Package 'AWAggregator'

June 4, 2026

Type Package

Title Attribute-Weighted Aggregation

Version 1.3.0

Description This package implements an attribute-weighted aggregation algorithm which leverages peptide-spectrum match (PSM) attributes to provide a more accurate estimate of protein abundance compared to conventional aggregation methods. This algorithm employs pre-trained random forest models to predict the quantitative inaccuracy of PSMs based on their attributes. PSMs are then aggregated to the protein level using a weighted average, taking the predicted inaccuracy into account. Additionally, the package allows users to construct their own training sets that are more relevant to their specific experimental conditions if desired.

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

Suggests AWAggregatorData, BiocStyle, ExperimentHub, knitr, rmarkdown, testthat (>= 3.0.0)

Imports dplyr, Peptides, progress, purrr, ranger, rlang, stats, stringr, tidyr, toOrdinal, utils

Depends R (>= 4.5.0)

biocViews Software, MassSpectrometry, Preprocessing, Proteomics, Regression

BugReports https://github.com/Tan-Jiahua/AWAggregator/issues

URL https://github.com/Tan-Jiahua/AWAggregator

VignetteBuilder knitr

Config/testthat/edition 3

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

git_branch devel

git_last_commit a452942

git_last_commit_date 2026-04-28

Repository Bioconductor 3.24

Date/Publication 2026-06-04

Author Jiahua Tan [aut, cre] (ORCID: <https://orcid.org/0000-0001-5839-1049>),
Gian L. Negri [aut] (ORCID: <https://orcid.org/0000-0001-7722-8888>),
Gregg B. Morin [aut] (ORCID: <https://orcid.org/0000-0001-8949-4374>),
David D. Y. Chen [aut] (ORCID: <https://orcid.org/0000-0002-3669-6041>)

Maintainer Jiahua Tan <jiahuatan@chem.ubc.ca>

Contents

`aggregateByAttributes` *Aggregate PSMs to Protein Level*

Description

Aggregates PSMs using a random forest model.

Usage

```
aggregateByAttributes(
  PSM,
  colOfReporterIonInt,
  ranger = NULL,
  predError = NULL,
  ratioCalc = FALSE
)
```

Arguments

`PSM`	A data frame containing all PSMs to be aggregated.
`colOfReporterIonInt`	A vector of column names representing reporter ion intensities across different channels.
`ranger`	The random forest model to be applied for aggregation.
`predError`	The predicted level of inaccuracy for the PSMs, obtained from external sources. Either the `ranger` model or `predError` must be specified.
`ratioCalc`	A logical value indicating whether relative reporter intensities are calculated using the total reporter intensities across all channels.

Value

A data frame containing protein abundance estimates.

Examples

```
library(AWAggregatorData)
data(sample.PSM.FP)
regr <- loadQuantInaccuracyModel(useAvgCV=FALSE)
# Load sample names (Sample 1 ~ Sample 9)
samples <- colnames(sample.PSM.FP)[grep('Sample', colnames(sample.PSM.FP))]
groups <- samples
df <- getPSMAttributes(
    PSM=sample.PSM.FP,
    fixedPTMs=c('229.1629', '57.0214'),
    colOfReporterIonInt=samples,
    groups=groups,
    setProgressBar=TRUE
)
aggregated_results <- aggregateByAttributes(
    PSM=df,
    colOfReporterIonInt=samples,
    ranger=regr,
    ratioCalc=FALSE
)
```

---

convertPDFormat *Convert Proteome Discoverer Output to Required Input Format*

Description

Converts output from Proteome Discoverer into the input format required by AWAggregator.

Usage

```
convertPDFormat(PSM, protein, colOfReporterIonInt)
```

Arguments

`PSM`	A data frame containing the PSM table from Proteome Discoverer to be converted.
`protein`	A data frame containing the corresponding protein table from Proteome Discoverer.
`colOfReporterIonInt`	A vector of column names for reporter ion intensities across different channels.

Value

A data frame in the format required by AWAggregator.

