Package ‘tnet’

February 20, 2015

Type Package
Version 3.0.11
Date 2012-11-19
Title tnet: Software for Analysis of Weighted, Two-mode, and Longitudinal networks
Author Tore Opsahl
Maintainer Tore Opsahl <tore@opsahl.co.uk>
Depends R (>= 2.13.0), igraph, survival
Description R package for analyzing weighted, two-mode, and longitudinal networks.
License CC BY-NC 3.0 + file LICENSE
Repository CRAN
Date/Publication 2012-11-20 07:05:49
NeedsCompilation no

R topics documented:

  tnet-package ..........................................................  2
  add_window_l ...........................................................  4
  as.static.tnet .........................................................  5
  as.tnet .................................................................  6
  betweenness_w ........................................................  7
  celegans.n306.net .................................................  8
  closeness_w ............................................................  9
  clustering_local_tm ............................................... 10
  clustering_local_w ................................................ 11
  clustering_tm .......................................................... 13
  clustering_w .......................................................... 14
  compress_ids ......................................................... 15
  Cross.Parker.Consulting.net.info ................................... 16
  Cross.Parker.Manufacturing.net.info ............................... 17
  Davis.Southern.women.2mode ....................................... 18
tnet-package

Collection of functions for analysing weighted networks, two-mode networks, and longitudinal networks

Description

This package is created to analyse weighted networks, two-mode networks, and longitudinal networks datasets. Binary ties limit the richness of network analyses as relations are unique. The two-mode structure contains a number of features lost when projection it to a one-mode network. Longitudinal datasets allow for an understanding of the causal relationship among ties, which is not the case in cross-sectional datasets as ties are dependent upon each other.

Details

Package: tnet
Type: Package
Version: 3.0.11
Date: 2012-11-19
This package is created to analyse weighted networks, two-mode networks, and longitudinal networks datasets. More information is available on http://toreopsahl.com/tnet/

It utilises three forms of data structures (it can automatically convert matrices etc into these formats, see the as.tnet-function):

1) simple weighted data in the following format (creator.node.id target.node.id tie.weight):

   1 2 4
   1 3 2

   Note: For undirected networks, each tie must be mentioned twice (see the symmetrise_w-function). For example,

   1 2 4
   2 1 4
   1 3 2
   3 1 2

2) two-mode data in the following format (primary.node.id secondar.node.id tie.weight.optional):

   1 1 1
   2 1 2

3) timed data in the following format (MySQL-timestamp.as.character.string creator.node.id target.node.id tie.weight):

   "2007-09-12 13:45:00" 1 2 1
   "2007-09-12 13:46:31" 1 2 1

   If ties are repeated, the tie increases the weighted. The weight column decides how much weight is added at each time (this can take a negative value to decrease the weight).

   Attribute files are read as follows:

   0 1 3
   0 3 2
   1 3 3

   where the first row refers to node 1, the second row to node 2, etc. The first column refers to the first attribute, second column to the second attribute and so on.

   A big thank you to the igraph guys as this package relies on their work for many of the more computational tasks!

Author(s)

Tore Opsahl; http://toreopsahl.com

References

http://toreopsahl.com/tnet/

Examples

# Generate a random weighted graph
rg <- rg_w(nodes=100, arcs=300, directed=TRUE)

# Calculate clustering coefficient
clustering_w(rg)
add_window_l

Add smoothing window to a longitudinal network

Description

This function adds negative ties (i.e., a smoothing window) to a longitudinal network.

Usage

add_window_l(netLwindow]R1L removeNnodes]trueI

Arguments

net Longitudinal network
window Number of days before ties 'expire'.
remove.nodes Whether or not nodes should be removed from the network if they have no more ties. This function adds a self-loop with a negative weight at the time of a node’s last tie plus the length of the window.

Value

Returns the longitudinal network with negative arcs.

Note

version 1.0.0

Author(s)

Tore Opsahl; http://toreopsahl.com

References

tore@opsahl.co.uk

Examples

t <- c('2007-09-12 13:45:00',
  '2007-09-12 13:46:31',
  '2007-09-12 13:47:54',
  '2007-09-12 13:48:21',
  '2007-09-12 13:49:27',
  '2007-09-12 13:58:14',
  '2007-09-12 13:52:17',
  '2007-09-12 13:56:59');
i <- c(1,1,1,1,1,1,1,1);
j <- c(2,2,2,2,2,3,3);
w <- c(1,1,1,1,1,1,1,1);
as.static.tnet

sample <- data.frame(t, i, j, w);

## Run the programme
add_window.l(sample, window=21)

---

as.static.tnet  Transform a longitudinal network to a static edgelist network

Description

This function transforms a longitudinal network to a static edgelist

Usage

as.static.tnet(ld)

Arguments

- **ld**: Longitudinal network

Value

Returns the data in an edgelist format.

Note

version 1.0.0

Author(s)

Tore Opsahl; http://toreopsahl.com

References

tore@opsahl.co.uk

Examples

t <- c('2007-09-12 13:45:00',
      '2007-09-12 13:46:31',
      '2007-09-12 13:47:54',
      '2007-09-12 13:48:21',
      '2007-09-12 13:49:27',
      '2007-09-12 13:58:14',
      '2007-09-12 13:52:17',
      '2007-09-12 13:56:59');
i <- c(1,1,1,1,1,1,1,1);
j <- c(2,2,2,2,2,2,3,3);
w <- c(1,1,1,1,-1,1,1);
net <- data.frame(t, i, j, w);

## Run the programme
as.static.tnet(net)

### as.tnet

*Ensures that networks conform to the tnet standards*

**Description**

Checks that a network conforms to the tnet standards, and attaches a label. If the type parameter is not set, the network is assumed to be a binary two-mode network, a weighted one-mode network, or a longitudinal network if there are 2, 3, or 4 columns respectively. Moreover, if a matrix is entered (more than 4 columns and rows), it is assumed to be a weighted one-mode network if square or a two-mode network if non-square.

**Usage**

`as.tnet(net, type=NULL)`

**Arguments**

- **net**: A network in an edgelist or matrix format. It can be a weighted one-mode network, a binary two-mode network, a weighted two-mode network, or a longitudinal network. If the data-object has two-columns, it is assumed to be a binary two-mode network; three columns, weighted one-mode network; four columns, longitudinal; five or more and the same number of rows and columns, weighted one-mode network; five or more and –not– the same number of rows and columns, it is assumed to be a two-mode network.

