

# Package: MSstatsQC (via r-universe)

June 3, 2026

**Version** 2.30.0

**Type** Package

**Title** Longitudinal system suitability monitoring and quality control for proteomic experiments

**Description** MSstatsQC is an R package which provides longitudinal system suitability monitoring and quality control tools for proteomic experiments.

**License** Artistic License 2.0

**URL** <http://msstats.org/msstatsqc>

**LazyData** True

**BugReports** <https://groups.google.com/forum/#!forum/msstatsqc>

**RoxygenNote** 7.3.3

**Imports** dplyr,plyr, plotly, ggplot2, ggExtra, stats, grid, MSnbase, qcmetrics, h2o, FrF2, car, reshape2, jsonlite

**Suggests** knitr, rmarkdown, testthat, RforProteomics

**VignetteBuilder** knitr

**biocViews** Software, QualityControl, Proteomics, MassSpectrometry, Normalization

**Encoding** UTF-8

**Config/pak/sysreqs** cmake libglib-dev libgmp3-dev make default-jdk libicu-dev libuv1-dev libxml2-dev libnetcdf-dev libssl-dev zlib1g-dev

**Repository** <https://bioc-release.r-universe.dev>

**Date/Publication** 2026-04-28 12:46:29 UTC

**RemoteUrl** <https://github.com/bioc/MSstatsQC>

**RemoteRef** RELEASE\_3\_23

**RemoteSha** 893580f7edb6a68b0cd2bd310049f3ac06247c83

## Contents

ChangePointEstimator . . . . .	2
CUSUMChart . . . . .	4
DataProcess . . . . .	5
DecisionMap . . . . .	6
MissingDataMap . . . . .	7
MSnbaseToMSstatsQC . . . . .	8
MSstatsQC.ML.deployR . . . . .	8
MSstatsQC.ML.sim.size.detectR . . . . .	9
MSstatsQC.ML.trainR . . . . .	10
mzQCToMSstatsQC . . . . .	11
QCloudDDA . . . . .	12
QCloudSRM . . . . .	13
QuiCDIA . . . . .	13
RadarPlot . . . . .	14
RemoveMissing . . . . .	15
RiverPlot . . . . .	16
S9Site54 . . . . .	17
XmRChart . . . . .	18
<b>Index</b>	<b>20</b>

---

ChangePointEstimator *A function to identify the time of a change in the mean or variability of a metric*

---

### Description

A function to identify the time of a change in the mean or variability of a metric

### Usage

```
ChangePointEstimator(
  data = NULL,
  peptide,
  L = 1,
  U = 5,
  metric,
  normalization = TRUE,
  ytitle = "Change Point Plot - mean",
  type = "mean",
  selectMean = NULL,
  selectSD = NULL
)
```

**Arguments**

data	comma-separated (.csv), metric file. It should contain a "Precursor" column and the metrics columns. It should also include "Annotations" for each observation.
peptide	the name of precursor of interest.
L	Lower bound of the guide set.
U	Upper bound of the guide set.
metric	the name of metric of interest.
normalization	TRUE metric is standardized and FALSE if not standardized.
ytitle	the y-axis title of the plot. Defaults to "Change Point Plot - mean". The x-axis title is by default "QCno-name of peptide"
type	the type of the control chart. Two values can be assigned, "mean" or "variability". Default is "mean".
selectMean	the mean of a metric. It is used when mean is known. It is NULL when mean is not known. The default is NULL.
selectSD	the standard deviation of a metric. It is used when standard deviation is known. It is NULL when mean is not known. The default is NULL.

**Value**

A plot of likelihood statistics versus time per peptide and metric generated from CP.data.prepare data frame.

**Examples**

```
# First process the data to make sure it's ready to use
sampleData <- DataProcess(S9Site54)
head(sampleData)
# Find the name of the peptides
levels(sampleData$Precursor)
# Calculate change point statistics
ChangePointEstimator(data = sampleData, peptide = "VLVLDTDYK", metric = "BestRetentionTime")
ChangePointEstimator(
  data = sampleData, peptide = "VLVLDTDYK", metric = "BestRetentionTime",
  ytitle = "Change Point Plot - variability", type = "variability"
)
ChangePointEstimator(
  data = sampleData, peptide = "VLVLDTDYK", metric = "BestRetentionTime",
  selectMean = 27.78, selectSD = 8.19
)
ChangePointEstimator(data = sampleData, peptide = "DDGSWEVIEGYR", metric = "TotalArea")
ChangePointEstimator(
  data = sampleData, peptide = "DDGSWEVIEGYR", metric = "TotalArea",
  selectMean = 35097129, selectSD = 34132861
)
ChangePointEstimator(data = sampleData, peptide = "TAAYVNAIEK", metric = "MaxFWHM")
```

