Package 'clevRvis'

May 1, 2024

Type Package

Title Visualization Techniques for Clonal Evolution

Version 1.4.0

Description clevRvis provides a set of visualization techniques for clonal evolution. These include shark plots, dolphin plots and plaice plots. Algorithms for time point interpolation as well as therapy effect estimation are provided. Phylogeny-aware color coding is implemented. A shiny-app for generating plots interactively is additionally provided.

License LGPL-3

Encoding UTF-8

LazyData false

Requires shiny, ggraph, igraph, ggiraph, cowplot, htmlwidgets, readxl, dplyr, readr, purrr, tibble, patchwork, R.utils, shinyWidgets, colorspace, shinyhelper, shinycssloaders, ggnewscale, shinydashboard, DT, colourpicker, grDevices, methods, utils, stats, ggplot2, magrittr, tools

Imports shiny, ggraph, igraph, ggiraph, cowplot, htmlwidgets, readxl, dplyr, readr, purrr, tibble, patchwork, R.utils, shinyWidgets, colorspace, shinyhelper, shinycssloaders, ggnewscale, shinydashboard, DT, colourpicker, grDevices, methods, utils, stats, ggplot2, magrittr, tools

VignetteBuilder knitr

Suggests knitr, rmarkdown, BiocStyle

biocViews Software, ShinyApps, Visualization

BugReports https://github.com/sandmanns/clevRvis/issues

URL https://github.com/sandmanns/clevRvis

RoxygenNote 7.2.1

git_url https://git.bioconductor.org/packages/clevRvis

git_branch RELEASE_3_19

git_last_commit ad5fe16

git_last_commit_date 2024-04-30

Repository Bioconductor 3.19

Date/Publication 2024-05-01

Author Sarah Sandmann [aut, cre] (<https://orcid.org/0000-0002-5011-0641>)

Maintainer Sarah Sandmann <sarah.sandmann@uni-muenster.de>

Contents

`clevRvisShiny` *A user interface to perform all analyses with clevRvis.*

Description

clevRvis is a tool that allows you to visualize changes in the subclonal architecture of tumors. Simple tree visualization (shark plot) a more detailed visualization (dolphin plot) and an allele-aware visualization (plaice plot) are available. Moreover, this tool provides fully automatic algorithms for time point interpolation and therapy effect estimation in the presence of lacking input data.

Usage

```
clevRvisShiny()
```

Arguments

None

Details

Detailed information on the usage of clevRvisShiny are available within the shiny app (see Tutorial).

Value

None

Author(s)

Sarah Sandmann <sarah.sandmann@uni-muenster.de>

Examples

```
if(interactive()){
    clevRvisShiny()
}
```

---

clevRvis_package *Visualization Techniques for Clonal Evolution*

Description

clevRvis is a tool that allows you to visualize changes in the subclonal architecture of tumors. Simple tree visualization (shark plot) a more detailed visualization (dolphin plot) and an allele-aware visualization (plaice plot) are available. Moreover, this tool provides fully automatic algorithms for interpolating time points and estimating therapy effect in the presence of lacking input data.

Details

The package contains a function performing the whole analysis using a shiny user interface - `clevRvisShiny`.

Additionally, all functions for classical use in R are available:

1) `createSeaObject`: create the seaObject needed for further visualization options

2) `sharkPlot`: basic graph visualization (shark plot) of clonal evolution.

3) `extSharkplot`: basic graph visualization with extension showing Cancer Cell Fractions (CCFs) as point size for each clone at each selected time point.

4) `dolphinPlot`: detailed visualization (dolphin plot) of clonal evolution showing the development of all clones over time and their clonal prevalences.

5) `combinedPlot`: interactive linked visualization of shark and dolphin plot together.

6) `plaicePlot`: detailed visualization on the allelic level (plaice plot) of clonal evolution. Clonal prevalences on the top half and percentage of remaining healthy alleles on the bottom half.

