

# Package ‘MSstatsConvert’

April 27, 2026

**Title** Import Data from Various Mass Spectrometry Signal Processing  
Tools to MSstats Format

**Version** 1.21.6

**Description**

MSstatsConvert provides tools for importing reports of Mass Spectrometry data processing tools into R format suitable for statistical analysis using the MSstats and MSstatsTMT packages.

**License** Artistic-2.0

**Encoding** UTF-8

**LazyData** true

**Roxygen** list(markdown = TRUE)

**RoxygenNote** 7.3.3

**biocViews** MassSpectrometry, Proteomics, Software, DataImport,  
QualityControl

**Depends** R (>= 4.0)

**Imports** data.table, log4r, methods, checkmate, utils, stringi, Rcpp,  
parallel

**Suggests** tinytest, covr, knitr, arrow, rmarkdown

**LinkingTo** Rcpp

**Collate** 'clean\_ProteinProspector.R' 'clean\_Metamorpheus.R'  
'clean\_DIANN.R' 'clean\_Philosopher.R' 'clean\_Spectronaut.R'  
'clean\_SpectroMine.R' 'clean\_Skyline.R'  
'clean\_ProteomeDiscoverer.R' 'clean\_Progenesis.R'  
'clean\_OpenSWATH.R' 'clean\_OpenMS.R' 'clean\_MaxQuant.R'  
'clean\_DIAUmpire.R' 'MSstatsConvert\_core\_functions.R'  
'RcppExports.R' 'converters\_DIANNtoMSstatsFormat.R'  
'converters\_DIAUmpiretoMSstatsFormat.R'  
'converters\_FragPipetoMSstatsFormat.R'  
'converters\_MaxQtoMSstatsFormat.R'  
'converters\_MaxQtoMSstatsTMTFormat.R'  
'converters\_MetamorpheusToMSstatsFormat.R'  
'converters\_OpenMStoMSstatsFormat.R'  
'converters\_OpenMStoMSstatsTMTFormat.R'  
'converters\_OpenSWATHtoMSstatsFormat.R'  
'converters\_PDtoMSstatsFormat.R'  
'converters\_PDtoMSstatsTMTFormat.R'  
'converters\_PhilosophertoMSstatsTMTFormat.R'

'converters\_ProgenesistoMSstatsFormat.R'  
 'converters\_ProteinProspectortoMSstatsTMTFormat.R'  
 'converters\_SkylinetoMSstatsFormat.R'  
 'converters\_SpectroMinetoMSstatsTMTFormat.R'  
 'converters\_SpectronautoMSstatsFormat.R'  
 'utils\_MSstatsConvert.R' 'utils\_annotation.R'  
 'utils\_anomaly\_score.R' 'utils\_balanced\_design.R'  
 'utils\_checks.R' 'utils\_classes.R' 'utils\_clean\_features.R'  
 'utils\_data\_health.R' 'utils\_documentation.R'  
 'utils\_dt\_operations.R' 'utils\_filtering.R' 'utils\_fractions.R'  
 'utils\_logging.R' 'utils\_shared\_peptides.R'

**VignetteBuilder** knitr

**git\_url** <https://git.bioconductor.org/packages/MSstatsConvert>

**git\_branch** devel

**git\_last\_commit** bf0b062

**git\_last\_commit\_date** 2026-04-24

**Repository** Bioconductor 3.24

**Date/Publication** 2026-04-26

**Author** Mateusz Staniak [aut],  
 Devon Kohler [aut],  
 Anthony Wu [aut, cre],  
 Meena Choi [aut],  
 Ting Huang [aut],  
 Olga Vitek [aut]

**Maintainer** Anthony Wu <wu.anthon@northeastern.edu>

## Contents

.addFractions . . . . .	4
.adjustIntensities . . . . .	5
.aggregatePSMstoPeptideIons . . . . .	5
.checkAnnotation . . . . .	6
.checkDDA . . . . .	6
.checkDuplicatedMeasurements . . . . .	7
.checkMSstatsParams . . . . .	7
.checkMultiRun . . . . .	8
.checkOverlappedFeatures . . . . .	8
.cleanByFeature . . . . .	9
.cleanRawDIANN . . . . .	9
.cleanRawDIAUmpire . . . . .	11
.cleanRawMaxQuant . . . . .	11
.cleanRawMetamorpheus . . . . .	12
.cleanRawOpenMS . . . . .	12
.cleanRawOpenSWATH . . . . .	13
.cleanRawPD . . . . .	13
.cleanRawPDMSstats . . . . .	14
.cleanRawPDTMT . . . . .	15
.cleanRawPhilosopher . . . . .	15
.cleanRawProgenesis . . . . .	16

.cleanRawSkyline . . . . .	16
.cleanRawSpectroMineTMT . . . . .	17
.cleanRawSpectronaut . . . . .	17
.countCommonFeatures . . . . .	18
.fillValues . . . . .	19
.filterByPattern . . . . .	19
.filterByScore . . . . .	20
.filterExact . . . . .	21
.filterFewMeasurements . . . . .	21
.filterManyColumns . . . . .	22
.filterOverlapped . . . . .	23
.findAvailable . . . . .	23
.fixBasicColumns . . . . .	24
.fixColumnTypes . . . . .	24
.fixMissingValues . . . . .	25
.getChannelColumns . . . . .	25
.getDataTable . . . . .	26
.getFullDesign . . . . .	26
.getMissingRunsPerFeature . . . . .	27
.getOverlappingFeatures . . . . .	27
.getPhilosopherInput . . . . .	28
.handleFiltering . . . . .	28
.handleFractions . . . . .	29
.handleFractionsLF . . . . .	29
.handleFractionsTMT . . . . .	30
.handleIsotopicPeaks . . . . .	30
.handleSharedPeptides . . . . .	31
.handleSingleFeaturePerProtein . . . . .	31
.logConverterOptions . . . . .	32
.logSuccess . . . . .	32
.makeBalancedDesign . . . . .	33
.makeExactFilterMessage . . . . .	33
.makeScoreFilterMessage . . . . .	34
.mergeAnnotation . . . . .	35
.MSstatsFormat . . . . .	35
.nullAppender . . . . .	36
.onLoad . . . . .	36
.removeOverlappingFeatures . . . . .	37
.removeSharedPeptides . . . . .	37
.resolveFractionTies . . . . .	38
.selectMSstatsColumns . . . . .	38
.sharedParametersAmongConverters . . . . .	39
.standardizeColnames . . . . .	40
.summarizeMultipleMeasurements . . . . .	40
.summarizeMultiplePSMs . . . . .	41
.validateMSstatsConverterParameters . . . . .	41
as.data.frame.MSstatsValidated . . . . .	42
as.data.table.MSstatsValidated . . . . .	43
CheckDataHealth . . . . .	43
DIANNtoMSstatsFormat . . . . .	44
DIAUmpiretoMSstatsFormat . . . . .	47
FragPipetoMSstatsFormat . . . . .	49

getDataType . . . . .	50
getInputFile . . . . .	51
MaxQtoMSstatsFormat . . . . .	51
MaxQtoMSstatsTMTFormat . . . . .	53
MetamorpheusToMSstatsFormat . . . . .	55
MSstatsAnomalyScores . . . . .	56
MSstatsBalancedDesign . . . . .	57
MSstatsClean . . . . .	59
MSstatsConvert . . . . .	63
MSstatsImport . . . . .	64
MSstatsInputFiles-class . . . . .	65
MSstatsLogsSettings . . . . .	65
MSstatsMakeAnnotation . . . . .	66
MSstatsPreprocess . . . . .	67
MSstatsSaveSessionInfo . . . . .	69
OpenMStoMSstatsFormat . . . . .	70
OpenMStoMSstatsTMTFormat . . . . .	71
OpenSWATHtoMSstatsFormat . . . . .	72
PDtoMSstatsFormat . . . . .	74
PDtoMSstatsTMTFormat . . . . .	76
PhilosophertoMSstatsTMTFormat . . . . .	77
ProgenesistoMSstatsFormat . . . . .	79
ProteinProspectortoMSstatsTMTFormat . . . . .	81
SkylinetoMSstatsFormat . . . . .	82
SpectroMinetoMSstatsTMTFormat . . . . .	84
SpectronauttoMSstatsFormat . . . . .	85

<b>Index</b>	<b>89</b>
--------------	-----------

---

.addFractions	<i>Add a Fraction column to the output of MSstatsPreprocess</i>
---------------	---

---

### Description

Add a Fraction column to the output of MSstatsPreprocess

### Usage

```
.addFractions(input)
```

### Arguments

input	output of MSstatsPreprocess
-------	-----------------------------

### Value

data.table

---

.adjustIntensities      *Fix invalid intensities: infinite to NA, between 0 and 1 to 0*

---

### Description

Fix invalid intensities: infinite to NA, between 0 and 1 to 0

### Usage

```
.adjustIntensities(input)
```

### Arguments

input                      data.table

### Value

data.table

---

.aggregatePSMstoPeptideIons  
*Aggregate multiple PSMs to a single peptide ion.*

---

### Description

Aggregate multiple PSMs to a single peptide ion.

### Usage

```
.aggregatePSMstoPeptideIons(input, feature_columns, summary_function = sum)
```

### Arguments

input                      data.table preprocessed by one of the cleanRaw\* functions.

feature\_columns  
chr, names of columns that define features.

summary\_function  
function that will be used to aggregate intensities if needed.

### Value

data.table

---

<code>.checkAnnotation</code>	<i>Check if the annotation is valid</i>
-------------------------------	---

---

**Description**

Check if the annotation is valid

**Usage**

```
.checkAnnotation(input, annotation)
```

**Arguments**

<code>input</code>	data processed by the <code>MSstatsClean</code>
<code>annotation</code>	annotation created by the <code>MSstatsMakeAnnotation</code> function

**Value**

TRUE invisibly if the annotation is correct, throws an error otherwise

---

<code>.checkDDA</code>	<i>Check validity of DDA data</i>
------------------------	-----------------------------------

---

**Description**

Check validity of DDA data

**Usage**

```
.checkDDA(input)
```

**Arguments**

<code>input</code>	<code>data.table</code> preprocessed by one of the <code>cleanRaw*</code> functions.
--------------------	--

**Value**

logical

logical, TRUE means that the input dataset comes from a DDA experiment

---

`.checkDuplicatedMeasurements`

*Check if there are duplicated measurements within run*

---

### **Description**

Check if there are duplicated measurements within run

### **Usage**

```
.checkDuplicatedMeasurements(input)
```

### **Arguments**

input                    output of MSstatsPreprocess

### **Value**

character vector of feature labels

---

`.checkMSstatsParams`

*Check validity of parameters to the MSstatsImport function.*

---

### **Description**

Check validity of parameters to the MSstatsImport function.

### **Usage**

```
.checkMSstatsParams(  
  input,  
  annotation,  
  feature_columns,  
  remove_shared_peptides,  
  remove_single_feature_proteins,  
  feature_cleaning  
)
```

### **Value**

none, throws an error if any of the assertions fail

---

`.checkMultiRun`      *Check if fractionation exists*

---

**Description**

Check if fractionation exists

**Usage**

```
.checkMultiRun(input)
```

**Arguments**

input                  output of MSstatsPreprocess

**Value**

list of two elements: `has_fractions` (logical) indicates if fractions was detected in the input dataset, `is_risky` (logical) indicates if there was a problem with detecting fractionation.

---

`.checkOverlappedFeatures`  
*Check if any features are measured in multiple fractions*

---

**Description**

Check if any features are measured in multiple fractions

**Usage**

```
.checkOverlappedFeatures(input)
```

**Arguments**

input                  output of MSstatsPreprocess

**Value**

data.table

---

.cleanByFeature      *Perform by-feature operations.*

---

### Description

Perform by-feature operations.

### Usage

```
.cleanByFeature(  
  input,  
  feature_columns,  
  cleaning_control,  
  anomaly_metrics = c()  
)
```

### Arguments

`input`                    `data.table` preprocessed by one of the `cleanRaw*` functions.  
`feature_columns`            character vector of names of columns that define features.  
`cleaning_control`            named list of two or three elements. See the documentation for `MSstatsImport` for details.  
`anomaly_metrics`            character vector of quality metric column names to be used as features in an anomaly detection model.

### Value

`data.table`

---

.cleanRawDIANN      *Clean raw Diann files*

---

### Description

Clean raw Diann files

### Usage

```
.cleanRawDIANN(  
  msstats_object,  
  MBR = TRUE,  
  quantificationColumn = "FragmentQuantCorrected",  
  global_qvalue_cutoff = 0.01,  
  qvalue_cutoff = 0.01,  
  pg_qvalue_cutoff = 0.01,  
  calculateAnomalyScores = FALSE,  
  anomalyModelFeatures = c(),  
  labeledAminoAcids = NULL  
)
```

## Arguments

- `msstats_object` an object of class `MSstatsDIANNFiles`.
- `MBR` True if analysis was done with match between runs
- `quantificationColumn`  
Use 'FragmentQuantCorrected'(default) column for quantified intensities for DIANN 1.8.x. Use 'FragmentQuantRaw' for quantified intensities for DIANN 1.9.x. Use 'auto' for quantified intensities for DIANN 2.x where each fragment intensity is a separate column, e.g. `Fr0Quantity`.
- `global_qvalue_cutoff`  
The qvalue cutoff for the `Q.Value` column, i.e. the run-specific precursor q-value. Default is 0.01.
- `qvalue_cutoff` If `MBR` is false, the qvalue cutoff for the `Global.Q.Value` column, i.e. global precursor q-value. If `MBR` is true, the qvalue cutoff for the `Lib.Q.Value` column, i.e. the q-value for the library created after the first `MBR` pass. Default is 0.01.
- `pg_qvalue_cutoff`  
If `MBR` is false, the qvalue cutoff for the `Global.PG.Q.Value` column, i.e. the global q-value for the protein group. If `MBR` is true, the qvalue cutoff for the `Lib.PG.Q.Value` column, i.e. the protein group q-value for the library created after the first `MBR` pass. Default is 0.01.
- `calculateAnomalyScores`  
Default is `FALSE`. If `TRUE`, will run anomaly detection model and calculate anomaly scores for each feature. Used downstream to weigh measurements in differential analysis.
- `anomalyModelFeatures`  
character vector of quality metric column names to be used as features in the anomaly detection model. List must not be empty if `calculateAnomalyScores=TRUE`.
- `labeledAminoAcids`  
Character vector of single-letter amino acid codes that carry the SILAC label in protein turnover experiments, e.g. `c("K")` or `c("K", "R")`. Supplying this vector opts in to protein-turnover mode; the exact amino acids determine behaviour only in the `ModifiedSequence`-parsing path described below.  
**Channel-based path** (DIA-NN 2.x exports that include a `Channel` column): when `labeledAminoAcids` is non-NULL *and* the input contains a `Channel` column, `Channel` values are mapped directly to `IsotopeLabelType` ("H" → "H", "L" → "L", anything else → NA). The amino acid codes in `labeledAminoAcids` are **not** used to validate or filter `ModifiedSequence` in this path.  
**ModifiedSequence-parsing path** (DIA-NN 1.x exports without a `Channel` column): when `labeledAminoAcids` is non-NULL and no `Channel` column is present, each `ModifiedSequence` is inspected for SILAC suffixes of the form (SILAC-<AA>-H) or (SILAC-<AA>-L), where <AA> is one of the supplied amino acid codes. Matching sequences are classified as "H" or "L"; sequences carrying neither suffix receive `IsotopeLabelType = NA`. The SILAC suffix is then stripped from `PeptideSequence`.  
When NULL (default), protein-turnover mode is disabled and all peptides receive `IsotopeLabelType = "Light"`.

## Value

`data.table`

---

.cleanRawDIAUmpire      *Clean raw DIAUmpire files*

---

**Description**

Clean raw DIAUmpire files

**Usage**

```
.cleanRawDIAUmpire(msstats_object, use_frag, use_pept)
```

**Arguments**

msstats\_object    Object that inherits from MSstatsInputFiles class.  
use\_frag            TRUE will use the selected fragment for each peptide. 'Selected\_fragments' column is required.  
use\_pept            TRUE will use the selected fragment for each protein 'Selected\_peptides' column is required.

**Value**

data.table

---

.cleanRawMaxQuant      *Clean raw output from MaxQuant*

---

**Description**

Clean raw output from MaxQuant

**Usage**

```
.cleanRawMaxQuant(  
  msstats_object,  
  protein_id_col,  
  remove_by_site = FALSE,  
  channel_columns = "Reporterintensitycorrected"  
)
```

**Arguments**

msstats\_object    object that inherits from MSstatsInputFiles class.  
protein\_id\_col    character, name of a column with names of proteins.  
remove\_by\_site    logical, if TRUE, proteins only identified by site will be removed.  
channel\_columns    character, regular expression that identifies channel columns in TMT data.

