Package ‘isobar’

January 14, 2017

**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. Features modules for integrating and validating PTM-centric datasets (isobar-PTM). More information on http://www.ms-isobar.org.

**Version**  1.20.0

**Author**  Florian P Breitwieser <florian.bw@gmail.com> and Jacques Colinge <jacques.colinge@inserm.fr>, with contributions from Alexey Stukalov <stukalov@biochem.mpg.de>, Xavier Robin <xavier.robin@unige.ch> and Florent Gluck <florent.gluck@unige.ch>

**Maintainer**  Florian P Breitwieser <florian.bw@gmail.com>

**biocViews**  Proteomics, MassSpectrometry, Bioinformatics, MultipleComparisons, QualityControl

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

**Imports**  distr, plyr, biomaRt, ggplot2

**Suggests**  MSnbase, OrgMassSpecR, XML, RJJSONIO, Hmisc, gplots, RColorBrewer, gridExtra, limma, boot, DBI, MASS

**LazyLoad**  yes

**License**  LGPL-2

**URL**  https://github.com/fbreitwieser/isobar

**BugReports**  https://github.com/fbreitwieser/isobar/issues


**NeedsCompilation**  no

**R topics documented:**

- isobar-package ................................................................. 3
- calc.delta.score ........................................................... 4
R topics documented:
calcPeptidePosition .................................................. 5
calculate.pvalues .......................................................... 5
calculate.dNSAF ............................................................ 6
calculate.emPAI ............................................................ 7
correct.peptide.ratios .................................................... 8
distr-methods ............................................................... 9
fit distributions ........................................................... 10
getPeptideModifContext .................................................. 11
getPhosphoRSProbabilities ............................................. 11
getPtmInfo ................................................................. 13
groupMemberPeptides .................................................... 15
human.protein.names ..................................................... 16
IBSpectra-class ............................................................ 16
IBSpectra.log ............................................................... 19
Isobar util functions ...................................................... 20
isobar-analysis ............................................................. 20
isobar-import .............................................................. 23
isobar-plots ............................................................... 25
isobar-preprocessing ..................................................... 26
isobar-reports ............................................................. 27
isobar.data ................................................................. 29
maplot.protein ............................................................. 29
NoiseModel-class .......................................................... 30
number.ranges ............................................................ 32
observedKnownSites ...................................................... 32
peptide.count ............................................................. 33
Protein and peptide ratio calculation and summarization .... 34
ProteinGroup-class ......................................................... 37
proteinInfo-methods ...................................................... 39
proteinNameAndDescription ........................................... 41
ratiosReshapeWide ...................................................... 42
reporter.protein-methods ............................................... 42
sanitize ................................................................. 43
shared.ratios ............................................................. 43
shared.ratios.sign ......................................................... 44
specificities ............................................................. 44
spectra.count2 ........................................................... 45
subsetIBSpectra ........................................................... 46
Tlsd-class ................................................................. 47
TlsParameter-class ......................................................... 48
writeHscoreData ........................................................... 49
writeIBSpectra ............................................................ 49
Index ................................................................. 50
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 with OA isobaric tags, such as iTRAQ and TMT.

Details

Package: isobar
Version: 1.1.2
biocViews: Proteomics, MassSpectrometry, 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

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
calc.delta.score

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>

---

calc.delta.score Calculate Delta Score from Ion Score

Description

Calculates delta score from raw search engine score by subtracting the best matching hit with the second best matching. Data needs to have not only the best hit per spectrum, but multiple, to be able to calculate the delta score. filterSpectraDeltaScore calls calc.delta.score and filters spectra below a minum delta score.

Usage

calc.delta.score(my.data)
filterSpectraDeltaScore(my.data, min.delta.score=10, do.remove=FALSE)

Arguments

my.data IBSpectra data frame.
min.delta.score Minimum delta score.
do.remove If TRUE, spectra below the min.prob threshold are not just set as 'use.for.quant=FALSE' but removed.

Value

Returns data with additional column 'delta.score'.

Author(s)

Florian P. Breitwieser
**calcPeptidePosition**

Recalculate peptide start positions based on protein sequence

**Description**

Function to recalculate start position of peptide in protein when it is missing or wrong.

**Usage**

```
calcPeptidePosition(peptide.info, protein.info, calc.il.peptide)
```

**Arguments**

- **peptide.info**: Peptide info object of ProteinGroup.
- **protein.info**: Protein info object of ProteinGroup.
- **calc.il.peptide**: Should the 'real' peptide (I/L difference) be calculated?

**calculate-pvalues**

Calculate and Adjust Ratio and Sample p-values.

**Description**

Functions for calculating and adjusting ratios and sample p-values. Usually, these are called by proteinRatios or peptideRatios.

**Usage**

```
calculate.ratio.pvalue(lratio, variance, ratiodistr = NULL)
calculate.sample.pvalue(lratio, ratiodistr)
calculate.mult.sample.pvalue(lratio, ratiodistr, strict.pval, lower.tail, n.possible.val, n.observed.val)
adjust.ratio.pvalue(quant.tbl, p.adjust, sign.level, globally = FALSE)
```

**Arguments**

- **lratio**: log 10 protein or peptide ratios.
- **ratiodistr**: Fitted ratio distribution/
- **variance**: Variance of lratios.
- **strict.pval**: If FALSE, missing ratios are ignored. If TRUE, missing ratios are penalized by giving them a sample.pval of 0.5.
- **lower.tail**: lower.tail of distribution?
- **n.possible.val**: Number of possible ratios.
- **n.observed.val**: Number of observed ratios.
- **quant.tbl**: Quantification table (from proteinRatios or peptideRatios).
**p.adjust**  
P-value adjustment method (see ?p.adjust).

