Package 'TPP'

October 12, 2016

Type Package

Title Analyze thermal proteome profiling (TPP) experiments

Version 2.2.6

Author Dorothee Childs, Holger Franken, Mikhail Savitski and Wolfgang Huber

Maintainer Dorothee Childs <dorothee.childs@embl.de>

Depends R (>= 3.2.0), Biobase, openxlsx (>= 2.4.0), ggplot2 (>= 2.0.0)

Imports VGAM, reshape2, nls2, foreach, grid, gridExtra, doParallel, parallel, RColorBrewer, RCurl, plyr, VennDiagram

Suggests BiocStyle, knitr, testthat

Description Analyze thermal proteome profiling (TPP) experiments with varying temperatures (TR) or compound concentrations (CCR).

License Artistic-2.0

VignetteBuilder knitr

biocViews Proteomics, MassSpectrometry

RoxygenNote 5.0.1

NeedsCompilation no

R topics documented:

`analyzeTPPCCR` *Analyze TPP-CCR experiment*

Description

Performs analysis of a TPP-CCR experiment by invoking routines for data import, data processing, normalization, curve fitting, and production of the result table.

Usage

```
analyzeTPPCCR(configTable, data = NULL, resultPath = NULL,
  idVar = "gene_name", fcStr = "rel_fc_", naStrs = c("NA", "n/d", "NaN",
  "<NA>"), qualColName = "qupm", normalize = TRUE,
  ggplotTheme = tppDefaultTheme(), nCores = "max", nonZeroCols = "qssm",
  r2Cutoff = 0.8, fcCutoff = 1.5, slopeBounds = c(1, 50),
  plotCurves = TRUE, verbose = FALSE, xlsxExport = TRUE,
  fcTolerance = 0.1)
```

Arguments

`configTable`	dataframe, or character object with the path to a file, that specifies important details of the TPP-CCR experiment. See Section `details` for instructions how to create this object.
`data`	single dataframe, containing fold change measurements and additional annotation columns to be imported. Can be used instead of specifying the file path in the `configTable` argument.
`resultPath`	location where to store dose-response curve plots and results table.
`idVar`	character string indicating which data column provides the unique identifiers for each protein.

fcStr	character string indicating which columns contain the actual fold change values. Those column names containing the suffix fcStr will be regarded as containing fold change values.
naStrs	character vector indicating missing values in the data table. When reading data from file, this value will be passed on to the argument na.strings in function read.delim.
qualColName	character string indicating which column can be used for additional quality criteria when deciding between different non-unique protein identifiers.
normalize	perform median normalization (default: TRUE).
ggplotTheme	ggplot theme for dose response curve plots.
nCores	either a numerical value given the desired number of CPUs, or 'max' to automatically assign the maximum possible number (default).
nonZeroCols	character string indicating a column that will be used for filtering out zero values.
r2Cutoff	Quality criterion on dose response curve fit.
fcCutoff	Cutoff for highest compound concentration fold change.
slopeBounds	Bounds on the slope parameter for dose response curve fitting.
plotCurves	boolan value indicating whether dose response curves should be plotted. Deactivating plotting decreases runtime.
verbose	print name of each fitted or plotted protein to the command line as a means of progress report.
xlsxExport	produce results table in xlsx format and store at the location specified by the resultPath argument.
fcTolerance	tolerance for the fcCutoff parameter. See details.

Details

Invokes the following steps:

1. Import data using the tppccrImport function.
2. Perform normalization by fold change medians (optional) using the tppccrNormalize function. To perform normalization, set argument normalize=TRUE.
3. Fit and analyse dose response curves using the tppccrCurveFit function.
4. Export results to Excel using the tppExport function.

The default settings are tailored towards the output of the python package isobarQuant, but can be customised to your own dataset by the arguments idVar, fcStr, naStrs, qualColName.

If resultPath is not specified, result files are stored at the path defined in the first entry of configTable$Path. If the input data are not specified in configTable, no result path will be set. This means that no output files or dose response curve plots are produced and analyzeTPPCCR just returns the results as a data frame.

The function analyzeTPPCCR reports intermediate results to the command line. To suppress this, use suppressMessages.

The dose response curve plots will be stored in a subfolder with name `DoseResponse_Curves` at the location specified by `resultPath`.

Only proteins with fold changes bigger than `[fcCutoff * (1 - fcTolerance)` or smaller than `1/(fcCutoff * (1 - fcTolerance))]` will be used for curve fitting. Additionally, the proteins fulfilling the fcCutoff criterion without tolerance will be marked in the output column `meets_FC_requirement`.

Value

A data frame in which the fit results are stored row-wise for each protein.

References

Savitski, M. M., Reinhard, F. B., Franken, H., Werner, T., Savitski, M. F., Eberhard, D., ... & Drewes, G. (2014). Tracking cancer drugs in living cells by thermal profiling of the proteome. Science, 346(6205), 1255784.

See Also

tppDefaultTheme

Examples

```
data(hdacCCR_smallExample)
tppccrResults <- analyzeTPPCCR(configTable=hdacCCR_config,
                               data=hdacCCR_data, nCores=1)
```

`analyzeTPPTR` *Analyze TPP-TR experiment*

Description

Performs analysis of a TPP-TR experiment by invoking routines for data import, data processing, normalization, curve fitting, and production of the result table.

