Package 'flowPlots'

May 1, 2024

Type Package

Version 1.52.0

Title flowPlots: analysis plots and data class for gated flow cytometry data

Description Graphical displays with embedded statistical tests for gated ICS flow cytometry data, and a data class which stores ``stacked'' data and has methods for computing summary measures on stacked data, such as marginal and polyfunctional degree data.

Author N. Hawkins, S. Self

Maintainer N. Hawkins <hawkins@fhcrc.org>

Depends R (>= 2.13.0), methods

LazyData yes

License Artistic-2.0

biocViews ImmunoOncology, FlowCytometry, CellBasedAssays, Visualization, DataRepresentation

Suggests vcd

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

git_branch RELEASE_3_19

git_last_commit c3af73a

git_last_commit_date 2024-04-30

Repository Bioconductor 3.19

Date/Publication 2024-05-01

Contents

`adultsNeonates` *The adultsNeonates example data set of "stacked" data.*

Description

The adultsNeonates data is an example of stacked data from an ICS Flow Cytometry assay.

Usage

```
data(adultsNeonates)
```

Value

data frame with 1 column of marker combination percentages and several cols of 'demographic' data describing the percentage in a given row.

Author(s)

N. Hawkins, Fred Hutchinson Cancer Research Center, Seattle, WA

References

TR Kollmann, J Crabtree, A Rein-Weston, D Blimkie, F Thomai, X Wang, J Furlong, E Fortuno III, A Hajjar, N Hawkins, S Self, and C Wilson, The neonatal innate immune system is not less responsive than the adult but responds differently, 2009, J Immunology, 183, 7150-7160

See Also

StackedData

Examples

```
# Load adultsNeoates data
data(adultsNeonates)
```

---

computeMarginalData-methods
Method computeMarginalData from Class "StackedData"

Description

This function is a method of the StackedData class which computes the marginal data which can be stored in the marginalData slot of a StackedData object. This method relies on the marker data slot being assigned in the StackedData object.

Usage

```
computeMarginalData(object, byVarNames, idVarName, percentVarName, groupVarName)
```

Arguments

object	an object of the StackedData class
byVarNames	character; the names of the variables specifying the subsets of interest in the data
idVarName	character; the name of the id variable in the data
percentVarName	character; the name of the variable holding the percentages to be summed when computing the pfd summary data
groupVarName	character; the name of the variable specifying the group assignment in the data

Value

data frame of marginal data

Methods

signature(object = "StackedData", byVarNames = "character", idVarName = "character", percentVarName = "ch
Compute the marginal data.

Author(s)

N. Hawkins, Fred Hutchinson Cancer Research Center, Seattle, WA

See Also

StackedData, marginalData

Examples

```
# Load stacked data
data(adultsNeonates)
# Create a stacked data object
stackedDataObject = new("StackedData", stackedData=adultsNeonates)

# Compute the marker data and set the marker data slot
markerNames = c("TNFa","IL6","IL12","IFNa")
markers = computeMarkers(markerNames,includeAllNegativeRow=TRUE)
markers(stackedDataObject) = markers

# Compute the marginal data and set the marginal data slot
byVarNames = c("stim", "concGroup", "cell")
marginalData = computeMarginalData(stackedDataObject, byVarNames, "id", "percentAll", "group")
marginalData(stackedDataObject) = marginalData
```

---

computeMarkers-methods

Method computeMarkers from Class "StackedData"

Description

This function is a method of the StackedData class which computes the markers which can be stored in the markers data slot of a StackedData object. The marker matrix should match the order of the rows in the stacked data file. The stacked data should be sorted so that the order of each 'stack' in the file is the same. This method can be used to compute a marker matrix. If that matrix does not match the order of the 'stack', then the user can generate the marker matrix separately and assign it to the marker the data slot in a StackedData object.

Usage

```
computeMarkers(markerNames, includeAllNegativeRow)
```

Arguments

markerNames	character; vector of the names of the markers
includeAllNegativeRow	logical; TRUE, if the stacked data contains the all-negative row of markers; for example, TNFa-IL6-IL12-IFNa-

Value

matrix of 0's and 1's; rows represent marker combinations, cols represent markers.

Methods

`signature(markerNames = "character", includeAllNegativeRow = "logical")` Compute the markers. Include the all negative row if the data includes the all negative case, such as: TNFa-IFNg-IL2-, in this case with 3 markers.

Author(s)

N. Hawkins, Fred Hutchinson Cancer Research Center, Seattle, WA

See Also

StackedData, markers

Examples

```
# Compute the marker data
markerNames = c("TNFa","IL6","IL12","IFNa")
markers = computeMarkers(markerNames,includeAllNegativeRow=TRUE)

## If you're using a StackedData object to compute summary data

# Create a stacked data object
stackedDataObject = new("StackedData")

# Assign the markers to the marker data slot
markers(stackedDataObject) = markers
```

`computePFDData-methods`

Method computePFDData from Class "StackedData"

Description

This function is a method of the StackedData class which computes the polyfunctional degree (pfd) data (usually recorded as percentages of reactive cells) which can be stored in the pfdData slot of a StackedData object. PFD=1 refers to cells which are producing only one marker. PFD=2 refers to cells which are producing exactly two markers. Similarly, up to PFD=n, where n is the number of markers. This method relies on the marker data slot being assigned in the StackedData object.

Usage

```
computePFDData(object, byVarNames, idVarName, percentVarName, groupVarName)
```

Arguments

object	an object of the StackedData class
byVarNames	character; the names of the variables specifying the subsets of interest in the data
idVarName	character; the name of the id variable in the data
percentVarName	character; the name of the variable holding the percentages to be summed when computing the pfd summary data
groupVarName	character; the name of the variable specifying the group assignment in the data

Value

data frame of pfd data

Methods

signature(object = "StackedData", byVarNames = "character", idVarName = "character", percentVarName = "ch
: Computes the pfd data.

