Package: HaDeX2 (via r-universe)

May 11, 2026

Title Analysis and Visualisation of Hydrogen/Deuterium Exchange Mass Spectrometry Data

Version 1.0.0

Description Processing, analysis and visualization of Hydrogen Deuterium eXchange monitored by Mass Spectrometry experiments (HDX-MS). 'HaDeX2' introduces a new standardized and reproducible workflow for the analysis of the HDX-MS data, including uncertainty propagation, data aggregation and visualization on 3D structure. Additionally, it covers data exploration, quality control and generation of publication-quality figures. All functionalities are also available in the accompanying 'shiny' app.

Depends R (>= 3.5)

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.3.3

Imports checkmate, data.table, dplyr, ggplot2, glue, gridExtra, magick, purrr, readr, readxl, r3dmol (>= 0.1.2), remotes, stringi, tidyr, ggiraph

Suggests bookdown, digest, knitr, magrittr, pander, renv, rmarkdown, microbenchmark, testthat, vdiffr, scales, shiny, spelling

VignetteBuilder knitr

Language en-US

URL https://hadexversum.github.io/HaDeX2/

Config/pak/sysreqs libcairo2-dev cmake libfontconfig1-dev libfreetype6-dev git make libmagick++-dev gsfonts libicu-dev libpng-dev libuv1-dev libssl-dev libx11-dev

Repository https://hadexversum.r-universe.dev

Date/Publication 2026-02-08 23:27:21 UTC

RemoteUrl https://github.com/hadexversum/hadex2

RemoteRef HEAD

RemoteSha 41769f60887545bb3f7861341970f8f02efa320e

Contents

`HaDeX2-package` *HaDeX2*

Description

The `HaDeX2` package is a toolbox for the analysis of HDX-MS data.

Author(s)

Weronika Puchala, Michal Burdukiewicz.

See Also

Useful links:

- https://hadexversum.github.io/HaDeX2/

`add_stat_dependency` *Calculates confidence limits*

Description

Returns relation with confidence limits for each peptide.

Usage

```
add_stat_dependency(
  calc_dat,
  confidence_level = 0.98,
  theoretical = FALSE,
  fractional = TRUE
)
```

Arguments

`calc_dat`	data produced by `calculate_diff_uptake` function.
`confidence_level`	confidence limit - from range [0, 1].
`theoretical`	`logical`, determines if values are theoretical.
`fractional`	`logical`, determines if values are fractional.

Details

This function checks if the values are statistically significant based on provided criteria using Houde test.

Value

calc_dat extended by column specifying if given peptide is relevant in given confidence limit. The value of the confidence limit is added as an attribute - as well as parameters used to calculate (theoretical/fractional).

Examples

```
calc_dat <- calculate_diff_uptake(alpha_dat)
result <- add_stat_dependency(calc_dat)
head(result)
```

`alpha_dat` *Elongation factor eEF1B subunit alpha*

Description

Originally published in: Bondarchuk, T. V., Shalak, V. F., Lozhko, D. M., Fatalska, A., Szczepanowski, R. H., Liudkovska, V., Tsuvariev, O. Y., Dadlez, M., El'skaya, A. V., & Negrutskii, B. S. (2022). Quaternary organization of the human eEF1B complex reveals unique multi-GEF domain assembly. Nucleic Acids Research, 50(16), 9490–9504. <doi:/10.1093/nar/gkac685>

Author(s)

Bondarchuk, T. V., Shalak, V. F., Lozhko, D. M., Fatalska, A., Szczepanowski, R. H., Liudkovska, V., Tsuvariev, O. Y., Dadlez, M., El'skaya, A. V., & Negrutskii, B. S.

References

doi:10.1093/nar/gkac685

`calculate_aggregated_diff_uptake`
Calculates aggregated deuterium uptake difference for one time point

---

Description

Function aggregates the differential deuterium uptake values from peptide level into single-amino resolution using 'weighted approach' (defined in 'vignette("datafiles")'). For visualization use `plot_aggregated_uptake`

Usage

```
calculate_aggregated_diff_uptake(diff_uptake_dat, time_t)
```

Arguments

`diff_uptake_dat`	differential uptake data, product of e.q. `create_diff_uptake_dataset`
`time_t`	chosen time point

Value

a data.frame object

Examples

```
diff_uptake_dat <- create_diff_uptake_dataset(alpha_dat)
calculate_aggregated_diff_uptake(diff_uptake_dat, time_t = 5)
```

---

calculate_aggregated_test_results

Aggregate test result

Description

Function aggregates peptide-level information into residue level. Significance method indicates if the difference is significant or not - if the number of significant peptides over this residue is bigger than the number of insignificant.

Usage

```
calculate_aggregated_test_results(
  p_diff_uptake_conf_dat,
  method = c("significance", "weiss"),
  time_t = 1,
  skip_amino = 1
)
```

Arguments

p_diff_uptake_conf_dat	uptake data with confidence, as produced by create_p_diff_uptake_dataset_with_confidence function
method	method of aggregation: significance or weiss method
time_t	chosen time point
skip_amino	integer, indicator how many amino acids from the N-terminus should be omitted in visualization

Details

Only peptides without modification are aggregated.

Value

a data.frame object

See Also

read_hdx

Examples

```
p_diff_uptake_dat <- create_p_diff_uptake_dataset(alpha_dat)
p_diff_uptake_conf_dat <- create_p_diff_uptake_dataset_with_confidence(p_diff_uptake_dat)
calculate_aggregated_test_results(p_diff_uptake_conf_dat, method = "significance")
calculate_aggregated_test_results(p_diff_uptake_conf_dat, method = "weiss")
```

---

calculate_aggregated_uptake

Calculates aggregated deuterium uptake for one time point

Description

Function aggregates the deuterium uptake values from peptide level into single-amino resolution using 'weighted approach' (defined in 'vignette("datafiles")'. For visualization use plot_aggregated_uptake

Usage

```
calculate_aggregated_uptake(
  kin_dat,
  state = unique(kin_dat[["State"]])[1],
  time_t
)
```

Arguments

kin_dat	single state uptake data, product of e.q. create_uptake_dataset
state	state included in calculations
time_t	chosen time point

Value

a data.frame object

Examples

```
# disabled due to long execution time

kin_dat <- create_uptake_dataset(alpha_dat, states = "Alpha_KSCN")
head(create_aggregated_uptake_dataset(kin_dat))
```

`calculate_auc` *Calculate Area Under the Curve*

Description

Calculates area under the deuterium uptake curve

Usage

```
calculate_auc(
  uptake_dat,
  protein = uptake_dat[["Protein"]][1],
  state = uptake_dat[["State"]][1],
  preserve_values = FALSE
)
```

Arguments

`uptake_dat`	data with deuterium uptake values, calculated e.q. by `create_uptake_dataset`
`protein`	chosen protein
`state`	state included in calculations
`preserve_values`	indicator if the original columns form uptake_dat should be preserve in the result

Details

The AUC is calculated on the data normalized to unit square by division by maximum values of exposure time and deuterium uptake, respectively.

Value

a `data.frame` object

See Also

`read_hdx` `create_uptake_dataset`

Examples

```
uptake_dat <- create_uptake_dataset(alpha_dat)
head(calculate_auc(uptake_dat))
```

`calculate_back_exchange` *Back exchange estimation*

Description

Calculates back-exchange for a state

Usage

```
calculate_back_exchange(
  dat,
  protein = dat[["Protein"]][1],
  states = unique(dat[["State"]]),
  time_100 = max(dat[["Exposure"]])
)
```

Arguments

`dat`	data imported by the `read_hdx` function
`protein`	selected protein
`states`	selected biological states for given protein
`time_100`	time point of measurement for fully deuterated sample

Details

Back-exchange is a reverse exchange phenomenon, important to acknowledge when working with HDX data. This function calculates back-exchange values for one biological state of the selected protein. For visualization of back-exchange data use `plot_coverage_heatmap` with displayed value specified as 'back-exchange'. For the definition of back-exchange see the 'vignette("datafiles")'.

Value

a `data.frame` object

See Also

`read_hdx` `plot_coverage_heatmap`

Examples

```
head(calculate_back_exchange(alpha_dat))
```

`calculate_confidence_limit_values` *Calculate the value of confidence limit*

Description

Calculates confidence limit values for prepared provided, based on chosen parameters.

Usage

```
calculate_confidence_limit_values(
  diff_uptake_dat,
  confidence_level = 0.98,
  theoretical = FALSE,
  fractional = TRUE,
  n_rep = NULL
)
```

Arguments

`diff_uptake_dat`	differential data calculated using calculate_diff_uptake function
`confidence_level`	confidence level for the test, from range [0, 1]
`theoretical`	`logical`, determines if values are theoretical
`fractional`	`logical`, determines if values are fractional
`n_rep`	number of replicates

Details

Function `calculate_confidence_limit_values` calculates confidence limit using Houde test. The confidence limits are calculated on whole provided dataset. If the user wishes to calculate confidence limit for one, two or more time points, the provided data should be adjusted accordingly.

Value

range of confidence limit interval

References

Houde, D., Berkowitz, S.A., and Engen, J.R. (2011). The Utility of Hydrogen/Deuterium Exchange Mass Spectrometry in Biopharmaceutical Comparability Studies. J Pharm Sci 100, 2071–2086.

See Also

`read_hdx` `calculate_diff_uptake` `create_diff_uptake_dataset`

Examples

```
diff_uptake_dat <- calculate_diff_uptake(alpha_dat)
calculate_confidence_limit_values(diff_uptake_dat)
```

`calculate_diff_uptake` *Calculate differential uptake*

Description

Calculates differential deuterium uptake between two selected biological states.

Usage

```
calculate_diff_uptake(
  dat,
  protein = unique(dat[["Protein"]][1]),
  states = unique(dat[["State"]])[1:2],
  time_0 = min(dat[["Exposure"]]),
  time_t = unique(dat[["Exposure"]])[3],
  time_100 = max(dat[["Exposure"]]),
  deut_part = 0.9
)
```

Arguments

`dat`	data imported by the `read_hdx` function
`protein`	chosen protein
`states`	vector of two states for chosen protein. Order is important, as the deuterium uptake difference is calculated as state_1 - state_2
`time_0`	minimal exchange control time point of measurement [min]
`time_t`	time point of the measurement for which the calculations are done [min]
`time_100`	maximal exchange control time point of measurement [min]
`deut_part`	deuterium percentage in solution used in experiment, value from range [0, 1]

Details

Function `calculate_diff_uptake` calculates differential values based on provided criteria for peptides for chosen protein in selected states. The methods of calculation of deuterium uptake difference, fractional deuterium uptake difference with respect to minimal/maximal exchange controls or theoretical tabular values are thoroughly described in the 'Data processing' article, as well as law of propagation of uncertainty, used to calculate uncertainty.

Value

a `data.frame` object.

See Also

read_hdx calculate_state_uptake

Examples

```
diff_dat <- calculate_diff_uptake(alpha_dat)
head(diff_dat)
```

`calculate_exp_masses` *Calculate measured mass, aggregated from the replicates of the experiment*

Description

Calculate the measured mass (with the uncertainty of the measurement) as aggregated data from the replicates of the experiment.

Usage

```
calculate_exp_masses(dat)
```

Arguments

`dat`	data as imported by the `read_hdx` function

Details

Each measurement is repeated at least three times to obtain reliable result and to calculate uncertainty of the measurement. For more information on how the data is aggregated or how the uncertainty is calculated, see the documentation.

Value

a `data.frame` object.

See Also

read_hdx calculate_exp_masses_per_replicate calculate_state_uptake

Examples

```
calculate_exp_masses(alpha_dat)
```

`calculate_exp_masses_per_replicate` *Calculate measured mass for each replicate of the experiment*

Description

Calculate the measured mass from partial results, per each replicate of the experiment.