---

CUSUMChart	<i>A function to create cumulative sum charts for mean (CUSUMm) and cumulative sum charts for variability (CUSUMv) control charts</i>
------------	---

---

### Description

A function to create cumulative sum charts for mean (CUSUMm) and cumulative sum charts for variability (CUSUMv) control charts

### Usage

```
CUSUMChart(
  data = NULL,
  peptide,
  L = 1,
  U = 5,
  metric,
  normalization = TRUE,
  ytitle = "CUSUMm",
  type = "mean",
  selectMean = NULL,
  selectSD = NULL,
  referenceValue = 0.5,
  decisionInterval = 5
)
```

### Arguments

data	comma-separated (.csv), metric file. It should contain a "Precursor" column and the metrics columns. It should also include "Annotations" for each observation.
peptide	the name of precursor of interest.
L	Lower bound of the guide set.
U	Upper bound of the guide set.
metric	the name of metric of interest.
normalization	TRUE if metric is standardized and FALSE if not standardized.
ytitle	the y-axis title of the plot. Defaults to "CUSUMm". The x-axis title is by default "Time : name of peptide"
type	the type of the control chart. Two values can be assigned, "mean" or "variability". Default is "mean"
selectMean	the mean of a metric. It is used when mean is known. It is NULL when mean is not known. The default is NULL.
selectSD	the standard deviation of a metric. It is used when standard deviation is known. It is NULL when mean is not known. The default is NULL.
referenceValue	the value that is used to tune the control chart for a proper shift size
decisionInterval	the threshold to detect an out-of-control observation

**Value**

A plot of positive and negative CUSUM statistics versus time per peptide and metric generated from CUSUM.data.prepare data frame.

**Examples**

```
# First process the data to make sure it's ready to use
sampleData <- DataProcess(S9Site54)
head(sampleData)
# Find the name of the peptides
levels(sampleData$Precursor)
# Calculate CUSUM statistics
CUSUMChart(data = sampleData, peptide = "VLVLDTDYK", metric = "BestRetentionTime")
CUSUMChart(
  data = sampleData, peptide = "VLVLDTDYK", metric = "BestRetentionTime",
  ytitle = "CUSUMv", type = "variability"
)
CUSUMChart(
  data = sampleData, peptide = "VLVLDTDYK", metric = "BestRetentionTime",
  selectMean = 27.78, selectSD = 8.19
)
CUSUMChart(data = sampleData, peptide = "DDGSWEVIEGYR", metric = "TotalArea")
CUSUMChart(
  data = sampleData, peptide = "DDGSWEVIEGYR", metric = "TotalArea",
  selectMean = 35097129, selectSD = 34132861
)
CUSUMChart(data = sampleData, peptide = "TAAYVNAIEK", metric = "MaxFWHM")
```

---

DataProcess

*A data processing function*

---

**Description**

A data processing function

**Usage**

```
DataProcess(data = NULL)
```

**Arguments**

data                    Comma-separated (\*.csv), QC file format. It should contain a Precursor column and the metrics columns.

**Value**

A data frame that processes using input.sanity.check function.

**Examples**

```
# The data is "S9Site54" which is defined in the package.
data <- DataProcess(S9Site54)
```

---

DecisionMap	<i>A function to create heatmaps to compare performance with user defined performance criteria</i>
-------------	--

---

**Description**

A function to create heatmaps to compare performance with user defined performance criteria

**Usage**

```
DecisionMap(
  data = NULL,
  method = "XmR",
  peptideThresholdRed = 0.7,
  peptideThresholdYellow = 0.5,
  L = 1,
  U = 5,
  type = "mean",
  title = "heatmap plot",
  listMean = NULL,
  listSD = NULL
)
```

**Arguments**

data	Comma-separated (*.csv), QC file format. It should contain a Precursor column and the metrics columns.
method	It is either "CUSUM" or "XmR"
peptideThresholdRed	Is a threshold that marks percentage of peptides above it red on the heatmap. Defaults to 0.7
peptideThresholdYellow	Is a threshold that marks percentage of peptides above it and below the peptide-ThresholdRed, yellow on the heatmap. Defaults to 0.5
L	Lower bound of the guide set. Defaults to 1
U	Upper bound of the guide set. Defaults to 5
type	can take two values, "mean" or "dispersion". Defaults to "mean"
title	the title of the plot. Defaults to "heatmap plot"
listMean	List of the means for the metrics. If you don't know the means leave it as NULL and they will be calculated automatically by using L and U. The default is NULL.