**Value**

data.table

---

*.cleanRawMetamorpheus* *Clean raw Metamorpheus files*

---

**Description**

Clean raw Metamorpheus files

**Usage**

```
.cleanRawMetamorpheus(msstats_object, MBR = TRUE, qvalue_cutoff = 0.05)
```

**Arguments**

`msstats_object` an object of class `MSstatsMetamorpheusFiles`.  
`MBR` If TRUE, the function will include peaks detected by MBR  
`qvalue_cutoff` The q-value cutoff for filtering peaks detected by MBR

**Value**

data.table

---

*.cleanRawOpenMS* *Clean raw output from OpenMS*

---

**Description**

Clean raw output from OpenMS

**Usage**

```
.cleanRawOpenMS(msstats_object)
```

**Arguments**

`msstats_object` an object of class `MSstatsSpectroMineFiles`.

**Value**

data.table

---

`.cleanRawOpenSWATH`      *Clean raw OpenSWATH files*

---

**Description**

Clean raw OpenSWATH files

**Usage**

```
.cleanRawOpenSWATH(msstats_object)
```

**Arguments**

`msstats_object` an object of class `MSstatsSpectroMineFiles`.

**Value**

`data.table`

---

`.cleanRawPD`                      *Clean raw Proteome Discoverer data*

---

**Description**

Clean raw Proteome Discoverer data

**Usage**

```
.cleanRawPD(  
  msstats_object,  
  quantification_column,  
  protein_id_column,  
  sequence_column,  
  remove_shared,  
  remove_protein_groups = TRUE,  
  intensity_columns_regexp = "Abundance"  
)
```

**Arguments**

`msstats_object` an object of class `MSstatsSpectroMineFiles`.

`quantification_column`  
chr, name of a column used for quantification.

`protein_id_column`  
chr, name of a column with protein IDs.

`sequence_column`  
chr, name of a column with peptide sequences.

`remove_shared` lgl, if TRUE, shared peptides will be removed.

`remove_protein_groups`  
 if TRUE, proteins with `numProteins > 1` will be removed.

`intensity_columns_regex`  
 regular expressions that defines intensity columns. Defaults to "Abundance", which means that columns that contain the word "Abundance" will be treated as corresponding to intensities for different channels.

**Value**

data.table

---

`.cleanRawPDMSstats`     *Clean raw PD output*

---

**Description**

Clean raw PD output

**Usage**

```
.cleanRawPDMSstats(
  msstats_object,
  quantification_column,
  protein_id_column,
  sequence_column,
  remove_shared,
  run_column = "SpectrumFile"
)
```

**Arguments**

`msstats_object`    an object of class `MSstatsSpectroMineFiles`.

`quantification_column`  
                     chr, name of a column used for quantification.

`protein_id_column`  
                     chr, name of a column with protein IDs.

`sequence_column`  
                     chr, name of a column with peptide sequences.

`remove_shared`    lgl, if TRUE, shared peptides will be removed.

**Value**

data.table

---

.cleanRawPDTMT      *Clean raw TMT data from Proteome Discoverer*

---

### Description

Clean raw TMT data from Proteome Discoverer

### Usage

```
.cleanRawPDTMT(  
  msstats_object,  
  remove_shared = TRUE,  
  remove_protein_groups = TRUE,  
  protein_id_column = "ProteinAccessions",  
  intensity_columns_regexp = "Abundance",  
  run_column = "SpectrumFile"  
)
```

### Arguments

`msstats_object` an object of class `MSstatsSpectroMineFiles`.  
`remove_shared` `lgl`, if `TRUE`, shared peptides will be removed.  
`remove_protein_groups`  
    if `TRUE`, proteins with `numProteins > 1` will be removed.  
`protein_id_column`  
    `chr`, name of a column with protein IDs.  
`intensity_columns_regexp`  
    regular expressions that defines intensity columns. Defaults to "Abundance", which means that columns that contain the word "Abundance" will be treated as corresponding to intensities for different channels.

### Value

`data.table`

---

.cleanRawPhilosopher      *Clean raw Philosopher files*

---

### Description

Clean raw Philosopher files

### Usage

```
.cleanRawPhilosopher(  
  msstats_object,  
  protein_id_col,  
  peptide_id_col,  
  channels,  
  remove_shared_peptides  
)
```

**Arguments**

msstats\_object object of class MSstatsPhilosopherFiles  
 protein\_id\_col character name of a column that identifies proteins  
 peptide\_id\_col character name of a column that identifies peptides  
 channels character vector of channel labels  
 remove\_shared\_peptides logical, if TRUE, shared peptides will be removed based on the IsUnique column from Philosopher output

**Value**

data.table

---

*.cleanRawProgenesis* *Clean raw Progenesis output*

---

**Description**

Clean raw Progenesis output

**Usage**

```
.cleanRawProgenesis(msstats_object, runs, fix_colnames = TRUE)
```

**Arguments**

msstats\_object an object of class MSstatsSpectroMineFiles.  
 runs chr, vector of Run labels.  
 fix\_colnames lgl, if TRUE, one of the rows will be used as colnames.

**Value**

data.table

---

*.cleanRawSkyline* *Clean raw data from Skyline*

---

**Description**

Clean raw data from Skyline

**Usage**

```
.cleanRawSkyline(msstats_object)
```

**Arguments**

msstats\_object an object of class MSstatsSpectroMineFiles.

**Value**

data.table

---

```
.cleanRawSpectroMineTMT
```

*Clean raw SpectroMine TMT data*

---

### Description

Clean raw SpectroMine TMT data

### Usage

```
.cleanRawSpectroMineTMT(msstats_object)
```

### Arguments

msstats\_object an object of class MSstatsSpectroMineFiles.

### Value

data.table

---

```
.cleanRawSpectronaut
```

*Clean raw Spectronaut output.*

---

### Description

Clean raw Spectronaut output.

### Usage

```
.cleanRawSpectronaut(  
  msstats_object,  
  intensity,  
  calculateAnomalyScores,  
  anomalyModelFeatures,  
  peptideSequenceColumn = "EG.ModifiedSequence",  
  heavyLabels = NULL  
)
```

### Arguments

msstats\_object an object of class MSstatsSpectronautFiles.

intensity Intensity column to use. Accepts legacy enum values 'PeakArea' (default, uses F.PeakArea), 'NormalizedPeakArea' (uses F.NormalizedPeakArea). Can also be any raw Spectronaut column name passed as a string (e.g. "FG.MS1Quantity"); the column name is standardized internally. For protein turnover workflows the recommended default is "FG.MS1Quantity".

calculateAnomalyScores Default is FALSE. If TRUE, will run anomaly detection model and calculate anomaly scores for each feature. Used downstream to weigh measurements in differential analysis.

<code>anomalyModelFeatures</code>	character vector of quality metric column names to be used as features in the anomaly detection model. List must not be empty if <code>calculateAnomalyScores=TRUE</code> .
<code>peptideSequenceColumn</code>	Name of the Spectronaut column that contains the peptide sequence. Defaults to <code>"EG.ModifiedSequence"</code> . The value is standardized internally (dots and spaces removed) before column lookup.
<code>heavyLabels</code>	Character list identifying the heavy isotope labels as it appears inside square brackets in the peptide sequence column, e.g. <code>c("Lys6")</code> matches peptides containing <code>[Lys6]</code> . <code>c("Lys6", "Arg10")</code> matches peptides containing either <code>[Lys6]</code> or <code>[Arg10]</code> . Supports any novel label name reported by Spectronaut (e.g. <code>"Leu6"</code> , <code>"Phe10"</code> ). When provided, peptides are classified as heavy ( <code>IsotopeLabelType = "H"</code> ), light ( <code>IsotopeLabelType = "L"</code> ), or unlabeled ( <code>IsotopeLabelType = NA</code> ) based on its labeled sequence. When <code>NULL</code> (default) all peptides receive <code>IsotopeLabelType = "L"</code> . Useful for protein turnover experiments.

**Value**

`data.table`

---

`.countCommonFeatures` *Get common values from two vectors of features*

---

**Description**

Get common values from two vectors of features

**Usage**

```
.countCommonFeatures(features_1, features_2)
```

**Arguments**

<code>features_1</code>	vector of feature names
<code>features_2</code>	vector of feature_names

**Value**

character vector of common values of `features_1` and `features_2`

---

.fillValues                      *Set column to a single value*

---

**Description**

Set column to a single value

**Usage**

```
.fillValues(input, fill_list)
```

**Arguments**

input	data.table preprocessed by one of the cleanRaw* functions.
fill_list	named list, names correspond to column names, elements to values that will be used in the columns.

**Value**

data.table

---

.filterByPattern                      *Handle filtering by pattern*

---

**Description**

Handle filtering by pattern

**Usage**

```
.filterByPattern(input, col_name, patterns, filter, drop)
```

**Arguments**

input	data.table preprocessed by one of the .cleanRaw* functions.
col_name	chr, name of the column with peptide sequences.
patterns	chr, regular expression - matching peptides will be removed from the data.
filter	lgl, if TRUE, peptides will be actually filtered.
drop	lgl, if TRUE, the column will be dropped.

**Value**

data.table

---

`.filterByScore`      *Filter PSMs / proteins by a given score column.*

---

### **Description**

Filter PSMs / proteins by a given score column.

### **Usage**

```
.filterByScore(  
  input,  
  score_column,  
  score_threshold,  
  direction,  
  behavior,  
  handle_na = "keep",  
  fill_value = NA,  
  filter = TRUE,  
  drop = TRUE  
)
```

### **Arguments**

<code>input</code>	data.table preprocessed by one of the <code>.cleanRaw*</code> functions.
<code>score_column</code>	chr, name of the column that contains scores.
<code>score_threshold</code>	num, values below or above this threshold will be removed from the data.
<code>direction</code>	chr, if "greater" only values above the threshold will be retained, if "smaller" - below the threshold.
<code>behavior</code>	chr, if "remove", values below/above the threshold will be removed, if "replace", they will be set to <code>fill_value</code> .
<code>fill_value</code>	if <code>behavior = "replace"</code> , values below/above the threshold will be replaced with <code>fill_value</code> . Defaults to NA.
<code>filter</code>	If TRUE, filtering will be performed.
<code>drop</code>	if TRUE, <code>score_column</code> will be removed.

### **Value**

data.table

---

.filterExact                      *Filter out specified symbols.*

---

**Description**

Filter out specified symbols.

**Usage**

```
.filterExact(  
  input,  
  col_name,  
  filter_symbols,  
  behavior,  
  fill_value,  
  filter,  
  drop  
)
```

**Arguments**

input	data.table preprocessed by one of the .cleanRaw* functions.
col_name	chr, name of the column that will be the base for filtering
filter_symbols	character vector of symbols that will be removed
behavior	chr, if "remove", values below/above the threshold will be removed, if "replace", they will be set to fill_value.
fill_value	if behavior = "replace", values below/above the threshold will be replaced with fill_value. Defaults to NA.
filter	lgl, if TRUE, decoy proteins will be removed from the data.
drop	lgl, if TRUE, column that contains decoy proteins will be dropped.

**Value**

data.table

---

.filterFewMeasurements  
*Remove features with a small number of (non-missing) measurements across runs*

---

**Description**

Remove features with a small number of (non-missing) measurements across runs

**Usage**

```
.filterFewMeasurements(
  input,
  min_intensity,
  remove_few,
  feature_columns = NULL
)
```

**Arguments**

`input` `data.table` pre-processed by one of the `.cleanRaw*` functions.

`min_intensity` minimum intensity that will be considered non-missing.

`remove_few` logical, if TRUE, features that have less than three measurements will be removed. If FALSE, only features with all missing runs will be removed.

`feature_columns` chr, vector of names of columns that define features.

**Value**

`data.table`

---

`.filterManyColumns` *Filter rows that contain specified symbols in multiple columns.*

---

**Description**

Filter rows that contain specified symbols in multiple columns.

**Usage**

```
.filterManyColumns(input, filter_columns, filter_symbols)
```

**Arguments**

`input` `data.table` preprocessed by one of the `cleanRaw*` functions.

`filter_columns` chr, names of columns in which elements will be matched and removed.

`filter_symbols` chr, vector of strings. Rows with corresponding elements in `filter_columns` will be removed.

**Value**

`data.table`

---

.filterOverlapped      *Remove overlapped features*

---

**Description**

Remove overlapped features

**Usage**

.filterOverlapped(input, summary\_function, overlapped\_features)

**Arguments**

input                    data.table preprocessed by one of the .cleanRaw\* functions and merged with annotation.  
summary\_function        summary function (mean, sum, max) that will be used to pick one feature from multiple overlapping features  
overlapped\_features     features that overlap.

**Value**

data.table

---

.findAvailable            *Select an available options from a set of possibilities*

---

**Description**

Select an available options from a set of possibilities

**Usage**

.findAvailable(possibilities, option\_set, fall\_back = NULL)

**Arguments**

possibilities    possible legal values of a variable  
option\_set        set of values that includes one of the possibilities  
fall\_back        if there is none of the possibilities in option\_set, or there are multiple hits, default to fall\_back

**Value**

same as option\_set, usually character

---

<code>.fixBasicColumns</code>	<i>Remove underscores from sequences and change intensity type to numeric</i>
-------------------------------	---

---

**Description**

Remove underscores from sequences and change intensity type to numeric

**Usage**

```
.fixBasicColumns(input)
```

**Arguments**

<code>input</code>	<code>data.table</code>
--------------------	-------------------------

**Value**

`data.table`

---

<code>.fixColumnTypes</code>	<i>Change classes of multiple columns</i>
------------------------------	---

---

**Description**

Change classes of multiple columns

**Usage**

```
.fixColumnTypes(
  input,
  numeric_columns = NULL,
  character_columns = NULL,
  factor_columns = NULL
)
```

**Arguments**

<code>input</code>	<code>data.table</code> preprocessed by one of the <code>cleanRaw*</code> functions.
<code>numeric_columns</code>	<code>chr</code> , vector of names of columns that will be converted to numeric.
<code>character_columns</code>	<code>chr</code> , vector of names of columns that will be converted to character.
<code>factor_columns</code>	<code>chr</code> , vector of names of columns that will be converted to factor.

**Value**

`data.table`

---

.fixMissingValues      *Change labels for missing values*

---

**Description**

Change labels for missing values

**Usage**

```
.fixMissingValues(input, fix_missing = NULL)
```

**Arguments**

input	output of MSstatsPreprocess
fix_missing	missing values can be labeled by NA, 0 or both. If NULL, data were processed by Skyline, so missing values will be denoted by both NA and 0. If "na_to_zero", NA values will be replaced by 0. If "zero_to_na", 0 values will be replaced by NA

**Value**

data.table

---

.getChannelColumns      *Get intensity columns from wide-format data*

---

**Description**

Get intensity columns from wide-format data

**Usage**

```
.getChannelColumns(col_names, ...)
```

**Arguments**

col_names	names of columns, where some of the columns store intensity value for different channels
...	varying number of strings that define channel columns.

**Value**

character vector of column names that correspond to channel intensities

---

<code>.getDataTable</code>	<i>Read file from a provided path or convert given data.frame to data.table</i>
----------------------------	---

---

**Description**

Read file from a provided path or convert given data.frame to data.table

**Usage**

```
.getDataTable(input, ...)
```

**Arguments**

<code>input</code>	report from a signal processing tool or a path to it
<code>...</code>	additional parameters for <code>data.table::fread</code>

**Value**

data.table

---

<code>.getFullDesign</code>	<i>Create a data.frame of each combination of values for given variables</i>
-----------------------------	--

---

**Description**

Create a data.frame of each combination of values for given variables

**Usage**

```
.getFullDesign(input, group_col, feature_col, measurement_col, is_tmt)
```

**Arguments**

<code>input</code>	output of <code>MSstatsPreprocess</code>
<code>group_col</code>	name of column in input. Combination of values of <code>feature_col</code> and <code>measurement_col</code> will be created within each unique value of this column
<code>is_tmt</code>	if TRUE, data will be treated as coming from TMT experiment.
<code>'feature_col'</code>	name of the column that labels features
<code>'measurement_col'</code>	name of a column with measurement labels - Runs in label-free case, Channels in TMT case.

**Value**

data.table

---

`.getMissingRunsPerFeature`  
*Get names of missing runs*

---

**Description**

Get names of missing runs

**Usage**

`.getMissingRunsPerFeature(input)`

**Arguments**

input                      output of MSstatsPreprocess

**Value**

data.table

---

`.getOverlappingFeatures`  
*Get features that are overlapped among multiple runs*

---

**Description**

Get features that are overlapped among multiple runs

**Usage**

`.getOverlappingFeatures(input)`

**Arguments**

input                      data.table preprocessed by one of the `.cleanRaw*` functions and merged with annotation.

