

Package ‘SwathXtend’

May 2, 2024

Type Package

Title SWATH extended library generation and statistical data analysis

Version 2.26.0

Date 2017-06-25

Author J WU and D Pascovici

Maintainer Jemma Wu <jwu@proteome.org.au>

Depends e1071, openxlsx, VennDiagram, lattice

Description Contains utility functions for integrating spectral libraries for SWATH and statistical data analysis for SWATH generated data.

biocViews Software

License GPL-2

RoxygenNote 5.0.1

NeedsCompilation no

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

git_branch RELEASE_3_19

git_last_commit ffeafe5

git_last_commit_date 2024-04-30

Repository Bioconductor 3.19

Date/Publication 2024-05-01

Contents

| | |
|-------------------------------|---|
| applyttest | 2 |
| applyttestPep | 3 |
| buildSpectraLibPair | 4 |
| canonicalFormat | 5 |
| checkQuality | 6 |
| cleanLib | 7 |
| coverage | 8 |

| | |
|-----------------------------------|----|
| cv | 8 |
| fdr.crit | 9 |
| getFdrBins | 9 |
| ionCorGS | 10 |
| medianNorm | 11 |
| mlr | 11 |
| mlrGroup | 12 |
| mlrrep | 13 |
| outputLib | 14 |
| plotAll | 14 |
| plotDensities | 15 |
| plotErrorBarsLines | 16 |
| plotRelativeDensities | 17 |
| plotRIICor | 17 |
| plotRTCOR | 18 |
| plotRTResd | 19 |
| quantification.accuracy | 20 |
| readLibFile | 21 |
| reliabilityCheckLibrary | 22 |
| reliabilityCheckSwath | 22 |
| swath.means | 23 |

| | |
|--------------|-----------|
| Index | 25 |
|--------------|-----------|

| | |
|------------|--|
| applyttest | <i>Utility to apply a t-test to all rows of a matrix</i> |
|------------|--|

Description

Generate fold change and t-test p-value for all rows of a data matrix

Usage

```
applyttest(mat, Group, doLogs = TRUE, numerator = levels(Group)[1])
```

Arguments

| | |
|-----------|--|
| mat | Matrix containing data, possibly with missing values |
| Group | Group with two levels of length equal to the number of matrix columns |
| doLogs | True/false, log data before applying test |
| numerator | The level of the group used as numerator for the fold change, by default the first one |

Value

Data frame with two values, t-test p-value and fold change.

See Also[applyttestPep](#)**Examples**

```
mat = matrix(rnorm(600), nrow=100)
mat[1:20, 1:3] = 3+mat[1:20, 1:3] # create some differences
mat[30, 1:3] = NA # and some missing values
mat[100,] = NA

applyttest(mat, Group = rep(c("A", "B"), each=3), doLogs=FALSE)
applyttest(abs(mat), Group = rep(c("A", "B"), each=3), doLogs=TRUE)
```

`applyttestPep`*Function to apply t-test separately for all peptides of each protein*

Description

Generate fold changes and p-values for each protein (col 1) determined by a number of peptides (col 2).

Usage

```
applyttestPep(peptides, Group, doLogs = TRUE, numerator = levels(as.factor(Group))[1])
```

Arguments

| | |
|-----------|---|
| peptides | Data frame with two descriptive columns: proteins, peptides, then data in the remaining ncol - 2 columns. |
| Group | Factor describing data membership. Must have two levels, and length = ncol(mat) - 2. |
| doLogs | TRUE/FALSE, log-transform data prior to analysis |
| numerator | The group level used as the numerator in the fold change. |

Value

Data frame with rows Protein, fold change and p-value.

See Also[applyttest](#)

Examples

```
# make random matrix with first 10 proteins differentially expressed
mat = exp(6+matrix(rnorm(6000), ncol=6))
Protein = sort(paste("P", sample(1:300, 1000, replace=TRUE)))
Peptide = paste("Pep", 1:1000)
for (j in 1:10) mat[Protein == unique(Protein)[j], 4:6] = 3*mat[Protein == unique(Protein)[j], 1:3]

res = applyttestPep(data.frame(Protein, Peptide, mat), rep(c("A", "B"), each=3), numerator="B")
# first 10 proteins should have fold change 3
plot(log(res$FC), -log(res$pval), col=rainbow(2)[1+ as.numeric(1:1000 > 10)])

# add some missing values
mat[5:20,4] = NA
res = applyttestPep(data.frame(Protein, Peptide, mat), rep(c("A", "B"), each=3), numerator="B")
# first 10 proteins should have fold change 3
plot(log(res$FC), -log(res$pval), col=rainbow(2)[1+ as.numeric(1:1000 > 10)])
```

buildSpectraLibPair *Build a spectra library by integrating a pair of spectrum libraries*

