Package 'bipartiteD3'

May 7, 2026

Type Package

Title Interactive Bipartite Graphs

Version 0.3.2

Description Generates interactive bipartite graphs using the D3 library.
Designed for use with the 'bipartite' analysis package.
Includes open source 'viz-js' library
Adapted from examples at <https://bl.ocks.org/NPashaP> (released under GPL-3).

License GPL-3

Encoding UTF-8

Imports RColorBrewer (>= 1.1), r2d3 (>= 0.2.2), purrr (>= 0.2.5), dplyr (>= 0.7.5), tidyr (>= 0.8), stringr(>= 1.3), tibble (>= 1.4)

RoxygenNote 7.3.1

Suggests knitr, bipartite, vegan, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Author Chris Terry [aut, cre]

Maintainer Chris Terry <christerry3@btinternet.com>

Repository CRAN

Date/Publication 2024-09-02 22:10:02 UTC

Contents

`Array2DF` *Convert bipartite-style arrays to dataframe*

Description

Returns a data frame in the format internally required for bipartiteD3 where the first two columns list the interacting species, and subsequent columns list the link strengths in each site.

Usage

```
Array2DF(
  Array,
  PrimaryLab = "Primary",
  SecondaryLab = "Secondary",
  SiteNames = NULL
)
```

Arguments

`Array`	An array of bipartite format
`PrimaryLab`	Label for the primary level of the bipartite web, e.g. 'Plants'
`SecondaryLab`	Label for the secondary level of the bipartite web, e.g. 'Pollinators'
`SiteNames`	Vector of names for the different sires (array slices). By default takes names of input array if they exist.

Details

Array2DF expects an array of multiple bipartite webs as may be created by the webs2array() function in bipartite. This structure includes row and column names to indicate the species, and a named third dimension giving the names of each of the sites

Note an array of this format can be passed directly to bipartite_D3 since it will test for an array and apply Array2DF() anyway.

Value

A data.frame where the first column is the primary interactor, the second the secondary interactor and subsequent named columns detail the link strengths

Examples

```
## Not run:
data(Safariland, vazquenc, package='bipartite')
allin1 <- bipartite::webs2array(Safariland, vazquenc)
Array2DF(allin1)

## End(Not run)
```

bipartite_D3 *Generate interactive bipartite networks*

Description

Plots one or more interactive bipartite graphs. Data can be supplied either in bipartite package format or as a data frame and generates an html widget. There are range of display options, see vignette for examples.

Usage

```
bipartite_D3(
  data,
  filename = "bipartiteD3Script",
  PrimaryLab = "Primary",
  SecondaryLab = "Secondary",
  SiteNames = NULL,
  colouroption = c("monochrome", "brewer", "manual")[1],
  HighlightLab = "Unlinked",
  HighlightCol = "#3366CC",
  monoChromeCol = "rgb(56,43,61)",
  ColourBy = c(1, 2)[2],
  BrewerPalette = "Accent",
  NamedColourVector,
  MainFigSize = NULL,
  SortPrimary = NULL,
  SortSecondary = NULL,
  mp = c(1, 1),
  MinWidth = 10,
  Pad = 1,
  IndivFigSize = c(200, 400),
  BarSize = 35,
  Orientation = c("vertical", "horizontal")[1],
  EdgeMode = c("straight", "smooth")[2],
  BoxLabPos = NULL,
  IncludePerc = TRUE,
  PercentageDecimals = 0,
  PercPos = NULL,
  CSS_Output_Supress = FALSE,
  PRINT = FALSE
)
```

Arguments

data	Food web or webs to be plotted. Can be either in data.frame format (Sp1,Sp2, Site1, Site2 etc) or bipartite style matrix, list of matrices or array.
filename	Character string to name the .js and .css files. Do not include a file extension.

