# Package ‘ITNr’

**March 31, 2023**

**Type** Package

**Title** Analysis of the International Trade Network

**Version** 0.7.0

**Author** Matthew Smith

**Maintainer** Matthew Smith &lt;matt_smith.90@hotmail.co.uk&gt;

**Description** Functions to clean and process international trade data into an international trade network (ITN) are provided. It then provides a set of functions to undertake analysis and plots of the ITN (extract the backbone, centrality, blockmodels, clustering). Examining the key players in the ITN and regional trade patterns.

**Depends** R (&gt;= 2.15.1), network

**License** GPL-3

**Encoding** UTF-8

**Imports** stats, circlize, graphics, RColorBrewer, xergm.common, reshape2, maps, blockmodeling, igraph, utils, dplyr, plyr, cowplot, ggplot2, GGally, fastmatch, intergraph, sna, tnet, WDI, networkDynamic

**LazyData** true

**RoxygenNote** 7.2.1

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2023-03-31 14:10:11 UTC

## R topics documented:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>abs_diff_mat</td>
<td>2</td>
</tr>
<tr>
<td>cap_lat_lon</td>
<td>3</td>
</tr>
<tr>
<td>Contradrclean</td>
<td>3</td>
</tr>
<tr>
<td>core_periphery_weighted</td>
<td>4</td>
</tr>
<tr>
<td>ei_group</td>
<td>5</td>
</tr>
<tr>
<td>ei_ind</td>
<td>5</td>
</tr>
<tr>
<td>ei_network</td>
<td>6</td>
</tr>
</tbody>
</table>
**Description**

This takes a dataframe of node attributes and convert one into a absolute difference matrix

**Usage**

```
abs_diff_mat(DF, attrname)
```

**Arguments**

- **DF**
  - Dataframe of node attribute
- **attrname**
  - names of the attribute from the dataframe to create the matrix for.
**Value**

Absolute difference matrix

---

**Description**

Dataframe of capital city latitude and longitude coordinates

**Usage**

cap_lat_lon

---

**Comtradrclean**

**Comtradr data clean**

**Description**

This function takes (import) trade data downloaded from comtrade - potentially using the comtradr package, cleans it and transforms it into a network. Adding a number of country level attributes to nodes in the network, including: regional partition, GDP, GDP per capita, GDP growth and FDI. However, it is important to note the limits of using comtradr to construct a network. Firstly when downloading the data with comtradr, you must specify reporters and partners – yet you cannot put “all” for both – only for either reporters or partners. Then for the other you are limited to a character vector of country names, length five or fewer. Therefore, this will not give you a full network. However, this function can be applied to trade data downloaded from UN Comtrade (download csv and read into R as a dataframe), or any other trade data. You just make sure it has the following column names: reporter_iso, partner_iso, trade_value_usd and year. Some dataformats may have different names. Also - it is important to note that this function is for import data.

**Usage**

Comtradrclean(DF, YEAR, threshold, cutoff)

**Arguments**

- **DF**
  - Dataframe of trade data downloaded (potentially using the comtradr package)
- **YEAR**
  - Year
- **threshold**
  - Apply a threshold - TRUE, Extract the backbone - FALSE
- **cutoff**
  - Threshold - cutoff level, Backbone - significance level

**Value**

International Trade Network - igraph object
core_periphery_weighted

Core-Periphery for Weighted Networks

Description

This function implements rich club core-periphery algorithm (Ma & Mondragón, 2015) to identify members of the core and periphery in weighted networks.

Usage

core_periphery_weighted(gs, type)

Arguments

gs International Trade Network - igraph object. Note for networks not produced using ITNr there needs to be a vertex attribute "name" and edge attribute "weight"

type directed/undirected

Value

List - 1.) igraph object with core-periphery results added as a node attribute. 2.) Dataframe of core-periphery results.

References


Examples

require(igraph)

##Create random International Trade Network (igraph object)
ITN<-erdos.renyi.game(50,0.05,directed = TRUE)

##Add edge weights
E(ITN)$weight<-runif(ecount(ITN), 0, 1)

##Add vertex names
V(ITN)$name<-1:vcount(ITN)

##Implement core-periphery algorithm
ITNcp<-core_periphery_weighted(ITN,"directed")
ei_group  

*Group level E\-I Index*

**Description**

This function calculates the E-I Index (External-internal) at the group/attribute level.

