Package 'MALDIcellassay'

August 29, 2024

Type Package

Title Automated MALDI Cell Assays Using Dose-Response Curve Fitting

Version 0.4.47

Description Conduct automated cell-based assays using Matrix-Assisted Laser Desorption/Ionization (MALDI) methods for high-throughput screening of signals responsive to treatments. The package efficiently identifies high variance signals and fits dose-response curves to them. Quality metrics such as Z', V', log2FC, and CRS are provided for evaluating the potential of signals as biomarkers. The methodologies were introduced by Weigt et al. (2018) <doi:10.1038/s41598-018-29677-z> and refined by Unger et al. (2021) <doi:10.1038/s41596-021-00624-z>.

License MIT + file LICENSE

Imports methods, ggplot2, nplr, dplyr, tidyr, forcats, scales, MALDIquant, MALDIquantForeign, tibble, svMisc, purrr

Encoding UTF-8

LazyData true

LazyDataCompression xz

RoxygenNote 7.3.2

Suggests rmarkdown, knitr

VignetteBuilder knitr

Depends R (>= 4.2)

URL https://github.com/CeMOS-Mannheim/MALDIcellassay

BugReports https://github.com/CeMOS-Mannheim/MALDIcellassay/issues

NeedsCompilation no

Author Thomas Enzlein [aut, cre, cph] (<https://orcid.org/0000-0003-1789-4090>)

Maintainer Thomas Enzlein <t.enzlein@hs-mannheim.de>

Repository CRAN

Date/Publication 2024-08-29 19:10:09 UTC

Contents

`Blank2022intmat` *Blank2022intmat*

Description

Intensity matrix from MALDI mass spectrometry data of EOC cells treated with different concentrations of SAHA. It is used to demonstrate the usage of `MALDIcellassay`.

Usage

```
data(Blank2022intmat)
```

Format

Matrix with concentrations of original spectra as rownames and m/z-values as colnames.

Details

The concentrations include: 0, 0.04, 0.12, 0.37, 1.11, 3.33, 10 and 30 uM of SAHA at 4 replicates each. The original spectra were trimmed to 400-900 Da mass-range to keep the file size small. The peaks are the result of `MALDIquant::intensityMatrix(Blank2022peaks, Blank2022spec)`

References

Blank, M., Enzlein, T. & Hopf, C. LPS-induced lipid alterations in microglia revealed by MALDI mass spectrometry-based cell fingerprinting in neuroinflammation studies. Sci Rep 12, 2908 (2022). https://doi.org/10.1038/s41598-022-06894-1

`Blank2022peaks` *Blank2022peaks*

Description

Peaks from MALDI mass spectrometry data of EOC cells treated with different concentrations of SAHA. It is used to demonstrate the usage of `MALDIcellassay`.

Usage

```
data(Blank2022peaks)
```

Format

A list of MALDIquant::MassPeaks-objects named with the respective concentration.

Details

The concentrations include: 0, 0.04, 0.12, 0.37, 1.11, 3.33, 10 and 30 uM of SAHA at 4 replicates each. The original spectra were trimmed to 400-900 Da mass-range to keep the file size small. The peaks are the result of applying `MALDIquant::detectPeaks` to `Blank2022spec` with arguments `SNR = 3, method = "SuperSmoother"`.

References

Blank, M., Enzlein, T. & Hopf, C. LPS-induced lipid alterations in microglia revealed by MALDI mass spectrometry-based cell fingerprinting in neuroinflammation studies. Sci Rep 12, 2908 (2022). https://doi.org/10.1038/s41598-022-06894-1

`Blank2022res` *Blank2022res*

Description

Object of class MALDIcellassay from MALDI mass spectrometry data of EOC cells treated with different concentrations of SAHA. It is used to demonstrate the usage of `MALDIcellassay`.

Usage

```
data(Blank2022res)
```

Format

Matrix with concentrations of original spectra as rownames and m/z-values as colnames.

Details

The concentrations include: 0, 0.04, 0.12, 0.37, 1.11, 3.33, 10 and 30 uM of SAHA at 4 replicates each. The original spectra were trimmed to 400-900 Da mass-range to keep the file size small. The peaks are the result of `fitCurve(spec = Blank2022spec, SinglePointRecal = TRUE, normMz = 760.585, alignTol = 0.1, normTol = 0.1)`

References

Blank, M., Enzlein, T. & Hopf, C. LPS-induced lipid alterations in microglia revealed by MALDI mass spectrometry-based cell fingerprinting in neuroinflammation studies. Sci Rep 12, 2908 (2022). https://doi.org/10.1038/s41598-022-06894-1

Blank2022spec *Blank2022spec*

Description

MALDI mass spectrometry data of EOC cells treated with different concentrations of SAHA. It is used to demonstrate the usage of `MALDIcellassay`.

