ApplyWindow

Retrieve an intensitynet object focused on a given area

Get the intensitynet object delimited by the given window
**AreEventsRelated**

**Usage**

```r
ApplyWindow(obj, x_coords, y_coords)
```

```r
## S3 method for class 'intensitynet'
ApplyWindow(obj, x_coords, y_coords)
```

**Arguments**

- `obj`: intensitynet object
- `x_coords`: vector containing the x coordinate limits of the window
- `y_coords`: vector containing the y coordinate limits of the window

**Value**

intensitynet object delimited by the window (sub-part of the original)

**Examples**

```r
data("und_intnet_chicago")
sub_intnet_chicago <- ApplyWindow(und_intnet_chicago,
                                  x_coords = c(300, 900),
                                  y_coords = c(500, 1000))
```

---

**AreEventsRelated** Checks if events are related to the intensitynet object

**Description**

Checks if events are related to the intensitynet object

**Usage**

```r
AreEventsRelated(obj)
```

```r
## S3 method for class 'intensitynet'
AreEventsRelated(obj)
```

**Arguments**

- `obj`: Intensitynet object

**Value**

TRUE if related, FALSE otherwise
Examples

```r
data("und_intnet_chicago")
AreEventsRelated(und_intnet_chicago)
```

---

CalculateDistancesMtx.netTools

*Calculates the distances between all pairs of nodes from the given network*

---

Description

Calculates the distances between all pairs of nodes from the given network

Usage

```r
## S3 method for class 'netTools'
CalculateDistancesMtx(obj)
```

Arguments

- `obj` netTools object -> list(): with the node coordinates 'x' and 'y'

Value

distances matrix

---

**dir_intnet_chicago**

*This data is an intensitynet object containing a directed network. The base data used is from Chicago, extracted from the spatstat package.*

---

Description

This data is an intensitynet object containing a directed network. The base data used is from Chicago, extracted from the spatstat package.

Usage

`dir_intnet_chicago`

Format

An object of class intensitynetDir (inherits from intensitynet) of length 6.

Source

Calculate all the edge intensities of the graph. It’s more fast than using iteratively the function EdgeIntensity for all edges.

### S3 method for class 'intensitynet'

```r
EdgeIntensitiesAndProportions(obj)
```

**Arguments**

- `obj` intensitynet object

**Value**

intensitynet class object where the graph contains all the edge intensities as an attribute

---

**Description**

Given two nodes, calculates its edge intensity

If not calculated, calculates the intensity of the edge with nodes; `node_id1`, `node_id2`. If the edge already contains an intensity, the function gives it directly without re-calculation.

### S3 method for class 'intensitynet'

```r
EdgeIntensity(obj, node_id1, node_id2)
```

**Arguments**

- `obj` intensitynet object
- `node_id1` First node ID of the edge
- `node_id2` Second node ID of the edge

**Value**

Intensity of the edge
GeoreferencedGgplot2.netTools

Plot heatmaps of a network

Description

This function uses internally the package 'ggplot2' to plot heatmaps of a network

Usage

## S3 method for class 'netTools'
GeoreferencedGgplot2(obj, ...)

Arguments

obj netTools object -> list( intnet: intensitynet object, data_df: dataframe( xcoord: x coordinates of the nodes, ycoord: y coordinates of the nodes, value: vector values to plot ), net_vertices: chosen vertices to plot the heatmap (or its related edges in case to plot the edge heatmap), net_edges chosen edges to plot the heatmap, can be either the edge id’s or its node endpoints (e.j. c(1,2, 2,3, 7,8)), heat_type: data which the heatmap will refer, mode: ('moran', 'getis', 'v_intensity', 'e_intensity' or mark), show_events: boolean to show or not the events as orange squares, alpha optional argument to set the transparency of the events (show_events = TRUE). The range is from 0.1 (transparent) to 1 (opaque). Default: alpha = 1 )