Usage

```
analyzeTPPTR(configTable, data = NULL, resultPath = NULL,
  idVar = "gene_name", fcStr = "rel_fc_", ciStr = NULL, naStrs = c("NA",
  "n/d", "NaN", "<NA>"), qualColName = "qupm", normalize = TRUE,
  normReqs = tpptrDefaultNormReqs(), ggplotTheme = tppDefaultTheme(),
  nCores = "max", startPars = c(Pl = 0, a = 550, b = 10),
  maxAttempts = 500, plotCurves = TRUE, fixedReference = NULL,
  pValMethod = "maxQuant", pValFilter = list(minR2 = 0.8, maxPlateau = 0.3),
  pValParams = list(binWidth = 300), verbose = FALSE, xlsxExport = TRUE)
```

Arguments

configTable	dataframe, or character object with the path to a file, that specifies important details of the TPP-TR experiment. See Section details for instructions how to create this object.
data	single dataframe, or list of dataframes, containing fold change measurements and additional annotation columns to be imported. Can be used instead of specifying the file path in the configTable argument.
resultPath	location where to store melting curve plots, intermediate results, and the final results table.
idVar	character string indicating which data column provides the unique identifiers for each protein.
fcStr	character string indicating which columns contain the actual fold change values. Those column names containing the suffix fcStr will be regarded as containing fold change values.
ciStr	character string indicating which columns contain confidence intervals for the fold change measurements. If specified, confidence intervals will be plotted around the melting curves.
naStrs	character vector indicating missing values in the data table. When reading data from file, this value will be passed on to the argument na.strings in function read.delim.
qualColName	character string indicating which column can be used for additional quality criteria when deciding between different non-unique protein identifiers.
normalize	perform normalization (default: TRUE).
normReqs	list of filtering criteria for construction of the normalization set.
ggplotTheme	ggplot theme for melting curve plots.
nCores	either a numerical value given the desired number of CPUs, or 'max' to automatically assign the maximum possible number (default).
startPars	start values for the melting curve parameters. Will be passed to function nls for curve fitting.
maxAttempts	maximal number of curve fitting attempts if model does not converge.
plotCurves	boolan value indicating whether melting curves should be plotted. Deactivating plotting decreases runtime.
fixedReference	name of a fixed reference experiment for normaliztion. If NULL (default), the experiment with the best R2 when fitting a melting curve through the median fold changes is chosen as the reference.
pValMethod	Method for p-value computation. Currently restricted to 'maxQuant' (see Cox & Mann (2008)).
pValFilter	optional list of filtering criteria to be applied before p-value computation.
pValParams	optional list of parameters for p-value computation.
verbose	print name of each fitted protein to the command lin as a means of progress report.
xlsxExport	boolean value indicating whether to produce result table in .xlsx format (requires package openxlsx and a zip application to be installed).

Details

Invokes the following steps:

1. Import data using the `tpptrImport` function.
2. Perform normalization (optional) using the `tpptrNormalize` function. To perform normalization, set argument `normalize=TRUE`. The normalization will be filtered according to the criteria specified in the `normReqs` argument (also see the documentation of `tpptrNormalize` and `tpptrDefaultNormReqs` for further information).
3. Fit melting curves using the function `tpptrCurveFit`.
4. Produce result table using the function `tpptrAnalyzeMeltingCurves`.
5. Export results to Excel using the function `tppExport`.

The default settings are tailored towards the output of the python package isobarQuant, but can be customised to your own dataset by the arguments `idVar, fcStr, naStrs, qualColName`.

If `resultPath` is not specified, the location of the first input file specified in `configTable` will be used. If the input data are not specified in `configTable`, no result path will be set. This means that no output files or melting curve plots are produced and `analyzeTPPTR` just returns the results as a data frame.

The function `analyzeTPPTR` reports intermediate results to the command line. To suppress this, use `suppressMessages`.

The `configTable` argument is a dataframe, or the path to a spreadsheet (tab-delimited text-file or xlsx format). Information about each experiment is stored row-wise. It contains the following columns:

- `Path`:location of each datafile. Alternatively, data can be directly handed over by the `data` argument.
- `Experiment`: unique experiment names.
- `Condition`: experimental conditions of each dataset.
- Label columns: each isobaric label names a column that contains the temperatures administered for the label in the individual experiments.

The argument `nCores` could be either 'max' (use all available cores) or an upper limit of CPUs to be used.

If `doPlot = TRUE`, melting curve plots are generated seperately for each protein and stored in separate pdfs. Each file is named by the unique protein identifier. Filenames are truncated to 255 characters (requirement by most operation systems). Truncated filenames are indicated by the suffix "_truncated[d]", where [d] is a unique number to avoid redundancies. All melting curve plots are stored in a subfolder with name `Melting_Curves` at the location specified by `resultPath`.

If the melting curve fitting procedure does not converge, it will be repeatedly started from perturbed starting parameters (maximum iterations defined by argument `maxAttempts`).

Value

A data frame in which the fit results are stored row-wise for each protein.

References

Savitski, M. M., Reinhard, F. B., Franken, H., Werner, T., Savitski, M. F., Eberhard, D., ... & Drewes, G. (2014). Tracking cancer drugs in living cells by thermal profiling of the proteome. Science, 346(6205), 1255784.