Author(s)

N. Hawkins, Fred Hutchinson Cancer Research Center, Seattle, WA

See Also

StackedData, pfdData

Examples

```
# Load stacked data
data(adultsNeonates)
# Create a stacked data object
stackedDataObject = new("StackedData", stackedData=adultsNeonates)

# Compute the marker data and set the marker data slot
markerNames = c("TNFa","IL6","IL12","IFNa")
markers = computeMarkers(markerNames,includeAllNegativeRow=TRUE)
markers(stackedDataObject) = markers

# Compute the pfd data and set the pfd data slot
byVarNames = c("stim", "concGroup", "cell")
pfdData = computePFDData(stackedDataObject, byVarNames, "id", "percentAll", "group")
pfdData(stackedDataObject) = pfdData
```

computePFDGroupStatsList *Compute Group Stats on PFD Data to Be Used In a Legend*

Description

This function can be used with ternaryplot() to add PFD group stats to, say, the legend. The stats computed are group size (N), pfd group mean, and pfd group standard deviation.

Usage

```
computePFDGroupStatsList(groupPFDDataList, pfdValues=1:3, numDigitsMean=3, numDigitsSD=2)
```

Arguments

groupPFDDataList	one list item per group, each list item contains a matrix of PFD percentages; the rows are subjects, and the columns are pfd categories.
pfdValues	vector of the PFD values that the columns in each matrix in the groupPFDDataList represent; eg. 1:3 for (PFD1,PFD2,PFD3).
numDigitsMean	return a mean rounded to this number of digits
numDigitsSD	return a standard deviation rounded to this number of digits

Value

a list; each list item contains the stats for a group as a 3 element character vector containing the size of the group, the mean PFD, and the standard deviation of the PFD.

Author(s)

N. Hawkins, Fred Hutchinson Cancer Research Center, Seattle, WA

See Also

StackedData, pfdData

Examples

```
## Load PFD data to plot
data(pfdDF)
pfdDataSubset = subset(pfdDF, stim=="LPS" & concGroup==3 & cell=="mDC")

## Prepare the PFD Data for a call to ternaryplot()
ternaryData = makeTernaryData(pfdDataSubset, 1, 2, 3:4)
colnames(ternaryData) = c("PFD1", "PFD2", "PFD3-4")

## Make a ternary plot
library(vcd)
```

```
ternaryplot(ternaryData, cex=.5, col=as.numeric(pfdDataSubset$group)*2, main="Stimulation = LPS,
   Concentration Group = 3, Cell = mDC")

## Compute Group Stats to use in the legend of the ternary plot
adultPFDData = subset(pfdDataSubset, group=="adult", select=c(PFD1:PFD3))
neoPFDData = subset(pfdDataSubset, group=="neonate", select=c(PFD1:PFD3))
groupPFDDataList = list(adultPFDData, neoPFDData)

## Specifically, compute the PFD Group Stats List
pfdGroupStatsList = computePFDGroupStatsList(groupPFDDataList, pfdValues=1:3, numDigitsMean=3,
   numDigitsSD=2)
groupNames = c("Adults","Neonates")

## Create group names for the legend based on the PFD Group Stats List
legendNames = legendPFDStatsGroupNames(pfdGroupStatsList,groupNames)
grid_legend(0.8, 0.7, pch=c(20,20), col=c(2,4), legendNames, title = "Group (n), mean/sd:",
   gp=gpar(cex=.8))
```

`computePFDPartsData-methods`

Method computePFDPartsData from Class "StackedData"

Description

This function is a method of the StackedData class which computes the pfdParts data which can be stored in the pfdPartsData slot of a StackedData object. This method relies on the marker data slot being assigned in the StackedData object.

Usage

```
computePFDPartsData(object, byVarNames, idVarName, percentVarName, groupVarName)
```

Arguments

`object`	an object of the StackedData class
`byVarNames`	character; the names of the variables specifying the subsets of interest in the data
`idVarName`	character; the name of the id variable in the data
`percentVarName`	character; the name of the variable holding the percentages to be summed when computing the pfd summary data
`groupVarName`	character; the name of the variable specifying the group assignment in the data

Value

a list. Each element of the list is a data frame containing the component percents for a given degree of polyfunctionality (PFD), except for the max PFD since there is only one possible combination for the max PFD.

Methods

signature(object = "StackedData", byVarNames = "character", idVarName = "character", percentVarName = "ch
Compute the pfd parts data.

Author(s)

N. Hawkins, Fred Hutchinson Cancer Research Center, Seattle, WA

See Also

StackedData, pfdPartsData

Examples

```
# Load stacked data
data(adultsNeonates)
# Create a stacked data object
stackedDataObject = new("StackedData", stackedData=adultsNeonates)

# Compute the marker data and set the marker data slot
markerNames = c("TNFa","IL6","IL12","IFNa")
markers = computeMarkers(markerNames,includeAllNegativeRow=TRUE)
markers(stackedDataObject) = markers

# Compute the pfd parts data and set the pfd parts data slot
byVarNames = c("stim", "concGroup", "cell")
pfdPartsData = computePFDPartsData(stackedDataObject, byVarNames, "id", "percentAll", "group")
pfdPartsData(stackedDataObject) = pfdPartsData
```

---

computeProfileData-methods

Method computeProfileData from Class "StackedData"

Description

This function is a method of the StackedData class which computes the profile data which can be stored in the profileData slot of a StackedData object. This method relies on the marker data slot being assigned in the StackedData object. No computation is required. Instead, the marker combination percentages data is re-organized to be 'horizontal' rather than 'stacked' (vertical). This makes it ready for plotting via the GroupListBoxplot() function.

