

Package ‘Director’

March 25, 2024

Title A dynamic visualization tool of multi-level data

Version 1.28.0

Description Director is an R package designed to streamline the visualization of molecular effects in regulatory cascades. It utilizes the R package htmltools and a modified Sankey plugin of the JavaScript library D3 to provide a fast and easy, browser-enabled solution to discovering potentially interesting downstream effects of regulatory and/or co-expressed molecules. The diagrams are robust, interactive, and packaged as highly-portable HTML files that eliminate the need for third-party software to view. This enables a straightforward approach for scientists to interpret the data produced, and bioinformatics developers an alternative means to present relevant data.

URL <https://github.com/kzouchka/Director>

BugReports <https://github.com/kzouchka/Director/issues>

Depends R (>= 4.0)

License GPL-3 + file LICENSE

biocViews Visualization

LazyData true

Imports htmltools, utils, grDevices

RoxygenNote 5.0.1

NeedsCompilation no

git_url <https://git.bioconductor.org/packages/Director>

git_branch RELEASE_3_18

git_last_commit 26cac23

git_last_commit_date 2023-10-24

Repository Bioconductor 3.18

Date/Publication 2024-03-25

Author Katherine Icaay [aut, cre]

Maintainer Katherine Icaay <kat.icaay@gmail.com>

R topics documented:

| | |
|--------------------------|----|
| append2List | 2 |
| createList | 3 |
| drawSankey | 4 |
| filterNumeric | 5 |
| filterRelation | 6 |
| filterSubset | 7 |
| initSankey | 8 |
| makeSankey | 9 |
| mesenchymal | 10 |
| poorprog | 11 |
| writeSankey | 11 |

| | |
|--------------|-----------|
| Index | 13 |
|--------------|-----------|

| | |
|-------------|--------------------|
| append2List | <i>append2List</i> |
|-------------|--------------------|

Description

Appends a data frame containing additional relationship information to an existing List having 6 columns: source, target, description, value, sourcefc, targetfc. Order matters! For example, add a map of transcripts to genes to a List of miRNAs and their target transcripts so that the final List connects miRNAs -> transcripts -> genes.

Usage

```
append2List(List, appendList, description = "description",
             sourcefc = "sourcefc", targetfc = "targetfc", value = "value",
             target = "target", source = "source", appendMatch = TRUE)
```

Arguments

| | |
|-------------|---|
| List | Data frame containing the necessary columns above. e.g. Formatted with createList function. |
| appendList | Data frame or matrix to append to List. |
| description | Column name of appendList corresponding to the descriptions to append. |
| sourcefc | Column name of appendList corresponding to the sourcefcs to append. |
| targetfc | Column name of appendList corresponding to the targetfcs to append. |
| value | Column name of appendList corresponding to the relationship values to append. |
| target | Column name of appendList corresponding to the targets to append. |
| source | Column name of appendList corresponding to the sources to append. |
| appendMatch | Filter and remove 1) rows in List that contain targets without a corresponding source in appendList, and 2) rows in appendList that contain sources without a corresponding target in List. |

Value

a combined List.

Examples

```
tempList <- createList(data.frame(source=c("A","B","C"),
  target=c("D","E","G"),
  description=c("consonant","vowel","consonant"),
  value=runif(3,-1,1),
  sourcefc=runif(3,-2,2),
  targetfc=runif(3,-2,2)))
tempAppendList <- data.frame(source="D",target="I",
  description="vowel",value=runif(1,-1,1),
  sourcefc=runif(1,-2,2), targetfc=runif(1,-2,2))
append2List(tempList,tempAppendList) # Will combine only 1 row from each list.
append2List(tempList,tempAppendList, appendMatch=FALSE) # Will combine all rows
```

| | |
|-------------------------|-------------------|
| <code>createList</code> | <i>createList</i> |
|-------------------------|-------------------|

Description

Take a subset of the input data frame or matrix corresponding to the required Sankey values.