... extra arguments for the ggplot

GeoreferencedPlot.netTools

Plot the given network using its node coordinates

Description

Plot the given network using its node coordinates

Usage

## S3 method for class 'netTools'
GeoreferencedPlot(obj, ...)

GetEventCorrection

**Arguments**

obj netTools object -> list(intnet: intensitynet object, vertex_labels: list of labels for the vertices, edge_labels: list of labels for the edges, xy_axes: boolean to show or not the x and y axes, enable_grid: boolean to draw or not a background grid, show_events: boolean to show or not the events as orange squares, show_events option to show the events as orange squares, FALSE by default, alpha optional argument to set the transparency of the events (show_events = TRUE). The range is from 0.1 (transparent) to 1 (opaque). Default: alpha = 1, path: vector with the nodes of the path to be highlighted. Default NULL)

... extra arguments for the plot

---

GetEventCorrection Gives the event correction value related to the intensitynet object

**Description**

Gives the event correction value related to the intensitynet object

**Usage**

GetEventCorrection(obj)

## S3 method for class 'intensitynet'
GetEventCorrection(obj)

**Arguments**

obj intensitynet object

**Value**

integer, event correction value

**Examples**

data("und_intnet_chicago")
GetEventCorrection(und_intnet_chicago)
**GetEvents**

*Gives the events related to the intensitynet object*

**Description**

Returns a matrix containing the events information, i.e. coordinates and categories

**Usage**

```
GetEvents(obj)
```

```r
## S3 method for class 'intensitynet'
GetEvents(obj)
```

**Arguments**

- `obj` intensitynet object

**Value**

matrix containing the event information

**Examples**

```r
data("und_intnet_chicago")
GetEvents(und_intnet_chicago)
```

---

**GetGraph**

*Gives the graph related to the intensitynet object*

**Description**

Returns the `igraph` class network related to the intensitynet object

**Usage**

```
GetGraph(obj)
```

```r
## S3 method for class 'intensitynet'
GetGraph(obj)
```

**Arguments**

- `obj` intensitynet object
GetGraphType

Value

igraph class object

Examples

data("und_intnet_chicago")
GetGraph(und_intnet_chicago)

GetGraphType

Gives the type of graph related to the intensitynet object

Description

Gives the type of graph related to the intensitynet object

Usage

GetGraphType(obj)

## S3 method for class 'intensitynet'
GetGraphType(obj)

Arguments

obj intensitynet object

Value

graph type in characters

Examples

data("und_intnet_chicago")
GetGraphType(und_intnet_chicago)
InitGraph.netTools  

Creates an igraph network with the given data

Description

Creates an igraph network with the given data
Set igraph network node coordinates as its attributes

Usage

```r
## S3 method for class 'netTools'
InitGraph(obj)
```

```r
## S3 method for class 'netTools'
SetNetCoords(obj)
```

Arguments

- `obj` netTools object -> list(graph: igraph, list(): with the node coordinates 'x' and 'y')

Value

igraph network
igraph network with the given coordinates as the attributes of the nodes

intensitynet  Constructor of the class intensitynet.

Description

This constructor creates an intensitynet object using an adjacency matrix, the coordinates of the nodes and the coordinates of the events.

Usage

```r
intensitynet(
  adjacency_mtx,
  node_coords,
  event_data,
  graph_type = c("undirected", "directed", "mixed"),
  event_correction = 5
)
```
IsIntensitynet

Arguments

- **adjacency_mtx**: Network adjacency matrix
- **node_coords**: Nodes latitude and longitude matrix (coordinates)
- **event_data**: DataFrame with event latitude and longitude coordinates (mandatory columns) and optional attributes related to the events
- **graph_type**: Network type: 'undirected' (default), 'directed' or 'mixed'
- **event_correction**: Value that determines how far can be an event to be considered part of a segment (default 5). This value highly depends on the given coordinate system

Value

intensitynet class object containing: graph = <igraph>, events = <matrix>, graph_type = c('directed', 'undirected', 'mixed'), distances = <matrix>, event_correction = <integer>, events_related = <boolean>

