

# Package ‘MSbox’

May 7, 2026

**Type** Package

**Title** Mass Spectrometry Tools

**Version** 1.4.8

**Author** Yonghui Dong

**Maintainer** Yonghui Dong <yonghui.dong@gmail.com>

**Description** Common mass spectrometry tools described in John Roboz (2013) <doi:10.1201/b15436>. It allows checking element isotopes, calculating (isotope labelled) exact monoisotopic mass, m/z values and mass accuracy, and inspecting possible contaminant mass peaks, examining possible adducts in electrospray ionization (ESI) and matrix-assisted laser desorption ionization (MALDI) ion sources.

**Depends** R (>= 3.5.0)

**Imports** stringr, crayon, xml2, stats

**License** GPL-2

**URL** <https://github.com/YonghuiDong/MSbox>

**BugReports** <https://github.com/YonghuiDong/MSbox/issues/new>

**Encoding** UTF-8

**RoxygenNote** 7.2.2

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2022-12-08 14:42:31 UTC

## Contents

adduct . . . . .	2
contam . . . . .	3
describe . . . . .	3
doStat . . . . .	4
E_iso . . . . .	4

getCV . . . . .	5
getFC . . . . .	6
getMax . . . . .	6
getP . . . . .	7
Iso_mass . . . . .	7
Iso_mz . . . . .	8
mass . . . . .	8
mz . . . . .	9
ppm . . . . .	10
searchDB . . . . .	10
what . . . . .	11
<b>Index</b>	<b>12</b>

---

adduct	<i>Common adducts</i>
--------	-----------------------

---

## Description

calculate common adduct ions in positive or negative ion mode

## Usage

```
adduct(F, mode = c("+", "-"))
```

## Arguments

F	chemical formula, case insensitive
mode	ionization mode, either positive '+' or negative '-'

## Author(s)

Yonghui Dong

## Examples

```
adduct('C1H4', mode = '-')  
adduct('C1h4', mode = '+')
```

---

contam	<i>Contaminants in MS</i>
--------	---------------------------

---

**Description**

check the possible contaminants

**Usage**

```
contam(mz, mode = NULL, ppm = 10)
```

**Arguments**

mz	suspected m/z value
mode	ionization mode, either positive '+' or negative '-'
ppm	mass tolerance, default value = 10

**Author(s)**

Yonghui Dong

**Examples**

```
contam(33.0335, ppm = 10, mode = '+')  
contam(44.998, ppm = 10, mode = '-')
```

---

describe	<i>Get the compound information</i>
----------	-------------------------------------

---

**Description**

get compound formula and structure from <https://cactus.nci.nih.gov/chemical/structure>

**Usage**

```
describe(chem, representation = "formula", info = FALSE)
```

**Arguments**

chem	chemical name of the compound
representation	representation methods, formula is default
info	extra molecular information that users can query

**Author(s)**

Yonghui Dong

**Examples**

```
## Not run:
describe('malic acid', "formula")
describe(c('malic acid', 'citric acid', 'tartaric acid'), "smiles")

## End(Not run)
```

---

doStat *Performing statistics*

---

**Description**

performing statistics, including calculating fold change, p-values and VIP values

**Usage**

```
doStat(x, Group = NULL)
```

**Arguments**

x                    sample ion intensity matrix, row sample, column feature.  
Group                sample group information

**Value**

a dataframe with statistical information

**Examples**

```
dat <- matrix(runif(2*300), ncol = 2, nrow = 300)
rownames(dat) <- 1:dim(dat)[1]
myGroup <- rep_len(LETTERS[1:3], 300)
ret <- doStat(dat, Group = myGroup)
```

---

E\_iso *Element isotopes*

---

**Description**

check element isotope information

**Usage**

```
E_iso(S)
```

**Arguments**

S element, can be element symbol (i.e. C) or full name (i.e. Carbon). Both Element symbol and full name are case insensitive.

**Author(s)**

Yonghui Dong

**Examples**

```
E_iso('Na') # element symbol
E_iso('nA') # element symbol, case insensitive
E_iso('Carbon') # element full name
E_iso('carBon') # element full name, case insensitive
```

---

getCV

*Calculate coefficient of variation (CV)*

---

**Description**

Calculate coefficient of variation (CV), also known as relative standard deviation (RSD) among different sample groups

**Usage**

```
getCV(x, Group = NULL)
```

**Arguments**

x sample ion intensity matrix, row sample, column feature.  
Group sample group information

**Value**

a dataframe with mean values and cv

**Examples**

```
dat <- matrix(runif(2*300), ncol = 2, nrow = 300)
myGroup <- rep_len(LETTERS[1:2], 300)
ret <- getCV(dat, Group = myGroup)
```

getFC *calculate fold change*

---

**Description**

calculate fold change among different samples.

**Usage**

```
getFC(x, Group = NULL)
```

**Arguments**

x                    sample ion intensity matrix, row sample, column feature.  
Group                sample group information

**Value**

a dataframe with mean values and fold changes

**Examples**

```
dat <- matrix(runif(2*300), ncol = 2, nrow = 300)  
myGroup <- rep_len(LETTERS[1:2], 300)  
ret <- getFC(dat, Group = myGroup)
```

---

getMax *Get the sample name which has the max ion intensity*

---

**Description**

get the sample name which has the max ion intensity

**Usage**

```
getMax(x)
```

**Arguments**

x                    sample ion intensity matrix, row sample, column feature.

