Package 'isobar'

March 26, 2013

Title Analysis and quantitation of isobarically tagged MSMS proteomics data

Description isobar provides methods for preprocessing, normalization, and report generation for the analysis of quantitative mass spectrometry proteomics data labeled with isobaric tags, such as iTRAQ and TMT.

Version 1.4.0

Author Florian P Breitwieser <fbreitwieser@cemm.oeaw.ac.at> and Jacques Colinge <jcolinge@cemm.oeaw.ac.at>, with contributions from Xavier Robin <xavier.robin@unige.ch>

Maintainer Florian P Breitwieser <fbreitwieser@cemm.oeaw.ac.at>

biocViews Proteomics, MassSpectrometry, Bioinformatics,MultipleComparisons, QualityControl

Depends R (>= 2.10.0), Biobase, stats, methods, plyr

Imports distr

Suggests MSnbase, OrgMassSpecR, XML, biomaRt, ggplot2, Hmisc, RJSONIO

LazyLoad yes

License LGPL-2

URL http://bioinformatics.cemm.oeaw.ac.at

Collate utils.R ProteinGroup-class.R IBSpectra-class.R IBSpectra-plots.R NoiseModel-class.R ratio-methods.R sharedpep-methods.R report-utils.R MSnSet-methods.R zzz.R

R topics documented:

---

isobar-package *Analysis and quantitation of isobarically tagged MSMS proteomics data*

Description

isobar provides methods for preprocessing, normalization, and report generation for the analysis of quantitative mass spectrometry proteomics data labeled withOA isobaric tags, such as iTRAQ and TMT.

Details

Package:	isobar
Version:	1.1.2
biocViews:	Proteomics, MassSpectrometray, Bioinformatics, MultipleComparisons, QualityControl
Depends:	R (>= 2.9.0), Biobase, stats, methods, ggplot2
Imports:	distr, biomaRt
Suggests:	MSnbase,XML
LazyLoad:	yes
License:	LGPL-2
URL:	http://bioinformatics.cemm.oeaw.ac.at
Collate:	utils.R ProteinGroup-class.R IBSpectra-class.R NoiseModel-class.R ratio-methods.R sharedpep-methods.R

Index:

IBSpectra-class IBSpectra objects

NoiseModel-class	NoiseModel objects
ProteinGroup-class	ProteinGroup objects
do.log	Log functions for IBSpectra objects
fitCauchy	Fit weighted and unweighted Cauchy and Normal distributions
groupMemberPeptides	Peptide info for protein group members
human.protein.names	Info on proteins
ibspiked_set1	Isobar Data packages
isobar-analysis	IBSpectra analysis: Protein and peptide ratio calculation
isobar-import	Loading data into IBSpectra objects using readIBSpectra
isobar-package	Analysis and quantitation of isobaric tag Proteomics data
isobar-plots	IBSpectra plots
isobar-preprocessing	IBSpectra preprocessing
isobar-reports	Isobar reports
maplot.protein	MAplot for individual proteins
number.ranges	Helper function to transform number lists to ranges
proteinInfo-methods	Methods for Function proteinInfo
proteinRatios	protein and peptide ratios
sanitize	Helper function for LaTeX export
shared.ratios	Shared ratio calculation
shared.ratios.sign	Plot and get significantly shared ratios.

Further information is available in the following vignettes:

isobar	Isobar Overview (source, pdf)
isobar-devel	Isobar for developers (source, pdf)

Author(s)

Florian P Breitwieser <fbreitwieser@cemm.oeaw.ac.at> and Jacques Colinge <jcolinge@cemm.oeaw.ac.at>, with contributions from Xavier Robin <xavier.robin@unige.ch>

Maintainer: Florian P Breitwieser <fbreitwieser@cemm.oeaw.ac.at>

---

calculate.dNSAF *dNSAF approximate abundance calculations.*

Description

Distributed normalized spectral abundance factor (dNSAF) is a label free quantitative measure of protein abundance based on spectral counts which are corrected for peptides shared by multiple proteins. Original publication: Zhang Y et al., Analytical Chemistry (2010).

Usage

```
calculate.dNSAF(protein.group)
```

Arguments

protein.group	ProteinGroup object. Its @proteinInfo slot data.frame must contain a length column.

Value

Named numeric vector of dNSAF values.

Author(s)

Florian P Breitwieser

References

Zhang Y et al., Analytical Chemistry (2010)

See Also

proteinInfo, getProteinInfoFromUniprot, calculate.emPAI, ProteinGroup

Examples

```
data(ibspiked_set1)
protein.group <- proteinGroup(ibspiked_set1)
calculate.dNSAF(protein.group)
```

---

calculate.emPAI *emPAI approximate abundance calculations.*

Description

The Exponentially Modified Protein Abundance Index (emPAI) is a label free quantitative measure of protein abundance based on protein coverage by peptide matches. The original publication is Ishihama Y, et al., Proteomics (2005).

Usage

```
calculate.emPAI(protein.group, protein.g = reporterProteins(protein.group), ...)
n.observable.peptides(...)
observable.peptides(seq, nmc = 1, min.length = 6, min.mass = 600, max.mass = 4000,
                    custom = list(code = "U", mass = 150.953636), ...)
```

Arguments

protein.group	ProteinGroup object. Its @proteinInfo slot data.frame must contain a sequence column to calculate the number of observable peptides per protein.
protein.g	Protein group identifiers.
seq	Protein sequence.
nmc	Number of missed cleavages.
min.length	Minimum length of peptide.

min.mass	Minimum mass of peptide.
max.mass	Maximum mass of peptide.
custom	User defined residue for Digest.
...	Further arguments to observable.peptides/Digest.

Details

The formula is

$$emPAI = 10^{\frac{N<-observed}{N<-observable}} - 1$$

N_observed is the number of observed peptides - we use the count of unique peptide without consideration of charge state. N_observable is the number of observable peptides. Sequence cleavage is done using Digest.

Value

Named numeric vector of emPAI values.

Author(s)

Florian P Breitwieser

References

Ishihama Y, et al., Proteomics (2005)

See Also

Digest, proteinInfo, getProteinInfoFromUniprot, calculate.dNSAF, ProteinGroup

Examples

```
data(ibspiked_set1)
protein.group <- proteinGroup(ibspiked_set1)
calculate.emPAI(protein.group,protein.g=protein.g(protein.group,"CERU"))
```

---

fit distributions *Fit weighted and unweighted Cauchy and Normal distributions*

Description

Functions to fit the probability density functions on ratio distribution.

Usage

```
fitCauchy(x,round.digits=NULL)
fitNorm(x, portion = 0.75)
fitWeightedNorm(x, weights)
fitNormalCauchyMixture(x)
fitGaussianMixture(x, n = 500)
fitGumbel(x)
fitTd(x)
```

Arguments

x	Ratios
round.digits	If not null, round fitted parameters to given precision.
weights	Weights
portion	Central portion of data to take for computation
n	number of sampling steps

Value

Cauchy,Norm

Author(s)

Florian P Breitwieser, Jacques Colinge.

