

# Package ‘MSstatsConvert’

May 17, 2024

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

**Version** 1.14.0

## 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** 7.3.0

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

**Depends** R (>= 4.0)

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

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

**Collate** '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'  
'converters\_DIANNtoMSstatsFormat.R'  
'converters\_DIAUmpiretoMSstatsFormat.R'  
'converters\_FragPipetoMSstatsFormat.R'  
'converters\_MaxQtoMSstatsFormat.R'  
'converters\_MetamorpheusToMSstatsFormat.R'  
'converters\_OpenMStoMSstatsFormat.R'  
'converters\_OpenSWATHtoMSstatsFormat.R'  
'converters\_PDtoMSstatsFormat.R'  
'converters\_ProgenesistoMSstatsFormat.R'  
'converters\_SkylinetoMSstatsFormat.R'  
'converters\_SpectronauttoMSstatsFormat.R'

'utils\_MSstatsConvert.R' 'utils\_annotation.R'  
 'utils\_balanced\_design.R' 'utils\_checks.R' 'utils\_classes.R'  
 'utils\_clean\_features.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** RELEASE\_3\_19

**git\_last\_commit** cf718ed

**git\_last\_commit\_date** 2024-04-30

**Repository** Bioconductor 3.19

**Date/Publication** 2024-05-16

**Author** Mateusz Staniak [aut, cre],  
 Meena Choi [aut],  
 Ting Huang [aut],  
 Olga Vitek [aut]

**Maintainer** Mateusz Staniak <mtst@mstaniak.pl>

## Contents

.addFractions . . . . .	4
.adjustIntensities . . . . .	5
.aggregatePSMstoPeptideIons . . . . .	5
.checkAnnotation . . . . .	6
.checkDDA . . . . .	6
.checkDuplicatedMeasurements . . . . .	7
.checkMSstatsParams . . . . .	7
.checkMultiRun . . . . .	8
.checkOverlappedFeatures . . . . .	8
.cleanByFeature . . . . .	9
.cleanRawDIANN . . . . .	9
.cleanRawDIAUmpire . . . . .	10
.cleanRawMaxQuant . . . . .	10
.cleanRawMetamorpheus . . . . .	11
.cleanRawOpenMS . . . . .	11
.cleanRawOpenSWATH . . . . .	12
.cleanRawPD . . . . .	12
.cleanRawPDMSstats . . . . .	13
.cleanRawPDTMT . . . . .	14
.cleanRawPhilosopher . . . . .	14
.cleanRawProgenesis . . . . .	15
.cleanRawSkyline . . . . .	16
.cleanRawSpectroMineTMT . . . . .	16
.cleanRawSpectronaut . . . . .	17
.countCommonFeatures . . . . .	17

.fillValues . . . . .	18
.filterByPattern . . . . .	18
.filterByScore . . . . .	19
.filterExact . . . . .	20
.filterFewMeasurements . . . . .	20
.filterManyColumns . . . . .	21
.filterOverlapped . . . . .	22
.findAvailable . . . . .	22
.fixBasicColumns . . . . .	23
.fixColumnTypes . . . . .	23
.fixMissingValues . . . . .	24
.getChannelColumns . . . . .	24
.getCorrectFraction . . . . .	25
.getDataTable . . . . .	25
.getFullDesign . . . . .	26
.getMissingRunsPerFeature . . . . .	26
.getOverlappingFeatures . . . . .	27
.handleFiltering . . . . .	27
.handleFractions . . . . .	28
.handleFractionsLF . . . . .	28
.handleFractionsTMT . . . . .	29
.handleIsotopicPeaks . . . . .	29
.handleSharedPeptides . . . . .	30
.handleSingleFeaturePerProtein . . . . .	30
.logConverterOptions . . . . .	31
.logSuccess . . . . .	32
.makeBalancedDesign . . . . .	32
.makeExactFilterMessage . . . . .	33
.makeScoreFilterMessage . . . . .	33
.mergeAnnotation . . . . .	34
.MSstatsFormat . . . . .	35
.nullAppender . . . . .	35
.onLoad . . . . .	36
.removeOverlappingFeatures . . . . .	36
.removeSharedPeptides . . . . .	37
.selectMSstatsColumns . . . . .	37
.sharedParametersAmongConverters . . . . .	38
.standardizeColnames . . . . .	39
.summarizeMultipleMeasurements . . . . .	39
.summarizeMultiplePSMs . . . . .	40
.validatePDTMTInputColumns . . . . .	40
as.data.frame.MSstatsValidated . . . . .	41
as.data.table.MSstatsValidated . . . . .	41
DIANNtoMSstatsFormat . . . . .	42
DIAUmpiretoMSstatsFormat . . . . .	43
FragPipetoMSstatsFormat . . . . .	45
getDataType . . . . .	47
getInputFile . . . . .	48