Usage

```
computeProfileData(object, byVarNames, idVarName, percentVarName, groupVarName)
```

Arguments

object	an object of the StackedData class
byVarNames	character; the names of the variables specifying the subsets of interest in the data
idVarName	character; the name of the id variable in the data
percentVarName	character; the name of the variable holding the percentages to be summed when computing the pfd summary data
groupVarName	character; the name of the variable specifying the group assignment in the data

Value

data frame of profile data

Methods

signature(object = "StackedData", byVarNames = "character", idVarName = "character", percentVarName = "ch
: Compute the profile data.

Author(s)

N. Hawkins, Fred Hutchinson Cancer Research Center, Seattle, WA

See Also

StackedData, profileData

Examples

```
# Load stacked data
data(adultsNeonates)
# Create a stacked data object
stackedDataObject = new("StackedData", stackedData=adultsNeonates)

# Compute the marker data and set the marker data slot
markerNames = c("TNFa","IL6","IL12","IFNa")
markers = computeMarkers(markerNames,includeAllNegativeRow=TRUE)
markers(stackedDataObject) = markers

# Compute the profile percent data and set the profile percent data slot
byVarNames = c("stim", "concGroup", "cell")
profilePercent = computeProfileData(stackedDataObject, byVarNames, "id", "percentAll", "group")
profileData(stackedDataObject) = profilePercent
```

GroupListBoxplot *A Boxplot Function With Embedded Statistical Tests for Comparing Groups*

Description

A function which creates boxplots side-by-side, with points overlaid, to compare groups. Group sizes and p-values from tests comparing groups can be printed on the plot.

Usage

```
GroupListBoxplot(dataList, ymaxBoxplot=NA,addToYmax2=.2,addToYmax1=.1,boxWidth=.10,boxColor=8,
 boxOutliers=TRUE,groupColorVector=c(2,4,5,6,7,9,3,10,11,8,1),boxlty=1,boxlwd=1,
   medlty=1,medlwd=3,medpch=NA, medcex=NA, legendInclude=TRUE,
 legendGroupNames=paste("Group ", 1:length(dataList),sep=""),legendX=NA,legendY=NA,legendCEX=1,
 legendPCH=1,legendColors=1:length(legendGroupNames),legendLTY=NA,legendLWD=NA,legendTitle=NA,
  legendPoints=TRUE,legendLines=FALSE,printPoints=TRUE,pointChar=1,pointCEX=1,
 pointColor=1:length(legendGroupNames),pointJitter=.25,mainTitle="Boxplots", mainTitleCEX=1,
 mainTitleFont=1,mainTitleLine=1,testTitleCEX=1,testTitleFont=1,testTitleLine=0,xlabel="X Axis",
 ylabel="Y Axis",xylabelsCEX=1,xylabelsFont=1,xAxisLabels=NA,xAxisCEX=1, xAxisFont=1,
 xAxisRotation=1,xMtext="",xMtextCEX=1,xMtextFont=1,xAtMtext=0,yAxisCEX=1,yAxisFont=1,
 yAxisRotation=1,plotBoxLWD=1,testsRoundDigits=2,pCEX=1,pFont=1,yP=NA,pvalueLabel="p",
 betweenGroupTestsCompute=TRUE,pairedGroups=FALSE,printNs=TRUE,nCEX=1,nFont=1,yN=NA)
```

Arguments

dataList	A list of data frames. Each data frame contains the data for a group. Each column in the data frame represents a time point or category point on the plot.
ymaxBoxplot	ymax for the boxplot will be set by R's boxplot() fcn, unless set here, default=NA
addToYmax2	this value is added to ymax if printing p-vals and N's, default=.2
addToYmax1	this value is added to ymax if only printing p-vals or N's, default=.1
boxWidth	width of the box, default=.10
boxColor	color of the box, default=8
boxOutliers	print or suppress outlier points on the plot, default=TRUE
groupColorVector	Vector of unique colors for a set of up to 11 groups, default=c(2,4,5,6,7,9,3,10,11,8,1)
boxlty	Box outline type, default=1
boxlwd	Box outline width, default=1
medlty	median line type, default=1
medlwd	median line width, default=3
medpch	median point character, default=NA
medcex	median point size expansion, default=NA

legendInclude	Print legend on plot, default=TRUE
legendGroupNames	names to use in the legend items, default=paste("Group ", 1:length(dataList), sep="")
legendX	legend X location, default=NA
legendY	legend Y location, default=NA
legendCEX	point size for legend text, default=1
legendPCH	symbol or character to print next to legend names, default=1
legendColors	colors of the points or lines next to the legend names, default=1:length(legendGroupNames)
legendLTY	line type to print next to legend names, default=NA
legendLWD	width of line to print next to legend names, default=NA
legendTitle	legend title, default=NA
legendPoints	print points next to legend names, default=TRUE
legendLines	print lines next to legend names, default=FALSE
printPoints	overlay the points on the boxplot, default=TRUE
pointChar	symbol or character to plot, default=1
pointCEX	size of point plotted, default=1
pointColor	color of point plotted – can be single value, vector, matrix, or list, default=1:length(legendGroupNames)
pointJitter	amount to scatter points around group x position, default=.25
mainTitle	text for main title, default="Boxplots"
mainTitleCEX	size of main title, default=1
mainTitleFont	regular or bold font, default=1
mainTitleLine	margin line on which to print main title - minimum is 0, default=1
testTitleCEX	size of stat test title, default=1
testTitleFont	regular or bold font, default=1
testTitleLine	margin line on which to print title - minimum is 0, default=0
xlabel	label for X axis, default="X Axis"
ylabel	label for Y axis, default="Y Axis"
xylabelsCEX	size of x and y axes labels, default=1
xylabelsFont	regular or bold font for X and Y axes labels, default=1
xAxisLabels	labels for ticks on x axis, default=NA
xAxisCEX	size of labels on x ticks, default=1
xAxisFont	regular or bold font for x tick labels, default=1
xAxisRotation	horizontal or vertical x tick labels, default=1
xMtext	text to place in x-axis margin, default=""
xMtextCEX	size of text to place in x-axis margin, default=1
xMtextFont	regular or bold font for text in x-axis margin, default=1
xAtMtext	x position for text in x-axis margin, size of y tick labels, default=0

