge weights or binarize edges.
split	method by which to split network given non-binary data. "median": median split (excluding the median), "mean": mean split, "forceEqual": creates equally sized groups by partitioning median observations to the smaller group, "quartile": uses the top and bottom quartile as groups
paired	Logical. Can be TRUE or FALSE to indicate whether the samples are dependent or not. If paired is TRUE, relabeling is performed within each pair of observations. If paired is FALSE, relabeling is not restricted to pairs of observations. Note that, currently, dependent data is assumed to entail one group measured twice.
AND	Logical. Can be TRUE or FALSE to indicate whether the AND-rule or the OR-rule should be used to define the edges in the network. Defaults to TRUE. Only necessary for binary data.
test.edges	Logical. Can be TRUE or FALSE to indicate whether or not differences in individual edges should be tested.

edges	Character or list. When 'all', differences between all individual edges are tested. When provided a list with one or more pairs of indices referring to variables, the provided edges are tested. A Holm-Bonferroni correction is applied to control for multiple testing.
progressbar	Logical. Should the pbar be plotted in order to see the progress of the estimation procedure? Defaults to TRUE.
...	additional optional arguments to be passed to the NCT function internally (paired, AND, test.edges, edges, progressbar)

Details

The NCT method is computationally intensive. It is recommended that users test a subset of nodes at a time using the nodes argument, rather than testing all nodes simultaneously.

In order to be interpreted in a meaningful way, the significance of impact statistics should be explicitly tested.

The `NCT` function from the [NetworkComparisonTest](#) uses a permutation test to determine the significance of structure invariances between two networks. Because impact statistics are mathematically defined as structural invariance between two networks, NCT is an appropriate method to test the significance of impact statistics.

`impact.NCT` returns an object of class NCT, which includes p-values for invariances.

Value

`impact` returns a list where each element is an object of class NCT

Examples

```
out <- impact.NCT(depression[,1:5], it=5, nodes="psychomotor_retardation")

NCT1 <- impact.NCT(depression, it=25, nodes="psychomotor_retardation")
NCT1$psychomotor_retardation$glstrinv.pval
NCT1$psychomotor_retardation$nwinv.pval
## Both significant

NCT2 <- impact.NCT(social, it=25, nodes="Kim", binary.data=TRUE)
NCT2$Kim$glstrinv.pval
NCT2$Kim$nwinv.pval
## Only global strength impact is significant

##Note: for speed, 25 permutations are iterated here; more permutations are necessary in practice
```

plot.all.impact	<i>Plot "all.impact" objects</i>
-----------------	----------------------------------

Description

Convenience function for generating impact plots

Usage

```
## S3 method for class 'all.impact'
plot(x, order = c("alphabetical", "value"),
     zscore = FALSE, abs_val = FALSE, ...)
```

Arguments

x	an output object from an impact function (class all.impact)
order	"alphabetical" orders nodes alphabetically, "value" orders nodes from highest to lowest impact value
zscore	logical. Converts raw impact statistics to z-scores for plotting
abs_val	logical. Plot absolute values of global strength impacts
...	other plotting specifications (ggplot2)

Details

Inputting an object of class global.impact or structure.impact will return a line plot that shows the relative impacts of each node. Inputting a all.impact object will return both of these plots simultaneously

Examples

```
out <- impact(depression[,1:5])
plot(out)

out1 <- impact(depression)
plot(out1)
plot(out1, order="value", zscore=TRUE)
```

plot.edge.impact	<i>Plot "edge.impact" objects</i>
------------------	-----------------------------------

Description

Convenience function for generating edge impact plots

Usage

```
## S3 method for class 'edge.impact'
plot(x, nodes = c("first", "all"),
     type.edgeplot = c("contrast", "single"), title = NULL, ...)
```

Arguments

x	an output object from an impact function (edge.impact)
nodes	specifies which impact graph(s) to be plotted. Can be given as a character string of desired node(s) (e.g., c("node1", "node2")) or as a numeric vector of column numbers (e.g., c(1,2)).

type.edgeplot	"contrast" returns two separate networks: one for low values of the given node, and one for high values. "single" returns a network where edges represent the edge impact of the given node.
title	if not otherwise specified, title is automatically generated
...	other plotting specifications (qgraph)