- **type**: If you would like to specify the type of network. This could be "weighted one-mode tnet", "binary two-mode tnet", "weighted two-mode tnet", or "longitudinal tnet".

**Value**

Returns the network with an attached label.

**Note**

version 1.0.0

**Author(s)**

Tore Opsahl; http://toreopsahl.com
betweenness_w

Examples

```r
## Load sample data
sample <- rbind(
c(1,2,4),
c(1,3,2),
c(2,1,4),
c(2,3,4),
c(2,4,1),
c(2,5,2),
c(3,1,2),
c(3,2,4),
c(4,2,1),
c(5,2,2),
c(5,6,1),
c(6,5,1))

## Run the programme
as.tnet(sample)
```

Description

This function calculates betweenness scores for nodes in a weighted network based on the `distance_w`-function.
Note: This algorithm relies on the igraphs package’s implementation of Dijkstra’s algorithm. Currently, it does not find multiple shortest paths if two exist.

Usage

`betweenness_w(net, directed=NULL, alpha=1)`

Arguments

- `net`: A weighted edgelist
- `directed`: logical, whether the network is directed or undirected. Default is `NULL`, this means that the function checks whether the edgelist is directed or not.
- `alpha`: sets the alpha parameter in the generalised measures from Opsahl, T., Agneessens, F., Skvoretz, J., 2010. Node Centrality in Weighted Networks: Generalizing Degree and Shortest Paths. Social Networks. If this parameter is set to 1 (default), the Dijkstra shortest paths are used. The length of these paths rely simply on the tie weights and disregards the number of nodes on the paths.

Value

Returns a data.frame with two columns: the first column contains the nodes’ ids, and the second column contains the nodes’ betweenness scores.
Note

version 1.0.0

Author(s)

Tore Opsahl; http://toreopsahl.com

References


Examples

```r
## Load sample data
sampledata <- rbind(
  c(1,2,1),
  c(1,3,5),
  c(2,1,1),
  c(2,4,6),
  c(3,1,5),
  c(3,4,10),
  c(4,2,6),
  c(4,3,10))

## Run the programme
betweenness_w(sampledata)
```

celegans.n306.net  The neural network of the Caenorhabditis elegans worm (C.elegans)

Description

This dataset contains the neural network of the Caenorhabditis elegans worm (C.elegans). It was studied by Watts and Strogatz (1998). The network contains 306 nodes that represent neurons. Two neurons are connected if at least one synapse or gap junction exist between them. The weight is the number of synapses and gap junctions. This network was obtained from the Collective Dynamics Group’s website.

Usage

celegans.n306.net

Format

A data frame with three columns. The first is the id of the sender; the second is the id of the receiver; and the third is the weight of the tie.
closeness_w

Author(s)
Tore Opsahl; http://toreopsahl.com

References
http://toreopsahl.com/datasets/

closeness_w Closeness centrality in a weighted network

Description
This function calculates closeness scores for nodes in a weighted network based on the distance_w-function.

Usage
closeness_w(net, directed=NULL, gconly=TRUE, precomp.dist=NULL, alpha=1)

Arguments
net A weighted edgelist
directed Logical: whether the edgelist is directed or undirected. Default is NULL, then the function detects this parameter.
gconly Logical: whether to calculate closeness only on the main component (traditional closeness). Default is TRUE. If this parameter is set to FALSE, a closeness measure for all nodes is computed. For details, see http://toreopsahl.com/2010/03/20/closeness-centrality-in-networks-with-disconnected-components/
precomp.dist If you have already computed the distance matrix using distance_w-function, you can enter the name of the matrix-object here.
alpha sets the alpha parameter in the generalised measures from Opsahl, T., Agneessens, F., Skvoretz, J., 2010. Node Centrality in Weighted Networks: Generalizing Degree and Shortest Paths. Social Networks. If this parameter is set to 1 (default), the Dijkstra shortest paths are used. The identification procedure of these paths rely simply on the tie weights and disregards the number of nodes on the paths.

Value
Returns a data.frame with three columns: the first column contains the nodes’ ids, the second column contains the closeness scores, and the third column contains the normalised closeness scores (i.e., divided by N-1).

Note
version 1.0.0
**Author(s)**

Tore Opsahl; http://toreopsahl.com

**References**


**Examples**

```r
## Load sample data
sampledata <- rbind(
  c(1,2,4),
  c(1,3,2),
  c(2,1,4),
  c(2,3,4),
  c(2,4,1),
  c(2,5,2),
  c(3,1,2),
  c(3,2,4),
  c(4,2,1),
  c(5,2,2),
  c(5,6,1),
  c(6,5,1))

## Run the programme
closeness_w(sampledata)
```

---

**clustering_local_tm**  
*Redefined local clustering coefficient for two-mode networks*

**Description**


**Usage**

`clustering_local_tm(net)`

**Arguments**

- `net`  
  A binary or weighted two-mode edgelist

**Value**

Returns the local clustering coefficient for the primary node set (the first of an edgelist or the rows of a matrix)
clustering_local_w

Note

version 1.0.0

Author(s)

Tore Opsahl; http://toreopsahl.com

References


Examples

# Weighted two-mode network
net <- cbind(
  i=c(1,1,2,2,2,3,4,5,6),
p=c(1,2,1,3,4,2,3,4,3,5,5),
w=c(3,5,6,1,2,6,2,1,3,1,2))

## Run binary clustering function
clustering_local_tm(net[,,1:2])

## Run weighted clustering function
clustering_local_tm(net)

clustering_local_w Barrat et al. (2004) generalised local clustering coefficient

Description

This function calculates Barrat et al. (2004) generalised local clustering coefficient. See http://toreopsahl.com/2009/01/23/weighted-local-clustering-coefficient/ for a detailed description. By default it defines the triplet value as the average of the two tie weights; however it can also define it differently. See the blog post.

Note: If there are very large tie weights in a network, the geometric method in R fails. However, this can be fixed by transforming the values.

net[,"w"] <- (net[,"w"])/min(net[,"w"])

This step is not required unless you receive warnings when running the function.

Usage

clustering_local_w(net, measure = "am")
Arguments

- **net**: A weighted edgelist
- **measure**: The measure-switch control the method used to calculate the value of the triplets.
  - *am* implies the arithmetic mean method (default)
  - *gm* implies the geometric mean method
  - *mi* implies the minimum method
  - *ma* implies the maximum method
  - *bi* implies the binary measures
  This can be `c("am", "gm", "mi", "ma", "bi")` to calculate all.