yAxisCEX	size of y tick labels, default=1
yAxisFont	regular or bold font for y axis tick labels, default=1
yAxisRotation	horizontal or vertical y tick labels, default=1
plotBoxLWD	line width of box drawn around entire plot, default=1
testsRoundDigits	number of digits to report in p-value, default=2
pCEX	size of p-value text, default=1
pFont	regular or bold font for p-value text, default=1
yP	y position for p-value text, default=NA
pvalueLabel	label to use to precede numerical p-value, default="p"
betweenGroupTestsCompute	include group comparison p-values in plots, default=TRUE
pairedGroups	are the groups paired?, default=FALSE
printNs	include sample sizes on plots, default=TRUE
nCEX	size of the sample size text to print, default=1
nFont	regular or bold text for n's, default=1
yN	y position for the sample size text, default=NA

Author(s)

N. Hawkins, Fred Hutchinson Cancer Research Center, Seattle, WA

See Also

boxplot

Examples

```
# Create Sample dataList
group1DataFrame = as.data.frame(cbind(1:3,4:6))
group2DataFrame = as.data.frame(cbind(4:6,7:9))
dataList = list(group1DataFrame, group2DataFrame)

# Make the plot
GroupListBoxplot(dataList, xlabel="Cytokine", ylabel="Percent of CD4 Cells",
   xAxisLabels=c("IFNg","TNFa"), mainTitle="Compare Innate Immune Response",
   legendGroupNames=c("Group 1","Group 2"))

## -- Adults vs. Neonates Data ----------------------------------

## Marginal Data boxplot

# Get the data
data(marginalDF)
marginalDataSubset = subset(marginalDF, stim=="LPS" & concGroup==3 & cell=="mDC")
dataList = makeDataList(marginalDataSubset, "group", 1:5)
```

```
# Make the group boxplot of marginal data
GroupListBoxplot(dataList, xlabel="Cytokine", ylabel="Percent of All Cells",
   xAxisLabels=c("TNFa","IL6","IL12","IFNa","AnyMarker"),
   mainTitle="Stimulation = LPS and Concentration Group = 3 and Cell = mDC",
   legendGroupNames=c("Adults","Neonates"), pointColor=c(2,4), testTitleCEX=.8, nCEX=.8,
   pCEX=.8, legendColor=c(2,4), legendCEX=.7)
```

`legendPFDStatsGroupNames` *Create Group Names With Embedded Stats to Use in a Plot Legend*

Description

This function returns a vector of groupNames of the form: "Adults (25) 1.5/.6", which represents the group name, number of subjects in the group, the pfd mean / pfd standard deviation, where pfd = polyfunctional degree.

Usage

```
legendPFDStatsGroupNames(pfdGroupStatsList,groupNames)
```

Arguments

`pfdGroupStatsList`	a list of vectors containing the pfd group stats of group size, pfd group mean, and pfd group standard deviation
`groupNames`	a vector of group names, such as c("Adult", "Neonate")

Value

character vector; the length is equal to the number of groups. Each element of the vector contains the formatted stats for a group. For a subset of the adultsNeonates data, the 2-element vector result looks like this: "Adults (24), 1.47/0.10" "Neonates (30), 1.41/0.13"

Author(s)

N. Hawkins, Fred Hutchinson Cancer Research Center, Seattle, WA

See Also

`StackedData`

Examples

```
## Load the PFD data and prepare it for a call to ternaryplot()

data(pfdDF)
pfdDataSubset = subset(pfdDF, stim=="LPS" & concGroup==3 & cell=="mDC")
ternaryData = makeTernaryData(pfdDataSubset, 1, 2, 3:4)
colnames(ternaryData) = c("PFD1", "PFD2", "PFD3-4")

## Make a ternary plot

library(vcd)
ternaryplot(ternaryData, cex=.5, col=as.numeric(pfdDataSubset$group)*2, main="Stimulation = LPS,
   Concentration Group = 3, Cell = mDC")

## Compute PFD Stats to print in plot legend

adultPFDData = subset(pfdDataSubset, group=="adult", select=c(PFD1:PFD3))
neoPFDData = subset(pfdDataSubset, group=="neonate", select=c(PFD1:PFD3))
groupPFDDataList = list(adultPFDData, neoPFDData)
pfdGroupStatsList = computePFDGroupStatsList(groupPFDDataList, pfdValues=1:3, numDigitsMean=3,
   numDigitsSD=2)
groupNames = c("Adults","Neonates")

## Create group names including the PFD Stats for the legend

legendNames = legendPFDStatsGroupNames(pfdGroupStatsList,groupNames)


## Add the legend to the ternary plot

grid_legend(0.8, 0.7, pch=c(20,20), col=c(2,4), legendNames, title = "Group (n), mean/sd:",
   gp=gpar(cex=.8))
```

---

makeBarplotData *Prepare Profile Data for a Call to Barplot()*

Description

This function takes a dataframe of profile data and prepares a matrix of data for input to barplot()

Usage

```
makeBarplotData(profileData, profileColumns, groupVariableName)
```

Arguments

`profileData`	dataframe of profile data, such as cell categories of cells
`profileColumns`	the columns of profileData to include in the barplot
`groupVariableName`	the column in the dataframe containing the group info

Value

a matrix whose rows represent different profile categories and whose columns represent different groups. Each cell in the matrix contains the mean value for the group for a given profile category. For profile data with 16 cytokine combinations for each of 2 groups, the matrix returned with have dimensions (16,2).

