# Package: InterpretMSSpectrum (via r-universe)

October 26, 2026

**Type** Package

**Title** Interpreting High Resolution Mass Spectra

**Version** 1.5.3

**Date** 2026-04-10

**Maintainer** Jan Lisec <jan.lisec@bam.de>

**Description** High resolution mass spectrometry yields often large data sets of spectra from compounds which are not present in available libraries. These spectra need to be annotated and interpreted. 'InterpretMSSpectrum' provides a set of functions to perform such tasks for Electrospray-Ionization and Atmospheric-Pressure-Chemical-Ionization derived data in positive and negative ionization mode.

**License** GPL-3

**URL** https://github.com/janlisec/InterpretMSSpectrum

**Depends** R (>= 2.10.0)

**Imports** enviPat, plyr, rcdk

**Suggests** DBI, doParallel, foreach, parallel, Rdisop, RSQLite, testthat (>= 3.0.0), vdiffr

**Config/testthat/edition** 3

**Encoding** UTF-8

**Language** en-US

**LazyData** true

**RoxygenNote** 7.3.3

**NeedsCompilation** no

**Author** Jan Lisec [aut, cre] (ORCID: <https://orcid.org/0000-0003-1220-2286>), Jaeger Carsten [aut]

**Repository** https://cran.r-universe.dev

**Date/Publication** 2026-04-10 13:53:01 UTC

**RemoteUrl** https://github.com/cran/InterpretMSSpectrum

**RemoteRef** HEAD

**RemoteSha** 2ca83a467ec008e16f361face2b730f754993785

# Contents


Adducts . . . 2
apci_spectrum . . . 3
chemical_elements . . . 3
CountChemicalElements . . . 4
esi_spectrum . . . 4
findMAIN . . . 5
fml2expr . . . 7
GenerateMetaboliteSQLiteDB . . . 8
get_exactmass . . . 9
GetGroupFactor . . . 10
IMS_parallel . . . 10
InterpretMSSpectrum . . . 11
InterpretTP . . . 13
mScore . . . 15
neutral_losses_APCI . . . 16
neutral_losses_ESI . . . 17
OrbiMS1 . . . 18
param.default . . . 19
PlotSpec . . . 20
ReadSpecClipboard . . . 22
sendToMSF . . . 23

**Index** **25**


---

`Adducts` *Default adduct lists used by* `findMAIN`.

---

## Description

Default adduct lists used by `findMAIN`.

## Usage

```
data(Adducts)
```

## Format

A list of two character vectors:

**Positive** default adducts used in ESI(+) mode

**Negative** default adducts used in ESI(-) mode

## Source

A reasonable selection of frequent adducts based on the list in R-package CAMERA

---

`apci_spectrum` *APCI spectrum*

---

## Description

Example spectrum of Glutamic acid (3TMS) measured on a Bruker impact II.

## Usage

```
data(apci_spectrum)
```

## Format

A data frame with 47 observations on the following 2 variables:

**mz** a numeric vector ion masses

**int** a numeric vector of intensities

---

`chemical_elements` *List of chemical elements.*

---

## Description

List of chemical elements.

## Usage

```
data(chemical_elements)
```

## Format

A data frame with 103 observations on the following 2 variables:

**name** a character vector of elemental abbreviations

**mass** a numeric vector of exact masses of the elements main isotope

`CountChemicalElements` *CountChemicalElements.*

## Description

`CountChemicalElements` will split a character (chemical formula) into its elements and count their occurrence.

## Usage

```
CountChemicalElements(x = NULL, ele = NULL)
```

## Arguments

| | |
|---|---|
| `x` | Chemical formula. |
| `ele` | Character vector of elements to count particularly or counting all contained if NULL. |

## Details

No testing for any chemical alphabet is performed. Elements may occur several times and will be summed up in this case without a warning.

## Value

A named numeric with counts for all contained or specified elements.

`esi_spectrum` *ESI spectrum*

## Description

Example spectrum of Sedoheptulose 7-phosphate measured on a Bruker impact II.

## Usage

```
data(esi_spectrum)
```

## Format

A data frame with 42 observations on the following 2 variables:

**mz** a numeric vector ion masses

**int** a numeric vector of intensities

findMAIN *findMAIN.*

## Description

findMAIN will evaluate an ESI spectrum for the potential main adducts, rank obtained suggestions and allow the deduction of the neutral mass of the measured molecule.