MaxQtoMSstatsFormat . . . . .	49
MetamorpheusToMSstatsFormat . . . . .	50
MSstatsBalancedDesign . . . . .	52
MSstatsClean . . . . .	53
MSstatsConvert . . . . .	56
MSstatsImport . . . . .	57
MSstatsInputFiles-class . . . . .	58
MSstatsLogsSettings . . . . .	59
MSstatsMakeAnnotation . . . . .	60
MSstatsPreprocess . . . . .	61
MSstatsSaveSessionInfo . . . . .	63
OpenMStoMSstatsFormat . . . . .	63
OpenSWATHtoMSstatsFormat . . . . .	65
PDtoMSstatsFormat . . . . .	67
ProgenesistoMSstatsFormat . . . . .	69
SkylinetoMSstatsFormat . . . . .	70
SpectronauttoMSstatsFormat . . . . .	72

**Index** **75**

---

<i>.addFractions</i>	<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

---

*.checkAnnotation*      *Check if the annotation is valid*

---

**Description**

Check if the annotation is valid

**Usage**

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

**Arguments**

input	data processed by the MSstatsClean
annotation	annotation created by the MSstatsMakeAnnotation function

**Value**

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

---

*.checkDDA*      *Check validity of DDA data*

---

**Description**

Check validity of DDA data

**Usage**

```
.checkDDA(input)
```

**Arguments**

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

### 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.

### Value

data.table

---

.cleanRawDIANN      *Clean raw Diann files*

---

### Description

Clean raw Diann files

### Usage

```
.cleanRawDIANN(  
  msstats_object,  
  MBR = TRUE,  
  quantificationColumn = "FragmentQuantCorrected"  
)
```

### 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. 'FragmentQuantRaw' can be used instead.

**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)`

**Arguments**

msstats\_object an object of class MSstatsMetamorpheusFiles.

**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\_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

---

.cleanRawPDMSstats     *Clean raw PD output*

---

**Description**

Clean raw PD output

**Usage**

```
.cleanRawPDMSstats(  
  msstats_object,  
  quantification_column,  
  protein_id_column,  
  sequence_column,  
  remove_shared  
)
```

**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"  
)
```

**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)
```

### **Arguments**

*msstats\_object* an object of class *MSstatsSpectronautFiles*.  
*intensity* chr, specifies which column will be used for Intensity.

### **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**

*features\_1* vector of feature names  
*features\_2* vector of feature\_names

### **Value**

character vector of common values of *features\_1* and *features\_2*

---

<code>.fillValues</code>	<i>Set column to a single value</i>
--------------------------	-------------------------------------

---

**Description**

Set column to a single value

**Usage**

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

**Arguments**

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

**Value**

data.table

---

<code>.filterByPattern</code>	<i>Handle filtering by pattern</i>
-------------------------------	------------------------------------

---

**Description**

Handle filtering by pattern

**Usage**

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

**Arguments**

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

**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

input	data.table preprocessed by one of the .cleanRaw* functions.
score_column	chr, name of the column that contains scores.
score_threshold	num, values below or above this threshold will be removed from the data.
direction	chr, if "greater" only values above the threshold will be retained, if "smaller" - below the threshold.
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	If TRUE, filtering will be performed.
drop	if TRUE, score_column 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

<code>input</code>	data.table preprocessed by one of the <code>.cleanRaw*</code> functions.
<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.
<code>filter</code>	lgl, if TRUE, decoy proteins will be removed from the data.
<code>drop</code>	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.  
`features_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**

<code>input</code>	data.table preprocessed by one of the <code>.cleanRaw*</code> functions and merged with annotation.
<code>summary_function</code>	summary function (mean, sum, max) that will be used to pick one feature from multiple overlapping features
<code>overlapped_features</code>	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**