Description

Build a spectra library by integrating a pair of spectrum libraries

Usage

```
buildSpectraLibPair(baseLib, extLib, hydroIndex, method = c("time", "hydro",
  "hydrosequence"), includeLength = FALSE, labelBase = NA, labelAddon = NA,
  formatBase = c("peakview", "openswath"), formatExt = c("peakview",
  "openswath"), outputFormat = c("peakview", "openswath"),
  outputFile = "extendedLibrary.txt", plot = FALSE,
  clean = TRUE, merge = TRUE, ...)
```

Arguments

| | |
|---------------|--|
| baseLib | a base library data frame or file |
| extLib | an external/addon library data frame or file |
| hydroIndex | a data frame or file containing peptide hydrophobicity index |
| method | a character string to specify the RT alignment method. One of "time" (default), "hydro" and "hydrosequence" can be selected. |
| includeLength | a logic value representing if include peptide length as a feature for predicting retention time. Only applicable when method is "hydro". |
| labelBase | a character string to specify the labels of proteins from the base library |
| labelAddon | a character string to specify the labels of proteins from the addon library |

| | |
|--------------|---|
| formatBase | a character string denoting the file format of base library file. One of "peakview" (default) and "openswath" |
| formatExt | a character string denoting the file format of addon library file. One of "peakview" (default) and "openswath" |
| outputFormat | a character string denoting the file format of the output integrated library. One of "peakview" (default) and "openswath" |
| outputFile | A character string to specify the spectra library created |
| plot | a logic value, representing if plots during processing will be plotted or not |
| clean | a logic value, representing if the input libraries will be cleaned before integration. Default value is True. |
| merge | a logic value, representing if the output will be the merged library (default) or the adjusted add-on library. |
| ... | Additional parameters to pass in. |

Value

A data frame of the integrated spectrum library

Examples

```
libfiles <- paste(system.file("files",package="SwathXtend"),
c("Lib2.txt","Lib3.txt"),sep="/")
Lib2_3 <- buildSpectraLibPair(libfiles[1], libfiles[2],
outputFormat="peakview", clean=TRUE, nomod=TRUE, nomc=TRUE)
```

| | |
|-----------------|--|
| canonicalFormat | <i>Standardise a spectrum library data frame</i> |
|-----------------|--|

Description

Standardise a spectrum library data frame

Usage

```
canonicalFormat(dat, format = c("peakview", "openswath"))
```

Arguments

| | |
|--------|--|
| dat | a data frame of a spectrum library |
| format | a character string, representing the format of the input spectrum library. One of "peakview" (default) and "openswath" |

Value

a data frame of the library in canonical format

Examples

```
file <- paste(system.file("files", package="SwathXtend"), "Lib1.txt", sep="/")
dat <- read.delim2(file, sep="\t", stringsAsFactor = FALSE, header=TRUE)
dat <- try(canonicalFormat(dat, format = "peakview"))
```

checkQuality

Checking for the integration quality of two libraries

Description

Checking for the integration quality of two libraries

Usage

```
checkQuality(datBaseLib, datExtLib, ...)
```

Arguments

| | |
|------------|------------------------------------|
| datBaseLib | a data frame of the base library |
| datExtLib | a data frame of the add-on library |
| ... | Additional parameters to pass in |

Value

A list of quality indicators, including squared retention time (RT) correlation coefficient, root mean squared errors of RT residuals, and median of relative ion intensity correlation coefficient

Examples

```
libfiles <- paste(system.file("files", package="SwathXtend"),
  c("Lib2.txt", "Lib3.txt"), sep="/")
datBaseLib <- readLibFile(libfiles[1])
datExtLib <- readLibFile(libfiles[2])
res <- checkQuality(datBaseLib, datExtLib)
```

| | |
|----------|----------------------------------|
| cleanLib | <i>Spectrum library cleaning</i> |
|----------|----------------------------------|