Author(s)

N. Hawkins, Fred Hutchinson Cancer Research Center, Seattle, WA

See Also

GroupListBoxplot

Examples

```
data(profileDF)
profileDataSubset = subset(profileDF, stim=="LPS" & concGroup==3 & cell=="mDC")
profileColumns = 1:16
barplotData = makeBarplotData(profileDataSubset, profileColumns, groupVariableName="group")
barplotDataWithLegend = cbind(barplotData, NA, NA)
barColors = gray(0:15/15)[16:1]
barplot(barplotDataWithLegend, col=barColors, main="Stimulation = LPS
   Concentration Group = 3
   Cell = mDC")
legendNames = rownames(barplotData)
legend(2.75, 100, legend=legendNames[16:1], col=barColors[16:1], cex=.8, pch=20)
```

---

makeDataList *Prepare Data in a Data Frame for a Call to GroupListBoxplot()*

Description

This function makes a list of data, where each item of the list is a data frame and contains data for a group. The list can be used as input for a call to GroupListBoxplot().

Usage

```
makeDataList(theData, groupVariableName, columnsToKeep)
```

Arguments

theData	data.frame containing data for 1 or more groups
groupVariableName	character; name of the column in theData identifying group
columnsToKeep	numeric vector specifying the column numbers in theData to include in the list of data.

Value

a list; each item of the list is a data frame and contains data for a group. The rows of each data frame are for subject. The cols of each data frame represent categories to be plotted on the x-axis.

Author(s)

N. Hawkins, Fred Hutchinson Cancer Research Center, Seattle, WA

See Also

GroupListBoxplot

Examples

```
# Load the data
data(marginalDF)
marginalDataSubset = subset(marginalDF, stim=="LPS" & concGroup==3 & cell=="mDC")

# Make a data list
dataList = makeDataList(marginalDataSubset, "group", 1:5)

# Make a plot using the data list
GroupListBoxplot(dataList, xlabel="Cytokine", ylabel="Percent of All Cells",
   xAxisLabels=c("TNFa","IL6","IL12","IFNa","AnyMarker"),
   mainTitle="Stimulation = LPS and Concentration Group = 3 and Cell = mDC",
   legendGroupNames=c("Adults","Neonates"), pointColor=c(2,4), testTitleCEX=.8, nCEX=.8,
   pCEX=.8, legendColor=c(2,4), legendCEX=.7)
```

---

makeTernaryData *Prepare PFD Data for a Call to ternaryplot()*

Description

This function takes a dataframe of polyfunctional degree (pfd) data and prepares a matrix with 3 columns (egs. PFD1,PFD2,PFD3, or PFD1-2, PFD3-4, PFD5-6) to use as input to ternaryplot() in the vcd pkg. If multiple columns are specified for one of the three columns, the percentages for the multiple columns will be summed.

Usage

```
makeTernaryData(pfdData, columns1, columns2, columns3,
   columnNames=c("PFD=1","PFD=2","PFD=3"))
```

Arguments

pfdData	data frame containing data for 1 or more groups
columns1	numeric; column(s) of pfd data to place in the first column of the matrix
columns2	numeric; column(s) of pfd data to place in the second column of the matrix
columns3	numeric; column(s) of pfd data to place in the third column of the matrix
columnNames	(optional) character vector of names for the three columns of the matrix

Value

matrix of 3 columns; rows = subjects, cols = data for each point on the triangle

Author(s)

N. Hawkins, Fred Hutchinson Cancer Research Center, Seattle, WA

See Also

StackedData

Examples

```
## Load the PFD Data to plot in a ternary plot

data(pfdDF)
pfdDataSubset = subset(pfdDF, stim=="LPS" & concGroup==3 & cell=="mDC")

## Prepare the PFD Data for a call to ternaryplot()

ternaryData = makeTernaryData(pfdDataSubset, 1, 2, 3:4)
colnames(ternaryData) = c("PFD1", "PFD2", "PFD3-4")

## Make the ternary plot

library(vcd)
ternaryplot(ternaryData, cex=.5, col=as.numeric(pfdDataSubset$group)*2, main="Stimulation = LPS,
   Concentration Group = 3, Cell = mDC")

## Prepare a legend with group stats

adultPFDData = subset(pfdDataSubset, group=="adult", select=c(PFD1:PFD3))
neoPFDData = subset(pfdDataSubset, group=="neonate", select=c(PFD1:PFD3))
groupPFDDataList = list(adultPFDData, neoPFDData)
pfdGroupStatsList = computePFDGroupStatsList(groupPFDDataList, pfdValues=1:3, numDigitsMean=3,
   numDigitsSD=2)
groupNames = c("Adults","Neonates")
legendNames = legendPFDStatsGroupNames(pfdGroupStatsList,groupNames)
grid_legend(0.8, 0.7, pch=c(20,20), col=c(2,4), legendNames, title = "Group (n), mean/sd:",
   gp=gpar(cex=.8))
```

`marginalData-methods` *Method marginalData from Class "StackedData"*

Description

This function is a method of the StackedData class which can retrieve the marginalData from a StackedData object or which can assign the marginalData data slot of a StackedData object.

