Package 'QUALIFIER'

October 12, 2016

Type Package

Title Quality Control of Gated Flow Cytometry Experiments

Version 1.16.1

Author Mike Jiang,Greg Finak,Raphael Gottardo

Maintainer Mike Jiang <wjiang2@fhcrc.org>

Depends R (>= 2.14.0),flowCore,flowViz,ncdfFlow,flowWorkspace, data.table,reshape

Imports MASS,hwriter,lattice,stats4,flowCore,flowViz,methods,flowWorkspace,latticeExtra,grDevices,tools, Biobase,XML,grid

Description Provides quality control and quality assessment tools for gated flow cytometry data.

License Artistic-2.0

biocViews Infrastructure, FlowCytometry, CellBasedAssays

Suggests RSVGTipsDevice, knitr

VignetteBuilder knitr

NeedsCompilation no

R topics documented:

`QUALIFIER-package` *QUALIFIER: A package that provides automated flow data quality assessment based on gated cell populations*

Description

The package provide two important methods:`qaCheck` and `plot`, which allows users to use `formula` as a general and flexible way to specify the tubes,channels,statistics and gated populations to perform differnt QA tasks.

Details

Package:	QUALIFIER
Version:	0.99.1
Date:	2011-12-02
Depends:	R (>= 2.14), flowCore
License:	Artistic-2.0

Author(s)

Mike Jiang,Greg Finak

Maintainer: Mike Jiang <wjiang2@fhcrc.org>

References

http://www.rglab.org/

`getQAStats` *Get statistics from gated data for QA check.*

Description

`getQAStats` calculates and extract statistics of each gated cell population in the gating hierarchies generated by flowWorkspace package.

Usage

```
getQAStats(obj, ...)
```

Arguments

obj	A gating hierarchy that stores the gated cell populations for one FCS file; or a GatingSet or a GatingSetList containing multiple gating hierarchies or an environment that stores the GatingSet and all other information.
...	other arguments isFlowCore: A logical scalar indicating whether the statistics are the original ones in flowJo xml workspace or the re-calculated version by flowCore in R. isMFI a logical flag indicating whether to calculate MFI which causes the reading of raw data isSpike a logical flag indicating whether to calculate spike for each channel which causes the reading of raw data isRaw logical whether to calculate the MFI value in raw scale. pops a numeric or character vector as the population indices specifing a subset of populations to extract stats from isChannel a logical flag indicating whether to extract channel information from 1d gates in order to perform channel-specific QA nslaves: An integer scalar indicating the number of nodes. It is used for parallel computing when obj is GatingHierarchy or GatingSet. When the parallel package is loaded and nslaves is NULL, its value is automatically decided by available number of nodes. When it is set to 1, then forced to run in serial mode.

Details

This is the second preprocessing step followed by parsing gating template from flowJo workspace with parseWorkspace. Different QA checks can be performed after this step is done. when obj is an environment the results are stored as a dataframe with the name of "statsOfGS" in the environment.

Value

a data frame when obj is a GatingHierarchy or GatingSet

Author(s)

Mike Jiang,Greg Finak

Maintainer: Mike Jiang <wjiang2@fhcrc.org>

See Also

qaCheck,qaTask,qaReport

Examples

```
## Not run:

 getQAStats(G[[1]])#extract stats from a gating hierarchy
 getQAStats(G[[1]], isMFI = TRUE, pops = c("/boundary/lymph/CD3/CD4"))#extract stats from a gating hierarchy
 getQAStats(G)#from a gating set
```

```
## End(Not run)
```

`initDB` *Initializes the data environment*

Description

Initializes and prepares the data environment for storing the QA data

Usage

```
initDB(db = .db)
```

Arguments

db	An `environment` storing all the QA data. By default it is an hidden global environment `.db`.

Examples

```
db <- new.env()
initDB(db)
```

`ITNQASTUDY` *The example QA data included in the package*

Description

Once 'ITNQASTUDY' is loaded by `data` function, an `environment` 'db' is generated, which contains:

- gs a `list` of `GatingSet` objects
- gstbl a `data.frame` that records the 'gsid' and 'gsname' of each object within `gs`
- stats a`data.table` that stores the statistics pre-caluclated from `gs`
- outlierResult a `data.frame` stores the outlier detection results
- GroupOutlierResult a `data.frame` stores the group outlier detection results

Details

`qaCheckList` is a example `spreadsheet` of the definitions of qaTasks

`tubesevents` is an example `spreadsheet` of thresholds for total number of events based on each 'tube'

plot-methods *plot the statistics for a particular cell population of a group of samples*

Description

plot the statistics for a particular cell population of a group of samples,this method is usually called after qaCheck to visualize the QA results.

