

# Package ‘intensitynet’

May 8, 2026

**Version** 1.4.0

**Date** 2023-04-09

**Title** Intensity Analysis of Spatial Point Patterns on Complex Networks

**Maintainer** Pol Llagostera <pol.llagostera@udl.cat>

**Depends** R (>= 3.6.0)

**Suggests** spatstat (>= 2.3.0), testthat (>= 3.0.0)

**Imports** ggplot2 (>= 3.3.2), igraph (>= 1.2.5), intergraph (>= 2.0.2),  
Matrix (>= 1.5.1), methods (>= 3.6.3), sna (>= 2.6),  
spatstat.geom (>= 2.3.1), spdep (>= 1.2.1), viridis (>= 0.5.1)

**Description** Tools to analyze point patterns in space occurring over planar network structures derived from graph-related intensity measures for undirected, directed, and mixed networks. This package is based on the following research: Eckardt and Mateu (2018) <[doi:10.1080/10618600.2017.1391695](https://doi.org/10.1080/10618600.2017.1391695)>. Eckardt and Mateu (2021) <[doi:10.1007/s11749-020-00720-4](https://doi.org/10.1007/s11749-020-00720-4)>.

**License** GPL-3

**Encoding** UTF-8

**RoxygenNote** 7.2.3

**LazyData** true

**Config/testthat/edition** 3

**NeedsCompilation** no

**Author** Pol Llagostera [aut, cre],  
Matthias Eckardt [aut]

**Repository** CRAN

**Date/Publication** 2023-04-11 06:10:03 UTC

## Contents

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
mix_intnet_chicago . . . . .	12
NodeGeneralCorrelation . . . . .	12
nodeIntensity.intensitynetDir . . . . .	14
nodeIntensity.intensitynetMix . . . . .	14
nodeIntensity.intensitynetUnd . . . . .	15
NodeLocalCorrelation . . . . .	15
PathTotalWeight . . . . .	16
plot.intensitynet . . . . .	17
PlotHeatmap . . . . .	18
PlotNeighborhood . . . . .	19
PointToLine.netTools . . . . .	20
PointToSegment.netTools . . . . .	21
PointToSegment_deprecated.netTools . . . . .	21
RelateEventsToNetwork . . . . .	22
RelateEventsToNetwork.intensitynetDir . . . . .	22
RelateEventsToNetwork.intensitynetMix . . . . .	23
RelateEventsToNetwork.intensitynetUnd . . . . .	23
SetEdgeIntensity.netTools . . . . .	24
SetNetworkAttribute.intensitynet . . . . .	24
SetNodeIntensity.netTools . . . . .	25
ShortestNodeDistance.intensitynet . . . . .	25
ShortestPath . . . . .	26
summary . . . . .	27
Undirected2RandomDirectedAdjMtx.netTools . . . . .	27
und_intnet_chicago . . . . .	28

**Index** **29**

---

ApplyWindow	<i>Retrieve an intensitynet object focused on a given area</i>
-------------	--

---

**Description**

Get the intensitynet object delimited by the given window

**Usage**

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

```
## 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**

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

---

AreEventsRelated	<i>Checks if events are related to the intensitynet object</i>
------------------	--

---

**Description**

Checks if events are related to the intensitynet object

**Usage**

```
AreEventsRelated(obj)
```

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

**Arguments**

obj	Intensitynet object
-----	---------------------

**Value**

TRUE if related, FALSE otherwise

**Examples**

```
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**

```
## 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**

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

---

EdgeIntensitiesAndProportions.intensitynet  
*Calculate all the edge intensities of the graph.*

---

**Description**

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

**Usage**

```
## S3 method for class 'intensitynet'  
EdgeIntensitiesAndProportions(obj)
```

**Arguments**

obj                    intensitynet object

**Value**

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

---

EdgeIntensity.intensitynet  
*Given two nodes, calculates its edge intensity*

---

**Description**

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.

**Usage**

```
## S3 method for class 'intensitynet'  
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, ...)
```

**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: 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	<i>Gives the events related to the intensitynet object</i>
-----------	--

---

**Description**

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

**Usage**

```
GetEvents(obj)

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

**Arguments**

obj            intensitynet object

**Value**

matrix containing the event information

**Examples**

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

---

GetGraph	<i>Gives the graph related to the intensitynet object</i>
----------	---

---

**Description**

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

**Usage**

```
GetGraph(obj)

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

**Arguments**

obj            intensitynet object

**Value**

igraph class object

**Examples**