See Also

proteinRatios

Examples

```
library(distr)
data(ibspiked_set1)
data(noise.model.hcd)
# calculate protein ratios of Trypsin and CERU_HUMAN. Note: this is only
#  for illustration purposes. For estimation of sample variability, data
#  from all protein should be used
pr <- proteinRatios(ibspiked_set1,noise.model=noise.model.hcd,
          cl=as.character(c(1,1,2,2)),combn.method="intraclass",protein=c("136429","P00450"))

# fit a Cauchy distribution
ratiodistr <- fitCauchy(pr$lratio)
plot(ratiodistr)
```

---

groupMemberPeptides *Peptide info for protein group members*

Description

For a given reporter protein group identifier, information on its peptides is returned. It contains information on how the peptides are shared and in which member they occur.

Usage

```
groupMemberPeptides(x, reporter.protein.g, ordered.by.pos = TRUE, only.first.pos = TRUE)
```

Arguments

x	ProteinGroup object
reporter.protein.g	group reporter protein
ordered.by.pos	if TRUE, start position of peptides in proteins is exported and peptides are ordered by position
only.first.pos	if TRUE, only first occurence of peptide in protein is reported

Value

list of two: [1] peptide.info: data.frame peptide specificity n.shared.groups n.shared.proteins start.pos [2] group.member.peptides: data.frame each column corresponds to a group member, and each row to a peptide

Author(s)

Florian P Breitwieser

Examples

```
data(ibspiked_set1)
protein.group <- proteinGroup(ibspiked_set1)
ceru.rat <- protein.g(protein.group,"CERU_RAT")
groupMemberPeptides(protein.group,ceru.rat)

## find protein groups with members
t <- table(proteinGroupTable(protein.group)$reporter.protein)
t[t>2]
protein.g <- names(t)[t>2][1]
groupMemberPeptides(protein.group,protein.g)
```

---

human.protein.names *Info on proteins*

Description

Gather human readable information from protein group codes.

Usage

```
my.protein.info(x, protein.g)


human.protein.names(my.protein.info)
```

Arguments

x	ProteinGroup object
protein.g	protein
my.protein.info	Return value of function my.protein.info

Author(s)

Florian P Breitwieser

IBSpectra-class *IBSpectra Class for Isobarically Tagged Quantitative MS Proteomics Data*

Description

This class represents a quantitative MS proteomics experiment labeled using Isobaric tags (iTRAQ, TMT). IBSpectra is a abstract class which is implemented in the IBSpectraTypes classes iTRAQ4plexSpectra, iTRAQ8plexSpectra, TMT2plexSpectra and TMT6plexSpectra.

It contains per-spectrum meassurements of the reporter tag intensity and m/z in assayData, and protein grouping in proteinGroup.

Objects from the Class

IBSpectra objects are typically created using the readIBSpectra method or by calls of the form new("iTRAQ4plexSpectra",data=NULL,data.ions=NULL,...).

Slots

IBSpectra extends eSet which is a container for high-throughput assays and experimental metadata. Slots introduced in eSet (for more details on slots and methods refer to eSet help):

assayData: Contains matrices 'ions' and 'mass storing reporter tag intensities and m/z values for each tag and spectrum. Can be accessed by reporterIntensities and reporterMasses. Class: AssayData

phenoData: Contains experimenter-supplied variables describing phenotypes behind reporter tags. Class: AnnotatedDataFrame-class

featureData: Describes the spectra's retention time, charge, peptide sequence, etc and can be accessed by fData. Class: AnnotatedDataFrame

experimentData: Contains details of experimental methods. Class: MIAME

annotation: UNUSED. Label associated with the annotation package used in the experiment. Class: character

protocolData: UNUSED. Contains equipment-generated variables describing reporter tags. Class: AnnotatedDataFrame

log: character matrix logging isotope impurity correction, normalization, etc.

Slots introduced in IBSpectra:

proteinGroup: A ProteinGroup object describing peptide and protein identifications grouped by shared peptides.

reporterTagNames: A character vector denoting the reporter tag labels.

reporterMasses: The 'true' m/z of the reporter tags in the MS/MS spectrum, used to isolate m/z-intensity pairs from peaklist.

isotopeImpurities: Manufacturer supplied isotope impurities, need to be set per batch and used for correction by correctIsotopeImpurities.

Constructor

See readIBSpectra for creation based on peaklist (e.g. MGF format) and identification files (Mascot and Phenyx output).

new(type,data): Creates a IBSpectra object.

type Denotes the type of IBSpectra, either 'iTRAQ4plexSpectra','iTRAQ8plexSpectra','TMT2plexSpectra' or 'TMT6plexSpectra'. Call IBSpectraTypes() to see a list of the implemented types.

data A 'data.frame' in a ibspectra-csv format.

Coercion

In the code snippets below, x is a IBSpectra object. IBSpectra object can be coerced to

as(x, "data.frame"): Creates a data.frame containing all identification and quantitation information. Peptide matching to multiple proteins produce multiple lines.

as(x, "data.frame.concise"): Creates a data.frame containing all identification and quantitation information. Proteins are concatenated - so the resulting data.frame has one line per spectrum.

as(x, "MSnSet"): Coerces to a MSnSet object (package MSnbase).

as(msnset,"IBSpectra"): Coerces a MSnSet to IBSpectra object.

Accessors

In the following code snippets, x is a IBSpectra object.

proteinGroup(x): Gets and sets the ProteinGroup.

isotopeImpurities(x): Gets and sets the isotope impurities of the isobaric tags as defined by the manufacturers per batch.

reporterData(x,element="ions",na.rm=FALSE,na.rm.f='any',...): Gets and sets the element ('ions' or 'mass') for each tag and spectrum. '...' is handed down to spectrumSel, so it is possible to select for peptides or proteins. If na.rm is TRUE, than spectra missing quantitative information in 'any' or 'all' channels (parameter na.rm.f) are removed.

reporterIntensities(x,...): Convenience function, calls reporterData(...,element="ions")

reporterMasses(x,...): Convenience function, calls reporterData(...,element="mass")

spectrumTitles(x,...): Gets the spectrum titles. '...' is passed down to spectrumSel.

classLabels(x): Gets and sets the class labels in phenoData. Used for summarization, see also estimateRatio and phenoData.

Methods

In the following code snippets, x is a IBSpectra object.

subsetIBSpectra(x, protein=NULL, peptide=NULL, direction="exclude",specificity): Get a 'subset' of IBSpectra: include or exclude proteins or peptides. When selection is based on proteins, it can be defined to exclude only peptides which are specific to the protein ('reporter-specific'), specific to the group ('group-specific') or which are shared with other proteins ('unspecific'). See subsetIBSpectra.

spectrumSel(x,peptide,protein,specificity="reporter-specific"): Gets a boolean vector selecting the corresponding spectra: If peptide is given, all spectra assigned to this peptide. If protein is given, all spectra assigned to peptides of this protein with specificity 'specificity'. See also ProteinGroup.