Usage

```
data(Blank2022spec)
```

Format

A list of MALDIquant::MassSpectrum-objects named with the respective concentration.

Details

The concentrations include: 0, 0.04, 0.12, 0.37, 1.11, 3.33, 10 and 30 uM of SAHA at 4 replicates each. The original spectra were trimmed to 400-900 Da mass-range to keep the file size small.

References

Blank, M., Enzlein, T. & Hopf, C. LPS-induced lipid alterations in microglia revealed by MALDI mass spectrometry-based cell fingerprinting in neuroinflammation studies. Sci Rep 12, 2908 (2022). https://doi.org/10.1038/s41598-022-06894-1

calculateChauvenetCriterion *Calculate Chauvenet's criterion for outlier detection*

Description

Calculate Chauvenet's criterion for outlier detection

Usage

```
calculateChauvenetCriterion(x)
```

Arguments

`x`	numeric, values (e.g. intensities) to test for outliers

Details

Note that, as for all outlier detection criteria: Excluding data points from your measurement should only be conducted with extreme care. Even if this (or any other) function tells you that a data point is an outlier, you might still want to have it in your sample population especially if you are not sure if your data is normal distributed. See Wikipedia for details of the algorithm.

Value

logical vector, TRUE for detected outliers.

Examples

```
set.seed(42)

#no outlier
sample <- rnorm(n = 8, mean = 0, sd = 0.01)
calculateChauvenetCriterion(sample)

# introduce outlier
sample[1] <- 1
calculateChauvenetCriterion(sample)
```

`calculateCurveFit` *Calculate the fit for a dose-response curve*

Description

Calculate the fit for a dose-response curve

Usage

```
calculateCurveFit(intmat, idx, verbose = TRUE, ...)
```

Arguments

`intmat`	Intensity matrix as generated by `MALDIquant::intensityMatrix()` with rownames as the respective concentrations of the spectra.
`idx`	Numeric vector of the mz indices to perform the fit.
`verbose`	Logical, print logs to console.
`...`	Additional arguments passed to `nplr::nplr()`.

Value

List of curve fits.

Examples

```
data(Blank2022intmat)

# for faster runtime we let it run on 5 peaks only
fits <- calculateCurveFit(Blank2022intmat, idx = 1:5)
```

`calculatePeakStatistics` *Calculate peak statistics*

Description

Calculate peak statistics

Usage

```
calculatePeakStatistics(curveFits, singlePeaks, spec)
```

Arguments

`curveFits`	list of curve fits as returned by `MALDIcellassay::calculateCurveFit()`.
`singlePeaks`	list of MALDIquant::MassPeaks.
`spec`	list of MALDIquant::MassSpectrum.

Value

A tibble with peak statistics.

Examples

```
data(Blank2022intmat)
fits <- calculateCurveFit(Blank2022intmat, idx = 1:5)

peakstats <- calculatePeakStatistics(curveFits = fits,
                                     singlePeaks = Blank2022peaks,
                                     spec = Blank2022spec)
head(peakstats)
```

`calculateSSMD` *Calculate strictly standardized mean difference (SSMD)*

Description

Calculate strictly standardized mean difference (SSMD)

Usage

```
calculateSSMD(res, internal = TRUE, nConc = 2)
```

Arguments

`res`	Object of class MALDIassay
`internal`	Logical, currently only the internal implementation, using `nConc` top and bottom concentrations, is implemented.
`nConc`	Numeric, number of top and bottom concentrations to be used to calculate the pseudo positive and negative control. Only used if `internal` is TRUE

Details

The strictly standardized mean difference (SSMD) is a measure of effect size. It is the mean divided by the standard deviation of a difference between the positive and negative control.

$$\gamma = \frac{\mid \mu_n - \mu_p \mid}{\sqrt{\sigma_n^2 + \sigma_p^2}}$$

The SSMD can be easily be interpreted as it denotes the difference between positve and negative controls in units of standard deviation.