## Usage

```
findMAIN(
  spec,
  adductmz = NULL,
  ionmode = c("positive", "negative")[1],
  adducthyp = NULL,
  ms2spec = NULL,
  rules = NULL,
  mzabs = 0.01,
  ppm = 5,
  mainpkthr = 0.005,
  collapseResults = TRUE
)

## S3 method for class 'findMAIN'
plot(x, rank = 1, correct_mass = NULL, ...)

## S3 method for class 'findMAIN'
print(x, ...)
```

## Arguments

| | |
|---|---|
| spec | A mass spectrum. Either a matrix or data frame, the first two columns of which are assumed to contain the 'mz' and 'intensity' values, respectively. |
| adductmz | Manually specified peak for which adducthyp should be tested, or 'NULL' (default), to test all main peaks. What is a main peak, is governed by mainpkthr. |
| ionmode | Ionization mode, either "positive" or "negative". Can be abbreviated. |
| adducthyp | Adduct hypotheses to test for each main peak. Defaults to c("[M+H]+","[M+Na]+","[M+K]+") for positive mode and c("[M-H]-","[M+Cl]-","[M+HCOOH-H]-") for negative mode. |
| ms2spec | Second spectrum limiting main peak selection. If available, MS^E or bbCID spectra may allow further exclusion of false positive adduct ions, as ions of the intact molecule (protonated molecule, adduct ions) should have lower intensity in the high-energy trace than in low-energy trace. |
| rules | Adduct/fragment relationships to test, e.g. c("[M+Na]+", "[M+H-H2O]+"), or 'NULL' for default set (see Adducts) |

| | |
|---|---|
| mzabs | Allowed mass error, absolute (Da). |
| ppm | Allowed mass error, relative (ppm), which is _added_ to 'mzabs'. |
| mainpkthr | Intensity threshold for main peak selection, relative to base peak. |
| collapseResults | If a neutral mass hypothesis was found more than once (due to multiple adducts suggesting the same neutral mass), return only the one with the highest adduct peak. Should normally kept at TRUE, the default. |
| x | Object of class findMAIN. |
| rank | Rank of the suggestion to plot (can be a numeric vector). |
| correct_mass | If provided will indicate correct suggestion by green color. |
| ... | Further parameters. |

## Details

Electrospray ionization (ESI) mass spectra frequently contain a number of different adduct ions, multimers and in-source fragments [M+H]+, [M+Na]+, [2M+H]+, [M+H-H2O]+, making it difficult to decide on the compound's neutral mass. This functions aims at determining the main adduct ion and its type (protonated, sodiated etc.) of a spectrum, allowing subsequent database searches e.g. using MS-FINDER, SIRIUS or similar.

## Value

A list-like 'findMAIN' object for which 'print', 'summary' and 'plot' methods are available. Each list element represents a potential spectra annotation, ranked according to a combined score. The spectrum is annotated with columns indicating the determined isotopic groups (isogr) and their likely charge. Further, information on the potential set of adducts and their ppm error is attached. The score aims to integrate all this information using formula S=sum(w_i x s_i). In short, we sum up i weighted score components (currently i=4). Currently these components are calculated based on the explained intensity (adduct sets which annotate a higher amount of the total spectrum intensity are better), the mass error (adduct sets with lower mass error are better), the support by isotopic peaks (adduct sets with fitting isotopes are better) and the number of adducts (adduct sets with a larger number of adducts are better). The individual scores for each adduct set are attached as an attribute to the respective list element and can be easily observed by applying the 'summary' or the 'plot' function on the 'findMAIN' object.

## References

Jaeger C, Meret M, Schmitt CA, Lisec J (2017), <doi:10.1002/rcm.7905>.

## Examples

```
utils::data(esi_spectrum, package = "InterpretMSSpectrum")
fmr <- InterpretMSSpectrum::findMAIN(esi_spectrum)
plot(fmr)
head(summary(fmr))
InterpretMSSpectrum::InterpretMSSpectrum(fmr[[1]], precursor=263, param="ESIpos")
fmr <- InterpretMSSpectrum::findMAIN(esi_spectrum[6:9,], adducthyp = "[M+H]+")
plot(fmr)
```

```
# set up a spectrum containing a double charged peak
spec <- data.frame(mz = c(372.1894, 372.6907, 373.1931, 380), int = c(100, 40, 8, 2))
InterpretMSSpectrum:::findiso(spec)
# allow a double charged adduct hypothesis (not standard)
fmr <- InterpretMSSpectrum::findMAIN(spec, adducthyp = c("[M+H]+", "[M+2H]2+"))
summary(fmr)
attr(fmr[[1]],"scores")
plot(fmr, rank = 1:4)
plot(fmr, rank = 2)

# add the correct M+H to this spectrum as a minor peak
spec <- rbind(spec, c(742.3648+1.007, 10))
(fmr <- InterpretMSSpectrum::findMAIN(spec, adducthyp = c("[M+H]+", "[M+2H]2+")))
summary(fmr)
plot(fmr, rank = 1)
plot(fmr, rank = 2)

# compare specific hypotheses manually
# get correct result
InterpretMSSpectrum::findMAIN(spec, adductmz = 743.3718, adducthyp = "[M+H]+")
# enforce wrong result
InterpretMSSpectrum::findMAIN(spec, adductmz = 743.3718, adducthyp = "[M+2H]2+")
```

---

`fml2expr` *Turn text with chemical formulas into expression vector.*

---

## Description

It is often helpful to annotate a Figure with chemical formulas. However, to increase readability of chemical formulas, certain conventions have to be met. These concern, among others, superscript and subscript text which, in R, can only be provided via expressions. 'fml2expr' will convert any character vector into an expression vector, aiming to identify and format potential contained chemical formulas.