See Also

tppDefaultTheme, tpptrImport, tpptrNormalize, tpptrCurveFit, tpptrAnalyzeMeltingCurves

Examples

```
data(hdacTR_smallExample)
tpptrResults <- analyzeTPPTR(configTable=hdacTR_config, data=hdacTR_data, nCores=1)
```

---

hdacCCR_config *The configuration table to analyze hdacCCR_data.*

Description

The configuration table to analyze hdacCCR_data.

Details

`hdacCCR_config` is a data frame that specifies the experiment names, isobaric labels, and the administered drug concentrations at each label.

---

hdacCCR_data *TPP-CCR example dataset (replicates 1 and 2)*

Description

Example subset of a Panobinostat TPP-CCR dataset (replicates 1 and 2)

Details

A list with two subsets of a dataset obtained by TPP-CCR experiments to investigate drug effects for HDAC inhibitor Panobinostat. It contains 7 HDACs as well as a random selection of 493 further proteins.

You can use this dataset to explore the TPP package functionalities without invoking the whole time consuming analysis on the big dataset.

The originial dataset is located in the folder `'example_data/CCR_example_data'` in the package's installation directory. You can find it on your system by the R command `system.file('example_data', package = 'TPP')`

`hdacCCR_smallExample` *Example subsets of a Panobinostat TPP-CCR dataset (replicates 1 and 2) and the corresponding configuration table to start the analysis.*

Description

Example dataset obtained by TPP-CCR experiments for analysis by the TPP-package. It contains all necessary arguments to start the analysis (config table and list of data frames).

`hdacTR_config` *The configuration table to analyze hdacTR_data.*

Description

The configuration table to analyze hdacTR_data.

Details

`hdacTR_config` is a data frame that specifies the experiment name, isobaric labels, and the administered temperatures at each label.

`hdacTR_data` *TPP-TR example dataset.*

Description

Example subset of a dataset obtained by TPP-TR experiments to investigate possible targets for HDAC inhibitor Panobinostat.

Details

`hdacTR_data` is a list of data frames that contain measurements for HDACs as well as a random selection of 500 further proteins.

You can use this dataset to explore the `TPP` package functionalities without invoking the whole time consuming analysis on the whole dataset.

The originial dataset is located in the folder `'example_data/TR_example_data'` in the package's installation directory. You can find it on your system by the `R` command `system.file('example_data', package = 'TPP')`

hdacTR_resultsTable_smallExample *Example of a TPP-TR result table.*

Description

Example of a TPP-TR result table.

Details

Contains the data object `resultTable`.

hdacTR_smallExample *Example subset of a Panobinostat TPP-TR dataset and the corresponding configuration table to start the analysis.*

Description

Example dataset obtained by TPP-TR experiments for analysis by the TPP-package. It contains all necessary arguments to start the analysis (config table and list of data frames).

resultTable *Example of a TPP-TR result table.*

Description

Example of a TPP-TR result table.

Details

`resultTable` is a data frame that contains the measurements of several TPP-TR experiments, the fitted melting curve parameters, as well as p-values and the results of additional quality checks for each protein. It can be used as input for the function `tppQCPlotsCorrelateExperiments`.

TPP *Thermal proteome profiling (TPP)*

Description

TPP is a toolbox for analyzing thermal proteome profiling (TPP) experiments.

Usage

```
.onLoad(libname, pkgname)
```

Arguments

libname	a character string giving the library directory where the package defining the namespace was found. Passed to .onLoad function.
pkgname	a character string giving the name of the package. Passed to .onLoad function.

Details

In order to start a TPP-TR analysis, use function analyzeTPPTR. For a TPP-CCR analysis, use function analyzeTPPCCR. See the vignette for detailed instructions.

Value

No return value defined for this document.

References

Savitski, M. M., Reinhard, F. B., Franken, H., Werner, T., Savitski, M. F., Eberhard, D., ... & Drewes, G. (2014). Tracking cancer drugs in living cells by thermal profiling of the proteome. Science, 346(6205), 1255784.

tppccrCurveFit *Fit dose response curves*

Description

tppccrCurveFit fits logistic dose response curves to fold change measurements of a TPP-CCR experiment.

Usage

```
tppccrCurveFit(data = NULL, fcTable = NULL, cpdEffects = NULL,
  slopeBounds = c(1, 50), nCores = "max", verbose = FALSE)
```

Arguments

data	list of expressionSet objects containing protein fold changes for dose response curve fitting.
fcTable	optional long table with fold changes for each experiment. Can be provided instead of the input argument data.
cpdEffects	optional long table of compound effects per protein and experiment. Can be provided instead of the input argument data.
slopeBounds	bounds on the slope parameter for dose response curve fitting.
nCores	either a numerical value given the desired number of CPUs, or 'max' to automatically assign the maximum possible number (default).
verbose	print name of each fitted protein to the command line as a means of progress report.

Details

data is a list of expressionSet objects created by tppccrImport. If desired, it can be already preprocessed by tppccrNormalize or tppccrTransform. It contains the isobaric labels and administered drug concentrations in the phenoData and user-defined protein properties in the featureData. Protein IDs are stored in the featureNames.

Measurements and compound effects for curve fitting can be provided by the arguments fcTable and cpdEffects, instead of being stored in expressionSets in data.