Usage

```
calculate_exp_masses_per_replicate(dat)
```

Arguments

`dat`	data as imported by the `read_hdx` function

Details

Each replicate of the experiment generates measurements of the mass for obtained charge values for the peptide. This is an effect of the properties of mass spectrometry, that measures the mass to charge ratio (learn more about Mass Spectrometry in the documentation). The possible charge values depend on the sequence of the peptide. The separate measurement (for each replicate in given state in given time point) can be distinguished by the 'File' value.

Value

a `data.frame` object.

See Also

`read_hdx` `calculate_exp_masses` `calculate_state_uptake`

Examples

```
head(calculate_exp_masses_per_replicate(alpha_dat))
```

`calculate_kinetics` *Calculate kinetics data*

Description

Calculate kinetics of the hydrogen-deuteration exchange for given peptide in given state.

Usage

```
calculate_kinetics(
  dat,
  protein = dat[["Protein"]][1],
  sequence = dat[["Sequence"]][1],
  state = dat[["State"]][1],
  start = dat[["Start"]][1],
  end = dat[["End"]][1],
  time_0 = min(dat[["Exposure"]]),
  time_100 = max(dat[["Exposure"]]),
  deut_part = 0.9
)
```

Arguments

`dat`	dat data imported by the `read_hdx` function.
`protein`	protein chosen protein.
`sequence`	sequence of chosen peptide.
`state`	biological state of chosen peptide.
`start`	start position of chosen peptide.
`end`	end position of chosen peptide.
`time_0`	minimal exchange control time point of measurement.
`time_100`	maximal exchange control time point of measurement.
`deut_part`	deuterium percentage in solution used in experiment, value from range [0, 1].

Details

The function calculates deuteration data for all available data points for given peptide in chosen biological state.. All four variants (relative & theoretical combinations) of deuterium uptake computations are supported. Manual correction of percentage of deuterium the protein was exposed to during the exchange in theoretical calculations is provided. To visualize obtained data we recommend using `plot_uptake_curve` function. The first version doesn't support filled Modification and Fragment columns. IMPORTANT! The kinetic data is often described as deuterium uptake curve data. We use this terms interchangeable.

Value

a `data.frame` object.

See Also

read_hdx calculate_state_uptake plot_uptake_curve

Examples

```
# by default: for the first peptide
calculate_kinetics(alpha_dat)
```

`calculate_MHP` *Calculate MHP of the peptide*

Description

Calculate the mass of the singly charged monoisotopic (or not) molecular ion of for given peptide.

Usage

```
calculate_MHP(Sequence, mono = FALSE)
```

Arguments

Sequence	sequence of the peptide (string) or vector of sequences. Each letter of the sequence of the peptide represents different amino acid (three letter representation not allowed)
mono	logical value to determine if the mass should be monoisotopic or not. FALSE by default

Details

This function calculates the mass of the singly charged monoisotopic (or not) molecular ion for given peptide. It is the sum of the residue masses plus the masses of the terminating group (H and OH). The source of the masses can be found here: http://www.matrixscience.com/help/aa_help.html. Keep in mind that this function returns the value of an unmodified peptide.

Value

vector of numeric MHP values of provided Sequences

See Also

read_hdx calculate_state_uptake

Examples

```
calculate_MHP("CHERICHERILADY")
calculate_MHP("CHERICHERILADY", mono = TRUE)
```

`calculate_p_value` *Create p-value dataset*

Description

Create p-value dataset

Usage

```
calculate_p_value(
  dat,
  protein = unique(dat[["Protein"]])[1],
  state_1 = unique(dat[["State"]])[1],
  state_2 = unique(dat[["State"]])[2],
  p_adjustment_method = "none",
  confidence_level = 0.98
)
```

Arguments

`dat`	data imported by the `read_hdx` function.
`protein`	chosen protein.
`state_1`	biological state for chosen protein. From this state values the second state values are subtracted to get the deuterium uptake difference.
`state_2`	biological state for chosen protein. This state values are subtracted from the first state values to get the deuterium uptake difference.
`p_adjustment_method`	method of adjustment P-values for multiple comparisons. Possible methods: "BH" (Benjamini & Hochberg correction), "bonferroni" (Bonferroni correction) and "none" (default).
`confidence_level`	confidence level for the t-test.

Details

This function calculates P-value based on the supplied data. Unpaired t-Student test (with supplied parameters) is used to establish if the null hypothesis (there is no difference between measured mass values between two selected biological states) can be rejected, based on the experimental mass values from replicates of the experiment - for peptide in given time point of measurement. For the peptides that have only one replicate of the measurement (in any state) the P-value cannot be calculated and is assigned with NA value.

Value

a `data.frame` object.

See Also

read_hdx calculate_exp_masses_per_replicate plot_volcano create_diff_uptake_dataset create_p_diff_uptake_dataset

Examples

```
p_dat <- calculate_p_value(alpha_dat)
head(p_dat)
```

`calculate_peptide_kinetics` *Calculate kinetics dataset*

Description

Calculate kinetics of the hydrogen-deuteration exchange for given peptide in multiple biological states.

Usage

```
calculate_peptide_kinetics(
  dat,
  protein = dat[["Protein"]][1],
  sequence = dat[["Sequence"]][1],
  states = unique(dat[["State"]]),
  start = dat[["Start"]][1],
  end = dat[["End"]][1],
  time_0 = min(dat[["Exposure"]]),
  time_100 = max(dat[["Exposure"]]),
  deut_part = 0.9
)
```

Arguments

dat	dat data imported by the read_hdx function.
protein	chosen protein.
sequence	sequence of chosen peptide.
states	vector of states (for chosen protein), for which the calculations are done.
start	start position of chosen peptide.
end	end position of chosen peptide.
time_0	minimal exchange control time point of measurement.
time_100	maximal exchange control time point of measurement.
deut_part	deuterium percentage in solution used in experiment, value from range [0, 1].

Details

Function calculate_peptide_kinetics calculates kinetic data for chosen peptide in chosen biological states. It is a wrapper for calculate_kinetics but for multiple states. The output of this function can be visualized using plot_uptake_curve. IMPORTANT! The kinetic data is often described as deuterium uptake curve data. We use this terms interchangeable.

Value

a data.frame object.

See Also

calculate_kinetics calculate_state_uptake plot_uptake_curve

Examples

```
# by default calculated for the first peptide from the peptide pool
calculate_peptide_kinetics(alpha_dat)
```

`calculate_state_uptake` *Calculate deuterium uptake*

Description

Calculates deuteration uptake based on supplied parameters.

Usage

```
calculate_state_uptake(
  dat,
  protein = unique(dat[["Protein"]])[1],
  state = unique(dat[["State"]])[1],
  time_0 = min(dat[dat[["Exposure"]] > 0, ][["Exposure"]]),
  time_t = unique(dat[["Exposure"]])[3],
  time_100 = max(dat[["Exposure"]]),
  deut_part = 0.9
)
```

Arguments

dat	data as imported by the read_hdx function
protein	chosen protein
state	state included in calculations
time_0	minimal exchange control

time_t	chosen time point
time_100	maximal exchange control
deut_part	percentage of deuterium the protein was exposed to, value in range [0, 1]

Details

The function calculate_state_uptake calculates deuterium uptake (in different forms) for all of the peptides in given protein in given state based on supplied parameters: 'time_0', 'time_100' and 'time_t'. All four variants (fractional) are supplied (mean values and uncertainty). Manual correction of percentage of deuterium the protein was exposed to during the exchange in theoretical calculations is provided.

Methods of calculation and uncertainty are profoundly discussed in the vignette.

Value

a data.frame object

See Also

read_hdx create_uptake_dataset calculate_confidence_limit_values add_stat_dependency

Examples

```
head(calculate_state_uptake(alpha_dat))
```

---

create_aggregated_diff_uptake_dataset

Calculates aggregated uptake difference for peptide pool

Description

This is a wrapper function for calculate_aggregated_diff_uptake, which calculates aggregated differential uptake for only one time point. This function returns values for all time points from 'diff_uptake_dat'.

Usage

```
create_aggregated_diff_uptake_dataset(diff_uptake_dat)
```

Arguments

diff_uptake_dat	differential uptake data, product of e.q. create_diff_uptake_dataset

Value

a data.frame object

Examples

```
# disabled due to long execution time

diff_uptake_dat <- create_diff_uptake_dataset(alpha_dat)
head(create_aggregated_diff_uptake_dataset(diff_uptake_dat))
```

`create_aggregated_uptake_dataset` *Calculates the aggregated uptake for peptide pool*

Description

This is a wrapper function for `calculate_aggregated_uptake`, which calculates aggregated uptake for only one time point. This function returns values for all time points from 'kin_dat'.

Usage

```
create_aggregated_uptake_dataset(kin_dat)
```

Arguments

`kin_dat`	single state uptake data, product of e.q. `create_uptake_dataset`

Value

a `data.frame` object

Examples

```
# disabled due to long execution time

kin_dat <- create_uptake_dataset(alpha_dat, states = "Alpha_KSCN")
create_aggregated_uptake_dataset(kin_dat)
```

`create_control_dataset` *Create dataset with control*

Description

This function adds selected experimental maximal exchange control as a measurement for all biological states.

Usage

```
create_control_dataset(
  dat,
  control_protein = dat[["Protein"]][1],
  control_state = dat[["State"]][1],
  control_exposure = max(dat[["Exposure"]])
)
```

Arguments

`dat`	data imported by the `read_hdx` function.
`control_protein`	maximal exchange control protein, from dat.
`control_state`	maximal exchange control state, from dat.
`control_exposure`	maximal exchange control exposure (time point of measurement), from dat.

Details

Function `create_control_dataset` creates a dataset (similar to the output of `read_hdx` function), with maximal exchange control for all the states, based on provided parameters. The other functions are operating within a state, so the control is prepared for each state. The chosen maximal exchange control is distinguishable by the value '99999' in 'Exposure' control.

Value

a `data.frame` object.

See Also

`read_hdx` `calculate_state_uptake`

Examples

```
head(create_control_dataset(alpha_dat))
```

`create_diff_uptake_dataset` *Generate differential dataset*

Description

Calculates differential deuterium uptake values between two states.

Usage

```
create_diff_uptake_dataset(
  dat,
  protein = unique(dat[["Protein"]])[1],
  state_1 = unique(dat[["State"]])[1],
  state_2 = unique(dat[["State"]])[2],
  time_0 = min(dat[["Exposure"]]),
  time_100 = max(dat[["Exposure"]]),
  deut_part = 0.9
)
```

Arguments

`dat`	data imported by the `read_hdx` function.
`protein`	chosen protein.
`state_1`	biological state for chosen protein. From this state values the second state values are subtracted to get the deuterium uptake difference.
`state_2`	biological state for chosen protein. This state values are subtracted from the first state values to get the deuterium uptake difference.
`time_0`	minimal exchange control time point of measurement [min].
`time_100`	maximal exchange control time point of measurement [min].
`deut_part`	deuterium percentage in solution used in experiment, value from range [0, 1].

Details

The function `create_diff_uptake_dataset` generates a dataset with differential values between given biological states (state_1 - state_2). For each peptide of chosen protein for time points of measurement between minimal and maximal control time points of measurement deuterium uptake difference, fractional deuterium uptake difference with respect to controls or theoretical tabular values are calculated, with combined and propagated uncertainty. Each peptide has an ID, based on its start position. Function output can be visualized as a differential (Woods) plot, butterfly differential plot or chiclet differential plot.

Value

a `data.frame` object.

See Also

read_hdx calculate_diff_uptake plot_differential_butterfly plot_differential_chiclet plot_differential

Examples

```
diff_uptake_dat <- create_diff_uptake_dataset(alpha_dat)
head(diff_uptake_dat)
```

---

create_kinetic_dataset

Create kinetics dataset for a list of peptides and their states

---

Description

Generates the data set of deuterium uptake between selected time points based on supplied peptide list.

