Package 'specL'

October 12, 2026

Type Package

Title specL - Prepare Peptide Spectrum Matches for Use in Targeted Proteomics

Version 1.46.0

Depends R (>= 4.1), DBI (>= 0.5), methods (>= 3.3), protViz (>= 0.7), RSQLite (>= 1.1), seqinr (>= 3.3)

Suggests BiocGenerics, BiocStyle (>= 2.2), knitr (>= 1.15), rmarkdown, RUnit (>= 0.4)

Description provides a functions for generating spectra libraries that can be used for MRM SRM MS workflows in proteomics. The package provides a BiblioSpec reader, a function which can add the protein information using a FASTA formatted amino acid file, and an export method for using the created library in the Spectronaut software. The package is developed, tested and used at the Functional Genomics Center Zurich <https://fgcz.ch>.

License GPL-3

URL http://bioconductor.org/packages/specL/

Collate read.bibliospec.R genSwathIonLib.R annotate.protein_id.R AllGenerics.R specL.R specLSet.R cdsw.R zzz.R

biocViews MassSpectrometry, Proteomics

LazyData true

BugReports https://github.com/fgcz/specL/issues

VignetteBuilder knitr

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

git_branch RELEASE_3_23

git_last_commit 4a57dea

git_last_commit_date 2026-04-28

Repository Bioconductor 3.23

Date/Publication 2026-10-11

Author Christian Panse [aut, cre] (ORCID:
<https://orcid.org/0000-0003-1975-3064>),
Jonas Grossmann [aut] (ORCID: <https://orcid.org/0000-0002-6899-9020>),
Christian Trachsel [aut],
Witold E. Wolski [ctb]

Maintainer Christian Panse <cp@fgcz.ethz.ch>

Contents

annotate.protein_id . . . 2
cdsw . . . 4
genSwathIonLib . . . 5
iRTpeptides . . . 7
ms1.p2069 . . . 8
peptideStd . . . 9
plot-methods . . . 10
read.bibliospec . . . 10
show-methods . . . 11
specL-class . . . 11
specLSet-class . . . 13
write.spectronaut-methods . . . 14

Index **15**

`annotate.protein_id` *Annotate protein_id*

Description

This function assigns the protein identifier for a list of tandem mass specs having a peptide sequence assigned.

Usage

```
annotate.protein_id(data, file = NULL, fasta = read.fasta(file = file,
        as.string = TRUE, seqtype = "AA"), digestPattern = "(([RK])|(^)|(^M))")
```

Arguments

`data`	list of records containing mZ and peptide sequences.
`file`	file name of a FASTA file.
`fasta`	a fasta object as returned by the `seqinr::read.fasta(...)` method.
`digestPattern`	a regex pattern which can be used by the `grep` command. the default regex pattern assumes a tryptic digest.

Details

The protein sequences a read by the `read.fasta` function of the `seqinr` package. The protein identifier is written to the protein `proteinInformation` variable.

If the function is called on a multi-core architecture it uses `mclapply`.

It is recommended to load the FASTA file prior to running `annotate.protein_id` using

`myFASTA <- read.fasta(file = file, as.string = TRUE, seqtype = "AA")`

instead of providing the FASTA file name to the function.

Value

it returns a list object.

Author(s)

Jonas Grossmann and Christian Panse, 2014

See Also

`?read.fasta` of the `seqinr` package.

http://www.uniprot.org/help/fasta-headers

Examples

```
# annotate.protein_id

# our Fasta sequence
  irtFASTAseq <- paste(">zz|ZZ_FGCZCont0260|",
  "iRT_Protein_with_AAAAK_spacers concatenated Biognosys\n",
  "LGGNEQVTRAAAAKGAGSSEPVTGLDAKAAAAKVEATFGVDESNAKAAAAKYILAGVENS",
  "KAAAAKTPVISGGPYEYRAAAAKTPVITGAPYEYRAAAAKDGLDAASYYAPVRAAAAKAD",
  "VTPADFSEWSKAAAAKGTFIIDPGGVIRAAAAKGTFIIDPAAVIRAAAAKLFLQFGAQGS",
  "PFLK\n")

# be realistic, do it from file
  Tfile <- file();  cat(irtFASTAseq, file = Tfile);

#use read.fasta from seqinr
  fasta.irtFASTAseq <-read.fasta(Tfile, as.string=TRUE, seqtype="AA")
  close(Tfile)

#annotate with proteinID
# -> here we find all psms from the one proteinID above
  peptideStd <- specL::annotate.protein_id(peptideStd,
  fasta=fasta.irtFASTAseq)

#show indices for all PSMs where we have a proteinInformation
 which(unlist(lapply(peptideStd,
  function(x){nchar(x$proteinInformation)>0})))
```