7) `exploreTrees`: Generate alternative parental relations to explore alternative trees

Author(s)

Sarah Sandmann <sarah.sandmann@uni-muenster.de>

See Also

`clevRvisShiny`, `createSeaObject`, `sharkPlot`, `extSharkPlot`, `dolphinPlot`, `combinedPlot`, `plaicePlot`, `exploreTrees`

Examples

```
timepoints <- c(0,50,100)
parents <- c(0,1,1,3,0,5,6)
fracTable <- matrix(
                    c(  20, 10,  0,  0,  0,  0,  0,
                        40, 20, 15,  0, 30, 10,  0,
                        50, 25, 15, 10, 40, 20, 15),
                    ncol = length(timepoints))

#Generating a seaObject with timepoint interpolation
seaObject <- createSeaObject(fracTable, parents, timepoints)


#Basic shark plot showing legend and title
#png('basicShark.png',height = 800, width=600)
sharkPlot(seaObject, showLegend = TRUE, main = 'Example Shark plot')
#dev.off()

#extended shark plot, showing CCF as point size only for measured timepoints,
#legend and title
#png('extendedShark.png',height = 700, width = 1500)
extSharkPlot(seaObject, timepoints = timepoints, showLegend = TRUE,
            main = 'Example Extended Shark plot')
#dev.off()

#Default dolphin plot, with vertical lines showing all time points, custom
#y axis label and triangles indicating the measured time points
#png('dolphinPlot.png',height = 800, width=1750)
dolphinPlot(seaObject, showLegend = TRUE, vlines = slot(seaObject,"timepoints"),
            vlab = slot(seaObject,"timepoints"), vlabSize = 2,
            ylab = 'Cancer cell fractions (CCFs)',
            markMeasuredTimepoints = timepoints)
#dev.off()

#Basic shark plot linked to dolphin plot
combinedPlot(seaObject, showLegend = TRUE, vlines = timepoints,
            vlab = timepoints, vlabSize = 2, ylab = 'Cancer cell fraction',
            separateIndependentClones = TRUE)

#static plaice plot showing biallelic events + annotations
annotsTable <- data.frame(x = c(24,55), y = c(-40,-5),
                          col = c('black', 'white'), lab = c('TP53', 'UBA1'))
#png('plaicePlot.png',height = 800, width = 1750)
plaicePlot(seaObject, showLegend = TRUE,  vlines = timepoints,
            vlab = timepoints, vlabSize = 4, ylab = TRUE,
            separateIndependentClones = TRUE, clonesToFill = c(0,0,1,0,0,6,0),
            annotations = annotsTable, interactivePlot = FALSE)
#dev.off()

#seaObject with enabled timepoint interpolation and therapy effect estimation
#between timepoint 50 and 100
seaObject <- createSeaObject(fracTable, parents, timepoints,
```

```
                          timepointInterpolation = TRUE,
                          therapyEffect = c(50,100))

#Default dolphin plot showing estimated therapy effect, with vertical
#lines showing all time points, custom y axis label and triangles indicating
#the measured time points
#png('dolphinPlotTherapy.png',height = 800, width=1750)
dolphinPlot(seaObject, showLegend = TRUE, vlines = slot(seaObject,"timepoints"),
            vlab = slot(seaObject,"timepoints"), vlabSize = 2,
            ylab = 'Cancer cell fractions (CCFs)',
            markMeasuredTimepoints = timepoints)
#dev.off()

#Explore alternative valid trees
timepoints <- c(0,30,75,150)
fracTable <- matrix(
            c( 100, 45,  0,  0,
                20,  0,  0,  0,
                30,  0, 20,  5,
                98,  0, 55, 40),
            ncol=length(timepoints))
trees <- exploreTrees(fracTable, timepoints)
```

`combinedPlot` *Generate a combined basic graph and detailed visualization of clonal evolution*

Description

Given a sea object containing layout information, a shark and dolphin plot can be plotted together - linked and interactive.

Usage

```
combinedPlot(seaObject,
             shark = TRUE,
             dolphin = TRUE,
             shape = "spline",
             borderCol = NULL,
             vlines = NULL,
             vlineCol = "#6E6E66",
             vlab = NULL,
             vlabSize = 3,
             pos = "center",
             separateIndependentClones = FALSE,
             showLegend = FALSE,
             markMeasuredTimepoints = NULL,
             downloadWidget = NULL,
```

```
              mainDph = NULL,
              mainPosDph = "middle",
              mainSizeDph = 5,
              mainShk = NULL,
              xlab = NULL,
              ylab = NULL,
              pad.left = 0.005,
              annotations = NULL,
              width = 12,
              height = 9)
```