**sign.level**  
Ratio significance level.

**globally**  
Whether the p-values should be adjusted over all conditions, or individually in each condition.

### Author(s)

Florian P. Breitwieser

### See Also

*proteinRatios, peptideRatios*

### Examples

```rlratio <- c(-1,-1,seq(from=-1,to=1,by=.25),1,1)
variance <- c(0,1,rep(0.1,9),0,1)
ratiodistr.precise <- new("Norm",mean=0,sd=.25)
ratiodistr.wide <- new("Norm",mean=0,sd=.5)

data.frame(lratio, variance, 
  ratio.pvalue=calculate.ratio.pvalue(lratio, variance), 
  sample.pvalue.precise=calculate.sample.pvalue(lratio, ratiodistr.precise), 
  sample.pvalue.wide=calculate.sample.pvalue(lratio, ratiodistr.wide))
```

---

### 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

```r
calculate.dNSAF(protein.group, use.mw = FALSE, normalize = TRUE, 
  combine.f = mean)
```

### Arguments

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

**use.mw**  
Use MW to account for protein size

**normalize**  
Normalize dSAF to dNSAF?

**combine.f**  
How to handle proteins seen only with shared peptides?
calculate.emPAI

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), normalize = FALSE, observed.pep = c("pep", "mod.charge.pep"), use.mw = FALSE, combine.f = mean, ..., nmc = 0, report.all = FALSE)
n.observable.peptides(...)
observable.peptides(seq, nmc = 1, min.length = 6, min.mass = 600, max.mass = 4000, custom = list(code = c("B", "Z", "J", "U"), mass = c(164.554862, 278.61037, 213.12392, 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.
normalize Normalize to sum = 1?.
observed.pep What counts as observed peptide?
report.all TOADD
use.mw Use MW to normalize for protein size
combine.f How to handle proteins seen only with shared peptides?
### correct.peptide.ratios

**Sequence**

<table>
<thead>
<tr>
<th>seq</th>
<th>Protein sequence.</th>
</tr>
</thead>
<tbody>
<tr>
<td>nmc</td>
<td>Number of missed cleavages.</td>
</tr>
<tr>
<td>min.length</td>
<td>Minimum length of peptide.</td>
</tr>
<tr>
<td>min.mass</td>
<td>Minimum mass of peptide.</td>
</tr>
<tr>
<td>max.mass</td>
<td>Maximum mass of peptide.</td>
</tr>
<tr>
<td>custom</td>
<td>User defined residue for Digest.</td>
</tr>
</tbody>
</table>

... Further arguments to observable.peptides/Digest.

### Details

The formula is

\[
em\text{PAI} = 10^{\frac{N_{\text{observed}}}{N_{\text{observable}}}} - 1
\]

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


### See Also

Digest, proteinInfo, getProteinInfoFromUniprot, calculate.dNSAF, ProteinGroup

### Examples

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

---

### correct.peptide.ratios

Correct peptide ratios with protein ratios from a separate experiment.

### Description

Correct peptide ratios with protein ratios from a separate experiment.

### Usage

```r
correct.peptide.ratios(ibspectra, peptide.quant.tbl, protein.quant.tbl, protein.group.combined, adjust.variance = TRUE, correlation = 0, recalculate.pvalue = TRUE)
```
Arguments

- **ibspectra** IBSpectra object.
- **peptide.quant.tbl** Calculated with peptideRatios.
- **protein.quant.tbl** Calculated with proteinRatios.
- **protein.group.combined** ProteinGroup object generated on both PTM and protein data.
- **adjust.variance** Adjust variance of ratios.
- **correlation** Assumed correlation between peptide and protein ratios for variance adjustment.
- **recalculate.pvalue** Recalculate p-value after variance adjustment.

Author(s)

Florian P. Breitwieser

distr-methods

Functions for distribution calculations

description

calcProbXGreaterThanY calculates the probability that X >= Y. calcProbXDiffNormals calculates the probabilities of a set of normals, defined by the vectors mu_Y and sd_Y are greater or less than the reference distribution Y.

Usage

```r
calcProbXGreaterThanY(X, Y, rel.tol = .Machine$double.eps^0.25, subdivisions = 100L)
calcProbXDiffNormals(X, mu_Y, sd_Y, ..., alternative = c("greater", "less", "two-sided"), progress = FALSE)
#calcCumulativeProbXGreaterThanY(Xs, mu_Ys, sd_Ys, alternative = c("greater", "less", "two-sided"), rel.tol = .Machine$double.eps^0.25, subdivisions = 100L)
distrprint(X, round.digits = 5)
twodistr.plot(X, Y, n.steps = 1000, min.q = 10^-3)
```

Arguments

- **X** Object of the class Distribution.
- **Y** Object of the class Distribution.
- **min.q** minimum quantile
- **n.steps** Number of steps.
- **mu_Y** Numeric vector of parameter mu of a Normal.
- **sd_Y** Numeric vector of parameter sd of a Normal.
- **subdivisions** the maximum number of subintervals
- **rel.tol** relative accuracy requested
- **...** Additional arguments to calcProbXGreaterThanY.
- **alternative** "less", "greater", or "two-sided".
- **progress** Show text progress bar?
- **round.digits** Round digits for printing.
fit distributions

Author(s)

Florian P. Breitwieser

Examples

library(distr)
calcProbXGreaterThanY(Norm(0,.25),Norm(1,.25))

Description

Functions to fit the probability density functions on ratio distribution.