Value

Returns a data.frame with at least two columns: the first column contains the nodes’ ids, and the remaining columns contain the corresponding clustering scores.

Note

version 1.0.0

Author(s)

Tore Opsahl; http://toreopsahl.com

References

arXiv:cond-mat/0311416

Examples

```r
# Generate a random graph
#density: 300/(100*99)=0.03030303;
#this should be average from random samples
rg <- rg_w(nodes=100,arcs=300,weights=1:10,directed=FALSE)

# Run clustering function
clustering_local_w(rg)
```
clustering_tm

Redefining clustering coefficient for two-mode networks

Description

Note: If you are having problems with this function (i.e., run out of memory or it being slow for simulations), there is a quicker and much more memory efficient c++ function. However, this function is not fully integrated in R, and requires a few extra steps. Send me an email to get the source-code and Windows-compiled files.

Usage
clustering_tm(net, subsample=1, seed=NULL)

Arguments

net
A binary or weighted two-mode edgelist

subsample
Whether a only a subset of 4-paths should we used when calculating the measure. This is particularly useful when running out of memory analysing large networks. If it is set to 1, all the 4-paths are analysed. If it set to a value below one, this is roughly the proportion of 4-paths that will be analysed. If it is set to an integer greater than 1, this number of ties that form the first part of a 4-path that will be analysed. Note: The c++ functions are better as they analyse the full network.

seed
If a subset of 4-paths is analysed, by setting this parameter, the results are reproducible.

Value
Returns the outcome of the equation presented in the paper

Note
version 1.0.0

Author(s)
Tore Opsahl; http://toreopsahl.com

References
Examples

# Weighted two-mode network
net <- cbind(
  i=c(1,1,2,2,3,3,4,4,5,5,6),
  p=c(1,2,1,3,4,2,3,4,3,5,5),
  w=c(3,5,6,1,2,6,2,1,3,1,2))

## Run binary clustering function
clustering_tm(net[,1:2])

## Run weighted clustering function
clustering_tm(net)

clustering_w Generalised clustering coefficient

Description

Note: If you are having problems with this function (i.e., run out of memory or it being slow for simulations), there is a quicker and much more memory efficient c++ function. However, this function is not fully integrated in R, and requires a few extra steps. Send me an email to get the source-code and Windows-compiled files.

Usage

clustering_w(net, measure = "am")

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>net</td>
<td>A weighted edgelist</td>
</tr>
<tr>
<td>measure</td>
<td>The measure-switch control the method used to calculate the value of the triplets. am implies the arithmetic mean method (default) gm implies the geometric mean method mi implies the minimum method ma implies the maximum method bi implies the binary measure This can be c(&quot;am&quot;, &quot;gm&quot;, &quot;mi&quot;, &quot;ma&quot;, &quot;bi&quot;) to calculate all.</td>
</tr>
</tbody>
</table>

Value

Returns the outcome of the equation presented in the paper for the method specific (measure)

Note

version 1.0.0
compress_ids

Author(s)
Tore Opsahl; http://toreopsahl.com

References

Examples
```r
## Generate a random graph
density: 300/(100*99)=0.03030303;
#this should be average from random samples
rg <- rg_w(nodes=100,arcs=300,weights=1:10)

## Run clustering function
clustering_w(rg)
```

compress_ids  Remove non-active nodes from one-mode/two-mode/longitudinal networks

Description
The compress_ids function removes non-active nodes from one-mode/two-mode/longitudinal networks.

Usage
```
compress_ids(net,type=NULL)
```

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>net</td>
<td>A network in an edgelist or matrix format. See as.tnet</td>
</tr>
<tr>
<td>type</td>
<td>See as.tnet</td>
</tr>
</tbody>
</table>

Value
Returns a list with either 2 or 3 objects. The first one is the network with the compressed id. The second object is the translation table between the original node identification numbers and the newly assigned. For two-mode networks, the second object is the translation table of the primary nodes, and the third object is the translation table for the secondary nodes.

Note
version 1.0.0
Description

This dataset contains two intra-organizational networks from a consulting company (46 employees). These networks was used by Cross and Parker (2004).

In the first network, the ties are differentiated on a scale from 0 to 5 in terms of frequency of information or advice requests ("Please indicate how often you have turned to this person for information or advice on work-related topics in the past three months"). 0: I Do Not Know This Person; 1: Never; 2: Seldom; 3: Sometimes; 4: Often; and 5: Very Often.

In the second network, ties are differentiated in terms of the value placed on the information or advice received ("For each person in the list below, please show how strongly you agree or disagree...")
with the following statement: In general, this person has expertise in areas that are important in the kind of work I do."). The weights in this network is also based on a scale from 0 to 5: 0: I Do Not Know This Person; 1: Strongly Disagree; 2: Disagree; 3: Neutral; 4: Agree; and 5: Strongly Agree.

In addition to the relational data, the dataset also contains information about the people (nodal attributes). The following attributes are known: the organisational level (1 Research Assistant; 2: Junior Consultant; 3: Senior Consultant; 4: Managing Consultant; 5: Partner), gender (1: male; 2: female), region (1: Europe; 2: USA), and location (1: Boston; 2: London; 3: Paris; 4: Rome; 5: Madrid; 6: Oslo; 7: Copenhagen).

See http://toreopsahl.com/datasets/

Usage

Cross.Parker.Consulting.net.info
Cross.Parker.Consulting.net.value
Cross.Parker.Consulting.node.gender
Cross.Parker.Consulting.node.location
Cross.Parker.Consulting.node.orglevel
Cross.Parker.Consulting.node.region

Format

The networks are data frames with three columns. The first column is the id of the sender, the second column is the id of the receiver, and the third column is the weight of the tie. The nodal attributes are vectors.

References

http://toreopsahl.com/datasets/

Cross.Parker.Manufacturing.net.info

Intra-organisational networks

Description

This dataset contains two intra-organizational networks from a research team in a manufacturing company (77 employees). These networks was used by Cross and Parker (2004).

In the first network, the ties among the researchers are differentiated in terms of advice ("Please indicate the extent to which the people listed below provide you with information you use to accomplish your work"). The weights are based on the following scale: 0: I Do Not Know This Person/I Have Never Met this Person; 1: Very Infrequently; 2: Infrequently; 3: Somewhat Infrequently; 4: Somewhat Frequently; 5: Frequently; and 6: Very Frequently.
The second network is based on the employees’ awareness of each others’ knowledge and skills (“I understand this person’s knowledge and skills. This does not necessarily mean that I have these skills or am knowledgeable in these domains but that I understand what skills this person has and domains they are knowledgeable in”). The weight scale in this network is: 0: I Do Not Know This Person/I Have Never Met this Person; 1: Strongly Disagree; 2: Disagree; 3: Somewhat Disagree; 4: Somewhat Agree; 5: Agree; and 6: Strongly Agree.