**Value**

a data frame

**Examples**

```
dat <- cbind.data.frame(mz = c(100, 101, 300), mz2 = c(0, 0, 1), mz3 = c(1, 9, 1))
rownames(dat) <- c("A", "B", "C")
out <- getMax(dat)
```

getP

*get p-values***Description**

get p-values from Post Hoc analysis

**Usage**

```
getP(x, Group = NULL)
```

**Arguments**

x                    sample ion intensity matrix, row sample, column feature.  
Group                sample group information

**Value**

a data frame

**Examples**

```
dat <- matrix(runif(2*300), ncol = 2, nrow = 300)
myGroup <- rep_len(LETTERS[1:3], 300)
out <- getP(dat, Group = myGroup)
```

Iso\_mass

*Isotope labelled molecular mass***Description**

Calculate isotope labelled molecular mass

**Usage**

```
Iso_mass(F, iso)
```

**Arguments**

F                    chemical formula, case insensitive  
iso                  labelled elements, case insensitive

**Author(s)**

Yonghui Dong

**Examples**

```
Iso_mass(F = 'C7H6O4', iso = '[13]C2[2]H3') # Two 13C and three 2H are labeled
```

---

Iso_mz	<i>Isotope labelled molecular mass</i>
--------	--

---

**Description**

Calculate isotope labelled m/z

**Usage**

```
Iso_mz(F, iso, z)
```

**Arguments**

F	chemical formula, case insensitive
iso	labelled elements, case insensitive
z	charge

**Author(s)**

Yonghui Dong

**Examples**

```
Iso_mz(F = 'C7H6O4', iso = '[13]C2[2]H3', z = -1) # Two 13C and three 2H are labeled
```

---

mass	<i>molecular mass</i>
------	-----------------------

---

**Description**

calculate accurate molecular mass

**Usage**

```
mass(F, caseSensitive = FALSE)
```

**Arguments**

**F** chemical formula, case insensitive

**caseSensitive** if case sensitive is 'FALSE' (default), the elements are separated by numbers. for instance, Carbon dioxide can be written as 'c1o2' or any combination of the two elements in lower or upper cases. However, the number of elements should be clearly stated in the chemical formula. if case sensitive is 'TRUE', the elements are separated by upper case letters. For instance, Carbon dioxide must be written as 'C1O2' or 'CO2'. You don't need to write the number of the element if it is 1.

**Author(s)**

Yonghui Dong

**Examples**

```
mass('C7h7o1')
mass('C7H7O', caseSensitive = TRUE)
mass(c('C7H7O4', 'C'), caseSensitive = TRUE) # vector input
mass(c('c7h7O4', 'c1'))
```

---

mz

---

*Calculate accurate mass-to-charge ratio*


---

**Description**

Calculate accurate mass-to-charge ratio (m/z)

**Usage**

```
mz(m, z, caseSensitive = FALSE)
```

**Arguments**

**m** chemical formula of an ion, case insensitive

**z** charge

**caseSensitive** if case sensitive is 'FALSE' (default), the elements are separated by numbers. for instance, Carbon dioxide can be written as 'c1o2' or any combination of the two elements in lower or upper cases. However, the number of elements should be clearly stated in the chemical formula. if case sensitive is 'TRUE', the elements are separated by upper case letters. For instance, Carbon dioxide must be written as 'C1O2' or 'CO2'. You don't need to write the number of the element if it is 1.

**Author(s)**

Yonghui Dong

**Examples**

```
mz('C7h7o1', z = 1)
mz('C7H7O', z = 1, caseSensitive = TRUE)
mz(c('C7H7O4', 'C'), z = -1, caseSensitive = TRUE) # vector input
mz(c('c7h7O4', 'c1'), z = -1)
```

---

ppm	<i>mass accuracy</i>
-----	----------------------

---

**Description**

calculate the mass accuracy of measured m/z. lazy input allowed

**Usage**

```
ppm(m, t, lazy = TRUE)
```

**Arguments**

m	measured m/z
t	theoretical m/z
lazy	if lazy input is allowed

**Author(s)**

Yonghui Dong

**Examples**

```
ppm(155.03383, 155.03388) # with m/z value
ppm(155.03383, .03388) # lazy input when the integer parts of m and t are the same
ppm(155.03384, mz('C7H7O4', z = 1)) # with ion formula
```

---

searchDB	<i>Search in customized database</i>
----------	--------------------------------------

---

**Description**

search in customized database based on accurate m/z and RT

**Usage**

```
searchDB(DF, DB, ppm = 5, RT = 0.2, useRT = FALSE)
```

**Arguments**

DF	input file, should contain at least a column named mz
DB	database, should contain at least a column named mz
ppm	mass tolerance, default 5ppm
RT	retention time tolerance, default 0.2min
useRT	should RT be considered during database search?

**Author(s)**

Yonghui Dong

**Examples**

```
DF <- cbind.data.frame(mz = c(100.001, 100.1), RT = c(10, 11))
DB <- cbind.data.frame(mz = c(100.001, 100.1), RT = c(10, 12.1))
searchDB(DF, DB, ppm = 5, RT = 0.2, useRT = TRUE)
```

---

what *search for m/z in from the idiom metabolomics database*

---

**Description**

tentative metabolite identification based on m/z value search

**Usage**

```
what(myMZ, mode = NULL, ppm = 5, useDB = "HMDB")
```

**Arguments**

myMZ	m/z values
mode	ionization mode, either positive '+' or negative '-'
ppm	mass tolerance, default value = 10
useDB	which database to use, HMDB or KEGG? default is HMDB

**Author(s)**

Yonghui Dong

**Examples**

```
a = what(133.014, mode = '-', ppm = 10)
```

# Index

adduct, [2](#)

contam, [3](#)

describe, [3](#)

doStat, [4](#)

E\_iso, [4](#)

getCV, [5](#)

getFC, [6](#)

getMax, [6](#)

getP, [7](#)

Iso\_mass, [7](#)

Iso\_mz, [8](#)

mass, [8](#)

mz, [9](#)

ppm, [10](#)

searchDB, [10](#)

what, [11](#)