Examples

```r
library(spatstat)
data(chicago)
chicago_df <- as.data.frame(chicago[['data']]) # Get as dataframe the data from Chicago

# Get the adjacency matrix. One way is to create an igraph object from the edge coordinates.
edges <- cbind(chicago[['domain']][['from']], chicago[['domain']][['to']])
chicago_net <- igraph::graph_from_edgelist(edges)

# And then use the igraph function 'as_adjacency_matrix'
chicago_adj_mtx <- as.matrix(igraph::as_adjacency_matrix(chicago_net))
chicago_node_coords <- data.frame(xcoord = chicago[['domain']][['vertices']][['x']],
ycoord = chicago[['domain']][['vertices']][['y']])

# Create the intensitynet object, in this case will be undirected
intnet_chicago <- intensitynet(chicago_adj_mtx,
                        node_coords = chicago_node_coords,
                        event_data = chicago_df)
```

IsIntensitynet

Is this class object intensitynet?

Description

Determine if the given object is from the class intensitynet

Usage

IsIntensitynet(obj)
Arguments

obj The object which will be checked if it belongs to the intensitynet class

Value

boolean, 'TRUE' if the argument obj is an intensitynet object

Examples

data("und_intnet_chicago")
IsIntensitynet(und_intnet_chicago)

mix_intnet_chicago

This data is an intensitynet object containing a mixed network. The base data used is from Chicago, extracted from the spatstat package.

Description

This data is an intensitynet object containing a mixed network. The base data used is from Chicago, extracted from the spatstat package.

Usage

mix_intnet_chicago

Format

An object of class intensitynetMix (inherits from intensitynet) of length 6.

Source

https://rdrr.io/cran/spatstat.data/man/chicago.html
NodeGeneralCorrelation

Calculate dependence statistics on the network

Description

It allows to compute different dependence statistics on the network for the given vector and for neighborhoods of distinct order. Such statistics are; correlation, covariance, Moran’s I and Geary’s C.

Usage

NodeGeneralCorrelation(
  obj,
  dep_type,
  lag_max,
  intensity,
  partial_neighborhood = TRUE
)

## S3 method for class 'intensitynet'
NodeGeneralCorrelation(
  obj,
  dep_type = c("correlation", "covariance", "moran", "geary"),
  lag_max,
  intensity,
  partial_neighborhood = TRUE
)

Arguments

- **obj**: intensitynet object
- **dep_type**: 'correlation', 'covariance', 'moran', 'geary'. The type of dependence statistic to be computed.
- **lag_max**: Maximum geodesic lag at which to compute dependence
- **intensity**: Vector containing the values to calculate the specified dependency in the network. Usually the node mean intensities.
- **partial_neighborhood**: use partial neighborhood (TRUE) or cumulative (FALSE). TRUE by default

Value

A vector containing the dependence statistics (ascending from order 0).
Examples

data("und_intnet_chicago")
g <- und_intnet_chicago$graph

gen_corr <- NodeGeneralCorrelation(und_intnet_chicago, dep_type = 'correlation', lag_max = 2,
                                      intensity = igraph::vertex_attr(g)$intensity)

nodeIntensity.intensitynetDir

Calculates the mean intensity of the given node (for directed networks)

Description

Given a node, calculates its mean intensities regarding in and out edges associated with the node.

Usage

## S3 method for class 'intensitynetDir'
MeanNodeIntensity(obj, node_id)

Arguments

obj intensitynetDir object
node_id ID of the node

Value

mean intensities of the given node for in and out edges

nodeIntensity.intensitynetMix

Calculates the mean intensity of the given node (for mixed networks)

Description

Given a node, calculates its mean intensities depending on the edges associated with the node, those intensities are: in, out (for directed edges), undirected and total intensity.

Usage

## S3 method for class 'intensitynetMix'
MeanNodeIntensity(obj, node_id)
nodeIntensity.intensitynetUnd

**Arguments**

- `obj`: intensitynetMix object
- `node_id`: ID of the node

**Value**

mean intensities of the given node for undirected edges, in and out directed and total intensity.