Usage

```
createList(inputList, inputFC = NULL, node = "genes", fc = "foldChange",
  source = "source", target = "target", description = "description",
  value = "value", sourcefc = "sourcefc", targetfc = "targetfc")
```

Arguments

| | |
|------------------------|---|
| <code>inputList</code> | Data frame or matrix containing the necessary parameters described below. |
| <code>inputFC</code> | Data frame or matrix containing node names (source and target) and corresponding quantitative values. If this input is defined, then input-specific parameters 'node' and 'fc' should be defined. Inputs 'source', 'target', 'description' and 'value' are still referenced from <code>inputList</code> . |
| <code>node</code> | Column name of <code>inputFC</code> containing names to display of source and target nodes. Paths defined in <code>inputList</code> identify which nodes are sources and which are targets. |
| <code>fc</code> | Column name of <code>inputFC</code> containing quantitative values representing the nodes. |
| <code>source</code> | Column name of <code>inputList</code> containing names to display of starting nodes. Paths are drawn from these points to their corresponding target nodes. |
| <code>target</code> | Column name of <code>inputList</code> containing names to display of destination nodes. Paths are drawn to these point from their corresponding source nodes. |

| | |
|-------------|---|
| description | Optional column name of inputList containing additional information about connection, e.g. the gene name of a transcript target node, or family name of related target genes. |
| value | Column name of inputList containing quantitative values representing the relationship between sources and targets. |
| sourcefc | Column name of inputList containing quantitative values representing the sources. |
| targetfc | Column name of inputList containing quantitative values representing the targets. |

Value

a data.frame List

Examples

```
nodevals <- runif(5,-2,2)
templist <- data.frame(source=c("A","B","C","D"),
  target=c("C","D","E","E"),
  addedInfo=c("c","d","vowel","vowel"),
  relationValue=runif(4,-1,1),
  sourceValue=nodevals[1:4],
  targetValue=nodevals[c(3,4,5,5)])
tempFC <- data.frame(genes=c("A","B","C","D","E"), foldChange=runif(5,-2,2))
# inputList only
createList(templist, description="addedInfo", value="relationValue",
  sourcefc="sourceValue", targetfc="targetValue")
# inputList and inputFC
createList(templist, tempFC, value="relationValue",sourcefc="sourceValue",
  targetfc="targetValue")
```

drawSankey

drawSankey

Description

Create an HTML document that can be viewed and saved to file. Diagram properties can be modified in this function, `makeSankey()` and `initSankey()`.

Usage

```
drawSankey(List, height = NULL, legendfont = "sans-serif",
  legendsize = 12, width = 1000, caption = "Sankey figure",
  nodeValue = "node values", pathValue = "path values", directory = NULL)
```

Arguments

| | |
|------------|--|
| List | Data frame containing 6 columns: source, target, description, value, sourcefc, targetfc. |
| height | Pixel height of the figure to draw. If empty, the figure will be given a pixel height proportional to the number of rows in List up to a maximum 1800px or minimum of 300px. These can be overridden by defining this parameter. |
| legendfont | Font of the legend text. |
| legendsize | Font size of the legend text. |
| width | Pixel width of the figure to draw. By default, 1000px. |
| caption | Sankey figure caption. HTML formatting is possible. |
| nodeValue | Description of node scale in legend. |
| pathValue | Description of path scale in legend. |
| directory | Absolute path to output directory. If null, the working directory obtained from getwd() will be used. This is required if D3 and sankey JS files were downloaded with initSankey(). |

Value

HTML document containing diagram.

Examples

```
Level1 <- createList(poorprog$Level1)
Level2 <- createList(poorprog$Level2)
templist <- append2List(Level1,Level2)
initSankey()
templist2 <- makeSankey(templist, averagePath=TRUE)
sankey <- drawSankey(templist2)
library(htmltools) # can also be launched with
html_print(sankey)
```

 filterNumeric

filterNumeric

Description

Filter a quantitative column in List for minimum, maximum, or absolute value.

Usage

```
filterNumeric(List, column, min = NULL, max = NULL, absolute = NULL)
```

Arguments

| | |
|----------|--|
| List | Data frame containing 6 columns: source, target, description, value, sourcefc, targetfc. |
| column | Name of column in List to filter. |
| min | Minimum value to filter for in column. |
| max | Maximum value to filter for in column. |
| absolute | Absolute value to filter for in column. |

Value

a filtered List

Examples

```
templist <- createList(data.frame(source=c("A","B","C"),
  target=c("D","E","G"),
  description=c("consonant","vowel","consonant"),
  value=runif(3,-1,1),
  sourcefc=runif(3,-2,2),
  targetfc=runif(3,-2,2)))
filterNumeric(templist,"sourcefc", absolute=0.5)
filterNumeric(templist, "targetfc", max=0) # only take down-regulated targets
```

| | |
|----------------|-----------------------|
| filterRelation | <i>filterRelation</i> |
|----------------|-----------------------|

Description

Filter source-target relationships in List for a specific type: inversely related sourcefc-targetfc pairs only (inverseFC), positively related sourcefc-targetfc pairs only (correlatedFC), negative value scores only (inverseValue), or positive value scores only (correlatedValue). Default is to not apply any filtering.