Arguments

seaObject	A seaObject.
shark	A boolean defining whether or not to draw a shark plot (default: TRUE).
dolphin	A boolean defining whether or not to draw a dolphin plot (default: TRUE).
shape	The type of shape to construct the plot out of. The options are "spline" and "polygon" (default: "spline"").
borderCol	A color for the border line. If "NULL" then no border will be drawn (default: NULL).
vlines	A vector of positions at which to draw vertical lines (default: NULL).
vlineCol	A color value for the vertical lines (default: "#6E6E66").
vlab	A character vector containing labels for each of the vertical lines (default: NULL).
vlabSize	An integer value for the vertical labels size (default: 3).
pos	Plotting position of the clones. Options are "center", "bottom" or "top" (default: "center").
separateIndependentClones	Boolean defining whether independently-arising clones (with parent 0) should be separated by blank space in the plot (default: FALSE).
showLegend	A boolean indicating whether to show a legend at the left side of the plot (default: FALSE).
markMeasuredTimepoints	A vector of x positions at which to draw triangles on the bottom of the plot (default: NULL).
downloadWidget	File to safe HTML to (default: NULL).
mainDph	A string corresponding to the dolphin plot's main title (default: NULL).
mainPosDph	A string defining the dolphin plot's title position. Options are 'left', 'middle' or 'right', always above the plot (default: "middle").
mainSizeDph	An integer value defining the size of the dolphin plot's title (default: 5).
mainShk	A string corresponding to the shark plot's main title (default: NULL).
xlab	A string defining the label of the x axis (default: NULL).
ylab	A string defining the label of the y axis. Automatically, a vertical line showing 100% will be plotted (default: NULL).

pad.left	The amount of "ramp-up" to the left of the first time point. Given as a fraction of the total plot width (default: 0.005).
annotations	A data.frame with: columns "x" (x position), "y" (y position), "lab" (annotation text) and "col" (color of the text either black or white) (default: NULL).
width	An integer value indicating the with of the output widget (default: 12).
height	An integer value indicating the height of the output widget (default: 9).

Details

Dolphin plots may be chosen to be plotted along with basic shark plots (for details see dolphinPlot and sharkPlot). Both plots are internally connected. By hovering on one of the clones, it is automatically highlighted in both, shark and dolphin plot.

Important note: extended shark plots and dolphin plots can NOT be visualized together.

Value

None

Author(s)

Sarah Sandmann <sarah.sandmann@uni-muenster.de>

Examples

```
timepoints <- c(0,30,75,150)
fracTable <- matrix(
            c( 100, 45, 00, 00,
                02, 00, 00, 00,
                03, 00, 02, 01,
                98, 00, 95, 40),
            ncol=length(timepoints))
parents <- c(0,1,1,3)
seaObject <- createSeaObject(fracTable = fracTable,
                             parents = parents,
                             timepoints = timepoints,
                             timepointInterpolation = TRUE)

combinedPlot(seaObject, borderCol = 'white', showLegend = TRUE)
```

---

createSeaObject *Create a seaObject*

Description

clevRvis needs a seaObject for the visualization of clonal evolution by means of any available plot. When generating the seaObject, extra time points may be interpolated, therapy effect may be estimated.

Usage

```
createSeaObject(fracTable,
                parents,
                timepoints,
                col = NULL,
                cloneLabels = NULL,
                originTimepoint = NULL,
                timepointInterpolation = TRUE,
                therapyEffect = NULL)
```

Arguments

fracTable	A numeric matrix containing tumor fraction estimates for all clones at all time points.
parents	An integer vector specifying parental relationships between clones.
timepoints	A numeric vector specifying the time points for each column of the matrix.
col	A vector of colors to use when plotting each clone (default: NULL).
cloneLabels	A character vector of names to assign to each clone when plotting a legend (default: NULL).
originTimepoint	Time point when the first clone emerges (must be before the first measured time point) (default: NULL).
timepointInterpolation	When set to true extra time points will be estimated between measured time points and before the first measure time point to improve the visualization (default: TRUE).
therapyEffect	A single numeric value indicating the time point when to estimate the effect of therapy or a numeric vector containing two consecutive measured time points, therapy effect time point will be in the middle (default: NULL).