Usage

```
  # Get the marginal data from a StackedData object
   marginalData(object)

  # Set the marginal data slot of a StackedData object
  ## S4 replacement method for signature 'StackedData'
marginalData(object) <- value
```

Arguments

`object`	an object of the StackedData class
`value`	a replacement value

Value

data frame of marginal data.

Methods

`signature(object="StackedData")` Get the value of the marginalData slot in the stackedDataObject.

`signature(object = "StackedData", value = "data.frame")` Set the value of the marginalData slot in the stackedDataObject.

Author(s)

N. Hawkins, Fred Hutchinson Cancer Research Center, Seattle, WA

See Also

`StackedData`, `marginalData`

Examples

```
# Load the marginal data and set the marginal data slot
data(marginalDF)
# Create a stacked data object
stackedDataObject = new("StackedData")
# Set the marginal data slot
marginalData(stackedDataObject) = marginalDF

# Get the marginal data from the stacked data object
marginalData = marginalData(stackedDataObject)
```

---

marginalDF *An example of marginalData*

Description

The marginal data computed from the adultsNeonates data.

Usage

```
data(marginalDF)
```

Value

data frame; There are (n+1) columns in the data frame for the marginal percentages; one col for each marker and 1 col for "anyMarker", which is the sum of the individual marker cols. The data frame also has cols of 'demographic' data describing each row of percentages.

Author(s)

N. Hawkins, Fred Hutchinson Cancer Research Center, Seattle, WA

See Also

StackedData

Examples

```
# Load marginal data
data(marginalDF)
```

markerMatrix *An example of the markers data.*

Description

The markers data created to match the order of the marker combinations in each subset of interest in the adultsNeonates data.

Usage

```
data(markerMatrix)
```

Value

matrix of 0's and 1's; rows = marker combinations, cols = marker names.

Author(s)

N. Hawkins, Fred Hutchinson Cancer Research Center, Seattle, WA

See Also

StackedData

Examples

```
# Load marker data
data(markerMatrix)
```

markers-methods *Method markers from Class "StackedData"*

Description

This function is a method of the StackedData class which retrieves the markers from a StackedData object or which assigns the markers data slot of a StackedData object.

Usage

```
   # Get the marker matrix from a StackedData object.
   markers(object)

   # Set the marker matrix slot of a StackedData object.
   ## S4 replacement method for signature 'StackedData'
markers(object) <- value
```

Arguments

`object`	an object of the StackedData class
`value`	a replacement value

Value

matrix of markers data.

Methods

`signature(object = "StackedData")` Get the marker matrix from the object.

`signature(object = "StackedData", value = "matrix")` Set the value of the markers data slot in the object.

Author(s)

N. Hawkins, Fred Hutchinson Cancer Research Center, Seattle, WA

See Also

`StackedData`, `markers`

Examples

```
# Load the marker data and set the marker data slot
data(markerMatrix)

# Create a stacked data object
stackedDataObject = new("StackedData")

# Set the marker data slot
markers(stackedDataObject) = markerMatrix

# Get the marker data from the stacked data object
markers = markers(stackedDataObject)
```

`pfdData-methods` *Method pfdData from Class "StackedData"*

Description

This function is a method of the StackedData class which retrieves the pfdData from a StackedData object or which assigns the pfdData data slot of a StackedData object.

Usage

```
    # Get the pfdData from a StackedData object.
    pfdData(object)

    # Set the pfdData slot of a StackedData object.
    ## S4 replacement method for signature 'StackedData'
pfdData(object) <- value
```

Arguments

`object`	an object of the StackedData class
`value`	a replacement value

Value

data frame of pfd data.

Methods

`signature(object = "StackedData")` Get the pfdData from the object.

`signature(object = "StackedData", value = "data.frame")` Set the value of the pfdData slot in the object.

Author(s)

N. Hawkins, Fred Hutchinson Cancer Research Center, Seattle, WA

See Also

StackedData, pfdData

Examples

```
# Load the pfd data and set the pfd data slot
data(pfdDF)

# Create a stacked data object
stackedDataObject = new("StackedData")

# Set the pfd data slot
pfdData(stackedDataObject) = pfdDF

# Get the pfd data from the stacked data object
pfdData = pfdData(stackedDataObject)
```

pfdDF *An example of pfdData*

Description

The polyfunctional degree (pfd) data computed from the adultsNeonates data.

Usage

```
data(pfdDF)
```

Value

data frame; The data frame contains n columns of PFD percentages of reactive cells, where n is the number of markers in the data set, as well as 'demographic' data for each row of percentages. PFD=1 refers to the percentages of reactive cells which are producing only one marker, irregardless of which marker. PFD=2 refers to cells which are producing exactly two markers. Similarly, up to PFD=n markers.

Author(s)

N. Hawkins, Fred Hutchinson Cancer Research Center, Seattle, WA

See Also

StackedData

Examples

```
# Load pfd data
data(pfdDF)
```

pfdPartsData-methods *Method pfdPartsData from Class "StackedData"*

Description

This function is a method of the StackedData class which retrieves the pfdPartsData from a StackedData object or which assigns the pfdPartsData data slot of a StackedData object.

Usage

```
    # Get the pfdPartsData from a Stacked Data object.
    pfdPartsData(object)

    # Set the pfdPartsData of a Stacked Data object.
    ## S4 replacement method for signature 'StackedData'
pfdPartsData(object) <- value
```

Arguments

`object`	an object of the StackedData class
`value`	a replacement value

Value

a list. Each element of the list is a data frame containing the component percents for a given degree of polyfunctionality (PFD), except for the max PFD since there is only one possible combination for the max PFD.

Methods

`signature(object = "StackedData")` Get the pfdPartsData from the object.

`signature(object = "StackedData", value = "list")` Set the value of the pfdPartsData slot in the object.