**Value**

data.table

---

`.getPhilosopherInput`    *Convert Philosopher parameters to consistent format*

---

**Description**

Convert Philosopher parameters to consistent format

**Usage**

```
.getPhilosopherInput(input, path, folder)
```

**Arguments**

<code>input</code>	data.frame of msstats.csv file produced by Philosopher
<code>path</code>	character. Path to a file or directory containing msstats.csv output(s) from Philosopher. Used when input is NULL.
<code>folder</code>	logical. If TRUE, path is treated as a directory and all msstats files within it are read. If FALSE, path is treated as a single file path.

---

`.handleFiltering`    *Handle PSM/proteins scores*

---

**Description**

Handle PSM/proteins scores

**Usage**

```
.handleFiltering(input, score_filtering, exact_filtering, pattern_filtering)
```

**Arguments**

<code>input</code>	data.table preprocessed by one of the <code>.cleanRaw*</code> functions.
<code>score_filtering</code>	list of by-score filtering controls.
<code>exact_filtering</code>	list of exact filtering controls.
<code>pattern_filtering</code>	list of by-pattern filtering controls.

**Value**

data.table

---

.handleFractions      *Check if there are overlapping features and remove if needed*

---

**Description**

Check if there are overlapping features and remove if needed

**Usage**

```
.handleFractions(input)
```

**Arguments**

input      data.table preprocessed by one of the .cleanRaw\* functions and merged with annotation.

**Value**

data.table

---

.handleFractionsLF      *Handle overlapping features*

---

**Description**

Handle overlapping features

**Usage**

```
.handleFractionsLF(input)
```

**Arguments**

input      output of MSstatsPreprocess

**Value**

data.table

---

`.handleFractionsTMT`    *Remove peptide ions overlapped among multiple fractions of the same biological mixture*

---

**Description**

Remove peptide ions overlapped among multiple fractions of the same biological mixture

**Usage**

```
.handleFractionsTMT(input)
```

**Arguments**

`input`                    `data.table` preprocessed by one of the `.cleanRaw*` functions and merged with annotation.

**Value**

`data.table`

---

`.handleIsotopicPeaks`    *Handle isotopic peaks*

---

**Description**

Handle isotopic peaks

**Usage**

```
.handleIsotopicPeaks(input, aggregate = FALSE)
```

**Arguments**

`input`                    `data.table` preprocessed by one of the `cleanRaw*` functions.  
`aggregate`                if TRUE, isotopic peaks will be summed.

**Value**

`data.table`

---

*.handleSharedPeptides* *Handle shared peptides.*

---

**Description**

Handle shared peptides.

**Usage**

```
.handleSharedPeptides(  
  input,  
  remove_shared = TRUE,  
  protein_column = "ProteinName",  
  peptide_column = "PeptideSequence"  
)
```

**Arguments**

`input` data.table pre-processed by one of the `.cleanRaw*` functions.  
`remove_shared` lgl, if TRUE, shared peptides will be removed  
`protein_column` chr, name of the column with names of proteins.  
`peptide_column` chr, name of the column with peptide sequences.

**Value**

data.table

---

*.handleSingleFeaturePerProtein*  
*Remove proteins only identified by a single feature*

---

**Description**

Remove proteins only identified by a single feature

**Usage**

```
.handleSingleFeaturePerProtein(input, remove_single_feature)
```

**Arguments**

`input` data.table pre-processed by one of the `.cleanRaw*` functions.  
`remove_single_feature` lgl, if TRUE, proteins with a single feature will be removed.

**Value**

data.table

---

`.logConverterOptions`    *Log information about converter options*

---

### Description

Log information about converter options

### Usage

```
.logConverterOptions(
  feature_columns,
  remove_shared_peptides,
  remove_single_feature_proteins,
  feature_cleaning,
  is_tmt = FALSE
)
```

### Arguments

`feature_columns`  
character vector of names of columns that define spectral features.

`remove_shared_peptides`  
logical, if TRUE shared peptides will be removed.

`remove_single_feature_proteins`  
logical, if TRUE, proteins that only have one feature will be removed.

`feature_cleaning`  
named list with maximum two (for MSstats converters) or three (for MSstatsTMT converter) elements. If `handle_few_measurements` is set to "remove", feature with less than three measurements will be removed (otherwise it should be equal to "keep"). `summarize_multiple_psms` is a function that will be used to aggregate multiple feature measurements in a run. It should return a scalar and accept an `na.rm` parameter. For MSstatsTMT converters, setting `remove_psms_with_any_missing` will remove features which have missing values in a run from that run.

`is_tmt`            If TRUE, the dataset comes from a TMT experiment

### Value

TRUE invisibly if message was logged

---

`.logSuccess`            *Make a message about successful data cleaning/importing*

---

### Description

Make a message about successful data cleaning/importing

**Usage**

```
.logSuccess(tool, event)
```

**Arguments**

tool                    name of a signal processing tool

**Value**

TRUE invisibly if logging was successful

---

`.makeBalancedDesign`    *Fill missing rows to create balanced design*

---

**Description**

Fill missing rows to create balanced design

**Usage**

```
.makeBalancedDesign(input, fill_missing, anomaly_metrics = c())
```

**Arguments**

input                    output of MSstatsPreprocess  
fill\_missing            if TRUE, missing Intensities values will be added to data  
anomaly\_metrics                    character vector of quality metric column names to be used as features in an anomaly detection model. and marked as NA

**Value**

data.table

---

`.makeExactFilterMessage`                    *Make a message about filtering based on fixed values*

---

**Description**

Make a message about filtering based on fixed values

**Usage**

```
.makeExactFilterMessage(col_name, filter_symbols, behavior, fill_value)
```

**Arguments**

<code>col_name</code>	chr, name of the column that will be the base for filtering
<code>filter_symbols</code>	character vector of symbols that will be removed
<code>behavior</code>	chr, if "remove", values below/above the threshold will be removed, if "replace", they will be set to <code>fill_value</code> .
<code>fill_value</code>	if <code>behavior = "replace"</code> , values below/above the threshold will be replaced with <code>fill_value</code> . Defaults to NA.

**Value**

character - message

---

`.makeScoreFilterMessage`

*Make a message about filtering based on a score*

---

**Description**

Make a message about filtering based on a score

**Usage**

```
.makeScoreFilterMessage(
  score_column,
  score_threshold,
  direction,
  behavior,
  fill_value
)
```

**Arguments**

<code>score_column</code>	chr, name of the column that contains scores.
<code>score_threshold</code>	num, values below or above this threshold will be removed from the data.
<code>direction</code>	chr, if "greater" only values above the threshold will be retained, if "smaller" - below the threshold.
<code>behavior</code>	chr, if "remove", values below/above the threshold will be removed, if "replace", they will be set to <code>fill_value</code> .
<code>fill_value</code>	if <code>behavior = "replace"</code> , values below/above the threshold will be replaced with <code>fill_value</code> . Defaults to NA.

**Value**

character - message

---

`.mergeAnnotation`      *Merge annotation with feature data*

---

**Description**

Merge annotation with feature data

**Usage**

```
.mergeAnnotation(input, annotation)
```

**Arguments**

`annotation`      `data.table` with annotation  
`data.table`      preprocessed by one of the `.cleanRaw` functions.

**Value**

`data.table`

---

`.MSstatsFormat`      *Output format for further analysis by MSstats*

---

**Description**

Output format for further analysis by MSstats

**Usage**

```
.MSstatsFormat(input, anomaly_metrics = c())
```

**Arguments**

`input`      `data.table`  
`anomaly_metrics`      character vector of quality metric column names to be used as features in an anomaly detection model

**Value**

object of class `MSstatsValidated` that inherits from `data.frame`

---

.nullAppender	<i>log4r appender used not to write messages</i>
---------------	--

---

**Description**

A convenience function written to save time on checking if messages should be printed or logs should be written to a file.

**Usage**

```
.nullAppender(level, ...)
```

**Arguments**

level	log level
...	messages - ignored

**Value**

NULL invisibly

---

.onLoad	<i>Set default logging object when package is loaded</i>
---------	--

---

**Description**

Set default logging object when package is loaded

**Usage**

```
.onLoad(...)
```

**Arguments**

...	ignored
-----	---------

**Value**

none, sets options called MSstatsLog and MSstatsMsg

---

.removeOverlappingFeatures

*Replace intensities of overlapped fractions with NA, keeping only one fraction*

---

### **Description**

Replace intensities of overlapped fractions with NA, keeping only one fraction

### **Usage**

```
.removeOverlappingFeatures(input)
```

### **Arguments**

input                      output of MSstatsPreprocess

### **Value**

data.table

---

.removeSharedPeptides *Remove peptides assigned to more than one protein.*

---

### **Description**

Remove peptides assigned to more than one protein.

### **Usage**

```
.removeSharedPeptides(input, protein_column, peptide_column)
```

### **Arguments**

input                      data.table pre-processed by one of the .cleanRaw\* functions.  
protein\_column    chr, name of the column with names of proteins.  
peptide\_column    chr, name of the column with peptide sequences.

### **Value**

data.table

---

`.resolveFractionTies` *Resolve ties when multiple fractions share the maximum number of measurements for a given feature. In the case of a tie, the fraction with the highest mean intensity is selected.*

---

**Description**

Resolve ties when multiple fractions share the maximum number of measurements for a given feature. In the case of a tie, the fraction with the highest mean intensity is selected.

**Usage**

```
.resolveFractionTies(input, max_fractions)
```

**Arguments**

<code>input</code>	output of <code>MSstatsPreprocess</code>
<code>max_fractions</code>	<code>data.table</code> of fractions that share the maximum number of unique runs per feature, as produced by <code>.removeOverlappingFeatures</code>

**Value**

`data.table` with columns `feature` and `Fraction`, containing one selected fraction per feature

---

`.selectMSstatsColumns` *Select columns for MSstats format*

---

**Description**

Select columns for MSstats format

**Usage**

```
.selectMSstatsColumns(input, anomaly_metrics)
```

**Arguments**

<code>input</code>	<code>data.table</code>
--------------------	-------------------------

**Value**

`data.table`

---

.sharedParametersAmongConverters

*A dummy function to store shared documentation items for converters.*

---

## Description

A dummy function to store shared documentation items for converters.

## Usage

.sharedParametersAmongConverters()

## Arguments

removeFewMeasurements  
TRUE (default) will remove the features that have 1 or 2 measurements across runs.

useUniquePeptide  
TRUE (default) removes peptides that are assigned for more than one proteins. We assume to use unique peptide for each protein.

summaryforMultipleRows  
max or sum - when there are multiple measurements for certain feature and certain run, use highest or sum of multiple intensities. Default is max for label-free converters and sum for TMT converters.

removeProtein\_with1Feature  
TRUE will remove the proteins which have only 1 feature, which is the combination of peptide, precursor charge, fragment and charge. FALSE is default.

removeProtein\_with1Peptide  
TRUE will remove the proteins which have only 1 peptide and charge. FALSE is default.

removeOxidationMpeptides  
TRUE will remove the peptides including 'oxidation (M)' in modification. FALSE is default.

removeMpeptides  
TRUE will remove the peptides including 'M' sequence. FALSE is default.

use\_log\_file  
logical. If TRUE, information about data processing will be saved to a file.

append  
logical. If TRUE, information about data processing will be added to an existing log file.

verbose  
logical. If TRUE, information about data processing will be printed to the console.

log\_file\_path  
character. Path to a file to which information about data processing will be saved. If not provided, such a file will be created automatically. If append = TRUE, has to be a valid path to a file.

...  
additional parameters to data.table::fread.

---

`.standardizeColnames` *Change column names to match read.table/read.csv/read.delim conventions*

---

**Description**

Change column names to match read.table/read.csv/read.delim conventions

**Usage**

```
.standardizeColnames(col_names)
```

**Arguments**

`col_names` chr, vector of column names

**Value**

character vector

---

`.summarizeMultipleMeasurements`  
*Summarize multiple measurements per feature in a single run*

---

**Description**

Summarize multiple measurements per feature in a single run

**Usage**

```
.summarizeMultipleMeasurements(  
  input,  
  aggregator,  
  feature_columns,  
  anomaly_metrics = c()  
)
```

**Arguments**

`input` data.table pre-processed by one of the `.cleanRaw*` functions.  
`aggregator` function that will be used to aggregate duplicated values.  
`feature_columns` chr, vector of names of columns that define features.  
`anomaly_metrics` character vector of quality metric column names to be used as features in an anomaly detection model.

**Value**

data.table

---

.summarizeMultiplePSMs

*Pick one PSM from a data.table of several PSMs.*

---

### Description

Pick one PSM from a data.table of several PSMs.

### Usage

```
.summarizeMultiplePSMs(input, summary_function)
```

### Arguments

input                    data.table preprocessed by one of the .cleanRaw\* functions.  
summary\_function        function that will be used to aggregate intensities if needed.

### Value

character - label of a chosen PSM

---

.validateMSstatsConverterParameters

*Generic parameter validation for all MSstats converters using configuration object*

---

### Description

Generic parameter validation for all MSstats converters using configuration object

### Usage

```
.validateMSstatsConverterParameters(config)
```

### Arguments

config                    A list containing all converter parameters. See details for required structure.

### Details

The config list should contain the input and optionally other parameters:

- input: input data (required)
- annotation: annotation data (optional)
- intensity: intensity type (optional)
- filter\_with\_Qvalue: Q-value filter setting (default: FALSE)
- qvalue\_cutoff: Q-value cutoff (default: 0.01)

- useUniquePeptide: unique peptide setting (default: TRUE)
- removeFewMeasurements: remove few measurements setting (default: TRUE)
- removeProtein\_with1Feature: remove single feature proteins setting (default: FALSE)
- summaryforMultipleRows: aggregation function (default: max)
- calculateAnomalyScores: anomaly detection setting (default: FALSE)
- anomalyModelFeatures: anomaly model features (default: c())
- anomalyModelFeatureTemporal: temporal features (default: c())
- removeMissingFeatures: missing feature threshold (default: 0.5)
- anomalyModelFeatureCount: feature count for anomaly model (default: 100)
- runOrder: run order data (default: NULL)
- n\_trees: number of trees (default: 100)
- max\_depth: max tree depth (default: "auto")
- numberOfCores: number of cores (default: 1)
- use\_log\_file: logging setting (default: TRUE)
- append: append setting (default: FALSE)
- verbose: verbose setting (default: TRUE)
- log\_file\_path: log file path (default: NULL)
- excludedFromQuantificationFilter: filter setting (default: NULL)

**Value**

NULL (throws error if validation fails)

---

as.data.frame.MSstatsValidated

*Convert output of converters to data.frame*

---

**Description**

Convert output of converters to data.frame

**Usage**

```
## S3 method for class 'MSstatsValidated'
as.data.frame(x, ...)
```

**Arguments**

x                    object of class MSstatsValidated  
 ...                  Additional arguments to be passed to or from other methods.

**Value**

data.frame

---

```
as.data.table.MSstatsValidated
  Convert output of converters to data.table
```

---

**Description**

Convert output of converters to data.table

**Usage**

```
## S3 method for class 'MSstatsValidated'
as.data.table(x, ...)
```

**Arguments**

x                    object of class MSstatsValidated  
 ...                  Additional arguments to be passed to or from other methods.

**Value**

data.tables

---

```
CheckDataHealth        Takes as input the output of the SpectronauttoMSstatsFormat function
and calculates various quality metrics to assess the health of the data.
Requires Anomaly Detection model to be fit.
```

---

**Description**

Takes as input the output of the SpectronauttoMSstatsFormat function and calculates various quality metrics to assess the health of the data. Requires Anomaly Detection model to be fit.