Description

Spectrum library cleaning

Usage

```
cleanLib(datLib, clean = TRUE, intensity.cutoff = 5, conf.cutoff = 0.99,  
         nomod = FALSE, nomc = FALSE, enz = c("trypsin", "gluc", "chymotrypsin"))
```

Arguments

| | |
|------------------|--|
| datLib | a data frame for a spectrum library |
| clean | a logic value indicating if the library will be cleaned. Default value is TRUE. |
| intensity.cutoff | A number value to specify cut off for relative intensity of fragment ions. Only ions with intensity higher than the cut off value (default as 5) will be kept. |
| conf.cutoff | A number value to specify cut off for precursor confidence. Only ions with confidence higher than the cut off value (default as 0.99) will be kept. |
| nomod | a logic value, representing if the modified peptides and its fragment ions will be removed. True (default) means will be removed. |
| nomc | a logic value, representing if peptides with miss cleavages are removed. Default vaue is False (not to remove). |
| enz | A character string representing the enzyme which can be one of "trypsin" (default), "gluc", or "chymotrypsin" |

Value

a data frame of a cleaned spectrum library by the specified criteria

Examples

```
file <- paste(system.file("files", package="SwathXtend"), "Lib1.txt", sep="/")  
dat <- read.delim2(file, sep="\t", header=TRUE, stringsAsFactors=FALSE)  
dat <- canonicalFormat(dat)  
dat <- cleanLib(dat)
```

| | |
|----------|--|
| coverage | <i>A function to calculate the coverage percentage</i> |
|----------|--|

Usage

```
coverage(a, b)
```

Arguments

| | |
|---|--|
| a | A vector of numerical or string elements |
| b | A vector of numerical or string elements |

Details

The percentage of a that is covered by b

Value

A numeric value representing the coverage percentage of b for a which is defined as the ratio of intersection of a and b over the size of a

Examples

```
coverage(c('a', 'b', 'c'), c('b', 'c', 'd'))
```

| | |
|----|--|
| cv | <i>A function to calculate the CV (Coefficient of Variation)</i> |
|----|--|

Usage

```
cv(v)
```

Arguments

| | |
|---|------------------|
| v | A numeric vector |
|---|------------------|

Value

A numeric vector representing the Coefficient of Variance.

Examples

```
cv(rnorm(100))
```

| | |
|----------|---|
| fdr.crit | <i>A function to calculate the number of samples pass fdr threshold</i> |
|----------|---|

Usage

```
fdr.crit(dswat.fdr)
```

Arguments

dswat.fdr A data frame of fdr values of a Swath result

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==> Define data, use random,
##--or do help(data=index) for the standard data sets.

## The function is currently defined as

file= paste(system.file("files", package="SwathXtend"),
"Swath_result_Lib2.xlsx", sep="/")

dswat.fdr = readWorkbook(file, sheet='FDR')

dat = fdr.crit(dswat.fdr)
```

| | |
|------------|---|
| getFdrBins | <i>Function to calculate the percentage of fdrs in each bin</i> |
|------------|---|

Usage

```
getFdrBins(mat.fdr, Bins = c(0, 0.01, 0.1, 0.2, 0.3, 0.4, 0.5, 0.8, 1))
```

Arguments

mat.fdr A matrix of fdr values

Bins A numeric vector representing the bins. For n bins, there will be n+1 numbers in the vector.

Value

A numeric vector representing the percentage of each FDR bin.

Examples

```
#
fswaths = paste(system.file("files", package="SwathXtend"), c("Swath_result_Lib2.xlsx", "Swath_result_Lib2_3.xlsx"))

fdr.seed = readWorkbook(fswaths[1], sheet='FDR')
fdr.ext = readWorkbook(fswaths[2], sheet='FDR')

Bins = c(0, .01, .1, .2, .3, .4, .5, .8, 1)

res = getFdrBins(as.matrix(fdr.ext[, -c(1:7)]), Bins)
```

ionCorGS

Gold standard relative ion intensity correlation (spearman)

Description

This data set gives the relative ion intensity spearman correlation for 2023 peptides as the gold standard for benchmarking the matching quality of two peptide assay libraries.

Usage