Usage

```
filterRelation(List, relation = c("none", "inverseFC", "correlatedFC",
  "inverseValue", "correlatedValue"), sourcefc = "sourcefc",
  targetfc = "targetfc", value = "value")
```

Arguments

| | |
|----------|--|
| List | Data frame containing 6 columns: source, target, description, value, sourcefc, targetfc. |
| relation | One of: none, inverseFC, correlatedFC, inverseValue, correlatedValue. Default is none. |
| sourcefc | Column name of List corresponding to sourcefcs to filter. |
| targetfc | Column name of List correspondig to targetfcs to filter. |
| value | Column name of List correspondig to value to filter. |

Value

a filtered List.

Examples

```
tempList <- createList(data.frame(source=c("A","B","C"),
  target=c("D","E","G"),
  description=c("consonant","vowel","consonant"),
  value=runif(3,-1,1),
  sourcefc=runif(3,-2,2),
  targetfc=runif(3,-2,2)))
filterRelation(tempList,"inverseValue")
filterRelation(tempList,"correlatedValue")
filterRelation(tempList,"inverseFC")
filterRelation(tempList,"correlatedFC")
```

 filterSubset

filterSubset

Description

Filter up to two qualitative columns (source and target) in List for a subset of names.

Usage

```
filterSubset(List, sourceSubset = NULL, targetSubset = NULL,
  invert = FALSE, source = "source", target = "target",
  join = c("union", "intersect"))
```

Arguments

| | |
|--------------|--|
| List | Data frame containing 6 columns: source, target, description, value, sourcefc, targetfc. |
| sourceSubset | Vector of source names to keep. |
| targetSubset | Vector of target names to keep. |
| invert | Take the inverse selection of defined subset. |
| source | Column name of List containing source names. |
| target | Column name of List containing target names. |
| join | If both subsets are defined, take either union or intersect of subsets found. |

Value

a filtered List

Examples

```
tempList <- createList(data.frame(source=c("A","B","C"),
  target=c("D","E","G"),
  description=c("consonant","vowel","consonant"),
  value=runif(3,-1,1),
  sourcefc=runif(3,-2,2),
  targetfc=runif(3,-2,2)))
filterSubset(tempList,source="source", target="description",
sourceSubset="C", targetSubset="consonant")
filterSubset(tempList,target="description", targetSubset="consonant")
filterSubset(tempList,target="description", targetSubset="consonant", invert=TRUE)
```

initSankey

initSankey

Description

Internally generates supporting JavaScript and CSS files.

Usage

```
initSankey(pathOpacity = 0.2, pathHover = 0.5,
  font = "lato, helvetica, sans-serif", fontsize = NULL,
  fontsizeProportion = TRUE, d3js = NULL, sankeyjsFile = NULL,
  d3jsMethod = "auto", sankeyjsMethod = "auto")
```

Arguments

| | |
|--------------------|--|
| pathOpacity | Opacity of connecting path between nodes in the figure. |
| pathHover | Opacity of connecting path between nodes upon mouseover. |
| font | Font used for the node names and additional mouseover text in figure. |
| fontsize | Pixel font size used for the visible node names. Use to adjust range of font sizes (with proportions) or to set a single font size when fontsizeProportion is disabled. |
| fontsizeProportion | Boolean to enable/disable text being proportional to node widths. When enabled, all node names will appear with parameter fontsize. |
| d3js | Path to download latest zip version of D3 library. e.g. https://github.com/mbostock/d3/releases/download/ See http://www.d3js.org for more details. If NULL, will use version 3.5.16 currently installed with Director. |
| sankeyjsFile | Path to download sankey javascript file. If NULL, will use version installed with Director (https://raw.githubusercontent.com/d3/d3-plugins/master/sankey/sankey.js) |
| d3jsMethod | Function method to use to download D3 library. ?download.file for more detail on parameter. |
| sankeyjsMethod | Function method to use to download sankey script. ?download.file for more detail on parameter. |

Value

global JavaScript and CSS files.

Examples

```
initSankey() # Generates supporting JavaScript and CSS files.
```

makeSankey

makeSankey

Description

Takes a list of source-target pairs and assigns colours to nodes and connections based on value, sourcefc and targetfc. Output is a list with List\$reference = input List with additional description values, \$valDomain = path values, \$valRange = path colours, \$targetDomain = target names, \$targetRange = target node colours, \$sourceDomain = source names, \$sourceRange = source node colours.

