

# Package ‘MSstatsBig’

June 30, 2025

**Type** Package

**Title** MSstats Preprocessing for Larger than Memory Data

**Version** 1.6.0

**Description** MSstats package provide tools for preprocessing, summarization and differential analysis of mass spectrometry (MS) proteomics data. Recently, some MS protocols enable acquisition of data sets that result in larger than memory quantitative data. MSstats functions are not able to process such data. MSstatsBig package provides additional converter functions that enable processing larger than memory data sets.

**License** Artistic-2.0

**Encoding** UTF-8

**RoxygenNote** 7.2.3

**Imports** arrow, DBI, dplyr, MSstats, MSstatsConvert, readr, sparklyr, utils

**Suggests** knitr, rmarkdown

**VignetteBuilder** knitr

**biocViews** MassSpectrometry, Proteomics, Software

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

**git\_branch** RELEASE\_3\_21

**git\_last\_commit** c5abd1d

**git\_last\_commit\_date** 2025-04-15

**Repository** Bioconductor 3.21

**Date/Publication** 2025-06-29

**Author** Mateusz Staniak [aut, cre],  
Devon Kohler [aut]

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

## Contents

|                                        |          |
|----------------------------------------|----------|
| bigFragPipetoMSstatsFormat . . . . .   | 2        |
| bigSpectronautoMSstatsFormat . . . . . | 3        |
| MSstatsAddAnnotationBig . . . . .      | 5        |
| MSstatsPreprocessBig . . . . .         | 5        |
| <b>Index</b>                           | <b>8</b> |

---

bigFragPipetoMSstatsFormat

*Convert out-of-memory FragPipe files to MSstats format.*

---

## Description

Convert out-of-memory FragPipe files to MSstats format.

## Usage

```
bigFragPipetoMSstatsFormat(
  input_file,
  output_file_name,
  backend,
  max_feature_count = 20,
  filter_unique_peptides = FALSE,
  aggregate_psms = FALSE,
  filter_few_obs = FALSE,
  remove_annotation = FALSE,
  connection = NULL
)
```

## Arguments

|                        |                                                                                                                                                                 |
|------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------|
| input_file             | name of the input text file in 10-column MSstats format.                                                                                                        |
| output_file_name       | name of an output file which will be saved after pre-processing                                                                                                 |
| backend                | "arrow" or "sparklyr". Option "sparklyr" requires a spark installation and connection to spark instance provided in the 'connection' parameter.                 |
| max_feature_count      | maximum number of features per protein. Features will be selected based on highest average intensity.                                                           |
| filter_unique_peptides | If TRUE, shared peptides will be removed. Please refer to the 'Details' section for additional information.                                                     |
| aggregate_psms         | If TRUE, multiple measurements per PSM in a Run will be aggregated (by taking maximum value). Please refer to the 'Details' section for additional information. |

`filter_few_obs` If TRUE, feature with less than 3 observations across runs will be removed. Please refer to the 'Details' section for additional information.

`remove_annotation` If TRUE, columns BioReplicate and Condition will be removed to reduce output file size. These will need to be added manually later before using `dataProcess` function. Only applicable to sparklyr backend.

`connection` Connection to a spark instance created with the 'spark\_connect' function from 'sparklyr' package.

**Value**

either arrow object or sparklyr table that can be optionally collected into memory by using `dplyr::collect` function.

**Examples**

```
converted_data <- bigFragPipetoMSstatsFormat(
  system.file("extdata", "fgexample.csv", package = "MSstatsBig"),
  "output_file.csv",
  backend = "arrow")
converted_data <- dplyr::collect(converted_data)
head(converted_data)
```

---

bigSpectronauttoMSstatsFormat

*Convert out-of-memory Spectronaut files to MSstats format.*

---

**Description**

Convert out-of-memory Spectronaut files to MSstats format.

**Usage**

```
bigSpectronauttoMSstatsFormat(
  input_file,
  output_file_name,
  backend,
  filter_by_excluded = FALSE,
  filter_by_identified = FALSE,
  filter_by_qvalue = TRUE,
  qvalue_cutoff = 0.01,
  max_feature_count = 20,
  filter_unique_peptides = FALSE,
  aggregate_psms = FALSE,
  filter_few_obs = FALSE,
  remove_annotation = FALSE,
  connection = NULL
)
```

**Arguments**

|                                     |                                                                                                                                                                                                                         |
|-------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>input_file</code>             | name of the input text file in 10-column MSstats format.                                                                                                                                                                |
| <code>output_file_name</code>       | name of an output file which will be saved after pre-processing                                                                                                                                                         |
| <code>backend</code>                | "arrow" or "sparklyr". Option "sparklyr" requires a spark installation and connection to spark instance provided in the 'connection' parameter.                                                                         |
| <code>filter_by_excluded</code>     | if TRUE, will filter by the 'F.ExcludedFromQuantification' column.                                                                                                                                                      |
| <code>filter_by_identified</code>   | if TRUE, will filter by the 'EG.Identified' column.                                                                                                                                                                     |
| <code>filter_by_qvalue</code>       | if TRUE, will filter by EG.Qvalue and PG.Qvalue columns.                                                                                                                                                                |
| <code>qvalue_cutoff</code>          | cutoff which will be used for q-value filtering.                                                                                                                                                                        |
| <code>max_feature_count</code>      | maximum number of features per protein. Features will be selected based on highest average intensity.                                                                                                                   |
| <code>filter_unique_peptides</code> | If TRUE, shared peptides will be removed. Please refer to the 'Details' section for additional information.                                                                                                             |
| <code>aggregate_psms</code>         | If TRUE, multiple measurements per PSM in a Run will be aggregated (by taking maximum value). Please refer to the 'Details' section for additional information.                                                         |
| <code>filter_few_obs</code>         | If TRUE, feature with less than 3 observations across runs will be removed. Please refer to the 'Details' section for additional information.                                                                           |
| <code>remove_annotation</code>      | If TRUE, columns BioReplicate and Condition will be removed to reduce output file size. These will need to be added manually later before using <code>dataProcess</code> function. Only applicable to sparklyr backend. |
| <code>connection</code>             | Connection to a spark instance created with the 'spark_connect' function from 'sparklyr' package.                                                                                                                       |