If specified, fcTable needs to be a long table with column names "id" (the protein names), "concentration" (the fold changes), "labelName" (the isobaric label to each measurement), and "experiment" (e.g. "Vehicle_1" or "Panobinostat_1").

If specified, cpdEffects needs to be a long table with column names "id" (the protein names), "cpdEff" (character vector of compound effects, may contain NAs), and "experiment" (e.g. "Vehicle_1" or "Panobinostat_1").

Value

A list of expressionSet objects storing fold changes, the fitted curve parameters, as well as row and column metadata. In each expressionSet S, the fold changes can be accessed by exprs(S). Protein expNames can be accessed by featureNames(S). Isobaric labels and the corresponding concentrations are returned by S$label and S$concentration. The fitted curve parameters are stored in codefeatureData(S).

See Also

tppccrImport, tppccrNormalize, tppccrTransform

Examples

```
data(hdacCCR_smallExample)
tppccrData <- tppccrImport(configTable=hdacCCR_config,
                           data=hdacCCR_data)
tppccrNorm <- tppccrNormalize(data=tppccrData)
tppccrTransformed <- tppccrTransform(data=tppccrNorm)
```

```
tppccrFitted <- tppccrCurveFit(data=tppccrTransformed, nCores=1)
```

`tppccrImport` *Import TPP-CCR dataset for analysis by the TPP package.*

Description

`tppccrImport` imports a table of protein fold changes and stores them in an ExpressionSet for use in the TPP package.

Usage

```
tppccrImport(configTable, data = NULL, idVar = "gene_name",
  fcStr = "rel_fc_", naStrs = c("NA", "n/d", "NaN", "<NA>"),
  qualColName = "qupm", nonZeroCols = "qssm")
```

Arguments

`configTable`	either a dataframe or the path to a spreadsheet. In both cases it specifies necessary information of the TPP-CCR experiment.
`data`	dataframe containing fold change measurements and additional annotation columns to be imported. Can be used instead of specifying the file path in `configTable`.
`idVar`	character string indicating which data column provides the unique identifiers for each protein.
`fcStr`	character string indicating which columns contain the actual fold change values. Those column names containing the suffix `fcStr` will be regarded as containing fold change values.
`naStrs`	character vector indicating missing values in the data table. When reading data from file, this value will be passed on to the argument `na.strings` in function `read.delim`.
`qualColName`	character string indicating which column can be used for additional quality criteria when deciding between different non-unique protein identifiers.
`nonZeroCols`	character string indicating a column that will be used for filtering out zero values.

Details

The imported dataset has to contain measurements obtained by a TPP-CCR experiment. Fold changes need to be pre-computed using the lowest concentration as reference.

The dataset can be specified by filename in the `configTable` argument, or given directly in the `data` argument

The default settings are adjusted to analyse data of the pyhton package `isobarQuant`. You can also customise them for your own dataset.

The `configTable` argument is a dataframe, or the path to a spreadsheet (tab-delimited text-file without quoted strings, or xlsx format). Information about each experiment is stored row-wise. It contains the following columns:

- Path: location of the datafile. Alternatively, data can be directly handed over by the data argument.
- Experiment: unique experiment name.
- Label columns: each isobaric label names a column that contains the concentration administered for the label in the individual experiments.

During data import, proteins with NAs in the data column specified by idVar receive unique generic IDs so that they can be processed by the package.

Value

ExpressionSet object storing the measured fold changes, as well as row and column metadata. In each ExpressionSet S, the fold changes can be accessed by exprs(S). Protein expNames can be accessed by featureNames(S). Isobaric labels and the corresponding concentrations are returned by S$label and S$concentration.

See Also

tpptrImport, tppccrCurveFit

Examples

```
data(hdacCCR_smallExample)
tppccrData <- tppccrImport(configTable=hdacCCR_config,
data = hdacCCR_data)
```

---

tppccrNormalize *Normalize data from TPP-CCR experiments*

Description

Normalize each fold change column by its median.

Usage

```
tppccrNormalize(data)
```

Arguments

data	list of expressionSets with measurements to be normalized

Value

List of expressionSet objects storing the normalized fold changes, as well as row and column metadata. In each expressionSet S, the fold changes can be accessed by exprs(S). Protein names can be accessed by featureNames(S). Isobaric labels and the corresponding concentrations are returned by S$label and S$concentration.

Examples

```
data(hdacCCR_smallExample)
tppccrData <- tppccrImport(configTable=hdacCCR_config, data = hdacCCR_data)
tppccrNorm <- tppccrNormalize(data=tppccrData)
head(exprs(tppccrNorm[[1]]))
```

`tppccrNormalizeToReference` *Normalize fold changes of TPP-CCR experiment to a reference column*

Description

Normalize fold changes of TPP-CCR experiment to a reference column (usually that with the lowest concentration) to ensure that the transformation by tppccrTransform yields values between 0 and 1.

Usage

```
tppccrNormalizeToReference(data, refCol = NULL)
```

Arguments

`data`	expressionSet object containing the data to be normalized
`refCol`	column number to use as a reference. Will contain only 1s after the normalization.

Value

List of expressionSet objects storing the normalized fold changes, as well as row and column metadata. In each expressionSet `S`, the fold changes can be accessed by `exprs(S)`. Protein expNames can be accessed by `featureNames(S)`. Isobaric labels and the corresponding concentrations are returned by `S$label` and `S$concentration`.