**Usage**

```r
ei_group(gs, attrname)
```

**Arguments**

- `gs` : igraph object
- `attrname` : Attribute name

**Value**

Group level results dataframe

**Examples**

```r
require(igraph)
##Create random network (igraph object)
gs<-erdos.renyi.game(75,0.05,directed = TRUE)

##Add vertex names
V(gs)$name<-1:vcount(gs)

## Add an attribute
V(gs)$letters<- rep(LETTERS[1:5],15)

##Calculate the Group E-I Results
EI_GROUP_DATAFRAME<-ei_group(gs,"letters")
```

---

ei_ind  

*Individual/Node level E\-I Index*

**Description**

This function calculates the E-I Index (External-internal) at the individual/node level.

**Usage**

```r
ei_ind(gs, attrname)
```

**Arguments**

- `gs` : igraph object
- `attrname` : Attribute name
ei_network

Network level E-I Index

Description

This function calculates the E-I Index (External-internal) at the network level.

Usage

ei_network(gs, attrname)

Arguments

   gs     igraph object
   attrname     Attribute name

Value

Group level results dataframe

Examples

require(igraph)
##Create random network (igraph object)
gs<-erdos.renyi.game(30,0.05,directed = TRUE)

##Add vertex names
V(gs)$name<-1:vcount(gs)

## Add an attribute
V(gs)$letters<- rep(LETTERS[1:5],6)

##Calculate the Individual E-I Results
EI_IND_DATAFRAME<-ei_ind(gs,"letters")

---

ei_network     Network level E-I Index
Examples

```r
require(igraph)
##Create random network (igraph object)
gs<-erdos.renyi.game(75,0.05,directed = TRUE)

##Add vertex names
V(gs)$name<-1:vcount(gs)

## Add an attribute
V(gs)$letters<- rep(LETTERS[1:5],15)

#Calculate the Group E-I Results
EI_NETWORK<-ei_network(gs,"letters")
```

---

**ELEnet16**

**Electrical Automotive Goods 2016 Network**

---

**Description**

Electrical Automotive Goods 2016 Network. Electrical automotive goods category as defined by Amighini & Gogoni (2014)

**Usage**

ELEnet16

**References**


---

**ELEnetList**

**List of Electrical Automotive Goods Networks (2006-2016)**

---

**Description**

List of Electrical Automotive Goods Networks for 2006 - 2016. Electrical automotive goods category as defined by Amighini & Gogoni (2014)

**Usage**

ELEnetList

**References**

get.backbone

Description

This function extracts the backbone of a network.

Usage

get.backbone(G, alpha, directed = TRUE)

Arguments

G  
igraph network

alpha  
Significance level

directed  
Default is TRUE

Value

Backbone of the network

References


Examples

require(igraph)

##Create a random (directed) network
gs<-erdos.renyi.game(50,0.2,directed = TRUE)

##Add edge weights to the network
E(gs)$weight<-runif(ecount(gs), 0, 1)

##Extract backbone at 0.05 significance level
backbone<-get.backbone(gs,0.1)
**isEmpty**

---

**Description**

This function checks whether data is numeric(0) and returns NA if this is true and the value of the data otherwise.

**Usage**

`isEmpty(x)`

**Arguments**

- **x**: Data

**Value**

NA or the data

---

**ITNadjust**

---

**Description**

This function adjusts ITN matrices so they are the same size.

**Usage**

`ITNadjust(MATlist, j)`

**Arguments**

- **MATlist**: A list of ITN matrices
- **j**: Element of matrix list to compare with others

**Value**

Matrix
Examples

```r
## Create a list of random matrices (of different sizes)
## Labels - letters of alphabet (can represent actor names)
mat1 <- matrix(round(runif(10*10)), 10, 10)
rownames(mat1) <- LETTERS[1:10]
colnames(mat1) <- LETTERS[1:10]

mat2 <- matrix(round(runif(10*10)), 10, 10)
rownames(mat2) <- LETTERS[10:19]
colnames(mat2) <- LETTERS[10:19]

mat3 <- matrix(round(runif(12*12)), 12, 12)
rownames(mat3) <- LETTERS[15:26]
colnames(mat3) <- LETTERS[15:26]

## Create matrix list
MATlist <- list(mat1, mat2, mat3)

## Adjust matrix 1 so that it has additional rows/actors not
## in the original matrix
mat1adjust <- ITNadjust(MATlist, 1)
```

**ITNblock_plot**

**ITN Blockmodel Plot**

**Description**

This function calculates block membership for the ITN and then plots the network, with node colour according to block membership.