**Value**

either arrow object or sparklyr table that can be optionally collected into memory by using `dplyr::collect` function.

**Examples**

```
converted_data <- bigSpectronautoMSstatsFormat(
  system.file("extdata", "spectronaut_input.csv", package = "MSstatsBig"),
  "output_file.csv",
  backend="arrow")
converted_data <- dplyr::collect(converted_data)
head(converted_data)
```

---

MSstatsAddAnnotationBig

*Merge annotation to output of MSstatsPreprocessBig*


---

## Description

Merge annotation to output of MSstatsPreprocessBig

## Usage

```
MSstatsAddAnnotationBig(input, annotation)
```

## Arguments

|            |                                |
|------------|--------------------------------|
| input      | output of MSstatsPreprocessBig |
| annotation | run annotation                 |

## Value

table of 'input' and 'annotation' merged by Run column.

## Examples

```
converted_data <- bigFragPipetoMSstatsFormat(
  system.file("extdata", "fgexample.csv", package = "MSstatsBig"),
  "output_file.csv",
  backend = "arrow")
converted_data <- dplyr::collect(converted_data)
head(converted_data)
# Change annotation as an example:
converted_data$Condition <- NULL
converted_data$BioReplicate <- NULL
annot <- data.frame(Run = unique(converted_data[["Run"]]))
annot$BioReplicate <- rep(1:53, times = 2)
annot$Condition <- rep(1:2, each = 53)
head(MSstatsAddAnnotationBig(converted_data, annot))
```

---

|                      |                                                                                              |
|----------------------|----------------------------------------------------------------------------------------------|
| MSstatsPreprocessBig | <i>General converter for larger-than-memory csv files in MSstats format 10-column format</i> |
|----------------------|----------------------------------------------------------------------------------------------|

---

## Description

General converter for larger-than-memory csv files in MSstats format 10-column format

**Usage**

```
MSstatsPreprocessBig(
  input_file,
  output_file_name,
  backend,
  max_feature_count = 20,
  filter_unique_peptides = FALSE,
  aggregate_psms = FALSE,
  filter_few_obs = FALSE,
  remove_annotation = FALSE,
  connection = NULL
)
```

**Arguments**

|                                     |                                                                                                                                                                                                                         |
|-------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>input_file</code>             | name of the input text file in 10-column MSstats format.                                                                                                                                                                |
| <code>output_file_name</code>       | name of an output file which will be saved after pre-processing                                                                                                                                                         |
| <code>backend</code>                | "arrow" or "sparklyr". Option "sparklyr" requires a spark installation and connection to spark instance provided in the 'connection' parameter.                                                                         |
| <code>max_feature_count</code>      | maximum number of features per protein. Features will be selected based on highest average intensity.                                                                                                                   |
| <code>filter_unique_peptides</code> | If TRUE, shared peptides will be removed. Please refer to the 'Details' section for additional information.                                                                                                             |
| <code>aggregate_psms</code>         | If TRUE, multiple measurements per PSM in a Run will be aggregated (by taking maximum value). Please refer to the 'Details' section for additional information.                                                         |
| <code>filter_few_obs</code>         | If TRUE, feature with less than 3 observations across runs will be removed. Please refer to the 'Details' section for additional information.                                                                           |
| <code>remove_annotation</code>      | If TRUE, columns BioReplicate and Condition will be removed to reduce output file size. These will need to be added manually later before using <code>dataProcess</code> function. Only applicable to sparklyr backend. |
| <code>connection</code>             | Connection to a spark instance created with the 'spark_connect' function from 'sparklyr' package.                                                                                                                       |

**Details**

Filtering and aggregation may be very time consuming and the ability to perform them in a given R session depends on available memory, settings of external packages, etc. Hence, all value of related parameters ('filter\_unique\_peptides', 'aggregate\_psms', 'filter\_few\_obs') are set to FALSE by default and only feature selection is performed, which saves both computation time and memory. Appropriately configured spark backend provides the most consistent way to perform these operations.

**Value**

either arrow object or sparklyr table that can be optionally collected into memory by using `dplyr::collect` function.

**Examples**

```
converted_data <- bigFragPipetoMSstatsFormat(  
  system.file("extdata", "fgexample.csv", package = "MSstatsBig"),  
  "tencol_format.csv",  
  backend="arrow")  
procd <- MSstatsPreprocessBig("tencol_format.csv", "proc_out.csv", backend = "arrow")  
head(dplyr::collect(procd))
```

# Index

bigFragPipetoMSstatsFormat, [2](#)  
bigSpectronauttoMSstatsFormat, [3](#)  
MSstatsAddAnnotationBig, [5](#)  
MSstatsPreprocessBig, [5](#)