Value

Numeric vector of strictly standardized mean differences (SSMD)

Examples

```
# see example for `fitCurve()` to see how this data was generated
data(Blank2022res)

calculateSSMD(Blank2022res, nConc = 2)
```

`calculateVPrime` *Calculate V'-Factor*

Description

Calculate V'-Factor

Usage

```
calculateVPrime(res, internal = TRUE)
```

Arguments

`res`	Object of class MALDIassay
`internal`	Logical, currently only the internal implementation, using `nConc` top and bottom concentrations, is implemented.

Details

The V'-factor is a generalization of the Z'-factor to a dose-response curve. See M.-A. Bray and A. Carpenter, Advanced assay development guidelines for image-based high content screening and analysis for details. It is defined as:

$$V' = 1 - 6 * \sigma_f / |\mu_p - \mu_n|$$

with

$$\sigma_f = \sqrt{1/N * \sum y_f it - y_m easured^2}$$

In other words, σ_f is the standard deviation of residuals.

Note, we do not need to estimate the variance for the mean of the positive and negative value. So, this function uses the top and bottom asymptote directly instead of taking the top and bottom concentrations in consideration.

Value

Numeric vector of V'-factors

Examples

```
# see example for `fitCurve()` to see how this data was generated
data(Blank2022res)

calculateVPrime(Blank2022res)
```

`calculateZPrime` *Calculate Z'-factor of assay quality*

Description

Calculate Z'-factor of assay quality

Usage

```
calculateZPrime(res, internal = TRUE, nConc = 2)
```

Arguments

`res`	Object of class MALDIassay
`internal`	Logical, currently only the internal implementation, using `nConc` top and bottom concentrations, is implemented.
`nConc`	Numeric, number of top and bottom concentrations to be used to calculate the pseudo positive and negative control. Only used if `internal` is TRUE

Details

The most common way to measure the quality of an assay is the so-called Z'-factor, which describes the separation of the positive and negative control in terms of their standard deviations σ_p and σ_n. The Z'-factor is defined as Ji-Hu Zhang et al., A simple statistical parameter for use in evaluation and validation of high throughput screening assays.

$$Z' = 1 - (3 * (\sigma_p + \sigma_n))/|\mu_p - \mu_n|$$

where μ_p and μ_p is the mean value of the positive (response expected) and negative (no response expected) control, respectively. Therefore, the assay quality is **independent of the shape of the concentration response curve** and solely depend on two control values.

Note, if `internal` is set to TRUE, the `nConc` highest concentrations are assumed as positive control, whereas the `nConc` lowest concentrations are used as negative.

Value	**Interpretation**
Z' ~ 1	perfect assay
1 > Z' > 0.5	excellent assay
0.5 > Z' > 0	moderate assay
Z' = 0	good only for yes/no response
Z' < 0	unacceptable

Value

Numeric vector of Z'-factors.

Examples

```
# see example for `fitCurve()` to see how this data was generated
data(Blank2022res)
calculateZPrime(Blank2022res, nConc = 2)
```

`checkRecalibration` *Check the recalibration of spectra from a MALDIassay object*

Description

Dashed gray lines indicate the mz used for re-calibration ± the tolerance. Red dashed line indicate the mz used for re-calibration and solid lines indicate peaks. The spectrum will show the peak used for re-calibration ± 10x the tolerance.

Usage

```
checkRecalibration(object, idx)
```

Arguments

`object`	Object of class MALDIassay
`idx`	Numeric, index of spectrum to plot

Value

ggplot object

Examples

```
# see example for `fitCurve()` to see how this data was generated
data(Blank2022res)
checkRecalibration(Blank2022res, idx = 1:8)
```

`extractIntensity` *Extract intensity using peaks as template*

Description

Extract intensity using peaks as template

Usage

```
extractIntensity(mz, peaks, spec, tol)
```

Arguments

mz	numeric, mz values to be extracted from the peaks/spectra
peaks	MALDIquant::MassPeaks list
spec	MALDIquant::MassSpectrum list
tol	numeric, tolerance in Da

Value

MALDIquant::MassPeaks list with extracted intensities from spec at m/z of peaks = pseudo peaks. Useful in combination with sdMassSpectrum to get standard deviation of peaks as intensity matrix.