Details

Inputting a `edge.impact` object will return network plots. Depending on the `type.edgeplot` argument, two types of networks are possible. Using "contrast" will return "true" estimated networks from the data, separated by a median split on the selected node. Using "single" will return a network where the edges represent the edge impacts for the selected node (e.g., thick positive edges represent a strong positive edge impact)

Examples

```
out <- edge.impact(depression[450:550,1:3])
plot(out, nodes="anhedonia", type.edgeplot="single")

out1 <- edge.impact(depression)
plot(out1, nodes="concentration_problems")
plot(out1, nodes="psychomotor_retardation",
      type.edgeplot="single")

out2 <- impact(depression)
plot(out2$Edge, nodes="concentration_problems")
```

plot.expectedInf	<i>Plot "expectedInf" objects</i>
------------------	-----------------------------------

Description

Convenience function for plotting expected influence

Usage

```
## S3 method for class 'expectedInf'
plot(x, order = c("alphabetical", "value"),
      zscore = FALSE, ...)
```

Arguments

x	an output object from an <code>expectedInf</code> (class <code>expectedInf</code>)
order	"alphabetical" orders nodes alphabetically, "value" orders nodes from highest to lowest impact value
zscore	logical. Converts raw impact statistics to z-scores for plotting
...	other plotting specifications (ggplot2)

Details

Inputting an object of class `expectedInf` will return a line plot that shows the relative one-step and/or two-step expected influence of each node.

Examples

```
out1 <- expectedInf(myNetwork)
plot(out1$step1)

plot(out1, order="value", zscore=TRUE)

igraph_obj <- igraph::graph_from_adjacency_matrix(cor(depression))
ei_igraph <- expectedInf(igraph_obj)

qgraph_obj <- qgraph::qgraph(cor(depression), plot=FALSE)
ei_qgraph <- expectedInf(qgraph_obj)

Ising_adj_mat <- IsingFit::IsingFit(social, plot=FALSE)$weiadj
ei_Ising <- expectedInf(Ising_adj_mat)
plot(ei_Ising)
```

plot.global.impact	<i>Plot "global.impact" objects</i>
--------------------	-------------------------------------

Description

Convenience function for generating global strength impact plots

Usage

```
## S3 method for class 'global.impact'
plot(x, order = c("alphabetical", "value"),
     zscore = FALSE, abs_val = FALSE, ...)
```

Arguments

<code>x</code>	an output object from an impact function (class <code>global.impact</code>)
<code>order</code>	"alphabetical" orders nodes alphabetically, "value" orders nodes from highest to lowest impact value
<code>zscore</code>	logical. Converts raw impact statistics to z-scores for plotting
<code>abs_val</code>	logical. Plot absolute values of global strength impacts
<code>...</code>	other plotting specifications (ggplot2)

Details

Inputting an object of class `global.impact` will return a line plot that shows the relative global impacts of each node.

Examples

```

out <- global.impact(depression[,1:5])
plot(out)

out1 <- global.impact(depression)
plot(out1)
plot(out1, order="value", zscore=TRUE)
out2 <- impact(depression)
plot(out2$Global.Strength)

```

plot.structure.impact *Plot "structure.impact" objects*

Description

Convenience function for generating network structure impact plots

Usage

```

## S3 method for class 'structure.impact'
plot(x, order = c("alphabetical", "value"),
     zscore = FALSE, ...)

```

Arguments

x	an output object from an impact function (class structure.impact)
order	"alphabetical" orders nodes alphabetically, "value" orders nodes from highest to lowest impact value
zscore	logical. Converts raw impact statistics to z-scores for plotting
...	other plotting specifications (ggplot2)

Details

Inputting an object of class network.impact will return a line plot that shows the relative network impacts of each node.

Examples

```

out <- structure.impact(depression[,1:5])
plot(out)

out1 <- structure.impact(depression)
plot(out1)
plot(out1, order="value", zscore=TRUE)
out2 <- impact(depression)
plot(out2$Network.Structure)

```

social

*Simulated Social Engagement Data***Description**

This simulated dataset contains binary social engagement scores for 16 individuals. For 400 social media posts on a group forum, individuals were given a score of 1 if they engaged in group conversation regarding the post, and a score of 0 if they did not engage with the post.