Author(s)

Florian P. Breitwieser

See Also

ProteinGroup, isobar-preprocessing, isobar-analysis, isobar-plots

Examples

```
data(ibspiked_set1)
ibspiked_set1
head(reporterIntensities(ibspiked_set1))
head(reporterMasses(ibspiked_set1))
proteinGroup(ibspiked_set1)
isotopeImpurities(ibspiked_set1)

# create new object
set.seed(123)
data <- data.frame(spectrum=letters,
                   peptide=sample(c("pepA","pepB","pepC"),26,TRUE),
                   start.pos=1,
                   modif=sample(c("::X:::",":Y::::","::Z:::"),26,TRUE),
                   accession=c("protein1","protein2"))
data.ions <- matrix(rnorm(26*2,1000,50),
                    ncol=2,dimnames=list(letters,NULL))
data.mass <- matrix(rep(c(126.1,127.1),26),
                    ncol=2,byrow=TRUE,dimnames=list(letters,NULL))
ib <- new("TMT2plexSpectra",data,data.ions,data.mass)
ib
reporterIntensities(ib)
isotopeImpurities(ib) <- matrix(c(0.8,0.1,0.2,0.9),nrow=2)
reporterIntensities(correctIsotopeImpurities(ib))
```

IBSpectra.log *Log functions for IBSpectra objects*

Description

The slot log of IBSpectra objects contains a matrix with two columns which contain a timestamp and message. Rownames relate to the item logged.

Used by correctIsotopeImpurities and normalize.

Usage

```
do.log(x, name, msg)

get.log(x, name)

is.logged(x, name)
```

Arguments

x	IBSpectra object
name	Name of property to be logged (translates to row name).
msg	Message to be logged for name.

Details

A warning message will be displayed if a already logged property is logged again.

Value

do.log: IBSpectra object with updated log. get.log:

Author(s)

Florian P Breitwieser

See Also

IBSpectra-class

Examples

```
data(ibspiked_set1)
ib <- normalize(correctIsotopeImpurities(ibspiked_set1))
ib@log
```

Isobar util functions *Isobar util functions*

Description

Utility functions. paste0 as a shorthand to paste(...,sep="") in versions of R pre 2.14.

Usage

```
paste0(..., sep = "")
a %inrange% b
```

Arguments

...	Arguments to paste.
sep	Separator.
a	values.
b	range.

Author(s)

Florian P Breitwieser

Examples

```
1:10
```

isobar-analysis *IBSpectra analysis: Protein and peptide ratio calculation*

Description

Calculates the relative abundance of a peptide or protein in one tag compared to another.

Usage

```
estimateRatio(ibspectra, noise.model = NULL, channel1, channel2, protein, peptide, ...)
estimateRatioForPeptide(peptide, ibspectra, noise.model, channel1, channel2, combine = TRUE, ...)
estimateRatioForProtein(protein, ibspectra, noise.model, channel1, channel2, combine = TRUE, method = "isob

## S4 method for signature 'numeric,numeric,missing'
estimateRatioNumeric(channel1,channel2,summarize.f=median, ...)

## S4 method for signature 'numeric,numeric,NoiseModel'
estimateRatioNumeric(channel1,channel2,noise.model,ratiodistr=NULL,variance.function="maxi",
                     sign.level=0.05,sign.level.rat=sign.level,sign.level.sample=sign.level,
                     remove.outliers=TRUE,outliers.args=list(method = "iqr", outliers.coef = 1.5),
                          n.sample=NULL,method="isobar",fc.threshold=1.3,
                     channel1.raw=NULL,channel2.raw=NULL,use.na=FALSE,preweights=NULL)

## S4 method for signature 'IBSpectra,ANY,character,character,character,missing'
estimateRatio(ibspectra,noise.model,channel1,channel2,
                                                    protein,peptide,...)

## S4 method for signature 'IBSpectra,ANY,character,character,character,NULL'
estimateRatio(ibspectra,noise.model,channel1,channel2,
                                                  protein,peptide=NULL,...)

## S4 method for signature 'IBSpectra,ANY,character,character,missing,character'
estimateRatio(ibspectra,noise.model,channel1,channel2,protein,peptide,...)
## S4 method for signature 'IBSpectra,ANY,character,character,NULL,character'
estimateRatio(ibspectra,noise.model,channel1,channel2,protein=NULL,peptide,...)
```

Arguments

ibspectra	IBSpectra object.
noise.model	NoiseModel object.
channel1	Tag channel 1. Can either be a character denoting a 'reporter name' or a numeric vector whose value should be summarized.Ratio is calculated as channel2/channel1.
channel2	Tag channel 2. Can either be a character denoting a 'reporter name' or a numeric vector whose value should be summarized. Ratio is calculated as channel2/channel1.
protein	Protein(s) of interest. If present, channel1 and channel2 must be reporter names. Provide either proteins or peptides.

peptide	Peptide(s) of interest. If present, channel1 and channel2 must be reporter names. Provide either proteins or peptides.
combine	If true, a single ratio is returned even for multiple peptides/spectra. If false, a data.frame with a row for each peptide/protein is returned.
specificity	See specificities.
quant.w.grouppeptides	Proteins which should be quantified with group specific peptides. Normally, only reporter specific peptides are used.
ratiodistr	distr object of ratio distribution.
variance.function	Defines how the variance for ratio is calculated. 'ev' is the estimator variance and thus 1/sum(1/variances). 'wsv' is the weighted sample variance. 'maxi' method takes the maximum of the former two variances.
sign.level	Significiance level.
sign.level.rat	Signal p-value significiance level.
sign.level.sample	Sample p-value significiance level.
remove.outliers	Should outliers be removed?
outliers.args	Arguments for outlier removal, see OUTLIERS function (TODO).
n.sample	For testing purposes: Only take a subset (sample) of the data.
method	method taken for ratio computation and selection: one of 'isobar','libra','multiq','pep','ttest' and 'compare.all'.
fc.threshold	When method equals fc, takes this as fold change threshold.
summarize.f	A method for summarizing spectrum ratios when no other information is available. For example median or mean.
channel1.raw	When given, noise estimation is based on channel1.raw and channel2.raw. These are the intensities of the channels before normalization.
channel2.raw	See channel1.raw.
use.na	Use NA values to calculate ratio. Experimental feature - use with caution.
preweights	Specifies weigths for each spectrum. Experimental feature - use with caution.
...	Passed down to estimateRatioNumeric methods.

Value

In general, a named character vector with the following elements: - lratio: log ratio - variance - n.spectra: number of spectra available in the ratio calculation - p.value.rat: Signal p-value. NA if called w/o ratiodistr - p.value.sample: Sample p-value. NA if called w/o ratiodistr - is.significant: NA if called w/o ratiodistr

If combine=FALSE, estimateRatio returns a data.frame, with columns as described above.