Examples

```
data(Blank2022peaks)
data(Blank2022spec)

int <- extractIntensity(mz = c(409, 423, 440),
                        peaks = Blank2022peaks,
                        spec = Blank2022spec,
                        tol = 0.2)
head(int)
```

---

extractSpots *Extract the spot coordinates*

Description

Extract the spot coordinates

Usage

```
extractSpots(spec)
```

Arguments

spec	list of MALDIquant::MassSpectrum or MALDIquant::MassPeaks objects

Value

Character vector of spot names. If multiple spots are used (e.g. for average spectra) they will be concatenate.

Examples

```
data(Blank2022spec)
head(extractSpots(Blank2022spec))
```

`filterVariance` *Filter for high variance signals*

Description

Filter for high variance signals

Usage

```
filterVariance(
  vars,
  method = c("mean", "median", "q25", "q75", "none"),
  verbose = TRUE
)
```

Arguments

`vars`	Numeric vector, variances of signals
`method`	Character, filtering method. One of "mean" (default), "median", "q25", "q75" (25 and 75% quantile) or "none".
`verbose`	Logical, print logs to console.

Value

Indices of spectra with a high variance

Examples

```
data(Blank2022intmat)
# get variance of each peak
vars <- apply(Blank2022intmat, 2, var)
highVarIndicies <- filterVariance(vars, method = "mean", verbose = TRUE)
```

`fitCurve` *Fit dose-response curves*

Description

Fit dose-response curves

Usage

```
fitCurve(
  spec,
  unit = c("M", "mM", "uM", "nM", "pM", "fM"),
  varFilterMethod = c("mean", "median", "q25", "q75", "none"),
  monoisotopicFilter = FALSE,
  averageMethod = c("mean", "median", "sum"),
  normMz = NULL,
  normTol = 0.1,
  alignTol = 0.01,
  binTol = 2e-04,
  SNR = 3,
  halfWindowSize = 3,
  allowNoMatches = TRUE,
  normMeth = c("mz", "TIC", "PQN", "median", "none"),
  SinglePointRecal = TRUE,
  verbose = TRUE
)
```

Arguments

spec	List of MALDIquant::MassSpectrum
unit	Character, unit of concentration. Used to calculate the concentration in Moles so that pIC50 is correct. Set to "M" if you dont want changes in your concentrations.
varFilterMethod	Character, function applied for high variance filtering. One of the following options mean (default), median, q25, q75 or none (no filtering).
monoisotopicFilter	Logical, filter peaks and just use monoisotopic peaks for curve fit.
averageMethod	Character, aggregation method for average mass spectra ("mean" or "median")
normMz	Numeric, mz used for normalization AND for single point recalibration.
normTol	Numeric, tolerance in Dalton to match normMz
alignTol	Numeric, tolerance for spectral alignment in Dalton.
binTol	Numeric, tolerance for binning of peaks.
SNR	Numeric, signal to noise ratio for peak detection.
halfWindowSize	2ction. See MALDIquant::detectPeaks().
allowNoMatches	Logical, if normMz can not be found in a spectrum, proceed and exclude spectrum or stop
normMeth	Character, normalization method. Can either be "TIC", "PQM", "median" or "mz". If "mz" then the normMz is used. If none no normalization is done.
SinglePointRecal	Logical, perform single point recalibration to normMz
verbose	Logical, print logs to console.

Value

Object of class `MALDIassay`. The most important slot is `fits` which contains the IC50 curve fits.

Examples

```
data(Blank2022spec)

fitCurve(spec = Blank2022spec,
         SinglePointRecal = TRUE,
         normMz = 760.585,
         alignTol = 0.1,
         normTol = 0.1,
         varFilterMethod = "mean")
```

`getAllMz` *Get all mz value of an MALDIassay-object*

Description

Get all mz value of an MALDIassay-object

Usage

```
getAllMz(object, excludeNormMz = FALSE)
```

Arguments

`object`	Object of class MALDIassay
`excludeNormMz`	Logical, remove normMz from list of mz values.