**Usage**

```r
ITNblock_plot(gs, LABEL)
```

**Arguments**

- **gs**: International Trade Network - igraph object
- **LABEL**: Should labels be present - TRUE/FALSE

**Value**

Network Plot - nodes coloured based on block membership
Examples

```r
require(igraph)
require(sna)
require(intergraph)

##Create random International Trade Network (igraph object)
ITN<-erdos.renyi.game(75,0.05,directed = TRUE)

##Add edge weights
E(ITN)$weight<-runif(ecount(ITN), 0, 1)

##Blockmodel plot
block_plot<-ITNblock_plot(ITN,FALSE)
```

Description

This function calculates block membership for ITN and structural equivalence between countries

Usage

```r
ITNblock_se(gs)
```

Arguments

- `gs` International Trade Network - igraph object

Value

List object containing block membership and structural equivalence matrix results

Examples

```r
require(igraph)
require(sna)
require(intergraph)

##Create random International Trade Network (igraph object)
ITN<-erdos.renyi.game(50,0.05,directed = TRUE)

##Add edge weights
E(ITN)$weight<-runif(ecount(ITN), 0, 1)

##Blockmodel & structural equivalence analysis
blockse<-ITNblock_se(ITN)
```
ITNcentrality_binary

**Description**

This function calculates a number of centrality metrics for the binary International Trade Network (ITN).

**Usage**

`ITNcentrality_binary(gs)`

**Arguments**

- `gs` : International Trade Network - igraph object

**Value**

Table of centrality results (dataframe)

**Examples**

```r
require(igraph)
##Create random International Trade Network (igraph object)
ITN<-erdos.renyi.game(75,0.05,directed = TRUE)

##Add edge weights
E(ITN)$weight<-runif(ecount(ITN), 0, 1)

##Add vertex names
V(ITN)$name<-1:vcount(ITN)

##Calculate the centrality measures
ITNCENT<-ITNcentrality(ITN)
```

---

ITNcentrality

**Description**

This function calculates a number of centrality metrics for the weighted International Trade Network (ITN).

**Usage**

`ITNcentrality(gs)`

**Arguments**

- `gs` : International Trade Network - igraph object

**Value**

Table of centrality results (dataframe)

**Examples**

```r
require(igraph)
##Create random International Trade Network (igraph object)
ITN<-erdos.renyi.game(75,0.05,directed = TRUE)

##Add edge weights
E(ITN)$weight<-runif(ecount(ITN), 0, 1)

##Add vertex names
V(ITN)$name<-1:vcount(ITN)

##Calculate the centrality measures
ITNCENT<-ITNcentrality(ITN)
```
Arguments

gs International Trade Network - binary igraph object

Value

Table of centrality results (dataframe)

Examples

```r
require(igraph)
##Create random International Trade Network (igraph object)
ITN<-erdos.renyi.game(75,0.05,directed = TRUE)

##Add vertex names
V(ITN)$name<-1:vcount(ITN)

##Calculate the centrality measures
ITNCENT<-ITNcentrality_binary(ITN)
```

---

ITNcluster  

ITN Cluster

Description

This function calculates cluster membership for ITN

Usage

ITNcluster(gs)

Arguments

gs International Trade Network - igraph object (with region attribute)

Value

Cluster object containing various cluster membership results

Examples

```r
##Load ITN
data(ELEnet16)

##Cluster Analysis
CLU<-ITNcluster(ELEnet16)
```
**ITNcorr**

**ITN Correlation Plot**

**Description**

This function plots the correlation between degree and strength scores.