## Usage

```
fml2expr(x, expr = TRUE)
```

## Arguments

| | |
|---|---|
| `x` | Character vector containing chemical formulas. |
| `expr` | Will return vector of characters when FALSE and expressions when TRUE. |

## Value

A vector of expressions.

## Examples

```
texts <- c(
  "C6H12Cl6",         # simple formula
  "H3Na",             # simple formula
  "Text with blank",  # simple text
  "A1B2",             # No valid elements but looks like formula
  "Fe(NO3)3",         # group with index
  "SO4^2-",           # charged molecule
  "Na+", "Cl-",       # simple ions
  paste0("H", intToUtf8(0x2082), "O"), # unicode-subscript
  "Ca(OH)2-"          # group with index and charge
)

# Check that all examples can be converted to expression
all(sapply(texts , function(x) { !inherits(try(fml2expr(x), silent=TRUE), "try-error") }))
exprs <- fml2expr(texts)
str(exprs)

# Plot with legend
plot(1:10, 1:10, pch = 16, col = 1:10, xlim = c(0, 11), ylim = c(0, 11))
legend("topright", legend = exprs, col = 1:10, pch = 16, title = "Formeln")

# Careful! text() does not accept an expression vector
text(3:4,3:4,labels=exprs[3:4])
for (i in 5:6) text(i,i,labels=exprs[[i]])

# you can also return a named character vector (names are input, values are modified)
fml2expr(texts, expr = FALSE)
```

---

`GenerateMetaboliteSQLiteDB` *GenerateMetaboliteSQLiteDB.*

---

## Description

`GenerateMetaboliteSQLiteDB` will set up a SQLite data base containing potential metabolite formulas, their masses and isotopic distribution for use with InterpretMSSpectrum.

## Usage

```
GenerateMetaboliteSQLiteDB(
  dbfile = "SQLite_APCI.db",
  ionization = c("APCI", "ESI")[1],
  mass_range = c(100, 105),
  ncores = 1,
  silent = TRUE
)
```

### Arguments

| | |
|---|---|
| dbfile | Path and file name of the final SQLiteDB or NULL to return the data frame. |
| ionization | Has to be specified to account for different plausibility rules and elemental composition. |
| mass_range | For testing use default range, otherwise use your measurement range. |
| ncores | Number of cores. Use as many as possible. |
| silent | Set to FALSE to get progress messages. |

### Details

The process takes a long time for larger masses (>400 Da). Parallel processing with 8 cores is highly recommended. Alternatively pre-processed versions can be downloaded on request to <jan.lisec@bam.de>. To process a 1 Da range (from 900 to 901) for ESI does take approximately 5 minutes on 8 cores.

### Value

Returns the resulting data frame invisible. Will write an SQL_DB if 'dbfile' provides a valid path and file name.

### Examples

```
# using the default values will compute be relatively fast, but for higher masses it
# is getting much slower
db <- GenerateMetaboliteSQLiteDB(dbfile = NULL)
```

---

## get_exactmass *get_exactmass.*

---

### Description

Get the exact mass for chemical formulas.

### Usage

```
get_exactmass(x)
```

### Arguments

| | |
|---|---|
| x | Vector of chemical formulas. |

### Value

A named vector of exact masses.

## Examples

```
InterpretMSSpectrum::get_exactmass(c("C6H12O6", "Na", "H1"))
```

---

`GetGroupFactor` *GetGroupFactor.*

---

## Description

`GetGroupFactor` will split a numeric vector according to a specified gap value. This is often a useful tool and therefore exported to the namespace.

## Usage

```
GetGroupFactor(x, gap)
```

## Arguments

| | |
|---|---|
| x | Numeric vector. |
| gap | Difference between two consecutive values at which a split is generated. |

## Value

A factor vector of length(x) indicating the different groups in x.

## Examples

```
x <- c(1:3,14:12,6:9)
GetGroupFactor(x=x, gap=2)
split(x, GetGroupFactor(x=x, gap=2))
```

---

`IMS_parallel` *IMS_parallel.*

---

## Description

`IMS_parallel` is a parallel implementation of `InterpretMSSpectrum`.

## Usage

```
IMS_parallel(
  spectra = NULL,
  ncores = 8,
  precursor = NULL,
  correct_peak = NULL,
  ...
)
```

### Arguments

| | |
|---|---|
| spectra | List of spectra. |
| ncores | Number of cores available. |
| precursor | vector of precursor masses of length(spectra). |
| correct_peak | Potentially a vector of correct Peaks, see InterpretMSSpectrum for details. |
| ... | Further parameters passed directly to InterpretMSSpectrum. |

### Details

For mass processing and testing it may be sufficient to use InterpretMSSpectrum without plotting functionality. However, function is likely to be deprecated or integrated as an option into the main function in the future.