Usage

```
makeSankey(List, averagePath = FALSE, nodeMin = "blue", nodeMax = "red",
  pathMin = "blue", pathMax = "red", noughtColor = "#f5f5f0",
  nought = 0, noughtPath = NULL, noughtPathColor = NULL)
```

Arguments

| | |
|-----------------|--|
| List | Data frame containing 6 columns: source, target, description, value, sourcefc, targetfc. |
| averagePath | Boolean to either keep List\$value as-is, or calculate List\$value for intermediary nodes (i.e. source nodes that were previously target nodes) as an average of previous path List\$values. |
| nodeMin | Colour assigned to minimum node value. |
| nodeMax | Colour assigned to maximum node value. |
| pathMin | Colour assigned to minimum path value. |
| pathMax | Colour assigned to maximum path value. |
| noughtColor | Colour assigned to nought value. |
| nought | 'Zero' value dividing node and paths into two distinct sets. i.e. positive and negative. |
| noughtPath | Optional parameter that sets a different 'zero' value for paths than for nodes. |
| noughtPathColor | Optional parameter that assigns a different colour to the path 'zero' value from the node 'zero' value. |

Value

a list of data.frames and colour vectors.

Examples

```
templist <- createList(data.frame(source=c("A","B","C"),
  target=c("D","E","G"),
  description=c("consonant","vowel","consonant"),
  value=runif(3,-1,1),
  sourcefc=runif(3,-2,2),
  targetfc=runif(3,-2,2)))
initSankey()
templist2 <- makeSankey(templist)
```

mesenchymal

Analysis results of Yang et al.'s (2013) master microRNA regulatory network

Description

Data frames listing a set of genes differentially expressed between mesenchymal and three other serous ovarian cancer subtypes, eight key miRNAs predicted to target them, and significantly enriched pathways (FDR < 0.1). Each row contains a miRNA-gene/gene-pathway pair, a description, expression correlation (path values), and expression fold-change (node values).

Usage

```
ovca
```

Format

a list instance containing 2 data frames.

Value

data frame

Source

The Cancer Genome Atlas. Yang et al., 2013.

| | |
|----------|--|
| poorprog | <i>Analysis results for poor prognosis serous ovarian cancer</i> |
|----------|--|

Description

Data frames listing a set of genes differentially expressed between long surviving (good prognosis) and short surviving (poor prognosis) cases, their putative targeting miRNAs, and significantly enriched pathways (FDR < 0.1). Each row contains a miRNA-gene/gene-pathway pair, a description, expression correlation (path values), and expression fold-change (node values).

Usage

```
ovca
```

Format

a list instance containing 2 data frames.

Value

data frame

Source

The Cancer Genome Atlas

| | |
|-------------|--------------------|
| writeSankey | <i>writeSankey</i> |
|-------------|--------------------|

Description

Save sankey figure as a simple HTML file accessible outside of R and shiny. Functions `initSankey`, `makeSankey` and `drawSankey` must be performed before this step to ensure a proper figure is saved.

Usage

```
writeSankey(name = NULL, title = NULL, directory = NULL)
```

Arguments

| | |
|-----------|---|
| name | Name to give file. If no path given, the working directory OR path set in <code>Director</code> will be used. Same name will be given as the title. |
| title | Title of the HTML file produced. The file name is used by default. |
| directory | Absolute path to output directory. If null, the working directory obtained from <code>getwd()</code> will be used. If no absolute path is given (i.e. no "/" is grepped), it will assume a new folder will be created in the working directory. |

Value

a dynamic HTML file in the specified directory that is readable in any internet browser so long as the 'www' subfolder is included.

Examples

```
Level1 <- createList(poorprog$Level1)
Level2 <- createList(poorprog$Level2)
templist <- append2List(Level1,Level2)
initSankey() # initializes working directory
templist2 <- makeSankey(templist, averagePath=TRUE) # Calculate node and path values
sankey <- drawSankey(templist2)
writeSankey("temp") # Save figure as the file 'temp.html' in working directory.
```

Index

- * **append**
 - append2List, 2
- * **download**
 - initSankey, 8
- * **draw**
 - drawSankey, 4
- * **filter**
 - filterNumeric, 5
 - filterRelation, 6
 - filterSubset, 7
- * **html**
 - writeSankey, 11
- * **input**
 - createList, 3
- * **sankey**
 - drawSankey, 4
 - initSankey, 8
 - makeSankey, 9
 - writeSankey, 11

append2List, 2

createList, 3

drawSankey, 4

filterNumeric, 5

filterRelation, 6

filterSubset, 7

initSankey, 8

makeSankey, 9

mesenchymal, 10

poorprog, 11

writeSankey, 11