**Usage**

```
CheckDataHealth(input)
```

**Arguments**

input                  MSstats input which is the output of Spectronaut converter

**Value**

list of two data.tables

---

DIANNtoMSstatsFormat *Import Diann files*

---

## Description

Import Diann files

## Usage

```
DIANNtoMSstatsFormat(
  input,
  annotation = NULL,
  global_qvalue_cutoff = 0.01,
  qvalue_cutoff = 0.01,
  pg_qvalue_cutoff = 0.01,
  useUniquePeptide = TRUE,
  removeFewMeasurements = TRUE,
  removeOxidationMpeptides = TRUE,
  removeProtein_with1Feature = TRUE,
  MBR = TRUE,
  labeledAminoAcids = NULL,
  quantificationColumn = "FragmentQuantCorrected",
  calculateAnomalyScores = FALSE,
  anomalyModelFeatures = c(),
  anomalyModelFeatureTemporal = c(),
  removeMissingFeatures = 0.5,
  anomalyModelFeatureCount = 100,
  runOrder = NULL,
  n_trees = 100,
  max_depth = "auto",
  numberOfCores = 1,
  use_log_file = TRUE,
  append = FALSE,
  verbose = TRUE,
  log_file_path = NULL,
  ...
)
```

## Arguments

<code>input</code>	name of MSstats input report from Diann, which includes fragment-level data. Output fragment data with <code>-export-quant</code> flag in DIA-NN 2.0
<code>annotation</code>	name of 'annotation.txt' data which includes Condition, BioReplicate, Run.
<code>global_qvalue_cutoff</code>	The qvalue cutoff for the Q.Value column, i.e. the run-specific precursor q-value. Default is 0.01.
<code>qvalue_cutoff</code>	If MBR is false, the qvalue cutoff for the Global.Q.Value column, i.e. global precursor q-value. If MBR is true, the qvalue cutoff for the Lib.Q.Value column, i.e. the q-value for the library created after the first MBR pass. Default is 0.01.

pg_qvalue_cutoff	If MBR is false, the qvalue cutoff for the Global.PG.Q.Value column, i.e. the global q-value for the protein group. If MBR is true, the qvalue cutoff for the Lib.PG.Q.Value column, i.e. the protein group q-value for the library created after the first MBR pass. Default is 0.01.
useUniquePeptide	should unique peptides be removed
removeFewMeasurements	should proteins with few measurements be removed
removeOxidationMpeptides	should peptides with oxidation be removed
removeProtein_with1Feature	should proteins with a single feature be removed
MBR	True if analysis was done with match between runs
labeledAminoAcids	<p>Character vector of single-letter amino acid codes that carry the SILAC label in protein turnover experiments, e.g. c("K") or c("K", "R"). Supplying this vector opts in to protein-turnover mode; the exact amino acids determine behaviour only in the ModifiedSequence-parsing path described below.</p> <p><b>Channel-based path</b> (DIA-NN 2.x exports that include a Channel column): when labeledAminoAcids is non-NULL <i>and</i> the input contains a Channel column, Channel values are mapped directly to IsotopeLabelType ("H" → "H", "L" → "L", anything else → NA). The amino acid codes in labeledAminoAcids are <b>not</b> used to validate or filter ModifiedSequence in this path.</p> <p><b>ModifiedSequence-parsing path</b> (DIA-NN 1.x exports without a Channel column): when labeledAminoAcids is non-NULL and no Channel column is present, each ModifiedSequence is inspected for SILAC suffixes of the form (SILAC-&lt;AA&gt;-H) or (SILAC-&lt;AA&gt;-L), where &lt;AA&gt; is one of the supplied amino acid codes. Matching sequences are classified as "H" or "L"; sequences carrying neither suffix receive IsotopeLabelType = NA. The SILAC suffix is then stripped from PeptideSequence.</p> <p>When NULL (default), protein-turnover mode is disabled and all peptides receive IsotopeLabelType = "Light".</p>
quantificationColumn	Use 'FragmentQuantCorrected'(default) column for quantified intensities for DIANN 1.8.x. Use 'FragmentQuantRaw' for quantified intensities for DIANN 1.9.x. Use 'auto' for quantified intensities for DIANN 2.x where each fragment intensity is a separate column, e.g. Fr0Quantity.
calculateAnomalyScores	Default is FALSE. If TRUE, will run anomaly detection model and calculate anomaly scores for each feature. Used downstream to weigh measurements in differential analysis.
anomalyModelFeatures	character vector of quality metric column names to be used as features in the anomaly detection model. List must not be empty if calculateAnomalyScores=TRUE.
anomalyModelFeatureTemporal	character vector of temporal direction corresponding to columns passed to anomalyModelFeatures. Values must be one of: mean_decrease, mean_increase, dispersion_increase, or NULL (to perform no temporal feature engineering). Default is empty vector. If calculateAnomalyScores=TRUE, vector must have as many values as anomalyModelFeatures (even if all NULL).



```
# For DIANN 2.0, set quantificationColumn = 'auto'
input_file_path_2_0 = system.file("tinytest/raw_data/DIANN/diann_2.0.parquet",
                                package="MSstatsConvert")
annotation_file_path_2_0 = system.file("tinytest/raw_data/DIANN/annotation_diann_2.0.csv",
                                       package = "MSstatsConvert")
input_2_0 = arrow::read_parquet(input_file_path_2_0)
annot_2_0 = data.table::fread(annotation_file_path_2_0)
output_2_0 = DIANNtoMSstatsFormat(input_2_0, annotation = annot_2_0, MBR = FALSE,
                                  use_log_file = FALSE, quantificationColumn = 'auto')
head(output_2_0)
```

---

DIAUmpiretoMSstatsFormat

*Import DIA-Umpire files*

---

## Description

Import DIA-Umpire files

## Usage

```
DIAUmpiretoMSstatsFormat(
  raw.frag,
  raw.pep,
  raw.pro,
  annotation,
  useSelectedFrag = TRUE,
  useSelectedPep = TRUE,
  removeFewMeasurements = TRUE,
  removeProtein_with1Feature = FALSE,
  summaryforMultipleRows = max,
  use_log_file = TRUE,
  append = FALSE,
  verbose = TRUE,
  log_file_path = NULL,
  ...
)
```

## Arguments

raw.frag	name of FragSummary_date.xls data, which includes feature-level data.
raw.pep	name of PeptideSummary_date.xls data, which includes selected fragments information.
raw.pro	name of ProteinSummary_date.xls data, which includes selected peptides information.
annotation	name of annotation data which includes Condition, BioReplicate, Run information.
useSelectedFrag	TRUE will use the selected fragment for each peptide. 'Selected_fragments' column is required.

useSelectedPep	TRUE will use the selected peptide for each protein. 'Selected_peptides' column is required.
removeFewMeasurements	TRUE (default) will remove the features that have 1 or 2 measurements across runs.
removeProtein_with1Feature	TRUE will remove the proteins which have only 1 feature, which is the combination of peptide, precursor charge, fragment and charge. FALSE is default.
summaryforMultipleRows	max or sum - when there are multiple measurements for certain feature and certain run, use highest or sum of multiple intensities. Default is max for label-free converters and sum for TMT converters.
use_log_file	logical. If TRUE, information about data processing will be saved to a file.
append	logical. If TRUE, information about data processing will be added to an existing log file.
verbose	logical. If TRUE, information about data processing will be printed to the console.
log_file_path	character. Path to a file to which information about data processing will be saved. If not provided, such a file will be created automatically. If append = TRUE, has to be a valid path to a file.
...	additional parameters to data.table::fread.

**Value**

data.frame in the MSstats required format.

**Author(s)**

Meena Choi, Olga Vitek

**Examples**

```
diau_frag = system.file("tinytest/raw_data/DIAUmpire/dia_frag.csv",
  package = "MSstatsConvert")
diau_pept = system.file("tinytest/raw_data/DIAUmpire/dia_pept.csv",
  package = "MSstatsConvert")
diau_prot = system.file("tinytest/raw_data/DIAUmpire/dia_prot.csv",
  package = "MSstatsConvert")
annot = system.file("tinytest/raw_data/DIAUmpire/annot_diau.csv",
  package = "MSstatsConvert")
diau_frag = data.table::fread(diau_frag)
diau_pept = data.table::fread(diau_pept)
diau_prot = data.table::fread(diau_prot)
annot = data.table::fread(annot)
diau_frag = diau_frag[, lapply(.SD, function(x) if (is.integer(x)) as.numeric(x) else x)]
# In case numeric columns are not interpreted correctly

diau_imported = DIAUmpiretoMSstatsFormat(diau_frag, diau_pept, diau_prot,
  annot, use_log_file = FALSE)

head(diau_imported)
```

---

 FragPipeToMSstatsFormat

*Import FragPipe files*


---

## Description

Import FragPipe files

## Usage

```
FragPipeToMSstatsFormat(
  input,
  useUniquePeptide = TRUE,
  removeFewMeasurements = TRUE,
  removeProtein_with1Feature = FALSE,
  summaryforMultipleRows = max,
  use_log_file = TRUE,
  append = FALSE,
  verbose = TRUE,
  log_file_path = NULL,
  ...
)
```

## Arguments

input	name of FragPipe msstats.csv export. ProteinName, PeptideSequence, PrecursorCharge, FragmentIon, ProductCharge, IsotopeLabelType, Condition, BioReplicate, Run, Intensity are required.
useUniquePeptide	TRUE (default) removes peptides that are assigned for more than one proteins. We assume to use unique peptide for each protein.
removeFewMeasurements	TRUE (default) will remove the features that have 1 or 2 measurements across runs.
removeProtein_with1Feature	TRUE will remove the proteins which have only 1 feature, which is the combination of peptide, precursor charge, fragment and charge. FALSE is default.
summaryforMultipleRows	max or sum - when there are multiple measurements for certain feature and certain run, use highest or sum of multiple intensities. Default is max for label-free converters and sum for TMT converters.
use_log_file	logical. If TRUE, information about data processing will be saved to a file.
append	logical. If TRUE, information about data processing will be added to an existing log file.
verbose	logical. If TRUE, information about data processing will be printed to the console.
log_file_path	character. Path to a file to which information about data processing will be saved. If not provided, such a file will be created automatically. If append = TRUE, has to be a valid path to a file.
...	additional parameters to <code>data.table::fread</code> .

**Value**

data.frame in the MSstats required format.

**Author(s)**

Devon Kohler

**Examples**

```
fragpipe_raw = system.file("tinytest/raw_data/FragPipe/fragpipe_input.csv",
                           package = "MSstatsConvert")
fragpipe_raw = data.table::fread(fragpipe_raw)
fragpipe_imported = FragPipeToMSstatsFormat(fragpipe_raw, use_log_file = FALSE)
head(fragpipe_imported)
```

---

getDataType

*Get type of dataset from an MSstatsInputFiles object.*

---

**Description**

Get type of dataset from an MSstatsInputFiles object.

**Usage**

```
getDataType(msstats_object)

## S4 method for signature 'MSstatsInputFiles'
getDataType(msstats_object)
```

**Arguments**

msstats\_object object that inherits from MSstatsInputFiles class.

**Value**

character - label of a data type. Currently, "MSstats" or "MSstatsTMT"  
character "MSstats" or "MSstatsTMT".

**Examples**

```
evidence_path = system.file("tinytest/raw_data/MaxQuant/mq_ev.csv",
                             package = "MSstatsConvert")
pg_path = system.file("tinytest/raw_data/MaxQuant/mq_pg.csv",
                      package = "MSstatsConvert")
evidence = read.csv(evidence_path)
pg = read.csv(pg_path)
imported = MSstatsImport(list(evidence = evidence, protein_groups = pg),
                          "MSstats", "MaxQuant")

class(imported)
getDataType(imported) # "MSstats"
```

---

getInputFile	<i>Get one of files contained in an instance of MSstatsInputFiles class.</i>
--------------	--

---

**Description**

Get one of files contained in an instance of MSstatsInputFiles class.

**Usage**

```
getInputFile(msstats_object, file_type)

## S4 method for signature 'MSstatsInputFiles'
getInputFile(msstats_object, file_type = "input")

## S4 method for signature 'MSstatsPhilosopherFiles'
getInputFile(msstats_object, file_type = "input")
```

**Arguments**

msstats\_object object that inherits from MSstatsPhilosopherFiles class.  
file\_type character name of a type file. Usually equal to "input".

**Value**

data.table  
data.table  
data.table

**Examples**

```
evidence_path = system.file("tinytest/raw_data/MaxQuant/mq_ev.csv",
                             package = "MSstatsConvert")
pg_path = system.file("tinytest/raw_data/MaxQuant/mq_pg.csv",
                       package = "MSstatsConvert")
evidence = read.csv(evidence_path)
pg = read.csv(pg_path)
imported = MSstatsImport(list(evidence = evidence, protein_groups = pg),
                          "MSstats", "MaxQuant")
class(imported)
head(getInputFile(imported, "evidence"))
```

---

MaxQtoMSstatsFormat	<i>Import MaxQuant files</i>
---------------------	------------------------------

---

**Description**

Import MaxQuant files

**Usage**

```

MaxQtoMSstatsFormat(
  evidence,
  annotation,
  proteinGroups,
  proteinID = "Proteins",
  useUniquePeptide = TRUE,
  summaryforMultipleRows = max,
  removeFewMeasurements = TRUE,
  removeMpeptides = FALSE,
  removeOxidationMpeptides = FALSE,
  removeProtein_with1Peptide = FALSE,
  use_log_file = TRUE,
  append = FALSE,
  verbose = TRUE,
  log_file_path = NULL,
  ...
)

```

**Arguments**

evidence	name of 'evidence.txt' data, which includes feature-level data.
annotation	name of 'annotation.txt' data which includes Raw.file, Condition, BioReplicate, Run, IsotopeLabelType information.
proteinGroups	name of 'proteinGroups.txt' data. It needs to matching protein group ID. If proteinGroups=NULL, use 'Proteins' column in 'evidence.txt'.
proteinID	'Proteins'(default) or 'Leading.razor.protein' for Protein ID.
useUniquePeptide	TRUE (default) removes peptides that are assigned for more than one proteins. We assume to use unique peptide for each protein.
summaryforMultipleRows	max or sum - when there are multiple measurements for certain feature and certain run, use highest or sum of multiple intensities. Default is max for label-free converters and sum for TMT converters.
removeFewMeasurements	TRUE (default) will remove the features that have 1 or 2 measurements across runs.
removeMpeptides	TRUE will remove the peptides including 'M' sequence. FALSE is default.
removeOxidationMpeptides	TRUE will remove the peptides including 'oxidation (M)' in modification. FALSE is default.
removeProtein_with1Peptide	TRUE will remove the proteins which have only 1 peptide and charge. FALSE is default.
use_log_file	logical. If TRUE, information about data processing will be saved to a file.
append	logical. If TRUE, information about data processing will be added to an existing log file.
verbose	logical. If TRUE, information about data processing will be printed to the console.

`log_file_path` character. Path to a file to which information about data processing will be saved. If not provided, such a file will be created automatically. If `append = TRUE`, has to be a valid path to a file.

... additional parameters to `data.table::fread`.

**Value**

data.frame in the MSstats required format.

**Note**

Warning: MSstats does not support for metabolic labeling or iTRAQ experiments.

**Author(s)**

Meena Choi, Olga Vitek.

**Examples**

```
mq_ev = data.table::fread(system.file("tinytest/raw_data/MaxQuant/mq_ev.csv",
                                     package = "MSstatsConvert"))
mq_pg = data.table::fread(system.file("tinytest/raw_data/MaxQuant/mq_pg.csv",
                                     package = "MSstatsConvert"))
annot = data.table::fread(system.file("tinytest/raw_data/MaxQuant/annotation.csv",
                                     package = "MSstatsConvert"))
maxq_imported = MaxQtoMSstatsFormat(mq_ev, annot, mq_pg, use_log_file = FALSE)
head(maxq_imported)
```

---

MaxQtoMSstatsTMTFormat

*Generate MSstatsTMT required input format from MaxQuant output*

---

**Description**

Generate MSstatsTMT required input format from MaxQuant output

**Usage**

```
MaxQtoMSstatsTMTFormat(
  evidence,
  proteinGroups,
  annotation,
  which.proteinid = "Proteins",
  rmProt_Only_identified.by.site = FALSE,
  useUniquePeptide = TRUE,
  rmPSM_withfewMea_withinRun = TRUE,
  rmProtein_with1Feature = FALSE,
  summaryforMultipleRows = sum,
  use_log_file = TRUE,
  append = FALSE,
  verbose = TRUE,
```



```

proteinGroups = data.table::fread(system.file("tinytest/raw_data/MaxQuantTMT/mq_pg.csv",
                                             package = "MSstatsConvert"))
annotation.mq = data.table::fread(system.file("tinytest/raw_data/MaxQuantTMT/mq_annotation.csv",
                                             package = "MSstatsConvert"))
input.mq <- MaxQtoMSstatsTMTFormat(evidence, proteinGroups, annotation.mq)
head(input.mq)

```

---

MetamorpheusToMSstatsFormat

*Import Metamorpheus files*


---

## Description

Import Metamorpheus files

## Usage

```

MetamorpheusToMSstatsFormat(
  input,
  annotation = NULL,
  MBR = TRUE,
  qvalue_cutoff = 0.05,
  useUniquePeptide = TRUE,
  removeFewMeasurements = TRUE,
  removeProtein_with1Feature = FALSE,
  summaryforMultipleRows = max,
  use_log_file = TRUE,
  append = FALSE,
  verbose = TRUE,
  log_file_path = NULL,
  ...
)

```

## Arguments

input	name of Metamorpheus output file, which is tabular format. Use the AllQuantifiedPeaks.tsv file from the Metamorpheus output.
annotation	name of 'annotation.txt' data which includes Condition, BioReplicate.
MBR	If TRUE, the function will include peaks detected by MBR
qvalue_cutoff	The q-value cutoff for filtering peaks detected by MBR
useUniquePeptide	TRUE (default) removes peptides that are assigned for more than one proteins. We assume to use unique peptide for each protein.
removeFewMeasurements	TRUE (default) will remove the features that have 1 or 2 measurements across runs.
removeProtein_with1Feature	TRUE will remove the proteins which have only 1 feature, which is the combination of peptide, precursor charge, fragment and charge. FALSE is default.

summaryforMultipleRows	max or sum - when there are multiple measurements for certain feature and certain run, use highest or sum of multiple intensities. Default is max for label-free converters and sum for TMT converters.
use_log_file	logical. If TRUE, information about data processing will be saved to a file.
append	logical. If TRUE, information about data processing will be added to an existing log file.
verbose	logical. If TRUE, information about data processing will be printed to the console.
log_file_path	character. Path to a file to which information about data processing will be saved. If not provided, such a file will be created automatically. If append = TRUE, has to be a valid path to a file.
...	additional parameters to <code>data.table::fread</code> .