`listSD` List of the standard deviations for the metrics. If you don't know the standard deviations leave it as NULL and they will be calculated automatically by using L and U. The default is NULL.

### Value

A heatmap to aggregate results per metric generated from `heatmap.DataFrame` data frame.

### Examples

```
# First process the data to make sure it's ready to use
sampleData <- DataProcess(S9Site54)
head(sampleData)
# Draw Decision maker plot
DecisionMap(data = sampleData, method = "CUSUM")
DecisionMap(data = sampleData, method = "CUSUM", type = "variability")
DecisionMap(data = sampleData, method = "XmR")
DecisionMap(data = sampleData, method = "XmR", type = "variability")
```

---

MissingDataMap      *A function to summarize missing values*

---

### Description

A function to summarize missing values

### Usage

```
MissingDataMap(data)
```

### Arguments

`data`              Processed data

### Value

A plot of missing values.

### Examples

```
# The data is "S9Site54" which is defined in the package.
data <- DataProcess(S9Site54)
MissingDataMap(data)
```

---

MSnbaseToMSstatsQC *A function to convert MSnbase files to MSstatsQC format*

---

**Description**

A function to convert MSnbase files to MSstatsQC format

**Usage**

```
MSnbaseToMSstatsQC(msfile)
```

**Arguments**

msfile            data file to be converted

**Value**

A data frame that can be used with MSstatsQC

A csv file that is converted from raw files

**Examples**

```
library(RforProteomics)

msfile <- getPXD000001mzXML()

MSnbaseToMSstatsQC(msfile)
```

---

MSstatsQC.ML.deployR *A function to test random forest classifiers for QC data*

---

**Description**

A function to test random forest classifiers for QC data

**Usage**

```
MSstatsQC.ML.deployR(Test.set, guide.set, rf_model)
```

**Arguments**

Test.set	comma-separated (.csv), metric file. It should contain a "Precursor" column and the metrics columns. It should also include "Annotations" for each run.
guide.set	comma-separated (.csv), metric file. It should contain a "Precursor" column and the metrics columns. It should also include "Annotations" for each run.
rf_model	the model that was trained previously by MSstatsQC-ML training process

**Value**

Probability of failure predictions based on a trained model

**Examples**

```
S9Site54.dataML <- DataProcess(MSstatsQC::S9Site54[, ])
colnames(S9Site54.dataML)[1] <- c("idfile")
colnames(S9Site54.dataML)[2] <- c("peptide")
S9Site54.dataML$peptide <- as.factor(S9Site54.dataML$peptide)
S9Site54.dataML$idfile <- as.numeric(S9Site54.dataML$idfile)
S9Site54.dataML <- within(S9Site54.dataML, rm(Annotations, missing))
guide.set <- dplyr::filter(S9Site54.dataML, idfile <= 20)

rf_model <- MSstatsQC.ML.trainR(guide.set, sim.size = 10)

Test.set <- dplyr::filter(S9Site54.dataML, idfile > 20)

MSstatsQC.ML.deployR(Test.set, guide.set, rf_model = rf_model)

Test.set <- S9Site54.dataML

MSstatsQC.ML.deployR(Test.set, guide.set, rf_model = rf_model)
```

---

MSstatsQC.ML.sim.size.detectR

*A function to train random forest classifiers for QC data*

---

**Description**

A function to train random forest classifiers for QC data

**Usage**

```
MSstatsQC.ML.sim.size.detectR(guide.set, sim.start, sim.end)
```

**Arguments**

guide.set            comma-separated (.csv), metric file. It should contain a "Precursor" column and the metrics columns. It should also include "Annotations" for each run.

sim.start            enter min simulation size.

sim.end              enter max simulation size.