Author(s)

Florian P. Breitwieser, Jacques Colinge

See Also

ProteinGroup, IBSpectra, isobar-preprocessing, isobar-plots proteinRatios

Examples

```
data(ibspiked_set1)
data(noise.model.hcd)
ceru.human <- protein.g(proteinGroup(ibspiked_set1),"CERU_HUMAN")
ceru.rat <- protein.g(proteinGroup(ibspiked_set1),"CERU_RAT")
ceru.mouse <- protein.g(proteinGroup(ibspiked_set1),"CERU_MOUSE")
ceru.proteins <- c(ceru.human,ceru.rat,ceru.mouse)

## Calculate ratio based on all spectra of peptides specific
##  to CERU_HUMAN, CERU_RAT or CERU_MOUSE. Returns a named
##  numeric vector.
10^estimateRatio(ibspiked_set1,noise.model.hcd,
            channel1="114",channel2="115",
            protein=ceru.proteins)['lratio']

## If argument 'combine=FALSE', estimateRatio returns a data.frame
##  with one row per protein
10^estimateRatio(ibspiked_set1,noise.model.hcd,
            channel1="114",channel2="115",
            protein=ceru.proteins,combine=FALSE)[,'lratio']
## spiked material channel 115 vs 114:
##             CERU_HUMAN (P00450): 1
##             CERU_RAT   (P13635): 2
##             CERU_MOUSE (Q61147): 0.5
```

---

isobar-import *Loading data into IBSpectra objects using readIBSpectra*

Description

Read ibspectra-csv files and peaklist files as an IBSpectra object of type 'type' (see IBSpectra, e.g. iTRAQ4plexSpectra or TMT6plexSpectra). If peaklist.file is missing, it is assumed that id.file contains intensity and m/z columns for the reporter tags.

Usage

```
## S4 method for signature 'character,character'
readIBSpectra(type,id.file)
## S4 method for signature 'character,character,character'
readIBSpectra(
            type, id.file,peaklist.file,
            proteinGroupTemplate = NULL,
            mapping.file = NULL, mapping = c(peaklist="even",id="odd"),
            id.file.domap = NULL, annotate.spectra.f = NULL,
            peaklist.format = NULL, id.format = NULL,
            fragment.precision = NULL,fragment.outlier.prob = NULL,
            decode.titles = TRUE, scan.lines = 0, ...)
```

Arguments

type	Name of class of new IBSpectra object: iTRAQ4plexSpectra, iTRAQ8plexSpectra, TMT2plexSpectra, or TMT6plexSpectra
id.file	Database search results file in ibspectra.csv or mzIdentML format. See id.format. See the vignette for information on converting Mascot dat and Phenyx pidres files into ibspectra format.
peaklist.file	Peaklist file, typically in MGF format, see peaklist.format. MGF must be centroid!
proteinGroupTemplate	When having technical or biological repeats: First a template protein group is created which uses information from all runs, then this template is applied. It should increase comparability across runs.
mapping.file	If defined, spectum titles from the peaklist file are linked to the identifications via this file. This can be used when running HCD runs for quantification and CID runs for identification. See Koecher et al., 2009 for details.
mapping	Named character vector defining the names of columns in mapping.file. The names must be 'peaklist' and 'id', and the values must correspond to colnames of the mapping files.
id.file.domap	When using HCD-CID or a method akin and every spectrum is used for identification, the ID result files of the HCD run can be specfied in id.file.domap. Then, the results are merged after mapping the identification results.
annotate.spectra.f	Function which has the chance to annotate the spectra feature data before it is written to IBpectra object.
peaklist.format	"mgf" (Mascot Generic format) or "mcn" (iTracker Machine Readable output). When NULL, it detects the format on file name extension.
id.format	"ibspectra.csv" or "mzid" (PSI MzIdentML format). When NULL, file format is guessed based on extension.
fragment.precision	Fragment precision for extraction of reporter tags: for each tag and spectrum the m/z-intensity pair with it's mass closest to the known reporter tag mass is extracted within the window true_mass +/- fragment.precision/2.
fragment.outlier.prob	Fragment outlier probability filter: After all m/z-intensity pairs have been extracted, those pairs with the fragment.outlier.prob/2 most unprecise m/z values are filtered out.
decode.titles	Boolean. Decode spectrum titles in identification file using URLdecode. When extracting the DAT file from Mascot web interface, the spectrum titles are encoded - %20 instead of space, etc. Set decode.titles to TRUE to map these titles to the unescaped MGF titles.
scan.lines	Read files sequentially scan.lines lines at a time. Can help in case of memory issues, set to 10000 or higher, for example.
...	Further arguments handed down to initialize.

Author(s)

Florian P. Breitwieser, Jacques Colinge

See Also

ProteinGroup, IBSpectra, isobar-preprocessing, isobar-analysis, isobar-plots

Examples

```
data(ibspiked_set1)

# get identifier for Ceruplasmin proteins
ceru.acs <- protein.g(proteinGroup(ibspiked_set1),"CERU")
# create a smaller ibspectra w/ only Ceruplasmins
ib.ceru <- subsetIBSpectra(ibspiked_set1,protein=ceru.acs,"include")

# write it to a file
tf <- tempfile("isobar")
write.table(as.data.frame(ib.ceru),sep="\t",file=tf)

# read it again into an IBSpectra object
ib.ceru2 <- readIBSpectra("iTRAQ4plexSpectra",tf,id.format="ibspectra.csv")
ib.ceru2

unlink(tf)
```

---

isobar-plots *IBSpectra plots*

Description

Various plots are implement to assure data quality, and accompany preprocessing and analysis.

reporterMassPrecision

reporterMassPrecision(x)**:** Calculates and displays the deviation from the 'true' tag mass - as specified in the IBSpectra object - of each channel.

reporterIntensityPlot

reporterIntensityPlot(x)**:** Displays boxplots of intensity of channels before and after normalization - useful to check the result of normalization.

raplot

raplot(x,...)**:** Ratio-Absolute intensity plot - will be deprecated by maplot

x IBSpectra object

... Parameters to plot function.

plotRatio

plotRatio(x,channel1,channel2,protein,...)**:** Plots abundances of one protein

x IBSpectra object

channel1

channel2

protein

... Parameters to plot function.

maplot

maplot(x,channel1,channel2,...): Creates a ratio-versus-intensity plot.

x IBSpectra object.

maplot2

maplot2():

Author(s)

Florian P. Breitwieser, Jacques Colinge

See Also

IBSpectra, isobar-preprocessing isobar-analysis

Examples

```
data(ibspiked_set1)
maplot(ibspiked_set1,main="IBSpiked, not normalized")
maplot(normalize(ibspiked_set1),main="IBSpiked, normalized")
```

---

isobar-preprocessing *IBSpectra preprocessing*

Description

Preprocessing is a necessary step prior to analysis of data. In a sequential order, it is often neccassary to correct isotope impurities, to normalize, and subtract additive noise.