```
data(ionCorGS)
```

Format

A vector containing spearman correlation coefficient for 2023 peptides.

Value

a numeric vector

Source

APAF

References

APAF

| | |
|------------|--|
| medianNorm | <i>Utility to median normalize a matrix by columns</i> |
|------------|--|

Description

Divide appropriately to make all column medians equal to the max median

Usage

```
medianNorm(mat)
```

Arguments

mat Data matrix to normalize; matrix assumed positive

Value

Matrix of same dimensions.

Examples

```
mat = 100+matrix(rnorm(1000), ncol=10)
mat[,10] = mat[,10] + 2
layout(matrix(1:2, nrow=1))
boxplot(mat)
boxplot(medianNorm(mat))

# note: issues when medians close to 0.
```

| | |
|-----|--|
| mlr | <i>Function to implement mlr normalization</i> |
|-----|--|

Description

Calculate normalization factor, histogram peak and width at half peak for a vector

Usage

```
mlr(ratio, doplot)
```

Arguments

ratio Vector, typically of log ratios
doplot A logic value, wheter to plot the ratio histograms (FALSE as default)

Value

| | |
|------|----------------------|
| nf | Normalization factor |
| peak | Histogram peak |
| wdt | Width at half peak |

References

Find mlr reference.

Examples

```
mlr(rnorm(1000))  
# with shift  
mlr(0.5 + rnorm(10000))
```

mlrGroup

Function to do mlr normalization for a matrix group

Description

Do mlr normalization separately for each set of replicates first, then normalize the resulting matrix

Usage

```
mlrGroup(mat, Group)
```

Arguments

| | |
|-------|--|
| mat | Data matrix with replicates as columns |
| Group | Factor of length ncol(mat) |

Value

Resulting normalized matrix of the same size as the initial one

References

Find reference to mlr paper

See Also

[mlrrep](#), [mlr](#)

Examples

```
res = mlrGroup(iris[,-5], Group=as.factor(c("Sepal", "Sepal", "Petal", "Petal")))

layout(matrix(1:3, nrow=1))
boxplot(log(iris[,-5]), main="Log only")
boxplot(log(medianNorm(iris[,-5])), main="Median")
boxplot(log(res[[1]]), main="MLR")
```

mlrrep

*Function to do mlr normalizatiopn on a matrix of replicates***Description**

Calculate all pairwise ratios, log-transform them, find the least variable replicate.

Usage

```
mlrrep(mat)
```

Arguments

mat Data matrix with replicates as columns

Value

mat.norm Normalized data matrix; matrix assumed positive
 wdmatrix Square matrix of half peak widths for each ratio of replicates of size ncol(mat)
 nfmatrix Square matrix of normalization factors for each ratio of replicates of size ncol(mat)
 idx Index of replicate to be used as denominator yielding smallest widths

See Also

[mlr](#), [mlrGroup](#)

Examples

```
# Example using the iris data
mlrrep(iris[,-5])

# random data
mat = exp(matrix(rnorm(1000),ncol=4))
res = mlrrep(mat)
layout(matrix(1:2, nrow=1))
boxplot(log(res$mat.norm))
boxplot(log(mat))
```

| | |
|-----------|--|
| outputLib | <i>output a spectrum library into a PeakView format file</i> |
|-----------|--|

Description

output a spectrum library into a PeakView format file

Usage

```
outputLib(dat, filename = "NewLib.txt", format = c("peakview", "openswath"),
  nodup = TRUE)
```

Arguments

| | |
|----------|---|
| dat | A data frame of a spectrum library |
| filename | A character string for the name of the output. |
| format | A character string representing the output format. One of "peakview" (default) and "openswath". |
| nodup | A logic value, indicating if remove duplicated spectrum (default) |

Value

a file with the specified file name (lib.txt as default) will be saved under the current working directory

Examples

```
file <- paste(system.file("files", package="SwathXtend"), "Lib1.txt", sep="/")
dat <- readLibFile(file)
outputLib(dat)
```

| | |
|---------|---|
| plotAll | <i>Plot statistical plots for two libraries</i> |
|---------|---|

Description

Plot statistical plots for two libraries

Usage

```
plotAll(datBaseLib, datExtLib, file = "allplots.xlsx", ...)
```

Arguments

datBaseLib a data frame for a base spectrum library
datExtLib a data frame for a external spectrum library
file a character string for the output file
... Additional parameters to pass in

Value

a list of two data frames

Examples