**Value**

a plot for sim.size vs performance

**Examples**

```
# First process the data to make sure it's ready to use
S9Site54.dataML <- DataProcess(MSstatsQC::S9Site54[, ])
colnames(S9Site54.dataML)[1] <- c("idfile")
colnames(S9Site54.dataML)[2] <- c("peptide")
S9Site54.dataML$peptide <- as.factor(S9Site54.dataML$peptide)
S9Site54.dataML$idfile <- as.numeric(S9Site54.dataML$idfile)
S9Site54.dataML <- within(S9Site54.dataML, rm(Annotations, missing))
guide.set <- dplyr::filter(S9Site54.dataML, idfile <= 20)

MSstatsQC.ML.sim.size.detectR(guide.set, sim.start = 10, sim.end = 2500)
```

---

MSstatsQC.ML.trainR    *A function to train random forest classifiers for QC data*

---

**Description**

A function to train random forest classifiers for QC data

**Usage**

```
MSstatsQC.ML.trainR(
  guide.set,
  sim.size,
  guide.set.annotations = NULL,
  nfold = 5,
  a = 1.5,
  b = 2
)
```

**Arguments**

guide.set	comma-separated (.csv), metric file. It should contain a "Precursor" column and the metrics columns. It should also include "Annotations" for each run.
sim.size	enter simulation size.
guide.set.annotations	comma-separated (.csv), metric file with annotations such as pass and fail.
nfolds	fold for cross validation
a	lower threshold to define shift size
b	upper threshold to define shift size

**Value**

A trained model and performance indicators from train/validation/test splits

**Examples**

```
S9Site54.dataML <- DataProcess(MSstatsQC::S9Site54[, ])
colnames(S9Site54.dataML)[1] <- c("idfile")
colnames(S9Site54.dataML)[2] <- c("peptide")
S9Site54.dataML$peptide <- as.factor(S9Site54.dataML$peptide)
S9Site54.dataML$idfile <- as.numeric(S9Site54.dataML$idfile)
S9Site54.dataML <- within(S9Site54.dataML, rm(Annotations, missing))
guide.set <- dplyr::filter(S9Site54.dataML, idfile <= 20)

MSstatsQC.ML.trainR(guide.set, sim.size = 10)
```

---

mzQCToMSstatsQC

*A function to convert mzQC files to MSstatsQC format*


---

**Description**

A function to convert mzQC files to MSstatsQC format

**Usage**

```
mzQCToMSstatsQC(mzQCfile)
```

**Arguments**

mzQCfile	data file to be converted
----------	---------------------------

**Value**

A data frame that can be used with MSstatsQC

A csv file that is converted from raw files

**Examples**

```
library(RforProteomics)

msfile <- getPXD000001mzXML()

mzQCToMSstatsQC(msfile)
```

---

QCloudDDA

*DDA QC data from QCloud System*

---

**Description**

QC results generated from QCloud system

**Usage**

```
data(QCloudDDA)
```

**Format**

csv

**Details**

DDA QC data from QCloud System

**Value**

An example dataset generated from QCloud system

**Examples**

```
head(QCloudDDA)
```

---

QCloudSRM

*SRM QC data from QCloud System*

---

**Description**

QC results generated from QCloud system

**Usage**

```
data(QCloudSRM)
```

**Format**

csv

**Details**

SRM QC data from QCloud System

**Value**

An example dataset generated from QCloud system

**Examples**

```
head(QCloudSRM)
```

---

QuiCDIA

*DIA iRT data from QuiC System*

---

**Description**

QC results generated from QuiC system

**Usage**

```
data(QuiCDIA)
```

**Format**

csv

**Details**

DIA iRT data from QuiC System

**Value**

An example dataset generated from QuiC system

**Examples**

```
head(QuiCDIA)
```

---

RadarPlot	<i>A function to create radar plot to aggregate results from X and mR charts or CUSUMm and CUSUMv charts.</i>
-----------	---

---

**Description**

A function to create radar plot to aggregate results from X and mR charts or CUSUMm and CUSUMv charts.

**Usage**

```
RadarPlot(
  data = NULL,
  L = 1,
  U = 5,
  method = "XmR",
  listMean = NULL,
  listSD = NULL
)
```

**Arguments**

<code>data</code>	omma-separated (.csv), metric file. It should contain a "Precursor" column and the metrics columns. It should also include "Annotations" for each observation.
<code>L</code>	lower bound of the guide set.
<code>U</code>	upper bound of the guide set.
<code>method</code>	defines the method selected to construct control charts.
<code>listMean</code>	list of the means for each metric. It is used when mean is known. It is NULL when mean is not known. The default is NULL.
<code>listSD</code>	list of the standard deviations for each metric. It is used when standard deviation is known. It is NULL when mean is not known. The default is NULL. automatically by using L and U. The default is NULL.

