

# Package ‘MSstatsBig’

May 9, 2026

**Type** Package

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

**Version** 1.11.0

**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** 7.3.3

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

**Suggests** testthat, mockery, knitr, rmarkdown

**VignetteBuilder** knitr

**biocViews** MassSpectrometry, Proteomics, Software

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

**git\_branch** devel

**git\_last\_commit** ce69a87

**git\_last\_commit\_date** 2026-04-28

**Repository** Bioconductor 3.24

**Date/Publication** 2026-05-08

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

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

## Contents

.writeChunkToFile . . . . .	2
bigDIANNtoMSstatsFormat . . . . .	2
bigFragPipetoMSstatsFormat . . . . .	4
bigSpectronautoMSstatsFormat . . . . .	5

cleanDIANNChunk . . . . .	7
MSstatsAddAnnotationBig . . . . .	8
MSstatsPreprocessBig . . . . .	8
reduceBigDIANN . . . . .	10

## Index 12

---

<code>.writeChunkToFile</code>	<i>Write chunk to file</i>
--------------------------------	----------------------------

---

### Description

Write chunk to file

### Usage

```
.writeChunkToFile(input, output_path, pos)
```

### Arguments

<code>input</code>	Data frame
<code>output_path</code>	Path to output file
<code>pos</code>	Chunk position

---

`bigDIANNtoMSstatsFormat`

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

---

### Description

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

### Usage

```
bigDIANNtoMSstatsFormat(
  input_file,
  annotation = NULL,
  output_file_name,
  backend,
  MBR = TRUE,
  quantificationColumn = "FragmentQuantCorrected",
  global_qvalue_cutoff = 0.01,
  qvalue_cutoff = 0.01,
  pg_qvalue_cutoff = 0.01,
  max_feature_count = 100,
  filter_unique_peptides = FALSE,
  aggregate_psms = FALSE,
  filter_few_obs = FALSE,
  remove_annotation = FALSE,
  calculateAnomalyScores = FALSE,
```

```

    anomalyModelFeatures = c(),
    connection = NULL
)

```

### Arguments

**input\_file** name of the input text file in 10-column MSstats format.

**annotation** name of 'annotation.txt' data which includes Condition, BioReplicate, Run.

**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.

**MBR** True if analysis was done with match between runs

**quantificationColumn** Use 'FragmentQuantCorrected'(default) column for quantified intensities for DIANN 1.8.x. Use 'FragmentQuantRaw' for quantified intensities for DIANN 1.9.x. Use 'auto' for quantified intensities for DIANN 2.x where each fragment intensity is a separate column, e.g. Fr0Quantity.

**global\_qvalue\_cutoff** The qvalue cutoff for the Q.Value column, i.e. the run-specific precursor q-value. Default is 0.01.

**qvalue\_cutoff** If MBR is false, the qvalue cutoff for the Global.Q.Value column, i.e. global precursor q-value. If MBR is true, the qvalue cutoff for the Lib.Q.Value column, i.e. the q-value for the library created after the first MBR pass. Default is 0.01.

**pg\_qvalue\_cutoff** If MBR is false, the qvalue cutoff for the Global.PG.Q.Value column, i.e. the global q-value for the protein group. If MBR is true, the qvalue cutoff for the Lib.PG.Q.Value column, i.e. the protein group q-value for the library created after the first MBR pass. Default is 0.01.

**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.

**calculateAnomalyScores** If TRUE, will carry anomaly model features through pipeline

**anomalyModelFeatures** Character vector of column names to be carried through the pipeline

**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.

---

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 = 100,
  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.