Details

The basis for all plotting functions included in clevRvis is a seaObject. It contains information on the CCFs for all clones, at all measured time points. Additionally, parental information on the clones is included.

Additional time points may be interpolated, therapy effect may be estimated when generating seaObjects.

Time point interpolation is generally recommended to improve visualization of clonal evolution. When having less time points than clones, or many new clones emerging in one single measured time point, the extra time point estimation is strongly recommended to visualize the clonal evolution properly. If there is only one measured time point, time point interpolation is required and the time point of origin has to be specified manually, as there is no way of calculating it.

To visualize the effect of therapy on the clones' CCFs in case of missing measured data, a fully automatic approach for therapy effect estimation is available. When creating the seaObject, a specific time point can be defined (between two measured time points) or two measured time points can be selected (new therapy effect time point will be in the middle) for the estimation of the therapy effect.

Value

A sea Object with all relevant slots filled.

Author(s)

Sarah Sandmann <sarah.sandmann@uni-muenster.de>

Examples

```
timepoints <- c(0,30,75,150)
fracTable <- matrix(
            c( 100, 45, 00, 00,
                02, 00, 00, 00,
                03, 00, 02, 01,
                98, 00, 95, 40),
            ncol=length(timepoints))
parents <- c(0,1,1,3)
seaObject <- createSeaObject(fracTable = fracTable,
                             parents = parents,
                             timepoints = timepoints)
```

`dolphinPlot` *Generate a detailed visualization of clonal evolution*

Description

Dolphin plots provide a detailed visualization of clonal evolution. Plots show the development of all clones over time (x axis) and their clonal prevalences (y axis).

Usage

```
dolphinPlot(seaObject,
            shape = "spline",
            borderCol = NULL,
            pos = "center",
            vlines = NULL,
            vlineCol = "#6E6E66",
            vlab = NULL,
            vlabSize = 3,
            separateIndependentClones = FALSE,
            showLegend = FALSE,
            markMeasuredTimepoints = NULL,
            main = NULL,
            mainPos = "middle",
            mainSize = 5,
            xlab = NULL,
            ylab = NULL,
```

```
              pad.left = 0.005,
              annotations = NULL,
              annotSize = 3)
```

Arguments

seaObject	A seaObject.
shape	The type of shape to construct the plot out of. The options are "spline" and "polygon" (default: "spline").
borderCol	A color for the border line. If "NULL" then no border will be drawn (default: NULL).
pos	Plotting position of the clones. Options are "center" or "bottom" (default: "center").
vlines	A vector of positions at which to draw vertical lines (default: NULL).
vlineCol	A color value for the vertical lines (default: "#6E6E66").
vlab	A character vector containing labels for each of the vertical lines (default: NULL).
vlabSize	An integer value for the vertical labels size (default: 3).
separateIndependentClones	Boolean defining whether independently-arising clones (with parent 0) should be separated by blank space in the plot (default: FALSE).
showLegend	A boolean indicating whether to show a legend at the left side of the plot (default: FALSE).
markMeasuredTimepoints	A vector of x positions at which to draw triangles on the bottom of the plot (default: NULL).
main	A string corresponding to the plot's main title (default: NULL).
mainPos	A string defining the title's position. Options are 'left', 'middle' or 'right', always above the plot (default: "middle").
mainSize	An integer value defining the size of the title (default: 5).
xlab	A string defining the label of the x axis (default: NULL).
ylab	A string defining the label of the y axis. Automatically, a vertical line showing 100% will be plotted (default: NULL).
pad.left	The amount of "ramp-up" to the left of the first time point. Given as a fraction of the total plot width (default: 0.005).
annotations	A data.frame with: columns "x" (x position), "y" (y position), "lab" (annotation text) and "col" (color of the text either black or white) (default: NULL).
annotSize	An integer value defining the size of the annotations (default: 3).

Details

Dolphin plots displays detailed information on clonal evolution, showing the development of all clones over time (x axis) and their clonal prevalence (y axis). Information on phylogeny, CCFs and time course characterizing a clonal evolution are jointly visualized in this single plot.

Several basic options for customizing dolphin plots are available, e.g. switching between spline and polygon shape, bottom or central visualization, annotations, separating independent clones, adding vertical lines and labels, changing border and vertical lines colors, etc.

Dolphin plots may be chosen to be plotted along with basic shark plots (see `combinedPlot`).