Usage

```
## S4 method for signature 'qaTask,ANY'
plot(x, y, ...)
```

Arguments

x	a qaTask object
y	a formula describing the variables to be used for plotting. see qaCheck for more details.
...	arguments to control the output. pop:a character scalar indicating the population name.If provided,it overwrites the pop slot in qaTask object. subset:a logical expression as a filter. see qaCheck for more details. width,height:size specification for the svg output. dest: a character specifying the output path. It is NULL by default, which indicates using the regular R device as the output. Otherwise it outputs to a svg file. plotAll: a logical/character scalar indicating whether to plot the 1D/2D density plot for all the individual FCS files together with the summary plot(either xyplot or bwplot). It is only valid when dest is specified as non-null path. It is FALSE by default,indicating that only the FCS files that are marked as outliers by qaCheck are plotted. If TRUE, all FCS files are plotted ,which should be used cautously since it could be time consuming for a large dataset. When it is "none",no scatter plot will be generated. scatterPlot: a logical scalar. When TRUE, the density(scatter) plot is plotted instead of the summary plot(xyplot/bwplot) scatterPar: A list storing all the fliwViz arguments. see xyplot par:A list storing all the lattice arguments.If provided,it overwrites the par slot of qaTask object. outerStrip: a logical indicating whether to enable useOuterStrips strip.lines,strip.left.lines: arguments passed to useOuterStrips

Details

The method does the same thing as qaCheck in terms of parsing the formula and selecting the gated population,statistics and subsetting the samples. The difference is that it reads the outliers detection results saved in database and hightlight them in the summary plots. Two kinds of lattice plots are currently supported:xyplot and bwplot(boxplot),depends on the plotType in qaTask object. When the output path is provided by dest, the svg plot is generated. In svg plot, each dot or box (or only the one marked as outliers) is annotated by the tooltip or hyperlink.which further points to the individual density plot of the gated population.

with scatterPlot and subset arguments, scatter plots can be generated for the selected FCS files or sample groups,which allows users to investigate the individual outlier groups or files.

Author(s)

Mike Jiang,Greg Finak

Maintainer: Mike Jiang <wjiang2@fhcrc.org>

See Also

qaCheck,qaReport

Examples

```
## Not run:

data("ITNQASTUDY")
checkListFile<-file.path(system.file("data",package="QUALIFIER"),"qaCheckList.csv.gz")
qaTask.list<-read.qaTask(db,checkListFile)

#using formula to summing up the percentage of boundary events of each channel
#using the cutoff function to detect the FCS files that has the higher percentage of boundary events
#than the upper threshold provided by uBound
#Note that the percentages of all channels for each fcs file ("name" here indicates the fcs file name)
#are summed up through the formula
qaCheck(qaTask.list[["BoundaryEvents"]]
,sum(proportion) ~ RecdDt | name
,outlierfunc=outlier.cutoff
,uBound=0.0003
)

plot(qaTask.list[["BoundaryEvents"]],proportion ~ RecdDt | channel)



#using Interquartile Range based outlier detection function
#to find the outliers that has significant variance of MNC cell population among aliquots
#here the formula is implicitly provided by qaTask object

qaCheck(qaTask.list[["MNC"]],outlierfunc=qoutlier,alpha=1.5)

plot(qaTask.list[["MNC"]])
```

```
## End(Not run)
```

`proportion.outliers.robust` *outliers detection functions*

Description

Distribution based outlier detection functions.

qoutlier is IQR based outlier detection.

outlier.norm is based on normal distribution using Huber M-estimator of location with MAD scale

outlier.t is based on t-distribution.

outlier.cutoff is a simple cutoff-based outlier detection.