Value

numeric vector of mz values

Examples

```
# see example for `fitCurve()` to see how this data was generated
data(Blank2022res)
head(getAllMz(Blank2022res))
```

getAppliedMzShift *Extract applied mz-shift*

Description

Extract applied mz-shift

Usage

```
getAppliedMzShift(object)
```

Arguments

object	Object of class MALDIassay

Value

Numeric vector of mz-shits applied to spectra

Examples

```
# see example for `fitCurve()` to see how this data was generated
data(Blank2022res)
head(getAppliedMzShift(Blank2022res))
```

getAppliedNormFactors *Extract applied normalization factors*

Description

Extract applied normalization factors

Usage

```
getAppliedNormFactors(object)
```

Arguments

object	Object of class MALDIassay

Value

Numeric vector of normalization factors applied to spectra

Examples

```
# see example for `fitCurve()` to see how this data was generated
data(Blank2022res)
head(getAppliedNormFactors(Blank2022res))
```

getAvgPeaks *Extract peaks of average spectra*

Description

Extract peaks of average spectra

Usage

```
getAvgPeaks(object)
```

Arguments

object	Object of class MALDIassay

Value

List of MALDIquantMassPeaks

Examples

```
# see example for `fitCurve()` to see how this data was generated
data(Blank2022res)
getAvgPeaks(Blank2022res)[[1]]
```

getAvgSpectra *Extract average spectra*

Description

Extract average spectra

Usage

```
getAvgSpectra(object)
```

Arguments

object	Object of class MALDIassay

Value

List of MALDIquantMassSpectrum

Examples

```
# see example for `fitCurve()` to see how this data was generated
data(Blank2022res)
getAvgSpectra(Blank2022res)[[1]]
```

getBinTol *Get binning tolerance*

Description

Get binning tolerance

Usage

```
getBinTol(object)
```

Arguments

object	Object of class MALDIassay

Value

Numeric, tolerance used for binning in Dalton.

Examples

```
# see example for `fitCurve()` to see how this data was generated
data(Blank2022res)
getBinTol(Blank2022res)
```

getConc *Extract the concentrations used in a MALDIassay*

Description

Extract the concentrations used in a MALDIassay

Usage

```
getConc(object)
```

Arguments

object	Object of class MALDIassay

Value

Numeric vector, concentrations used in a MALDIassay

Examples

```
# see example for `fitCurve()` to see how this data was generated
data(Blank2022res)
head(getConc(Blank2022res))
```

getCurveFits *Extract curve fits*

Description

Extract curve fits

Usage

```
getCurveFits(object)
```

Arguments

object	Object of class MALDIassay

Value

List, containing the data used to do the fits as well as the nlpr curve fit .

Examples

```
# see example for `fitCurve()` to see how this data was generated
data(Blank2022res)
fits <- getCurveFits(Blank2022res)
```

getDirectory *Extract directory path*

Description

Extract directory path

Usage

```
getDirectory(object)
```

Arguments

object	Object of class MALDIassay

Value

List, containing the data used to do the fits as well as the nlpr curve fit .

Examples

```
# see example for `fitCurve()` to see how this data was generated
data(Blank2022res)
getDirectory(Blank2022res)
```

getFittingParameters *Get fitting parameters*

Description

Get fitting parameters

Usage

```
getFittingParameters(object, summarise = FALSE)
```

Arguments

object	Object of class MALDIassay
summarise	Logical, remove everything other then npar and mz from result.

Value

tibble of fitting parameters for each fitted m/z-value

Examples

```
# see example for `fitCurve()` to see how this data was generated
data(Blank2022res)
head(getFittingParameters(Blank2022res, summarise = FALSE))
```

getIntensityMatrix *Get the intensity matrix of single spectra for all fitted curves*

Description

Get the intensity matrix of single spectra for all fitted curves

Usage

```
getIntensityMatrix(object, avg = FALSE, excludeNormMz = FALSE)
```

Arguments

object	Object of class MALDIassay
avg	Logical, return single spectra intensity matrix (default) or average spectra intensity matrix
excludeNormMz	Logical, exclude normMz from intensity matrix.

Details

Note that the returned matrix only contains *m/z* values that were actually fitted. If a variance filtering step was applied this will not include **all** *m/z* values. If you wish to get a matrix of **all** *m/z* values use `MALDIquant::intensityMatrix(getSinglePeaks(object))`. For average spectra intensity matrix with **all** *m/z* values use `MALDIquant::intensityMatrix(getAvgPeaks(object), getAvgSpectra(object))`.