Usage

fitCauchy(x)
fitNorm(x, portion = 0.75)
fitWeightedNorm(x, weights)
fitNormalCauchyMixture(x)
fitGaussianMixture(x, n = 500)
fitTlsd(x)

Arguments

x Ratios
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

```r
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)
```

getPeptideModifContext

*Get context of modification*

**Description**

Gets neighboring amino acids around modification which can be used to find enriched motifs.

**Usage**

```r
getPeptideModifContext(protein.group, modif, n.aa.up = 7, n.aa.down = 7)
```

**Arguments**

- **protein.group**  ProteinGroup object.
- **modif**          Modification of interest.
- **n.aa.up**        Number of AA downstream to report.
- **n.aa.down**      Number of AA upstream to report.

getPhosphoRSProbabilities

*Generate input files for PhosphoRS, call it, and get modification site probabilities*

**Description**

Get phosphorylation site localization probabilities by calling PhosphoRS and parsing its output. `getPhosphoRSProbabilities` generates a XML input file for PhosphoRS calling `writePhosphoRSInput`, then executes phosphoRS.jar with java, and parses the XML result file with `readPhosphoRSOutput`. 

```r
```
getPhosphoRSProbabilities

Usage

getPhosphoRSProbabilities(id.file, mgf.file, massTolerance, activationType,
    simplify = FALSE, mapping.file = NULL, mapping =
    c(peaklist = "even", id = "odd"), pepmodif.sep =
    "##.##", besthit.only = TRUE, phosphors.cmd =
    paste("java -jar", system.file("phosphors",
        "phosphoRS.jar", package = "isobar")), file.basename =
    tempfile("phosphors."))

writePhosphoRSInput(phosphoRS.infile, id.file, mgf.file, massTolerance,
    activationType, mapping.file = NULL, mapping =
    c(peaklist = "even", id = "odd"), pepmodif.sep =
    "##.##", modif.masses = rbind(c("PHOS", "1",
    c("Oxidation_M", "2",
        "2:Oxidation:Oxidation:15.994919:null:0:M"),
    c("Cys_CAM", "3",
        "3:Carbamidomethylation:Carbamidomethylation:57.021464:null:0:C"),
    c("iTRAQ4plex", "4",
        "4:iTRAQ4:iTRAQ4:144.1544:null:0:KX"), c("iTRAQ8plex",
        "5", "5:iTRAQ8:iTRAQ8:304.308:null:0:KX"),
    c("TMT6plex", "7",

readPhosphoRSOutput(phosphoRS.outfile, simplify = FALSE, pepmodif.sep = "##.##", besthit.only =
    TRUE)

filterSpectraPhosphoRS(id.file, mgf.file, ..., min.prob = NULL, do.remove=FALSE)

Arguments

id.file Database search results file in ibspectra.csv or mzIdentML format. See IBSSpectra
    and isobar vignette for information on converting Mascot dat and Phenyx pidres
    files into ibspectra format.

mgf.file Peaklist file

massTolerance Fragment ion mass tolerance (in Da)

activationType Activation types of spectra. CID, HCD, or ETD.

simplify If TRUE, returns a data.frame instead of a list.

mapping.file Mapping file. See also readIBSpectra.

mapping Mapping columns.

besthit.only Only show best hit, simplifies result to data.frame instead of list.

phosphors.cmd PhosphoRS script.

file.basename Base name for creating phosphoRS input and output files.

phosphoRS.infile PhosphoRS input XML file name.

phosphoRS.outfile PhosphoRS output XML file name.
**getPtmInfo**

pepmodif.sep separator of peptide and modification in XML id
modif.masses masses and ID used for PhosphoRS
min.prob Threshold for PhosphoRS peptide probability to consider it for quantification
... Further arguments to getPhosphoRSProbabilities
do.remove If TRUE, spectra below the min.prob threshold are not just set as 'use.for.quant=FALSE' but removed.

**Details**

PhosphoRS is described in Taus et al., 2011. It can be downloaded from http://cores.imp.ac.at/protein-chemistry/download/ and used as Freeware. Java is required at runtime.

**Value**

If simplify=TRUE, a data.frame with the following columns: spectrum, peptide, modif, PepScore, PepProb, seqpos

If simplify=FALSE, a list (of spectra) of lists (of peptide identifications) of lists (with information about identification and localization). spectrum -> peptide 1, peptides 2, ... -> peptide. First level: - spectrum Second level: - peptide identifications for spectrum (might be more than one) Third level: - peptide: vector with peptide sequence and modification string - site.probs: matrix with site probabilities for each phospho site - isoforms: peptide score and probabilities for each isoform

**Author(s)**

Florian P Breitwieser

**References**

Taus et al., 2011

---

**getPtmInfo**

*Get PTM site information for identified proteins from public databases.*

**Description**

Get PTM site information for identified proteins from public databases.

**Usage**

getPtmInfoFromPhosphoSitePlus(protein.group, file.name = NULL, modif = "PHOS",
pssp.url = "http://www.phosphosite.org/downloads/",
mapping = c(PHOS = "Phosphorylation_site_dataset.gz",
ACET = "Acetylation_site_dataset.gz",
METH = "Methylation_site_dataset.gz",
SUMO = "Sumoylation_site_dataset.gz",
UBI = "Ubiquitination_site_dataset.gz"))

getPtmInfoFromNextprot(protein.group, nextprot.url = "http://www.nextprot.org/rest/entry/NX_XXX/ptm?format=json",
url.wildcard = "XXX")
Arguments

protein.group ProteinGroup object.
file.name File name to save downloaded data, defaults to the original file name (see mapping).
modif Selects dataset to download (see mapping).
psp.url PhosphoSitePlus main URL for datasets.
mapping Names of PhosphoSitePlus modification datasets, mapped by modif name.
nextprot.url URL for fetching Nextprot results.
url.wildcard wildcard to replace with Uniprot Protein AC.