Examples

```
data(sample.PSM.PD)
data(sample.prot.PD)
# Load sample names (Sample 1 ~ Sample 9)
samples <- colnames(sample.PSM.PD)[grep('Sample', colnames(sample.PSM.PD))]
df <- convertPDFormat(
    PSM=sample.PSM.PD,
    protein=sample.prot.PD,
    colOfReporterIonInt=samples
)
```

`fitQuantInaccuracyModel` *Fit a Random Forest Model*

Description

Trains a random forest model to predict the level of quantitative inaccuracy of PSMs.

Usage

```
fitQuantInaccuracyModel(
  PSM,
  numTrees = 500,
  useAvgCV = TRUE,
  importance = FALSE,
  seed,
  appliedAttributes = c(NA)
)
```

Arguments

`PSM`	A data frame containing all PSMs used for training.
`numTrees`	The number of trees to include in the random forest model.
`useAvgCV`	A logical value indicating whether to include the average CV attribute in the training. This parameter is ignored if `appliedAttributes` is specified.
`importance`	A logical value indicating whether to assess the importance of attributes.
`seed`	An integer seed for random number generation in the random forest.
`appliedAttributes`	A vector of attribute names to be used in training, replacing the default attributes.

Value

A trained random forest model.

Examples

```
library(ExperimentHub)
library(stringr)
eh <- ExperimentHub()
benchmarkSet3 <- eh[['EH9639']]
# Load sample names (Sample 'H1+Y0_1' ~ Sample 'H1+Y10_2')
samples <- colnames(benchmarkSet3)[
    grep('H1[+]Y[0-9]+_[1-2]', colnames(benchmarkSet3))
]
groups <- str_match(samples, 'H1[+]Y[0-9]+')[, 1]
PSM <- getPSMAttributes(
    PSM=benchmarkSet3,
    # TMTpro tag (304.2071) and N-ethylmaleimide (125.0476) are applied as
    # fixed PTMs
    fixedPTM=c('304.2071', '125.0476'),
    colOfReporterIonInt=samples,
    groups=groups
)
PSM <- getAvgScaledErrorOfLog2FC(
    PSM=PSM,
    colOfReporterIonInt=samples,
    groups=groups,
    expectedRelativeAbundance=list(
        `H1+Y0`=0, `H1+Y1`=1, `H1+Y5`=5, `H1+Y10`=10
    ),
    speciesAtConstLevel='HUMAN'
)
regr <- fitQuantInaccuracyModel(PSM, useAvgCV=TRUE, seed=3979)
```

`getAvgScaledErrorOfLog2FC` *Get Average Scaled Error of log2FC for PSMs in a Spike-in Dataset*

Description

Calculates the Average Scaled Error of log2FC values required for training sets.

Usage

```
getAvgScaledErrorOfLog2FC(
  PSM,
  colOfReporterIonInt,
  groups,
  expectedRelativeAbundance,
  speciesAtConstLevel
)
```

Arguments

`PSM`	A data frame containing all PSMs used for training.
`colOfReporterIonInt`	A vector of column names for reporter ion intensities across different channels.

groups A vector specifying sample groups.

expectedRelativeAbundance
A named list where group names are keys and the corresponding expected relative abundance values for species at varying concentrations are provided as values. Unknown ratios can be designated as NA.

speciesAtConstLevel
A string specifying the species that are spiked in at a constant level.

Value

A data frame containing PSMs with Average Scaled Error of log2FC values required for the random forest model.