Value

None

Author(s)

Sarah Sandmann <sarah.sandmann@uni-muenster.de>

Examples

```
timepoints <- c(0,30,75,150)
fracTable <- matrix(
            c( 100, 45, 00, 00,
                02, 00, 00, 00,
                03, 00, 02, 01,
                98, 00, 95, 40),
            ncol=length(timepoints))
parents <- c(0,1,1,3)
seaObject <- createSeaObject(fracTable = fracTable,
                             parents = parents,
                             timepoints = timepoints,
                             timepointInterpolation = TRUE)

dolphinPlot(seaObject, main = 'Example Dolphin Plot', pos = 'center')
```

`exploreTrees` *Explore alternative trees*

Description

clevRvis allows to explore alternative trees, i.e. alternative parental relations. Taking information on the CCFs and available time points as input, all possible parental relations are investigated and checked for validity.

Usage

```
exploreTrees(fracTable,
             timepoints)
```

Arguments

`fracTable`	A numeric matrix containing tumor fraction estimates for all clones at all time points.
`timepoints`	A numeric vector specifying the timepoints for each column of the matrix.

Details

To create a seaObject, the basis for all plotting functions in clevRvis, a fracTable, a timepoints vector and a parents vector are required. clevRvis provides an approach to determine all valid parental relations on the basis of the information provided in fracTable and the timepoints vector. Thereby, alternative trees can be explored.

To optimize run-time, the analysis is devided into 3 step procedure:

1) Possible parents are determined. If clone 1 has at any measured time point a lower CCF compared to clone 2, then clone 1 cannot be clone 2's parent.

2) Possible branched dependent evolution is investigated. If clone 2 can only develope from clone 1, the difference in CCFs for clone 1 and clone 2 is calculated. Every remaining clone with a CCF larger than the difference cannot develop from clone 1.

3) All remaining, possible parental relations are determined. An extensive validity check is performed using clevRvis (validity check when creating a seaObject). A maximum of 20,000 parental relations is investigated.

Value

A list of numeric vectors containing valid parental relations, apt to explore alternative trees.

Author(s)

Sarah Sandmann <sarah.sandmann@uni-muenster.de>

Examples

```
timepoints <- c(0,30,75,150)
fracTable <- matrix(
            c( 100, 45,  0,  0,
                20,  0,  0,  0,
                30,  0, 20,  5,
                98,  0, 55, 40),
            ncol=length(timepoints))

trees <- exploreTrees(fracTable, timepoints)
```

`extSharkPlot` *Generate an extended graph visualization of clonal evolution*

Description

An extended shark plot shows the basic graph visualization of clonal evolution with nodes representing clones and edges indicating their evolutionary relations and additional visualization of CCFs.

Usage

```
extSharkPlot(seaObject,
             showLegend = FALSE,
             main = NULL,
             timepoints = NULL,
             width = 10,
             interactivePlot = TRUE)
```

Arguments

seaObject	A seaObject.
showLegend	A boolean indicating whether to show the legend or not (default: FALSE).
main	A string corresponding to the plot's main title (default: NULL).
timepoints	By default, all time points available in the seaObject are visualized. Optionally, a selected set of available time points can be chosen (default: NULL).
width	An integer value indicating the width of the widget plot (default: 10).
interactivePlot	A boolean defining whether the plot should be interactive (default: TRUE; if using this function to export the extended shark plot, e.g. by png(), define interactivePlot = FALSE).

Details

An extended shark plots consists of two elements:

1) A basic shark plot: common trees with nodes representing clones and edges indicating their evolutionary relation. Phylogeny can be directly deduced from these plots.

2) Additionally, CCFs of each clone (rows) at each time point (columns) are shown as points next to the basic shark plot. The size of each point correlates with the CCF at the corresponding clone and time point.

Both plots are linked in an interactive widget.