```
libfiles <- paste(system.file("files", package="SwathXtend"),  
c("Lib2.txt", "Lib3.txt"), sep="/")  
datBaseLib <- readLibFile(libfiles[1])  
datExtLib <- readLibFile(libfiles[2])  
res <- plotAll(datBaseLib, datExtLib)
```

plotDensities

Utility to do side by side density plots

Description

Side by side density plots

Usage

```
plotDensities(data, group = rownames(data), xlab = "Log Abundance")
```

Arguments

data Data with samples as columns.
group Group of the same length as the number of columns of data
xlab Label to be printed

Value

No value returned, plotting only

Examples

```
plotDensities(iris[,-5], rep(c("A", "B"), each=2))
```

plotErrorBarsLines *Utility for clustering plots to plot lines and an overall trend*

Description

Prints faint lines for each profile, and a mean/error bars

Usage

```
plotErrorBarsLines(v, barSizes, lines, labels = NULL, col = "blue", ylim, ...)
```

Arguments

| | |
|----------|--|
| v | Overall trend, to be printed solid, length n |
| barSizes | Size of the error bars, length n |
| lines | Matrix of n columns, and as many rows as lines |
| labels | Labels to be printed on the x axis, length n |
| col | Colour for main trend line |
| ylim | Can specify limits so several graphs are on the same scale |
| ... | Additional parameters to pass in |

Value

No returned value; plot only.

See Also

[help, ~~~](#)

Examples

```
mat = matrix(rnorm(100), 10)
plotErrorBarsLines(apply(mat,1,FUN=mean), apply(mat,1,FUN=sd),
lines=mat, col="red", main="A random plot", xlab="Some label")
```

plotRelativeDensities *Plotting utility to overlay all relative densities*

Description

Overlay all relative densities

Usage

```
plotRelativeDensities(mat, Group = NULL, idx = NULL, main = "Densities")
```

Arguments

| | |
|-------|--|
| mat | Matrix with positive entries, samples as columns |
| Group | The factor showing the sample membership, of length ncol(mat) |
| idx | Number between 1:ncol(mat); which sample to use as denominator, first one by default |
| main | Title; optional |

Value

Plotting only

Examples

```
mat = matrix(abs(rnorm(50000)), ncol=5)
mat[,5] = mat[,5] + 2

plotRelativeDensities(mat, Group=c(rep("A",4),"B"), idx=1)
```

plotRIICor *Plot relative ion intensity correlation of two libraries*

Description

Plot relative ion intensity correlation of two libraries

Usage

```
plotRIICor(dat1, dat2, nomod = FALSE)
```

Arguments

| | |
|-------|---|
| dat1 | A data frame containing the first spectrum library |
| dat2 | A data frame containing the second spectrum library |
| nomod | a logic value, representing if the modified peptides and its fragment ions will be removed. FALSE (default) means not removing. |

Value

a data frame of relative ion intensity correlations for all ions

Examples

```
libfiles <- paste(system.file("files",package="SwathXtend"),
  c("Lib2.txt","Lib3.txt"),sep="/")
datBaseLib <- readLibFile(libfiles[1])
datExtLib <- readLibFile(libfiles[2])
plotRIICor(datBaseLib, datExtLib)
```

plotRTCor

Plot for retention time correlation of two libraries

Description

Plot for retention time correlation of two libraries

Usage

```
plotRTCor(dat1, dat2, label1, label2, nomod = FALSE)
```

Arguments

| | |
|--------|---|
| dat1 | A data frame containing the first spectrum library |
| dat2 | A data frame containing the second spectrum library |
| label1 | a character string representing the x axis label for plotting |
| label2 | a character string representing the y axis label for plotting |
| nomod | a logic value, representing if the modified peptides and its fragment ions will be removed. FALSE (default) means not removing. |

Value

retention time correlation coefficient

Examples

```
libfiles <- paste(system.file("files",package="SwathXtend"),
  c("Lib2.txt","Lib3.txt"),sep="/")
datBaseLib <- readLibFile(libfiles[1])
datExtLib <- readLibFile(libfiles[2])
plotRTCor(datBaseLib, datExtLib, "Lib2", "Lib5")
```

plotRTResd

Plot residuals for retention time prediction of two libraries

Description

Plot residuals for retention time prediction of two libraries

Usage