Isotope impurity correction

correctIsotopeImpurities(x): Returns impurity corrected IBSpectra object by solving a linear system of equations. See also isotopeImpurities.

Normalization

normalize(x,f=median,target="intensity",exclude.protein=NULL, use.protein=NULL,f.doapply=TRUE,log=
Normalizes the intensities for multiplicative errors. Those changes are most likely produced by pipetting errors, and different hybridization efficencies, but can also be due to biological reasons. By default, tag intensities are multiplied by a factor so that the median intensity is equal across tags.

f: f is applied to each column, unless f.doapply is FALSE. Then f is supposed to compute column-wise statistics of the matrix of intensities. E.g. colSums and colMeans.

target: One of "intensity" and "ratio".

exclude.proteins Spectra of peptides which might come from these proteins are excluded. Use for example for contaminants and proteins depleted in the experiment.

use.protein: If specified, only spectra coming from this protein are used. Use when a protein is spiked-in as normalization control.

f.isglobal: If true, f is applied on each column. If false, f is supposed to compute column-wise statistics of the matrix of intensities. E.g. colSums and colMeans.

log: Used when target=ratio.

Substract additive noise

subtractAdditiveNoise(x,method="quantile",shared=TRUE,prob=0.01): method 'quantile' method is supported for now. It take's the prob (0.01) quantile to estimate the noise level. This value is subtracted from all intensities, and all remaining intensities have to be at least that value.

prob See 'method'.

shared If channels are assumed similar in intensity and hence a shared noise level is reasonable. If not, then one level per channel is necessary.

Exclusion of proteins

exclude(x,proteins.to.exclude): Removes spectra which are assigned to proteins in protein.to.exclude from the object. This can be useful to remove contaminants. It create a new grouping based on the data which is left.

proteins.to.exclude Proteins to exclude.

Author(s)

Florian P. Breitwieser, Jacques Colinge

See Also

ProteinGroup, IBSpectra, isobar-analysis, isobar-plots

Examples

```
data(ibspiked_set1)
maplot(ibspiked_set1,main="IBSpiked, not normalized")
maplot(normalize(ibspiked_set1),main="IBSpiked, normalized")
```

---

isobar-reports *Isobar reports*

Description

Generation of LaTeX and XLS reports is helped with functions which facilitate the gathering of relevant information and creation of tikz plots. create.reports parses properties (by calling load.properties) and initialize environments and computations (by calling initialize.env) required by the reports, calls Sweave and pdflatex.

Usage

```
create.reports(properties.file = "properties.R", args = NULL,
               report.type = "protein",
               compile = FALSE, zip = FALSE,
               warn = 1)

load.properties(properties.file = "properties.R",
                global.properties.file = system.file("report","properties.R",package="isobar"),
                args = NULL)

initialize.env(env, report.type = "protein", properties.env)
```

Arguments

properties.file	File which holds the parameters for data analysis and report generation. It is parsed as R code after the global report configuration file global.properties.file and defines peaklists, identification files, significance levels, etc. See the global properties file for the available options and values.
global.properties.file	system.file("report","properties.R",package="isobar")
args	Additional (command line) arguments which overrids those in properties.file.
report.type	Currently, only protein is implemented.
compile	Compile LaTeX source to PDF? Requires pdflatex to be present. R CMD pdflatex will be executed twice on the Sweave result tex file.
zip	If true, tex, xls, and pdf files of all created reports and the properties.file are archived in a file named name.zip (name as defined as property) using zip.
warn	Warning level, see options.
env	Item to be initialized.
properties.env	Environment into which properties are read.

Details

The directory inst in the isobar installation directory system.file("inst",package="isobar") contains R, Sweave, and LaTeX files as examples of how to create XLS and PDF reports using isobar.

create_reports.R Call with Rscript. It is the main file which

1. parses command line options. --compile and --zip are parsed directly and given as arguments to create.reports. Other arguments are given load.properties.
2. calls a perl script to generate a XLS report
3. generates a LaTeX quality control and analysis report

for the XLS report the script pl/tab2xls.pl is used, which concetenates CSV files to a XLS. See Perl requirements. Sweave is called on report/isobar-qc.Rnw and report/isobar-analysis.Rnw. All files are written the working directory.

isobar-qc.Rnw Quality control Sweave file.

isobar-analysis.Rnw Data analysis Sweave file.

properties.R Default configuration for data analysis.

report-utils.tex LaTeX functions for plotting tikz graphics, etc.

Author(s)

Florian P Breitwieser

See Also

IBSpectra, isobar-preprocessing isobar-analysis

---

isobar.data *Isobar Data packages*

Description

ibspiked_set1 and ibspiked_set2 are objects of class iTRAQ4plexSpectra. It contains over 160 protein groups, over 1600 peptides from about 15,000 spectra each, mainly from background proteins and three spiked-in Ceruplasmins (CERU_HUMAN, CERU_MOUSE, CERU_RAT).

Usage

```
data(ibspiked_set1)
```

Format

iTRAQ4plexSpectra objects.

Source

isobar publication. Acquired on Orbitrap instrument w/ 20 offline-fractions and HCD fragmentation.

Examples

```
data(ibspiked_set1)
print(ibspiked_set1)
```

---

maplot.protein *Ratio intensity plot for individual proteins*

Description

Plots ratio-versus-intensity for a selected protein against a reference channel.

Usage

```
maplot.protein(x, relative.to, protein, noise.model = NULL, channels = NULL,
               xlim = NULL, ylim = NULL, identify = FALSE, add = FALSE,
               pchs = NULL, log="xy", legend.pos = "topright", names = NULL,
               legend.cex = 0.8, cols = pchs, ltys = NULL, main = protein,
               xlab = NULL, ylab = NULL, type="ma", ...)
```

Arguments

x	IBSpectra object
relative.to	a character vector specifying reporter tag names. Either of length 1 or same length as channels.
protein	Protein group identifier.
noise.model	NoiseModel object.
channels	Reporter tag names.
xlim	See par.
ylim	See par.
identify	boolean. If true, identify is called with peptide labels.
add	
pchs	a vector of the same length as channels. See pch in plot.default.
log	a character string which contains x if the x axis is to be logarithmic, y if the y axis is to be logarithmic and xy or yx if both axes are to be logarithmic.
legend.pos	see pos in legend.
names	a character string of the same length as channels, legend text.
legend.cex	see cex in legend.
cols	a vector of the same length as channels. See col in plot.default.
ltys	a vector of the same length as channels. See lty in plot.default.
main	a main title for the plot
xlab	a label for the x axis, defaults to a description of x.
ylab	a label for the y axis, defaults to a description of y.
type	type of plot
...	passed to plot.

Author(s)

Florian P. Breitwieser

NoiseModel-class *NoiseModel objects*

Description

A NoiseModel represent the technical variation which is dependent on signal intensity.