**Value**

A radar plot to aggregate results per metric generated from `XmR.Radar.Plot.DataFrame` data frame or `CUSUM.Radar.Plot.DataFrame` data frame.

## Examples

```
# First process the data to make sure it's ready to use
sampleData <- DataProcess(S9Site54)
head(sampleData)
# Draw XmR radar plot
RadarPlot(data = sampleData)
RadarPlot(data = sampleData, method = "CUSUM")
RadarPlot(
  data = sampleData,
  listMean = list(
    "BestRetentionTime" = 27.78,
    "TotalArea" = 35097129,
    "MaxFWHM" = 0.28,
    "MinStartTime" = 24
  ),
  listSD = list(
    "BestRetentionTime" = 8.19,
    "TotalArea" = 34132861,
    "MaxFWHM" = 0.054,
    "MinStartTime" = 24
  )
)
```

---

RemoveMissing

*A data processing function for removing missing values*

---

## Description

A data processing function for removing missing values

## Usage

```
RemoveMissing(data = NULL)
```

## Arguments

**data** Comma-separated (\*.csv), QC file format. It should contain a Precursor column and the metrics columns.

## Value

A data frame that processes using `input.sanity.check` function.

## Examples

```
# The data is "S9Site54" which is defined in the package.
data <- RemoveMissing(S9Site54)
```

---

RiverPlot	<i>A function to create river plot to aggregate results from X and mR charts or CUSUMm and CUSUMv charts.</i>
-----------	---

---

### Description

A function to create river plot to aggregate results from X and mR charts or CUSUMm and CUSUMv charts.

### Usage

```
RiverPlot(
  data = NULL,
  L = 1,
  U = 5,
  method = "XmR",
  listMean = NULL,
  listSD = NULL
)
```

### Arguments

<code>data</code>	omma-separated (.csv), metric file. It should contain a "Precursor" column and the metrics columns. It should also include "Annotations" for each observation.
<code>L</code>	lower bound of the guide set.
<code>U</code>	upper bound of the guide set.
<code>method</code>	defines the method selected to construct control charts.
<code>listMean</code>	list of the means for each metric. It is used when mean is known. It is NULL when mean is not known. The default is NULL.
<code>listSD</code>	list of the standard deviations for each metric. It is used when standard deviation is known. It is NULL when mean is not known. The default is NULL.

### Value

A river plot to aggregate results per metric generated from XmR.Summary.DataFrame data frame or CUSUM.Summary.DataFrame data frame.

### Examples

```
# First process the data to make sure it's ready to use
sampleData <- DataProcess(S9Site54)
head(sampleData)
# Draw XmR summary plot
RiverPlot(data = sampleData)
RiverPlot(
  data = sampleData, L = 1, U = 20, method = "XmR",
  listMean = list(
```

```
      "BestRetentionTime" = 27.78,  
      "TotalArea" = 35097129,  
      "MaxFWHM" = 0.28,  
      "MinStartTime" = 24  
    ),  
    listSD = list(  
      "BestRetentionTime" = 8.19,  
      "TotalArea" = 34132861,  
      "MaxFWHM" = 0.054,  
      "MinStartTime" = 24  
    )  
  )  
)
```

---

S9Site54

*CPTAC study 9.1 site 54 dataset*

---

### **Description**

system suitability testing results generated during CPTAC Study 9.1 for Site 54

### **Usage**

```
data(S9Site54)
```

### **Format**

csv

### **Details**

CPTAC system suitability testing data for Site 54 from Study 9.1

### **Value**

An example dataset generated from CPTAC study 9.1

### **References**

<http://www.mcponline.org/content/early/2015/02/18/mcp.M114.047050>

### **Examples**

```
head(S9Site54)
```

---

XmRChart	<i>A function to construct individual (X) and moving range (mR) control charts</i>
----------	--

---

**Description**

A function to construct individual (X) and moving range (mR) control charts

**Usage**

```
XmRChart(
  data = NULL,
  peptide,
  L = 1,
  U = 5,
  metric,
  normalization = FALSE,
  ytitle = "Individual observations",
  type = "mean",
  selectMean = NULL,
  selectSD = NULL
)
```

**Arguments**

data	comma-separated (.csv), metric file. It should contain a "Precursor" column and the metrics columns. It should also include "Annotations" for each observation.
peptide	the name of precursor of interest.
L	Lower bound of the guide set.
U	Upper bound of the guide set.
metric	the name of metric of interest.
normalization	TRUE if metric is standardized and FALSE if not standardized.
ytitle	the y-axis title of the plot. Defaults to "Individual observations". The x-axis title is by default "Time : name of peptide"
type	the type of the control chart. Two values can be assigned, "mean" or "variability". Default is "mean".
selectMean	the mean of a metric. It is used when mean is known. It is NULL when mean is not known. The default is NULL.
selectSD	the standard deviation of a metric. It is used when standard deviation is known. It is NULL when mean is not known. The default is NULL.