### Value

A list of InterpretMSSpectrum result objects which can be systematically evaluated. However, note that plotting is unfortunately not enabled for parallel processing.

### See Also

InterpretMSSpectrum

---

## InterpretMSSpectrum *Interpreting High-Res-MS spectra.*

---

### Description

InterpretMSSpectrum will read, evaluate and plot a deconvoluted mass spectrum (mass*intensity pairs) from either TMS-derivatized GC-APCI-MS data or ESI+/- data. The main purpose is to identify the causal metabolite or more precisely the sum formula of the molecular peak by annotating and interpreting all visible fragments and isotopes.

### Usage

```
InterpretMSSpectrum(
  spec = NULL,
  precursor = NULL,
  correct_peak = NULL,
  met_db = NULL,
  typical_losses_definition = NULL,
  silent = FALSE,
  dppm = 3,
  param = "APCIpos",
  formula_db = NULL
)
```

## Arguments

| | |
|---|---|
| spec | A 2-column matrix of mz/int pairs. If spec=NULL then InterpretMSSpectrum tries to read data from clipboard (i.e. two columns copied from an Excel spreadsheet). |
| precursor | The ion (m/z) from spec closest to this mass will be considered as precursor (can be nominal, i.e. if precursor=364 then 364.1234 would be selected from spectrum if it is closest). |
| correct_peak | For testing purposes. A character in the form of "name, formula, mz" to evaluate spectra against. Note! Separating character is ', '. |
| met_db | A metabolite DB (e.g. GMD or internal) can be provided to search for candidates comparing M+H ions (cf. Examples). |
| typical_losses_definition | A file name (e.g. D:/BuildingBlocks_GCAPCI.txt) from where to load relevant neutral losses (cf. Details). Alternatively an data frame with columns 'Name', 'Formula' and 'Mass'. |
| silent | Logical. If TRUE no plot is generated and no output except final candidate list is returned. |
| dppm | Specifies ppm error for Rdisop formula calculation. |
| param | Either a list of parameters or a character shortcut indicating a predefined set. Currently 'APCIpos', 'ESIpos' and 'ESIneg' are supported as shortcuts. If a list is provided, list elements will overwrite similar named entries from the list that can be accessed using utils::data(param.default, package = "InterpretMSSpectrum"). |
| formula_db | A pre calculated database of sum formulas and their isotopic fine structures can be used to extremely speed up the function. |

## Details

For further details refer to and if using please cite Jaeger et al. (2016, <doi:10.1021/acs.analchem.6b02743>) in case of GC-APCI and Jaeger et al. (2017, <doi:10.1002/rcm.7905>) for ESI data. The Interpretation is extremely speed up if 'formula_db' (a predetermined database of potential sum formulas) is provided within the function call. Within the package you may use GenerateMetaboliteSQLiteDB to prepare one for yourself or request a download link from <jan.lisec@bam.de> as de-novo calculation for a wide mass range may take several days.

## Value

An annotated plot of the mass spectrum and detailed information within the console. Main result, list of final candidate formulas and their putative fragments, will be returned invisibly.

## Examples

```
# load APCI test data
apci_spectrum <- InterpretMSSpectrum::apci_spectrum

# (otional) provide information of a correct peak as a character containing
# name, formula and ion mass -- separated by ',' as shown below
```

```
cp <- "Glutamic acid (3TMS), C14H33NO4Si3, 364.1790"

# (otional) provide a database of known peaks
mdb <- data.frame(
  "Name" = c("Glutamic acid (3TMS)", "other peak with same sum formula"),
  "Formula" = c("C14H33NO4Si3", "C14H33NO4Si3"),
  "M+H" = c(364.179, 364.179), stringsAsFactors = FALSE, check.names = FALSE
)

# (otional) provide a database of precalculated formulas to speed up the process
fdb <- system.file("extdata", "APCI_min.db", package = "InterpretMSSpectrum")

# apply function providing above arguments which will print to the console
# and open a new plot
InterpretMSSpectrum(spec = apci_spectrum, correct_peak = cp, met_db = mdb, formula_db = fdb)

## Not run:
s <- structure(
list(
 mz = #'c(112.98609, 197.963, 226.97786, 520.90757, 560.95715, 568.95507, 593.95389),
 int = c(100, 100, 100, 100, 100, 100, 100)
), class = "data.frame", row.names = c(NA, -7L)
)
IMSSparam <- InterpretMSSpectrum::param.default
IMSSparam$ionmode <- "negative"
IMSSparam$allowed_elements <- c("C", "H", "N", "O", "P", "S", "F")
#IMSSparam$minElements <- "C1F1"
IMSSparam$maxElements <- "P1S2"
IMSSparam["substitutions"] <- list(NULL)
IMSSparam["ruleset"] <- "ESI"
InterpretMSSpectrum(spec = s, param = IMSSparam, precursor = spec[nrow(spec),1])

## End(Not run)
```

---

`InterpretTP` *InterpretTP.*

---

## Description

`InterpretTP` is a wrapper function around InterpretMSSpectrum which will read, evaluate and plot a deconvoluted mass spectrum (mass*intensity pairs) from either TMS-derivatized GC-APCI-MS data or ESI+/- data. It allows to provide a chemical formula as a potential precursor of the spectrum. This formula will be used to set the parameters 'allowed_elements' and 'maxElements' during de-novo formula generation.