In addition to the relational data, the dataset also contains information about the people (nodal attributes). The following attributes are known: location (1: Paris; 2: Frankfurt; 3: Warsaw; 4: Geneva), tenure (1: 1-12 months; 2: 13-36 months; 3: 37-60 months; 4: 61+ months) and the organisational level (1: Global Dept Manager; 2: Local Dept Manager; 3: Project Leader; 4: Researcher).

See http://toreopsahl.com/datasets/

Usage

Cross.Parker.Manufacturing.net.info
Cross.Parker.Manufacturing.net.aware
Cross.Parker.Manufacturing.node.location
Cross.Parker.Manufacturing.node.orglevel
Cross.Parker.Manufacturing.node.tenure

Format

The networks are data frames with three columns. The first column is the id of the sender; the second column is the id of the receiver; and the third column is the weight of the tie. The nodal attributes are vectors.

References

http://toreopsahl.com/datasets/

Davis.Southern.women.2mode

Davis’ Southern Women network

Description

This dataset was collected by Davis and colleague in the 1930s. It contains the observed attendance by 18 Southern women (primary nodes) at 14 social events (secondary nodes). This has been projected onto a co-occurance one-mode network, and a one-mode network based on Newman's (2001) method.
Usage

Davis.Southern.women.2mode
Davis.Southern.women.1mode.Cooccurrence
Davis.Southern.women.1mode.Newman

Format

The two-mode network is a data frame with two columns (primary nodes and secondary nodes, respectively). The one-mode networks are data frames with three columns: the first column is the id of the sender; the second column is the id of the receiver; and the column third is the weight of the tie.

Author(s)

Tore Opsahl; http://toreopsahl.com

References

http://toreopsahl.com/datasets/

Description

This function calculates two degree measures: the number of contacts that a node is connected to, and the sum of weights on ties originating from a node (strength).

Usage

degree_tm(net, measure=c("degree","output"))

Arguments

net A two-mode network
measure specifies which measures should be calculated

Value

Returns a data.frame with two or three columns: the first column contains the nodes’ ids, and the remaining columns contain the scores of the measures specified in the measure-parameter.

Note

version 1.0.0
**degree_w**

**Description**

This function calculates two degree measures: the number of contacts that a node is connected to, and the sum of weights on ties originating from a node (out-strength). To calculate the reverse (in-degree, in-strength), specify type="in".

**Usage**

```r
degree_w(net, measure=c("degree","output"), type="out", alpha=1)
```

**Arguments**

- `net` A weighted edgelist
- `measure` specifies which measures should be calculated
- `type` shall out- or in-measures be calculated? Default is out. For undirected networks, this setting is irrelevant, but must be specified.
- `alpha` sets the alpha parameter in the generalised measures from Opsahl, T., Agneessens, F., Skvoretz, J., 2010. Node Centrality in Weighted Networks: Generalizing Degree and Shortest Paths. Social Networks. If this parameter is set to 1 (default), the sum of tie weights is used. This measure simply use the tie weights and disregards the number of nodes on the paths.

**Value**

Returns a data.frame with two or more columns: the first column contains the nodes’ ids, and the remaining columns contain the scores of the measures specified in the measure-parameter.
**dichotomise_tm**

**Note**

version 1.0.0

**Author(s)**

Tore Opsahl; http://toreopsahl.com

**References**


**Examples**

```r
## Load sample data
network <- rbind(
  c(1, 2, 4),
  c(1, 3, 2),
  c(2, 1, 4),
  c(2, 3, 4),
  c(2, 4, 1),
  c(2, 5, 2),
  c(3, 1, 2),
  c(3, 2, 4),
  c(4, 2, 1),
  c(5, 2, 2),
  c(5, 6, 1),
  c(6, 5, 1))

## Run the programme
degree_w(network)
```

---

**dichotomise_tm**

*Dichotomise a weighted two-mode network into a binary two-mode network*

**Description**

The dichotomise function creates a binary two-mode network from a weighted edgelist.

**Usage**

```r
dichotomise_tm(net, GT=0)
```

**Arguments**

- `net` A weighted two-mode network
- `GT` the cut-off parameter. Default is set to 0, so edges/arcs with a weight greater than 0 is set to 1.
**dichotomise_w**

**Value**

Returns the edgelist with edges below the cut-off removed, and all weights equal to 1.

**Note**

version 1.0.0

**Author(s)**

Tore Opsahl; http://toreopsahl.com

**References**


**Examples**

```r
## Load sample data
sample <- cbind(
  i=c(1,1,2,2,2,3,3,4,5,5,6),
  p=c(1,2,1,3,4,2,3,4,3,5,5),
  w=c(3,5,6,1,2,6,2,1,3,1,2))

## Run the programme
dichotomise_tm(sample, GT=2)
```

---

**dichotomise_w**

*Dichotomise a weighted one-mode network into a binary one-mode network*

**Description**

The dichotomise function creates a binary one-mode network from a weighted edgelist.

**Usage**

```r
dichotomise_w(net, GT=0)
```

**Arguments**

- **net** A weighted one-mode network
- **GT** the cut-off parameter. Default is set to 0, so edges/arcs with a weight greater than 0 is set to 1.

**Value**

Returns the edgelist with edges below the cut-off removed, and all weights equal to 1.
**distance_tm**

**Note**

version 1.0.0

**Author(s)**

Tore Opsahl; http://toreopsahl.com

**References**


**Examples**

```r
## Load sample data
distance_tm(sample, projection.method="sum")
```

**Description**

The shortest path length, or geodesic distance, between two nodes in a binary network is the minimum number of steps you need to make to go from one of them to the other. See the `distance_w()` function for more details.