cdsw *Generate Dynamic SWATH Window*

Description

This function computes dynamic SWATH windows (cdsw) for a given vector of numeric values, an psmSet class, or an specLSet class. The input R data object can be generated using the read.bibliospec function.

Usage

```
cdsw(x, n=20, overlap=1, ...)
```

Arguments

x	Numeric vector or psmSet class.
n	Number of desired SWATH windows. Default is set to 20.
overlap	Overlap of SWATH windows. The default is 1 Dalton.
...	pass arguments to hist function.

Details

The function determines the SWATH windows using the quantile function.

Value

The output is data.frame having the attributes from, to, mid, width, and counts. In the ideal output all bins should contain the same number of numeric values. This requirement is violated because the window borders are rounded with no digit after the comma.

Author(s)

Christian Panse, Christian Trachsel, 2015, 2016

See Also

The S3 class definition: showClass("psmSet")

Examples

```
# do not plot histogram
cdsw(peptideStd, plot = FALSE,  overlap = 0)

# plot hist
cdsw(peptideStd, freq = TRUE)

# pre-filtering
## cdsw(x=q1[350 <= q1 & q1 <= 1250], n=20, overlap = 0, freq=TRUE)
```

genSwathIonLib *Spectrum library generator for SWATH analysis*

Description

This function generates an ion library for SWATH analysis. It takes a R data object which contains a peak list. The R data object can be generated using the `read.bibliospec` function.

Usage

```
genSwathIonLib(data,
    data.fit = data,
    mascotIonScoreCutOFF=20,
    proteinIDPattern='',
    max.mZ.Da.error = 0.1,
    ignoreMascotIonScore = TRUE,
    topN = 10,
    fragmentIonMzRange = c(300, 1800),
    fragmentIonRange = c(4, 100),
    fragmentIonFUN = .defaultSwathFragmentIon,
    iRT = specL::iRTpeptides,
    AminoAcids = protViz::AA,
    breaks=NULL,
    NORMALIZE=.normalize_rt)
```

Arguments

`data`	data set containing mZ and peptide sequence.
`mascotIonScoreCutOFF`	a value for filtering the specs.
`proteinIDPattern`	a filter for protein.
`max.mZ.Da.error`	the mZ error in Dalton on ms2 level.
`ignoreMascotIonScore`	Boolean if mascot score is considered or not.
`topN`	returns the most `N` intense fragment ion only.
`fragmentIonMzRange`	mZ range filter of fragment ion.
`fragmentIonRange`	range filter of the number of identified fragment ion which are assigned in the spectrum library set in `fragmentIonTyp`. Use this option to generate a library with a minimum of five transitions for all peptides using `c(5,100)`, all peptides where at least not five transmissions found were omitted.

fragmentIonFUN	function (b, y) which derives all requested fragment ion out a given tuple of b and y ion. If the parameter is not specified the method uses an internal function similar as the example below.
iRT	optional table which contains iRT peptides. If an iRT table is provided (default) a lm is applied to normalize the rt in data. See also ?iRT. A necessary condition is that data contains at least two iRT peptides.
AminoAcids	a list containing of 1-letter code and mono-isotopic mass of the amino acids. Default uses the protViz::AA data set.
data.fit	data set containing mZ and peptide sequence which is used for normalizing rt using a linear model lm(formula = rt ~ aggregateInputRT * fileName, data). The rt aggregation for the model uses median.
breaks	provides a vector of SWATH windows. If q1 (precursor mass) and q3 (fragment ion) fall into the same SWATH window the fragment ion is ignored in the resulting ion library. The follwing code shows an example breaks=seq(400, 2000, by=25).
NORMALIZE	is a function(data, data.fit, iRT, plot=FALSE) normalizing the rt. The default is a internal R helper function using lm.