Usage

```
create_kinetic_dataset(
  dat,
  peptide_list,
  protein = dat[["Protein"]][1],
  time_0 = min(dat[["Exposure"]]),
  time_100 = max(dat[["Exposure"]]),
  deut_part = 0.9
)
```

Arguments

dat	dat data imported by the read_hdx function.
peptide_list	list of peptides for the calculation.
protein	chosen protein.
time_0	minimal exchange control time point of measurement.
time_100	maximal exchange control time point of measurement.
deut_part	deuterium percentage in solution used in experiment, value from range [0, 1].

Details

This is a wrapper for calculate_kinetics, but for the peptide list instead of one peptide.

Value

a data.frame object.

See Also

calculate_kinetics calculate_state_uptake plot_uptake_curve

Examples

```
peptide_list <- data.frame(Sequence = c("GFGDLKSPAGL", "FGDLKSPAGL"),
                           state = c("ALPHA_Gamma", "ALPHA_Gamma"),
                           start = c(1, 2), end = c(11, 11))
create_kinetic_dataset(alpha_dat, peptide_list)
```

---

create_overlap_distribution_dataset

Show overlap distribution data

Description

Generates the data of frequency of overlap of each amino in the protein sequence.

Usage

```
create_overlap_distribution_dataset(
  dat,
  protein = dat[["Protein"]][1],
  state = dat[["State"]][1],
  start = min(dat[["Start"]]),
  end = max(dat[["End"]]),
  protein_sequence = reconstruct_sequence(dat)
)
```

Arguments

dat	data imported by the read_hdx function.
protein	chosen protein.
state	biological state for chosen protein.
start	start position of chosen protein.
end	end position of chosen protein.
protein_sequence	data produced by reconstruct_sequence function.

Details

This data frame presents how many times (by how many peptides) a amino position in protein sequence is covered. This data is available in the GUI.

Value

a data.frame object.

See Also

read_hdx reconstruct_sequence

Examples

```
create_overlap_distribution_dataset(alpha_dat)
```

`create_p_diff_uptake_dataset` *Create differential uptake dataset with p-value*

Description

Creates differential deuterium uptake dataset with P-value from t-Student test for selected two biological states.

Usage

```
create_p_diff_uptake_dataset(
  dat,
  diff_uptake_dat = NULL,
  protein = unique(dat[["Protein"]])[1],
  state_1 = unique(dat[["State"]])[1],
  state_2 = unique(dat[["State"]])[2],
  p_adjustment_method = "none",
  confidence_level = 0.98,
  time_0 = min(dat[["Exposure"]]),
  time_100 = max(dat[["Exposure"]]),
  deut_part = 0.9
)
```

Arguments

dat	data imported by the read_hdx function.
diff_uptake_dat	differential uptake data
protein	chosen protein.
state_1	biological state for chosen protein. From this state values the second state values are subtracted to get the deuterium uptake difference.
state_2	biological state for chosen protein. This state values are subtracted from the first state values to get the deuterium uptake difference.

`p_adjustment_method`
: method of adjustment P-values for multiple comparisons. Possible methods: "BH" (Benjamini & Hochberg correction), "bonferroni" (Bonferroni correction) and "none" (default).

`confidence_level`
: confidence level for the t-test.

`time_0`
: minimal exchange control time point of measurement [min].

`time_100`
: maximal exchange control time point of measurement [min].

`deut_part`
: deuterium percentage in solution used in experiment, value from range [0, 1].

Details

For peptides in all of the time points of measurement (except for minimal and maximal exchange control) the deuterium uptake difference between state_1 and state_2 is calculated, with its uncertainty (combined and propagated as described in 'Data processing' article). For each peptide in time point the P-value is calculated using unpaired t-test. The deuterium uptake difference is calculated as the difference of measured masses in a given time point for two states. The tested hypothesis is that the mean masses for states from the replicates of the experiment are similar. The P-values indicates if the null hypothesis can be rejected - rejection of the hypothesis means that the difference between states is statistically significant at provided confidence level. The P-values can be adjusted using the provided method.

Value

a `data.frame` object with calculated deuterium uptake difference in different forms with their uncertainty, P-value and -log(P-value) for the peptides from the provided data.

References

Hageman, T. S. & Weis, D. D. Reliable Identification of Significant Differences in Differential Hydrogen Exchange-Mass Spectrometry Measurements Using a Hybrid Significance Testing Approach. Anal Chem 91, 8008–8016 (2019).

See Also

`read_hdx` `calculate_exp_masses_per_replicate`

Examples

```
p_diff_uptake_dat <- create_p_diff_uptake_dataset(alpha_dat)
head(p_diff_uptake_dat)
```

`create_p_diff_uptake_dataset_with_confidence` *Create differential dataset with statistical validity*

Description

Create differential dataset with statistical validity

Usage

```
create_p_diff_uptake_dataset_with_confidence(
  p_diff_uptake_dat,
  houde_interval = NULL,
  houde_interval_times = NULL,
  theoretical = FALSE,
  fractional = FALSE
)
```

Arguments

`p_diff_uptake_dat`	differential uptake data alongside the P-value as calculated by `create_p_diff_uptake_dataset`
`houde_interval`	interval value, as calculated by `calculate_confidence_limit_values`
`houde_interval_times`	specified time points to be used in calculation
`theoretical`	`logical`, determines if values are theoretical
`fractional`	`logical`, determines if values are fractional

Details

Function provides additional information on statistical relevance based on supplied data.

Value

a `data.frame` object.

See Also

`read_hdx` `create_p_diff_uptake_dataset` `calculate_confidence_limit_values`

Examples

```
p_diff_uptake_dat <- create_p_diff_uptake_dataset(alpha_dat)
p_diff_uptake_dat_confidence <- create_p_diff_uptake_dataset_with_confidence(p_diff_uptake_dat)
head(p_diff_uptake_dat_confidence)
```

`create_quality_control_dataset` *Experiment quality control*

Description

Checks how the uncertainty changes in a function of maximal exchange control time of measurement.

Usage

```
create_quality_control_dataset(
  dat,
  protein = dat[["Protein"]][1],
  state_1 = unique(dat[["State"]])[1],
  state_2 = unique(dat[["State"]])[2],
  time_t = unique(dat[["Exposure"]])[2],
  time_0 = min(dat[["Exposure"]]),
  deut_part = 0.9
)
```

Arguments

`dat`	data imported by the `read_hdx` function.
`protein`	chosen protein.
`state_1`	first biological state.
`state_2`	second biological state.
`time_t`	time point of the measurement for which the calculations are done.
`time_0`	minimal exchange control time point of measurement.
`deut_part`	deuterium percentage in solution used in experiment, value from range [0, 1].

Details

The function calculates mean uncertainty of all peptides and its uncertainty (standard error) based on given ‘time_0‘ and ‘time_t‘ as a function of ‘time_100‘. Both theoretical and experimental results for each state and their difference are supplied for comparison but only experimental calculations depends on ‘time_100‘ variable. The results are either in form of fractional or absolute values depending on the ‘fractional‘ parameter supplied by the user. This data can be useful for general overview of the experiment and analyze of the chosen time parameters.

Value

`data.frame` object with mean uncertainty per different ‘time_100‘ value. The values are shown as percentages.

See Also

read_hdx calculate_state_uptake plot_quality_control show_quality_control_data

Examples

```
create_quality_control_dataset(alpha_dat)
```

`create_replicate_dataset` *Create replicates data*

Description

Create replicate data set suitable for replicate histogram, for one or multiple time points of measurement.

Usage

```
create_replicate_dataset(
  dat,
  time_t = NULL,
  protein = unique(dat[["Protein"]])[1],
  state = dat[["State"]][1]
)
```

Arguments

dat	data as imported by the read_hdx function.
time_t	optional, for only one time point of measurement. If value is NULL, all time point from dat are preserved.
protein	chosen protein.
state	biological state for chosen protein.

Details

The function create_replicate_dataset creates a dataset with information about how many replicates are for peptides in specific state in time points of measurement.

Value

a data.frame object.

See Also

plot_replicate_histogram show_replicate_histogram_data

Examples

```
create_replicate_dataset(alpha_dat)
```

`create_state_comparison_dataset` *Creates comparison uptake dataset*

Description

Calculates deuterium uptake values for selected biological states in selected time point of measurements.

Usage

```
create_state_comparison_dataset(
  dat,
  protein = unique(dat[["Protein"]])[1],
  states = unique(dat[["State"]]),
  time_0 = min(dat[["Exposure"]]),
  time_t = unique(dat[["Exposure"]])[3],
  time_100 = max(dat[["Exposure"]]),
  deut_part = 0.9
)
```

Arguments

`dat`	data imported by the `read_hdx` function.
`protein`	chosen protein.
`states`	vector of states (for chosen protein), for which the calculations are done.
`time_0`	minimal exchange control time point of measurement [min].
`time_t`	time point of the measurement for which the calculations are done [min].
`time_100`	maximal exchange control time point of measurement [min].
`deut_part`	deuterium percentage in solution used in experiment, value from range [0, 1].

Details

Function `create_state_comparison_dataset` is a wrapper for `calculate_state_uptake` function, calls this function for all (default) or chosen states in states vector.

Value

a `data.frame` object.

See Also

read_hdx calculate_state_uptake

Examples

```
comparison_dat <- create_state_comparison_dataset(alpha_dat)
head(comparison_dat)
```

---

create_state_uptake_dataset

Create uptake dataset for chosen state

Description

Calculates deuterium uptake values for one biological state.

Usage

```
create_state_uptake_dataset(
  dat,
  protein = unique(dat[["Protein"]])[1],
  state = (dat[["State"]])[1],
  time_0 = min(dat[dat[["Exposure"]] > 0, ][["Exposure"]]),
  time_100 = max(dat[["Exposure"]]),
  deut_part = 0.9
)
```

Arguments

dat	data imported by the read_hdx function.
protein	chosen protein.
state	biological state for chosen protein.
time_0	minimal exchange control time point of measurement [min].
time_100	maximal exchange control time point of measurement [min].
deut_part	deuterium percentage in solution used in experiment, value from range [0, 1].

Details

The function create_uptake_dataset generates a dataset with deuterium uptake values in different forms. For each peptide in chosen protein in chosen state for time points of measurement between minimal and maximal control time points of measurement deuterium uptake, fractional deuterium uptake with respect to controls or theoretical tabular values are calculated, with combined and propagated uncertainty. Each peptide has an ID, based on its start position. This data can be presented in a form of comparison plot, butterfly plot or chiclet plot.

Value

a data.frame object.

See Also

read_hdx calculate_state_uptake

Examples

```
state_uptake_dat <- create_state_uptake_dataset(alpha_dat)
head(state_uptake_dat)
```

---

create_uptake_dataset *Create uptake dataset for multiple states*

Description

Calculates deuterium uptake values for selected biological states in multiple time points of measurements.

Usage

```
create_uptake_dataset(
  dat,
  protein = unique(dat[["Protein"]])[1],
  states = unique(dat[["State"]]),
  time_0 = min(dat[["Exposure"]]),
  time_100 = max(dat[["Exposure"]]),
  deut_part = 0.9
)
```

Arguments

dat	data imported by the read_hdx function.
protein	chosen protein.
states	list of biological states for chosen protein.
time_0	minimal exchange control time point of measurement [min].
time_100	maximal exchange control time point of measurement [min].
deut_part	deuterium percentage in solution used in experiment, value from range [0, 1].

Details

Function create_uptake_dataset generates a dataset with deuterium uptake values in different forms. For each peptide in chosen protein in chosen states for time points of measurement between minimal and maximal control time points of measurement deuterium uptake, fractional deuterium uptake with respect to controls or theoretical tabular values are calculated, with combined and propagated uncertainty. Each peptide has an ID, based on its start position. This function is a wrapper for create_state_uptake_dataset but for multiple states. The output of this function can be presented in a form of comparison plot.