**Usage**

```r
distance_tm(net, projection.method="sum", gonly=TRUE, subsample=1, seed=NULL)
```
Arguments

- **net**: A two-mode network
- **projection.method**: The way the two-mode network is projected. The sum method defines tie weights as the number of common nodes (e.g., events, projects etc) that two individuals had contact through. In certain cases, "Newman" might be better. See the projecting_tm-function.
- **gconly**: logical, whether the function should only be calculated for the giant component. Default is TRUE.
- **subsample**: Whether a only a subset of starting nodes should be used when calculating the measure. This is particularly useful when running out of memory analysing large networks. If it is set to 1, all distances are analysed. If it set to a value below one, this is roughly the proportion of starting noe that will be analysed. If it is set to an integer greater than 1, this number of starting nodes that will be analysed.
- **seed**: If a subset of starting nodes is analysed, by setting this parameter, the results are reproducible.

Value

Returns a distance matrix.

Note

version 1.0.0

Author(s)

Tore Opsahl; http://toreopsahl.com

References


Examples

```r
# Load networks
net <- cbind(
  i=c(1,1,2,2,3,3,4,5,5,6),
  p=c(1,2,1,3,4,2,3,4,3,5),
  w=c(3,5,6,1,2,6,2,1,3,1,2))

# Run the function
distance_tm(net)
```
Description

The shortest path length, or geodesic distance, between two nodes in a binary network is the minimum number of steps you need to make to go from one of them to the other. This distance is the quickest connection between nodes when all ties are the same. However, in a weighted network, all ties are not the same. See http://toreopsahl.com/2009/01/09/average-shortest-distance-in-weighted-networks/ for more details.

Usage

distance_w(net, directed=NULL, gonly=TRUE, subsample=1, seed=NULL)

Arguments

- **net**: A weighted edgelist
- **directed**: logical, whether the network is directed or undirected. Default is NULL, this means that the function checks whether the edgelist is directed or not.
- **gonly**: logical, whether the function should only be calculated for the giant component. Default is TRUE.
- **subsample**: Whether a only a subset of starting nodes should used when calculating the measure. This is particularly useful when running out of memory analysing large networks. If it is set to 1, all distances are analysed. If it set to a value below one, this is roughly the proportion of starting noes that will be analysed. If it is set to an interger greater than 1, this number of starting nodes that will be analysed.
- **seed**: If a subset of starting nodes is analysed, by setting this parameter, the results are reproducable.

Value

Returns a distance matrix.

Note

version 1.0.0

Author(s)

Tore Opsahl; http://toreopsahl.com

References

Examples

```r
## Load sample data
sample <- rbind(
  c(1,2,8),
  c(1,4,1),
  c(2,1,8),
  c(2,3,6),
  c(3,2,6),
  c(3,4,10),
  c(4,1,1),
  c(4,3,10))

## Run the programme
distance_w(sample)
```

Description

Freeman’s EIES networks (Freeman, 1979) was the main network used in Wasserman and Faust (1994). This dataset was collected in 1978 and contains three networks of researchers working on social network analysis. The first network contains the personal relationships among 48 of the researchers at the beginning of the study (time 1). The second network is the personal relationship at the end of the study (time 2). In these two networks, all ties have a weight between 0 and 4. 4 represents a close personal friend of the researcher’s; 3 represents a friend; 2 represents a person the researcher has met; 1 represents a person the researcher has heard of, but not met; and 0 represents a person unknown to the researcher. The third network is different. It is a matrix with the number of messages sent among 32 of the researchers that used an electronic communication tool (frequency matrix).

There are two pieces of information about each of the 32 researchers that were part of the third network (nodal attributes): the main disciplinary affiliation (1: sociology; 2: anthropology; 3: mathematics or statistics; and 4: others) and the number of citations each researcher had in the Social Science Citation Index in 1978.

See http://toreopsahl.com/datasets/

Usage

- `Freemans.EIES.net.1.n48`
- `Freemans.EIES.net.2.n48`
- `Freemans.EIES.net.3.n32`
- `Freemans.EIES.node.Name.n32`
- `Freemans.EIES.node.Citations.n32`
- `Freemans.EIES.node.Discipline.n32`
The networks are data frames with three columns. The first column is the id of the sender, the second column is the id of the receiver; and the third column is the weight of the tie. The attributes are vectors.

References

See http://toreopsahl.com/datasets/
Description

This is the co-authorship network of scientists based on preprints posted to Condensed Matter section of arXiv E-Print Archive between 1995 and 1999.

This network can be classified as a two-mode or affiliation network since there are two types of "nodes" (authors and papers) and connections exist only between different types of nodes. An author is connected to a paper if her or his name appeared on it.

Few network measures exist for two-mode networks, and therefore, these networks are often projected onto a one-mode (only one type of nodes) network by selecting one of the types of nodes and linking two nodes if they were connected to the same node (of the other kind).
Traditionally, the ties in projected one-mode networks do not have weights. Recent empirical studies of two-mode networks has created a weighted network by defining the weights as the number of co-occurrences (e.g., the number of papers that two authors had collaborated on).

This method was refined by Newman (2001). He argued that smaller collaborations created stronger social bonds among scientists than larger ones. Therefore, he extended this procedure and proposed to define weights among the nodes use the following formula:

\[ w_{ij} = \sum_p \frac{1}{N_p - 1} \]

where \( w_{ij} \) is the weight between node i and node j, \( p \) is the papers that they have collaborated on, and \( N_p \) is the number of authors on a paper. This implies that if two authors only write a single paper together with no other co-authors, they get a weight of 1. However, if they have a co-author, the weight on the tie between them is 0.5. If two authors have written two papers together without any co-author, the weight of their tie would be 2. A more complicated example is the tie between two authors who have written two papers together: one without any other co-author and one with one co-author. The first paper would give their tie a weight of 1, and the second tie would add 0.5 to the weight of this tie. Therefore, the weight is 1.5.

Note: This method has been explained in more detail in the following post:

This is the two-mode network. See http://toreopsahl.com/datasets/

**Usage**

Newman.Conmat.95.99.net.2mode
Newman.Conmat.95.99.net.1mode.wNewman

**Format**

The two-mode network is a data frame with two columns. The first column is the id of authors and the second column is the id of papers. The one-mode network is a data frame with three columns. The first two columns are ids of the authors, and the third column is the weight of the tie. This is calculated based on Newman’s (2001) method for defining tie weights. See the projecting_tm-function.

**References**

See http://toreopsahl.com/datasets/
Description

This network is the Facebook-like Social Network-dataset used in my Ph.D. thesis. This network has also been described in Patterns and Dynamics of Users’ Behaviour and Interaction: Network Analysis of an Online Community and used in Prominence and control: The weighted rich-club effect and Clustering in weighted networks. The network originates from a virtual community among students at University of California, Irvine. The edgelists include the users that sent or received at least one message during that period (1,899). A total number of 59,835 online messages were sent among over 20,296 directed ties.