<code>possibilities</code>	possible legal values of a variable
<code>option_set</code>	set of values that includes one of the possibilities
<code>fall_back</code>	if there is none of the possibilities in <code>option_set</code> , or there are multiple hits, default to <code>fall_back</code>

**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**

input	data.table
-------	------------

**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**

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

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

**Value**

`data.table`

---

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

---

**Description**

Get intensity columns from wide-format data

**Usage**

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

**Arguments**

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

**Value**

character vector of column names that correspond to channel intensities



---

.getCorrectFraction     *Get a name of fraction with the largest number of measurements or the largest average intensity*

---

**Description**

Get a name of fraction with the largest number of measurements or the largest average intensity

**Usage**

.getCorrectFraction(input)

**Arguments**

input                    output of MSstatsPreprocess

**Value**

character - label of the fraction that has most measurements or highest mean intensity for a given feature

---

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

---

**Description**

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

**Usage**

.getDataTable(input, ...)

**Arguments**

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

**Value**

data.table

---

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

---

**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 MSstatsPreprocess
<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_column'</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**

<code>input</code>	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

---

`.handleFiltering`

*Handle PSM/proteins scores*

---

**Description**

Handle PSM/proteins scores

**Usage**

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

**Arguments**

`input` data.table preprocessed by one of the `.cleanRaw*` functions.  
`score_filtering` list of by-score filtering controls.  
`exact_filtering` list of exact filtering controls.  
`pattern_filtering` 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)
```

**Arguments**

`input`                    output of MSstatsPreprocess  
`fill_missing`            if TRUE, missing Intensities values will be added to data 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

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.

### 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**

score_column	chr, name of the column that contains scores.
score_threshold	num, values below or above this threshold will be removed from the data.
direction	chr, if "greater" only values above the threshold will be retained, if "smaller" - below the threshold.
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.

**Value**

character - message

---

<i>.mergeAnnotation</i>	<i>Merge annotation with feature data</i>
-------------------------	---

---

**Description**

Merge annotation with feature data

**Usage**

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

**Arguments**

annotation	data.table with annotation
data.table	preprocessed by one of the <i>.cleanRaw*</i> functions.

**Value**

data.table

---

.MSstatsFormat      *Output format for further analysis by MSstats*

---

**Description**

Output format for further analysis by MSstats

**Usage**

.MSstatsFormat(input)

**Arguments**

input              data.table

**Value**

object of class MSstatsValidated that inherits from data.frame

---

.nullAppender      *log4r appender used not to write messages*

---

**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

---

<code>.onLoad</code>	<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

---

<code>.removeOverlappingFeatures</code>	<i>Replace intensities of overlapped fractions with NA, keeping only one fraction</i>
---	---

---

**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

---

*.selectMSstatsColumns* *Select columns for MSstats format*

---

### **Description**

Select columns for MSstats format

### **Usage**

```
.selectMSstatsColumns(input)
```

### **Arguments**

input            data.table

### **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

<code>removeFewMeasurements</code>	TRUE (default) will remove the features that have 1 or 2 measurements across runs.
<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>summaryforMultipleRows</code>	max(default) or sum - when there are multiple measurements for certain feature and certain run, use highest or sum of multiple intensities.
<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>removeProtein_with1Peptide</code>	TRUE will remove the proteins which have only 1 peptide and charge. FALSE is default.
<code>removeOxidationMpeptides</code>	TRUE will remove the peptides including 'oxidation (M)' in modification. FALSE is default.
<code>removeMpeptides</code>	TRUE will remove the peptides including 'M' sequence. FALSE is default.
<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> .

---

`.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)
```

**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.

**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

---

```
.validatePDTMTInputColumns
```

*Helper method to validate input has necessary columns*

---

**Description**

Helper method to validate input has necessary columns

**Usage**

```
.validatePDTMTInputColumns(  
  pd_input,  
  protein_id_column,  
  num_proteins_column,  
  channels  
)
```

**Arguments**

pd\_input                data.frame input  
protein\_id\_column        column name for protein passed from user  
num\_proteins\_column     column name for number of protein groups passed from user  
channels                list of column names for channels



---

```
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

---

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,
  use_log_file = TRUE,
  append = FALSE,
  verbose = TRUE,
  log_file_path = NULL,
  MBR = TRUE,
  quantificationColumn = "FragmentQuantCorrected",
  ...
)
```

## Arguments

input	name of MSstats input report from Diann, which includes feature-level data.
annotation	name of 'annotation.txt' data which includes Condition, BioReplicate, Run.
global_qvalue_cutoff	The global qvalue cutoff
qvalue_cutoff	local qvalue cutoff for library
pg_qvalue_cutoff	local qvalue cutoff for protein groups Run should be the same as filename.
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

<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>MBR</code>	True if analysis was done with match between runs
<code>quantificationColumn</code>	Use 'FragmentQuantCorrected' (default) column for quantified intensities. 'FragmentQuantRaw' can be used instead.
<code>...</code>	additional parameters to <code>data.table::fread</code> .