Details

PhosphoSitePlus datasets are downloaded and written to the working directory with its original name (see mapping) unless a file with that name exists, which is then parsed into a data.frame of suitable format.

Value

data.frame with (at least) the columns: isoform_ac, description, evidence, position

Note

PhosphoSitePlus is licensed under Creative Commons Attribution-NonCommercial-ShareAlike 3.0 Unported License and is freely available for non-commercial purpose, see http://www.phosphosite.org/staticDownloads.do.
neXtProt is licensed under the Creative Commons Attribution-NoDerivs License, see: http://creativecommons.org/licenses/by-nd/3.0.
Please read the conditions and use the data only if you agree.

Author(s)

Florian P. Breitwieser

References


Examples

```r
## Not run:
data(ib_phospho)
ptm.info.np <- getPtmInfoFromNextprot(proteinGroup(ib_phospho))
ptm.info.np <- ptm.info.np[grep("Phospho", ptm.info.np$modification.name),]
ptm.info.psp <- getPtmInfoFromPhosphoSitePlus(proteinGroup(ib_phospho), modif="PHOS")
```
groupMemberPeptides

str(ptm.info.np)
str(ptm.info.psp)

## End(Not run)

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 occurrence 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, TMT6plexSpectra and TMT10plexSpectra.

It contains per-spectrum measurements 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', 'TMT6plexSpectra' or 'TMT10plexSpectra'. 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.

ibSpectra.as.concise.data.frame(x): 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

```r
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

```r
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

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

---

**Isobar util functions**

**Description**

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

**Usage**

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

**Arguments**

- `...`: Arguments to paste.
- `sep`: Separator.
- `a`: values.
- `b`: range.

**Author(s)**

Florian P Breitwieser

**Examples**

```r
1:10
```

---

**IBSpectra analysis**

**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 = "isobar", ...)

## 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, ratio.distr = NULL, variance.function = "maxi", 
                      sign.level = 0.05, sign.level.rat = sign.level, sign.l, 
                      remove.outliers = TRUE, outliers.args = list(method = "isobar", fc.threshold = 1.3, 
                      channel1.raw = NULL, channel2.raw = NULL, use.na = FALSE)

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  Significance level.
sign.level.rat  Signal p-value significance level.
sign.level.sample  Sample p-value significance level.
remove.outliers  Should outliers be removed?
outliers.args  Arguments for outlier removal, see OUTLIERS function (TODO).
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 weights 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.

```r
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

```r
estimateRatio(ibspiked_set1, noise.model.hcd, channel1="114", channel2="115", protein=ceru.proteins, combine=FALSE)[,'lratio']
```

## spiked material channel 115 vs 114:

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>CERU_HUMAN (P00450):</td>
<td>1</td>
</tr>
<tr>
<td>CERU_RAT (P13635):</td>
<td>2</td>
</tr>
<tr>
<td>CERU_MOUSE (Q61147):</td>
<td>0.5</td>
</tr>
</tbody>
</table>

---

**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**

```r
## S4 method for signature 'character,character'
readIBSpectra(type, id.file)

# reads id file
## S4 method for signature 'character,character,character'
readIBSpectra(
  type, id.file, peaklist.file, sep = "\t", mapping.file = NULL, mapping = c(quantification.spectrum = "hcd", identification.spectrum = "cid"), id.file.domap = NULL, identifications.format = NULL, decode.titles = FALSE, ...)

# reads peaklist file
## S4 method for signature 'character,data.frame,character'
readIBSpectra(
  type, id.file, peaklist.file, annotate.spectra.f = NULL, peaklist.format = NULL, scan.lines = 0, fragment.precision = NULL, fragment.outlier.prob = NULL, ...)
```

**Arguments**

- **type** Name of class of new IBSpectra object: `iTRAQ4plexSpectra`, `iTRAQ8plexSpectra`, `TMT2plexSpectra`, `TMT6plexSpectra`, or `TMT10plexSpectra`
id.file Database search results file in ibspectra.csv or mzIdentML format. See identifications.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!

mapping.file If defined, spectrum 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 specified in id.file.domap. Then, the results are merged after mapping the identification results.

annotate.spectra.f Function which changes or annotates the spectra feature data before it is written to IBspectra object. This can be used to calculate and threshold additional scores, for example localization scores of post-translational modifications such as Delta Score (filterSpectraDeltaScore) or PhosphoRS site localization probabilities (annotateSpectraPhosphoRS).

peaklist.format "mgf" (Mascot Generic format) or "mcn" (iTracker Machine Readable output). When NULL, it detects the format on file name extension.

identifications.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.

sep sep argument of read.table

... 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 Cerupsmin proteins
ceru.acs <- protein.g(proteinGroup(ibspiked_set1),"CERU")
# create a smaller isbspectra w/ only Cerupsmins
ib.ceru <- subsetIBSpectra(ibspiked_set1,protein=ceru.acs,direction="include")

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

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

unlink(tf)

isobar-plots IBSpectra plots

Description

Various plots are implemented 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")

Description
Preprocessing is a necessary step prior to analysis of data. In a sequential order, it is often neccessary
   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=TRUE)
   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 pro-
  tein 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.

Subtract 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, IB Spectra, 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

```r
create.reports(properties.file = "properties.R",
    global.properties.file = system.file("report","properties.R", package = "isobar"),
    args = NULL, ...,
    recreate.properties.env = TRUE, recreate.report.env = TRUE)
```

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

```r
initialize.env(env, 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`  
  ```r
  system.file("report","properties.R",package="isobar")
  ```

- `args`  
  Additional (command line) arguments which overrides those in `properties.file`.