**Usage**

`ITNcorr(gs)`

**Arguments**

- `gs` International Trade Network - igraph object

**Value**

Correlation plot

**Examples**

```r
corr_plot <- ITNcorr(ITN)
```

---

**ITNdegdist**

**ITN Degree Distribution**

**Description**

This function plots the ITN (probability) degree distribution.

**Usage**

`ITNdegdist(gs)`

**Arguments**

- `gs` International Trade Network - igraph object
**ITNdynamic**

**Value**

Panel of ITN degree distribution plots

**Examples**

```r
require(igraph)

##Create random International Trade Network (igraph object)
ITN<-erdos.renyi.game(75,0.05,directed = TRUE)

##Plot degree distribution
deg_dist_plot<-ITN$degdist(ITN)
```

---

**ITNdynamic**  
*Dynamic ITN*

**Description**

This function produces a dynamic network object for ITNs. It cleans and adjusts the individual networks, so they are the same size. This dynamic network object can then be used to create animations, mapping changes over time and to calculate temporal network statistics.

**Usage**

```r
ITNdynamic(NETlist)
```

**Arguments**

- `NETlist`: A list of International Trade Networks (igraph objects)

**Value**

It returns the Dynamic Network Object

**Examples**

```r
require(igraph)

##Create a set of random International Trade Networks (igraph objects)
##and add vertex names
ITN1<-erdos.renyi.game(75,0.05,directed = TRUE)
V(ITN1)$name<-1:vcount(ITN1)
ITN2<-erdos.renyi.game(100,0.01,directed = TRUE)
V(ITN2)$name<-1:vcount(ITN2)
ITN3<-erdos.renyi.game(55,0.1,directed = TRUE)
V(ITN3)$name<-1:vcount(ITN3)
```
## Create network list
NETlist<-list(ITN1,ITN2,ITN3)

## Create Dynamic Network Object
ITNdyn<-ITNdynamic(NETlist)

---

**ITNhistdegdist**

**ITN Histogram Degree Distribution**

---

**Description**

This function plots the histogram degree distribution for the ITN.

**Usage**

ITNhistdegdist(gs)

**Arguments**

- **gs**: International Trade Network - igraph object

**Value**

Panel of ITN histogram degree distribution plots

**Examples**

```r
require(igraph)

## Create random International Trade Network (igraph object)
ITN<-erdos.renyi.game(75,0.05,directed = TRUE)

## Add edge weights
E(ITN)$weight<-runif(ecount(ITN), 0, 1)

## Plot degree distribution histogram
hist_deg_dist<-ITNhistdegdist(ITN)
```
### Description
The following function produces a plot showing imports (in degree) vs exports (out degree). This allows us to identify whether in the ITN, countries that export high levels also import high levels. The plot can be produced for either weighted or binary import and export ties.

### Usage
\[
\text{ITNimvex}(\text{gs}, \text{weighted})
\]

### Arguments
- **gs**: International Trade Network - igraph object
- **weighted**: TRUE - plot import strength vs export strength. FALSE - Import count Vs export count

### Value
Imports Vs Exports Plot

### Examples
```r
require(igraph)

##Create random International Trade Network (igraph object)
ITN<-erdos.renyi.game(75,0.05,directed = TRUE)

##Add edge weights
E(ITN)$weight<-runif(ecount(ITN), 0, 1)

##Plot binary import vs exports
imvex_plot<-ITNimvex(ITN,FALSE)
```

---

### Description
This function creates a panel of four plots of the ITN for a quick inspection. These include plots: (i) highlighting clusters using the fast greedy algorithm, (ii) node colours for communities detected using the spinglass algorithm, (iii) nodes coloured by regional partition and (iv) with nodes coloured by regional partition and node size based on outdegree centrality.
Usage

ITNplotset(gs)

Arguments

gs International Trade Network - igraph object

Value

Panel of ITN plots

Examples

##Load the network
data(ELEnet16)

##Plot set of network visualisations
ITNplotset(ELEnet16)

---

Description

This function calculates network level properties for the ITN. These include:
- Size (number of nodes)
- Density
- Reciprocity
- Diameter
- Average path length
- Average node strength
- Average Degree
- Betweenness Centralisation
- Closeness Centralisation
- Eigenvector Centralisation
- Out Degree Centralisation
- In Degree Centralisation
- All Degree Centralisation
- Clustering coefficient (transitivity)
- Clustering Weighted
- Region Homophily
- Degree Assortativity