Value

A matrix with columns as *m/z* values and rows as concentrations/spectra

Examples

```
# see example for `fitCurve()` to see how this data was generated
data(Blank2022res)
head(getIntensityMatrix(Blank2022res, avg = TRUE, excludeNormMz = TRUE) )
```

`getMzFromMzIdx` *Get the mz value associated with a mzIdx*

Description

Get the mz value associated with a mzIdx

Usage

```
getMzFromMzIdx(object, mzIdx)
```

Arguments

`object`	Object of class MALDIassay
`mzIdx`	numeric, index of mass of interest (see `getPeakStatistics()`)

Value

numeric, mz value

Examples

```
# see example for `fitCurve()` to see how this data was generated
data(Blank2022res)
getMzFromMzIdx(Blank2022res, mzIdx = 2)
```

getMzShift *Get mass shift for target mz*

Description

Get mass shift for target mz

Usage

```
getMzShift(peaks, targetMz, tol, tolppm = FALSE, verbose = TRUE)
```

Arguments

peaks	List of MALDIquant::MassPeak
targetMz	Numeric, target mass
tol	Numeric, tolerance around targetMz
tolppm	Logical, tolerance supplied in ppm
verbose	Logical, print logs to the console.

Value

List with two entries: `MzShift` The mass shift for each spectrum `specIdx` The index of the spectra with a match for targetMz

Examples

```
data(Blank2022peaks)
getMzShift(Blank2022peaks, targetMz = 760.585, tol = 0.1, tolppm = FALSE)
```

getNormFactors *Get normalization factors from peak data.frame*

Description

Get normalization factors from peak data.frame

Usage

```
getNormFactors(peaksdf, targetMz, tol, tolppm = TRUE, allowNoMatch = TRUE)
```

Arguments

peaksdf	data.frame with peaks information as generated by peaks2df()
targetMz	Numeric, target mass
tol	Numeric, tolerance around targetMz
tolppm	Logical, is the tolerance provided in ppm (TRUE) or Daltion (FALSE)
allowNoMatch	Logical, stop if targetMz is not found in single spectrum? If TRUE spectra without targetMz match will be excluded.

Value

```
List with two entries:
                    norm_factor The normalization factor for each spectrum
                 specIdx    The index of the spectra with a match for targetMz
```

Examples

```
data(Blank2022peaks)
getNormFactors(peaks2df(Blank2022peaks), targetMz = 760.585, tol = 0.1, tolppm = FALSE)
```

---

getNormMethod *Extract normalization method*

Description

Extract normalization method

Usage

```
getNormMethod(object)
```

Arguments

object	Object of class MALDIassay

Value

Character, normalization method used.

Examples

```
# see example for `fitCurve()` to see how this data was generated
data(Blank2022res)
getNormMethod(Blank2022res)
```

getNormMz *Extract m/z used for normalization*

Description

Extract m/z used for normalization

Usage

```
getNormMz(object)
```

Arguments

object	Object of class MALDIassay

Value

Numeric, m/z used for normalization

Examples

```
# see example for `fitCurve()` to see how this data was generated
data(Blank2022res)
getNormMz(Blank2022res)
```

getNormMzTol *Extract tolerance used for normalization*

Description

Extract tolerance used for normalization

Usage

```
getNormMzTol(object)
```

Arguments

object	Object of class MALDIassay

Value

Numeric, tolerance used for normalization

Examples

```
# see example for `fitCurve()` to see how this data was generated
data(Blank2022res)
getNormMzTol(Blank2022res)
```

getPeakStatistics *Extract peak statistics*

Description

Extract peak statistics

Usage

```
getPeakStatistics(object, summarise = FALSE)
```

Arguments

object	Object of class MALDIassay
summarise	Logical, return summarized results (one result per mz and not per mz and spectra)

Value

A tibble with peak statistics (R^2, fold-change, CV%, etc.)

Examples

```
# see example for `fitCurve()` to see how this data was generated
data(Blank2022res)
head(getPeakStatistics(Blank2022res, summarise = TRUE))
```

getRecalibrationError *Calculate remaining calibration error of a MALDIassay object*

Description

Calculate remaining calibration error of a MALDIassay object

Usage

```
getRecalibrationError(object)
```

Arguments

object	Object of class MALDIassay

Value

A tibble containing statistics about remaining calibration error

Examples

```
# see example for `fitCurve()` to see how this data was generated
data(Blank2022res)
getRecalibrationError(Blank2022res)
```

`getSinglePeaks` *Extract peaks of single spectra spectra (before average calculation)*

Description

Extract peaks of single spectra spectra (before average calculation)

Usage

```
getSinglePeaks(object)
```

Arguments

`object`	Object of class MALDIassay

Value

List of MALDIquantMassPeaks

Examples

```
# see example for `fitCurve()` to see how this data was generated
data(Blank2022res)
getSinglePeaks(Blank2022res)[[1]]
```