PrimaryLab	Character string to label left (lower) half of graph (e.g. 'plant' or 'host').
SecondaryLab	Character string to label right (upper) half of graph (e.g. 'pollinator' or 'parasitoid').
SiteNames	Character vector giving name or names of site or sites. Used as a title for facets.
colouroption	Either 'monochrome', 'brewer' or 'manual'.
HighlightLab	Name of interactor species to highlight. (Text must match).
HighlightCol	Colour to highlight species. Can be any format read by html, eg simple names: 'PINK', hexcode: '#FFC0CB' or rgb: 'rgb(255,192,203)'
monoChromeCol	If using monochrome option, what colour to use. Can be any format read by html, eg simple names 'PINK', hexcode '#FFC0CB' or rgb 'rgb(255,192,203)'
ColourBy	Which set of interactors to colour by. 1= primary, 2= secondary
BrewerPalette	RColorBrewer palette to use, e.g. 'Set3'. Be sure to select one with enough available colours (it will warn)
NamedColourVector	Named vector of colours for manual colour assignment. Can be any format read by html, eg simple names 'PINK', hexcode '#FFC0CB' or rgb 'rgb(255,192,203)'
MainFigSize	c(width, height). Size of html container for the whole figure
SortPrimary	Vector of order of species to arrange primary level. Default is alphabetical
SortSecondary	Vector of order of species to arrange secondary level. Default is alphabetical
mp	Numeric vector c(rows, columns) for distribution of facets
MinWidth	Numeric. Minimum size to shrink unselected interactors to.
Pad	Numeric. Whitespace gap between species.
IndivFigSize	c(width, height) Size of each facet, specifically the links
BarSize	Thickness of bars representing interactors
Orientation	Either 'horizontal' or 'vertical' orientation. Note that Vertical is currently much better supported!
EdgeMode	Set to 'straight' if you want to avoid default curly lines.
BoxLabPos	c(x_primary,x_secondary) To adjust position of species labels away from graph. Default is based on maximum length of labels.
IncludePerc	Logical - whether or not to show percentage links
PercentageDecimals	Number of decimal places to display percentages to. Useful if rare species are rounded to 0.
PercPos	c(x_p,x_s) To adjust position of percentages away from graph. Default is based on maximum length of labels.
CSS_Output_Supress	Logical - set to TRUE if you have changed the CSS file manually and don't want it overridden
PRINT	Logical - output generated JavaScript to screen?

Details

This function offers a straightforward way to generate an interactive bipartite graph. Hovering over a species will focus just on that species and its interactors, and display their relative fractions. Where multiple networks are examined simultaneously, the selection occurs in tandem.

It will try to download source code of a version of the vis JavaScript library, generate a JavaScript file (.js) and a Cascading Style Sheet (.css) and place them in the working directory. These are then used by r2d3() to create an html object.

When used in RStudio version 1.2+ this is visible in the viewer pane. If using an earlier version of Rstudio, the graph may appear as a blank space in the default RStudio viewer. In this case, you can still use knitr to create an html file and view it in a browser.

Guessing appropriate sizes for the figures can be a process of trial and error. The best values depend on the length of the labels, the number of interactions and their relative weighting. See Vignette for details. If the figure looks weirdly proportioned, the links appear to invert or one of the halves is notably longer than other other, the main figure margins are probably too small. It is often necessary to experiment a little with large or complex figures.

To include figures as a static plot for publication, there are several options. The r2D3 package provides the save_d3_png and save_d3_html functions to directly save d3 objects. This is normally the easiest. If using RStudio v1.2+, then it is possible to just export from the viewer pane.

Otherwise, from an html document generated by knitr, it is often useful to 'print to pdf' within the browser. Finally is possible to extract the svg segment that relates to the figure from the html file and save it directly as an svg file, which can then be used in e.g. Inkscape.

Value

Uses r2d3() to generate an html widget object. Can be viewed either in viewer pane (RStudio V1.2+) or with knitr. See Vignette. As a side effect, saves visjs.js (the vis plotting library), filename.js and filename.css to the working directory.