Value

None

Author(s)

Sarah Sandmann <sarah.sandmann@uni-muenster.de>

Examples

```
timepoints <- c(0,30,75,150)
fracTable <- matrix(
            c( 100, 45, 00, 00,
                02, 00, 00, 00,
                03, 00, 02, 01,
                98, 00, 95, 40),
```

```
            ncol=length(timepoints))
parents <- c(0,1,1,3)
seaObject <- createSeaObject(fracTable = fracTable,
                             parents = parents,
                             timepoints = timepoints)

extSharkPlot(seaObject, showLegend = TRUE, main = 'Example Shark Plot')
```

---

plaicePlot *Generate an allele-aware visualization of clonal evolution*

Description

Plaice plots provide an allele-aware visualization of clonal evolution. Plots show the development of all clones over time (x axis) and their clonal prevalences (y axis), and the ratio of remaining healthy alleles (lower plaice).

Usage

```
plaicePlot(seaObject,
            shape = "spline",
            borderCol = "black",
            vlines = NULL,
            vlineCol = "#6E6E66",
            vlab = NULL,
            vlabSize = 3,
            separateIndependentClones = FALSE,
            clonesToFill = NULL,
            showLegend = FALSE,
            markMeasuredTimepoints = NULL,
            main = NULL,
            mainPos = "middle",
            mainSize = 5,
            xlab = NULL,
            ylab = FALSE,
            pad.left = 0.005,
            annotations = NULL,
            annotationsSize = 3,
            interactivePlot = TRUE)
```

Arguments

seaObject	A seaObject.
shape	The type of shape to construct the plot out of. The options are "spline" and "polygon" (default: "spline"").
borderCol	A color for the border line. If "NULL" then no border will be drawn (default: "black").

`vlines` A vector of positions at which to draw vertical lines (default: NULL).

`vlineCol` A color value for the vertical lines (default: "#6E6E66").

`vlab` A character vector containing labels for each of the vertical lines (default: NULL).

`vlabSize` An integer value for the vertical labels size (default: 3).

`separateIndependentClones`
Boolean defining whether independently-arising clones (with parent 0) should be separated by blank space in the plot (default: FALSE).

`clonesToFill` An integer vector with the index of the clone's color to fill each clone. For example: clonesToFill <- c(0,0,0,2,0,0) clone 4 (and its children) will be filled with clone 2 color (default: NULL).

`showLegend` A boolean indicating whether to show a legend at the left side of the plot (default: FALSE).

`markMeasuredTimepoints`
A vector of x positions at which to draw triangles on the bottom of the plot (default: NULL).

`main` A string corresponding to the plot's main title (default: NULL).

`mainPos` A string defining the title's position. Options are 'left', 'middle' or 'right', always above the plot (default: "middle").

`mainSize` An integer value defining the size of the title (default: 5).

`xlab` A string defining the label of the x axis (default: NULL).

`ylab` A boolean defining whether or not to show the default y axis labels (default: FALSE).

`pad.left` The amount of "ramp-up" to the left of the first time point. Given as a fraction of the total plot width (default: 0.005).

`annotations` A data.frame with: columns "x" (x position), "y" (y position), "lab" (annotation text) and "col" (color of the text either black or white) (default: NULL).

`annotationsSize`
An integer value defining the size of the annotations (default: 3).

`interactivePlot`
A boolean defining whether the plot should be interactive (default: TRUE; if using this function to export the plaice plot, e.g. by png(), define interactivePlot = FALSE).

Details

Plaice plots are based on the bottom visualization of dolphin plots ("flatfish" = plaice), mirrored above and below the x-axis . They have been developed to improve the visualization of biallelic events.

Clonal evolution can be visualized in the upper half of the plot. Several options similar to the ones for dolphin plot are available as well for plaice plot: switching between spline and polygon shape, annotations, separating independent clones (recommended), adding vertical lines and labels, etc.

On the bottom half of the plot, a not-colored mirrored representation of clonal evolution is plotted. The user may choose the clones and their color, in order to show which clones caused biallelic events. When a clone is not colored, it indicates that a healthy allele remains.

Variants affecting the only available X- or Y-chromosome in male subjects can also be visualized using plaice plots.