`getSingleSpecIntensity` *Extract the intensities of single spectra for a given mzIdx*

Description

Extract the intensities of single spectra for a given mzIdx

Usage

```
getSingleSpecIntensity(object, mz_idx)
```

Arguments

`object`	Object of class MALDIassay
`mz_idx`	Integer, index of mz

Value

Numeric vector, intensities of mzIdx

Examples

```
# see example for `fitCurve()` to see how this data was generated
data(Blank2022res)
head(getSingleSpecIntensity(Blank2022res, 2))
```

`getSNR` *Extract SNR used for peak detection*

Description

Extract SNR used for peak detection

Usage

```
getSNR(object)
```

Arguments

`object`	Object of class MALDIassay

Value

Numeric, SNR used for peak detection

Examples

```
# see example for `fitCurve()` to see how this data was generated
data(Blank2022res)
getSNR(Blank2022res)
```

`getSpots` *Get the spot coordinates of spectra*

Description

Get the spot coordinates of spectra

Usage

```
getSpots(object, singleSpec = TRUE)
```

Arguments

object	Object of class MALDIassay
singleSpec	Logical, extract the spot coordinates of single spectra (default) or from average spectra.

Value

character vector of spot coordinates. In case of average spectra multiple spots are concatenated.

Examples

```
# see example for `fitCurve()` to see how this data was generated
data(Blank2022res)
# spots per spectrum
getSpots(Blank2022res, singleSpec = TRUE)

#spots per concentration
getSpots(Blank2022res, singleSpec = FALSE)
```

---

getVarFilterMethod *Extract variance filtering method*

Description

Extract variance filtering method

Usage

```
getVarFilterMethod(object)
```

Arguments

object	Object of class MALDIassay

Value

Character of variance filtering method used

Examples

```
# see example for `fitCurve()` to see how this data was generated
data(Blank2022res)
getVarFilterMethod(Blank2022res)
```

isMALDIassay *Check if object if of class MALDIassay*

Description

Check if object if of class MALDIassay

Usage

```
isMALDIassay(object)
```

Arguments

object	object to text

Value

logical, TRUE if object is of class MALDIassay

Examples

```
x <- 1
# FALSE
isMALDIassay(x)
# TRUE
isMALDIassay(Blank2022res)
```

loadSpectra *load bruker MALDI target plate spectra*

Description

load bruker MALDI target plate spectra

Usage

```
loadSpectra(Dir, filter = NA, nameSpectra = TRUE, verbose = TRUE)
```

Arguments

Dir	Character, parent directory of spectra.
filter	Character vector, filter out spectra which match the given vector.
nameSpectra	Logical, if TRUE the spectra in the resulting list will be named according to the dirname.
verbose	Logical, print logs to the console.

Value

List of MALDIquant::MassSpectra

Examples

```
dataDir <- system.file("extdata", package="MALDIcellassay")
unzip(file.path(dataDir, "example-raw-spectra.zip"))

loadSpectra("example-raw-spectra/")

unlink("example-raw-spectra/", recursive = TRUE)
```

---

loadSpectraMzML *load mzML spectra*

Description

load mzML spectra

Usage

```
loadSpectraMzML(Dir, filter = NA, nameSpectra = TRUE, verbose = TRUE)
```

Arguments

Dir	Character, parent directory of spectra.
filter	Character vector, filter out spectra which match the given vector.
nameSpectra	Logical, if TRUE the spectra in the resulting list will be named according to the dirname.
verbose	Logical, print logs to console

Value

List of MALDIquant::MassSpectra

Examples

```
dataDir <- system.file("extdata", package="MALDIcellassay")

loadSpectraMzML(file.path(dataDir, "Koch2024mzML"))
```

MALDIassay-class *Class MALDIassay*

Description

A class for holding MALDI assay related information.

Arguments

object	MALDIassay.

normalize *Normalize spectra and peaks*

Description

Normalize spectra and peaks

Usage

```
normalize(spec, peaks, normMeth, normMz, normTol)
```

Arguments

spec	List of MALDIquant::MassSpectrum
peaks	List of MALDIquant::MassPeaks
normMeth	Character, normalization method. Options are "TIC", "median" and "mz".
normMz	Numeric, mz used to normalize.
normTol	Numeric, tolerance around normMz.