---

nodeIntensity.intensitynetUnd

*Calculates the mean intensity of the given node (for undirected networks)*

---

**Description**

Calculates the mean intensity of the given node (intensity of all the edges of the node/number of edges of the node)

**Usage**

```r
## S3 method for class 'intensitynetUnd'
MeanNodeIntensity(obj, node_id)
```

**Arguments**

- `obj`: intensitynetUnd object
- `node_id`: ID of the node

**Value**

mean intensity of the given node

---

NodeLocalCorrelation

*Calculates local correlations based on nodes*

---

**Description**

Gives the node local Moran-I, Getis-Gstar or Geary-c correlations

**Usage**

```r
NodeLocalCorrelation(obj, dep_type = "moran", intensity)
```

```r
## S3 method for class 'intensitynet'
NodeLocalCorrelation(obj, dep_type = c("moran", "getis", "geary"), intensity)
```
PathTotalWeight

Arguments

- **obj**: intensitynet object
- **dep_type**: 'moran', 'getis' or 'geary'. Type of local correlation to be computed (Moran-i, Getis-Gstar, Geary-c), default = 'moran'.
- **intensity**: vector containing the values to calculate the specified correlation for each node in the network.

Value

a vector containing two values. The first value is a vector with the specified local correlations for each node. The second values is the given intensitynet class object but with the correlations added to the node attributes of its network.

Source


Examples

```r
## Not run:
data("und_intnet_chicago")
g <- und_intnet_chicago$graph
data_moran <- NodeLocalCorrelation(und_intnet_chicago, dep_type = 'moran', intensity = igraph::vertex_attr(g)$intensity)
moran_i <- data_moran$correlation
intnet <- data_moran$intnet
## End(Not run)
```

---

PathTotalWeight

**Calculates the total weight of the given path**

Description

Calculates the total weight of the given path

Usage

PathTotalWeight(obj, path_nodes, weight = NA)

## S3 method for class 'intensitynet'
PathTotalWeight(obj, path_nodes, weight = NA)
plot.intensitynet

Arguments

- **obj**: intensitynet object
- **path_nodes**: vector containing the node ID’s of the path
- **weight**: an string specifying the type of weight to be computed. If no weight type is provided, the function will calculate the total amount of edges. Default NA.

Value

total weight of the path

Examples

data("und_intnet_chicago")
PathTotalWeight(und_intnet_chicago, c('V115', 'V123', 'V125', 'V134'), weight = 'intensity')

---

plot.intensitynet  
*Plot intensitynet object*

Description

Plot intensitynet object

Usage

```r
## S3 method for class 'intensitynet'
plot(
  x,
  vertex_labels = "none",
  edge_labels = "none",
  xy_axes = TRUE,
  enable_grid = FALSE,
  show_events = FALSE,
  alpha = 1,
  path = NULL,
  ...
)
```

Arguments

- **x**: intensitynet object
- **vertex_labels**: list -> labels for the vertices
- **edge_labels**: list -> labels for the edges
- **xy_axes**: show the x and y axes
enable_grid  draw a background grid
show_events  option to show the events as orange squares, FALSE by default
alpha        optional argument to set the transparency of the events (show_events = TRUE).
The range is from 0.1 (transparent) to 1 (opaque). Default: alpha = 1
path         vector with the nodes of the path to be highlighted. Default NULL
...          extra arguments for the plot

Value
No return value, same as graphics::plot.

Examples

data("und_intnet_chicago")
plot(und_intnet_chicago)  # basic plot
plot(und_intnet_chicago, enable_grid = TRUE)  # with grid
plot(und_intnet_chicago, xy_axes = FALSE)  # without axes
plot(und_intnet_chicago, path = c("V1", "V2", "V24", "V25", "V26", "V48"))  # highlight a path

PlotHeatmap

Given an intensitynet object, plot network heatmaps

Description
Plot the network correlations or intensities.