Examples

```
## Simple Bipartite Style Data Set:
## Not run:  testdata <- data.frame(higher = c("bee1","bee1","bee1","bee2","bee1","bee3"),
lower = c("plant1","plant2","plant1","plant2","plant3","plant4"),
webID = c("meadow","meadow","meadow","meadow","meadow","meadow"), freq=c(5,9,1,2,3,7))
SmallTestWeb <- bipartite::frame2webs(testdata,type.out="array")


bipartite_D3(SmallTestWeb, filename = 'demo1')

## End(Not run)
## For more examples see vignette
```

BP_JS_Writer *Generate JavaScript file for a bipartite network*

Description

Function called by bipartite_D3() to write JavaScript and CSS file. In most cases it is better to use bipartite_D3() directly.

Usage

```
BP_JS_Writer(
  df,
  filename = "JSBP",
  colouroption = c("monochrome", "brewer", "manual")[1],
  HighlightLab = "Unlinked",
  HighlightCol = "#3366CC",
  monoChromeCol = "rgb(56,43,61)",
  ColourBy = c(1, 2)[2],
  BrewerPalette = "Accent",
  NamedColourVector,
  MainFigSize = NULL,
  SortPrimary = NULL,
  SortSecondary = NULL,
  mp = c(1, 1),
  MinWidth = 10,
  Pad = 1,
  IndivFigSize = c(200, 400),
  BarSize = 35,
  Orientation = c("vertical", "horizontal")[1],
  EdgeMode = c("straight", "smooth")[2],
  AxisLabels = NULL,
  FigureLabel = NULL,
  BoxLabPos = NULL,
  IncludePerc = TRUE,
  PercentageDecimals = 0,
  PercPos = NULL,
  CSS_Output_Supress = FALSE,
  PRINT = FALSE
)
```

Arguments

df	data.frame containing the names of the interactors and the link strengths. bipartite package data need to be passed through Matrix2DF or Array2DF first.
filename	character string to name the .js and .css files. Do not include a file extension
colouroption	Either 'monochrome', 'brewer' or 'manual'

HighlightLab	Name of interactor to highlight
HighlightCol	Highlight colour
monoChromeCol	If using monochrome option, what colour to use
ColourBy	Which set of interactors to colour by. 1= primary, 2= secondary
BrewerPalette	RColorBrewer palette
NamedColourVector	Named vector of colours for manual colour assignment
MainFigSize	Size of figure, used here to calculate facet spacing.
SortPrimary	Vector detailing order to arrange primary level. Default is alphabetical
SortSecondary	Vector detailing order to arrange secondary level. Default is alphabetical
mp	Numeric vector c(rows, columns)
MinWidth	Numeric. Minimum size to shrink unselected interactors to.
Pad	Numeric. Gap between species.
IndivFigSize	Size of each facet, specifically the interactions.
BarSize	Thickness of bars representing interactors
Orientation	Either 'horizontal' or 'vertical' orientation.
EdgeMode	Set to 'straight' to avoid curly lines.
AxisLabels	c('Primary','Secondary') to overide column names of dataframe
FigureLabel	Character vector, to allow overide of use of df column names
BoxLabPos	c(x_p,x_s) To adjust position of species labels. Default is based on maximum length of labels.
IncludePerc	Boolean. whether or not to show percentage links
PercentageDecimals	Number of decimal places to display percentages to. Useful if rare species are rounded to 0.
PercPos	c(x_p,x_s) To adjust position of percentages. Default is based on maximum length of labels.
CSS_Output_Supress	Boolean. Set to TRUE if you have changed the CSS file manually and don't want it over written
PRINT	Boolean. Output generated JavaScript to screen?

Value

As a side effect, saves visjs.js (vis plotting library), filename.js and filename.css to the working directory.

Examples

```
## Simple Data Set
testdata <- data.frame(higher = c("bee1","bee1","bee1","bee2","bee1","bee3"),
lower = c("plant1","plant2","plant1","plant2","plant3","plant4"),
 Meadow=c(5,9,1,2,3,7))

BP_JS_Writer(testdata,PRINT=TRUE)

## tidy up (to keep CRAN happy, not needed in real life use)
file.remove('vizjs.js')
file.remove('JSBP.js')
file.remove('JSBP.css')
```

`List2DF` *Convert bipartite-style list of matrices to dataframe*

Description

List2DF returns a data frame in the format internally required for bipartiteD3 where the first two columns list the interacting species, and subsequent columns list the link strengths in each site.