```
plotRTResd(dat1, dat2, nomod = FALSE)
```

Arguments

| | |
|-------|---|
| dat1 | A data frame containing the first spectrum library |
| dat2 | A data frame containing the second spectrum library |
| nomod | a logic value, representing if the modified peptides and its fragment ions will be removed. FALSE (default) means not removing. |

Value

root mean square error of prediction residuals

Examples

```
libfiles <- paste(system.file("files",package="SwathXtend"),
  c("Lib2.txt","Lib3.txt"),sep="/")
datBaseLib <- readLibFile(libfiles[1])
datExtLib <- readLibFile(libfiles[2])
plotRTResd(datBaseLib, datExtLib)
```

quantification.accuracy

Measurement of quantification accuracy of two Swath results

Usage

```
quantification.accuracy(dswat1, dswat2, Sample = NULL, method = c("cor", "cv", "bland.altman"),
  cor.method=c('pearson', 'spearman', 'kendall'), log = FALSE)
```

Arguments

| | |
|------------|---|
| dswat1 | A data frame of peptide peak area of the first Swath result |
| dswat2 | A data frame of peptide peak area of the second Swath result |
| Sample | A vector of strings representing the sample names of the Swath result |
| method | A string as one of "cor", "cv" and "bland.altman" |
| cor.method | A string as one of "pearson", "spearman", and "kendall" |
| log | A logical value indicating if the peak area will be log transformed before calculating the measurement. Default value is FALSE which means the peak area will not be transformed. |

Value

A list of two numeric vectors

| | |
|------|---|
| vcor | The measurement for the quantification accuracy for the same sample |
| rcor | The measurement for the quantification accuracy for the randomised sample |

Examples

```
fswaths = paste(system.file("files", package="SwathXtend"), c("Swath_result_Lib2.xlsx", "Swath_result_Lib2_3.xlsx"))
fdr.seed = readWorkbook(fswaths[1], sheet='FDR')
fdr.ext = readWorkbook(fswaths[2], sheet='FDR')

swa.seed = readWorkbook(fswaths[1], 2)
swa.ext = readWorkbook(fswaths[2], 2)

fdr.seed = fdr.crit(fdr.seed)
fdr.ext = fdr.crit(fdr.ext)

res = quantification.accuracy(swa.seed[fdr.seed$nfdr.pass >= 2,], swa.ext[fdr.ext$nfdr.pass >= 2,], method="cv")
```

readLibFile

*Load a spectrum library into a data frame***Description**

Load a spectrum library into a data frame

Usage

```
readLibFile(file, format = c("peakview", "openswath"), type = c("spectrum",
  "hydro"), clean = TRUE, ...)
```

Arguments

| | |
|--------|--|
| file | A file of a spectrum library, in .txt or .csv format, can be .gz files. |
| format | A character string denoting the file format. One of "peakview" (default) and "openswath". If the file format is "peakview", it requires the following columns: Q1: Q1 m/z (precursor m/z); Q3: Q3 m/z (fragment m/z); RT_detected: retention time; protein_name: protein name; isotype: isotype type; relative_intensity: fragment ion intensity; stripped_sequence: peptide sequences without modifications; modification_sequence: peptide sequences with modifications; prec_z: peptide charge; frg_type: fragment type; frg_z: fragment charge; frg_nr: ion number; iRT: calibrated retention time; uniprot_id: database accession number; decoy: whether the peptide a decoy or not; confidence: the confidence of the identified peptide; shared: whether the peptide is shared by multiple proteins; N: a ranking number for the protein. Optional columns for PeakView format libraries include: score: score for peptide identification; prec_y: the precursor ion intensity; rank: ion intensity ranking; mods: modification; nterm: N terminal modification; cterm: C terminal modification; If the file format is "openswath", it must contain the following columns: PrecursorMz: precursor m/z; ProductMz: fragment m/z; Tr_recalibrated: retention time; ProteinName: protein name; GroupLabel: isotype type; LibraryIntensity: fragment ion intensity; PeptideSequence: peptide sequences without modifications; FullUniModPeptideName: peptide sequences with modifications; UniprotID: database accession number; decoy: whether the peptide a decoy or not PrecursorCharge: precursor charge; FragmentType: fragment type (b or y ion); FragmentCharge: fragment charge; FragmentSeriesNumber: fragment ion number. |
| type | A character string denoting the file type. One of "spectrum" (default) and "hydro" |
| clean | A logic value, representing if the library will be cleaned. |
| ... | Additional parameters to pass in |

Value

a data frame of the library with cleaning process

Examples