## Usage

```
InterpretTP(fml = NULL, param = "APCIpos", ...)
```

## Arguments

| | |
|---|---|
| `fml` | A chemical formula of the standard used for transformation product generation. |
| `param` | Keyword or parameter list, similar as in InterpretMSSpectrum. |
| `...` | Arguments passed on to `InterpretMSSpectrum` |

`spec` A 2-column matrix of mz/int pairs. If spec=NULL then `InterpretMSSpectrum` tries to read data from clipboard (i.e. two columns copied from an Excel spreadsheet).

`precursor` The ion (m/z) from spec closest to this mass will be considered as precursor (can be nominal, i.e. if precursor=364 then 364.1234 would be selected from spectrum if it is closest).

`correct_peak` For testing purposes. A character in the form of "name, formula, mz" to evaluate spectra against. Note! Separating character is ', '.

`met_db` A metabolite DB (e.g. GMD or internal) can be provided to search for candidates comparing M+H ions (cf. Examples).

`typical_losses_definition` A file name (e.g. D:/BuildingBlocks_GCAPCI.txt) from where to load relevant neutral losses (cf. Details). Alternatively an data frame with columns 'Name', 'Formula' and 'Mass'.

`silent` Logical. If TRUE no plot is generated and no output except final candidate list is returned.

`dppm` Specifies ppm error for Rdisop formula calculation.

`formula_db` A pre calculated database of sum formulas and their isotopic fine structures can be used to extremely speed up the function.

## Details

For further details refer to InterpretMSSpectrum.

## Value

An annotated plot of the mass spectrum and detailed information within the console. Main result, list of final candidate formulas and their putative fragments, will be returned invisibly.

## Examples

```
# load test data
utils::data(apci_spectrum)

# provide information of a correct peak (if you know) as character
cp <- "Glutamic acid (3TMS), C14H33NO4Si3, 364.1790"

# provide database of known peaks and correct peak
mdb <- data.frame(
  "Name" = c("Glutamic acid (3TMS)", "other peak with same sum formula"),
  "Formula" = c("C14H33NO4Si3", "C14H33NO4Si3"),
  "M+H" = c(364.179, 364.179), stringsAsFactors = FALSE, check.names = FALSE
)

# provide a database of precalculated formulas to speed up the process
```

```
fdb <- system.file("extdata", "APCI_min.db", package = "InterpretMSSpectrum")

# apply function providing above arguments (dppm is set to 0.5 to reduce run time)
InterpretTP(fml = "C14H33NO4Si3", spec=apci_spectrum, param="APCIpos")
```

---

`mScore` *mScore.*

---

## Description

`mScore` will calculate a mass defect weighted score for an mz/int values measure for an isotopic cluster in comparison to the theoretically expected pattern.

## Usage

```
mScore(
  obs = NULL,
  the = NULL,
  dabs = 5e-04,
  dppm = 2,
  int_prec = 0.02,
  limit = 0,
  rnd_prec = 0
)
```

## Arguments

| | |
|---|---|
| `obs` | Observed (measured) values, a matrix with two rows (mz/int). |
| `the` | Theoretical (estimated from sum formula) values, a matrix with two rows (mz/int). |
| `dabs` | Absolute allowed mass deviation (the expected mass precision will influence mScore – see Details). |
| `dppm` | Relative allowed mass deviation (the expected mass precision will influence mScore – see Details). |
| `int_prec` | The expected intensity precision will influence mScore (see Details). |
| `limit` | minimal value of mScore. Should be left on zero. |
| `rnd_prec` | Rounding precision of mScore. |

## Details

The maximum expected average mass error should be specified in ppm. A observed pattern deviating that much from the theoretical pattern would still receive a reasonable (average) mScore while observations deviating stronger or less strong will reach lower or higher mScores respectively. Likewise the intensity precision should specify the average quality of your device to maintain stable isotopic ratios.

## Value

Scalar mScore giving the quality of the observed data if theoretical data are true.