Value

a data.frame object.

See Also

read_hdx calculate_state_uptake create_state_uptake_dataset

Examples

```
uptake_dat <- create_uptake_dataset(alpha_dat, states = c("Alpha_KSCN", "ALPHA_Gamma"))
head(uptake_dat)
```

---

get_n_replicates *Get number of replicates*

Description

Calculates the number of replicates from the experimental data.

Usage

```
get_n_replicates(dat, protein = dat[["Protein"]][1])
```

Arguments

dat	data imported by the read_hdx function.
protein	chosen protein

Details

Calculate the number of replicates of experiment.

Value

a numeric value.

See Also

read_hdx

Examples

```
get_n_replicates(alpha_dat)
```

---

get_peptide_sequence *Get peptide sequence based on the position*

Description

Gets the peptide sequence based on selected parameters (start and end position, modification).

Usage

```
get_peptide_sequence(
  dat,
  protein = dat[["Protein"]][1],
  start,
  end,
  modification = FALSE
)
```

Arguments

dat	any data frame that contains following information: protein, sequence, start, end, modification.
protein	chosen protein.
start	start position of the peptide of interest.
end	end position of the peptide of interest.
modification	logical value to indicate if peptide of interest has modification or not.

Details

Function returns peptide sequence for selected parameters. Peptide sequence is often required to properly identify peptide of interest, and to avoid mistakes sequence is returned by the function. Moreover, function uses the modification value to select peptide sequence.

Value

a character value.

See Also

read_hdx

Examples

```
get_peptide_sequence(alpha_dat, start = 1, end = 11)
```

`get_protein_coverage` *Get protein coverage*

Description

Calculate protein coverage by the peptides in selected biological state or states.

Usage

```
get_protein_coverage(
  dat,
  protein = dat[["Protein"]][1],
  states = unique(dat[["State"]]),
  protein_length = NULL
)
```

Arguments

`dat`	data imported by the `read_hdx` function
`protein`	selected protein
`states`	selected biological states
`protein_length`	`numeric`, indicates the length of the protein. If not provided, the maximal end value from the file is used.

Details

Function `get_protein_coverage` calculates the percentage coverage of the protein sequence, rounded to two decimal places.

Value

a `numeric` percentage value (rounded to two decimal places).

See Also

`read_hdx`

Examples

```
get_protein_coverage(alpha_dat)
get_protein_coverage(alpha_dat, protein_length = 230)
```

get_protein_redundancy *Get protein redundancy*

Description

Calculates the protein redundancy in the whole experiment (all biological states).

Usage

```
get_protein_redundancy(dat, protein_length = NULL)
```

Arguments

dat	data imported by the read_hdx function.
protein_length	numeric, indicates the length of the protein. If not provided, the maximal end value from the file is used.

Details

Function get_protein_redundancy calculates the redundancy of the protein, based on provided experimental data.

Value

a numeric value.

See Also

read_hdx

Examples

```
get_protein_redundancy(alpha_dat)
```

`get_replicate_list_sd` *Get replicates sd*

Description

Get list of peptides with their standard deviation.

Usage

```
get_replicate_list_sd(
  dat,
  protein = dat[["Protein"]][1],
  state = dat[["State"]][1],
  time_t = unique(dat[["Exposure"]])[3]
)
```

Arguments

`dat`	data as imported by the `read_hdx` function.
`protein`	chosen protein.
`state`	biological state for chosen protein.
`time_t`	time point of the measurement.

Details

Function gets the pepitde list in selected state with their standard deviation of measurement, calculated from the technical replicates. It is used for selection for measurement variability plot.

Value

a `data.frame` object.

See Also

`create_replicate_dataset`

Examples

```
get_replicate_list_sd(alpha_dat)
```

get_residue_positions *Get residue positions*

Description

Get residue positions

Usage

```
get_residue_positions(dat)
```

Arguments

dat	data imported by the read_hdx function.

Details

Prepares data table for high-resolution values. It creates a long data frame with each position in the sequence accompanied by the amino acid placed in this position.

Value

a data.frame object.

Examples

```
# disabled due to long execution time
get_residue_positions(alpha_dat)
```

get_structure_color *Get color palette for 3D structure*

Description

This function provides a set of color codes associated with aggregated values to be presented on 3D structure.

Usage

```
get_structure_color(
  aggregated_dat,
  differential = FALSE,
  fractional = TRUE,
  theoretical = FALSE,
  time_t
)
```

Arguments

aggregated_dat	aggregated data, either for single uptake or differential
differential	indicator if the aggregated_dat contains differential results
fractional	indicator if fractional values are used
theoretical	indicator if theoretical values are used
time_t	time point from aggregated_dat to be shown on the structure

Value

a list

Examples

```
# disabled due to long execution time

diff_uptake_dat <- create_diff_uptake_dataset(alpha_dat)
diff_aggregated_dat <- create_aggregated_diff_uptake_dataset(diff_uptake_dat)
get_structure_color(diff_aggregated_dat,
                    differential = TRUE,
                    time_t = 1)
```

HaDeX_GUI *HaDeX Graphical User Interface*

Description

Shows how to launch graphical user interface from HaDeXGUI package. If the GUI package is not installed, it asks user whether to install it.

Usage

```
HaDeX_GUI()
```

Value

No return value, called for side effects

Warning

Any ad-blocking software may cause malfunctions.

HaDeXify *HaDeX customized ggplot theme*

Description

This function HaDeXifies plot. It adds HaDeX logo and ggplot theme.

Usage

```
HaDeXify(plt)
```

Arguments

plt	ggplot object. Plot to HaDeXify.

Details

Function adds the logo of HaDeX package in the left down corner of the plot and the HaDeX theme.

Value

a ggplot object.

See Also

read_hdx plot_differential plot_butterfly

Examples

```
diff_uptake_dat <- create_diff_uptake_dataset(alpha_dat)
HaDeXify(plot_differential(diff_uptake_dat))
```

install_GUI *Installs GUI package from GitHub*

Description

Installs GUI package from GitHub

Usage

```
install_GUI()
```

Value

No return value, called for side effects

is_GUI_installed *Checks if GUI package is installed*

Description

Indicates presence or absence of HaDeXGUI package.

Usage

```
is_GUI_installed()
```

Value

logical value indicating availability of GUI package

plot_aggregated_differential_uptake *Plots aggregated uptake difference*

Description

Plots aggregated uptake difference

Usage

```
plot_aggregated_differential_uptake(
  aggregated_diff_dat,
  fractional = TRUE,
  theoretical = FALSE,
  time_100 = max(unique(aggregated_diff_dat[["Exposure"]])),
  panels = FALSE,
  interactive = FALSE
)
```

Arguments

aggregated_diff_dat	aggregated differential uptake data as calculated by create_aggregated_diff_uptake_dataset
fractional	logical, determines if values are fractional
theoretical	logical, determines if values are theoretical
time_100	maximal exchange control time point of measurement [min]
panels	logical, indicator if plot should be divided into panels or not
interactive	logical, whether plot should have an interactive layer created with with ggiraph, which would add tooltips to the plot in an interactive display (HTML/Markdown documents or shiny app)

Value

a ggplot object

See Also

create_diff_uptake_dataset create_aggregated_diff_uptake_dataset

Examples

```
# disabled due to long execution time

diff_uptake_dat <- create_diff_uptake_dataset(alpha_dat)
aggregated_diff_dat <- create_aggregated_diff_uptake_dataset(diff_uptake_dat)
plot_aggregated_differential_uptake(aggregated_diff_dat, panels = FALSE)
plot_aggregated_differential_uptake(aggregated_diff_dat, fractional = FALSE,
                                    theoretical = TRUE, panels = FALSE)
plot_aggregated_differential_uptake(aggregated_diff_dat, theoretical = TRUE,
                                    panels = TRUE)
```

---

plot_aggregated_uptake

Plots aggregated uptake

Description

Plots aggregated uptake

Usage

```
plot_aggregated_uptake(
  aggregated_dat,
  fractional = TRUE,
  theoretical = FALSE,
  time_100 = max(unique(aggregated_dat[["Exposure"]])),
  panels = FALSE,
  interactive = FALSE
)
```

Arguments

aggregated_dat	aggregated differential uptake data as calculated by create_aggregated_uptake_dataset
fractional	logical, determines if values are fractional
theoretical	logical, determines if values are theoretical
time_100	maximal exchange control time point of measurement [min]
panels	logical, indicator if plot should be divided into panels or not

interactive	logical, whether plot should have an interactive layer created with with ggiraph, which would add tooltips to the plot in an interactive display (HTML/Markdown documents or shiny app).

Value

a ggplot object

Examples

```
# disabled due to long execution time

kin_dat <- create_uptake_dataset(alpha_dat, states = "Alpha_KSCN")
aggregated_dat <- create_aggregated_uptake_dataset(kin_dat)
plot_aggregated_uptake(aggregated_dat, panels = FALSE)
plot_aggregated_uptake(aggregated_dat, fractional = FALSE, panels = FALSE)
plot_aggregated_uptake(aggregated_dat, fractional = FALSE, theoretical = TRUE, panels = TRUE)
```

---

plot_aggregated_uptake_structure

Plot aggregated uptake on the 3D structure

Description

Function plots the aggregated data (either one state deuterium uptake or differential deuterium uptake) on the 3d structure.

Usage

```
plot_aggregated_uptake_structure(
  aggregated_dat,
  differential = FALSE,
  fractional = TRUE,
  theoretical = FALSE,
  time_t,
  pdb_file_path
)
```

Arguments

aggregated_dat	aggregated data, either for single uptake or differential
differential	indicator if the aggregated_dat contains differential results
fractional	indicator if fractional values are used
theoretical	indicator if theoretical values are used
time_t	time point from aggregated_dat to be shown on the structure
pdb_file_path	file path to the pdb file

Value

a r3dmol object.

Examples

```
library(shiny)
# disabled due to its long time and producing 3rdmol object

diff_uptake_dat <- create_diff_uptake_dataset(alpha_dat)
diff_aggregated_dat <- create_aggregated_diff_uptake_dataset(diff_uptake_dat)
pdb_file_path <- system.file(package = "HaDeX2", "HaDeX/data/Model_eEF1Balpha.pdb")

plot_aggregated_uptake_structure(diff_aggregated_dat,
                                 differential = TRUE,
                                  time_t = 1,
                                  pdb_file_path = pdb_file_path)

kin_dat <- create_uptake_dataset(alpha_dat, states = "Alpha_KSCN" )
aggregated_dat <- create_aggregated_uptake_dataset(kin_dat)
plot_aggregated_uptake_structure(aggregated_dat,
                                  differential = FALSE,
                                  time_t = 1,
                                  pdb_file_path = pdb_file_path)
```

---

plot_amino_distribution

generate_amino_distribution

Description

Generates amino distribution based on the protein sequence and shows if the amino acid is hydrophobic or hydrophylic.

Usage

```
plot_amino_distribution(
  position_in_sequence,
  hydro_properties,
  protein,
  charge_colors,
  interactive = getOption("hadex_use_interactive_plots")
)
```

Arguments

position_in_sequence	custom format
hydro_properties	data with hydrophobic properties
protein	chosen protein
charge_colors	vector of desired colors
interactive	logical, whether plot should have an interactive layer created with with ggiraph, which would add tooltips to the plot in an interactive display (HTML/Markdown documents or shiny app).

Details

The data for this function is not packaged yet.

Value

a ggplot object.

---

plot_butterfly *Butterfly deuterium uptake plot*

Description

Butterfly plot of deuterium uptake values in time for one biological state.