Usage

```
proportion.outliers.robust(x, alpha = 0.01, isUpper = TRUE,
  isLower = TRUE)

proportion.outliers.mle(x, alpha = 0.01, isUpper = TRUE, isLower = TRUE)

qoutlier(x, alpha = 1.5, isUpper = TRUE, isLower = TRUE, plot = FALSE,
  ...)

outlier.norm(x, alpha = 0.01, z.cutoff = NULL, isUpper = TRUE,
  isLower = TRUE, plot = FALSE)

outlier.t(x, alpha = 0.01, z.cutoff = NULL, isUpper = TRUE,
  isLower = TRUE, plot = FALSE)

outlier.cutoff(x, lBound = NULL, uBound = NULL)
```

Arguments

`x`	An integer/numeric vector used as the input
`alpha,z.cutoff`	alpha is the percentage of the standard deviation from the center of the data. z.cutoff is the standardized z-score value. They are used as the distribution based thresholds.
`isUpper,isLower`	logical scalars indicating whether the outliers are checked at upper or lower side of the distribution.
`plot`	logical scalar indicating whether to visualize the outlier detection results.
`...`	other arguments to be passed to qoutlier function,currently it is ignored.
`lBound,uBound`	Numeric scalars used as cutoff threshold for either lower limit or upper limit

Details

These different outlier detection functions are used together with qaCheck method to perform outlier checks.

Value

a logical vector with the same length of input vector,indicating whether each entry of the input is a outlier.

Author(s)

Mike Jiang,Greg Finak

Maintainer: Mike Jiang <wjiang2@fhcrc.org>

See Also

qaCheck,qaReport

`qaPreprocess` *Preprocessing for QA check*

Description

A convenient wrapper that does saveToDB, getQAStats in one call

Usage

```
qaPreprocess(db = .db, gs, gs.name = "default gatingSet", metaFile,
  fcs.colname = "name", date.colname = NULL, date.format = "%m/%d/%y",
  ...)
```

Arguments

db	An environment storing all the QA data. By default it is an hidden global environment .db.
gs	A GatingSet containing multiple gating hierarchies
gs.name	A character scalar giving the name of the GatingSet.
metaFile	A character scalar giving the file path of the sample annotation data, which is a csv spreadsheet contains the meta information. Each row corresponds to one FCS file and The QUALIFIER package looks for the FCS filename from "name" column of the spreadsheet.
fcs.colname	A character scalar indicating column name that specify FCS file names in annotation data.
date.colname	A character scalar indicating column names that contains date information which are automatically formatted to date object in R

date.format	A character scalar indicating the format of date column , default is "%m/%d/%y".. see as.Date for more details.
...	other arguments passed to getQAStats

Value

a list of elements stored in the data environment.

Examples

```
## Not run:
#prepare the data environment
db<-new.env()
initDB(db)

qaPreprocess(db=db,gs=G
,metaFile=metaFile
,fcs.colname="FCS_Files"
,date.colname=c("RecdDt","AnalysisDt")
)


## End(Not run)
```

---

qaReport *create quality assessment report*

Description

create quality assessment report in HTML format, which contains the annotated svg summary plot for each QA task.

Usage

```
## S4 method for signature 'list'
qaReport(obj, outDir, plotAll = FALSE, gsid = NULL, subset,
  ...)
```

Arguments

obj	A qaTask or a list of qaTask objects
outDir	A character scalar giving the output path of QA report.
plotAll	A logical scalar passed down to the plot method.
gsid	an integer that uniquely identifies a gating set object. if missing, the latest added gating set is selected.
subset	A logical expression as a filter that is passed to plot. see qaCheck for more details.

...	other arguments: splash: A logical scalar indicating whether the QUALIFIER package splash tag 'generated by ...' should be written right after the report title. title: A character scalar indicating the title of the report subTitle: A character scalar indicating the subtitle of the report

Details

QA results need to be calculated by calling qaCheck method before using this function to generate meaningful report. It also reads the meta information of each QA task and generate the summary tables and svg plots. Svg plots provide tooltips containing the detail information about each sample and hyperlinks of densityplot for each individual FCS file.

Author(s)

Mike Jiang,Greg Finak

Maintainer: Mike Jiang <wjiang2@fhcrc.org>

See Also

qaCheck,plot

Examples

```
## Not run:
data("ITNQASTUDY")#load stats from disk
checkListFile<-file.path(system.file("data",package="QUALIFIER"),"qaCheckList.csv.gz")
qaTask.list<-read.qaTask(db,checkListFile)
qaReport(qaTask.list[[1]],outDir="~/output")
qaReport(qaTask.list[2:3],outDir="~/output")

## End(Not run)
```

---

qaTask-class *a class for storing important information of flow cytometry data quality assessment task*

Description

This class stores the meta information, the name of gated cell population associated with the QA task and the formula describing how the QA task is performed.

Details

`qaID`: A integer for the id of qaTask object

`qaName`: A character containing the QA task name

`description`: OA character containing the description of QA task.

`qaLevel`: A character vector containing QA task level, which is displayed in the html report.