## Examples

```
# get theoretical isotopic pattern of Glucose
glc <- c(180.063388, 0.920845, 181.066845, 0.065214, 182.068041, 0.013043)
glc <- matrix(glc, nrow=2)
mScore(obs=glc, the=glc)
# modify pattern by maximum allowable error (2ppm mass error, 2% int error)
glc_theoretic <- glc
glc[1,] <- glc[1,]+2*glc[1,]/10^6
glc[2,1:2] <- c(-0.02,0.02)+glc[2,1:2]
mScore(obs=glc, the=glc_theoretic)

# simulate mass and int defects
ef <- function(x, e) {runif(1,x-x*e,x+x*e)}
glc_obs <- glc
glc_obs[1,] <- sapply(glc[1,], ef, e=2*10^-6)
glc_obs[2,] <- sapply(glc[2,], ef, e=0.02)
mScore(obs=glc_obs, the=glc)
# simulate mass and int defects systematically
ef <- function(x, e) {runif(1,x-x*e,x+x*e)}
n <- 11
mz_err <- round(seq(0,5,length.out=n),3)
int_err <- round(seq(0,0.1,length.out=n),3)
mat <- matrix(NA, ncol=n, nrow=n, dimnames=list(mz_err, 100*int_err))
glc_obs <- glc
for (i in 1:n) {
 glc_obs[1,] <- sapply(glc[1,], ef, e=mz_err[i]*10^-6)
 for (j in 1:n) {
   glc_obs[2,] <- sapply(glc[2,], ef, e=int_err[j])
   mat[i,j] <- mScore(obs=glc_obs, the=glc)
 }
}
plot(x=1:n, y=1:n, type="n",axes=FALSE, xlab="mass error [ppm]", ylab="isoratio error [%]")
axis(3,at=1:n,rownames(mat),las=2); axis(4,at=1:n,colnames(mat),las=2); box()
cols <- grDevices::colorRampPalette(colors=c(2,6,3))(diff(range(mat))+1)
cols <- cols[mat-min(mat)+1]
text(x=rep(1:n,each=n), y=rep(1:n,times=n), labels=as.vector(mat), col=cols)
```

---

# neutral_losses_APCI *A data table defining typical neutral losses in GC-APCI-MS for silylated compounds.*

---

## Description

A data table defining typical neutral losses in GC-APCI-MS for silylated compounds.

### Usage

```
data(neutral_losses_ESI)
```

### Format

A data frame with 22 observations on the following 3 variables:

`Name` a character vector containing the fragment name used for plot annnotation

`Formula` a character vector containing chemical formulas

`Mass` a numeric vector containing the mass according to Formula

### Details

The data frame consists of two character columns ('Name' and 'Formula') and the numeric column 'Mass'. In a mass spectrum peak pairs are analyzed for mass differences similar to the ones defined in neutral_losses. If such a mass difference is observed, we can assume that the according 'Formula' is the true neutral loss observed in this spectrum. In a plot this peak pair would be connected by a grey line and annotated with the information from 'Name'. In formula evaluation this peak pair would be used to limit formula suggestions with respect to plausibility, i.e. if mass fragments A and B exist with mass difference 16.0313 than we can assume that the respective sum formulas have to be different by CH4. In consequence we can exclude sum formula suggestions for B which do not have a valid corresponding sum formula in A and vice versa.

### Source

This list has been put together manually by Jan Lisec analyzing multiple GC-APCI-MS data sets.

---

## `neutral_losses_ESI` *A data table defining neutral losses in LC-ESI-MS (positive mode).*

---

### Description

A data table defining neutral losses in LC-ESI-MS (positive mode).

### Usage

```
data(neutral_losses_ESI)
```

### Format

A data frame with 45 observations on the following 3 variables:

`Name` a character vector containing the fragment name used for plot annnotation

`Formula` a character vector containing chemical formulas

`Mass` a numeric vector containing the mass according to Formula

## Details

The data frame consists of two character columns ('Name' and 'Formula') and the numeric column 'Mass'. In a mass spectrum peak pairs are analyzed for mass differences similar to the ones defined in neutral_losses. If such a mass difference is observed, we can assume that the according 'Formula' is the true neutral loss observed in this spectrum. In a plot this peak pair would be connected by a grey line and annotated with the information from 'Name'. In formula evaluation this peak pair would be used to limit formula suggestions with respect to plausibility, i.e. if mass fragments A and B exist with mass difference 16.0313 than we can assume that the respective sum formulas have to be different by CH4. In consequence we can exclude sum formula suggestions for B which do not have a valid corresponding sum formula in A and vice versa.

## Source

This list has been put together manually by Jan Lisec analyzing multiple LC-ESI-MS (positive mode) data sets.

---

# OrbiMS1 *Orbitrap spectra*

---

## Description

A set of 550 MS1 pseudo-spectra of metabolite standards, acquired on an Orbitrap-type mass analyzer (Q Exactive, Thermo-Fisher) in electrospray ionization (ESI) positive mode. Spectra were generated from Thermo raw files using xcms/CAMERA.

## Usage

```
data(OrbiMS1)
```

## Format

A list with 550 matrices (spectra). Two attributes are attached to each spectrum:

**Formula** sum formula of (neutral) compound

**ExactMass** exact mass of (neutral) compound

param.default *Default parameter list for* InterpretMSSpectrum.