Usage

```
plot_butterfly(
  uptake_dat,
  theoretical = FALSE,
  fractional = FALSE,
  uncertainty_type = "ribbon",
  interactive = getOption("hadex_use_interactive_plots")
)
```

Arguments

uptake_dat	data produced by create_state_uptake_dataset function.
theoretical	logical, determines if values are theoretical.
fractional	logical, determines if values are fractional.
uncertainty_type	type of presenting uncertainty, possible values: "ribbon", "bars" or "bars + line".
interactive	logical, whether plot should have an interactive layer created with with ggiraph, which would add tooltips to the plot in an interactive display (HTML/Markdown documents or shiny app).

Details

Function plot_butterfly generates butterfly plot based on provided data and parameters. On X-axis there is peptide ID. On the Y-axis there is deuterium uptake in chosen form. Data from multiple time points of measurement is presented.

Value

a ggplot object.

See Also

create_state_uptake_dataset

Examples

```
state_uptake_dat <- create_state_uptake_dataset(alpha_dat)
plot_butterfly(state_uptake_dat)
```

---

plot_chiclet *Chiclet deuterium uptake plot*

Description

Chiclet plot of deuterium uptake values in time for one biological state.

Usage

```
plot_chiclet(
  uptake_dat,
  theoretical = FALSE,
  fractional = FALSE,
  show_uncertainty = FALSE,
  interactive = getOption("hadex_use_interactive_plots")
)
```

Arguments

uptake_dat	produced by create_state_uptake_dataset function.
theoretical	logical, determines if values are theoretical.
fractional	logical, determines if values are fractional.
show_uncertainty	logical, determines if the uncertainty is shown.
interactive	logical, whether plot should have an interactive layer created with with ggiraph, which would add tooltips to the plot in an interactive display (HTML/Markdown documents or shiny app).

Details

Function plot_chiclet produces a chiclet plot based on the same dataset as butterfly plot, as it is the different form of presenting the same data. On X-axis there is a peptide ID. On Y-axis are time points of measurement. Each tile for a peptide in time has a color value representing the deuterium uptake, in a form based on provided criteria (e.q. fractional). Each tile has a plus sign, which size represent the uncertainty of measurement for chosen value.

Value

a ggplot object.

See Also

create_state_uptake_dataset

Examples

```
state_uptake_dat <- create_state_uptake_dataset(alpha_dat)
plot_chiclet(state_uptake_dat)
```

---

plot_coverage *Peptide coverage*

Description

Plot the peptide coverage of the protein sequence

Usage

```
plot_coverage(
  dat,
  protein = dat[["Protein"]][1],
  states = NULL,
  show_blanks = TRUE,
  interactive = getOption("hadex_use_interactive_plots")
)
```

Arguments

dat	data imported by the read_hdx function
protein	selected protein
states	selected biological states for given protein
show_blanks	logical, indicator if the non-covered regions of the sequence are indicated in red
interactive	logical, whether plot should have an interactive layer created with with ggiraph, which would add tooltips to the plot in an interactive display (HTML/Markdown documents or shiny app)

Details

The function plot_coverage generates sequence coverage plot based on experimental data for selected protein in selected biological states. Only non-duplicated peptides are shown on the plot, next to each other.

The aim of this plot is to see the dependence between position of the peptide on the protein sequence. Their position on y-axis does not contain any information.

Value

a ggplot object

See Also

read_hdx plot_position_frequency

Examples

```
plot_coverage(alpha_dat)
plot_coverage(alpha_dat, show_blanks = FALSE)

diff_uptake_dat <- create_diff_uptake_dataset(alpha_dat)
plot_coverage(diff_uptake_dat)
```

---

plot_coverage_heatmap *Coverage heatmap*

Description

Coverage heatmap with color indicating specific value

Usage

```
plot_coverage_heatmap(
  x_dat,
  protein = x_dat[["Protein"]][1],
  state = NULL,
  value = NULL,
  time_t = NULL,
  interactive = getOption("hadex_use_interactive_plots")
)
```

Arguments

x_dat	data created using calculate_ or create_ function
protein	selected protein
state	selected biological state

value	value to be presented
time_t	chosen time point
interactive	logical, whether plot should have an interactive layer created with with ggiraph, which would add tooltips to the plot in an interactive display (HTML/Markdown documents or shiny app)

Details

Plots standard protein coverage but colored with desired value.

Value

a ggplot object

See Also

read_hdx plot_coverage

Examples

```
uptake_dat <- create_uptake_dataset(alpha_dat, states = "Alpha_KSCN")
plot_coverage_heatmap(uptake_dat)
plot_coverage_heatmap(x_dat = uptake_dat, value = "frac_deut_uptake", time_t = 0.167)
plot_coverage_heatmap(uptake_dat, value = "err_frac_deut_uptake", time_t = 0.167)

diff_uptake_dat <- create_diff_uptake_dataset(alpha_dat)
plot_coverage_heatmap(diff_uptake_dat)
plot_coverage_heatmap(diff_uptake_dat, value = "diff_frac_deut_uptake")
plot_coverage_heatmap(diff_uptake_dat, value = "diff_frac_deut_uptake", time_t = 0.167)
plot_coverage_heatmap(diff_uptake_dat, value = "err_diff_frac_deut_uptake", time_t = 0.167)

auc_dat <- calculate_auc(create_uptake_dataset(alpha_dat))
plot_coverage_heatmap(auc_dat, value = "auc")

bex_dat <- calculate_back_exchange(alpha_dat, state = "Alpha_KSCN")
plot_coverage_heatmap(bex_dat, value = "back_exchange")
```

---

plot_differential *Differential plot*

Description

Woods plot of differential deuterium uptake values between two biological states in time point.

Usage

```
plot_differential(
  diff_uptake_dat = NULL,
  diff_p_uptake_dat = NULL,
  skip_amino = 0,
  time_t = NULL,
  theoretical = FALSE,
  fractional = FALSE,
  show_houde_interval = FALSE,
  hide_houde_insignificant = FALSE,
  show_tstud_confidence = FALSE,
  hide_tstud_insignificant = FALSE,
  confidence_level = 0.98,
  all_times = FALSE,
  line_size = 1.5,
  interactive = getOption("hadex_use_interactive_plots")
)
```

Arguments

diff_uptake_dat	produced by calculate_diff_uptake or create_diff_uptake_dataset function.
diff_p_uptake_dat	produced by create_p_diff_uptake_dataset function.
skip_amino	integer, indicator how many amino acids from the N-terminus should be omitted
time_t	time point of measurement, if only one should be displayed on the plot.
theoretical	logical, determines if values are theoretical.
fractional	logical, determines if values are fractional.
show_houde_interval	logical, determines if houde interval is shown.
hide_houde_insignificant	logical, determines if statistically insignificant values using Houde test are hidden on the plot.
show_tstud_confidence	logical, determines if t-Student test validity is shown.
hide_tstud_insignificant	logical, determines if statistically insignificant values using t-Student test are hidden on the plot.
confidence_level	confidence level for the test, from range [0, 1].
all_times	logical, determines if all the time points from the supplied data should be displayed on the plots next to each other.
line_size	line size of the lines displayed on the plot.

`interactive` `logical`, whether plot should have an interactive layer created with with ggiraph, which would add tooltips to the plot in an interactive display (HTML/Markdown documents or shiny app).

Details

Function `plot_differential` presents provided data in a form of differential (Woods) plot. The plot shows difference in exchange for two biological states, selected in generation of dataset at one time point of measurement. On X-axis there is a position in a sequence, with length of a segment of each peptide representing its length. On Y-axis there is deuterium uptake difference in chosen form. Error bars represents the combined and propagated uncertainty. For Woods Plot there is available Houde test and t-Student test to see the statistically significant peptides. Selecting both of them simultaneously results in hybrid testing, as described in Weis et al. The statistically significant values are in color (red if the difference is positive, blue if negative), and the insignificant values are grey.

Value

a [ggplot2::ggplot()] object.

References

Hageman, T. S. & Weis, D. D. Reliable Identification of Significant Differences in Differential Hydrogen Exchange-Mass Spectrometry Measurements Using a Hybrid Significance Testing Approach. Anal Chem 91, 8008–8016 (2019).

See Also

`create_diff_uptake_dataset` `calculate_diff_uptake` `show_diff_uptake_data`

Examples

```
# disabled due to long execution time

diff_uptake_dat <- calculate_diff_uptake(alpha_dat,
                                 states = c("ALPHA_Gamma", "Alpha_KSCN"), time_t = 0.167)
plot_differential(diff_uptake_dat = diff_uptake_dat, time_t = 0.167)
plot_differential(diff_uptake_dat = diff_uptake_dat, time_t = 0.167, show_houde_interval = TRUE)
plot_differential(diff_uptake_dat = diff_uptake_dat, time_t = 0.167, skip_amino = 0)
plot_differential(diff_uptake_dat = diff_uptake_dat, time_t = 0.167, line_size = 1)
plot_differential(diff_uptake_dat = diff_uptake_dat, all_times = TRUE)
plot_differential(diff_uptake_dat = diff_uptake_dat, all_times = TRUE, show_houde_interval = TRUE)
plot_differential(diff_uptake_dat = diff_uptake_dat, all_times = TRUE, show_houde_interval = TRUE,
                  hide_houde_insignificant = TRUE)

diff_p_uptake_dat <- create_p_diff_uptake_dataset(alpha_dat, state_1 = "Alpha_KSCN",
                                                  state_2 = "ALPHA_Gamma")
plot_differential(diff_p_uptake_dat = diff_p_uptake_dat, all_times = TRUE,
                  show_tstud_confidence = TRUE)
plot_differential(diff_p_uptake_dat = diff_p_uptake_dat, all_times = FALSE,
                time_t = 0.167, show_tstud_confidence = TRUE, show_houde_interval = TRUE)
```

```
plot_differential(diff_p_uptake_dat = diff_p_uptake_dat, show_tstud_confidence = TRUE,
                  show_houde_interval = TRUE, all_times = FALSE)
plot_differential(diff_p_uptake_dat = diff_p_uptake_dat, show_tstud_confidence = TRUE,
            show_houde_interval = TRUE, all_times = FALSE, hide_houde_insignificant = TRUE)
plot_differential(diff_p_uptake_dat = diff_p_uptake_dat, show_tstud_confidence = TRUE,
            show_houde_interval = TRUE, all_times = FALSE, hide_houde_insignificant = TRUE,
                  hide_tstud_insignificant = TRUE)
```

`plot_differential_butterfly` *Butterfly differential deuterium uptake plot*

Description

Butterfly plot of differential deuterium uptake values between two biological states in time.

Usage

```
plot_differential_butterfly(
  diff_uptake_dat = NULL,
  diff_p_uptake_dat = NULL,
  theoretical = FALSE,
  fractional = FALSE,
  show_houde_interval = FALSE,
  show_tstud_confidence = FALSE,
  uncertainty_type = "ribbon",
  confidence_level = 0.98,
  interactive = getOption("hadex_use_interactive_plots")
)
```

Arguments

`diff_uptake_dat`	data produced by `calculate_diff_uptake` or `create_diff_uptake_dataset` function
`diff_p_uptake_dat`	differential uptake data alongside the P-value as calculated by `create_p_diff_uptake_dataset`
`theoretical`	`logical`, determines if values are theoretical
`fractional`	`logical`, determines if values are fractional
`show_houde_interval`	`logical`, determines if houde interval is shown
`show_tstud_confidence`	`logical`, determines if t-Student test validity is shown
`uncertainty_type`	type of presenting uncertainty, possible values: "ribbon", "bars" or "bars + line"

confidence_level
: confidence level for the test, from range [0, 1] Important if selected show_confidence_limit

interactive
: logical, whether plot should have an interactive layer created with with ggiraph, which would add tooltips to the plot in an interactive display (HTML/Markdown documents or shiny app).