`pop`: A character containing the name of the cell population ,which is also equivalent to the name of the node in the gating hierarchy tree. It basically tells qaCheck or plot methods the particular gated cell population from which statistics are extracted.

`formula`: a formula describing how the QA task is performed. see `qaCheck` for more details.

`type`: a character indicate how pop is matched.

`subset`: an expression specifying a subset of data to be applied QA task.

`plotType`: A character indicating how the QA result is plotted. Currently only "xyplot" and "bwplot" are supported.

`width`: A numeric scalar indicating the width of svg figure .

`height`: A numeric scalar indicating the height of svg figure .

`par`: A list storing all the lattice arguments to control the appearance of the plot. See `xyplot` for details.

`scatterPar`: A list storing the arguments to control the appearance of the individual plot. Example scatterPar value: list(type="densityplot",scales=list(x=list(log=TRUE))) this will set the log scale for x axis of the 1D-densityplot

`htmlReport`: A logical scalar indicating whether to plot the task when `qaReport` is called. By default, when there are no outliers detected for the qaTask,it is not plotted in the report. But sometime it is helpful to still look at the plot with the trend of the data like MFI stability over time. So by setting this flag to TRUE, we force this task to be plotted anyway.

`highlight`: A character scalar indicating the level on which the dots in svg output should be highlighted when they are hovered over by mouse. It is one of the visualization feature provided by svg format that helps to identify a couple of dots that share the same information (like FCS name,or sample ID). It should be a valid column name in the metaFile (see `getQAStats` for more details about the meta file.

`rFunc`: A regression function passed to some qa tasks to monitor the long term trend.

`outlierFunc`: An outlier detection function.

`outlierFunc_args`: named arguments passed to the outlier detection function.

`goutlierFunc`: A group outlier detection function.

`goutlierFunc_args`: named arguments passed to the group outlier detection function.

`db`: An environment containing the database connection, which stores the gating hierarchy,QA task list,sample information and outliers detection results. .

Objects from the Class

Objects can be created by using constructor: read.qaTask(db=.db,checkListFile)

Author(s)

Mike Jiang,Greg Finak

Maintainer: Mike Jiang <wjiang2@fhcrc.org>

See Also

qaCheck,plot,qaReport

---

queryStats *Perform the quality assessment for the qaTask object*

Description

queryStats method queries stats entries from db by qaTask object and formula

codeclearCheck function removes the outlier results detected by the previous qaCheck call on a particular gating set.

Perform the quality assessment for a particular QA Task based on the information provided by qaTask object.

Usage

```
queryStats(x, ...)

clearCheck(obj, gsid)

## S4 method for signature 'qaTask'
qaCheck(obj, ...)
```

Arguments

x	a qaTask object
...	formula: a formula describing the variables to be used for QA. When it is omitted or NULL, the formula stored in qaTask is used. It is generally of the form y ~ x | g1 * g2 * ... , y is the statistics to be checked in this QA, It must be one of the four types: "MFI": Median Fluorescence Intensity of the cell population specified by qaTask, "proportion": the percentage of the cell population specified by qaTask in the parent population, "count": the number of events of the cell population specified by qaTask, "spike": the variance of intensity over time of each channel ,which indicating the stability of the fluorescence intensity. x is normally used to specify the variable plotted on x-axis in plot method. when plotType of the qaTask is "bwplot", it is also taken as the conditioning variable that divides the samples into subgroups within which the outlierfunc is applied.

g1,g2,.... are the conditioning variables, which are used to divide the samples into subgroups and perform QA check whitin each individual groups.They may also be omitted,in which case the outliers detection is peformed in the entire sample set.

subset: a logical expression used as a filter.It follows the same syntax as the "subset" expression in `subset`.

Usage:

subset=channel%in%c('FITC-A')

subset=Tube=='CD8/CD25/CD4/CD3/CD62L'&channel%in%c('FITC-A')

outlierfunc:a `function` to be used for outlier detection. see `outlierFunctions` for more details.

gOutlierfunc:a `function` to be used for group outlier detection. see `outlierFunctions` for more details. rFunc:a `function` for fitting regression model within each individual subgroup.

isTerminal:a logical scalar indicating whether the pop is at terminal node of the gating path.

fixed:a logical scalar indicating whether the pop name is matched as it is .By default it is FALSE,which matches the gating path as the regular expression

`obj` a `qaTask` object

`gsid` an `integer` that uniquely identifies a gating set object. if missing, the latest added gating set is selected.