- `...`  
  Additional properties.

- `recreate.properties.env`  
  Whether a `properties.env` existing in the global environment should be used, or it should be recreated.

- `recreate.report.env`  
  Whether a `report.env` existing in the global environment should be used, or it should be recreated.

- `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 concatenates 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

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)
data(ibspiked_set2)
data(ib_phospho)

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, pch = NULL, log="xy", legend.pos = "topright", names = NULL, legend.cex = 0.8, cols = pch, lty = 1, main = protein, xlab = NULL, ylab = NULL, type="ma", show.lm = FALSE, ...)
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**: boolean. If true, `add` is called with additional information.
- **pch**: 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`.
- **lty**: 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
- **show.lm**: show LM

Author(s)

Florian P. Breitwieser

---

NoiseModel-class  

**NoiseModel objects**

Description

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

**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, or a matrix which defines the desired combination of channels (nrow=2).
- **one.to.one**: Set to false to learn noise model on 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 individually, 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(ibsiked_set1)
caru.proteins <- protein.g(proteinGroup(ibsiked_set1),"CERU")

# normalize
ibsiked_set1 <- normalize(correctIsotopeImpurities(ibsiked_set1))

# remove spiked proteins
ibsiked_set1.noceru <- exclude(ibsiked_set1,caru.proteins)
ibsiked_set1.justceru <- subsetIBSpectra(ibsiked_set1,protein=caru.proteins,direction="include")

# learn noise models
nm.i <- new("InverseNoiseModel",ibsiked_set1.noceru)
nm.e <- new("ExponentialNoiseModel",ibsiked_set1.noceru)

# learn on non-one.to.one data: not normalized, with spiked proteins
nm.n <- new("ExponentialNoiseModel",ibsiked_set1.justceru,one.to.one=FALSE)

maplot(ibsiked_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))`

---

observedKnownSites  
*Observed modification sites.*

**Description**

Functions to display the modification sites observed for each protein isoform and count the number of modified residues per protein.

**Usage**

`observedKnownSites(protein.group, protein.g, ptm.info, modif, modification.name = NULL)`

`modif.site.count(protein.group, protein.g = reporterProteins(protein.group), modif, take = max)`

`modif.sites(protein.group, protein.g = reporterProteins(protein.group), modif)`
peptide.count

Arguments

protein.group  ProteinGroup object.
protein.g       protein group identifier.
ptm.info       ptm information data.frame, see ?getPtmInfo.
modif          Modification to track, e.g. 'PHOS'.
modification.name Value to filter 'modification.name' column in ptm.info.
take           should be either max or min: When multiple isoforms are present, which value
                should be taken for the count?

Author(s)

Florian P. Breitwieser

Examples

data(ib_phospho)
data(ptm.info)

# Modification sites of reporter proteins:
# a list of protein groups,
# containing sub-lists of identified sites for each isoform
protein.modif.sites <- sort(modif.site.count(proteinGroup(ib_phospho),modif="PHOS"))

# Details on modification sites of proteins
# detected with most modifications
modif.sites(proteinGroup(ib_phospho),modif="PHOS",protein.g=names(tail(protein.modif.sites)))

# How many sites are known, and how many known sites have been observed?
observedKnownSites(proteinGroup(ib_phospho),modif="PHOS",protein.g=names(tail(protein.modif.sites)),ptm.info)

peptide.count Peptide counts, spectral counts and sequence coverage for Protein-
Group 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, ...)
Protein and peptide ratio calculation and summarization

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

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

Protein and peptide ratio calculation and summarization

*Calculating and Summarizing 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

```r
proteinRatios(ibspectra, noise.model, reporterTagNames = NULL,
              proteins = reporterProteins(proteinGroup(ibspectra)),
              peptide = NULL, cl = classLabels(ibspectra),
              combn.method = "global", combn.vs = NULL,
              symmetry = FALSE, summarize =
              FALSE, summarize.method = "mult.pval", min.detect =
              NULL, strict.sample.pval = TRUE, strict.ratio.pval =
              TRUE, orient.div = 0, sign.level = 0.05,
              sign.level.rat = sign.level, sign.level.sample =
              sign.level, ratiodistr = NULL, zscore.threshold =
              NULL, variance.function = "maxi", combine = FALSE,
              p.adjust = NULL, reverse = FALSE, cmbn = NULL,
```
Protein and peptide ratio calculation and summarization

before.summarize.f = NULL, ...)

peptideRatiosNotQuant(ibspectra, ..., peptide = unique(fData(ibspectra)[!fData(ibspectra)["use.for.quant"]], columns = c("peptide", "modif", "site.probs")))

peptideRatios(ibspectra, ..., peptide = peptides(proteinGroup(ibspectra), columns = c("peptide", "modif")))

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

combn.protein.tbl(cmbn, reverse = FALSE, ...)

summarize.ratios(ratios, by.column = "ac", summarize.method = "mult.pval", min.detect = NULL, n.combination = NULL, strict.sample.pval = TRUE, strict.ratio.pval = TRUE, orient.div = 0, sign.level = 0.05, sign.level.rat = sign.level, sign.level.sample = sign.level, variance.function = "maxi", ratiodistr = NULL)

Arguments

ibspectra IBSpectra object

x for combn.matrix: reporter names. See reporterTagNames. argument of proteinRatios.

ratios result of combn.protein.tbl

by.column Column(s) which are the identifiers. Usually "ac", "peptide" or c("peptide","modif")

cmbn result of combn.matrix

before.summarize.f Function which is called after calculating ratios before summarizing them.