Details

Function plot_differential_butterfly generates differential butterfly plot based on provided data and parameters. On X-axis there is peptide ID. On the Y-axis there is deuterium uptake difference in chosen form. Data from multiple time points of measurement is presented. If chosen, there are confidence limits based on Houde test on provided confidence level.

Value

a [ggplot2::ggplot()] object.

References

Houde, D., Berkowitz, S.A., and Engen, J.R. (2011). The Utility of Hydrogen/Deuterium Exchange Mass Spectrometry in Biopharmaceutical Comparability Studies. J Pharm Sci 100, 2071–2086.

See Also

read_hdx create_diff_uptake_dataset calculate_diff_uptake

Examples

```
diff_uptake_dat <- create_diff_uptake_dataset(alpha_dat)
plot_differential_butterfly(diff_uptake_dat = diff_uptake_dat)
plot_differential_butterfly(diff_uptake_dat = diff_uptake_dat, show_houde_interval = TRUE)

diff_p_uptake_dat <- create_p_diff_uptake_dataset(alpha_dat)
plot_differential_butterfly(diff_p_uptake_dat = diff_p_uptake_dat, show_tstud_confidence = TRUE)
```

---

plot_differential_chiclet

Chiclet differential deuterium uptake plot

Description

Chiclet plot of differential deuterium uptake values between two biological states in time.

Usage

```
plot_differential_chiclet(
  diff_uptake_dat = NULL,
  diff_p_uptake_dat = NULL,
  theoretical = FALSE,
  fractional = FALSE,
  show_houde_interval = FALSE,
  show_tstud_confidence = FALSE,
  confidence_level = 0.98,
  show_uncertainty = FALSE,
  interactive = getOption("hadex_use_interactive_plots")
)
```

Arguments

diff_uptake_dat	data produced by calculate_diff_uptake or create_diff_uptake_dataset function
diff_p_uptake_dat	differential uptake data alongside the P-value as calculated by create_p_diff_uptake_dataset
theoretical	logical, determines if values are theoretical
fractional	logical, determines if values are fractional
show_houde_interval	logical, determines if houde interval is shown
show_tstud_confidence	logical, determines if t-Student test validity is shown
confidence_level	confidence level for the test, from range [0, 1] Important if selected show_confidence_limit
show_uncertainty	logical, determines if the uncertainty is shown
interactive	logical, whether plot should have an interactive layer created with with ggiraph, which would add tooltips to the plot in an interactive display (HTML/Markdown documents or shiny app).

Details

Function plot_differential_chiclet generates chiclet differential plot based on provided data and parameters. On X-axis there is a peptide ID. On Y-axis are time points of measurement. Each tile for a peptide in time has a color value representing the deuterium uptake difference between chosen states, in a form based on provided criteria (e.q. fractional). Each tile has a plus sign, which size represent the uncertainty of measurement for chosen value.

Value

a [ggplot2::ggplot()] object.

See Also

create_diff_uptake_dataset calculate_diff_uptake

Examples

```
diff_uptake_dat <- create_diff_uptake_dataset(alpha_dat)
plot_differential_chiclet(diff_uptake_dat)
plot_differential_chiclet(diff_uptake_dat, show_houde_interval = TRUE)

diff_p_uptake_dat <- create_p_diff_uptake_dataset(alpha_dat)
plot_differential_chiclet(diff_p_uptake_dat = diff_p_uptake_dat,
                          show_tstud_confidence = TRUE)
plot_differential_chiclet(diff_p_uptake_dat = diff_p_uptake_dat,
                          show_tstud_confidence = TRUE, show_houde_interval = TRUE)
```

---

plot_differential_uptake_curve

Plot differential uptake curve

Description

Differential uptake curve for one peptide between two biological states.

Usage

```
plot_differential_uptake_curve(
  diff_uptake_dat = NULL,
  diff_p_uptake_dat = NULL,
  sequence = NULL,
  theoretical = FALSE,
  fractional = FALSE,
  uncertainty_type = "ribbon",
  log_x = TRUE,
  show_houde_interval = FALSE,
  show_tstud_confidence = FALSE,
  interactive = getOption("hadex_use_interactive_plots")
)
```

Arguments

diff_uptake_dat	produced by calculate_diff_uptake or create_diff_uptake_dataset function
diff_p_uptake_dat	differential uptake data alongside the P-value as calculated by create_p_diff_uptake_dataset
sequence	sequence of chosen peptide

`theoretical` `logical`, determines if plot shows theoretical values

`fractional` `logical`, determines if plot shows fractional values

`uncertainty_type` type of presenting uncertainty, possible values: "ribbon", "bars" or "bars + line"

`log_x` `logical`, determines if x axis shows logarithmic values

`show_houde_interval` `logical`, determines if houde interval is shown

`show_tstud_confidence` `logical`, determines if t-Student test validity is shown

`interactive` `logical`, whether plot should have an interactive layer created with with ggiraph, which would add tooltips to the plot in an interactive display (HTML/Markdown documents or shiny app).

Details

This plot shows the differential deuterium uptake between two biological states for selected peptides in different time points. The possibility to plot multiple differences (between state and mutant) for the peptide will be added soon.

Value

a `ggplot` object.

See Also

`read_hdx` `create_diff_uptake_dataset` `calculate_diff_uptake`

Examples

```
diff_uptake_dat <- create_diff_uptake_dataset(alpha_dat)
plot_differential_uptake_curve(diff_uptake_dat = diff_uptake_dat, sequence = "GDLKSPAGL")

diff_p_uptake_dat <- create_p_diff_uptake_dataset(alpha_dat)
plot_differential_uptake_curve(diff_p_uptake_dat = diff_p_uptake_dat,
                               sequence = "GDLKSPAGL", show_houde_interval = TRUE)
plot_differential_uptake_curve(diff_p_uptake_dat = diff_p_uptake_dat,
                               sequence = "GDLKSPAGL", show_houde_interval = TRUE,
                               show_tstud_confidence = TRUE)
plot_differential_uptake_curve(diff_p_uptake_dat = diff_p_uptake_dat,
                               sequence = "GDLKSPAGL", show_tstud_confidence = TRUE)
```

plot_manhattan *Manhattan plot*

Description

Manhattan plot with p-values from the t-Student test and peptide position.

Usage

```
plot_manhattan(
  p_dat,
  skip_amino = 0,
  plot_title = NULL,
  separate_times = TRUE,
  times = NULL,
  confidence_level = NULL,
  show_confidence_limit = TRUE,
  show_peptide_position = FALSE,
  interactive = getOption("hadex_use_interactive_plots")
)
```

Arguments

p_dat	data produced by the create_p_diff_uptake_dataset function.
skip_amino	integer, indicator how many amino acids from the N-terminus should be omitted
plot_title	title for the plot. If not provided, it is constructed in a form: "Differences between state_1 and state_2"
separate_times	logical, indicates if the data should be seen on the same plot, or on separate plots for each time point of measurement.
times	vector of time points of measurements to be included in the plot.
confidence_level	confidence level for the test, from range [0, 1].
show_confidence_limit	logical, indicates if the hybrid testing confidence intervals are shown.
show_peptide_position	logical, indicates if the peptide length and position in the sequence is shown. Otherwise, the peptides are represented by their ID.
interactive	logical, whether plot should have an interactive layer created with with ggiraph, which would add tooltips to the plot in an interactive display (HTML/Markdown documents or shiny app).

Details

The manhattan plot presents the P-values from t-student test, to see the regions of the protein with statistically significant changes between two biological states. On X-axis there is a position in a sequence, with length of a segment of each peptide representing its length. On Y-axis there is P-value from t-Student test.

Value

a ggplot object.

References

Hageman, T. S. & Weis, D. D. Reliable Identification of Significant Differences in Differential Hydrogen Exchange-Mass Spectrometry Measurements Using a Hybrid Significance Testing Approach. Anal Chem 91, 8008–8016 (2019).

See Also

read_hdx create_p_diff_uptake_dataset

Examples

```
p_dat <- create_p_diff_uptake_dataset(alpha_dat)
plot_manhattan(p_dat)
plot_manhattan(p_dat, separate_times = FALSE)
plot_manhattan(p_dat, show_peptide_position = TRUE, separate_times = FALSE)
plot_manhattan(p_dat, separate_times = FALSE, show_confidence_limit = FALSE)
```

---

plot_overlap *Plot overlap data*

Description

Generates overlapping peptide plot based on supplied data and parameters.

Usage

```
plot_overlap(dat, protein = dat[["Protein"]][1], state = dat[["State"]][1])
```

Arguments

dat	data imported by the read_hdx function.
protein	protein included in calculations
state	state included in calculations

Details

The overlap plot presents all the peptides in given state on the protein sequence. This plot is visible in GUI.

Value

a ggplot object.

See Also

read_hdx show_overlap_data

Examples

```
plot_overlap(alpha_dat)
```

`plot_overlap_distribution` *Plot overlap distribution*

Description

Generates overlap distribution plot based on supplied data and parameters.

Usage

```
plot_overlap_distribution(
  overlap_dist_dat,
  start = 1,
  end = max(overlap_dist_dat[["pos"]]),
  interactive = getOption("hadex_use_interactive_plots")
)
```

Arguments

`overlap_dist_dat`	produced by `create_overlap_distribution_dataset` function
`start`	start start position of chosen protein.
`end`	end position of chosen protein.
`interactive`	`logical`, whether plot should have an interactive layer created with with ggiraph, which would add tooltips to the plot in an interactive display (HTML/Markdown documents or shiny app).

Details

This plot presents how many times (by how many peptides) a amino position in protein sequence is covered. This plot is visible in GUI.

Value

a ggplot object.

See Also

read_hdx reconstruct_sequence create_overlap_distribution_dataset

Examples

```
overlap_dist_dat <- create_overlap_distribution_dataset(alpha_dat)
plot_overlap_distribution(overlap_dist_dat)
```

---

plot_peptide_charge_measurement

Plot peptide charge measurement

Description

Plot the charge measurements from replicates for peptide in specific time point.

Usage

```
plot_peptide_charge_measurement(
  dat,
  protein = dat[["Protein"]][1],
  state = dat[["State"]][1],
  sequence = dat[["Sequence"]][1],
  time_t = unique(dat[["Exposure"]])[3]
)
```

Arguments

dat	data as imported by the read_hdx function.
protein	chosen protein.
state	biological state for chosen protein.
sequence	sequence of chosen peptide.
time_t	time point of the measurement.

Details

This function shows the measurements of charge from different replicates for specific peptide in specific state in specific time point of measurement on the plot.

Value

a [ggplot2::ggplot()] object.

See Also

read_hdx show_peptide_charge_measurement

Examples

```
plot_peptide_charge_measurement(alpha_dat)
```

---

plot_peptide_mass_measurement

Plot peptide mass measurement

Description

Plot the mass measurements from replicates for peptide in specific time point.

Usage

```
plot_peptide_mass_measurement(
  dat,
  protein = dat[["Protein"]][1],
  state = dat[["State"]][1],
  sequence = dat[["Sequence"]][1],
  show_charge_values = TRUE,
  time_t = unique(dat[["Exposure"]])[3],
  interactive = getOption("hadex_use_interactive_plots")
)
```

Arguments

dat	data produced by read_hdx function
protein	chosen protein
state	biological state for chosen protein
sequence	sequence of chosen peptide
show_charge_values	logical, indicator if the data is shown for replicates or for charge values within the replicates
time_t	chosen time point
interactive	logical, whether plot should have an interactive layer created with with ggiraph, which would add tooltips to the plot in an interactive display (HTML/Markdown documents or shiny app)

Details

This function shows the measurements of mass from different replicates for specific peptide in specific state in specific time point of measurement on the plot. Moreover, on the plot is shown the average mass from the replicates, used later in calculations. The ribbon next to the dotted average mass indicates the uncertainty.

Value

a [ggplot2::ggplot()] object.