Examples

```
data(hdacCCR_smallExample)
tppccrData <- tppccrImport(configTable=hdacCCR_config, data = hdacCCR_data)
tppccrNorm <- tppccrNormalize(data=tppccrData)
# Normalize to lowest concentration (in the first column):
tppccrNormToRef <- tppccrNormalizeToReference(data=tppccrNorm, refCol=1)
# Obtain results per replicate:
refTransf_replicate1 <- tppccrNormToRef$Panobinostat_1
head(exprs(refTransf_replicate1))
# Perform transformation:
tppccrTransformed <- tppccrTransform(data=tppccrNormToRef)
# Obtain transformed measurements per replicate:
transf_replicate1 <- tppccrTransformed$Panobinostat_1
transf_replicate2 <- tppccrTransformed$Panobinostat_2
```

```
# Inspect transformed data in replicate 1:
effects_replicate1 <- featureData(transf_replicate1)$compound_effect
newData_repl1 <- data.frame(exprs(transf_replicate1),
                            Type=effects_replicate1)[!is.na(effects_replicate1),]
```

---

tppccrPlotCurves *Plot dose response curves*

Description

tppccrPlotCurves plots the logistic dose response curves, as well as the underlying fold change measurements for each TPP-CCR experiment in a study.

Usage

```
tppccrPlotCurves(data = NULL, fcTable = NULL, curvePars = NULL,
  resultPath = NULL, ggplotTheme = tppDefaultTheme(), nCores = "max",
  verbose = FALSE)
```

Arguments

data	list of expressionSet objects containing protein fold changes, as well as fitted curve parameters.
fcTable	optional long table with fold changes for each experiment. Can be provided instead of the input argument data.
curvePars	optional long table of curve parameters per protein and experiment. Can be provided instead of the input argument data.
resultPath	location where to store dose-response curve plots.
ggplotTheme	ggplot theme for dose response curve plots.
nCores	either a numerical value given the desired number of CPUs, or 'max' to automatically assign the maximum possible number (default).
verbose	print name of each plotted protein to the command line as a means of progress report.

Details

data is a list of expressionSet objects created by tppccrCurveFit. It contains the isobaric labels and administered drug concentrations in the phenoData and user-defined protein properties (including dose response curve parameters) in the featureData. Protein IDs are stored in the featureNames.

Measurements and compound effects for curve fitting can be provided by the arguments fcTable and cpdEffects, instead of being stored in expressionSets in data.

If specified, fcTable needs to be a long table with column names "id" (the protein names), "concentration" (the fold changes), "labelName" (the isobaric label to each measurement), and "experiment" (e.g. "Vehicle_1" or "Panobinostat_1").

If specified, curvePars needs to be a long table with column names "id" (the protein names), "param" (curve parameter per protein and experiment, see TPP:::drCurveParamNames(names=TRUE, info=FALSE) for possibilities), and "experiment" (e.g. "Vehicle_1" or "Panobinostat_1").

The dose response curve plots will be stored in a subfolder with name DoseResponse_Curves at the location specified by resultPath.

Value

A list of expressionSet objects storing fold changes, as well as row and column metadata. In each expressionSet S, the fold changes can be accessed by exprs(S). Protein expNames can be accessed by featureNames(S). Isobaric labels and the corresponding concentrations are returned by S$label and S$concentration. Paths to the produced plots are stored in codefeatureData(S)$plot.

See Also

tppccrCurveFit,tppDefaultTheme

Examples

```
data(hdacCCR_smallExample)
tppccrData <- tppccrImport(configTable=hdacCCR_config,
                           data=hdacCCR_data)
tppccrNorm <- tppccrNormalize(data=tppccrData)
tppccrTransformed <- tppccrTransform(data=tppccrNorm)
tppccrFitted <- tppccrCurveFit(data=tppccrTransformed, nCores=1)
protein1_to_5 <- sapply(tppccrFitted, function(d) d[1:5,])
tppccrPlotted <- tppccrPlotCurves(data=protein1_to_5, resultPath=getwd(), nCores=1)
```

---

tppccrResultTable *Summarize results of a TPP-CCR study*

Description

tppccrResultTable summarizes the outcomes of a TPP-CCR study in a results table and includes quality information about the estimated dose response curves.

Usage

```
tppccrResultTable(data, r2Cutoff = 0.8)
```

Arguments

data	list of expressionSet objects containing protein fold changes, as well as fitted curve parameters.

r2Cutoff	quality criterion on dose response curve fit.
	@details data is a list of expressionSet objects created by tppccrCurveFit or tppccrPlotCurves. It contains the isobaric labels and administered drug concentrations in the phenoData and user-defined protein properties (including dose response curve parameters) in the featureData. Protein IDs are stored in the featureNames.
	If data is the output of tppccrPlotCurves, plot locations are given in the plot column of the featureData.

Value

A data frame in which the results are stored row-wise for each protein, together with the original annotation from the input files.

See Also

tppccrCurveFit,tppccrPlotCurves

Examples

```
data(hdacCCR_smallExample)
tppccrData <- tppccrImport(configTable=hdacCCR_config,
                           data=hdacCCR_data)
tppccrNorm <- tppccrNormalize(data=tppccrData)
tppccrTransformed <- tppccrTransform(data=tppccrNorm)
tppccrFitted <- tppccrCurveFit(data=tppccrTransformed, nCores=1)
tppccrResults <- tppccrResultTable(data=tppccrFitted)
subset(tppccrResults, passed_filter_Panobinostat_1 & passed_filter_Panobinostat_2)
```

---

tppccrTransform *Transform fold changes of TPP-CCR experiment*

Description

Transform fold changes of TPP-CCR experiment to prepare them for dose response curve fitting.