Usage

```
List2DF(
  List,
  PrimaryLab = "Primary",
  SecondaryLab = "Secondary",
  SiteNames = NULL
)
```

Arguments

`List`	An list of bipartite format matrices
`PrimaryLab`	Label for the primary level of the bipartite web, e.g. 'Plants'
`SecondaryLab`	Label for the secondary level of the bipartite web, e.g. 'Pollinators'
`SiteNames`	Vector of names for the different sites (list elements). By default takes names of input matrices if they exist.

Details

List2DF expects an list of multiple bipartite webs as may be created by the frame2webs(type.out='list') function in bipartite. This structure includes row and column names to indicate the species, and a named third dimension giving the names of each of the sites

Note a list of this format can be passed directly to bipartite_D3 since it will test for an list and apply List2DF() anyway.

Value

A data.frame where the first column is the primary interactor, the second the secondary interactor and subsequent named columns detail the link strengths

Examples

```
## Not run:  testdata <- data.frame(higher = c("bee1","bee1","bee1","bee2","bee1","bee3"),
lower = c("plant1","plant2","plant1","plant2","plant3","plant4"),
webID = c("meadow","meadow","meadow","meadow","bog","bog"), freq=c(5,9,1,2,3,7))
bipartite::frame2webs(testdata, type.out = 'list')-> SmallTestWeb

List2DF(SmallTestWeb)

## End(Not run)
```

LoadVisJS *LoadVisJS*

Description

Used internally by BP_JS_Writer() and bipartite_D3() to store the necessary JavaScript library

Usage

```
LoadVisJS()
```

Details

This was originally sourced from the open source vis JavaScript library from vizjs.org (v1.1.0). However, since this is no longer maintained, the library is hardcoded in here.

Matrix2DF *Convert a bipartite-style matrix to dataframe*

Description

Matrix2DF returns a data frame in the format internally required for bipartiteD3 where the first two columns list the interacting species, and the third column lists the link strengths.

Usage

```
Matrix2DF(
  Matrix,
  PrimaryLab = "Primary",
  SecondaryLab = "Secondary",
  SiteLab = "Site"
)
```

Arguments

Matrix	Bipartite network in matrix format
PrimaryLab	Label for the primary level of the bipartite web, e.g. 'Plants'
SecondaryLab	Label for the secondary level of the bipartite web, e.g. 'Pollinators'
SiteLab	Name for the site

Details

Matrix2DF expects a matrix of the format used by bipartite, for example that created by frame2webs(). This structure includes row and column names to indicate the species, and a named third dimension giving the name of that site.

Note a matrix of this format can be passed directly to bipartite_D3() since it will test for a matrix and apply Matrix2DF() anyway.

Value

A data.frame where the first column is the primary interactor, the second the secondary interactor and third column detail the link strengths.

Examples

```
data(Safariland, package='bipartite')
Matrix2DF(Safariland)
```

OrderByCrossover *Find Species Order That Minimises Crossover*

Description

Find an order of species that is likely to minimise cross over. It builds upon the 'cca' method used in the bipartite package, but orders the compartments by size, which tends to give better effects.

Usage

```
OrderByCrossover(df)
```

Arguments

df	A network in data.frame format. (row names for primary layer, column names for secondary layer)

Value

A list containing 'PrimaryOrder' and 'SecondaryOrder', to be used with bipartite_d3()

Examples

```
## Not run:

data(Safariland, package='bipartite')


S_orders <- OrderByCrossover(Safariland)

bipartite_D3(Safariland,
  filename = 'SF_sorted',
  SortPrimary = S_orders[[1]],
  SortSecondary = S_orders[[2]])

## End(Not run)
```

Index