Examples

```
library(ExperimentHub)
library(stringr)
eh <- ExperimentHub()
benchmarkSet1 <- eh[['EH9637']]
# Load sample names (Sample 'H1+E1_1' ~ Sample 'H1+E6_3')
samples <- colnames(benchmarkSet1)[
    grep('H1[+]E[0-9]+_[1-4]', colnames(benchmarkSet1))
]
groups <- str_match(samples, 'H1[+]E[0-9]+')[, 1]
PSM <- getAvgScaledErrorOfLog2FC(
    PSM=benchmarkSet1,
    colOfReporterIonInt=samples,
    groups=groups,
    expectedRelativeAbundance=list(`H1+E1`=1, `H1+E2`=2, `H1+E6`=NA),
    speciesAtConstLevel='HUMAN'
)
```

---

getDistMetric *Get Distance Metric*

Description

Calculates the distance metric for PSMs. Distance metric reflects on whether the quantified ratio of each pair of samples of a PSM diverges from other PSMs in the same redundant/unique group. Redundant group, unique group and distance metric were originally defined in the iPQF method. Please refer to "iPQF: a new peptide-to-protein summarization method using peptide spectra characteristics to improve protein quantification" for more details.

Usage

```
getDistMetric(PSM, channel, setProgressBar = TRUE)
```

Arguments

PSM A data frame containing the PSMs for which distance metrics are to be calculated.

channel A vector specifying the channels used for calculating the distance metric.

setProgressBar A logical value indicating whether to display a progress bar.

Value

A vector of distance metrics for the specified PSMs.

References

Martina Fischer, Bernhard Y. Renard (2016). iPQF: A New Peptide-to-Protein Summarization Method Using Peptide Spectra Characteristics to Improve Protein Quantification. *Bioinformatics*, 32(7), 1040-1047.

Examples

```
library(ExperimentHub)
eh <- ExperimentHub()
benchmarkSet3 <- eh[['EH9639']]
# Load sample names (Sample 'H1+Y0_1' ~ Sample 'H1+Y10_2')
samples <- colnames(benchmarkSet3)[
    grep('H1[+]Y[0-9]+_[1-2]', colnames(benchmarkSet3))
]
df <- getDistMetric(
    PSM=benchmarkSet3,
    channel=samples,
    setProgressBar=TRUE
)
```

---

getPSMAttributes *Get attributes for PSMs*

Description

Retrieves attributes required for training or test sets.

Usage

```
getPSMAttributes(
  PSM,
  fixedPTMs,
  colOfReporterIonInt,
  groups,
  groupsExcludedFromCV = NA,
  setProgressBar = TRUE
)
```

Arguments

PSM	A data frame containing all PSMs used for training.
fixedPTMs	A numeric vector with the masses of fixed post-translational modifications (PTMs) in Da. Other PTMs will be treated as variable PTMs.
colOfReporterIonInt	A vector of column names for reporter ion intensities across different channels.
groups	A vector specifying sample groups.

groupsExcludedFromCV
: A vector of sample groups excluded from average CV calculations, which may occur due to a zero spike-in concentration for a species.

setProgressBar
: A logical value indicating whether to display a progress bar.

Value

A data frame containing the PSM table with attributes required for the random forest model.

Examples

```
library(ExperimentHub)
library(stringr)
eh <- ExperimentHub()
benchmarkSet3 <- eh[['EH9639']]
# Load sample names (Sample 'H1+Y0_1' ~ Sample 'H1+Y10_2')
samples <- colnames(benchmarkSet3)[
    grep('H1[+]Y[0-9]+_[1-2]', colnames(benchmarkSet3))
]
groups <- str_match(samples, 'H1[+]Y[0-9]+')[, 1]
PSM <- getPSMAttributes(
    PSM=benchmarkSet3,
    fixedPTM=c('304.2071', '125.0476'),
    colOfReporterIonInt=samples,
    groups=groups,
    groupsExcludedFromCV='H1+Y0'
)
```

---

mergeTrainingSets *Merge Training Sets*

Description

Extracts a similar number of PSMs from each input dataset and merges them into a single training set.

Usage

```
mergeTrainingSets(PSMList, numPSMs, setProgressBar = TRUE)
```

Arguments

PSMList
: A named list where dataset names are keys and the corresponding data frames of PSMs used for training are values.

numPSMs
: The minimum number of PSMs to extract from each dataset for merging.

setProgressBar
: A logical value indicating whether to display a progress bar.

Value

A data frame containing the merged PSM table from a subset of each input dataset.