Details

The function is the main contribution of the specL package. It generates the spectra library used in a SWATH analysis workflow out of a mass spectrometric measurement.

genSwathIonLib uses the core functions protViz::findNN, protViz::fragmentIon, and protViz::aa2mass.

The input is read by using read.bibliospec function of this package and passed by the data function parameter. If no BiblioSpec files are available also Mascot DAT files can be read using scripts contained in the protViz package exec folder.

If the protein information is lost you can benefit for the specL::annotate.protein_id.Rd method.

The function first appear in the protViz 0.1.45 package. It has been removed in protViz 0.2.6 to avoid package dependencies.

Value

The output is a data structure defined as specLSet object. The generic method ionlibrary returns a list of specL objects, specLSet also stores the input and normalized retention times.

Author(s)

Christian Panse, Christian Trachsel, and Jonas Grossmann 2012, 2013, 2014, 2016

See Also

The S4 class definition: showClass("specL") showClass("specLSet")

and the package vignette file. vignette('specL')

Examples

```
myFragmentIon <- function (b, y) {
    Hydrogen <- 1.007825
    Oxygen <- 15.994915
    Nitrogen <- 14.003074

    b1_ <- (b )
    y1_ <- (y )

    b2_ <- (b + Hydrogen) / 2
    y2_ <- (y + Hydrogen) / 2

    b3_ <- (b + 2 * Hydrogen) / 3
    y3_ <- (y + 2 * Hydrogen) / 3

    return( cbind(b1_, y1_, b2_, y2_, b3_, y3_) )
}

peptideStd.ionLib <- genSwathIonLib(data=peptideStd,
    data.fit=peptideStd.redundant,
    fragmentIonFUN=myFragmentIon)


summary(peptideStd.ionLib)


idx<-40

op <- par(mfrow = c(2,1))

plot(peptideStd.ionLib)

text(rt.input(peptideStd.ionLib)[idx],
    rt.normalized(peptideStd.ionLib)[idx],
    "X", cex=1.5)

plot(ionlibrary(peptideStd.ionLib)[[idx]])


## Not run:
    write.spectronaut(peptideStd.ionLib,
        file="peptideStd.ionLib.csv")

## End(Not run)
```

---

iRTpeptides *iRT peptides - independent retention time peptides*

Description

iRTpeptides data are used for genSwathIonLib rt normalization assuming.

iRTpeptides first appear in the protViz 0.1.45 package. It has been removed in protViz 0.2.10 to avoid package dependencies.

Format

contains a table

Author(s)

Jonas Grossmann and Christian Panse 2013

References

Using iRT, a normalized retention time for more targeted measurement of peptides. Escher C, Reiter L, MacLean B, Ossola R, Herzog F, Chilton J, MacCoss MJ, Rinner O. Source Proteomics. 2012 Apr;12(8):1111-21. doi: 10.1002/pmic.201100463.

Examples

```
plot(sort(iRTpeptides$rt))

plot(pim<-protViz::parentIonMass(as.character(iRTpeptides$peptide)) ~ iRTpeptides$rt)
```

---

ms1.p2069 *ms1 mass*

Description

ms1.p2069 contains 11830 peptide pre cursor m/z values of human proteins.

Format

contains a numeric vector

Author(s)

Christian Panse

peptideStd *Peptide standard*

Description

This dataset is a list of a peptide spectrum matches (protein identification result) from two standard runs.

Format

contains a list of peptide spectrum assignments.

Details

These standard runs (LCMS experiments) are routinely run on well maintained instruments. In this case a standard run consits of a digest of the FETUIN_BOVINE protein (400 amol) and iRT peptides.

Author(s)

Christian Panse, Christian Trachsel and Jonas Grossmann 2014

See Also

http://fgcz-data.uzh.ch/~cpanse/specL/data/

Examples

```
peakplot(peptideStd[[40]]$peptideSequence, peptideStd[[40]])

## Not run:

    download.file("http://fgcz-data.uzh.ch/~cpanse/specL/data/peptideStd.blib",
        destfile="peptideStd.blib")

 download.file("http://fgcz-data.uzh.ch/~cpanse/specL/data/peptideStd.redundant.blib",
        destfile="peptideStd.redundant.blib")

    # checksum

    if (require(tools)){
        md5sum(c("peptideStd.blib", "peptideStd.redundant.blib")) ==
            c("3f231931e54efd6516d7aa302073b17f",
                "8bab829d9e99344136613a17c0374b90")
    }

    peptideStd <- read.bibliospec("peptideStd.blib")
    peptideStd.redundant <- read.bibliospec("peptideStd.redundant.blib")
```

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

`plot-methods` *Method for Function* `plot` *in Package* **specL**

Description

This methode has no additional arguments.