See Also

read_hdx calculate_exp_masses_per_replicate calculate_exp_masses calculate_state_uptake calculate_diff_uptake

Examples

```
plot_peptide_mass_measurement(alpha_dat, sequence = "FGSDDEEESEEAKRLRE")
plot_peptide_mass_measurement(alpha_dat, sequence = "FGSDDEEESEEAKRLRE", show_charge_values = FALSE)
```

---

plot_position_frequency

Position frequency

Description

Plot the frequency of coverage for protein sequence

Usage

```
plot_position_frequency(
  dat,
  protein = dat[["Protein"]][1],
  state = dat[["State"]][1]
)
```

Arguments

dat	data as imported by the read_hdx function
protein	selected protein
state	selected biological state for given protein

Details

The function plot_position_frequency generates a histogram of the coverage frequency in selected biological states for selected protein. The position frequency plot presents how many times each position of the sequence is covered by different peptides.

Value

a ggplot object.

See Also

read_hdx plot_coverage

Examples

```
plot_position_frequency(alpha_dat)
```

---

plot_quality_control *Plot quality control data*

Description

Generates quality control plot based on supplied data.

Usage

```
plot_quality_control(qc_dat)
```

Arguments

qc_dat	data produced by create_quality_control_dataset function, scaled if necessary.

Details

This plot presents the mean uncertainty in function of selected maximal exchange control time of measurement. This plot is visible in GUI.

Value

a [ggplot2::ggplot()] object.

See Also

create_quality_control_dataset show_quality_control_data

Examples

```
qc_dat <- create_quality_control_dataset(alpha_dat)
plot_quality_control(qc_dat)
```

`plot_replicate_histogram` *Plot replicates histogram*

Description

Plot histogram on number of replicates per peptide in one or multiple time point of measurement.

Usage

```
plot_replicate_histogram(
  rep_dat,
  time_points = FALSE,
  interactive = getOption("hadex_use_interactive_plots")
)
```

Arguments

`rep_dat`	replicate data, created by `create_replicate_dataset` function.
`time_points`	`logical`, indicator if the histogram should show values aggregated for time points of measurements.
`interactive`	`logical`, whether plot should have an interactive layer created with with ggiraph, which would add tooltips to the plot in an interactive display (HTML/Markdown documents or shiny app).

Details

The function shows three versions of replicate histogram, based on supplied `rep_dat` and `time_points`. If `time_points` is selected, the histogram shows the number of replicates for time points of measurement, to spot if there were troubles with samples for specific time point of measurement. Then, on the X-axis is Exposure (in minutes) and on the Y-axis number of replicates. If `time_points` is not selected, on the X-axis there are peptide ID, and on the Y-axis there are numbers of replicates. If `rep_dat` contains data from one time point of measurement, the histogram colors reflect the number of replicates to highlight the outliers. If `rep_dat` contains multiple time point of measurement, the colors help to distinguish between them.

Value

a [ggplot2::ggplot()] object.

See Also

`create_replicate_dataset` `show_replicate_histogram_data`

Examples

```
rep_dat <- create_replicate_dataset(alpha_dat)
plot_replicate_histogram(rep_dat)

plot_replicate_histogram(rep_dat, time_points = TRUE)

rep_dat <- create_replicate_dataset(alpha_dat, time_t = 0.167)
plot_replicate_histogram(rep_dat)
```

---

plot_replicate_mass_uptake

Replicate mass uptake curve

Description

Plot the mass uptake curve for selected peptide to see the difference between replicates.

Usage

```
plot_replicate_mass_uptake(
  dat,
  protein = dat[["Protein"]][1],
  state = dat[["State"]][1],
  sequence = dat[["Sequence"]][1],
  aggregated = FALSE,
  log_x = TRUE,
  interactive = getOption("hadex_use_interactive_plots")
)
```

Arguments

dat	data imported by the read_hdx function
protein	selected protein
state	selected biological state for given protein
sequence	selected peptide sequence for given protein in given biological state
aggregated	logical, indicator if presented data is aggregated on replicate level
log_x	logical, indicator if the X axis values are transformed to log10
interactive	logical, whether plot should have an interactive layer created with with ggiraph, which would add tooltips to the plot in an interactive display (HTML/Markdown documents or shiny app)

Details

The function plot_replicate_mass_uptake generates a plot showing the mass uptake for selected protein in replicates of the experiments. The values can be presented in two ways: as aggregated values for each replicate, or before aggregation - measured values for charge values within a replicate. The mass uptake is generated by subtracting the MHP mass of a peptide from measured mass and the mass uptake is presented in Daltons.

Value

a ggplot object.

See Also

read_hdx calculate_exp_masses_per_replicate

Examples

```
plot_replicate_mass_uptake(alpha_dat)
plot_replicate_mass_uptake(alpha_dat, aggregated = TRUE)
```

---

plot_state_comparison *State deuterium uptake comparison*

Description

Comparison plot of deuterium uptake values in time point for biological states.

Usage

```
plot_state_comparison(
  uptake_dat,
  skip_amino = 0,
  theoretical = FALSE,
  fractional = FALSE,
  line_size = 1.5,
  all_times = FALSE,
  time_t = NULL,
  interactive = getOption("hadex_use_interactive_plots")
)
```

Arguments

uptake_dat	data produced by calculate_state_uptake function
skip_amino	integer, indicator how many amino acids from the N-terminus should be omitted
theoretical	logical, indicator if values are calculated using theoretical controls

fractional	logical, indicator if values are shown in fractional form
line_size	line size for the plot
all_times	logical, indicator is all all time should be plotted next to each other
time_t	chosen time point
interactive	logical, whether plot should have an interactive layer created with with ggiraph, which would add tooltips to the plot in an interactive display (HTML/Markdown documents or shiny app)

Details

The function plot_state_comparison generates a comparison plot, presenting deuterium uptake values of peptides from selected protein in selected biological states at one time point of measurement at once. On X-axis there is a position in a sequence, with length of a segment of each peptide representing its length. On Y-axis there is deuterium uptake in selected form. Error bars represents the combined and propagated uncertainty.

Value

a ggplot object

See Also

read_hdx calculate_state_uptake

Examples

```
uptake_dat <- calculate_state_uptake(alpha_dat)
plot_state_comparison(uptake_dat)

plot_state_comparison(uptake_dat, all_times = TRUE)
plot_state_comparison(uptake_dat, fractional = TRUE, all_times = TRUE)
```

---

plot_uncertainty *Uncertainty of the peptide measurements*

Description

Plot the uncertainty of the mass measurements - for aggregated data or before aggregation - to see if there is a region with uncertainty higher than acceptable

Usage

```
plot_uncertainty(
  dat,
  protein = dat[["Protein"]][1],
  state = dat[["State"]][1],
  skip_amino = 0,
  aggregated = TRUE,
  separate_times = TRUE,
  show_threshold = TRUE,
  interactive = getOption("hadex_use_interactive_plots")
)
```

Arguments

dat	data imported by the read_hdx function
protein	selected protein
state	selected biological state for given protein
skip_amino	integer, indicator how many amino acids from the N-terminus should be omitted
aggregated	logical, indicator if presented data is aggregated on replicate level
separate_times	logical, indicator if the values for different time points are presented separately
show_threshold	logical, indicator if the threshold of significance is shown
interactive	logical, whether plot should have an interactive layer created with with ggiraph, which would add tooltips to the plot in an interactive display (HTML/Markdown documents or shiny app)

Details

The function plot_uncertainty generates a plot of uncertainty of mass measurement of each peptide from selected protein in selected biological state. The values can be presented in two ways: as aggregated values for each replicate, or before aggregation - measured values for charge values within a replicate. On X-axis there is a position in a sequence, with length of a segment of each peptide representing its length. On Y-axis there is uncertainty of the measurement in Daltons. The threshold is set to 1 Da, as this value is associated with exchange.

Value

a ggplot object.

See Also

read_hdx calculate_exp_masses

Examples

```
plot_uncertainty(alpha_dat)
plot_uncertainty(alpha_dat, aggregated = FALSE)
plot_uncertainty(alpha_dat, aggregated = FALSE, separate_times = FALSE)
plot_uncertainty(alpha_dat, skip_amino = 1)
```

---

plot_uptake_curve *Deuterium uptake curve*

Description

Plot deuterium uptake curve for selected peptides

Usage

```
plot_uptake_curve(
  uc_dat,
  theoretical = FALSE,
  fractional = FALSE,
  uncertainty_type = "ribbon",
  log_x = TRUE,
  interactive = getOption("hadex_use_interactive_plots")
)
```

Arguments

uc_dat	data produced by calculate_kinetics or calculate_peptide_kinetics or create_kinetic_dataset functions
theoretical	logical, indicator if values are calculated using theoretical controls
fractional	logical, indicator if values are shown in fractional form
uncertainty_type	type of uncertainty presentation, possible values: "ribbon", "bars" or "bars + line"
log_x	logical, indicator if the X axis values are transformed to log10
interactive	logical, whether plot should have an interactive layer created with with ggiraph, which would add tooltips to the plot in an interactive display (HTML/Markdown documents or shiny app).

Details

The function plot_uptake_curve generates the deuterium uptake curve for selected peptides from selected protein. On X-axis there are time points of measurements. On Y-axis there is deuterium uptake in selected form. The combined and propagated uncertainty can be presented as ribbons or error bars.

Value

a ggplot object.

See Also

read_hdx calculate_kinetics calculate_peptide_kinetics create_kinetic_dataset

Examples

```
uc_dat <- calculate_kinetics(alpha_dat, protein = "db_eEF1Ba",
                             sequence = "GFGDLKSPAGL",
                             state = "Alpha_KSCN",
                             start = 1, end = 11,
                             time_0 = 0, time_100 = 1440)
plot_uptake_curve(uc_dat = uc_dat,
                  theoretical = FALSE,
                  fractional = TRUE)
```

---

plot_volcano *Volcano plot*

Description

Volcano plot for differential deuterium uptake between two biological states

Usage

```
plot_volcano(
  p_dat,
  state_1 = "",
  state_2 = "",
  adjust_axes = TRUE,
  show_confidence_limits = FALSE,
  confidence_level = 0.98,
  color_times = TRUE,
  show_insignificant_grey = FALSE,
  hide_insignificant = FALSE,
  fractional = FALSE,
  theoretical = FALSE,
  interactive = getOption("hadex_use_interactive_plots")
)
```

Arguments

p_dat	data produced by the create_p_diff_uptake_dataset function
state_1	selected biological state for given protein
state_2	selected biological state for given protein
adjust_axes	logical, indicator if the X-axis is symmetrical in relation to 0
show_confidence_limits	logical, indicates if the hybrid test confidence levels are shown
confidence_level	confidence level for the test, from range [0, 1]. It should be the same as used to prepare p_dat
color_times	logical, indicator if different time points are distinguishable by color
show_insignificant_grey	logical, indicator if the values not passing the test are shown in grey
hide_insignificant	logical, indicator if the values not passing the test are hidden
fractional	logical, indicator if values are shown in fractional form
theoretical	logical, indicator if values are calculated using theoretical controls
interactive	logical, whether plot should have an interactive layer created with with ggiraph, which would add tooltips to the plot in an interactive display (HTML/Markdown documents or shiny app).

Details

The function plot_volcano generates the volcano plot based on supplied p_dat. On X-axis there is differential deuterium uptake in selected form. On Y-axis there is the P-value from t-Student test between two biological states. Based on selected confidence level, the confidence limits are calculated to indicate statistically significant values - shown as red dotted lines. The values that are in upper left and right corners pass the hybrid test.

Value

a ggplot object.

References

Hageman, T. S. & Weis, D. D. Reliable Identification of Significant Differences in Differential Hydrogen Exchange-Mass Spectrometry Measurements Using a Hybrid Significance Testing Approach. Anal Chem 91, 8008–8016 (2019).