```
file <- paste(system.file("files",package="SwathXtend"),"Lib1.txt",sep="/")
dat <- readLibFile(file)
```

reliabilityCheckLibrary

A function to check the coverage of the extended library given the seed library

Usage

```
reliabilityCheckLibrary(seedlib.file, extlib.file)
```

Arguments

seedlib.file A string representing the seed library file
 extlib.file A string representing the extended library file

Value

A matrix of number of protein and peptide of the seed and extended library

Examples

```
files <- paste(system.file("files",package="SwathXtend"),
  c("Lib2.txt", "Lib2_3.txt") ,sep="/")
res = reliabilityCheckLibrary(files[1], files[2])
```

reliabilityCheckSwath *A function to check the coverage, fdr distributions, quantification accuracy etc of two Swath results*

Usage

```
reliabilityCheckSwath(seed.swathfile, ext.swathfile, max.fdrpass = 3, max.peptide = 2)
```

Arguments

- `seed.swathfile` A string representing the Swath results obtained using the seed library. The Swath result file should be a PeakView extracted Excel (.xlsx) file with six tabs: "Area - ions", "Area - peptides", "Area - proteins", "Score", "FDR" and "Observed RT". The SWATH result checking functions require that worksheet "Area - peptides" and "FDR" must exist.
- `ext.swathfile` A string representing the Swath results obtained using the extended library. The Swath result file should be a PeakView extracted Excel (.xlsx) file with six tabs: "Area - ions", "Area - peptides", "Area - proteins", "Score", "FDR" and "Observed RT". The SWATH result checking functions require that worksheet "Area - peptides" and "FDR" must exist.
- `max.fdrpass` A numeric value representing the maximum number of samples that pass the fdr threshold (0.01)
- `max.peptide` A numeric value representing the maximum number of peptides in a protein as a filter

Value

- `fdr.bins` a matrix of the FDR percentage in each of the 8 bins
- `dat.comb` a matrix of the various numbers as the SWATH filtering threshold changes. These numbers include protein, peptide, median correlation, cv and bland altman mesurement.

Examples

```
files <- paste(system.file("files", package="SwathXtend"),
c("Swath_result_Lib2.xlsx", "Swath_result_Lib2_3.xlsx"), sep="/")
res = reliabilityCheckSwath(files[1], files[2])
```

| | |
|--------------------------|---|
| <code>swath.means</code> | <i>Computer Swath mean peak area for duplicated samples</i> |
|--------------------------|---|

Usage

```
swath.means(dswath, Sample)
```

Arguments

- `dswath` a data frame of peak areas of Swath results
- `Sample` a vector of strings of the sample names in the Swath result

Value

A data frame with the mean peak area.

Examples

```
file = paste(system.file("files",package="SwathXtend"),"Swath_result_Lib2.xlsx", sep="/")  
dswat = readWorkbook(file, 2)  
Sample = rep(c('2perc','5perc','10perc'), each=3)  
res = swath.means(dswat, Sample)
```


Index

* datasets

- ionCorGS, [10](#)
- applyttest, [2](#), [3](#)
- applyttestPep, [3](#), [3](#)
- buildSpectralLibPair, [4](#)
- canonicalFormat, [5](#)
- checkQuality, [6](#)
- cleanLib, [7](#)
- coverage, [8](#)
- cv, [8](#)
- fdr.crit, [9](#)
- getFdrBins, [9](#)
- help, [16](#)
- ionCorGS, [10](#)
- medianNorm, [11](#)
- mlr, [11](#), [12](#), [13](#)
- mlrGroup, [12](#), [13](#)
- mlrrep, [12](#), [13](#)
- outputLib, [14](#)
- plotAll, [14](#)
- plotDensities, [15](#)
- plotErrorBarsLines, [16](#)
- plotRelativeDensities, [17](#)
- plotRIICor, [17](#)
- plotRTCor, [18](#)
- plotRTResd, [19](#)
- quantification.accuracy, [20](#)
- readLibFile, [21](#)
- reliabilityCheckLibrary, [22](#)
- reliabilityCheckSwath, [22](#)
- swath.means, [23](#)