Usage
PlotHeatmap(
  obj,
  heat_type = "none",
  intensity_type = "none",
  net_vertices = NULL,
  net_edges = NULL,
  show_events = FALSE,
  alpha = 1,
  ...
)

## S3 method for class 'intensitynet'
PlotHeatmap(
  obj,
  heat_type = c("none", "moran", "geary", "v_intensity", "e_intensity"),
  intensity_type = c("none"),
  net_vertices = NULL,
PlotHeatmap

net_edges = NULL,
show_events = FALSE,
alpha = 1,
...
)

Arguments

obj
intensitynet object

heat_type
a string with the desired heatmap to be plotted, the options are: 'moran': Local Moran-i correlation (with 999 permutations), 'geary': Local Geary-c correlation. The correlations will use the indicated intensity type, 'v_intensity': vertex mean intensity, 'e_intensity': edge intensity, mark name: name of the mark (string) to plot its edge proportion, 'none': plain map.

intensity_type
name of the vertex intensity used to plot the heatmap for moran, geary and v_intensity options (of the heat_type argument). The options are: For undirected networks: 'intensity'. For directed networks: 'intensity_in' or 'intensity_out'. For mixed networks: 'intensity_in', 'intensity_out', 'intensity_und' or 'intensity_all'. If the intensity parameter is 'none', the function will use, if exist, the intensity (undirected) or intensity_in (directed) values from the network nodes. If the heat_type is 'e_intensity', this parameter will be skipped and plot the edge intensities instead.

net_vertices
chosen vertices to plot the heatmap (or its related edges in case to plot the edge heatmap)

net_edges
chosen edges to plot the heatmap, can be either the edge id’s or its node endpoints (e.j. c(1,2, 2,3, 7,8))

show_events
option to show the events as orange squares, FALSE by default

alpha
optional argument to set the transparency of the events (show_events = TRUE). The range is from 0.1 (transparent) to 1 (opaque). Default: alpha = 1

extra arguments for the class ggplot

Value

The plot of the heatmap with class c(“gg”, "ggplot")

Examples

## Not run:
data(“und_intnet_chicago”)PlotHeatmap(und_intnet_chicago, heat_type='moran')

## End(Not run)
### PlotNeighborhood

*Plot the neighbors of a node including the closer events*

#### Description

Plot the net and the events in the neighborhood area of the given node.

#### Usage

```r
PlotNeighborhood(obj, node_id, ...)
```

#### Arguments

- `obj`: intensitynet object
- `node_id`: Id of the node which the plot will be focused
- `...`: Extra arguments for plotting

#### Value

No return value, just plots the neighborhood and the events.

#### Examples

```r
data("und_intnet_chicago")
PlotNeighborhood(und_intnet_chicago, node_id = "V300")
```

### PointToLine

*Gives the distance between an event and the line formed by two nodes.

#### Description

Gives the distance between an event and the line (not segment) formed by two nodes.

#### Usage

```r
## S3 method for class 'netTools'
PointToLine(obj)
```

#### Arguments

- `obj`: netTools object -> list(p1:c(coordx, coordy), p2:c(coordx, coordy), e:c(coordx, coordy))
Value
the distance to the line

---

**Description**
Gives the shortest distance between an event and a set of segments.

**Usage**
PointToSegment(obj)

**Arguments**
- **obj**: netTools object -> list(p1:matrix(coordx, coordy), p2:matrix(coordx, coordy), e:matrix(coordx, coordy))

**Value**
distance vector to each segment

---

**PointToSegment_deprecated.netTools**
*Gives the shortest distance between an event and the segment formed by two nodes.*

**Description**
Gives the shortest distance between an event and the segment formed by two nodes.

**Usage**
PointToSegment_deprecated(obj)

**Arguments**
- **obj**: netTools object -> list(p1:c(coordx, coordy), p2:c(coordx, coordy), e:c(coordx, coordy))

**Value**
distance to the segment
RelateEventsToNetwork \( \text{intensitynetDir} \)

**Description**

Calculates edgewise and mean nodewise intensities for the given intensitynet object and, for each edge, the proportions of all event covariates.

**Usage**

\[
\text{RelateEventsToNetwork}(\text{obj})
\]

**Arguments**

- **obj** intensitynet object

**Value**

proper intensitynet object (Undirected, Directed, or Mixed) with a graph containing the nodewise intensity in the node attributes and the edgewise intensities and event covariate proportions as edge attributes.