Constructor

new(type,ibspectra,reporterTagNames=NULL,one.to.one=TRUE,min.spectra=10,plot=FALSE, pool=FALSE
Creates a new NoiseModel object based on ibspectra object.

type: A non-virtual class deriving from NoiseModel: ExponentialNoiseModel, ExponentialNoANoiseModel, InverseNoiseModel, InverseNoANoiseModel

reporterTagNames: When NULL, all channels from ibspectra are taken (i.e. sampleNames(ibspectra)). Otherwise, specify subset of names

one.to.one: Set to false to learn noise model one a non one-to-one dataset

min.spectra: When one.to.one=FALSE, only take proteins with min.spectra to learn noise model.

plot: Set to true to plot data the noise model is learnt on.

pool: If false, a NoiseModel is estimated on each combination of channels indivdually, and then the parameters are averaged. If true, the ratios of all channels are pooled and then a NoiseModel is estimated.

Accessor methods

noiseFunction: Gets the noise function.

parameter: Gets and sets the parameters for the noise function.

variance: Gets the variance for data points based on the noise function and parameters.

stddev: Convenience function, sqrt(variance(...)).

lowIntensity: Gets and sets the low intensity slot, denoting the noise region.

naRegion: Gets and sets the na.region slot.

Examples

```
data(ibspiked_set1)

ceru.proteins <- protein.g(proteinGroup(ibspiked_set1),"CERU")

# normalize
ibspiked_set1 <- normalize(correctIsotopeImpurities(ibspiked_set1))

# remove spiked proteins
ibspiked_set1.noceru <- exclude(ibspiked_set1,ceru.proteins)
ibspiked_set1.justceru <- subsetIBSpectra(ibspiked_set1,protein=ceru.proteins,direction="include")

# learn noise models
nm.i <- new("InverseNoiseModel",ibspiked_set1.noceru)
nm.e <- new("ExponentialNoiseModel",ibspiked_set1.noceru)

#learn on non-one.to.one data: not normalized, with spiked proteins
nm.n <- new("ExponentialNoiseModel",ibspiked_set1.justceru,one.to.one=FALSE)

maplot(ibspiked_set1,noise.model=c(nm.e,nm.i,nm.n),ylim=c(0.1,10))
```

number.ranges *Helper function to transform number lists to ranges*

Description

1,2,3,4,5,8,9,10 -> 1-5,8-10

Usage

```
number.ranges(numbers)
```

Arguments

numbers	numeric

Value

character

Author(s)

Florian P Breitwieser

Examples

```
number.ranges(c(1,2,3,9,3,10,8,11))
```

peptide.count *Peptide counts, spectral counts and sequence coverage for ProteinGroup objects.*

Description

Report the peptide count, spectral count and sequence coverage for supplied proteins.

Usage

```
peptide.count(protein.group, protein.g = reporterProteins(protein.group),
        specificity = c("reporter-specific", "group-specific", "unspecific"), ...)

spectra.count(protein.group, protein.g = reporterProteins(protein.group),
        specificity = c("reporter-specific", "group-specific", "unspecific"),
        modif = NULL, ...)

sequence.coverage(protein.group, protein.g = reporterProteins(protein.group),
            specificity = c("reporter-specific", "group-specific", "unspecific"),
            simplify = TRUE, ...)
```

Arguments

protein.group	ProteinGroup object.
protein.g	Protein group identifier.
specificity	Specificity of peptides.
modif	Only count peptides having a certain modification.
simplify	If simplify=TRUE, a named numeric vector is returned, with the mean sequence coverage of the ACs of each protein.g supplied. Else, a list with the length of protein.g is returned having the sequence coverage for each protein AC.
...	Further arguments to peptides

Author(s)

Florian P Breitwieser

See Also

calculate.emPAI, calculate.dNSAF, ProteinGroup

Examples

```
data(ibspiked_set1)
sc <- spectra.count(proteinGroup(ibspiked_set1))
pc <- peptide.count(proteinGroup(ibspiked_set1))
plot(jitter(sc),jitter(pc),log="xy")
```

ProteinGroup-class *ProteinGroup objects*

Description

The ProteinGroup class is a container for identified peptides and proteins, and groups them to distinguish proteins with specific peptides.

Usage

```
ProteinGroup(from,template=NULL,proteinInfo=data.frame())

protein.ac(x, protein.g)
protein.g(x, pattern, variables=c("AC","name"), ...)
```

Arguments

from	data.frame object to create a ProteinGroup from. See Details from column specifications
template	'template' ProteinGroup object for grouping.
x	ProteinGroup object
protein	character string
proteinInfo	data.frame for proteinInfo slot

protein.g	character string, denoting a 'protein group'.
pattern	character string, see grep for details.
variables	AC maps a protein accession code to a protein group. name maps using protein information from proteinInfo.
...	Passed on to grep.

Details

The ProteinGroup class stores spectrum to peptide to protein mapping.

The proteins are grouped by their evidence, i. e. peptides:

- Peptides with changes only from Leucin to Isoleucin are considered the same, as they cannot be distinguished by MS.
- Proteins which are detected with the same peptides are grouped together to a 'indistinguishable protein'- normally these are splice variants.
- Proteins with specific peptides are 'reporters'.
- Proteins with no specific peptides are grouped under these 'reporters.

This information is stored in six slots:

spectra.n.peptides a named 'character' vector, names being spectrum identifier and values are peptides.

peptide.n.proteins a 'data.frame' containing the number of proteins the peptides could derive from.

peptide.n.protein a character 'matrix' linking peptides to proteins.

indistinguishable.proteins a 'matrix' contain.

Constructor

ProteinGroup(tbl.prot.pep,template=NULL): Creates a ProteinGroup object.

tbl.prot.pep A 'data.frame' with three columns: 1. Protein, 2. Peptide, 3. Spectrum.

template Optional ProteinGroup object the grouping is based upon.

Coercion

In the code snippets below, x is a ProteinGroup object.

as(from, "ProteinGroup"): Creates a ProteinGroup object from a data.frame.

as.data.frame(x, row.names = NULL, optional = FALSE): Creates a data.frame with columns protein (character), peptide (character), spectrum.

Accessors

In the following code snippets, x is a ProteinGroup object.

spectrumToPeptide(x): Gets spectrum to peptide assignment.

peptideSpecificity(x): Gets a 'data.frame' containing the peptide specificity: they can be reporter-specific, group-specific, or non-specific.

peptideNProtein(x): Gets peptide to protein assignment.

indistinguishableProteins(x): Gets the proteins which cannot be distinguished based on peptide evidence.

proteinGroupTable: Gets the protein grouping, listing reporters and group members.

peptides(x,protein=NULL,specificity=c("reporter-specific", "group-specific","unspecific"),columns="peptide"
Gets all peptides detected, or just those for a protein with the defined specificity. columns might define multiple columns of peptideSpecificity(x). set=union returns the union of peptides of all proteins defined, set=intersect returns the intersection.