Usage

OnlineSocialNetwork.n1899.net
OnlineSocialNetwork.n1899.lnet

Format

OnlineSocialNetwork.n1899.net: A data frame with three columns. The first column is the id of the sender, the second column is the id of the receiver, and the third column is the weight of the tie.
OnlineSocialNetwork.n1899.lnet: A data frame with four columns. The first column is the timestamp, the second column is the id of the sender, the third column is the id of the receiver, and the fourth column is the weight of the tie (always 1).

Author(s)

Tore Opsahl; http://toreopsahl.com

References

http://toreopsahl.com/datasets/

projecting_tm  Projecting binary and weighted two-mode networks onto weighted one-mode networks.

Description

This function is the implementation of the procedure outlined on http://toreopsahl.com/2009/05/01/projecting-two-mode-networks-onto-weighted-one-mode-networks/

Usage

projecting_tm(net, method = "sum")
**Arguments**

- **net**: A two-mode edgelist
- **method**: The method-switch control the method used to calculate the weights.  
  - binary sets all weights to 1  
  - sum sets the weights to the number of cooccurrences  

**Value**

Returns a one-mode network

**Note**

version 1.0.0

**Author(s)**

Tore Opsahl; http://toreopsahl.com

**References**


**Examples**

```r
## define two-mode network
two.mode.net <- cbind(i=c(1,1,2,2,2,2,3,4,5,5,5,5), p=c(1,2,1,2,3,4,5,2,3,4,5,6,6))

## Run the function
projecting_tm(two.mode.net, method="Newman")
```

---

**Description**


**Usage**

`reinforcement_tm(net)`
rg_reshuffling_l

**Arguments**

- `net`  
  A two-mode network

**Value**

Returns the score

**Note**

version 1.0.0

**Author(s)**

Tore Opsahl; http://toreopsahl.com

**References**


**Examples**

```r
## Load sample data
net <- rg_tm(10, 7, 0.4)

## Run the programme
reinforcement_tm(net)
```

---

**rg_reshuffling_l  Reshuffling a longitudinal network**

**Description**

This function reshuffles a longitudinal dataset.

**Usage**

`rg_reshuffling_l(net, keep.i = FALSE, keep.j = FALSE, seed = NULL)`

**Arguments**

- `net`  
  Longitudinal network
- `keep.i`  
  Whether or not the tie creators should be maintained
- `keep.j`  
  Whether or not the tie receivers should be maintained
- `seed`  
  the random seed. If you want it to have reproducible result, set using an integer

**Value**

Returns a reshuffled longitudinal network
**rg_reshuffling_tm**  
Reshuffle of a binary two-mode network

**Description**

This function randomly resuffles a binary two-mode edgelist whilst maintaining each nodes’ degree (both primary and secondary nodes).

**Usage**

```r
rg_reshuffling_tm(net, option="links", seed=NULL)
```

**Arguments**

- `net` A two-mode network
- `option` Either link reshuffling (option="links") or weight reshuffling (option="weights"), see Opsahl et al. (2008).
- `seed` seed for random generator, set if you want random yet reproducible results.
Value

Returns a binary two-mode edgelist.

Note

version 1.0.0

Author(s)

Tore Opsahl; http://toreopsahl.com

References


Examples

```r
## Load data
net <- rg_tm(10, 8, 0.4)

## Run the function on a subset
rg_reshuffling_tm(net, seed=1)
```

Description

This function randomly resuffles a weighted edgelist.

Usage

```r
rg_reshuffling_w(net, option="weights", directed=NULL, seed=NULL)
```

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>net</td>
<td>A weighted edgelist</td>
</tr>
<tr>
<td>option</td>
<td>what should be reshuffled: 1) weights (default): randomly assigns the weights to the edges; 2) links: maintain the degree distribution, but changes the contacts randomly.</td>
</tr>
<tr>
<td>directed</td>
<td>logical: is the network directed or undirected. Default: NULL</td>
</tr>
<tr>
<td>seed</td>
<td>seed for random generator, set if you want random yet reproducible results.</td>
</tr>
</tbody>
</table>
Value

Returns a randomised (reshuffled) network.

Note

version 1.0.0

Author(s)

Tore Opsahl; http://toreopsahl.com

References


Examples

```r
## Load sample data
sampledata<-rbind(
  c(1,2,4),
  c(1,3,2),
  c(2,1,4),
  c(2,3,4),
  c(2,4,1),
  c(2,5,2),
  c(3,1,2),
  c(3,2,4),
  c(4,2,1),
  c(5,2,2),
  c(5,6,1),
  c(6,5,1));

## Run the function
rg_reshuffling_w(sampledata, option="weights", directed=FALSE)
```

## rg_tm

Random binary and weighted two-mode network

Description

Creates classical random binary and weighted two-mode networks.
Usage

rg_tm(ni=100,np=100,ties=300,weights=1,seed=NULL)

Arguments

ni Number of nodes in the first set
np Number of nodes in the second set
ties Number of ties; if this value is between 0 and 1, a random network where each
tie is based on this probability will be produced
weights A tie weight vector to be randomly sampled. If set to 1 (default), all tie weights
will be 1, and hence a binary two-mode network will be created.
seed the random seed. If you want it to be non-reproducible, use NULL otherwise,
use a number

Value

Returns a random two-mode network

Note

version 1.0.0

Author(s)

Tore Opsahl; http://toreopsahl.com

References

Tore Opsahl. Triadic closure in two-mode networks: Redefining the global and local clustering
coefficients. arXiv:1006.0887

Examples

```r
## Run the programme
rg_tm(ni=1000,np=100,ties=300,weights=1)
```

Description

This function creates a classical random network with random edge weights.

Usage

rg_w(nodes=100,arcs=300,weights=1,directed=TRUE,seed=NULL)

Random weighted network generator
shrink_to_weighted_network

Arguments

- **nodes**: number of nodes
- **arcs**: number of arcs; if this value is between 0 and 1, a random network where each tie is based on this probability will be produced
- **weights**: A tie weight vector to be randomly sampled.
- **directed**: whether you want a directed or undirected network, values TRUE or FALSE
- **seed**: the random seed. If you want it to be non-reproducible, use NULL otherwise, use a number

Value

Returns a one-mode network with random weights.