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.

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 "versus.channel", all combinations against channel vs.

combn.method "global", "interclass", "intra-class", "versus.class" or "versus.channel". Defines which ratios are computed, based on class labels cl

method See combn.method

combn.vs vs argument for combn, if combn.method is "versus.class" or "versus.channel".

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.
Protein and peptide ratio calculation and summarization

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

zscore.threshold z-score threshold to apply

... 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 comparisons. 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
  r2

Author(s)
Florian P Breitwieser, Jacques Colinge

See Also
IBSpectra, isobar-preprocessing isobar-analysis
ProteinGroup-class

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.
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.
as.concise.data.frame(from): Creates a 'concise' data.frame with one spectrum per row, and protein ACs combined

Accessors

In the following code snippets, x is a ProteinGroup object.
spectrumToPeptide(x): Gets spectrum to peptide assignment.
peptideInfo(x): Peptide information such as protein start position.
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=)

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
proteinInfo-methods

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("A","B"),each=6,"C","D"))
pag <- ProteinGroup(tbl)
pag
proteinGroupTable(pag)

data(ibsспоред_set1)
pag <- proteinGroup(ibsспоред_set1)
ceru.proteins <- protein.g(pag,"CERU")

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

## peptides shared by all ceru proteins
peptides(pag,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, fields = c(accession = "id", name
proteinInfo-methods

getProteinInfoFromTheInternet(x)
getProteinInfoFromNextProt(x)
getProteinInfoFromBiomart(x, database = "Uniprot")
getProteinInfoFromBioDb(x, ..., con = NULL)
getProteinInfoFromEntrez(x, splice.by = 200)

Arguments

x 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
fields mapping of CSV field names to proteinInfo field names
... arguments to build database connection.
protein.info protein info data.frame

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

```r
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")
```

### Description

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

### Usage

```r
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.g is combined.

### Author(s)

Florian P Breitwieser

### Examples

```r
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, vs.class = NULL, sep = ".", cmbn = NULL,
short.names = FALSE)
```

**Arguments**
- `quant.tbl`: Output of proteinRatios or peptideRatios.
- `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 comparison name will be i.e. `class2` instead of `class2/class1`.

**Author(s)**
Florian P. Breitwieser

---

reporter.protein-methods

*Get reporter protein group identifier for protein group identifier*

**Description**
Methods for function `reporter.protein` in package `isobar`

**Methods**
```
signature(x = "ProteinGroup", protein.g = "character") Get reporter protein for protein group identifier.
```
sanitize

Helper function for LaTeX export

Description
Sanitizes strings for LaTeX

Usage
sanitize(str, dash = TRUE)

Arguments
str character string to be escaped
dash should 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
ibinspectra 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 estimateRatio.
specificities

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. un-
specific: peptides shared with other proteins.
spectra.count2

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")
Description

Location scale family T distribution, based on the original T function.

Objects from the Class

Objects can be created by calls of the form `new("Tlsd", df, location, scale).

Slots

gaps: Object of class "OptionalMatrix" ~
img: Object of class "rSpace" ~
param: Object of class "OptionalParameter" ~
r: Object of class "function" ~
d: Object of class "OptionalFunction" ~
p: Object of class "OptionalFunction" ~
q: Object of class "OptionalFunction" ~
.withSim: Object of class "logical" ~
.withArith: Object of class "logical" ~
.logExact: Object of class "logical" ~
.lowerExact: Object of class "logical" ~

Symmetry: Object of class "DistributionSymmetry" ~

Extends


Methods

No methods defined with class "Tlsd" in the signature.

Author(s)

Florian P. Breitwieser, based on original T distribution class.

Examples