Usage

ITNproperties(gs, weighted)

Arguments

gs International Trade Network - igraph object

weighted TRUE-weighted, FALSE-binary

Value

Table of centrality results (dataframe)
Examples

```r
# Load the network
data(ELEnet16)

# Calculate the network properties
ITNPROP <- ITNproperties(ELEnet16, TRUE)
```

---

**ITNproperties_base**  
*ITN Properties Base*

---

**Description**

This function calculates network level properties for the ITN. These include:  
- Size (number of nodes)  
- Density  
- Reciprocity  
- Diameter  
- Average path length  
- Average node strength  
- Average Degree  
- Betweenness Centralisation  
- Closeness Centralisation  
- Eigenvector Centralisation  
- Out Degree Centralisation  
- In Degree Centralisation  
- All Degree Centralisation  
- Clustering coefficient (transitivity)  
- Clustering Weighted  
- Degree Assortativity

**Usage**

```r
ITNproperties_base(gs, weighted)
```

**Arguments**

- `gs`  
  International Trade Network - igraph object

- `weighted`  
  TRUE-weighted, FALSE-binary

**Value**

Table of centrality results (dataframe)

**Examples**

```r
# Load the network
data(ELEnet16)

# Calculate the network properties
ITNPROP <- ITNproperties_base(ELEnet16, TRUE)
```
ITN_make_plot  Single Clean ITN Plot

Description
This function plots a single/clean ITN

Usage
ITN_make_plot(gs, LABEL, REGION)

Arguments
- gs: International Trade Network - igraph object
- LABEL: Should labels be present - TRUE/FALSE
- REGION: Should nodes be coloured on the basis of region TRUE/FALSE

Value
Panel of ITN plots

Examples

```r
##Load graph
data("ELEnet16")

##Otherwise download data from WITS and create an
##International Trade Network using WITSclean()

##Plot the network - No Label, colour by region
ITN_plot_example<-ITN_make_plot(ELEnet16,FALSE,TRUE)
```

ITN_map_plot  ITN plot on world map

Description
This function plots the international trade network on a world map

Usage
ITN_map_plot(gs)
make_trade_network

Arguments

gs  International Trade Network - igraph object

Value

Plot of the ITN on world map

Examples

```r
require(maps)
##Load the ITN
data(ELEnet16)

## Plot ITN on map - node size based on outdegree
ITN_map_plot(ELEnet16)
```

Description

This function takes (import) trade data and cleans it and transforms it into a network. This function can be applied to trade data downloaded from UN Comtrade (download csv and read into R as a dataframe), or any other trade data. You just make sure it has the following column names: reporter_iso, partner_iso and edge_weight. Some dataformats may have different names. Also - it is important to note that this function is for import data.

Usage

```r
make_trade_network(DF, threshold, cutoff)
```

Arguments

DF  Dataframe of trade data downloaded (potentially using the comtradr package)
threshold  Apply a threshold - TRUE, Extract the backbone - FALSE
cutoff  Threshold - cutoff level, Backbone - significance level

Value

International Trade Network - igraph object
mixing_matrix_igraph  Mixing Matrix

Description
This function calculates the mixing matrix for an igraph object

Usage
mixing_matrix_igraph(gs, attrname)

Arguments
  gs  igraph object.
  attrname  Attribute name (vertex attribute)

Value
Mixing matrix

Examples
require(igraph)
##Create random International Trade Network (igraph object)
gs<-erdos.renyi.game(50,0.05,directed = TRUE)

##Add vertex attributes
V(gs)$LETTER<-rep(LETTERS[1:5],10)

##Add vertex names
V(gs)$name<-1:vcount(gs)

##Calculate mixing matrix
mixing_matrix<-mixing_matrix_igraph(gs,"LETTER")

plot_degree_distribution  Plot Degree Distribution

Description
This function plots degree distribution for any graph

Usage
plot_degree_distribution(graph, a)
receiver_mat

Arguments

  graph  igraph object
  a  mode - "in","out","all

Value

  Panel of ITN degree distribution plots

Examples

require(igraph)
##Create random International Trade Network (igraph object)
ITN<--erdos.renyi.game(75,0.05,directed = TRUE)

##Plot out degree distribution
plot_degree_distribution(ITN,"in")

receiver_mat

Description

  This takes a dataframe of node attributes and convert one into a matrix of receiver attributes

Usage

receiver_mat(DF, attrname)

Arguments

  DF  Dataframe of node attribute
  attrname  names of the attribute from the dataframe to create the matrix for.