Usage

```
social
```

Format

a dataframe. Columns represent individuals (nodes) and rows represent engagement in social media group conversations

Examples

```
head(social)

out1 <- impact(social, binary.data=TRUE)
summary(out1)
plot(out1)

out2 <- edge.impact(social, binary.data=TRUE, gamma=0.2, nodes=c("Kim", "Bob", "Dan"))
summary(out2)
plot(out2)

# Visualize the difference in the social networks depending
# on whether or not Joe participated (large global strength impact)
par(mfrow=c(1,2))
qgraph::qgraph(out1$Edge$lo$Joe, title="Joe Absent")
qgraph::qgraph(out1$Edge$hi$Joe, title="Joe Present")

# Visualize the difference in the social networks depending
# on whether or not Don participated (large network structure impact)
par(mfrow=c(1,2))
qgraph::qgraph(out1$Edge$lo$Don, title="Don Absent")
qgraph::qgraph(out1$Edge$hi$Don, title="Don Present")
```

structure.impact

*Network Structure Impact***Description**

Generates the network structure impact of each specified node. Network structure impact can be interpreted as the degree to which the level of a node causes change in the network structure

Usage

```
structure.impact(input, gamma, nodes = c("all"), binary.data = FALSE,
  weighted = TRUE, split = c("median", "mean", "forceEqual", "quartiles"))
```

Arguments

<code>input</code>	a matrix or data frame of observations (not a network/edgelist). See included example datasets depression and social .
<code>gamma</code>	the sparsity parameter used in generating networks. Defaults to 0.5 for interval data and 0.25 for binary data
<code>nodes</code>	indicates which nodes should be tested. Can be given as a character string of desired nodes (e.g., <code>c("node1", "node2")</code>) or as a numeric vector of column numbers (e.g., <code>c(1,2)</code>).
<code>binary.data</code>	logical. Indicates whether the input data is binary
<code>weighted</code>	logical. Indicates whether resultant networks preserve edge weights or binarize edges. Note: unweighted networks will always result in a network structure impact of 0 or 1.
<code>split</code>	method by which to split network given non-binary data. "median": median split (excluding the median), "mean": mean split, "forceEqual": creates equally sized groups by partitioning median observations to the smaller group, "quartile": uses the top and bottom quartile as groups

Details

For an explanation of impact functions in general, see [impact](#).

Network structure impact computes network structure invariance as a function of node level. Network structure invariance is defined as the absolute value of the single largest edge invariance between two networks. A large network structure invariance is generally used to indicate the *instability* of network structure across groups. For instance, a low global strength invariance coupled with a high network structure invariance would mean that although the overall connectivity remains stable, the actual structure of those edges is unstable across groups.

Value

`structure.impact()` returns a list of class "structure.impact" which contains:

<code>impact</code>	a named vector containing the network structure impact for each node tested. Network structure impacts are given as absolute values
<code>edge</code>	a list of vectors. Each vector contains a the edge impact of the most strongly impacted edge (e.g., the network structure impact)
<code>lo</code>	a named vector containing the edge estimate for the lower half of the most strongly impacted edge
<code>hi</code>	a named vector containing the edge estimate for the upper half of the most strongly impacted edge

Examples

```
out <- structure.impact(depression[,1:3])

out1 <- structure.impact(depression)
out2 <- structure.impact(depression, gamma=0.65,
```

```
      nodes=c("sleep_disturbance", "psychomotor_retardation"))
out3 <- structure.impact(social, binary.data=TRUE)
out4 <- structure.impact(social, nodes=c(1:6, 9), binary.data=TRUE)

summary(out1)
plot(out1)

#Determine which edge drove network structure impact of "sadness"
out1$edge$sadness
```


Index

*Topic **datasets**

depression, [2](#)

social, [14](#)

depression, [2](#), [3](#), [5](#), [6](#), [8](#), [15](#)

edge.impact, [2](#), [6](#), [7](#)

expectedInf, [4](#)

global.impact, [5](#), [6](#), [7](#)

impact, [3](#), [6](#), [6](#), [7](#), [9](#), [15](#)

impact.NCT, [8](#)

NCT, [9](#)

NetworkComparisonTest, [8](#), [9](#)

plot.all.impact, [9](#)

plot.edge.impact, [10](#)

plot.expectedInf, [11](#)

plot.global.impact, [12](#)

plot.structure.impact, [13](#)

social, [3](#), [5](#), [6](#), [8](#), [14](#), [15](#)

structure.impact, [6](#), [7](#), [14](#)