```
showClass("Tlsd")
```
**TlsParameter-class**  
*Class "TlsParameter"

**Description**

The parameter of a location scale t distribution, used by Tlsd-class

**Objects from the Class**

Objects can be created by calls of the form `new("TlsParameter", ...)`. Usually an object of this class is not needed on its own, it is generated automatically when an object of the class Tlsd is instantiated.

**Slots**

- `df`: Object of class "numeric"
- `location`: Object of class "numeric"
- `scale`: Object of class "numeric"
- `name`: Object of class "character"

**Extends**


**Methods**

No methods defined with class "TlsParameter" in the signature.

**Author(s)**

Florian P. Breitwieser, based on original TParameter class.

**See Also**

Tlsd

**Examples**

`showClass("TlsParameter")`
writeHscoreData  Write identifications into a format suitable for Hscore.

Description

Write identifications into a format suitable for Hscore.

Usage

writeHscoreData(outfile, ids, massfile = "defs.txt")

Arguments

outfile  Output file.
ids  IBSpectra identifications data.frame (ie fData).
massfile  Definition file for Hscore.

Author(s)

Florian P. Breitwieser

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

*Topic `\textasciitilde\textasciitilde` other possible keyword(s) `\textasciitilde\textasciitilde`
  reporter.protein-methods, 42
*Topic `\textasciitilde\textasciitilde`\textasciitilde\textasciitilde\textasciitilde
  calculate.dNSAF
  report.protein-methods, 42
  calculate.dNSAF, 6
*Topic `\textasciitilde\textasciitilde`\textasciitilde\textasciitilde\textasciitilde
  calculate.emPAI
  report.protein-methods, 42
  calculate.emPAI, 7
*Topic `\textasciitilde\textasciitilde`\textasciitilde\textasciitilde\textasciitilde
  getPhosphoRSProbabilities
  report.protein-methods, 42
  getPhosphoRSProbabilities, 11
*Topic `\textasciitilde\textasciitilde`
  Tlsd-class, 47
  TlsParameter-class, 48
*Topic `\textasciitilde\textasciitilde`
  isobar.data, 29
  specificities, 44
*Topic `\textasciitilde\textasciitilde`
  proteinInfo-methods, 39
  reporter.protein-methods, 42
*Topic `\textasciitilde\textasciitilde`\textasciitilde\textasciitilde\textasciitilde
  isobar-package, 3
  %inrange% (Isobar util functions), 20
AbscontDistribution, 47
AbscontDistribution-class
  (distr-methods), 9
AcDcLcDistribution, 47
adjust.ratio.pvalue
  (calculate-pvalues), 5
AnnotatedDataFrame, 17
as.data.frame, IBSpectra-method
  (IBSpectra-class), 16
as.data.frame, ProteinGroup-method
  (ProteinGroup-class), 37
as.data.frame, IBSpectra
  (IBSpectra-class), 16
as.data.frame, ProteinGroup
  (ProteinGroup-class), 37
AssayData, 17
calc.delta.score, 4
calc.pep.delta.score
  (calc.delta.score), 4
calcCumulativeProbXGreaterThanY
  (distr-methods), 9
calcPeptidePosition, 5
calcProbXDifNormals (distr-methods), 9
calcProbXGreaterThanY (distr-methods), 9
calculate-pvalues, 5
calculate.dNSAF, 6, 8, 34
calculate.emPAI, 7, 34
calculate.mult.sample.pvalue
  (calculate-pvalues), 5
calculate.ratio.pvalue
  (calculate-pvalues), 5
calculate.sample.pvalue
  (calculate-pvalues), 5
Cauchy, 10
class: IBSpectra (IBSpectra-class), 16
class: NoiseModel (NoiseModel-class), 30
class: ProteinGroup
  (ProteinGroup-class), 37
classLabels (IBSpectra-class), 16
classLabels, IBSpectra-method
  (IBSpectra-class), 16
classLabels<- (IBSpectra-class), 16
classLabels<-, IBSpectra-method
  (IBSpectra-class), 16
coerce, data.frame, ProteinGroup-method
  (ProteinGroup-class), 37
coerce, IBSpectra, data.frame-method
  (IBSpectra-class), 16
combn.matrix (Protein and peptide ratio calculation and summarization), 34
combn.protein.tbl (Protein and peptide ratio calculation and summarization), 34
connect.nodes (isobar-reports), 27
correct.peptide.ratios, 8
correctIsotopeImpurities, 17, 19
correctIsotopeImpurities
  (isobar-preprocessing), 26
correctIsotopeImpurities, IBSpectra-method
  (isobar-preprocessing), 26
create.meta.reports (isobar-reports), 27
INDEX

naRegion, NoiseModel-method (NoiseModel-class), 30
naRegion<-(NoiseModel-class), 30
naRegion<-, NoiseModel-method (NoiseModel-class), 30
noise.model.hcd(isobar.data), 29
noiseFunction(NoiseModel-class), 30
noiseFunction, NoiseModel-method (NoiseModel-class), 30
NoiseModel(NoiseModel-class), 30
NoiseModel, IBSpectra-method (NoiseModel-class), 30
NoiseModel-class, 30
Norm, 10
normalize, 19
normalize(isobar-preprocessing), 26
number.ranges, 32
observable.peptides, 8
observable.peptides(calculate.emPAI), 7
observedKnownSites, 32
OptionalParameter, 48
p.adjust, 36
Parameter, 48
parameter(NoiseModel-class), 30
parameter, NoiseModel-method (NoiseModel-class), 30
Parameter-class(distr-methods), 9
parameter<-(NoiseModel-class), 30
parameter<-, NoiseModel-method (NoiseModel-class), 30
paste, 40
paste0(Isobar util functions), 20
peptide.count, 33
peptideInfo(ProteinGroup-class), 37
peptideInfo, ProteinGroup-method (ProteinGroup-class), 37
peptideInfo-methods (ProteinGroup-class), 37
peptideNProtein(ProteinGroup-class), 37
peptideNProtein, ProteinGroup-method (ProteinGroup-class), 37
peptideRatios, 6
peptideRatios(Protein and peptide ratio calculation and summarization), 34