### Value

data.frame in the MSstats required format.

### Author(s)

Anthony Wu

### Examples

```
input = system.file("tinytest/raw_data/Metamorpheus/QuantifiedPeaks.tsv",
                    package = "MSstatsConvert")
input = data.table::fread(input)
annot = system.file("tinytest/raw_data/Metamorpheus/annotation.csv",
                    package = "MSstatsConvert")
annot = data.table::fread(annot)
metamorpheus_imported = MSstatsConvert:::MetamorpheusToMSstatsFormat(input, annotation = annot)
head(metamorpheus_imported)
```

---

MSstatsAnomalyScores *Run Anomaly Model*

---

### Description

Detects anomalous measurements in mass spectrometry data using an isolation forest algorithm. This function identifies unusual precursor measurements based on quality metrics and their temporal patterns. For features with insufficient quality metric data, it assigns anomaly scores based on the median score of similar features (same peptide and charge combination). The model supports parallel processing for improved performance on large datasets.

**Usage**

```
MSstatsAnomalyScores(
  input,
  quality_metrics,
  temporal_direction,
  missing_run_count,
  n_feat,
  run_order,
  n_trees,
  max_depth,
  cores
)
```

**Arguments**

input	data.table preprocessed by the MSstatsBalancedDesign function
quality_metrics	character vector of quality metrics to use in the model
temporal_direction	character vector of same length as quality_metrics indicating temporal feature to create.
missing_run_count	numeric, maximum allowed fraction of missing runs per feature.
n_feat	numeric, maximum number of features per protein to use in the model.
run_order	data.frame with two columns: Run and Order. Order should be numeric and indicate the order of runs.
n_trees	numeric, number of trees to use in the isolation forest model. Default is 100.
max_depth	numeric or "auto", maximum depth of each tree. Default is "auto" which sets depth to $\log_2(N)$ where N is the number of runs.
cores	numeric, number of cores to use for parallel processing. Default is 1.

**Value**

data.table

---

MSstatsBalancedDesign *Creates balanced design by removing overlapping fractions and filling incomplete rows*

---

**Description**

Creates balanced design by removing overlapping fractions and filling incomplete rows

**Usage**

```
MSstatsBalancedDesign(
  input,
  feature_columns,
  fill_incomplete = TRUE,
  handle_fractions = TRUE,
  fix_missing = NULL,
  remove_few = TRUE,
  anomaly_metrics = c()
)
```

**Arguments**

**input** data.table processed by the MSstatsPreprocess function

**feature\_columns** str, names of columns that define spectral features

**fill\_incomplete** if TRUE (default), ensures that rows with missing data for specific features are added as NA. For example, if the y10 ion of peptideA is measured in the "disease" samples but entirely missing for the "healthy" samples, rows with NA values will be created for the y10 ion of peptideA in the "healthy" group. This process increases the number of rows to account for all possible feature-sample combinations.

**handle\_fractions** if TRUE (default), overlapping fractions will be resolved

**fix\_missing** str, optional. Defaults to NULL, which means no action. If not NULL, must be one of the options: "zero\_to\_na" or "na\_to\_zero". If "zero\_to\_na", Intensity values equal exactly to 0 will be converted to NA. If "na\_to\_zero", missing values will be replaced by zeros.

**remove\_few** lgl, if TRUE, features with one or two measurements across runs will be removed.

**anomaly\_metrics** character vector of names of columns with quality metrics

**Value**

data.frame of class MSstatsValidated

**Examples**

```
unbalanced_data = system.file("tinytest/raw_data/unbalanced_data.csv",
                             package = "MSstatsConvert")
unbalanced_data = data.table::as.data.table(read.csv(unbalanced_data))
balanced = MSstatsBalancedDesign(unbalanced_data,
                                 c("PeptideSequence", "PrecursorCharge",
                                   "FragmentIon", "ProductCharge"))
dim(balanced) # Now balanced has additional rows (with Intensity = NA)
# for runs that were not included in the unbalanced_data table
```

---

MSstatsClean	<i>Clean files generated by a signal processing tools.</i>
--------------	--

---

### Description

Clean files generated by a signal processing tools.

Clean DIAUmpire files

Clean MaxQuant files

Clean OpenMS files

Clean OpenSWATH files

Clean Progenesis files

Clean ProteomeDiscoverer files

Clean Skyline files

Clean SpectroMine files

Clean Spectronaut files

Clean Philosopher files

Clean DIA-NN files

Clean Metamorpheus files

Clean Protein Prospector files

### Usage

```
MSstatsClean(msstats_object, ...)
```

```
## S4 method for signature 'MSstatsDIAUmpireFiles'  
MSstatsClean(msstats_object, use_frag, use_pept)
```

```
## S4 method for signature 'MSstatsMaxQuantFiles'  
MSstatsClean(  
  msstats_object,  
  protein_id_col,  
  remove_by_site = FALSE,  
  channel_columns = "Reporterintensitycorrected"  
)
```

```
## S4 method for signature 'MSstatsOpenMSFiles'  
MSstatsClean(msstats_object)
```

```
## S4 method for signature 'MSstatsOpenSWATHFiles'  
MSstatsClean(msstats_object)
```

```
## S4 method for signature 'MSstatsProgenesisFiles'  
MSstatsClean(msstats_object, runs, fix_colnames = TRUE)
```

```
## S4 method for signature 'MSstatsProteomeDiscovererFiles'  
MSstatsClean(  
  msstats_object,
```

```
msstats_object,  
quantification_column,  
protein_id_column,  
sequence_column,  
remove_shared,  
remove_protein_groups = TRUE,  
intensity_columns_regexp = "Abundance"  
)  
  
## S4 method for signature 'MSstatsSkylineFiles'  
MSstatsClean(msstats_object)  
  
## S4 method for signature 'MSstatsSpectroMineFiles'  
MSstatsClean(msstats_object)  
  
## S4 method for signature 'MSstatsSpectronautFiles'  
MSstatsClean(  
  msstats_object,  
  intensity,  
  calculateAnomalyScores,  
  anomalyModelFeatures,  
  peptideSequenceColumn = "EG.ModifiedSequence",  
  heavyLabels = NULL  
)  
  
## S4 method for signature 'MSstatsPhilosopherFiles'  
MSstatsClean(  
  msstats_object,  
  protein_id_col,  
  peptide_id_col,  
  channels,  
  remove_shared_peptides  
)  
  
## S4 method for signature 'MSstatsDIANNFiles'  
MSstatsClean(  
  msstats_object,  
  MBR = TRUE,  
  quantificationColumn = "FragmentQuantCorrected",  
  global_qvalue_cutoff = 0.01,  
  qvalue_cutoff = 0.01,  
  pg_qvalue_cutoff = 0.01,  
  calculateAnomalyScores = FALSE,  
  anomalyModelFeatures = c(),  
  labeledAminoAcids = NULL  
)  
  
## S4 method for signature 'MSstatsMetamorpheusFiles'  
MSstatsClean(msstats_object, MBR = TRUE, qvalue_cutoff = 0.05)  
  
## S4 method for signature 'MSstatsProteinProspectorFiles'  
MSstatsClean(msstats_object)
```

**Arguments**

- `msstats_object` object that inherits from `MSstatsInputFiles` class.
- `...` additional parameter to specific cleaning functions.
- `use_frag` TRUE will use the selected fragment for each peptide. 'Selected\_fragments' column is required.
- `use_pept` TRUE will use the selected fragment for each protein 'Selected\_peptides' column is required.
- `protein_id_col` character, name of a column with names of proteins.
- `remove_by_site` logical, if TRUE, proteins only identified by site will be removed.
- `channel_columns` character, regular expression that identifies channel columns in TMT data.
- `runs` chr, vector of Run labels.
- `fix_colnames` lgl, if TRUE, one of the rows will be used as colnames.
- `quantification_column` chr, name of a column used for quantification.
- `protein_id_column` chr, name of a column with protein IDs.
- `sequence_column` chr, name of a column with peptide sequences.
- `remove_shared` lgl, if TRUE, shared peptides will be removed.
- `remove_protein_groups` if TRUE, proteins with `numProteins > 1` will be removed.
- `intensity_columns_regexp` regular expressions that defines intensity columns. Defaults to "Abundance", which means that columns that contain the word "Abundance" will be treated as corresponding to intensities for different channels.
- `intensity` Intensity column to use. Accepts legacy enum values 'PeakArea' (default, uses `F.PeakArea`), 'NormalizedPeakArea' (uses `F.NormalizedPeakArea`). Can also be any raw Spectronaut column name passed as a string (e.g. "FG.MS1Quantity"); the column name is standardized internally. For protein turnover workflows the recommended default is "FG.MS1Quantity".
- `calculateAnomalyScores` Default is FALSE. If TRUE, will run anomaly detection model and calculate anomaly scores for each feature. Used downstream to weigh measurements in differential analysis.
- `anomalyModelFeatures` character vector of quality metric column names to be used as features in the anomaly detection model. List must not be empty if `calculateAnomalyScores=TRUE`.
- `peptideSequenceColumn` Name of the Spectronaut column that contains the peptide sequence. Defaults to "EG.ModifiedSequence". The value is standardized internally (dots and spaces removed) before column lookup.
- `heavyLabels` Character list identifying the heavy isotope labels as it appears inside square brackets in the peptide sequence column, e.g. `c("Lys6")` matches peptides containing [Lys6]. `c("Lys6", "Arg10")` matches peptides containing either [Lys6] or [Arg10]. Supports any novel label name reported by Spectronaut (e.g. "Leu6", "Phe10"). When provided, peptides are classified as heavy (`IsotopeLabelType`

	= "H"), light (IsotopeLabelType = "L"), or unlabeled (IsotopeLabelType = NA) based on its labeled sequence. When NULL (default) all peptides receive IsotopeLabelType = "L". Useful for protein turnover experiments.
peptide_id_col	character name of a column that identifies peptides
channels	character vector of channel labels
remove_shared_peptides	logical, if TRUE, shared peptides will be removed based on the IsUnique column from Philosopher output
MBR	True if analysis was done with match between runs
quantificationColumn	Use 'FragmentQuantCorrected'(default) column for quantified intensities for DIANN 1.8.x. Use 'FragmentQuantRaw' for quantified intensities for DIANN 1.9.x. Use 'auto' for quantified intensities for DIANN 2.x where each fragment intensity is a separate column, e.g. FrOQuantity.
global_qvalue_cutoff	The qvalue cutoff for the Q.Value column, i.e. the run-specific precursor q-value. Default is 0.01.
qvalue_cutoff	If MBR is false, the qvalue cutoff for the Global.Q.Value column, i.e. global precursor q-value. If MBR is true, the qvalue cutoff for the Lib.Q.Value column, i.e. the q-value for the library created after the first MBR pass. Default is 0.01.
pg_qvalue_cutoff	If MBR is false, the qvalue cutoff for the Global.PG.Q.Value column, i.e. the global q-value for the protein group. If MBR is true, the qvalue cutoff for the Lib.PG.Q.Value column, i.e. the protein group q-value for the library created after the first MBR pass. Default is 0.01.
labeledAminoAcids	<p>Character vector of single-letter amino acid codes that carry the SILAC label in protein turnover experiments, e.g. c("K") or c("K", "R"). Supplying this vector opts in to protein-turnover mode; the exact amino acids determine behaviour only in the ModifiedSequence-parsing path described below.</p> <p><b>Channel-based path</b> (DIA-NN 2.x exports that include a Channel column): when labeledAminoAcids is non-NULL <i>and</i> the input contains a Channel column, Channel values are mapped directly to IsotopeLabelType ("H" → "H", "L" → "L", anything else → NA). The amino acid codes in labeledAminoAcids are <b>not</b> used to validate or filter ModifiedSequence in this path.</p> <p><b>ModifiedSequence-parsing path</b> (DIA-NN 1.x exports without a Channel column): when labeledAminoAcids is non-NULL and no Channel column is present, each ModifiedSequence is inspected for SILAC suffixes of the form (SILAC-&lt;AA&gt;-H) or (SILAC-&lt;AA&gt;-L), where &lt;AA&gt; is one of the supplied amino acid codes. Matching sequences are classified as "H" or "L"; sequences carrying neither suffix receive IsotopeLabelType = NA. The SILAC suffix is then stripped from PeptideSequence.</p> <p>When NULL (default), protein-turnover mode is disabled and all peptides receive IsotopeLabelType = "Light".</p>

## Value

data.table  
data.table  
data.table

```
data.table
data.table
data.table
data.table
data.table
data.table
data.table
data.table
data.table
data.table
```

### Examples

```
evidence_path = system.file("tinytest/raw_data/MaxQuant/mq_ev.csv",
                             package = "MSstatsConvert")
pg_path = system.file("tinytest/raw_data/MaxQuant/mq_pg.csv",
                      package = "MSstatsConvert")
evidence = read.csv(evidence_path)
pg = read.csv(pg_path)
imported = MSstatsImport(list(evidence = evidence, protein_groups = pg),
                          "MSstats", "MaxQuant")
cleaned_data = MSstatsClean(imported, protein_id_col = "Proteins")
head(cleaned_data)
```

---

MSstatsConvert

*MSstatsConvert: An R Package to Convert Data from Mass Spectrometry Signal Processing Tools to MSstats Format*

---

### Description

MSstatsConvert helps convert data from different types of mass spectrometry experiments and signal processing tools to a format suitable for statistical analysis with the MSstats and MSstatsTMT packages.

### Main functions

[MSstatsLogsSettings](#) for logs management, [MSstatsImport](#) for importing files created by signal processing tools, [MSstatsClean](#) for re-formatting imported files into a consistent format, [MSstatsPreprocess](#) for preprocessing cleaned files, [MSstatsBalancedDesign](#) for handling fractions and creating balanced data.

### Author(s)

**Maintainer:** Anthony Wu <wu.anthon@northeastern.edu>

Authors:

- Mateusz Staniak <mtst@mstaniak.pl>
- Devon Kohler <kohler.d@northeastern.edu>

- Meena Choi <mnchoi67@gmail.com>
- Ting Huang <thuang0703@gmail.com>
- Olga Vitek <o.vitek@northeastern.edu>

---

MSstatsImport

*Import files from signal processing tools.*


---

### Description

Import files from signal processing tools.

### Usage

```
MSstatsImport(input_files, type, tool, tool_version = NULL, ...)
```

### Arguments

<code>input_files</code>	list of paths to input files or <code>data.frame</code> objects. Interpretation of this parameter depends on values of parameters <code>type</code> and <code>tool</code> .
<code>type</code>	chr, "MSstats" or "MSstatsTMT".
<code>tool</code>	chr, name of a signal processing tool that generated input files.
<code>tool_version</code>	not implemented yet. In the future, this parameter will allow handling different versions of each signal processing tools.
<code>...</code>	optional additional parameters to <code>data.table::fread</code> .

### Value

an object of class `MSstatsInputFiles`.

### Examples

```
evidence_path = system.file("tinytest/raw_data/MaxQuant/mq_ev.csv",
                             package = "MSstatsConvert")
pg_path = system.file("tinytest/raw_data/MaxQuant/mq_pg.csv",
                      package = "MSstatsConvert")
evidence = read.csv(evidence_path)
pg = read.csv(pg_path)
imported = MSstatsImport(list(evidence = evidence, protein_groups = pg),
                          "MSstats", "MaxQuant")
class(imported)
head(getInputFile(imported, "evidence"))
```

---

 MSstatsInputFiles-class

*Class to model files that describe a single MS dataset.*


---

### Description

Class to model files that describe a single MS dataset.