Value

The method plots on the current device.

Methods

`signature(x = "specL")` Plots the specL determined ions.

`signature(x = "specLSet")` Plots retention time versus retention time.

`read.bibliospec` *BiblioSpec Reader*

Description

This function reads a BiblioSpec generated file and returns a list of tandem mass specs (psm objects), peptide assignments, retention times, and modifications records. The type of the data structire which is returned is called psmSet.

Usage

```
read.bibliospec(file)
```

Arguments

`file`	the name of the BiblioSpec generated SQLite file.

Details

The function performs a SQL query on the SQLite files generated by bibliospec using the `RSQLite` package. The function is required for generating spec libraries used in a SWATH workflow.

BiblioSpec files are generated by using Skyline.

Value

It returns a list which can be read by the `genSwathIonLib` function and the `protViz::peakplot` function.

Author(s)

Christian Panse, 2014, 2015

See Also

https://skyline.gs.washington.edu/labkey/project/home/software/Skyline/begin.view

https://skyline.gs.washington.edu/labkey/project/home/software/BiblioSpec/begin.view

http://www.sqlite.org/

?SQLite

Examples

```
read.bibliospec
```

---

show-methods *Methods for Function* show *in Package* **specL** ~~

Description

Methods for function show in package **specL** ~~ writes specL or specLSet objects to a file or console.

Methods

`signature(x = "specL")` Prints specL object data to the console.

`signature(x = "specLSet")` Prints specL object data to the console.

---

specL-class *Class* "specL"

Description

This class is used to store, print, and plot the generated results of the package.

Objects from the Class

Objects can be created by calls of the form `new("specL", ...)`.

Slots

`group_id`: Object of class `"character"` just an id

`peptide_sequence`: Object of class `"character"` AA sequence

`proteinInformation`: Object of class `"character"` a string contains the protein identifier.

`q1`: Object of class `"numeric"` peptide weight m/Z as measured by the MS device

`q1.in_silico`: Object of class `"numeric"` peptide weight m/Z computed in-silico.

`q3`: Object of class `"numeric"` measured fragment ions.

`q3.in_silico`: Object of class `"numeric"` in-silico derived fragment ions.

`score`: Object of class `"numeric"` taken from bibliospec - psm score

`prec_z`: Object of class `"numeric"` pre-cursor charge.

`frg_type`: Object of class `"character"` fragmenbt ion type, e.g., b or y ion.

`frg_nr`: Object of class `"numeric"` fragment ion number

`frg_z`: Object of class `"numeric"` fragment ion charge.

`relativeFragmentIntensity`: Object of class `"numeric"` percentage base peaks of frament ions.

`irt`: Object of class `"numeric"` independent retention time in seconds.

`peptideModSeq`: Object of class `"numeric"` a vector contains the mass diff between AA and mod AA.

`mZ.error`: Object of class `"numeric"` a string contains the protein identifier.

`filename`: Object of class `"character"` a string contains the filename of the ions.

Methods

plot `signature(x = "specL")`: plots the fragment ions of `specL` object.

show `signature(x = "specL")`: shows the content of `specL` object.

write.spectronaut `signature(x = "specL")`: writes the `specL` object to a ASCII file.

Note

No notes yet.

Author(s)

Christian Panse 2014

See Also

`genSwathIonLib`

Examples

```
showClass("specL")
```

specLSet-class *Class* "specLSet"

Description

This class is used to store, show, and write generated results of the package.

Objects from the Class

Objects can be created by calls of the form `new("specLSet", ...)`.

Slots

`input.parameter`: A list of parameter values.

`ionlibrary`: A list of "specL" objects.

`rt.normalized`: A numeric vector of normalized retention time values.

`rt.input`: A numeric vector of retention time values.