Usage

```
tppccrTransform(data, fcCutoff = 1.5, fcTolerance = 0.1)
```

Arguments

data	expressionSet object containing the data to be transformed.
fcCutoff	cutoff for highest compound concentration fold change.
fcTolerance	tolerance for the fcCutoff parameter. See details.

Details

Only proteins with fold changes bigger than [fcCutoff * (1 - fcTolerance) or smaller than 1/(fcCutoff * (1 - fcTolerance))] will be used for curve fitting. Additionally, the proteins fulfilling the fcCutoff criterion without tolerance will be marked in the output column meets_FC_requirement.

Value

List of expressionSet objects storing the transformed fold changes, as well as row and column metadata. In each expressionSet S, the fold changes can be accessed by exprs(S). Protein expNames can be accessed by featureNames(S). Isobaric labels and the corresponding concentrations are returned by S$label and S$concentration.

Examples

```
data(hdacCCR_smallExample)
tppccrData <- tppccrImport(configTable=hdacCCR_config, data = hdacCCR_data)
tppccrNorm <- tppccrNormalize(data=tppccrData)
# Perform transformation:
tppccrTransformed <- tppccrTransform(data=tppccrNorm)
# Obtain transformed measurements per replicate:
transf_replicate1 <- tppccrTransformed$Panobinostat_1
transf_replicate2 <- tppccrTransformed$Panobinostat_2
# Inspect transformed data in replicate 1:
effects_replicate1 <- featureData(transf_replicate1)$compound_effect
newData_repl1 <- data.frame(exprs(transf_replicate1),
                            Type=effects_replicate1)[!is.na(effects_replicate1),]
```

---

tppDefaultTheme *Default ggplot theme for melting curve plots.*

Description

Default theme to be passed to the gplots produced by the TPP package.

Usage

```
tppDefaultTheme()
```

Details

Internally, the theme is used as an argument for the function ggplot2::theme_set in order specify the appearance of the melting curve plots.

The specified plot properties include bold font and increased font size for axis labels and title, as well as a 90 degree angle for y axis labels.

Value

ggplot theme with default settings for melting plot apparance.

Examples

```
# Import data:
data(hdacTR_smallExample)
tpptrData <- tpptrImport(configTable=hdacTR_config, data=hdacTR_data)
# Obtain template with default settings:
normRequirements <- tpptrDefaultNormReqs()
print(normRequirements)
# Relax filter on the 10th fold change column for
# normalization set production:
normRequirements$fcRequirements[3,3] <- 0.25
# Perform normalization:
tpptrNorm <- tpptrNormalize(data=tpptrData, normReqs=)
```

---

tppExport *Produce Excel table of TPP-TR or TPP-CCR experiment.*

Description

Produce Excel table of TPP-TR or TPP-CCR experiment out of the data frame returned by `tpptrAnalyzeMeltingCurves`

Usage

```
tppExport(tab, file, expNames = NULL, expColors = NULL)
```

Arguments

`tab`	Table with results of the TPP analysis.
`file`	path for storing results table
`expNames`	character vector of experiment names of the same length as expColors.
`expColors`	character vector of background colors to group the result columns belonging to different experiments.

Value

No value returned.

Examples

```
data(hdacTR_resultsTable_smallExample)
tppExport(resultTable, "tpptr_example_results.xlsx")
```

`tppQCPlotsCorrelateExperiments` *Visually compare fold changes of different TPP experiments.*

Description

Plot pairwise relationships between the proteins in different TPP experiments.

Usage

```
tppQCPlotsCorrelateExperiments(tppData, annotStr = "", path = NULL,
  ggplotTheme = tppDefaultTheme())
```

Arguments

`tppData`	List of expressionSets with data to be plotted.
`annotStr`	String with additional information to be added to the plot.
`path`	Location where to store resulting plot.
`ggplotTheme`	ggplot theme for the created plots.

Value

List of plots for each experiment.

See Also

`tppDefaultTheme`

Examples

```
data(hdacTR_smallExample)
tpptrData <- tpptrImport(configTable=hdacTR_config, data=hdacTR_data)
# Quality control (QC) plots BEFORE normalization:
tppQCPlotsCorrelateExperiments(tppData=tpptrData,
annotStr="Non-normalized Fold Changes")
# Quality control (QC) plots AFTER normalization:
tpptrNorm <- tpptrNormalize(data=tpptrData, normReqs=tpptrDefaultNormReqs())
tpptrDataNormalized <- tpptrNorm$normData
tppQCPlotsCorrelateExperiments(tppData=tpptrDataNormalized,
annotStr="Normalized Fold Changes")
```

`tpptrAnalyzeMeltingCurves` *Analyze fitted curve parameters to detect significant shifts in melting points.*

Description

Compute p-values for the pairwise comparisons of melting curve shifts between different conditions.