Value

None

Author(s)

Sarah Sandmann <sarah.sandmann@uni-muenster.de>

Examples

```
timepoints <- c(0,30,75,150)
fracTable <- matrix(
            c( 100, 45, 00, 00,
                02, 00, 00, 00,
                03, 00, 02, 01,
                98, 00, 95, 40),
            ncol=length(timepoints))
parents <- c(0,1,1,3)
seaObject <- createSeaObject(fracTable = fracTable,
                             parents = parents,
                             timepoints = timepoints,
                             timepointInterpolation = TRUE)

plaicePlot(seaObject, shape = "spline",
            vlines = c(0,150), vlab = c("day 0","day 150"),
            main = 'Example plot', clonesToFill = c(0,1,0))
```

---

seaObject-class *Class seaObject*

Description

Represents a seaObject class, containing all necessary input to generate shark plots, dolphin plots and plaice plots

Methods

show `show(seaObject)`: summary of the seaObject.

ytop `ytop(seaObject)`: get value of ytop.

ytop<- `ytop(seaObject)<-`: assign value to ytop.

ybtm `ybtm(seaObject)`: get value of ybtm.

ybtm<- `ybtm(seaObject)<-`: assign value to ybtm.

xpos(x) `xpos(seaObject)`: get value of xpos

xpos(x)<- `xpos(seaObject)`: assign value to xpos

col(x) `col(seaObject)`: get value of col

col(x)<- `col(seaObject)`: assign value to col

timepoints(x) `timepoints(seaObject)`: get value of timepoints

timepoints(x)<- `timepoints(seaObject)`: assign value to timepoints

fracTable(x) `fracTable(seaObject)`: get value of fracTable

fracTable(x)<- `fracTable(seaObject)`: assign value to fracTable

parents(x) `parents(seaObject)`: get value of parents

parents(x)<- `parents(seaObject)`: assign value to parents

nestLevels(x) `nestLevels(seaObject)`: get value of nestLevels

nestLevels(x)<- `nestLevels(seaObject)`: assign value to nestLevels

cloneFamily(x) `cloneFamily(seaObject)`: get value of cloneFamily

cloneFamily(x)<- `cloneFamily(seaObject)`: assign value to cloneFamily

cloneLabels(x) `cloneLabels(seaObject)`: get value of cloneLabels

cloneLabels(x)<- `cloneLabels(seaObject)`: assign value to cloneLabels

defaultLabels(x) `defaultLabels(seaObject)`: get value of defaultLabels

defaultLabels(x)<- `defaultLabels(seaObject)`: assign value to defaultLabels

originTimepoint(x) `originTimepoint(seaObject)`: get value of originTimepoint

originTimepoint(x)<- `originTimepoint(seaObject)`: assign value to originTimepoint

Author(s)

Sarah Sandmann <sarah.sandmann@uni-muenster.de>

Examples

```
timepoints <- c(0,30,75,150)
fracTable <- matrix(
            c( 100, 45, 00, 00,
                02, 00, 00, 00,
                03, 00, 02, 01,
                98, 00, 95, 40),
            ncol=length(timepoints))
parents <- c(0,1,1,3)
seaObject <- createSeaObject(fracTable = fracTable,
                             parents = parents,
                             timepoints = timepoints)
timepoints(seaObject)
timepoints(seaObject) <- c(0,20,75,150)
```

`sharkPlot` *Generate a basic graph visualization of clonal evolution*

Description

A shark plot shows the basic graph visualization of clonal evolution with nodes representing clones and edges indicating their evolutionary relations.

Usage

```
sharkPlot(seaObject,
            showLegend = FALSE,
            main = NULL)
```

Arguments

`seaObject`	A seaObject.
`showLegend`	A boolean indicating whether to show the legend or not (default: FALSE).
`main`	A string corresponding to the plot's main title (default: NULL).

Details

A shark plot is the basic approach for visualization: common trees, with nodes representing clones and edges indicating their evolutionary relation. Phylogeny can be directly deduced from these plots.

Shark plots also offer an extension to visualize the changes in CCF along time for each clone. CCFs of each clone (rows) at each time point (columns) are shown as points next to the basic shark plot (see `extSharkPlot`).

Value

None

Author(s)

Sarah Sandmann <sarah.sandmann@uni-muenster.de>

Examples

```
timepoints <- c(0,30,75,150)
fracTable <- matrix(
            c( 100, 45, 00, 00,
                02, 00, 00, 00,
                03, 00, 02, 01,
                98, 00, 95, 40),
            ncol=length(timepoints))
parents <- c(0,1,1,3)
seaObject <- createSeaObject(fracTable = fracTable,
```

```
                              parents = parents,
                              timepoints = timepoints)

sharkPlot(seaObject, showLegend = TRUE, main = 'Example Shark Plot')
```

Index