MSstatsDIAUmpireFiles: class for DIAUmpire files.

MSstatsMaxQuantFiles: class for MaxQuant files.

MSstatsOpenMSFiles: class for OpenMS files.

MSstatsOpenSWATHFiles: class for OpenSWATH files.

MSstatsProgenesisFiles: class for Progenesis files.

MSstatsProteomeDiscovererFiles: class for ProteomeDiscoverer files.

MSstatsSkylineFiles: class for Skyline files.

MSstatsSkylineFiles: class for SpectroMine files.

MSstatsSpectronautFiles: class for Spectronaut files.

MSstatsPhilosopherFiles: class for Philosopher files.

MSstatsDIANNFiles: class for DIA-NN files.

MSstatsFragPipeFiles: class for FragPipe files.

MSstatsMetamorpheusFiles: class for Metamorpheus files.

MSstatsProteinProspectorFiles: class for ProteinProspector files.

### Slots

`files` named list of files generated by a signal processing tools. In most cases, this will be a single file named `input`. In some cases, multiple files are used, for example MaxQuant outputs `evidence` and `proteinGroups` files.

`type` character: "MSstats" or "MSstatsTMT".

`tool` character: name of a signal processing tools that generated the output. Possible values are: DIAUmpire, MaxQuant, OpenMS, OpenSWATH, Progenesis, ProteomeDiscoverer, Skyline, SpectroMine, Spectronaut.

`version` description of a software version of the signal processing tool. Not implemented yet.

---

 MSstatsLogsSettings

*Set how MSstats will log information from data processing*


---

### Description

Set how MSstats will log information from data processing

**Usage**

```
MSstatsLogsSettings(
  use_log_file = TRUE,
  append = FALSE,
  verbose = TRUE,
  log_file_path = NULL,
  base = "MSstats_log_",
  pkg_name = "MSstats"
)
```

**Arguments**

<code>use_log_file</code>	logical. If TRUE, information about data processing will be saved to a file.
<code>append</code>	logical. If TRUE, information about data processing will be added to an existing log file.
<code>verbose</code>	logical. If TRUE, information about data processing will be printed to the console.
<code>log_file_path</code>	character. Path to a file to which information about data processing will be saved. If not provided, such a file will be created automatically. If <code>append = TRUE</code> , has to be a valid path to a file.
<code>base</code>	start of the file name.
<code>pkg_name</code>	currently "MSstats", "MSstatsPTM" or "MSstatsTMT". Each package can use its own separate log settings.

**Value**

TRUE invisibly in case of successful logging setup.

**Examples**

```
# No logging and no messages
MSstatsLogsSettings(FALSE, FALSE, FALSE)
# Log, but do not display messages
MSstatsLogsSettings(TRUE, FALSE, FALSE)
# Log to an existing file
file.create("new_log.log")
MSstatsLogsSettings(TRUE, TRUE, log_file_path = "new_log.log")
# Do not log, but display messages
MSstatsLogsSettings(FALSE)
```

---

MSstatsMakeAnnotation *Create annotation*

---

**Description**

Create annotation

**Usage**

```
MSstatsMakeAnnotation(input, annotation, ...)
```

**Arguments**

input            data.table preprocessed by the MSstatsClean function  
 annotation      data.table  
 ...              key-value pairs, where keys are names of columns of annotation

**Value**

data.table

**Examples**

```
evidence_path = system.file("tinytest/raw_data/MaxQuant/mq_ev.csv",
                             package = "MSstatsConvert")
pg_path = system.file("tinytest/raw_data/MaxQuant/mq_pg.csv",
                      package = "MSstatsConvert")
evidence = read.csv(evidence_path)
pg = read.csv(pg_path)
imported = MSstatsImport(list(evidence = evidence, protein_groups = pg),
                          "MSstats", "MaxQuant")
cleaned_data = MSstatsClean(imported, protein_id_col = "Proteins")
annot_path = system.file("tinytest/raw_data/MaxQuant/annotation.csv",
                         package = "MSstatsConvert")
mq_annot = MSstatsMakeAnnotation(cleaned_data, read.csv(annot_path),
                                 Run = "Rawfile")
head(mq_annot)
```

---

MSstatsPreprocess	<i>Preprocess outputs from MS signal processing tools for analysis with MSstats</i>
-------------------	---

---

**Description**

Preprocess outputs from MS signal processing tools for analysis with MSstats

**Usage**

```
MSstatsPreprocess(
  input,
  annotation,
  feature_columns,
  remove_shared_peptides = TRUE,
  remove_single_feature_proteins = TRUE,
  feature_cleaning = list(remove_features_with_few_measurements = TRUE,
                          summarize_multiple_psms = max),
  score_filtering = list(),
  exact_filtering = list(),
  pattern_filtering = list(),
  columns_to_fill = list(),
  aggregate_isotopic = FALSE,
  anomaly_metrics = c(),
  ...
)
```

**Arguments**

<code>input</code>	data.table processed by the MSstatsClean function.
<code>annotation</code>	annotation file generated by a signal processing tool.
<code>feature_columns</code>	character vector of names of columns that define spectral features.
<code>remove_shared_peptides</code>	logical, if TRUE shared peptides will be removed.
<code>remove_single_feature_proteins</code>	logical, if TRUE, proteins that only have one feature will be removed.
<code>feature_cleaning</code>	named list with maximum two (for MSstats converters) or three (for MSstatsTMT converter) elements. If <code>handle_few_measurements</code> is set to "remove", feature with less than three measurements will be removed (otherwise it should be equal to "keep"). <code>summarize_multiple_psms</code> is a function that will be used to aggregate multiple feature measurements in a run. It should return a scalar and accept an <code>na.rm</code> parameter. For MSstatsTMT converters, setting <code>remove_psms_with_any_missing</code> will remove features which have missing values in a run from that run.
<code>score_filtering</code>	a list of named lists that specify filtering options. Details are provided in the vignette.
<code>exact_filtering</code>	a list of named lists that specify filtering options. Details are provided in the vignette.
<code>pattern_filtering</code>	a list of named lists that specify filtering options. Details are provided in the vignette.
<code>columns_to_fill</code>	a named list of scalars. If provided, columns with names defined by the names of this list and values corresponding to its elements will be added to the output data.frame.
<code>aggregate_isotopic</code>	logical. If TRUE, isotopic peaks will be summed.
<code>anomaly_metrics</code>	character vector of names of columns with quality metrics. Default is missing and is not required if anomaly model not run.
<code>...</code>	additional parameters to <code>data.table::fread</code> .

**Value**

data.table

**Examples**

```
evidence_path = system.file("tinytest/raw_data/MaxQuant/mq_ev.csv",
                             package = "MSstatsConvert")
pg_path = system.file("tinytest/raw_data/MaxQuant/mq_pg.csv",
                      package = "MSstatsConvert")
evidence = read.csv(evidence_path)
pg = read.csv(pg_path)
```

```

imported = MSstatsImport(list(evidence = evidence, protein_groups = pg),
                          "MSstats", "MaxQuant")
cleaned_data = MSstatsClean(imported, protein_id_col = "Proteins")
annot_path = system.file("tinytest/raw_data/MaxQuant/annotation.csv",
                         package = "MSstatsConvert")
mq_annot = MSstatsMakeAnnotation(cleaned_data, read.csv(annot_path),
                                Run = "Rawfile")

# To filter M-peptides and oxidatin peptides
m_filter = list(col_name = "PeptideSequence", pattern = "M",
                filter = TRUE, drop_column = FALSE)
oxidation_filter = list(col_name = "Modifications", pattern = "Oxidation",
                       filter = TRUE, drop_column = TRUE)
msstats_format = MSstatsPreprocess(
  cleaned_data, mq_annot,
  feature_columns = c("PeptideSequence", "PrecursorCharge"),
  columns_to_fill = list(FragmentIon = NA, ProductCharge = NA),
  pattern_filtering = list(oxidation = oxidation_filter, m = m_filter)
)
# Output in the standard MSstats format
head(msstats_format)

```

---

MSstatsSaveSessionInfo

*Save session information*


---

## Description

Save session information

## Usage

```

MSstatsSaveSessionInfo(
  path = NULL,
  append = TRUE,
  base = "MSstats_session_info_"
)

```

## Arguments

path	optional path to output file. If not provided, "MSstats_session_info" and current timestamp will be used as a file name
append	if TRUE and file given by the path parameter already exists, session info will be appended to the file
base	beginning of a file name

## Value

TRUE invisibly after session info was saved

**Examples**

```
MSstatsSaveSessionInfo("session_info.txt")
MSstatsSaveSessionInfo("session_info.txt", base = "MSstatsTMT_session_info_")
```

---

OpenMStoMSstatsFormat *Import OpenMS files*

---

**Description**

Import OpenMS files

**Usage**

```
OpenMStoMSstatsFormat(
  input,
  annotation = NULL,
  useUniquePeptide = TRUE,
  removeFewMeasurements = TRUE,
  removeProtein_with1Feature = FALSE,
  summaryforMultipleRows = max,
  use_log_file = TRUE,
  append = FALSE,
  verbose = TRUE,
  log_file_path = NULL,
  ...
)
```

**Arguments**

<code>input</code>	name of MSstats input report from OpenMS, which includes feature(peptide ion)-level data.
<code>annotation</code>	name of 'annotation.txt' data which includes Condition, BioReplicate, Run. Run should be the same as filename.
<code>useUniquePeptide</code>	TRUE (default) removes peptides that are assigned for more than one proteins. We assume to use unique peptide for each protein.
<code>removeFewMeasurements</code>	TRUE (default) will remove the features that have 1 or 2 measurements across runs.
<code>removeProtein_with1Feature</code>	TRUE will remove the proteins which have only 1 feature, which is the combination of peptide, precursor charge, fragment and charge. FALSE is default.
<code>summaryforMultipleRows</code>	max or sum - when there are multiple measurements for certain feature and certain run, use highest or sum of multiple intensities. Default is max for label-free converters and sum for TMT converters.
<code>use_log_file</code>	logical. If TRUE, information about data processing will be saved to a file.
<code>append</code>	logical. If TRUE, information about data processing will be added to an existing log file.

verbose	logical. If TRUE, information about data processing will be printed to the console.
log_file_path	character. Path to a file to which information about data processing will be saved. If not provided, such a file will be created automatically. If append = TRUE, has to be a valid path to a file.
...	additional parameters to <code>data.table::fread</code> .

**Value**

data.frame in the MSstats required format.

**Author(s)**

Meena Choi, Olga Vitek.

**Examples**

```
openms_raw = data.table::fread(system.file("tinytest/raw_data/OpenMS/openms_input.csv",
                                           package = "MSstatsConvert"))
openms_imported = OpenMStoMSstatsFormat(openms_raw, use_log_file = FALSE)
head(openms_imported)
```

---

OpenMStoMSstatsTMTFormat

*Generate MSstatsTMT required input format for OpenMS output*

---

**Description**

Generate MSstatsTMT required input format for OpenMS output

**Usage**

```
OpenMStoMSstatsTMTFormat(
  input,
  useUniquePeptide = TRUE,
  rmPSM_withfewMea_withinRun = TRUE,
  rmProtein_with1Feature = FALSE,
  summaryforMultiplePSMs = sum,
  use_log_file = TRUE,
  append = FALSE,
  verbose = TRUE,
  log_file_path = NULL,
  ...
)
```

**Arguments**

<code>input</code>	MSstatsTMT report from OpenMS
<code>useUniquePeptide</code>	TRUE (default) removes peptides that are assigned for more than one proteins. We assume to use unique peptide for each protein.
<code>rmPSM_withfewMea_withinRun</code>	TRUE (default) will remove the features that have 1 or 2 measurements within each Run.
<code>rmProtein_with1Feature</code>	TRUE will remove the proteins which have only 1 peptide and charge. Default is FALSE.
<code>summaryforMultiplePSMs</code>	sum(default) or max - when there are multiple measurements for certain feature in certain run, select the feature with the largest summation or maximal value.
<code>use_log_file</code>	logical. If TRUE, information about data processing will be saved to a file.
<code>append</code>	logical. If TRUE, information about data processing will be added to an existing log file.
<code>verbose</code>	logical. If TRUE, information about data processing will be printed to the console.
<code>log_file_path</code>	character. Path to a file to which information about data processing will be saved. If not provided, such a file will be created automatically. If <code>append = TRUE</code> , has to be a valid path to a file.
<code>...</code>	additional parameters to <code>data.table::fread</code> .

**Value**

data.frame of class MSstatsTMT.

**Examples**

```
raw.om = data.table::fread(system.file("tinytest/raw_data/OpenMSTMT/openmstmt_input.csv",
                                     package = "MSstatsConvert"))
input.om <- OpenMStoMSstatsTMTFormat(raw.om)
head(input.om)
```

---

OpenSWATHtoMSstatsFormat

*Import OpenSWATH files*

---

**Description**

Import OpenSWATH files

**Usage**

```

OpenSWATHtoMSstatsFormat(
  input,
  annotation,
  filter_with_mscore = TRUE,
  mscore_cutoff = 0.01,
  useUniquePeptide = TRUE,
  removeFewMeasurements = TRUE,
  removeProtein_with1Feature = FALSE,
  summaryforMultipleRows = max,
  use_log_file = TRUE,
  append = FALSE,
  verbose = TRUE,
  log_file_path = NULL,
  ...
)

```

**Arguments**

input	name of MSstats input report from OpenSWATH, which includes feature-level data.
annotation	name of 'annotation.txt' data which includes Condition, BioReplicate, Run. Run should be the same as filename.
filter_with_mscore	TRUE(default) will filter out the features that have greater than mscore_cutoff in m_score column. Those features will be removed.
mscore_cutoff	Cutoff for m_score. Default is 0.01.
useUniquePeptide	TRUE (default) removes peptides that are assigned for more than one proteins. We assume to use unique peptide for each protein.
removeFewMeasurements	TRUE (default) will remove the features that have 1 or 2 measurements across runs.
removeProtein_with1Feature	TRUE will remove the proteins which have only 1 feature, which is the combination of peptide, precursor charge, fragment and charge. FALSE is default.
summaryforMultipleRows	max or sum - when there are multiple measurements for certain feature and certain run, use highest or sum of multiple intensities. Default is max for label-free converters and sum for TMT converters.
use_log_file	logical. If TRUE, information about data processing will be saved to a file.
append	logical. If TRUE, information about data processing will be added to an existing log file.
verbose	logical. If TRUE, information about data processing will be printed to the console.
log_file_path	character. Path to a file to which information about data processing will be saved. If not provided, such a file will be created automatically. If append = TRUE, has to be a valid path to a file.
...	additional parameters to data.table::fread.

**Value**

data.frame in the MSstats required format.

**Author(s)**

Meena Choi, Olga Vitek.

**Examples**

```
os_raw = system.file("tinytest/raw_data/OpenSWATH/openswath_input.csv",
                    package = "MSstatsConvert")
annot = system.file("tinytest/raw_data/OpenSWATH/annot_os.csv",
                  package = "MSstatsConvert")
os_raw = data.table::fread(os_raw)
annot = data.table::fread(annot)

os_imported = OpenSWATHtoMSstatsFormat(os_raw, annot, use_log_file = FALSE)
head(os_imported)
```

---

PDtoMSstatsFormat

*Import Proteome Discoverer files*

---

**Description**

Import Proteome Discoverer files

**Usage**

```
PDtoMSstatsFormat(
  input,
  annotation,
  useNumProteinsColumn = FALSE,
  useUniquePeptide = TRUE,
  summaryforMultipleRows = max,
  removeFewMeasurements = TRUE,
  removeOxidationMpeptides = FALSE,
  removeProtein_with1Peptide = FALSE,
  which.quantification = "Precursor.Area",
  which.proteinid = "Protein.Group.Accessions",
  which.sequence = "Sequence",
  use_log_file = TRUE,
  append = FALSE,
  verbose = TRUE,
  log_file_path = NULL,
  ...
)
```

**Arguments**

input	PD report or a path to it.
annotation	name of 'annotation.txt' or 'annotation.csv' data which includes Condition, BioReplicate, Run information. 'Run' will be matched with 'Spectrum.File'.
useNumProteinsColumn	TRUE removes peptides which have more than 1 in # Proteins column of PD output.
useUniquePeptide	TRUE (default) removes peptides that are assigned for more than one proteins. We assume to use unique peptide for each protein.
summaryforMultipleRows	max or sum - when there are multiple measurements for certain feature and certain run, use highest or sum of multiple intensities. Default is max for label-free converters and sum for TMT converters.
removeFewMeasurements	TRUE (default) will remove the features that have 1 or 2 measurements across runs.
removeOxidationMpeptides	TRUE will remove the peptides including 'oxidation (M)' in modification. FALSE is default.
removeProtein_with1Peptide	TRUE will remove the proteins which have only 1 peptide and charge. FALSE is default.
which.quantification	Use 'Precursor.Area'(default) column for quantified intensities. 'Intensity' or 'Area' can be used instead.
which.proteinid	Use 'Protein.Accessions'(default) column for protein name. 'Master.Protein.Accessions' can be used instead.
which.sequence	Use 'Sequence'(default) column for peptide sequence. 'Annotated.Sequence' can be used instead.
use_log_file	logical. If TRUE, information about data processing will be saved to a file.
append	logical. If TRUE, information about data processing will be added to an existing log file.
verbose	logical. If TRUE, information about data processing will be printed to the console.
log_file_path	character. Path to a file to which information about data processing will be saved. If not provided, such a file will be created automatically. If append = TRUE, has to be a valid path to a file.
...	additional parameters to <code>data.table::fread</code> .