Value

List of lists of normalized MALDIquant::MassSpectrum, normalized MALDIquant::MassPeaks, normalization factors as well as indicies of spectra containing the normMz in case of normMeth = "mz",

Examples

```
data(Blank2022spec)
data(Blank2022peaks)
norm <- normalize(Blank2022spec, Blank2022peaks, normMeth = "mz", normMz = 760.585, normTol = 0.1)

# normalization factors
norm$factor
```

normalizeByFactor *Apply normalization factors to spectra*

Description

Apply normalization factors to spectra

Usage

```
normalizeByFactor(spec, factors)
```

Arguments

spec	List of MALDIquant::MassSpectrum or MALDIquant::MassPeaks
factors	Numeric vector of normalization factors. See getNormFactors().

Value

```
List of normalized Spectra or Peaks
```

Examples

```
#' data(Blank2022peaks)
normFactors <-  getNormFactors(peaks2df(Blank2022peaks),
                               targetMz = 760.585,
                               tol = 0.1,
                               tolppm = FALSE)
normPeaks <- normalizeByFactor(Blank2022peaks,
                               normFactors$norm_factor)
```

peaks2df *Convert a list of peaks to a data.frame*

Description

Convert a list of peaks to a data.frame

Usage

```
peaks2df(peaks)
```

Arguments

peaks	(list of) MALDIquant::MassPeaks

Value

Data.frame with peak data

Examples

```
data(Blank2022peaks)

peakdf <- peaks2df(Blank2022peaks[1:2])
head(peakdf)
```

`plotCurves` *generate ggplot objects for each of the curve fits in a MALDIassay object*

Description

generate ggplot objects for each of the curve fits in a MALDIassay object

Usage

```
plotCurves(object, mzIdx = NULL, errorbars = c("none", "sd", "sem"))
```

Arguments

`object`	object of class MALDIassay
`mzIdx`	numeric, indicies of mz values to plot (see `getPeakStatistics()`). Note, fc_thresh and R2_thresh filters do not apply if mzIdx is set!
`errorbars`	character, add error bars to plot. Either standard error of the mean (`sem`) or standard deviation (`sd`) in regards to the measurement replicates or no errorbars (`none`).

Value

list of ggplot objects

Examples

```
data(Blank2022res)
plotCurves(Blank2022res, mzIdx = 2, errorbars = "sd")
```

plotPeak *Plot a peak of interest from a MALDIassay object*

Description

Plot a peak of interest from a MALDIassay object

Usage

```
plotPeak(object, mzIdx, tol = 0.8)
```

Arguments

object	object of class MALDIassay
mzIdx	numeric, index of mass of interest (see getPeakStatistics())
tol	numeric, tolerance around peak to plot

Value

ggplot object

Examples

```
data(Blank2022res)
plotPeak(Blank2022res, mzIdx = 2)
```

sdMassSpectrum *Compute standard-deviation spectra*

Description

This is a fork from sgibb's MALDIquant::averageMassSpectra() function. It is now able to compute "standard-deviation spectra".

Usage

```
sdMassSpectrum(l, labels, ...)
```

Arguments

l	list, list of MassSpectrum objects.
labels	list, list of factors (one for each MassSpectrum object) to do groupwise averaging.
...	arguments to be passed to underlying functions (currently only mc.cores is supported).

Value

Returns a single (no labels given) or a list (labels given) of standard-deviation spectra as MassSpectrum objects.

Examples

```
data(Blank2022spec)

sdMassSpectrum(Blank2022spec, labels = names(Blank2022spec))[[1]]
```

`shiftMassAxis` *Shift mass axis*

Description

Shift mass axis

Usage

```
shiftMassAxis(spec, mzdiff)
```

Arguments

`spec`	List of MALDIquant::MassSpectrum or MALDIquant::MassPeaks
`mzdiff`	Numeric vector, see getMzShift()

Value

List of MALDIquant::MassSpectrum or MALDIquant::MassPeaks with shifted mass axis.

Examples

```
data(Blank2022spec)
# raw mz
head(Blank2022spec[[1]]@mass)

# shifted mz
shifted <-shiftMassAxis(Blank2022spec[1:2], c(0.5, 0.5))
head(shifted[[1]]@mass)
```

`transformConc2Log` *Convert concentration to log10 and replace zero's*

Description

Convert concentration to log10 and replace zero's

Usage

```
transformConc2Log(conc)
```

Arguments

`conc`	numeric, concentrations.

Value

numeric, log10 transformed concentrations

Examples

```
transformConc2Log(c(0.1, 0.01,0.001))
```

Index