## Description

Default parameter list for InterpretMSSpectrum.

## Usage

```
data(param.default)
```

## Format

A data frame with 22 observations on the following 3 variables:

ionization ESI or APCI – will influence expected peak width and precision as well as adducts.

ionmode positive or negative – will influence expected adducts.

allowed_elements Passed to Rdisop in formula generation.

maxElements Passed to Rdisop in formula generation.

minElements Passed to Rdisop in formula generation.

substitutions Will be deprecated in the future.

quick_isos TRUE = via Rdisop, FALSE = via enviPat (often more correct)

score_cutoff Specifies initial filtering step threshold per fragment. Sum Formulas with score_i < score_cutoff*max(score) will be removed.

neutral_loss_cutoff Specifies the allowed deviation in mDa for neutral losses to be accepted from the provided neutral loss list.

## Details

Default parameter list used by InterpretMSSpectrum, serving also as a template for custom lists. Basically every option which needs to be modified rarely went in here. Specific parameter set modifications (i.e. for 'APCIpos') are provided and can be called using the character string as a shortcut. Alternatively, a named list can be provided where all contained parameters will receive the new specified values.

PlotSpec *Plot Mass Spectrum.*

## Description

PlotSpec will read, evaluate and plot a deconvoluted mass spectrum (mass*intensity pairs) from TMS-derivatized GC-APCI-MS data. The main purpose is to visualize the relation between deconvoluted masses.

## Usage

```
PlotSpec(
  x = NULL,
  masslab = 0.1,
  rellab = FALSE,
  cutoff = 0.01,
  cols = NULL,
  txt = NULL,
  mz_prec = 4,
  ionization = NULL,
  neutral_losses = NULL,
  neutral_loss_cutoff = NULL,
  substitutions = NULL,
  precursor = NULL,
  xlim = NULL,
  ylim = NULL,
  mar = c(2, 2, 0.5, 0) + 0.5
)
```

## Arguments

| | |
|---|---|
| x | A two-column matrix with ("mz", "int") information. |
| masslab | The cutoff value (relative to basepeak) for text annotation of peaks. |
| rellab | TRUE/FALSE. Label masses relative to largest mass in plot (if TRUE), absolute (if FALSE) or to specified mass (if numeric). |
| cutoff | Show only peaks with intensity higher than cutoff*I(base peak). This will limit the x-axis accordingly. |
| cols | Color vector for peaks with length(cols)==nrow(x). |
| txt | Label peaks with specified text (column 1 specifies x-axis value, column 2 specifies label). Labels will be converted to expressions following the ruleset to format chemical formulas (see Examples). |
| mz_prec | Numeric precision of m/z (=number of digits to plot). |
| ionization | Either APCI or ESI (important for main peak determination). |
| neutral_losses | Data frame of defined building blocks (Name, Formula, Mass). If not provided data("neutral_losses") will be used. |

neutral_loss_cutoff
: Specifies the allowed deviation in mDa for neutral losses to be accepted from the provided neutral loss list.

substitutions
: May provide a two column table of potential substitutions (for adducts in ESI-MS).

precursor
: Internally main peaks will be determined up to a supposed precursor obtained by 'DetermineIsomainPeaks' and annotations will only be plotted up to this mass. To plot annotations for the full mass range, set 'precursor' to a higher mass.

xlim
: To specify xlim explicitly (for comparative plotting).

ylim
: To specify ylim explicitly (for comparative plotting).

mar
: Set to NULL to allow specifying margins outside the function (important for use in Shiny App context with brush event).

## Value

An annotated plot of the mass spectrum.

## Examples

```
# load APCI test data and apply function
utils::data(apci_spectrum, package = "InterpretMSSpectrum")
PlotSpec(x=apci_spectrum, ionization="APCI")

# normalize test data by intensity
s <- apci_spectrum
s[,2] <- s[,2]/max(s[,2])
PlotSpec(x=s)

# use relative labelling
PlotSpec(x=s, rellab=s[which.max(s[,2]),1])

# avoid annotation of masses and fragments
PlotSpec(x=s, masslab=NULL, neutral_losses=NA)

# provide individual neutral loss set
tmp <- data.frame("Name"=c("Loss1","C2H6"),"Formula"=c("",""),"Mass"=c(90.05,27.995))
PlotSpec(x=s, neutral_losses=tmp)

# provide additional color and annotation information per peak
PlotSpec(x=s, cols=1+(s[,2]>0.1), txt=data.frame("x"=s[s[,2]>0.1,1],"txt"="txt"))

# annotate a sum formula
PlotSpec(x=s, txt=data.frame("x"=s[which.max(s[,2]),1],"txt"="C6H12O6"))

# or annotate a mix of sum formula and text
txt <- data.frame(
 "x" = s[order(s[,2],decreasing=TRUE)[1:3],1],
 "txt" = c("C6H12O6", "some text", "H2SO3")
)
PlotSpec(x=s, txt=txt)
```

```
# if 'txt' contains a column SMILES with valid SMILES codes, they will be rendered additionally
txt[,"SMILES"] <- c("O=CC(O)C(O)C(O)C(O)CO", "", "OS(=O)(=O)")
PlotSpec(x=s, txt=txt)
txt[,"w"] <- rep(0.4,3)
PlotSpec(x=s, txt=txt)

# simulate a Sodium adduct to the spectrum (and annotate using substitutions)
p <- which.max(s[,2])
s <- rbind(s, c(21.98194+s[p,1], 0.6*s[p,2]))
PlotSpec(x=s, substitutions=matrix(c("H","Na"),ncol=2,byrow=TRUE))

# add structural formulas by providing SMILES codes

# use predefined ESI parameter set
PlotSpec(x=s, ionization="ESI")
```