Author(s)

N. Hawkins, Fred Hutchinson Cancer Research Center, Seattle, WA

See Also

StackedData, pfdPartsData

Examples

```
# Load the pfdParts data and set the pfdParts data slot
data(pfdPartsList)

# Create a stacked data object
stackedDataObject = new("StackedData")

# Set the pfdParts data slot
pfdPartsData(stackedDataObject) = pfdPartsList

# Get the pfd parts data
pfdPartsData = pfdPartsData(stackedDataObject)
```

pfdPartsList *An example of pfdPartsData*

Description

The pfd parts data computed from the adultsNeonates data.

Usage

```
data(pfdPartsList)
```

Value

a list; pdfPartsData is a list of data frames. The first data frame holds the compositional percentages for PFD=1; that is, of the cells producing only one marker, the percentage of cells which express marker1, the percentage of cells which produce marker2, etc. For example, if there are 4 markers, the data frame for PFD=1 will have a percentage column for each marker, and the sum of those 4 cols will equal 100 The second data frame has n-choose-2 columns of percentages, where n is the number of markers and 2 is the polyfunctional degree; i.e. the 2 in PFD=2. And, so on. The length of the list is (maxPFD-1), since there is only 1 way to achieve maxPFD; that is, all markers are positive when PFD=maxPFD. Each data frame also contains 'demographic' data describing each row of percentages.

Author(s)

N. Hawkins, Fred Hutchinson Cancer Research Center, Seattle, WA

See Also

StackedData

Examples

```
# Load pfd parts data
data(pfdPartsList)
```

profileData-methods *Method profileData from Class "StackedData"*

Description

This function is a method of the StackedData class which retrieves the profileData from a StackedData object or which assigns the profileData data slot of a StackedData object.

Usage

```
    # Get the profileData from a StackedData object.
    profileData(object)

    # Set the profileData slot of a StackedData object.
    ## S4 replacement method for signature 'StackedData'
profileData(object) <- value
```

Arguments

object	an object of the StackedData class
value	a replacement value

Value

data frame of profile data.

Methods

`signature(object = "StackedData")` Get the profileData from the object.

`signature(object = "StackedData", value = "data.frame")` Set the value of the profileData slot in the object.

Author(s)

N. Hawkins, Fred Hutchinson Cancer Research Center, Seattle, WA

See Also

StackedData, profileData

Examples

```
# Load the profile data, create a stackedData object and set the profile data slot
data(profileDF)
# Create a stacked data object
stackedDataObject = new("StackedData")
# Set the profile data slot
profileData(stackedDataObject) = profileDF

# Get the profile data from the stacked data object
profileData = profileData(stackedDataObject)
```

profileDF *An example of profileData*

Description

The profile data computed from the adultsNeonates data.

Usage

```
data(profileDF)
```

Value

a data frame containing cols for the percentages for each of the marker combinations (the profile percentages), as well as 'demographic' data describing each row of percentages. For example, if there are 4 markers in the data set, there will be 2^4 columns for the profile percentages if all possible combinations are included. If the all-negative combination is excluded, there will be (2^4-1) cols of profile percentages.

Author(s)

N. Hawkins, Fred Hutchinson Cancer Research Center, Seattle, WA

See Also

StackedData

Examples

```
# Load profile data
data(profileDF)
```

readStackedData-methods *Method readStackedData from Class "StackedData"*

Description

This function is a method of the StackedData class which reads a csv file of stacked data into a data frame, which can be stored in the stackedData data slot of a StackedData object. This function is a wrapper for the R base function, read.csv(), for users not so familiar with R.

Usage

```
readStackedData(fileName)
```

Arguments

`fileName` character; the full name of the file containing stacked data in csv format.

Value

data frame of stacked data.

Methods

`signature(fileName = "character")` Wrapper function for read.csv() to read a file of stacked data.

Author(s)

N. Hawkins, Fred Hutchinson Cancer Research Center, Seattle, WA

See Also

`StackedData`, `read.csv`

Examples

```
stackedDataFrame = readStackedData(fileName=system.file("extdata","adultsNeonates.csv", package="flowPlots"))
```

StackedData-class *Class "StackedData"*

Description

Stacked data refers to gated data originating from an ICS Flow Cytometry experiment where the marker combinations for a subset of interest, say a given cell type, stimulus, and concentration, are "stacked". A common type of marker is a cytokine. A subset of stacked data could look like this:

id	group	stim	concGroup	cell	percentAll	count	totalCount	percentReactive	cytCombo
a2004	adult	LPS	3	mDC	0.00	0	700	0.000000	TNFa+IL6+IL12+IFNa+
a2004	adult	LPS	3	mDC	0.43	3	700	0.940625	TNFa+IL6+IL12+IFNa-
a2004	adult	LPS	3	mDC	0.00	0	700	0.000000	TNFa+IL6+IL12-IFNa+
a2004	adult	LPS	3	mDC	21.86	153	700	47.818750	TNFa+IL6+IL12-IFNa-
a2004	adult	LPS	3	mDC	0.00	0	700	0.000000	TNFa+IL6-IL12+IFNa+
a2004	adult	LPS	3	mDC	0.29	2	700	0.634375	TNFa+IL6-IL12+IFNa-

```
a2004  adult  LPS      3  mDC     0.00    0      700      0.000000 TNFa+IL6-IL12-IFNa+

a2004  adult  LPS      3  mDC    19.71   138      700     43.115625 TNFa+IL6-IL12-IFNa-
```

Details

The marker combinations in the stacked data should be ordered within each subset of interest (for example: subjectID, celltype, concentration, and stimulation) to match the marker matrix. If the data are not in this order, it should be sorted into this order before using the computeProfileData, computeMarginalData, computePFDData, computePFDPartsData methods. The computeMarkers method can be used to compute the marker matrix used by these methods. If the matrix computed does not match the order of your data, then you can supply your own marker matrix, assign it to the marker data slot of a StackedData object, and then use the 'compute' methods to compute the other types of data.