**Examples**

```r
data("und_intnet_chicago")
intnet_chicago <- RelateEventsToNetwork(und_intnet_chicago)
```

---

RelateEventsToNetwork \( \text{intensitynetDir} \)

**Description**

Calculates edgewise and mean nodewise intensities for Directed networks and, for each edge, the proportions of all event covariates.

**Usage**

```r
## S3 method for class 'intensitynetDir'
RelateEventsToNetwork(\text{obj})
```

**Arguments**

- **obj** intensitynetDir object
Value

proper intensitynetDir object with a graph containing the nodewise intensity in the node attributes and the edgewise intensities and event covariate proportions as edge attributes.

Description

Calculates edgewise and mean nodewise intensities for Mixed networks and, for each edge, the proportions of all event covariates.

Usage

## S3 method for class 'intensitynetMix'
RelateEventsToNetwork(obj)

Arguments

obj intensitynetMix object

Value

proper intensitynetMix object with a graph containing the nodewise intensity in the node attributes and the edgewise intensities and event covariate proportions as edge attributes.

Description

Calculates edgewise and mean nodewise intensities for Undirected networks and, for each edge, the proportions of all event covariates.

Usage

## S3 method for class 'intensitynetUnd'
RelateEventsToNetwork(obj)

Arguments

obj intensitynetUnd object
**Value**

proper intensitynetUnd object with a graph containing the nodewise intensity in the node attributes and the edgewise intensities and event covariate proportions as edge attributes.

---

**SetEdgeIntensity.netTools**

Sets the given intensities as an edge attribute to the given igraph network.

---

**Description**

Sets the given intensities as an edge attribute to the given igraph network.

**Usage**

```r
## S3 method for class 'netTools'
SetEdgeIntensity(obj)
```

**Arguments**

- `obj`: netTools object -> list(graph: igraph, node_id1: node id, node_id2: node id, intensity: edge intensity)

**Value**

igraph network with the given intensities as attributes of the edges.

---

**SetNetworkAttribute.intensitynet**

Set attributes to the network edges or nodes.

**Description**

Set attributes to the network edges or nodes.

**Usage**

```r
## S3 method for class 'intensitynet'
SetNetworkAttribute(obj, where, name, value)
```

**Arguments**

- `obj`: intensitynet object
- `where`: 'vertex' or 'edge'. where to set the attribute
- `name`: name of the attribute
- `value`: vector containing the data for the attribute
Value

intensitynet object containing the network with the added attributes

SetNodeIntensity.netTools

Sets the given intensities as a node attribute to the given igraph network

Description

Sets the given intensities as a node attribute to the given igraph network

Usage

## S3 method for class 'netTools'
SetNodeIntensity(obj)

Arguments

obj netTools object -> list(graph: igraph, node_id: node id, intensity: node intensity)

Value

igraph network with the given intensities as attributes of the nodes

ShortestNodeDistance.intensitynet

Given two nodes, gives its shortest distance based on the minimum amount of edges

Description

Calculates the shortest distance path between two nodes (based on the minimum amount of edges). The function also returns the total weight of the path, if the weight is not available, returns the number of edges.

Usage

## S3 method for class 'intensitynet'
ShortestNodeDistance(obj, node_id1, node_id2)

Arguments

obj intensitynet object
node_id1 id of the starting node
node_id2 id of the end node
ShortestPath

Value
distance of the path and the nodes of the path

---

ShortestPath

Given two nodes, calculates the shortest path and its total weight

Description

Calculates the shortest path between two vertices (based on the minimum amount of edges) and calculates its total weight

Usage

ShortestPath(obj, node_id1, node_id2, weight = NA, mode = "all")

## S3 method for class 'intensitynet'
ShortestPath(obj, node_id1, node_id2, weight = NA, mode = "all")

Arguments

- **obj**: intensitynet object
- **node_id1**: starting node
- **node_id2**: ending node
- **weight**: an string, calculate the shortest path based on this type of weight. If no weight type is provided, the function will calculate the shortest path based on the minimum amount of edges. Default NA.
- **mode**: Character 'in', 'out', 'all' (default). Gives whether the shortest paths to or from the given vertices should be calculated for directed graphs. If out then the shortest paths from the vertex, if in then to it will be considered. If all, the default, then the corresponding undirected graph will be used, ie. not directed paths are searched. This argument is ignored for undirected graphs.