Details

`qaCheck` method parses the formula stored in qaTask or explicitly provided by the argument and select the appropriate gated population,extract the statistics that is pre-calculated by `getQAStats` and perform the outlier detection within a certain sample groups specified by the conditioning variables or x term in formula. Then the outliers detection results are save in database and ready for query or plotting.

Author(s)

Mike Jiang,Greg Finak

Maintainer: Mike Jiang <wjiang2@fhcrc.org>

See Also

`plot`,`getQAStats`

Examples

```
## Not run:

data("ITNQASTUDY")
checkListFile<-file.path(system.file("data",package="QUALIFIER"),"qaCheckList.csv.gz")
qaTask.list<-read.qaTask(db,checkListFile)
```

```
#using t-distribution based outlier detection function
#applied the linear regression on each group to detect the significant MFI change over time
qaCheck(qaTask.list[["MFIOverTime"]]
,outlierfunc=outlier.t
,rFunc=rlm
,alpha=0.05
)
plot(qaTask.list[["MFIOverTime"]],y=MFI~RecdDt|stain
,subset="channel%in%c('FITC-A')"
,rFunc=rlm
)


#detect the outliers that has lower percentage of RBC Lysis than the threshold provided by lBound
qaCheck(qaTask.list[["RBCLysis"]]
,formula=proportion ~ RecdDt | Tube
,outlierfunc=outlier.cutoff
,lBound=0.8
)

plot(qaTask.list[["RBCLysis"]])

## End(Not run)
```

`read.qaTask` *Loading* `qaTask`*s from a csv file*

Description

The csv file contains the definition of one `qaTask`s

Usage

```
read.qaTask(checkListFile, ...)
```

Arguments

`checkListFile`	A character scalar giving the file path, which is a csv spreadsheet contains the detailed information of each QA task. It should have the columns: 'qaID', 'qaName', 'description', 'qaLevel','pop', 'type', 'formula', 'subset', 'plotType'. See the slots of `qaTask-class` for more details.
`...`	other arguments • db `environment` See the slots of `qaTask-class` for more details.

Value

a list of `qaTask` objects

Examples

```
## Not run:
 checkListFile <- file.path(system.file("data", package = "QUALIFIER"), "qaCheckList.csv.gz")
 qaTask.list <- read.qaTask(db, checkListFile)
 qaTask.list[[1]]

## End(Not run)
```

---

saveToDB *Save the gating set and annotation data into the data environment.*

Description

Save the gating set and annotation data into the data environment.

Usage

```
saveToDB(db = .db, gs, gs.name = "default gatingSet", metaFile,
  fcs.colname = "name", date.colname = NULL, date.format)
```

Arguments

db	An environment storing all the QA data. By default it is an hidden global environment .db.
gs	A GatingSet containing multiple gating hierarchies
gs.name	A character scalar giving the name of the GatingSet.
metaFile	A character scalar giving the file path of the sample annotation data, which is a csv spreadsheet contains the meta information. Each row corresponds to one FCS file and The QUALIFIER package looks for the FCS filename from "name" column of the spreadsheet.
fcs.colname	A character scalar indicating column name that specify FCS file names in annotation data.
date.colname	A character scalar indicating column names that contains date information which are automatically formatted to date object in R
date.format	A character scalar indicating the format of date column , default is "%m/%d/%y".. see as.Date for more details.

Value

An unique id for GatingSet that is generated incrementally.

Examples

```
## Not run:
#prepare the data environment
db<-new.env()
initDB(db)

metaFile="~/rglab/workspace/QUALIFIER/misc/ITN029ST/FCS_File_mapping.csv"
##append the annotation  and Gating set to db
metaFile<-"FCS_File_mapping.csv"
saveToDB(db=db,gs=G
,metaFile=metaFile
,fcs.colname="FCS_Files"
,date.colname=c("RecdDt","AnalysisDt")
)


## End(Not run)
```

---

save_db *save/load the data environment to/from disk*

Description

save and load the data environment that contains both statistics and GatingSets.

Usage

```
save_db(db = .db, path, overwrite = FALSE, cdf = "link", ...)

load_db(path)
```

Arguments

db	An environment storing all the QA data. By default it is an hidden global environment .db.
path	character data path that stores the db.
overwrite	logical whether to overwrite the existing folder
cdf	character the option to control cdf file operation. see save_gs for more details.
...	other arguments passed to save_gs

Examples

```
## Not run:
save_db(db, path = "./PreprocessedData")
db <- load_db(path = "./PreprocessedData")

## End(Not run)
```

Index