Usage

```
tpptrAnalyzeMeltingCurves(data, pValMethod = "maxQuant",
  pValFilter = list(minR2 = 0.8, maxPlateau = 0.3),
  pValParams = list(binWidth = 300))
```

Arguments

`data`	list of ExpressionSets containing fold changes and metadata. Their featureData fields contain the fitted melting curve parameters.
`pValMethod`	Method for p-value computation. Currently restricted to 'maxQuant' (see Cox & Mann (2008)).
`pValFilter`	optional list of filtering criteria to be applied before p-value computation.
`pValParams`	optional list of parameters for p-value computation.

Details

The `pValParams` argument is a list that can contain optional parameters for the chosen p-value computation `pValMethod`. The following options are available:

1. `pValMethod = "maxQuant"`:
   `pValParams=list(binWidth=[your_binWidth])`.

Value

A data frame in which the fit results are stored row-wise for each protein.

References

Cox, J., & Mann, M. (2008). MaxQuant enables high peptide identification rates, individualized ppb-range mass accuracies and proteome-wide protein quantification. Nature biotechnology, 26(12), 1367-1372.

Examples

```
data(hdacTR_smallExample)
tpptrData <- tpptrImport(hdacTR_config, hdacTR_data)
tpptrNorm <- tpptrNormalize(data=tpptrData,
                            normReqs=tpptrDefaultNormReqs())
normalizedData <- tpptrNorm$normData
## Not run:
# Fit melting curves to each protein
# (can take some time depending on device used):
 fittedData <- tpptrCurveFit(normalizedData, nCores=1)
 resultTable <- tpptrAnalyzeMeltingCurves(fittedData)
 subset(resultTable, fulfills_all_4_requirements)$Protein_ID

## End(Not run)
```

`tpptrCurveFit` *Fit melting curves to all proteins in a dataset.*

Description

Fit melting curves to all proteins in a dataset.

Usage

```
tpptrCurveFit(data, dataCI = NULL, resultPath = NULL,
  ggplotTheme = tppDefaultTheme(), doPlot = TRUE, startPars = c(Pl = 0, a
  = 550, b = 10), maxAttempts = 500, nCores = "max", verbose = FALSE)
```

Arguments

`data`	list of `ExpressionSets` with protein fold changes for curve fitting.
`dataCI`	list of `ExpressionSets` with protein fold change confidence intervals for curve fitting. Default to NULL.
`resultPath`	location where to store the melting curve plots.
`ggplotTheme`	ggplot theme for melting curve plots.
`doPlot`	boolan value indicating whether melting curves should be plotted, or whether just the curve parameters should be returned.
`startPars`	start values for the melting curve parameters. Will be passed to function `nls` for curve fitting.
`maxAttempts`	maximal number of curve fitting attempts if model does not converge.
`nCores`	either a numerical value given the desired number of CPUs, or 'max' to automatically assign the maximum possible number (default).
`verbose`	plot name of each fitted protein to the command lin as a means of progress report.

Details

If the melting curve fitting procedure does not converge, it will be repeatedly started from perturbed starting parameters (maximum iterations defined by argument `maxAttempts`)

If `doPlot = TRUE`, melting curves are be plotted in individual files per protein. Each file is named by its unique identifier. Filenames are truncated to 255 characters (requirement by most operation systems). Truncated filenames are indicated by the suffix "_truncated[d]", where [d] is a unique number to avoid redundancies.

The melting curve plots will be stored in a subfolder with name `Melting_Curves` at the location specified by `resultPath`.

Value

A list of ExpressionSets storing the data together with the melting curve parameters for each experiment. Each ExpressionSet contains the measured fold changes, as well as row and column metadata. In each ExpressionSet `S`, the fold changes can be accessed by `exprs(S)`. Protein expNames can be accessed by `featureNames(S)`. Isobaric labels and the corresponding temperatures are returned by `S$label` and `S$temperature`.

See Also

`tppDefaultTheme`

Examples

```
data(hdacTR_smallExample)
tpptrData <- tpptrImport(configTable=hdacTR_config, data=hdacTR_data)
tpptrNorm <- tpptrNormalize(data=tpptrData, normReqs=tpptrDefaultNormReqs())
normalizedData <- tpptrNorm$normData
hdacSubsets <- lapply(normalizedData,
                      function(d) d[grepl("HDAC", featureNames(d))])
tpptrFittedHDACs <- tpptrCurveFit(hdacSubsets, nCores=1)
# Show estimated parameters for vehicle and treatment experiments:
pData(featureData(tpptrFittedHDACs[["Vehicle_1"]]))
pData(featureData(tpptrFittedHDACs[["Panobinostat_1"]]))
```

---

tpptrDefaultNormReqs *Default filter criteria for fold change normalization*

Description

Filter criteria as described in the publication.

Usage

```
tpptrDefaultNormReqs()
```

Value

List with two entries: 'fcRequirements' describes filtering requirements on fold change columns, 'otherRequirements' contains criteria on additional metadata columns.

Examples

```
data(hdacTR_smallExample)
tpptrData <- tpptrImport(configTable=hdacTR_config, data=hdacTR_data)
tpptrNorm <- tpptrNormalize(data=tpptrData, normReqs=tpptrDefaultNormReqs())
```

`tpptrImport` *Import TPP-TR datasets for analysis by the* TPP *package.*

Description

`tpptrImport` imports several tables of protein fold changes and stores them in a list of ExpressionSets for use in the TPP package.