Value
total weight of the shortest path and the path vertices with class igraph.vs

Examples

data("und_intnet_chicago")
ShortestPath(und_intnet_chicago, node_id1 = 'V1', node_id2 = 'V300', weight = 'intensity')
Summary of the intensitynet object

Description
Give information about the intensitynet object specific class (intensitynetUnd, intensitynetDir, or intensitynetMix), the network number of nodes, edges and events, the event correction value and, if the events had been related to the intensitynet object network.

Usage
## S3 method for class 'intensitynet'
summary(object, ...)

Arguments
object Intensitynet object
...
Extra parameters for the summary function

Value
list containing the displayed information

Examples

data("und_intnet_chicago")
summary(und_intnet_chicago)

Undirected2RandomDirectedAdjMtx.netTools
Converts a directed adjacency matrix to undirected

Description
Creates a directed adjacency matrix from an Undirected one with random directions (in-out edges) but with the same connections between nodes.

Usage
## S3 method for class 'netTools'
Undirected2RandomDirectedAdjMtx(obj)

Arguments
obj netTools object -> list(mtx: matrix)
und_intnet_chicago

**Value**

directed adjacency matrix with random directions

**Description**

This data is an intensitynet object containing an undirected network. The base data used is from Chicago, extracted from the spatstat package.

**Usage**

und_intnet_chicago

**Format**

An object of class `intensitynetUnd` (inherits from `intensitynet`) of length 6.

**Source**

Index

* datasets
  * dir_intnet_chicago, 4
  * mix_intnet_chicago, 12
  * und_intnet_chicago, 28
ApplyWindow, 2
AreEventsRelated, 3
CalculateDistancesMtx.netTools, 4
dir_intnet_chicago, 4
EdgeIntensitiesAndProportions.intensitynet, 5
EdgeIntensity.intensitynet, 5
GeoreferencedGgplot2.netTools, 6
GeoreferencedPlot.netTools, 6
GetEventCorrection, 7
GetEvents, 8
GetGraph, 8
GetGraphType, 9
InitGraph.netTools, 10
intensitynet, 10
IsIntensitynet, 11
MeanNodeIntensity.intensitynetDir
  * (nodeIntensity.intensitynetDir), 14
MeanNodeIntensity.intensitynetMix
  * (nodeIntensity.intensitynetMix), 14
MeanNodeIntensity.intensitynetUnd
  * (nodeIntensity.intensitynetUnd), 15
mix_intnet_chicago, 12
NodeGeneralCorrelation, 13
nodeIntensity.intensitynetDir, 14
nodeIntensity.intensitynetMix, 14
nodeIntensity.intensitynetUnd, 15
NodeLocalCorrelation, 15
PathTotalWeight, 16
plot.intensitynet, 17
PlotHeatmap, 18
PlotNeighborhood, 20
PointToLine.netTools, 20
PointToSegment
  * (PointToSegment.netTools), 21
PointToSegment.netTools, 21
PointToSegment_deprecated
  * (PointToSegment_deprecated.netTools), 21
PointToSegment_deprecated.netTools, 21
RelateEventsToNetwork, 22
RelateEventsToNetwork.intensitynetDir, 22
RelateEventsToNetwork.intensitynetMix, 23
RelateEventsToNetwork.intensitynetUnd, 23
SetEdgeIntensity.netTools, 24
SetNetCoords.netTools
  * (InitGraph.netTools), 10
SetNetworkAttribute.intensitynet, 24
SetNodeIntensity.netTools, 25
ShortestNodeDistance.intensitynet, 25
ShortestPath, 26
summary, 27
und_intnet_chicago, 28
Undirected2RandomDirectedAdjMtx.netTools, 27