peptideRatiosNotQuant(Protein and peptide ratio calculation and summarization), 34
peptides, 34, 45
peptides(ProteinGroup-class), 37
peptides, ProteinGroup, character-method (ProteinGroup-class), 37
peptides, ProteinGroup, missing-method (ProteinGroup-class), 37
peptideSpecificity, ProteinGroup-method (ProteinGroup-class), 37
phenoData, 18
plot, 30
plot.default, 30
plot.NoiseModel(NoiseModel-class), 30
plotRatio(isobar-plots), 25
plotRatio, IBSpectra, character, character-method (isobar-plots), 25
print_classlabels_tbl(isobar-reports), 27
print_groupsizes(isobar-reports), 27
print_longtablehdr(isobar-reports), 27
print_longtablehdr_peptide(isobar-reports), 27
print_protein_grp_info(isobar-reports), 27
print_protein_grp_tbl(isobar-reports), 27
print_protein_notquant_tbl(isobar-reports), 27
print_protein_quant_tbl(isobar-reports), 27
print_protein_quant_tbl(isobar-reports), 27
protein.g, 40, 46
protein.g(ProteinGroup-class), 37
protein.g, ProteinGroup, character, character-method (ProteinGroup-class), 37
protein.g, ProteinGroup, character-method (ProteinGroup-class), 37
proteinDescription(proteinNameAndDescription), 41
proteinGeneName(proteinNameAndDescription), 41
ProteinGroup, 7, 8, 17, 18, 22, 24, 27, 34, 45
ProteinGroup(ProteinGroup-class), 37
proteinGroup(IBSpectra-class), 16
ProteinGroup, data.frame, missing-method (ProteinGroup-class), 37
ProteinGroup, data.frame, NULL-method (ProteinGroup-class), 37
ProteinGroup, data.frame, ProteinGroup-method (ProteinGroup-class), 37
proteinGroup, IBSpectra-method (IBSpectra-class), 16
ProteinGroup-class, 37
proteinGroup.as.concise.data.frame (ProteinGroup-class), 37
proteinGroup<-(IBSpectra-class), 16
proteinGroup<-,IBSpectra-method (IBSpectra-class), 16
proteinGroupTable (ProteinGroup-class), 37
proteinGroupTable, ProteinGroup-method (ProteinGroup-class), 37
proteinID (proteinNameAndDescription), 41
proteinInfo, 7, 8
proteinInfo (proteinInfo-methods), 39
proteinInfo, ProteinGroup, character, missing-method (proteinInfo-methods), 39
proteinInfo, ProteinGroup, missing, character-method (proteinInfo-methods), 39
proteinInfo, ProteinGroup, missing, missing-method (proteinInfo-methods), 39
proteinInfo, ProteinGroup-method (proteinInfo-methods), 39
proteinInfo-methods, 39
proteinInfo <-- (proteinInfo-methods), 39
proteinInfo<-, ProteinGroup-method (proteinInfo-methods), 39
proteinInfoIsOnSpliceVariants (proteinInfo-methods), 39
proteinNameAndDescription, 41
proteinRatios, 6, 10, 22
proteinRatios (Protein and peptide ratio calculation and summarization), 34
protGgdata (isobar-plots), 25
protGgdata, ANY, character, character-method (isobar-plots), 25
raplot (isobar-plots), 25
raplot, IBSpectra-method (isobar-plots), 25
ratiosReshapeWide, 42
read.mzid (isobar-import), 23
readIBSpectra, 12, 16, 17
readIBSpectra (isobar-import), 23
readIBSpectra, character, character, character-method (isobar-import), 23
readIBSpectra, character, character, missing-method (isobar-import), 23
readIBSpectra, character, character-method (isobar-import), 23
readIBSpectra, character, data.frame, character-method (isobar-import), 23
readIBSpectra, character, data.frame, missing-method (isobar-import), 23
readPhosphoRSPoutput (getPhosphoRSPProbabilities), 11
readProteinGroup (ProteinGroup-class), 37
readProteinGroup2 (ProteinGroup-class), 37
reporter-specific (specificities), 44
reporter.protein (reporter.protein-methods), 42
reporter.protein, ProteinGroup, character-method (reporter.protein-methods), 42
reporter.protein-methods, 42
reporterData (IBSpectra-class), 16
reporterData, IBSpectra-method (IBSpectra-class), 16
reporterData<-(IBSpectra-class), 16
reporterData<-, IBSpectra-method (IBSpectra-class), 16
reporterIntensities, 17
reporterIntensities (IBSpectra-class), 16
reporterIntensities, IBSpectra-method (IBSpectra-class), 16
reporterIntensities<-(IBSpectra-class), 16
reporterIntensities<-, IBSpectra-method (IBSpectra-class), 16
reporterIntensityPlot (isobar-plots), 25
reporterIntensityPlot, IBSpectra-method (isobar-plots), 25
reporterIntensityPlot-methods (isobar-plots), 25
reporterMasses, 17
reporterMasses (IBSpectra-class), 16
reporterMasses, IBSpectra-method (IBSpectra-class), 16
reporterMasses<-(IBSpectra-class), 16
reporterMasses<-, IBSpectra-method (IBSpectra-class), 16
reporterMassPrecision (isobar-plots), 25
reporterMassPrecision, IBSpectra, logical-method (isobar-plots), 25
weightedMean (Protein and peptide ratio calculation and summarization), 34
weightedMean, numeric, numeric-method (Protein and peptide ratio calculation and summarization), 34
weightedVariance (Protein and peptide ratio calculation and summarization), 34
weightedVariance, numeric, numeric, missing-method (Protein and peptide ratio calculation and summarization), 34
weightedVariance, numeric, numeric, numeric-method (Protein and peptide ratio calculation and summarization), 34
write.table, 49
write.tex.commands (isobar-reports), 27
write.xls.report (isobar-reports), 27
writeData (IBSpectra-class), 16
writeData, IBSpectra-method (IBSpectra-class), 16
writeHscoreData, 49
writeIBSpectra, 49
writePhosphoRSInput (getPhosphoRSProbabilities), 11