Note

version 1.0.0

Author(s)

Tore Opsahl; http://toreopsahl.com

References

http://toreopsahl.com/tnet/

Examples

```r
gw(nodes=10, arcs=30, directed=FALSE, seed=1)
```

Description

This function creates a weighted edgelist from a list of edges where a duplicate means an increase in the weight.

Usage

```r
shrink_to_weighted_network(net)
```
Arguments

net can use both undirected and directed edgelist in the following format (sender.id receiver.id):

1 2
1 2
1 2
1 3
1 3

Value

Returns a weighted one-mode network, e.g.,

1 2 4
1 3 2

Note

version 1.0.0

Author(s)

Tore Opsahl; http://toreopsahl.com

References


Examples

```r
## Load sample data
sample <- rbind(
c(1,2),
c(1,2),
c(1,2),
c(1,2),
c(1,3),
c(1,3),
c(2,1),
c(2,1),
c(2,1),
c(2,3),
c(2,3),
c(2,3),
c(2,3),
c(2,4),
c(2,5),
c(2,5),
c(3,1),
c(3,1),
```
Description

The symmetrise_w-function creates an undirected one-mode network from a directed one-mode network.

Usage

symmetrise_w(net, method="MAX")

Arguments

net A one-mode network
method the method used to decide the weight of the undirected edge. It can be: "MAX" sets the weight to the maximum of the weight(s) of the arc(s) "MIN" sets the weight to the minimum of the weight(s) of the arc(s) "AMEAN" sets the weight to the average (arithmetic mean) of the weight(s) of the arc(s) "SUM" sets the weight to the sum of the weight(s) of the arc(s) "PROD" sets the weight to the product of the weight(s) of the arc(s) "DIFF" sets the weight to the absolute difference between the weight(s) of the arc(s)

Value

Returns the undirected network

Note

version 1.0.0

Author(s)

Tore Opsahl; http://toreopsahl.com
tnet_igraph

Exports a tnet network to an igraph object

Description

The tnet_igraph function creates an igraph object from a tnet network.

Usage

```r
tnet_igraph(net, type=NULL, directed=NULL)
```

Arguments

- `net`: A tnet network
- `type`: Type of tnet network, see `as.tnet`.
- `directed`: If a one-mode networks, this can be set to avoid testing whether the network is directed.

Value

Returns the igraph object.

Note

version 1.0.0

References


Examples

```r
## Load sample data
sample <- rbind(
c(1,2,2),
c(1,3,2),
c(2,1,4),
c(2,3,4),
c(2,4,1),
c(2,5,2),
c(3,1,2),
c(3,2,4),
c(5,2,2),
c(5,6,1))

## Run the programme
symmetrise_w(sample, method="MAX")
```
tnet_ucinet

Author(s)
Tore Opsahl; http://toreopsahl.com

References
http://toreopsahl.com/

Examples

```r
## Load sample data
sample <- rbind(
c(1,2,4),
c(1,3,2),
c(2,1,4),
c(2,3,4),
c(2,4,1),
c(2,5,2),
c(3,1,2),
c(3,2,4),
c(4,2,1),
c(5,2,2),
c(5,6,1),
c(6,5,1))

## Run the programme
tnet_igraph(sample, type="weighted one-mode tnet")
```

---

### tnet_ucinet

Exports a tnet network to a DL file for UCINET

---

**Description**

The tnet_sna function creates a DL file which can easily be imported into UCINET.

**Usage**

```r
tnet_ucinet(net, type=NULL, file=NULL)
```

**Arguments**

- `net`: A tnet network
- `type`: type of tnet network, see as.tnet.
- `file`: filename of output file. If this is set to NULL, a file is created in the working directory with the current time (e.g., tnet_ucinet_network-2011-04-10_150817.dl).

**Value**

Writes a UCINET dl file.
USairport.n500.net

The network among the 500 busiest US commercial airports.

Description

The nodes in this network is the 500 busiest commercial airports in the United States. A tie exists between two airports if a flight was scheduled between them in 2002. The weights corresponds to the number of seats available on the scheduled flights. Even thought this type of networks is directed by nature as a flight is scheduled from one airport and to another, the networks are highly symmetric (Barrat et al., 2004). Therefore, the version of this network is undirected (i.e., the weight of the tie from one airport towards another is equal to the weight of the reciprocal tie). This network was obtained from the Complex Networks Collaboratory’s website

See http://toreopsahl.com/datasets/

Note

version 1.0.0

Author(s)

Tore Opsahl; http://toreopsahl.com

References

http://toreopsahl.com/

Examples

```r
## Load sample data
sample <- rbind(
c(1,2,4),  
c(1,3,2),  
c(2,1,4),  
c(2,3,4),  
c(2,4,1),  
c(2,5,2),  
c(3,1,2),  
c(3,2,4),  
c(4,2,1),  
c(5,2,2),  
c(5,6,1),  
c(6,5,1))

## Run the programme
tnet_ucinet(sample, type="weighted one-mode tnet")
```
Usage
weighted_richclub_local_w

Format
A data frame with three columns. The first two columns are the nodes’ ids, and the third column is the weight of the tie.

References
See http://toreopsahl.com/datasets/

Description
This function calculates the local weighted rich-club coefficient proposed in Opsahl, T., 2008. Local weighted rich-club measure.
http://toreopsahl.com/2008/12/26/local-weighted-rich-club-measure/

Usage
weighted_richclub_local_w(net, prominence)

Arguments
net A weighted edgelist
prominence A vector with 1 denoting prominent, and 0 non-prominent. This list must be as long as the highest node id number.

Value
Returns a table with the fraction of phi(observed) over phi(null) for each k or s in the dataset.