Objects from the Class

Objects can be created by calls of the form:

`stackedDataObject = new("StackedData", stackedData=NA, profileData=NA, marginalData=NA, pfdData=NA, pfdPartsData=NA, markers=NA )`

Slots

`stackedData`**:** `"data.frame"` of stacked data

`profileData`**:** `"data.frame"` of profile data

`marginalData`**:** `"data.frame"` of marginal data

`pfdData`**:** `"data.frame"` of pfd data

`pfdPartsData`**:** `"list"` of `"data.frame"` 's of pfd parts data

`markers`**:** `"matrix"` of marker data

Methods

computeMarginalData `signature(object = "StackedData", byVarNames = "character", idVarName = "character", percentVarName = "character", groupVarName = "character")`

computeMarkers `signature(markerNames = "character", includeAllNegativeRow = "logical")`

computePFDData `signature(object = "StackedData", byVarNames = "character", idVarName = "character", percentVarName = "character", groupVarName = "character")`

computePFDPartsData `signature(object = "StackedData", byVarNames = "character", idVarName = "character", percentVarName = "character", groupVarName = "character")`

computeProfileData `signature(object = "StackedData", byVarNames = "character", idVarName = "character", percentVarName = "character", groupVarName = "character")` # getters

marginalData `signature(object = "StackedData")`

markers `signature(object = "StackedData")`

pfdData `signature(object = "StackedData")`

pfdPartsData `signature(object = "StackedData")`

profileData signature(object = "StackedData")

stackedData signature(object = "StackedData") # setters

marginalData signature(object = "StackedData", value = "data.frame")

markers signature(object = "StackedData", value = "matrix")

pfdData signature(object = "StackedData", value = "data.frame")

pfdPartsData signature(object = "StackedData", value = "list")

profileData signature(object = "StackedData", value = "data.frame")

stackedData signature(object = "StackedData", value = "data.frame")

Author(s)

N. Hawkins, Fred Hutchinson Cancer Research Center, Seattle, WA

See Also

markers, profileData, marginalData, pfdData, pfdPartsData, adultsNeonates

Examples

```
# View the Data Slots in StackedData
showClass("StackedData")

# Load stacked data
data(adultsNeonates)
# Create a stacked data object
stackedDataObject = new("StackedData", stackedData=adultsNeonates)

# Compute the marker data and set the marker data slot
# The marker matrix computed here matches the order of the marker combinations
# in the adultsNeonates stacked data within each subset of interest
markerNames = c("TNFa","IL6","IL12","IFNa")
markers = computeMarkers(markerNames,includeAllNegativeRow=TRUE)
markers(stackedDataObject) = markers

# Compute the profile data and set the profile data slot
byVarNames = c("stim", "concGroup", "cell")
profileData = computeProfileData(stackedDataObject, byVarNames, "id", "percentAll", "group")
profileData(stackedDataObject) = profileData

# Compute the marginal data and set the marginal data slot
byVarNames = c("stim", "concGroup", "cell")
marginalData = computeMarginalData(stackedDataObject, byVarNames, "id", "percentAll", "group")
marginalData(stackedDataObject) = marginalData

# Compute the pfd data and set the pfd data slot
byVarNames = c("stim", "concGroup", "cell")
pfdData = computePFDData(stackedDataObject, byVarNames, "id", "percentAll", "group")
pfdData(stackedDataObject) = pfdData

# Compute the pfd parts data and set the pfd parts data slot
```

```
byVarNames = c("stim", "concGroup", "cell")
pfdPartsData = computePFDPartsData(stackedDataObject, byVarNames, "id", "percentAll", "group")
pfdPartsData(stackedDataObject) = pfdPartsData

# Get the data from the stacked data object

markers = markers(stackedDataObject)
profileData = profileData(stackedDataObject)
marginalData = marginalData(stackedDataObject)
pfdData = pfdData(stackedDataObject)
pfdPartsData = pfdPartsData(stackedDataObject)
```

`stackedData-methods` *Method stackedData from Class "StackedData"*

Description

This function is a method of the StackedData class which retrieves the stackedData from a StackedData object or which assigns the stackedData data slot of a StackedData object.

Usage

```
   # Get the stacked data from the StackedData object
   stackedData(object)

   # Set the stacked data slot of a StackedData object
   ## S4 replacement method for signature 'StackedData'
stackedData(object) <- value
```

Arguments

`object`	an object of the StackedData class
`value`	a replacement value

Value

data frame of stacked data.

Methods

`signature(object = "StackedData")` Get the stackedData from the object.

`signature(object = "StackedData", value = "data.frame")` Set the value of the stackedData slot in the object.

Author(s)

N. Hawkins, Fred Hutchinson Cancer Research Center, Seattle, WA

See Also

StackedData

Examples

```
## Set the stacked data slot WHILE creating a new stacked data object

# Load stacked data
data(adultsNeonates)
# Create a stacked data object
stackedDataObject = new("StackedData", stackedData=adultsNeonates)

## Set the stacked data slot AFTER creating a new stacked data object

# Load stacked data
data(adultsNeonates)
# Create a stacked data object
stackedDataObject = new("StackedData")
# Set the stacked data slot
stackedData(stackedDataObject) = adultsNeonates

## Set the stacked data slot after creating a new stacked data object

stackedDataFrame = readStackedData(fileName=system.file("extdata","adultsNeonates.csv", package="flowPlots"))
stackedDataObject = new("StackedData")
stackedData(stackedDataObject) = stackedDataFrame
```

Index