Usage

```
tpptrImport(configTable, data = NULL, idVar = "gene_name",
  fcStr = "rel_fc_", naStrs = c("NA", "n/d", "NaN"), qualColName = "qupm")
```

Arguments

`configTable`	either a dataframe or the path to a spreadsheet. In both cases it specifies necessary information of the TPP-CCR experiment.
`data`	single dataframe, or list of dataframes, containing fold change measurements and additional annotation columns to be imported. Can be used instead of specifying the file path in `configTable`.
`idVar`	character string indicating which data column provides the unique identifiers for each protein.
`fcStr`	character string indicating which columns contain the actual fold change values. Those column names containing the suffix `fcStr` will be regarded as containing fold change values.
`naStrs`	character vector indicating missing values in the data table. When reading data from file, this value will be passed on to the argument `na.strings` in function `read.delim`.
`qualColName`	character string indicating which column can be used for additional quality criteria when deciding between different non-unique protein identifiers.

Details

The imported datasets have to contain measurements obtained by TPP-TR experiments. Fold changes need to be pre-computed using the lowest temperature as reference.

An arbitrary number of datasets can be specified by filename in the `Path`-column of the `configTable` argument, or given directly as a list of dataframes in the `data` argument. They can differ, for example, by biological replicate or by experimental condition (for example, treatment versus vehicle). Their names are defined uniquely by the `Experiment` column in `configTable`. Experimental conditions can be specified by an optional column in `configTable`.

The default settings are adjusted to analyse data of the pyhton package `isobarQuant`. You can also customise them for your own dataset.

The `configTable` argument is a dataframe, or the path to a spreadsheet (tab-delimited text-file without quoted strings, or xlsx format). Information about each experiment is stored row-wise. It contains the following columns:

- `Path`:location of each datafile. Alternatively, data can be directly handed over by the `data` argument.
- `Experiment`: unique experiment names.
- `Condition`: experimental conditions of each dataset.
- Label columns: each isobaric label names a column that contains the temperatures administered for the label in the individual experiments.

Proteins with NAs in the data column specified by `idVar` receive unique generic IDs so that they can be processed by the package.

Value

A list of ExpressionSets storing the imported data for experiment. Each ExpressionSet contains the measured fold changes, as well as row and column metadata. In each ExpressionSet `S`, the fold changes can be accessed by `exprs(S)`. Protein expNames can be accessed by `featureNames(S)`. Isobaric labels and the corresponding temperatures are returned by `S$label` and `S$temperature`

See Also

`tppccrImport`

Examples

```
data(hdacTR_smallExample)
tpptrData <- tpptrImport(hdacTR_config, hdacTR_data)
```

`tpptrNormalize` *Normalize protein fold changes*

Description

Normalizes fold changes determined by TPP-TR experiments over different experimental groups.

Usage

```
tpptrNormalize(data, normReqs = tpptrDefaultNormReqs(),
  qcPlotTheme = tppDefaultTheme(), qcPlotPath = NULL, startPars = c(Pl =
  0, a = 550, b = 10), maxAttempts = 1, fixedReference = NULL)
```

Arguments

`data`	List of `ExpressionSets` with protein fold changes to be normalized.
`normReqs`	List of filtering criteria for construction of the normalization set.
`qcPlotTheme`	ggplot theme for the created plots
`qcPlotPath`	location where plots of the curves fitted to the normalization set medians should be stored.
`startPars`	start values for the melting curve parameters. Will be passed to function `nls` for curve fitting.
`maxAttempts`	maximal number of curve attempts to fit melting curve to fold change medians when computing normalization factors.
`fixedReference`	name of a fixed reference experiment for normaliztion. If NULL (default), the experiment with the best R2 when fitting a melting curve through the median fold changes is chosen as the reference.

Details

Performs normalization of all fold changes in a given list of ExpressionSets. The normalization procedure is described in detail in Savitski et al. (2014). Whether normalization needs to be performed and what method is best suited depends on the experiment. Here we provide a reasonable solution for the data at hand.

We distinguish between filtering conditions on fold changes and on additional annotation columns. Correspondingly, `normReqs` contains two fields, `fcFilters` and `otherFilters`. Each entry contains a data frame with three columns specifying the column to be filtered, as well as upper and lower bounds. An example is given by `tpptrDefaultNormReqs`.

Value

A list of ExpressionSets storing the normalized data for each experiment. Each ExpressionSet contains the measured fold changes, as well as row and column metadata. In each ExpressionSet `S`, the fold changes can be accessed by `exprs(S)`. Protein expNames can be accessed by `featureNames(S)`. Isobaric labels and the corresponding temperatures are returned by `S$label` and `S$temperature`

References

Savitski, M. M. and Reinhard, F. BM. and Franken, H. and Werner, T. and Savitski, M. F. and Eberhard, D. and Molina, D. M. and Jafari, R. and Dovega, R. B. and Klaeger, S. and others (2014) Tracking cancer drugs in living cells by thermal profiling of the proteome. Science 346(6205), p. 1255784.

See Also

tpptrImport

Examples

```
data(hdacTR_smallExample)
tpptrData <- tpptrImport(hdacTR_config, hdacTR_data)
tpptrNorm <- tpptrNormalize(data=tpptrData, normReqs=tpptrDefaultNormReqs())
names(tpptrNorm)
```

Index