**Value**

data.frame in the MSstats required format.

**Author(s)**

Meena Choi, Olga Vitek

**Examples**

```
pd_raw = system.file("tinytest/raw_data/PD/pd_input.csv",
                    package = "MSstatsConvert")
annot = system.file("tinytest/raw_data/PD/annot_pd.csv",
                  package = "MSstatsConvert")
pd_raw = data.table::fread(pd_raw)
annot = data.table::fread(annot)

pd_imported = PDtoMSstatsFormat(pd_raw, annot, use_log_file = FALSE)
head(pd_imported)
```

---

PDtoMSstatsTMTFormat *Convert Proteome Discoverer output to MSstatsTMT format.*

---

**Description**

Convert Proteome Discoverer output to MSstatsTMT format.

**Usage**

```
PDtoMSstatsTMTFormat(
  input,
  annotation,
  which.proteinid = "Protein.Accessions",
  useNumProteinsColumn = TRUE,
  useUniquePeptide = TRUE,
  rmPSM_withfewMea_withinRun = TRUE,
  rmProtein_with1Feature = FALSE,
  summaryforMultipleRows = sum,
  use_log_file = TRUE,
  append = FALSE,
  verbose = TRUE,
  log_file_path = NULL,
  ...
)
```

**Arguments**

input	PD report or a path to it.
annotation	annotation with Run, Fraction, TechRepMixture, Mixture, Channel, BioReplicate, Condition columns or a path to file. Refer to the example 'annotation' for the meaning of each column.
which.proteinid	Use 'Protein.Accessions' (default) column for protein name. 'Master.Protein.Accessions' can be used instead to get the protein name with single protein.
useNumProteinsColumn	logical, TRUE (default) removes shared peptides by information of # Proteins column in PSM sheet.

useUniquePeptide	TRUE (default) removes peptides that are assigned for more than one proteins. We assume to use unique peptide for each protein.
rmPSM_withfewMea_withinRun	TRUE (default) will remove the features that have 1 or 2 measurements within each Run.
rmProtein_with1Feature	TRUE will remove the proteins which have only 1 peptide and charge. Default is FALSE.
summaryforMultipleRows	max or sum - when there are multiple measurements for certain feature and certain run, use highest or sum of multiple intensities. Default is max for label-free converters and sum for TMT converters.
use_log_file	logical. If TRUE, information about data processing will be saved to a file.
append	logical. If TRUE, information about data processing will be added to an existing log file.
verbose	logical. If TRUE, information about data processing will be printed to the console.
log_file_path	character. Path to a file to which information about data processing will be saved. If not provided, such a file will be created automatically. If append = TRUE, has to be a valid path to a file.
...	additional parameters to <code>data.table::fread</code> .

**Value**

data.frame of class MSstatsTMT

**Examples**

```
raw.pd = data.table::fread(system.file("tinytest/raw_data/PDTMT/pdmt_input.csv",
                                     package = "MSstatsConvert"))
annotation.pd = data.table::fread(system.file("tinytest/raw_data/PDTMT/pd_annotation.csv",
                                             package = "MSstatsConvert"))
head(raw.pd)
head(annotation.pd)
input.pd <- PDtoMSstatsTMTFormat(raw.pd, annotation.pd)
head(input.pd)
```

---

PhilosophertoMSstatsTMTFormat

*Convert Philosopher (Fragpipe) output to MSstatsTMT format.*

---

**Description**

Convert Philosopher (Fragpipe) output to MSstatsTMT format.

**Usage**

```
PhilosophertoMSstatsTMTFormat(
  input,
  annotation,
  protein_id_col = "Protein",
  peptide_id_col = "Peptide.Sequence",
  Purity_cutoff = 0.6,
  PeptideProphet_prob_cutoff = 0.7,
  useUniquePeptide = TRUE,
  rmPSM_withfewMea_withinRun = TRUE,
  rmPeptide_OxidationM = TRUE,
  rmProtein_with1Feature = FALSE,
  summaryforMultipleRows = sum,
  use_log_file = TRUE,
  append = FALSE,
  verbose = TRUE,
  log_file_path = NULL,
  ...
)
```

**Arguments**

<code>input</code>	data.frame of <code>msstats.csv</code> file produced by Philosopher
<code>annotation</code>	annotation with Run, Fraction, TechRepMixture, Mixture, Channel, BioReplicate, Condition columns or a path to file. Refer to the example 'annotation' for the meaning of each column. Channel column should be consistent with the channel columns (Ignore the prefix "Channel ") in <code>msstats.csv</code> file. Run column should be consistent with the Spectrum.File columns in <code>msstats.csv</code> file.
<code>protein_id_col</code>	Use 'Protein'(default) column for protein name. 'Master.Protein.Accessions' can be used instead to get the protein ID with single protein.
<code>peptide_id_col</code>	Use 'Peptide.Sequence'(default) column for peptide sequence. 'Modified.Peptide.Sequence' can be used instead to get the modified peptide sequence.
<code>Purity_cutoff</code>	Cutoff for purity. Default is 0.6
<code>PeptideProphet_prob_cutoff</code>	Cutoff for the peptide identification probability. Default is 0.7. The probability is confidence score determined by PeptideProphet and higher values indicate greater confidence.
<code>useUniquePeptide</code>	TRUE (default) removes peptides that are assigned for more than one proteins. We assume to use unique peptide for each protein.
<code>rmPSM_withfewMea_withinRun</code>	TRUE (default) will remove the features that have 1 or 2 measurements within each Run.
<code>rmPeptide_OxidationM</code>	TRUE (default) will remove the peptides including oxidation (M) sequence.
<code>rmProtein_with1Feature</code>	TRUE will remove the proteins which have only 1 peptide and charge. Default is FALSE.

summaryforMultipleRows	max or sum - when there are multiple measurements for certain feature and certain run, use highest or sum of multiple intensities. Default is max for label-free converters and sum for TMT converters.
use_log_file	logical. If TRUE, information about data processing will be saved to a file.
append	logical. If TRUE, information about data processing will be added to an existing log file.
verbose	logical. If TRUE, information about data processing will be printed to the console.
log_file_path	character. Path to a file to which information about data processing will be saved. If not provided, such a file will be created automatically. If append = TRUE, has to be a valid path to a file.
...	additional parameters to <code>data.table::fread</code> .

**Value**

data.frame of class MSstatsTMT

**Examples**

```
input_file_path = system.file("tinytest/raw_data/Philosopher/msstats.csv",
                             package = "MSstatsConvert")
annotation_file_path = system.file("tinytest/raw_data/Philosopher/MSstatsTMT_annotation.csv",
                                   package = "MSstatsConvert")
input = data.table::fread(input_file_path)
annotation = data.table::fread(annotation_file_path)
msstats_format = PhilosphertoMSstatsTMTFormat(input, annotation)
head(msstats_format)
```

---

ProgenisistoMSstatsFormat

*Import Progenis files*

---

**Description**

Import Progenis files

**Usage**

```
ProgenisistoMSstatsFormat(
  input,
  annotation,
  useUniquePeptide = TRUE,
  summaryforMultipleRows = max,
  removeFewMeasurements = TRUE,
  removeOxidationMpeptides = FALSE,
  removeProtein_with1Peptide = FALSE,
  use_log_file = TRUE,
  append = FALSE,
  verbose = TRUE,
```

```

    log_file_path = NULL,
    ...
)

```

### Arguments

input	name of Progenesis output, which is wide-format. 'Accession', 'Sequence', 'Modification', 'Charge' and one column for each run are required.
annotation	name of 'annotation.txt' or 'annotation.csv' data which includes Condition, BioReplicate, Run information. It will be matched with the column name of input for MS runs.
useUniquePeptide	TRUE (default) removes peptides that are assigned for more than one proteins. We assume to use unique peptide for each protein.
summaryforMultipleRows	max or sum - when there are multiple measurements for certain feature and certain run, use highest or sum of multiple intensities. Default is max for label-free converters and sum for TMT converters.
removeFewMeasurements	TRUE (default) will remove the features that have 1 or 2 measurements across runs.
removeOxidationMpeptides	TRUE will remove the peptides including 'oxidation (M)' in modification. FALSE is default.
removeProtein_with1Peptide	TRUE will remove the proteins which have only 1 peptide and charge. FALSE is default.
use_log_file	logical. If TRUE, information about data processing will be saved to a file.
append	logical. If TRUE, information about data processing will be added to an existing log file.
verbose	logical. If TRUE, information about data processing will be printed to the console.
log_file_path	character. Path to a file to which information about data processing will be saved. If not provided, such a file will be created automatically. If append = TRUE, has to be a valid path to a file.
...	additional parameters to <code>data.table::fread</code> .

### Value

data.frame in the MSstats required format.

### Author(s)

Meena Choi, Olga Vitek, Ulrich Omasits

### Examples

```

progenesis_raw = system.file("tinytest/raw_data/Progenesis/progenesis_input.csv",
                             package = "MSstatsConvert")
annot = system.file("tinytest/raw_data/Progenesis/progenesis_annot.csv",
                   package = "MSstatsConvert")
progenesis_raw = data.table::fread(progenesis_raw)

```

```

annot = data.table::fread(annot)

progenesis_imported = ProgenesisToMSstatsFormat(progenesis_raw, annot,
                                                use_log_file = FALSE)

head(progenesis_imported)

```

---

ProteinProspectortoMSstatsTMTFormat

*Generate MSstatsTMT required input format from Protein Prospector output*

---

## Description

Generate MSstatsTMT required input format from Protein Prospector output

## Usage

```

ProteinProspectortoMSstatsTMTFormat(
  input,
  annotation,
  useUniquePeptide = TRUE,
  removeFewMeasurements = TRUE,
  removeProtein_with1Feature = FALSE,
  summaryforMultipleRows = sum,
  use_log_file = TRUE,
  append = FALSE,
  verbose = TRUE,
  log_file_path = NULL
)

```

## Arguments

input	Input txt peptide report file from Protein Prospector with "Keep Replicates", "Mods in Peptide", and "Protein Mods" options selected.
annotation	data frame which contains column Run, Fraction, TechRepMixture, Mixture, Channel, BioReplicate, Condition.
useUniquePeptide	TRUE (default) removes peptides that are assigned for more than one proteins. We assume to use unique peptide for each protein.
removeFewMeasurements	TRUE (default) will remove the features that have 1 or 2 measurements across runs.
removeProtein_with1Feature	TRUE will remove the proteins which have only 1 feature, which is the combination of peptide, precursor charge, fragment and charge. FALSE is default.
summaryforMultipleRows	max or sum - when there are multiple measurements for certain feature and certain run, use highest or sum of multiple intensities. Default is max for label-free converters and sum for TMT converters.

use_log_file	logical. If TRUE, information about data processing will be saved to a file.
append	logical. If TRUE, information about data processing will be added to an existing log file.
verbose	logical. If TRUE, information about data processing will be printed to the console.
log_file_path	character. Path to a file to which information about data processing will be saved. If not provided, such a file will be created automatically. If append = TRUE, has to be a valid path to a file.

**Value**

data.frame of class "MSstatsTMT"

**Examples**

```
input = system.file("tinytest/raw_data/ProteinProspector/Prospector_TotalTMT.txt",
  package = "MSstatsConvert")
input = data.table::fread(input)
annot = system.file("tinytest/raw_data/ProteinProspector/Annotation.csv",
  package = "MSstatsConvert")
annot = data.table::fread(annot)
output <- ProteinProspectortoMSstatsTMTFormat(input, annot)
head(output)
```

---

SkylinetoMSstatsFormat

*Import Skyline files*

---

**Description**

Import Skyline files

**Usage**

```
SkylinetoMSstatsFormat(
  input,
  annotation = NULL,
  removeiRT = TRUE,
  filter_with_Qvalue = TRUE,
  qvalue_cutoff = 0.01,
  useUniquePeptide = TRUE,
  removeFewMeasurements = TRUE,
  removeOxidationMpeptides = FALSE,
  removeProtein_with1Feature = FALSE,
  use_log_file = TRUE,
  append = FALSE,
  verbose = TRUE,
  log_file_path = NULL,
  ...
)
```

**Arguments**

input	name of MSstats input report from Skyline, which includes feature-level data.
annotation	name of 'annotation.txt' data which includes Condition, BioReplicate, Run. If annotation is already complete in Skyline, use annotation=NULL (default). It will use the annotation information from input.
removeiRT	TRUE (default) will remove the proteins or peptides which are labeled 'iRT' in 'StandardType' column. FALSE will keep them.
filter_with_Qvalue	TRUE(default) will filter out the intensities that have greater than qvalue_cutoff in DetectionQValue column. Those intensities will be replaced with zero and will be considered as censored missing values for imputation purpose.
qvalue_cutoff	Cutoff for DetectionQValue. default is 0.01.
useUniquePeptide	TRUE (default) removes peptides that are assigned for more than one proteins. We assume to use unique peptide for each protein.
removeFewMeasurements	TRUE (default) will remove the features that have 1 or 2 measurements across runs.
removeOxidationMpeptides	TRUE will remove the peptides including 'oxidation (M)' in modification. FALSE is default.
removeProtein_with1Feature	TRUE will remove the proteins which have only 1 feature, which is the combination of peptide, precursor charge, fragment and charge. FALSE is default.
use_log_file	logical. If TRUE, information about data processing will be saved to a file.
append	logical. If TRUE, information about data processing will be added to an existing log file.
verbose	logical. If TRUE, information about data processing will be printed to the console.
log_file_path	character. Path to a file to which information about data processing will be saved. If not provided, such a file will be created automatically. If append = TRUE, has to be a valid path to a file.
...	additional parameters to <code>data.table::fread</code> .

**Value**

data.frame in the MSstats required format.

**Author(s)**

Meena Choi, Olga Vitek

**Examples**

```
skyline_raw = system.file("tinytest/raw_data/Skyline/skyline_input.csv",
                          package = "MSstatsConvert")
skyline_raw = data.table::fread(skyline_raw)
skyline_imported = SkylinetoMSstatsFormat(skyline_raw)
head(skyline_imported)
```

---

SpectroMine to MSstats TMTFormat

*Import data from SpectroMine*


---

## Description

Import data from SpectroMine

## Usage

```
SpectroMine to MSstats TMTFormat(
  input,
  annotation,
  filter_with_Qvalue = TRUE,
  qvalue_cutoff = 0.01,
  useUniquePeptide = TRUE,
  rmPSM_withfewMea_withinRun = TRUE,
  rmProtein_with1Feature = FALSE,
  summaryforMultipleRows = sum,
  use_log_file = TRUE,
  append = FALSE,
  verbose = TRUE,
  log_file_path = NULL,
  ...
)
```

## Arguments

input	data name of SpectroMine PSM output. Read PSM sheet.
annotation	data frame which contains column Run, Fraction, TechRepMixture, Mixture, Channel, BioReplicate, Condition. Refer to the example 'annotation.mine' for the meaning of each column.
filter_with_Qvalue	TRUE(default) will filter out the intensities that have greater than qvalue_cutoff in EG.Qvalue column. Those intensities will be replaced with NA and will be considered as censored missing values for imputation purpose.
qvalue_cutoff	Cutoff for EG.Qvalue. default is 0.01.
useUniquePeptide	TRUE (default) removes peptides that are assigned for more than one proteins. We assume to use unique peptide for each protein.
rmPSM_withfewMea_withinRun	TRUE (default) will remove the features that have 1 or 2 measurements within each Run.
rmProtein_with1Feature	TRUE will remove the proteins which have only 1 peptide and charge. Default is FALSE.
summaryforMultipleRows	max or sum - when there are multiple measurements for certain feature and certain run, use highest or sum of multiple intensities. Default is max for label-free converters and sum for TMT converters.