Examples

```
library(ExperimentHub)
eh <- ExperimentHub()
benchmarkSet1 <- eh[['EH9637']]
benchmarkSet2 <- eh[['EH9638']]
benchmarkSet3 <- eh[['EH9639']]
PSM <- mergeTrainingSets(
    PSMList=list(
        `Benchmark Set 1`=benchmarkSet1,
        `Benchmark Set 2`=benchmarkSet2,
        `Benchmark Set 3`=benchmarkSet3
    ),
    numPSMs=min(
        nrow(benchmarkSet1), nrow(benchmarkSet2), nrow(benchmarkSet3)
    ),
)
```

---

sample.prot.PD *Sample Protein Data from Protein Discoverer*

Description

This data frame represents sample proteins A0AV96, A0AVF1, A0AVT1, A0FGR8, and A0M8Q6, obtained from the search results of Proteome Discoverer. Columns unnecessary for the AWAggregator have been removed from the sample data.

Usage

```
data(sample.prot.PD)
```

Format

A data frame with 5 rows and 2 variables:

Accession The unique identifier assigned to the protein

Description The name of the protein exclusive of the identifier that appears in the Accession column

---

sample.PSM.FP *Sample PSM Data from FragPipe*

Description

This data frame represents sample peptide spectrum matches (PSMs) mapped to the proteins A0AV96, A0AVF1, A0AVT1, A0FGR8, and A0M8Q6, obtained from the search results of FragPipe. Columns unnecessary for the AWAggregator have been removed from the sample data.

Usage

```
data(sample.PSM.FP)
```

Format

A data frame with 118 rows and 20 variables:

Peptide Peptide amino acid sequence

Charge The charge state of the identified peptide

Calibrated Observed Mass Mass of the identified peptide after m/z calibration (in Da)

Calibrated Observed M/Z Mass-to-charge ratio of the peptide ion after m/z calibration

Delta Mass Difference between calibrated observed peptide mass and theoretical peptide mass (in Da)

Hyperscore The similarity score between observed and theoretical spectra

Number of Missed Cleavages Number of potential enzymatic cleavage sites within the identified sequence

Intensity Raw integrated precursor abundance for each PSM

Assigned Modifications Post-translational modifications within the identified sequence

Purity The proportion of total ion abundance in the inclusion window from the precursor

Protein Protein sequence header corresponding to the identified peptide sequence

Sample 1, Sample 2, Sample 3, Sample 4, Sample 5, Sample 6, Sample 7, Sample 8, Sample 9 Processed reporter ion intensities from sample 1 to 9

`sample.PSM.PD` *Sample PSM Data from Proteome Discoverer*

Description

This data frame represents sample peptide spectrum matches (PSMs) mapped to the proteins A0AV96, A0AVF1, A0AVT1, A0FGR8, and A0M8Q6, obtained from the search results of Proteome Discoverer. Columns unnecessary for the AWAggregator have been removed from the sample data.

Usage

```
data(sample.PSM.PD)
```

Format

A data frame with 128 rows and 21 variables:

Annotated Sequence The names of the flanking residues of a peptide in a protein

Modifications The static and dynamic modifications identified in the peptide

Number of Proteins The number of mapped proteins

Master Protein Accessions A description of the master proteins

Number of Missed Cleavages The number of potential enzymatic cleavage sites within the identified sequence

Charge The charge state of the peptide

mz in Da The mass-to-charge ratio of the precursor ion, in daltons

MHplus in Da The measured protonated monoisotopic mass of the peptides, in daltons

Delta mz in Da The difference between the measured charged mass (m/z in Da) and the theoretical mass of the same charge (z)

Isolation Interference in Percent The percentage of interference by co-isolation within the precursor isolation window

Average Reporter SN The average reporter S/N values

XCorr Scores the number of fragment ions that are common to two different peptides with the same precursor mass and calculates the cross-correlation score for all candidate peptides queried from the database

Sample 1, Sample 2, Sample 3, Sample 4, Sample 5, Sample 6, Sample 7, Sample 8, Sample 9 Processed reporter ion intensities from sample 1 to 9

Index