Value

  Receiver matrix
region_circle_plot  region_circle_plot

Description
This function creates a chord diagram/circle plot for levels of trade between regional partitions

Usage
region_circle_plot(gs)

Arguments
gs igraph ITN object (with attributes added)

Value
Circle Plot

Examples

##Load graph
data("ELEnet16")

##Create region circle plot
region_circle_plot(ELEnet16)

reorder_df  reorder_df

Description
Reorders the rows of one dataframe according to another vector (id vector)

Usage
reorder_df(df, col_sort, reorder_data)

Arguments
df dataframe to reorder
col_sort column on which the rows will be reordered
reorder_data vector with the new order
**Value**

Reordered dataframe

**Examples**

```
df <- data.frame(a = letters[1:3], b = LETTERS[4:6], c = 7:9)
reorder_data <- c("c", "a", "b")
df_new <- reorder_df(df, "a", reorder_data)
df_new
```

---

**Description**

This function rounds the numeric variables in a dataframe containing numeric and non-numeric data.

**Usage**

```
round_df(x, digits)
```

**Arguments**

- `x` dataframe
- `digits` digits to round to

**Value**

Dataframe with rounded numbers

**Examples**

```
# Create dataframe
ID = c("a", "b", "c", "d", "e")
Value1 = c(3.445662, 6.44566, 8.75551, 1.114522, 1.5551)
Value2 = c(8.2, 1.7, 6.4, 19.45459, 10.34524)
df <- data.frame(ID, Value1, Value2)

dataframe
```

```
# Round to 2 digits
rounddf <- round_df(df, 2)
```
### sender_mat

**Description**

This takes a dataframe of node attributes and convert one into a matrix of sender attributes.

**Usage**

```r
sender_mat(DF, attrname)
```

**Arguments**

- **DF**: Dataframe of node attribute
- **attrname**: names of the attribute from the dataframe to create the matrix for.

**Value**

Sender matrix

### WITSclean

**Description**

This function takes (import) trade data downloaded from WITS, cleans it and transforms it into a network. Adding a number of country level attributes to nodes in the network, including: regional partition, GDP, GDP per capita, GDP growth and FDI.

**Usage**

```r
WITSclean(CSVfile, YEAR, threshold, cutoff)
```

**Arguments**

- **CSVfile**: WITS csv file
- **YEAR**: Year
- **threshold**: Apply a threshold - TRUE, Extract the backbone - FALSE
- **cutoff**: Threshold - cutoff level, Backbone - significance level

**Value**

International Trade Network - igraph object
# Index

## * datasets
- cap_lat_lon, 3
- ELEnet16, 7
- ELEnetList, 7

- abs_diff_mat, 2
- cap_lat_lon, 3
- Comtradrclean, 3
- core_periphery_weighted, 4

- ei_group, 5
- ei_ind, 5
- ei_network, 6
- ELEnet16, 7
- ELEnetList, 7

- get.backbone, 8

- isEmpty, 9
- ITN_make_plot, 20
- ITN_map_plot, 20
- ITNadjust, 9
- ITNblock_plot, 10
- ITNblock_se, 11
- ITNcentrality, 12
- ITNcentrality_binary, 12
- ITNcluster, 13
- ITNcorr, 14
- ITNddegdist, 14
- ITNdynamic, 15
- ITNhistdegdist, 16
- ITNimvex, 17
- ITNplotset, 17
- ITNproperties, 18
- ITNproperties_base, 19

- make_trade_network, 21
- mixing_matrix_igraph, 22

- plot_degree_distribution, 22

- receiver_mat, 23
- region_circle_plot, 24
- reorder_df, 24
- round_df, 25

- sender_mat, 26

- WITSclean, 26