**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,
  intensity = "F.NormalizedPeakArea",
  filter_by_excluded = FALSE,
  filter_by_identified = FALSE,
  filter_by_qvalue = FALSE,
  qvalue_cutoff = 0.01,
  max_feature_count = 100,
  filter_unique_peptides = FALSE,
  aggregate_psms = FALSE,
  filter_few_obs = FALSE,
  remove_annotation = FALSE,
  calculateAnomalyScores = FALSE,
  anomalyModelFeatures = c(),
  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>intensity</code>	Name of the intensity column to be used in Spectronaut

<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>calculateAnomalyScores</code>	If TRUE, will carry anomaly model features through pipeline
<code>anomalyModelFeatures</code>	Character vector of column names to be carried through the pipeline
<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 <- bigSpectronauttoMSstatsFormat(
  system.file("extdata", "spectronaut_input.csv", package = "MSstatsBig"),
  "output_file.csv",
  backend="arrow")
converted_data <- dplyr::collect(converted_data)
head(converted_data)
```

---

cleanDIANNChunk	<i>Clean a single chunk of DIANN data</i>
-----------------	---

---

## Description

Clean a single chunk of DIANN data

## Usage

```
cleanDIANNChunk(  
  input,  
  output_path,  
  MBR,  
  quantificationColumn,  
  pos,  
  global_qvalue_cutoff = 0.01,  
  qvalue_cutoff = 0.01,  
  pg_qvalue_cutoff = 0.01,  
  calculateAnomalyScores = FALSE,  
  anomalyModelFeatures = c(),  
  annotation = NULL  
)
```

## Arguments

input	Data frame chunk
output_path	Path to output file
MBR	Boolean, whether MBR was used
quantificationColumn	Name of intensity column
pos	Chunk position (1 for first chunk, >1 for subsequent)
global_qvalue_cutoff	Global Q-value cutoff
qvalue_cutoff	Q-value cutoff
pg_qvalue_cutoff	Protein group Q-value cutoff
calculateAnomalyScores	Boolean for MSstats+ Model
anomalyModelFeatures	Character vector of features to use for MSstats+ Model
annotation	Annotation file or data frame

---

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


---

### 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 = 100,
  filter_unique_peptides = FALSE,
  aggregate_psms = FALSE,
  filter_few_obs = FALSE,
  remove_annotation = FALSE,
  calculateAnomalyScores = FALSE,
  anomalyModelFeatures = c(),
  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 dataProcess function. Only applicable to sparklyr backend.
<code>calculateAnomalyScores</code>	If TRUE, will carry anomaly model features through pipeline
<code>anomalyModelFeatures</code>	Character vector of column names to be carried through the pipeline
<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))
```

---

reduceBigDIANN

*Read and clean a large DIANN file in chunks*

---

### Description

Read and clean a large DIANN file in chunks

### Usage

```
reduceBigDIANN(
  input_file,
  output_path,
  MBR = TRUE,
  quantificationColumn = "FragmentQuantCorrected",
  global_qvalue_cutoff = 0.01,
  qvalue_cutoff = 0.01,
  pg_qvalue_cutoff = 0.01,
  calculateAnomalyScores = FALSE,
  anomalyModelFeatures = c(),
  annotation = NULL
)
```

### Arguments

<code>input_file</code>	Path to the input DIANN file
<code>output_path</code>	Path to the output CSV file
<code>MBR</code>	Boolean, whether MBR was used
<code>quantificationColumn</code>	Name of the column containing intensity values
<code>global_qvalue_cutoff</code>	Global Q-value cutoff
<code>qvalue_cutoff</code>	Q-value cutoff

pg\_qvalue\_cutoff            Protein group Q-value cutoff  
calculateAnomalyScores        Boolean for MSstats+ Model  
anomalyModelFeatures        Character vector of features to use for MSstats+ Model  
annotation                    Annotation file or data frame

**Value**

NULL. Writes to file.

# Index

## \* **internal**

- [.writeChunkToFile, 2](#)
  - [cleanDIANNChunk, 7](#)
  - [reduceBigDIANN, 10](#)
- [.writeChunkToFile, 2](#)
  
- [bigDIANNtoMSstatsFormat, 2](#)
- [bigFragPipetoMSstatsFormat, 4](#)
- [bigSpectronauttoMSstatsFormat, 5](#)
  
- [cleanDIANNChunk, 7](#)
  
- [MSstatsAddAnnotationBig, 8](#)
- [MSstatsPreprocessBig, 8](#)
  
- [reduceBigDIANN, 10](#)