**Value**

A plot of individual values or moving ranges versus time per peptide and metric generated from XmR.data.prepare data frame.

**Examples**

```
# First process the data to make sure it's ready to use
sampleData <- DataProcess(S9Site54)
head(sampleData)
# Find the name of the peptides
levels(sampleData$Precursor)
# Calculate X and mR statistics
XmRChart(data = sampleData, peptide = "VLVLDTDYK", metric = "BestRetentionTime")
XmRChart(
  data = sampleData, peptide = "VLVLDTDYK", metric = "BestRetentionTime",
  ytitle = "moving ranges", type = "variability"
)
XmRChart(
  data = sampleData, peptide = "VLVLDTDYK", metric = "BestRetentionTime",
  selectMean = 27.78, selectSD = 8.19
)
XmRChart(data = sampleData, peptide = "DDGSWEVIEGYR", metric = "TotalArea")
XmRChart(
  data = sampleData, peptide = "DDGSWEVIEGYR", metric = "TotalArea",
  selectMean = 35097129, selectSD = 34132861
)
XmRChart(data = sampleData, peptide = "TAAYVNAIEK", metric = "MaxFWHM")
XmRChart(data = sampleData, peptide = "LVNELTEFAK", metric = "MinStartTime")
```

# Index

- \* **MSnbase**,
  - MSnbaseToMSstatsQC, 8
- \* **Sum**,
  - CUSUMChart, 4
- \* **XmR**,
  - XmRChart, 18
- \* **XmR**
  - RadarPlot, 14
  - RiverPlot, 16
- \* **change**
  - ChangePointEstimator, 2
- \* **chart**
  - ChangePointEstimator, 2
  - CUSUMChart, 4
  - XmRChart, 18
- \* **control**
  - ChangePointEstimator, 2
  - CUSUMChart, 4
  - XmRChart, 18
- \* **cumulative**
  - CUSUMChart, 4
- \* **datasets**
  - QCloudDDA, 12
  - QCloudSRM, 13
  - QuiCDIA, 13
  - S9Site54, 17
- \* **deployment**
  - MSstatsQC.ML.deployR, 8
- \* **engineering**
  - MSstatsQC.ML.trainR, 10
- \* **feature**
  - MSstatsQC.ML.trainR, 10
- \* **heatmap**
  - DecisionMap, 6
- \* **input**
  - MSnbaseToMSstatsQC, 8
  - mzQCToMSstatsQC, 11
- \* **learning**,
  - MSstatsQC.ML.deployR, 8
  - MSstatsQC.ML.sim.size.detectR, 9
  - MSstatsQC.ML.trainR, 10
- \* **machine**
  - MSstatsQC.ML.deployR, 8
  - MSstatsQC.ML.sim.size.detectR, 9
  - MSstatsQC.ML.trainR, 10
- \* **mzqc**,
  - mzQCToMSstatsQC, 11
- \* **point**,
  - ChangePointEstimator, 2
- \* **qcmetrics**,
  - MSnbaseToMSstatsQC, 8
  - mzQCToMSstatsQC, 11
- \* **simulation**
  - MSstatsQC.ML.sim.size.detectR, 9
- \* **size**
  - MSstatsQC.ML.sim.size.detectR, 9
- \* **training**,
  - MSstatsQC.ML.trainR, 10
- ChangePointEstimator, 2
- CUSUMChart, 4
- DataProcess, 5
- DecisionMap, 6
- MissingDataMap, 7
- MSnbaseToMSstatsQC, 8
- MSstatsQC.ML.deployR, 8
- MSstatsQC.ML.sim.size.detectR, 9
- MSstatsQC.ML.trainR, 10
- mzQCToMSstatsQC, 11
- QCloudDDA, 12
- QCloudSRM, 13
- QuiCDIA, 13
- RadarPlot, 14
- RemoveMissing, 15
- RiverPlot, 16

*INDEX*

21

S9Site54, [17](#)

XmRChart, [18](#)