Author(s)

Florian P. Breitwieser

See Also

IBSpectra

Examples

```
tbl <- data.frame(spectrum=1:14,peptide=c(rep(letters[1:3],4),"a","x"),
                  modif=":",start.pos=1,
                  protein=c(rep(c("A","B"),each=6),"C","D"))
pg <- ProteinGroup(tbl)
pg
proteinGroupTable(pg)

data(ibspiked_set1)
pg <- proteinGroup(ibspiked_set1)
ceru.proteins <- protein.g(pg,"CERU")

## all ceru peptides
peptides(pg,ceru.proteins)

## peptides shared by all ceru proteins
peptides(pg,ceru.proteins, set=intersect)
```

---

proteinInfo-methods *Methods for Function proteinInfo*

Description

proteinInfo slot in Proteingroup objects contains information about proteins. proteinInfo method allows to get and set it.

getProteinInfoFromUniprot downloads information of contained proteins from Uniprot, getProteinInfoFromBiomart from Biomart.

Usage

```
  ## S4 method for signature 'ProteinGroup'
proteinInfo(x)

  ## S4 method for signature 'ProteinGroup,character,missing'
proteinInfo(x, protein.g, select="name", collapse=", ",
                                    simplify = TRUE, do.warn = TRUE)

  ## S4 method for signature 'ProteinGroup,missing,character'
proteinInfo(x, protein.ac, select="name", collapse=", ",
                                    simplify = TRUE, do.warn = TRUE)


  proteinInfoIsOnSpliceVariants(protein.info)

  getProteinInfoFromUniprot(x, splice.by = 200)

  getProteinInfoFromNextProt(x)

  getProteinInfoFromBiomart(x, database = "Uniprot")

  getProteinInfoFromBioDb(x, ..., con = NULL)

 getPtmInfoFromNextprot(protein.group, nextprot.url = "http://www.nextprot.org/rest/entry/NX_XXX/pt
```

Arguments

x	ProteinGroup object
protein.group	ProteinGroup object
protein.g	Protein group identifier. If supplied, only information for these proteins is returned.
protein.ac	Protein ACs. If supplied, only information for these proteins is returned.
select	indicating columns to select. See Details.
collapse	passed to paste to concatenate information of multiple protein in one protein group.
simplify	If true, a vector or matrix is returned, with the pasted protein information. If false, a list is returned.
do.warn	If true, report diagnostic warning messages.
splice.by	Chunk size for query of Uniprot database.
database	database from which the ACs stem from. Only Uniprot is supported for now.
con	database connection
...	arguments to build database connection.
protein.info	protein info data.frame
nextprot.url	URL for fetching Nextprot results. 'XXX' will be replaced by the Uniprot Protein AC.

Details

proteinInfo contains columns accession, name, gene_name, protein_name, and possibly length and sequence. accession is mapped with the entry AC is mapped to the entry AC in the database. getProteinInfoFromUniprot is the preferred methods to get the information. getProteinInfoFromBioDb is an example how to implement the query on a local database. Depending on the database, protein information might be available on protein ACs or also on the specific splice variants. This can be queried with the proteinInfoIsOnSpliceVariants function.

See Also

protein.g

Examples

```
data(ibspiked_set1)
pg <- proteinGroup(ibspiked_set1)

## Not run:
  proteinInfo(pg) <- getProteinInfoFromUniprot(pg)
  proteinInfo(pg) <- getProteinInfoFromBiomart(pg)

## End(Not run)

proteinInfo(pg,protein.g="P13635")
protein.g(pg,"CERU")
```

---

proteinNameAndDescription

Get protein gene names and description from protein info of protein group.

Description

Convenience functions to retrieve protein gene names and description for a list of protein group identifiers.

Usage

```
proteinNameAndDescription(protein.group, protein.g = reporterProteins(protein.group), collapse = FALSE)
proteinGeneName(protein.group, protein.g = reporterProteins(protein.group))
proteinDescription(protein.group, protein.g = reporterProteins(protein.group))
proteinID(protein.group, protein.g = reporterProteins(protein.group))
```

Arguments

protein.group	ProteinGroup object.
protein.g	protein group identifier.
collapse	If TRUE, the information for all protein.gs is combined.

Author(s)

Florian P Breitwieser

Examples

```
data(ibspiked_set1)
pg <- proteinGroup(ibspiked_set1)
protein.gs <- protein.g(pg,"CERU")
protein.gs
proteinNameAndDescription(pg,protein.gs)
proteinNameAndDescription(pg,protein.gs,collapse=TRUE)
proteinGeneName(pg,protein.gs)
proteinDescription(pg,protein.gs)
proteinID(pg,protein.gs)
```

---

ratiosReshapeWide *Reshape output of proteinRatios into wide format*

Description

Reshape output of proteinRatios into wide format

Usage

```
ratiosReshapeWide(quant.tbl, grouped.cols = TRUE,
                  vs.class = NULL, sep = ".", cmbn = NULL, short.names = FALSE)
```

Arguments

quant.tbl	Output of proteinRatios or peptideRatios.
grouped.cols	Whether the columns should be grouped next to each other.
vs.class	Only return ratios where class1 is vs.class
sep	Separator for column names in the reshape.
cmbn	Not functional.
short.names	If vs.class is set and short.names=TRUE, then the comparision name will be i.e. 'class2' instead of 'class2/class1'.

Author(s)

Florian P. Breitwieser

---

ratiosummarization *protein and peptide ratios*

Description

A set of functions to create ratios within groups and summarize them. proteinRatios serves as hub and calls combn.matrix, combn.protein.tbl and summarize.ratios successively. It can be used to calculate intra-class and inter-class ratios, to assess ratios and variability within and over cases.

Usage

proteinRatios(ibspectra, noise.model, reporterTagNames = NULL, proteins = reporterProteins(proteinGroup(ib
p.adjust = NULL, reverse=FALSE, combn=NULL, ...)

combn.matrix(x, method = "global", cl = NULL, vs = NULL)

combn.protein.tbl(ibspectra, noise.model, ratiodistr, proteins = NULL, cmbn, peptide = NULL, modif = NULL

summarize.ratios(ratios, summarize.method, min.detect, n.combination, strict.sample.pval = TRUE, strict.ratio