```
data("und_intnet_chicago")  
GetGraph(und_intnet_chicago)
```

---

GetGraphType	<i>Gives the type of graph related to the intensitynet object</i>
--------------	---

---

**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

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

## 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

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

**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**

```
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 a intensitynet object

**Examples**

```
data("und_intnet_chicago")
IsIntensitynet(und_intnet_chicago)
```

---

mix_intnet_chicago	<i>This data is an intensitynet object containing an mixed network. The base data used is from Chicago, extracted from the spatstat package.</i>
--------------------	--

---

**Description**

This data is an intensitynet object containing an 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**

<code>obj</code>	intensitynet object
<code>dep_type</code>	'correlation', 'covariance', 'moran', 'geary'. The type of dependence statistic to be computed.
<code>lag_max</code>	Maximum geodesic lag at which to compute dependence
<code>intensity</code>	Vector containing the values to calculate the specified dependency in the network. Usually the node mean intensities.
<code>partial_neighborhood</code>	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**

<code>obj</code>	intensitynetDir object
<code>node_id</code>	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)
```

**Arguments**

<code>obj</code>	intensitynetMix object
<code>node_id</code>	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**

```
## 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**

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

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

**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**

"A Local Indicator of Multivariate Spatial Association: Extending Geary's c, Geographical Analysis" Luc Anselin (2018) <doi:10.1111/gean.12164>

**Examples**

```
## 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	<i>Calculates the total weight of the given path</i>
-----------------	--

---

**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)
```

**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**

```
## 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,
  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.

<code>intensity_type</code>	name of the vertex intensity used to plot the heatmap for moran, geary and <code>v_intensity</code> options (of the <code>heat_type</code> argument). The options are; For undirected networks: <code>'intensity'</code> . For directed networks: <code>'intensity_in'</code> or <code>'intensity_out'</code> . For mixed networks: <code>'intensity_in'</code> , <code>'intensity_out'</code> , <code>'intensity_und'</code> or <code>'intensity_all'</code> . If the intensity parameter is <code>'none'</code> , the function will use, if exist, the intensity (undirected) or <code>intensity_in</code> (directed) values from the network nodes. If the <code>heat_type</code> is <code>'e_intensity'</code> , this parameter will be skipped and plot the edge intensities instead.
<code>net_vertices</code>	chosen vertices to plot the heatmap (or its related edges in case to plot the edge heatmap)
<code>net_edges</code>	chosen edges to plot the heatmap, can be either the edge id's or its node endpoints (e.j. <code>c(1,2, 2,3, 7,8)</code> )
<code>show_events</code>	option to show the events as orange squares, FALSE by default
<code>alpha</code>	optional argument to set the transparency of the events ( <code>show_events = TRUE</code> ). The range is from 0.1 (transparent) to 1 (opaque). Default: <code>alpha = 1</code>
<code>...</code>	extra arguments for the class <code>ggplot</code>

**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**

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

## S3 method for class 'intensitynet'
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**

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

---

PointToLine.netTools *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**

```
## 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



---

RelateEventsToNetwork *Calculates intensity statistics for the given intensitynet object*

---

### Description

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

### Usage

```
RelateEventsToNetwork(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

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

---

RelateEventsToNetwork.intensitynetDir  
*Calculates intensity statistics for the given intensitynet object*

---

### Description

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

### Usage

```
## S3 method for class 'intensitynetDir'
RelateEventsToNetwork(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.

---

RelateEventsToNetwork.intensitynetMix

*Calculates intensity statistics for the given intensitynet object*

---

**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.

---

RelateEventsToNetwork.intensitynetUnd

*Calculates intensity statistics for the given intensitynet object*

---

**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**

```
## 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**

```
## 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

**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

*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)
-----	--------------------------------------

**Value**

directed adjacency matrix with random directions

---

und_intnet_chicago	<i>This data is an intensitynet object containing an undirected network. The base data used is from Chicago, extracted from the spatstat package.</i>
--------------------	---

---

**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**

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

# 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, 12
- nodeIntensity.intensitynetDir, 14
- nodeIntensity.intensitynetMix, 14
- nodeIntensity.intensitynetUnd, 15
- NodeLocalCorrelation, 15
- PathTotalWeight, 16
- plot.intensitynet, 17
- PlotHeatmap, 18
- PlotNeighborhood, 19
- 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