**Value**

data.frame in the MSstats required format.

**Author(s)**

Elijah Willie

**Examples**

```
input_file_path = system.file("tinytest/raw_data/DIANN/diann_input.tsv",
                             package="MSstatsConvert")
annotation_file_path = system.file("tinytest/raw_data/DIANN/annotation.csv",
                                   package = "MSstatsConvert")
input = data.table::fread(input_file_path)
annot = data.table::fread(annotation_file_path)
output = DIANNtoMSstatsFormat(input, annotation = annot, MBR = FALSE,
                              use_log_file = FALSE)
head(output)
```

---

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(default) or sum - when there are multiple measurements for certain feature and certain run, use highest or sum of multiple intensities.
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(default) or sum - when there are multiple measurements for certain feature and certain run, use highest or sum of multiple intensities.
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	<i>Get type of dataset from an MSstatsInputFiles object.</i>
-------------	--

---

## 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    *Import MaxQuant files*

---

## 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(default) or sum - when there are multiple measurements for certain feature and certain run, use highest or sum of multiple intensities.
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)
```

---

MetamorpheusToMSstatsFormat

*Import Metamorpheus files*

---

**Description**

Import Metamorpheus files

**Usage**

```
MetamorpheusToMSstatsFormat(
  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 Metamorpheus output file, which is tabular format. Use the AllQuantifiedPeaks.tsv file from the Metamorpheus output.
<code>annotation</code>	name of 'annotation.txt' data which includes Condition, BioReplicate.
<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(default) or sum - when there are multiple measurements for certain feature and certain run, use highest or sum of multiple intensities.
<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 in the MSstats required format.

**Author(s)**

Anthony Wu

**Examples**

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

---

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
)
```

**Arguments**

input	data.table processed by the MSstatsPreprocess function
feature_columns	str, names of columns that define spectral features
fill_incomplete	if TRUE (default), Intensity values for missing runs will be added as NA
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.

### 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

*Clean files generated by a signal processing tools.*

---

### 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

**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,  
  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)  
  
## 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"
)

## S4 method for signature 'MSstatsMetamorpheusFiles'
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 chr, specifies which column will be used for Intensity.

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. 'FragmentQuantRaw' can be used instead.

**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.

---

MSstatsImport	<i>Import files from signal processing tools.</i>
---------------	---

---

## 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 data.frame 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.

## 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

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.
base	start of the file name.
pkg_name	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)
```



---

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,
  ...
)
```

## 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.

score\_filtering  
a list of named lists that specify filtering options. Details are provided in the vignette.

exact\_filtering  
a list of named lists that specify filtering options. Details are provided in the vignette.

pattern\_filtering  
a list of named lists that specify filtering options. Details are provided in the vignette.

columns\_to\_fill  
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.

aggregate\_isotopic  
logical. If TRUE, isotopic peaks will be summed.

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

**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(default) or sum - when there are multiple measurements for certain feature and certain run, use highest or sum of multiple intensities.
<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 append = TRUE, has to be a valid path to a file.
<code>...</code>	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)
```

---

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(default) or sum - when there are multiple measurements for certain feature and certain run, use highest or sum of multiple intensities.
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**

```
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(default) or sum - when there are multiple measurements for certain feature and certain run, use highest or sum of multiple intensities.
removeFewMeasurements	TRUE (default) will remove the features that have 1 or 2 measurements across runs.

<code>removeOxidationMpeptides</code>	TRUE will remove the peptides including 'oxidation (M)' in modification. FALSE is default.
<code>removeProtein_with1Peptide</code>	TRUE will remove the proteins which have only 1 peptide and charge. FALSE is default.
<code>which.quantification</code>	Use 'Precursor.Area'(default) column for quantified intensities. 'Intensity' or 'Area' can be used instead.
<code>which.proteinid</code>	Use 'Protein.Accessions'(default) column for protein name. 'Master.Protein.Accessions' can be used instead.
<code>which.sequence</code>	Use 'Sequence'(default) column for peptide sequence. 'Annotated.Sequence' can be used instead.
<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 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)
```

---

 ProgenisistoMSstatsFormat

*Import Progenesis files*


---

## Description

Import Progenesis 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(default) or sum - when there are multiple measurements for certain feature and certain run, use highest or sum of multiple intensities.
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.