Note
version 1.0.0

Author(s)
Tore Opsahl; http://toreopsahl.com
References

Opsahl et al., 2008. Prominence and control: The weighted rich-club effect. PRL 101

Examples

```r
## Load sample data
sample <- cbind(i=c(1,1,2,2,2,3,3,4,5,5,6), j=c(2,3,1,3,4,5,1,2,2,2,6,5), w=c(4,2,4,1,2,2,4,1,2,1,1))
prominence <- c(1,1,1,0,0,0)

## Run the function
weighted_richclub_local_w(sample, prominence)
```

---

**weighted_richclub_tm**  *The weighted rich-club effect (two-mode networks)*

**Description**

This function calculates the weighted rich-club coefficient proposed in Opsahl, T., Colizza, V., Panzarasa, P., Ramasco, J.J., 2008. Prominence and control: The weighted rich-club effect. PRL 101. It incorporates two extensions explained in this blog post http://toreopsahl.com/2009/05/29/weighted-rich-club-effect-a-more-appropriate-null-model-for-scientific-collaboration-networks/: 1) a new way of reshuffling (two-mode link reshuffling; 2) calculating significance levels if there are more than 100 random networks (see my PhD thesis; http://toreopsahl.com/publications/thesis/)

**Usage**

```r
weighted_richclub_tm(net, NR=1000, seed=NULL, projection.method="Newman", nbins=30)
```

**Arguments**

- **net** A binary two-mode edgelist
- **NR** number of random networks used.
- **seed** the random generators seed, used to produce random yet reproducible results.
- **projection.method** the method used to project the two-mode network to a weighted one-mode network: either "sum" or "Newman"
- **nbins** the number of bins in the output

**Value**

Returns a table with the fraction of \( \phi(\text{observed}) \) over \( \phi(\text{null}) \). Nbins controls the number of rows.
Note

version 1.0.0

Author(s)

Tore Opsahl; http://toreopsahl.com

References

Opsahl et al., 2008. Prominence and control: The weighted rich-club effect. PRL 101
http://toreopsahl.com/2009/05/29/weighted-rich-club-effect-a-more-appropriate-null-model-for-scientific-
collaboration-networks/

Examples

```r
## Load data
data(tnet)

## Run the function on a subset
weighted_richclub_tm(Newman.Conmat.95.99.net.2mode[1:100[,] , NR=10)
```

---

### weighted_richclub_w

**The weighted rich-club effect**

Description

This function calculates the weighted rich-club coefficient proposed in Opsahl, T., Colizza, V.,
101.

Usage

```r
weighted_richclub_w(net, rich="k", reshuffle="weights", NR=1000, nbins=30, seed=NULL, directed=NULL)
```

Arguments

- **net**: A weighted edgelist
- **rich**: specifies the richness parameter, either "k" or "s".
- **reshuffle**: specifies the reshuffling procedure used, either "weights" or "links".
- **NR**: number of random networks used.
- **nbins**: the number of bins in the output
- **seed**: the random generators seed, used to produce random yet reproducible results.
- **directed**: logical parameter: whether the network is directed or undirected.
Value

Returns a table with the fraction of phi(observable) over phi(null) for each k or s in the dataset.

Note

version 1.0.0

Author(s)

Tore Opsahl; http://toreopsahl.com

References

Opsahl et al., 2008. Prominence and control: The weighted rich-club effect. PRL 101

Examples

```r
## Load sample data
sample <- cbind(
  i=c(1,1,2,2,2,3,3,4,5,5,6),
  j=c(2,3,1,3,4,5,1,2,2,6,5),
  w=c(4,2,4,4,1,2,2,4,1,2,1))

## Run the function
weighted_richclub_w(sample, rich="k", reshuffle="weights")
```
Index

*Topic datasets

- celegans.n306.net, 8
- Cross.Parker.Consulting.net.info, 16
- Cross.Parker.Manufacturing.net.info, 17
- Davis.Southern.women.2mode, 18
- Freeman.s.EIES.net.1.n48, 26
- Newman.Condmat.95.99.net.2mode, 28
- OnlineSocialNetwork.n1899.net, 29
- USairport.n500.net, 42

Cross.Parker.Consulting.net.info
  - (Cross.Parker.Consulting.net.info), 16

Cross.Parker.Consulting.net.value
  - (Cross.Parker.Consulting.net.info), 16

Cross.Parker.Consulting.node.gender
  - (Cross.Parker.Consulting.net.info), 16

Cross.Parker.Consulting.node.location
  - (Cross.Parker.Consulting.net.info), 16

Cross.Parker.Consulting.node.orglevel
  - (Cross.Parker.Consulting.net.info), 16

Cross.Parker.Consulting.node.region
  - (Cross.Parker.Consulting.net.info), 16

OnlineSocialNetwork.n1899.1net
  - (OnlineSocialNetwork.n1899.net), 29

add_window_l, 4
as.static.tnet, 5
as.tnet, 6

betweenness_w, 7

celegans.n306.net, 8
closeness_w, 9
clustering_local_tm, 10
clustering_local_w, 11
clustering_tm, 13
clustering_w, 14
compress_ids, 15

Cross.Parker.Consulting.net.info, 16
Cross.Parker.Manufacturing.net.aware
  - (Cross.Parker.Manufacturing.net.info), 17
Cross.Parker.Manufacturing.net.info, 17
Cross.Parker.Manufacturing.node.location
  - (Cross.Parker.Manufacturing.net.info), 17
Cross.Parker.Manufacturing.node.orglevel
  - (Cross.Parker.Manufacturing.net.info), 17
Cross.Parker.Manufacturing.node.tenure
  - (Cross.Parker.Manufacturing.net.info), 17

Davis.Southern.women.1mode.Coooccurrence
  - (Davis.Southern.women.2mode), 18

Davis.Southern.women.1mode.Newman
  - (Davis.Southern.women.2mode), 18

Davis.Southern.women.2mode, 18
degree_tm, 19
degree_w, 20
dichotomise_tm, 21
dichotomise_w, 22
distance_tm, 23
distance_w, 25

Freeman.s.EIES.net.1.n48, 26
Freemans.EIES.net.2.n48
(Freemans.EIES.net.1.n48), 26
Freemans.EIES.net.3.n32
(Freemans.EIES.net.1.n48), 26
Freemans.EIES.node.Citations.n32
(Freemans.EIES.net.1.n48), 26
Freemans.EIES.node.Discipline.n32
(Freemans.EIES.net.1.n48), 26
Freemans.EIES.node.Name.n32
(Freemans.EIES.net.1.n48), 26

growth_l, 27

Newman.Condmat.95.99.net.1mode.wNewman
(Newman.Condmat.95.99.net.2mode), 28
Newman.Condmat.95.99.net.2mode, 28

Onlinesocialnetwork.n1899.net, 29

projecting_tm, 30

reinforcement_tm, 31
rg_reshuffling_l, 32
rg_reshuffling_tm, 33
rg_reshuffling_w, 34
rg_tm, 35
rg_w, 36

shrink_to_weighted_network, 37
symmetrise_w, 39

tnet (tnet-package), 2
tnet-package, 2
tnet_igraph, 40
tnet_ucinet, 41

USairport.n500.net, 42

weighted_richclub_local_w, 43
weighted_richclub_tm, 44
weighted_richclub_w, 45