Arguments

ibspectra	IBSpectra object
x	for combn.matrix: reporter names. See reporterTagNames. argument of proteinRatios.
ratios	result of combn.protein.tbl
cmbn	result of combn.matrix
combn	result of combn.matrix
noise.model	NoiseModel for spectra variances
reporterTagNames	Reporter tags to use. By default all reporterTagNames of ibspectra object.
proteins	proteins for which ratios are calculated - defaults to all proteins with peptides specific to them.
peptide	peptides for which ratios are calculated.
modif	Modification.
cl	Class labels. See also ?classLabels.
vs	Class label or reporter tag name. When combn.method is "versus.class", all combinations against class vs are computed, when combn.method is "verus.channel", all combinations against channel vs.
combn.method	"global", "interclass", or "intra-class". Defines which ratios are computed, based on class labels cl
method	"global", "interclass", or "intra-class". Defines which ratios are computed, based on class labels cl
symmetry	If true, reports also the inverse ratio
summarize	If true, ratios for each protein are summarized.
summarize.method	"isobar", for now.
min.detect	How many times must a ratio for a protein be present when summarizing? When NULL, defaults to the maximum number of combinations.
strict.sample.pval	If true, missing ratios are penalized by giving them a sample.pval of 0.5.
strict.ratio.pval	If true, take all ratios into account. If false, only take ratios into account which are in the same direction as the majority of ratios
orient.div	Number of ratios which might go in the wrong direction.
sign.level	Significance level

sign.level.rat	Significance level on ratio p-value
sign.level.sample	Significance level on sample p-value
ratiodistr	Protein ratio distribution
variance.function	Variance function
...	Passed to estimateRatio()
combine	If true, a single ratio for all proteins and peptides, resp., is calculated. See estimateRatio.
p.adjust	Set to one of p.adjust.methods to adjust ratio p-values for multiple comparisions. See p.adjust.
reverse	reverse
n.combination	number of combinations possible

Value

'data.frame': 11 variables:

lratio	log ratio
variance	variance
n.spectra	Number of spectra used for quantification
p.value.rat	Signal p-value (NA if ratiodistr is missing)
p.value.sample	Sample p-value (NA if ratiodistr is missing)
is.significant	Is the ratio significant? (NA if ratiodistr is missing)
protein	Protein quantified
r1	r1
r2	r2

Author(s)

Florian P Breitwieser, Jacques Colinge

See Also

IBSpectra, isobar-preprocessing isobar-analysis

Examples

```
combn.matrix(114:117,method="interclass",cl=as.character(c(1,1,2,2)))
combn.matrix(114:117,method="interclass",cl=as.character(c(1,1,2,2)))
combn.matrix(114:117,method="global")

data(ibspiked_set1)
data(noise.model.hcd)

ceru.proteins <- c("P13635","Q61147")
proteinRatios(ibspiked_set1,noise.model=noise.model.hcd,proteins=ceru.proteins,cl=c("T","T","C","C"),combn.meth
```

sanitize *Helper function for LaTeX export*

Description

Sanitizes strings for LaTeX

Usage

```
sanitize(str, dash = TRUE)
```

Arguments

str	character string to be escaped
dash	shoud a dash ('-') should be escaped to a '\nobreakdash-'?

Value

escaped character

Author(s)

iQuantitator,Florian P Breitwieser

Examples

```
sanitize("\textbf{123-123}")
```

shared.ratios *Shared ratio calculation*

Description

Calculate ratios of reporter proteins and subset proteins with shared peptides.

Usage

```
shared.ratios(ibspectra, noise.model, channel1 , channel2 , protein = reporterProteins(proteinGroup(ibspectra))
```

Arguments

ibspectra	IBspectra object.
noise.model	NoiseModel object.
channel1	channel1 to compare.
channel2	channel2 to compare.
protein	proteins for which the calculation should be made.
...	Additional arguments passed to estimteRatio.

Value

data.frame

Author(s)

Florian P.\ Breitwieser

See Also

shared.ratios.sign

---

shared.ratios.sign *Plot and get significantly shared ratios.*

Description

Plot and get significantly shared ratios.

Usage

shared.ratios.sign(ress, z.shared, min.spectra = 1, plot = TRUE)

Arguments

ress	Result of shared.ratios.
z.shared	z.
min.spectra	Minimal number of spectra needed.
plot	plot.

Author(s)

Florian P.\ Breitwieser

See Also

shared.ratios.

---

specificities *Peptide specificities*

Description

Peptides can appear in multiple proteins and therefore have different specificities.

Details

reporter specific: peptides specific to reporter. group specific: peptides specific to the group. unspecific: peptides shared with other proteins.

spectra.count2 *Spectral count for peptides and proteins in ProteinGroup objects.*

Description

Spectral count for peptides and proteins in ProteinGroup objects. It can - other than spectra.count - quantify the spectra count on the level of peptides, potenitally modifed, too,

Usage

```
spectra.count2(ibspectra, value = reporterProteins(protein.group),
        type = "protein.g", specificity = c("reporter-specific", "group-specific", "unspecific"),
          modif = NULL, combine = FALSE, subset = NULL, require.quant = NULL, ...)
```

Arguments

ibspectra	IBSpectra object.
value	List of protein group identifiers or peptides.
type	Either 'protein.g' or 'peptide'.
specificity	Specificity of peptides.
modif	Only count peptides having a certain modification.
combine	If TRUE, only one combined result is returned.
subset	Allows to specify an expression to subset link{featureData} of the ibspectra.
require.quant	If not NULL, it may be 'any' or 'all' to only consider spectra with quantitative information in at least one or all channels.
...	Further arguments to peptides

Author(s)

Florian P Breitwieser

See Also

spectra.count, ProteinGroup

Examples

```
data(ibspiked_set1)
pg <- proteinGroup(ibspiked_set1)
protein.gs <- protein.g(pg,"CERU")
sc <- spectra.count2(ibspiked_set1,protein.gs)
sc.ik <- spectra.count2(ibspiked_set1,protein.gs,modif="iTRAQ4plex_K")
rbind(spectra.counts=sc,spectra.counts_iTRAQk=sc.ik)
```

subsetIBSpectra *Subset IBSpectra objects*

Description

Returns an IBSpectra object which is a subset of the input, excluding or exclusively containing the peptides or proteins supplied.

Usage

```
subsetIBSpectra(x, protein = NULL, peptide = NULL,
                direction = "exclude",
               specificity = c(REPORTERSPECIFIC, GROUPSPECIFIC, UNSPECIFIC), ...)
```

Arguments

x	IBSpectra object.
protein	Protein group identifiers. Use protein.g to get protein group identifiers from protein database ACs.
peptide	Peptide sequences.
direction	either 'include' or 'exclude'.
specificity	When 'protein' is supplied: Which peptides should be selected? See specificities.
...	Further arguments passed to spectrumSel

Author(s)

Florian P Breitwieser

See Also

protein.g, spectrumSel, specificities

Examples

```
data(ibspiked_set1)

# get Keratin proteins
keratin.proteins <- protein.g(proteinGroup(ibspiked_set1),"Keratin")

# exclude Keratin proteins
subsetIBSpectra(ibspiked_set1,protein=keratin.proteins,direction="exclude")
```

writeIBSpectra *Write IBSpectra file as CSV in a format readable by readIBSpectra.*

Description

Write IBSpectra file using write.table with defaults in a format readable by readIBSpectra.

Usage

```
writeIBSpectra(ibspectra, file, sep = "\t", row.names = FALSE, ...)
```

Arguments

ibspectra	IBSpectra object
file	file name.
sep	field separator string.
row.names	indicates whether row.names should be written.
...	further arguments to write.table

Author(s)

Florian P Breitwieser

Index