<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 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)

```

---

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)
```

---

SpectronauttoMSstatsFormat

*Import Spectronaut files*

---

**Description**

Import Spectronaut files

**Usage**

```
SpectronauttoMSstatsFormat(  
  input,  
  annotation = NULL,  
  intensity = "PeakArea",  
  filter_with_Qvalue = TRUE,  
  qvalue_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 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	'PeakArea'(default) uses not normalized peak area. 'NormalizedPeakArea' uses peak area normalized by Spectronaut.
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 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(default) or sum - when there are multiple measurements for certain feature and certain run, use highest or sum of multiple intensities.
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, 10
- .cleanRawMaxQuant, 10
- .cleanRawMetamorpheus, 11
- .cleanRawOpenMS, 11
- .cleanRawOpenSWATH, 12
- .cleanRawPDMSstats, 13
- .cleanRawPDTMT, 14
- .cleanRawPhilosopher, 14
- .cleanRawProgenesis, 15
- .cleanRawSkyline, 16
- .cleanRawSpectroMineTMT, 16
- .cleanRawSpectronaut, 17
- .countCommonFeatures, 17
- .fillValues, 18
- .filterByPattern, 18
- .filterByScore, 19
- .filterExact, 20
- .filterFewMeasurements, 20
- .filterManyColumns, 21
- .filterOverlapped, 22
- .findAvailable, 22
- .fixBasicColumns, 23
- .fixColumnTypes, 23
- .fixMissingValues, 24
- .getChannelColumns, 24
- .getCorrectFraction, 25
- .getDataTable, 25
- .getFullDesign, 26
- .getMissingRunsPerFeature, 26
- .getOverlappingFeatures, 27
- .handleFiltering, 27
- .handleFractions, 28
- .handleFractionsLF, 28
- .handleFractionsTMT, 29
- .handleIsotopicPeaks, 29
- .handleSharedPeptides, 30
- .handleSingleFeaturePerProtein, 30
- .logConverterOptions, 31
- .logSuccess, 32
- .makeBalancedDesign, 32
- .makeExactFilterMessage, 33
- .makeScoreFilterMessage, 33
- .mergeAnnotation, 34
- .nullAppender, 35
- .onLoad, 36
- .removeOverlappingFeatures, 36
- .removeSharedPeptides, 37
- .selectMSstatsColumns, 37
- .sharedParametersAmongConverters, 38
- .standardizeColnames, 39
- .summarizeMultipleMeasurements, 39
- .summarizeMultiplePSMs, 40
- getDataType, 47
- MSstatsInputFiles-class, 58
- .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, 10
- .cleanRawMaxQuant, 10
- .cleanRawMetamorpheus, 11
- .cleanRawOpenMS, 11
- .cleanRawOpenSWATH, 12
- .cleanRawPD, 12
- .cleanRawPDMSstats, 13
- .cleanRawPDTMT, 14
- .cleanRawPhilosopher, 14
- .cleanRawProgenesis, 14
- .cleanRawSkyline, 16
- .cleanRawSpectroMineTMT, 16
- .cleanRawSpectronaut, 17
- .countCommonFeatures, 17
- .fillValues, 18
- .filterByPattern, 18
- .filterByScore, 19
- .filterExact, 20
- .filterFewMeasurements, 20
- .filterManyColumns, 21
- .filterOverlapped, 22
- .findAvailable, 22
- .fixBasicColumns, 23
- .fixColumnTypes, 23
- .fixMissingValues, 24
- .getChannelColumns, 24
- .getCorrectFraction, 25
- .getDataTable, 25
- .getFullDesign, 26
- .getMissingRunsPerFeature, 26
- .getOverlappingFeatures, 27