---

ReadSpecClipboard *ReadSpecClipboard.*

---

## Description

Read a mass spectrum from the windows clipboard.

## Usage

```
ReadSpecClipboard(con = "clipboard")
```

## Arguments

| | |
|---|---|
| con | A connection other than 'clipboard' can be provided. |

## Value

A spectrum as two-column matrix.

## Examples

```
## Not run:
  if (length(grep("Windows", utils::sessionInfo()$running))==1) {
    x <- InterpretMSSpectrum::apci_spectrum
    write.table(x, "clipboard", sep="\t", row.names=FALSE)
    InterpretMSSpectrum::ReadSpecClipboard()
  }

## End(Not run)
```

sendToMSF *Exporting spectra to MSFinder.*

## Description

Send spectrum to MSFinder.

## Usage

```
sendToMSF(x, ...)

## Default S3 method:
sendToMSF(
  x,
  precursormz,
  precursortype = "[M+H]+",
  outfile = NULL,
  MSFexe = NULL,
  ...
)

## S3 method for class 'findMAIN'
sendToMSF(x, rank = 1, ms2spec = NULL, outfile = NULL, MSFexe = NULL, ...)
```

## Arguments

| | |
|---|---|
| x | A matrix or 'findMAIN' object |
| ... | Arguments passed to methods of writeMSF. |
| precursormz | m/z of (de)protonated molecule or adduct ion |
| precursortype | adduct type, e.g. [M+H]+ or [M+Na]+. Accepted values are all adduct ions supported by MSFINDER. |
| outfile | Name of MAT file. If NULL, a temporary file is created in the per-session temporary directory (see tempdir). |
| MSFexe | Full path of MS-FINDER executable. This needs to be set according to your system. If NULL, MAT files are written but the program is not opened. |
| rank | Which rank from 'findMAIN' should be exported. |
| ms2spec | An (optional) MS2 spectrum to be passed to MSFINDER. If NULL, the MS1 spectrum used by 'findMAIN' is used. If dedicated MS2 spectra are available, this option should be used. |

## Details

In the default case 'x' can be a matrix or data frame, where the first two columns are assumed to contain the 'mz' and 'intensity' values, respectively. Further arguments 'precursormz' and 'precursortype' are required in this case. Otherwise 'x' can be of class findMAIN.

## Value

Full path of generated MAT file (invisibly).

## References

H.Tsugawa et al (2016) Hydrogen rearrangement rules: computational MS/MS fragmentation and structure elucidation using MS-FINDER software. Analytical Chemistry, 88, 7946-7958

## Examples

```
## Not run:
utils::data(esi_spectrum, package = "InterpretMSSpectrum")
fmr <- findMAIN(esi_spectrum)
sendToMSF(fmr, outfile="tmp.mat")
sendToMSF(fmr, outfile="tmp.mat", rank=1:3)

## End(Not run)
```

# Index


∗ **datasets**
  Adducts, 2
  apci_spectrum, 3
  chemical_elements, 3
  esi_spectrum, 4
  neutral_losses_APCI, 16
  neutral_losses_ESI, 17
  OrbiMS1, 18
  param.default, 19

Adducts, 2, *5*
apci_spectrum, 3

chemical_elements, 3
CountChemicalElements, 4

esi_spectrum, 4

findMAIN, 5
fml2expr, 7

GenerateMetaboliteSQLiteDB, 8, *12*
get_exactmass, 9
GetGroupFactor, 10

IMS_parallel, 10
InterpretMSSpectrum, *8*, *10*, 11, *11*, *13, 14*, *19*
InterpretTP, 13

mScore, 15

neutral_losses_APCI, 16
neutral_losses_ESI, 17

OrbiMS1, 18

param.default, 19
plot.findMAIN *(*findMAIN*)*, 5
PlotSpec, 20
print.findMAIN *(*findMAIN*)*, 5

ReadSpecClipboard, 22

sendToMSF, 23

tempdir, *23*

writeMSF, *23*