Methods

show `signature(x = "specLSet")`: shows the object content.

summary `signature(x = "specLSet")`: print summary of object content.

plot `signature(x = "specLSet")`: plots normalized versus input rt.

write.spectronaut `signature(x = "specLSet")`: writes object to ASCII file.

generate.consensus `signature(x = "specLSet")`: generates consensus specLSet object by combining all specL objects having a redundant group_id which is defined by `paste(x@q1.in_silico, x@peptideModSeq, sep='_')`.

merge.specLSet `signature(x = "specLSet")`: merges two specLSet objects.

ionlibrary `signature(x = "specLSet")`: returns a list of specL objects.

rt.input `signature(x = "specLSet")`: returns a numeric vector of input rt values.

rt.normalized `signature(x = "specLSet")`: returns a numeric vector of normalized rt values.

getProteinPeptideTable `signature(x = "specLSet")`: returns table of peptide protein mappings.

Note

No notes yet.

Author(s)

Christian Panse 2014

See Also

genSwathIonLib

Examples

```
showClass("specL")
showClass("specLSet")
```

`write.spectronaut-methods`
Methods for Function `write.spectronaut` *in Package* **specL**

Description

Methods for function `write.spectronaut` in package **specL** ~~ writes specL objects to a file in a format which can be read by the 'Spectronaut' software. additional arguments are

file A file name. default is `file='spec.txt'`.

Methods

`signature(x = "specL")` Prints specL object data to to a file.

`signature(x = "specLSet")` Prints specL object data to to a file.

Index

∗ **classes**
specL-class, 11
specLSet-class, 13
∗ **methods**
plot-methods, 10
show-methods, 11
write.spectronaut-methods, 14

annotate.protein_id, 2

BiblioSpec *(*read.bibliospec*)*, 10
bibliospec *(*read.bibliospec*)*, 10

cdsw, 4

generate.consensus *(*specLSet-class*)*, 13
generate.consensus,specLSet-method *(*specLSet-class*)*, 13
generateDynamicSwathWindow *(*cdsw*)*, 4
genSwathIonLib, 5, *12, 13*
getProteinPeptideTable *(*specLSet-class*)*, 13
getProteinPeptideTable,specLSet-method *(*specLSet-class*)*, 13

ionlibrary *(*specLSet-class*)*, 13
ionlibrary,specLSet-method *(*specLSet-class*)*, 13
iRT *(*iRTpeptides*)*, 7
irt *(*iRTpeptides*)*, 7
iRTpeptides, 7

merge.specLSet *(*specLSet-class*)*, 13
merge.specLSet,specLSet-method *(*specLSet-class*)*, 13
MRM *(*genSwathIonLib*)*, 5
ms1.p2069, 8

peptideStd, 9
plot *(*specLSet-class*)*, 13
plot,ANY-method *(*plot-methods*)*, 10
plot,specL-method *(*specL-class*)*, 11
plot,specL-methods *(*plot-methods*)*, 10
plot,specLSet-method *(*specLSet-class*)*, 13
plot,specLSet-methods *(*plot-methods*)*, 10
plot-methods, 10
plot.psm *(*read.bibliospec*)*, 10
plot.psmSet *(*read.bibliospec*)*, 10

read.bibliospec, 10
rt.input *(*specLSet-class*)*, 13
rt.input,specLSet-method *(*specLSet-class*)*, 13
rt.normalized *(*specLSet-class*)*, 13
rt.normalized,specLSet-method *(*specLSet-class*)*, 13

show,ANY-method *(*show-methods*)*, 11
show,specL-method *(*specL-class*)*, 11
show,specLSet-method *(*specLSet-class*)*, 13
show-methods, 11
Skyline *(*read.bibliospec*)*, 10
skyline *(*read.bibliospec*)*, 10
specL *(*genSwathIonLib*)*, 5
specL-class, 11
specLSet-class, 13
summary,specLSet-method *(*specLSet-class*)*, 13
summary.psmSet *(*read.bibliospec*)*, 10
SWATH *(*genSwathIonLib*)*, 5
swath *(*genSwathIonLib*)*, 5

write.spectronaut *(*write.spectronaut-methods*)*, 14
write.spectronaut,ANY-method *(*write.spectronaut-methods*)*, 14
write.spectronaut,specL-method *(*specL-class*)*, 11

write.spectronaut,specL-methods
(write.spectronaut-methods), 14
write.spectronaut,specLSet-method
(specLSet-class), 13
write.spectronaut,specLSet-methods
(write.spectronaut-methods), 14
write.spectronaut-methods, 14