- .handleFiltering, 27
- .handleFractions, 28
- .handleFractionsLF, 28
- .handleFractionsTMT, 29
- .handleIsotopicPeaks, 29
- .handleSharedPeptides, 30
- .handleSingleFeaturePerProtein, 30
- .logConverterOptions, 31
- .logSuccess, 32
- .makeBalancedDesign, 32
- .makeExactFilterMessage, 33
- .makeScoreFilterMessage, 33
- .mergeAnnotation, 34
- .nullAppender, 35
- .onLoad, 36
- .removeOverlappingFeatures, 36
- .removeSharedPeptides, 37
- .selectMSstatsColumns, 37
- .sharedParametersAmongConverters, 38
- .standardizeColNames, 39
- .summarizeMultipleMeasurements, 39
- .summarizeMultiplePSMs, 40
- .validatePDTMTInputColumns, 40
- as.data.frame.MSstatsValidated, 41
- as.data.table.MSstatsValidated, 41
- DIANNtoMSstatsFormat, 42
- DIAUmpiretoMSstatsFormat, 43
- FragPipetoMSstatsFormat, 45
- getDataType, 47
- getDataType, MSstatsInputFiles-method  
(getDataType), 47
- getInputFile, 48
- getInputFile, MSstatsInputFiles-method  
(getInputFile), 48
- getInputFile, MSstatsPhilosopherFiles-method  
(getInputFile), 48
- MaxQtoMSstatsFormat, 49
- MetamorpheusToMSstatsFormat, 50
- MSstatsBalancedDesign, 52, 57
- MSstatsClean, 53, 57
- MSstatsClean, MSstatsDIANNFiles-method  
(MSstatsClean), 53
- MSstatsClean, MSstatsDIAUmpireFiles-method  
(MSstatsClean), 53
- MSstatsClean, MSstatsMaxQuantFiles-method  
(MSstatsClean), 53
- MSstatsClean, MSstatsMetamorpheusFiles-method  
(MSstatsClean), 53
- MSstatsClean, MSstatsOpenMSFiles-method  
(MSstatsClean), 53
- MSstatsClean, MSstatsOpenSWATHFiles-method  
(MSstatsClean), 53
- MSstatsClean, MSstatsPhilosopherFiles-method  
(MSstatsClean), 53
- MSstatsClean, MSstatsProgenesisFiles-method  
(MSstatsClean), 53
- MSstatsClean, MSstatsProteomeDiscovererFiles-method  
(MSstatsClean), 53
- MSstatsClean, MSstatsSkylineFiles-method  
(MSstatsClean), 53

MSstatsClean,MSstatsSpectroMineFiles-method  
(MSstatsClean), 53

MSstatsClean,MSstatsSpectronautFiles-method  
(MSstatsClean), 53

MSstatsConvert, 56

MSstatsDIANNFiles-class  
(MSstatsInputFiles-class), 58

MSstatsDIAUmpireFiles-class  
(MSstatsInputFiles-class), 58

MSstatsFragPipeFiles-class  
(MSstatsInputFiles-class), 58

MSstatsImport, 57, 57

MSstatsInputFiles-class, 58

MSstatsLogsSettings, 57, 59

MSstatsMakeAnnotation, 60

MSstatsMaxQuantFiles-class  
(MSstatsInputFiles-class), 58

MSstatsMetamorpheusFiles-class  
(MSstatsInputFiles-class), 58

MSstatsOpenMSFiles-class  
(MSstatsInputFiles-class), 58

MSstatsOpenSWATHFiles-class  
(MSstatsInputFiles-class), 58

MSstatsPhilosopherFiles-class  
(MSstatsInputFiles-class), 58

MSstatsPreprocess, 57, 61

MSstatsProgenesisFiles-class  
(MSstatsInputFiles-class), 58

MSstatsProteomeDiscovererFiles-class  
(MSstatsInputFiles-class), 58

MSstatsSaveSessionInfo, 63

MSstatsSkylineFiles-class  
(MSstatsInputFiles-class), 58

MSstatsSpectroMineFiles-class  
(MSstatsInputFiles-class), 58

MSstatsSpectronautFiles-class  
(MSstatsInputFiles-class), 58

OpenMStoMSstatsFormat, 63

OpenSWATHtoMSstatsFormat, 65

PDtoMSstatsFormat, 67

ProgenesistoMSstatsFormat, 69

SkylinetoMSstatsFormat, 70

SpectronauttoMSstatsFormat, 72