<code>use_log_file</code>	logical. If TRUE, information about data processing will be saved to a file.
<code>append</code>	logical. If TRUE, information about data processing will be added to an existing log file.
<code>verbose</code>	logical. If TRUE, information about data processing will be printed to the console.
<code>log_file_path</code>	character. Path to a file to which information about data processing will be saved. If not provided, such a file will be created automatically. If <code>append = TRUE</code> , has to be a valid path to a file.
<code>...</code>	additional parameters to <code>data.table::fread</code> .

**Value**

data.frame of class MSstatsTMT

**Examples**

```
raw.mine = data.table::fread(system.file("tinytest/raw_data/SpectroMine/spectromine_input.csv",
                                         package = "MSstatsConvert"))
annotation.mine = data.table::fread(system.file("tinytest/raw_data/SpectroMine/spectromine_annotation.csv",
                                                package = "MSstatsConvert"))

head(raw.mine)
head(annotation.mine)
input.mine <- SpectroMinetoMSstatsTMTFormat(raw.mine, annotation.mine)
head(input.mine)
```

---

SpectronauttoMSstatsFormat

*Import Spectronaut files*

---

**Description**

Import Spectronaut files

**Usage**

```
SpectronauttoMSstatsFormat(
  input,
  annotation = NULL,
  intensity = "PeakArea",
  peptideSequenceColumn = "EG.ModifiedSequence",
  heavyLabels = NULL,
  excludedFromQuantificationFilter = TRUE,
  filter_with_Qvalue = FALSE,
  qvalue_cutoff = 0.01,
  useUniquePeptide = TRUE,
  removeFewMeasurements = TRUE,
  removeProtein_with1Feature = FALSE,
  summaryforMultipleRows = max,
  calculateAnomalyScores = FALSE,
  anomalyModelFeatures = c(),
```

```

anomalyModelFeatureTemporal = c(),
removeMissingFeatures = 0.5,
anomalyModelFeatureCount = 100,
runOrder = NULL,
n_trees = 100,
max_depth = "auto",
numberOfCores = 1,
use_log_file = TRUE,
append = FALSE,
verbose = TRUE,
log_file_path = NULL,
...
)

```

### Arguments

input	name of Spectronaut output, which is long-format. ProteinName, PeptideSequence, PrecursorCharge, FragmentIon, ProductCharge, IsotopeLabelType, Condition, BioReplicate, Run, Intensity, F.ExcludedFromQuantification are required. Rows with F.ExcludedFromQuantification=True will be removed.
annotation	name of 'annotation.txt' data which includes Condition, BioReplicate, Run. If annotation is already complete in Spectronaut, use annotation=NULL (default). It will use the annotation information from input.
intensity	Intensity column to use. Accepts legacy enum values 'PeakArea' (default, uses F.PeakArea), 'NormalizedPeakArea' (uses F.NormalizedPeakArea). Can also be any raw Spectronaut column name passed as a string (e.g. "FG.MS1Quantity"); the column name is standardized internally. For protein turnover workflows the recommended default is "FG.MS1Quantity".
peptideSequenceColumn	Name of the Spectronaut column that contains the peptide sequence. Defaults to "EG.ModifiedSequence". The value is standardized internally (dots and spaces removed) before column lookup.
heavyLabels	Character list identifying the heavy isotope labels as it appears inside square brackets in the peptide sequence column, e.g. c("Lys6") matches peptides containing [Lys6]. c("Lys6", "Arg10") matches peptides containing either [Lys6] or [Arg10]. Supports any novel label name reported by Spectronaut (e.g. "Leu6", "Phe10"). When provided, peptides are classified as heavy (IsotopeLabelType = "H"), light (IsotopeLabelType = "L"), or unlabeled (IsotopeLabelType = NA) based on its labeled sequence. When NULL (default) all peptides receive IsotopeLabelType = "L". Useful for protein turnover experiments.
excludedFromQuantificationFilter	Remove rows with F.ExcludedFromQuantification=TRUE Default is TRUE.
filter_with_Qvalue	FALSE(default) will not perform any filtering. TRUE will filter out the intensities that have greater than qvalue_cutoff in EG.Qvalue column. Those intensities will be replaced with zero and will be considered as censored missing values for imputation purpose.
qvalue_cutoff	Cutoff for EG.Qvalue. default is 0.01.
useUniquePeptide	TRUE (default) removes peptides that are assigned for more than one proteins. We assume to use unique peptide for each protein.

removeFewMeasurements	TRUE (default) will remove the features that have 1 or 2 measurements across runs.
removeProtein_with1Feature	TRUE will remove the proteins which have only 1 feature, which is the combination of peptide, precursor charge, fragment and charge. FALSE is default.
summaryforMultipleRows	max or sum - when there are multiple measurements for certain feature and certain run, use highest or sum of multiple intensities. Default is max for label-free converters and sum for TMT converters.
calculateAnomalyScores	Default is FALSE. If TRUE, will run anomaly detection model and calculate anomaly scores for each feature. Used downstream to weigh measurements in differential analysis.
anomalyModelFeatures	character vector of quality metric column names to be used as features in the anomaly detection model. List must not be empty if calculateAnomalyScores=TRUE.
anomalyModelFeatureTemporal	character vector of temporal direction corresponding to columns passed to anomalyModelFeatures. Values must be one of: mean_decrease, mean_increase, dispersion_increase, or NULL (to perform no temporal feature engineering). Default is empty vector. If calculateAnomalyScores=TRUE, vector must have as many values as anomalyModelFeatures (even if all NULL).
removeMissingFeatures	Remove features with missing values in more than this fraction of runs. Default is 0.5. Only used if calculateAnomalyScores=TRUE.
anomalyModelFeatureCount	Feature selection for anomaly model. Anomaly detection works on the precursor-level and can be much slower if all features used. We will by default filter to the top-100 highest intensity features. This can be adjusted as necessary. To turn feature-selection off, set this value to a high number (e.g. 10000). Only used if calculateAnomalyScores=TRUE.
runOrder	Temporal order of MS runs. Should be a two column data.table with columns Run and Order, where Run matches the run name output by Spectronaut and Order is an integer. Used to engineer the temporal features defined in anomalyModelFeatureTemporal.
n_trees	Number of trees to use in isolation forest when calculateAnomalyScores=TRUE. Default is 100.
max_depth	Max tree depth to use in isolation forest when calculateAnomalyScores=TRUE. Default is "auto" which calculates depth as $\log_2(N)$ where N is the number of runs. Otherwise must be an integer.
numberOfCores	Number of cores for parallel processing anomaly detection model. When > 1, a logfile named 'MSstats_anomaly_model_progress.log' is created to track progress. Only works for Linux & Mac OS. Default is 1.
use_log_file	logical. If TRUE, information about data processing will be saved to a file.
append	logical. If TRUE, information about data processing will be added to an existing log file.
verbose	logical. If TRUE, information about data processing will be printed to the console.

`log_file_path` character. Path to a file to which information about data processing will be saved. If not provided, such a file will be created automatically. If `append = TRUE`, has to be a valid path to a file.

... additional parameters to `data.table::fread`.

**Value**

data.frame in the MSstats required format.

**Author(s)**

Meena Choi, Olga Vitek

**Examples**

```
spectronaut_raw = system.file("tinytest/raw_data/Spectronaut/spectronaut_input.csv",
                             package = "MSstatsConvert")
spectronaut_raw = data.table::fread(spectronaut_raw)
spectronaut_imported = SpectronauttoMSstatsFormat(spectronaut_raw, use_log_file = FALSE)
head(spectronaut_imported)
```

# Index

## \* internal

- .MSstatsFormat, 35
- .addFractions, 4
- .adjustIntensities, 5
- .aggregatePSMstoPeptideIons, 5
- .checkAnnotation, 6
- .checkDDA, 6
- .checkDuplicatedMeasurements, 7
- .checkMSstatsParams, 7
- .checkMultiRun, 8
- .checkOverlappedFeatures, 8
- .cleanByFeature, 9
- .cleanRawDIANN, 9
- .cleanRawDIAUmpire, 11
- .cleanRawMaxQuant, 11
- .cleanRawMetamorpheus, 12
- .cleanRawOpenMS, 12
- .cleanRawOpenSWATH, 13
- .cleanRawPDMSstats, 14
- .cleanRawPDTMT, 15
- .cleanRawPhilosopher, 15
- .cleanRawProgenesis, 16
- .cleanRawSkyline, 16
- .cleanRawSpectroMineTMT, 17
- .cleanRawSpectronaut, 17
- .countCommonFeatures, 18
- .fillValues, 19
- .filterByPattern, 19
- .filterByScore, 20
- .filterExact, 21
- .filterFewMeasurements, 21
- .filterManyColumns, 22
- .filterOverlapped, 23
- .findAvailable, 23
- .fixBasicColumns, 24
- .fixColumnTypes, 24
- .fixMissingValues, 25
- .getChannelColumns, 25
- .getDataTable, 26
- .getFullDesign, 26
- .getMissingRunsPerFeature, 27
- .getOverlappingFeatures, 27
- .getPhilosopherInput, 28
- .handleFiltering, 28
- .handleFractions, 29
- .handleFractionsLF, 29
- .handleFractionsTMT, 30
- .handleIsotopicPeaks, 30
- .handleSharedPeptides, 31
- .handleSingleFeaturePerProtein, 31
- .logConverterOptions, 32
- .logSuccess, 32
- .makeBalancedDesign, 33
- .makeExactFilterMessage, 33
- .makeScoreFilterMessage, 34
- .mergeAnnotation, 35
- .nullAppender, 36
- .onLoad, 36
- .removeOverlappingFeatures, 37
- .removeSharedPeptides, 37
- .resolveFractionTies, 38
- .selectMSstatsColumns, 38
- .sharedParametersAmongConverters, 39
- .standardizeColnames, 40
- .summarizeMultipleMeasurements, 40
- .summarizeMultiplePSMs, 41
- .validateMSstatsConverterParameters, 41
- getDataType, 50
- MSstatsConvert, 63
- MSstatsInputFiles-class, 65
- .MSstatsFormat, 35
- .addFractions, 4
- .adjustIntensities, 5
- .aggregatePSMstoPeptideIons, 5
- .checkAnnotation, 6
- .checkDDA, 6
- .checkDuplicatedMeasurements, 7
- .checkMSstatsParams, 7
- .checkMultiRun, 8
- .checkOverlappedFeatures, 8
- .cleanByFeature, 9
- .cleanRawDIANN, 9
- .cleanRawDIAUmpire, 11
- .cleanRawMaxQuant, 11

- .cleanRawMetamorpheus, 12
- .cleanRawOpenMS, 12
- .cleanRawOpenSWATH, 13
- .cleanRawPD, 13
- .cleanRawPDMSstats, 14
- .cleanRawPDTMT, 15
- .cleanRawPhilosopher, 15
- .cleanRawProgenesis, 16
- .cleanRawSkyline, 16
- .cleanRawSpectroMineTMT, 17
- .cleanRawSpectronaut, 17
- .countCommonFeatures, 18
- .fillValues, 19
- .filterByPattern, 19
- .filterByScore, 20
- .filterExact, 21
- .filterFewMeasurements, 21
- .filterManyColumns, 22
- .filterOverlapped, 23
- .findAvailable, 23
- .fixBasicColumns, 24
- .fixColumnTypes, 24
- .fixMissingValues, 25
- .getChannelColumns, 25
- .getDataTable, 26
- .getFullDesign, 26
- .getMissingRunsPerFeature, 27
- .getOverlappingFeatures, 27
- .getPhilosopherInput, 28
- .handleFiltering, 28
- .handleFractions, 29
- .handleFractionsLF, 29
- .handleFractionsTMT, 30
- .handleIsotopicPeaks, 30
- .handleSharedPeptides, 31
- .handleSingleFeaturePerProtein, 31
- .logConverterOptions, 32
- .logSuccess, 32
- .makeBalancedDesign, 33
- .makeExactFilterMessage, 33
- .makeScoreFilterMessage, 34
- .mergeAnnotation, 35
- .nullAppender, 36
- .onLoad, 36
- .removeOverlappingFeatures, 37
- .removeSharedPeptides, 37
- .resolveFractionTies, 38
- .selectMSstatsColumns, 38
- .sharedParametersAmongConverters, 39
- .standardizeColnames, 40
- .summarizeMultipleMeasurements, 40
- .summarizeMultiplePSMs, 41
- .validateMSstatsConverterParameters, 41
- as.data.frame.MSstatsValidated, 42
- as.data.table.MSstatsValidated, 43
- CheckDataHealth, 43
- DIANNtoMSstatsFormat, 44
- DIAUmpiretoMSstatsFormat, 47
- FragPipetoMSstatsFormat, 49
- getDataType, 50
- getDataType,MSstatsInputFiles-method (getDataType), 50
- getInputFile, 51
- getInputFile,MSstatsInputFiles-method (getInputFile), 51
- getInputFile,MSstatsPhilosopherFiles-method (getInputFile), 51
- MaxQtoMSstatsFormat, 51
- MaxQtoMSstatsTMTFormat, 53
- MetamorpheusToMSstatsFormat, 55
- MSstatsAnomalyScores, 56
- MSstatsBalancedDesign, 57, 63
- MSstatsClean, 59, 63
- MSstatsClean,MSstatsDIANNFiles-method (MSstatsClean), 59
- MSstatsClean,MSstatsDIAUmpireFiles-method (MSstatsClean), 59
- MSstatsClean,MSstatsMaxQuantFiles-method (MSstatsClean), 59
- MSstatsClean,MSstatsMetamorpheusFiles-method (MSstatsClean), 59
- MSstatsClean,MSstatsOpenMSFiles-method (MSstatsClean), 59
- MSstatsClean,MSstatsOpenSWATHFiles-method (MSstatsClean), 59
- MSstatsClean,MSstatsPhilosopherFiles-method (MSstatsClean), 59
- MSstatsClean,MSstatsProgenesisFiles-method (MSstatsClean), 59
- MSstatsClean,MSstatsProteinProspectorFiles-method (MSstatsClean), 59
- MSstatsClean,MSstatsProteomeDiscovererFiles-method (MSstatsClean), 59
- MSstatsClean,MSstatsSkylineFiles-method (MSstatsClean), 59
- MSstatsClean,MSstatsSpectroMineFiles-method (MSstatsClean), 59
- MSstatsClean,MSstatsSpectronautFiles-method (MSstatsClean), 59

- MSstatsConvert, [63](#)
- MSstatsConvert-package
  - (MSstatsConvert), [63](#)
- MSstatsDIANNFiles-class
  - (MSstatsInputFiles-class), [65](#)
- MSstatsDIAUmpireFiles-class
  - (MSstatsInputFiles-class), [65](#)
- MSstatsFragPipeFiles-class
  - (MSstatsInputFiles-class), [65](#)
- MSstatsImport, [63](#), [64](#)
- MSstatsInputFiles-class, [65](#)
- MSstatsLogsSettings, [63](#), [65](#)
- MSstatsMakeAnnotation, [66](#)
- MSstatsMaxQuantFiles-class
  - (MSstatsInputFiles-class), [65](#)
- MSstatsMetamorpheusFiles-class
  - (MSstatsInputFiles-class), [65](#)
- MSstatsOpenMSFiles-class
  - (MSstatsInputFiles-class), [65](#)
- MSstatsOpenSWATHFiles-class
  - (MSstatsInputFiles-class), [65](#)
- MSstatsPhilosopherFiles-class
  - (MSstatsInputFiles-class), [65](#)
- MSstatsPreprocess, [63](#), [67](#)
- MSstatsProgenesisFiles-class
  - (MSstatsInputFiles-class), [65](#)
- MSstatsProteinProspectorFiles-class
  - (MSstatsInputFiles-class), [65](#)
- MSstatsProteomeDiscovererFiles-class
  - (MSstatsInputFiles-class), [65](#)
- MSstatsSaveSessionInfo, [69](#)
- MSstatsSkylineFiles-class
  - (MSstatsInputFiles-class), [65](#)
- MSstatsSpectroMineFiles-class
  - (MSstatsInputFiles-class), [65](#)
- MSstatsSpectronautFiles-class
  - (MSstatsInputFiles-class), [65](#)
  
- OpenMStoMSstatsFormat, [70](#)
- OpenMStoMSstatsTMTFormat, [71](#)
- OpenSWATHtoMSstatsFormat, [72](#)
  
- PDtoMSstatsFormat, [74](#)
- PDtoMSstatsTMTFormat, [76](#)
- PhilosophertoMSstatsTMTFormat, [77](#)
- ProgenisistoMSstatsFormat, [79](#)
- ProteinProspectortoMSstatsTMTFormat,  
[81](#)
  
- SkylinetoMSstatsFormat, [82](#)
- SpectroMinetoMSstatsTMTFormat, [84](#)
- SpectronauttoMSstatsFormat, [85](#)