Houde, D., Berkowitz, S.A., and Engen, J.R. (2011). The Utility of Hydrogen/Deuterium Exchange Mass Spectrometry in Biopharmaceutical Comparability Studies. J Pharm Sci 100, 2071–2086.

See Also

create_p_diff_uptake_dataset

Examples

```
p_dat <- create_p_diff_uptake_dataset(alpha_dat)
plot_volcano(p_dat, show_confidence_limits = TRUE)

plot_volcano(p_dat, show_confidence_limits = TRUE, show_insignificant_grey = TRUE)
plot_volcano(p_dat, show_confidence_limits = TRUE, hide_insignificant = TRUE)
```

`prepare_hdxviewer_export` *Prepares data export for HDX-Viewer*

Description

This function produces the data in format suitable for HDX-Viewer. If necessary, this result can be downloaded as the csv file with download indicator set to true.

Usage

```
prepare_hdxviewer_export(
  x_dat,
  differential = FALSE,
  fractional = TRUE,
  theoretical = FALSE,
  download = FALSE,
  file_path = tempdir()
)
```

Arguments

`x_dat`	one state deuterium uptake data or differential uptake data
`differential`	indicator of x_dat type
`fractional`	indicator if fractional values are used
`theoretical`	indicator if theoretical values are used
`download`	indicator if the result should be downloaded as csv file
`file_path`	path for saving downloaded file

Value

a `data.frame` object

Examples

```
# disabled due to long execution time

kin_dat <- create_uptake_dataset(alpha_dat, states = "Alpha_KSCN" )
aggregated_dat <- create_aggregated_uptake_dataset(kin_dat)
diff_uptake_dat <- create_diff_uptake_dataset(alpha_dat)
diff_aggregated_dat <- create_aggregated_diff_uptake_dataset(diff_uptake_dat)
prepare_hdxviewer_export(aggregated_dat, differential = FALSE)
# prepare_hdxviewer_export(aggregated_dat, differential = TRUE) # shouldnt work
prepare_hdxviewer_export(diff_aggregated_dat, differential = TRUE)
# prepare_hdxviewer_export(aggregated_dat, differential = FALSE, download = TRUE) # writes a file
```

---

read_hdx *Read HDX-MS data file*

Description

Import HDX-MS datafile and validate its content

Usage

```
read_hdx(filename, separator = ",")
```

Arguments

filename	a file supplied by the user. Formats allowed: .csv, .xlsx and .xls.
separator	a value separating the columns.

Details

The function read_hdx generates a dataset read from the supplied datafile. The files produced by DynamX 3.0 or 2.0 in 'cluster data' format and 'tables' file from HDeXaminer are handled. Moreover, the data should include at least two replicates of the experiment to calculate the uncertainty of the measurement. For the files of HDeXaminer origin, the rows with no complete information (e.q. missing 'Exp Cent' value) are removed. The 'Confidence' column is preserved as the user should have impact on accepting rows based on their Confidence flag. Moreover, those files need action from the user - to confirm data processing (e.q. FD time point), choose accepted confidence values and make some change of the labels using update_hdexaminer_file function. For further information check the documentation. IMPORTANT! The files of HDeXaminer origin MUST be processed by hand or by update_hdexaminer_file function to fit the input of processing functions e.q. calculate_state_uptake or calculate_diff_uptake.

Value

a data.frame object

See Also

update_hdexaminer_file create_control_dataset calculate_state_uptake

Examples

```
dat <- read_hdx(system.file(package = "HaDeX2",
                    "HaDeX/data/alpha.csv"))
head(dat)
```

`reconstruct_sequence` *Reconstruct protein sequence*

Description

Reconstruct protein sequence from experimental data

Usage

```
reconstruct_sequence(
  dat,
  protein = dat[["Protein"]][1],
  states = unique(dat[["State"]]),
  end = NULL
)
```

Arguments

dat	data read by read_hdx
protein	selected protein
states	selected biological states for given protein
end	numeric, end position of the protein

Details

The function reconstruct_sequence generates protein sequence from supplied experimental data. For a position not covered, letter x is shown. If the C-terminus of protein is not covered, there is a possibility to manually fix the protein length by passing a value to the ‘end‘ parameter.

Value

a character object.

See Also

read_hdx

Examples

```
reconstruct_sequence(alpha_dat)
```

`show_aggregated_uptake_data` *Show aggregated values in friendly form*

Description

Function plots the aggregated uptake data with regard to submitted parameters in a friendly form. Designed for GUI.

Usage

```
show_aggregated_uptake_data(
  aggregated_dat,
  differential = FALSE,
  fractional = TRUE,
  theoretical = FALSE
)
```

Arguments

`aggregated_dat`	aggregated uptake data
`differential`	indicator if the aggregated_dat is single-state or differential
`fractional`	`logical`, determines if values are fractional
`theoretical`	`logical`, determines if values are theoretical

Value

a `data.frame` object

Examples

```
# disabled due to long execution time

kin_dat <- create_uptake_dataset(alpha_dat, states = "Alpha_KSCN")
aggregated_dat <- create_aggregated_uptake_dataset(kin_dat)
show_aggregated_uptake_data(aggregated_dat)

diff_uptake_dat <- create_diff_uptake_dataset(alpha_dat)
aggregated_diff_dat <- create_aggregated_diff_uptake_dataset(diff_uptake_dat)
show_aggregated_uptake_data(aggregated_diff_dat, differential = TRUE)
```

show_coverage_heatmap_data *Coverage heatmap data*

Description

This function prepares the data used in coverage heatmap to be shown in user-friendly way.

Usage

```
show_coverage_heatmap_data(x_dat, value = NULL)
```

Arguments

x_dat	data created using calculate_ or create_ function
value	value to be presented

Value

a data.table object

Examples

```
# auc data
auc_dat <- calculate_auc(create_uptake_dataset(alpha_dat))
show_coverage_heatmap_data(auc_dat, value = "auc")

# back-exchange
bex_dat <- calculate_back_exchange(alpha_dat)
show_coverage_heatmap_data(bex_dat, value = "back_exchange")
```

show_diff_uptake_data *Differential deuterium uptake data*

Description

Present differential deuterium uptake values in selected form

Usage

```
show_diff_uptake_data(
  diff_uptake_dat,
  theoretical = FALSE,
  fractional = FALSE,
  renamed = TRUE
)
```

Arguments

diff_uptake_dat	data produced by create_diff_uptake_dataset function
theoretical	logical, indicator if values are calculated using theoretical controls
fractional	logical, indicator if values are shown in fractional form
renamed	logical, indicator if the names of the columns are renamed to user-friendly ones. Currently FALSE not implemented.

Details

The function show_uptake_data generates a subsets of the diff_uptake_dat based on selected parameters. The numerical values are rounded to 4 places. The names of columns are changed to user-friendly ones.

Value

a data.frame object

See Also

read_hdx create_diff_uptake_dataset

Examples

```
diff_uptake_dat <- create_diff_uptake_dataset(alpha_dat)
head(show_diff_uptake_data(diff_uptake_dat))
```

---

show_diff_uptake_data_confidence

Differential uptake data with confidence

Description

Present differential deuterium uptake values in selected form, accompanied by the significance

Usage

```
show_diff_uptake_data_confidence(
  diff_uptake_dat,
  theoretical = FALSE,
  fractional = FALSE,
  confidence_level = 0.98,
  hybrid = FALSE
)
```

Arguments

diff_uptake_dat	data produced by create_diff_uptake_dataset function
theoretical	logical, indicator if values are calculated using theoretical controls
fractional	logical, indicator if values are shown in fractional form
confidence_level	confidence level for the test, from range [0, 1].
hybrid	logical, indicator if the hybrid testing was applied in diff_uptake_dat.

Details

The function show_uptake_data generates a subsets of the uptake dat based on selected parameters. It contains the information if the value is statistically significant at selected confidence level. The numerical values are rounded to 4 places. The names of columns are changed to user-friendly ones.

Value

a data.frame object

See Also

create_diff_uptake_dataset plot_differential

Examples

```
diff_uptake_dat <- create_diff_uptake_dataset(alpha_dat)
show_diff_uptake_data_confidence(diff_uptake_dat)
```

---

show_overlap_data *Show data on peptide overlap*

Description

Presents peptide overlap on protein sequence data, based on the supplied parameters.

Usage

```
show_overlap_data(
  dat,
  protein = dat[["Protein"]][1],
  state = dat[["State"]][1],
  start = min(dat[["Start"]]),
  end = max(dat[["End"]])
)
```

Arguments

dat	data imported by the read_hdx function.
protein	chosen protein.
state	biological state for chosen protein.
start	start position of chosen protein.
end	end position of chosen protein.

Details

The data frame presents all the peptides in given state, with its start and end position on the protein sequence. This data is available in the GUI.

Value

a data.frame object.

See Also

read_hdx plot_overlap

Examples

```
show_overlap_data(alpha_dat)
```

---

show_p_diff_uptake_data

Differential deuterium uptake data

Description

Present differential deuterium uptake values in selected form

Usage

```
show_p_diff_uptake_data(
  p_diff_uptake_dat,
  theoretical = FALSE,
  fractional = FALSE,
  renamed = TRUE
)
```

Arguments

p_diff_uptake_dat	data produced by create_p_diff_uptake_dataset function
theoretical	logical, indicator if values are calculated using theoretical controls
fractional	logical, indicator if values are shown in fractional form
renamed	logical, indicator if the names of the columns are renamed to user-friendly ones. Currently FALSE not implemented.

Details

The function show_uptake_data generates a subsets of the diff_uptake_dat based on selected parameters. The numerical values are rounded to 4 places. The names of columns are changed to user-friendly ones.

Value

a data.frame object

See Also

read_hdx create_diff_uptake_dataset

Examples

```
p_diff_uptake_dat <- create_p_diff_uptake_dataset(alpha_dat)
head(show_p_diff_uptake_data(p_diff_uptake_dat))
```

---

show_peptide_charge_measurement

Show peptide charge measurement

Description

Show the charge measurements from replicates for peptide in specific time point.

Usage

```
show_peptide_charge_measurement(
  dat,
  protein = dat[["Protein"]][1],
  state = dat[["State"]][1],
  sequence = dat[["Sequence"]][1],
  time_t = unique(dat[["Exposure"]])[3]
)
```

Arguments

dat	data as imported by the read_hdx function.
protein	chosen protein.
state	biological state for chosen protein.
sequence	sequence of chosen peptide.
time_t	time point of the measurement.

Details

This function shows the measurements of charge from different replicates for specific peptide in specific state in specific time point of measurement.

Value

a data.frame object.

See Also

read_hdx plot_peptide_charge_measurement

Examples

```
show_peptide_charge_measurement(alpha_dat)
```

---

show_peptide_mass_measurement

Show peptide mass measurement

Description

Show the mass measurements from replicates for peptide in specific time point.

Usage

```
show_peptide_mass_measurement(
  rep_mass_dat,
  protein = rep_mass_dat[["Protein"]][1],
  state = rep_mass_dat[["State"]][1],
  sequence = rep_mass_dat[["Sequence"]][1],
  time_t = unique(rep_mass_dat[["Exposure"]])[3]
)
```

Arguments

rep_mass_dat	data produced by calculate_exp_masses_per_replicate function.
protein	chosen protein.
state	biological state for chosen protein.
sequence	sequence of chosen peptide.
time_t	time point of the measurement.

Details

This function shows the measurements of mass from different replicates for specific peptide in specific state in specific time point of measurement.

Value

a data.frame object.

See Also

read_hdx calculate_exp_masses_per_replicate calculate_exp_masses calculate_state_uptake calculate_diff_uptake

Examples

```
rep_mass_dat <- calculate_exp_masses_per_replicate(alpha_dat)
show_peptide_mass_measurement(rep_mass_dat)
```

---

show_quality_control_data

Show quality control data

Description

Generates quality control data, based on the supplied parameters.

Usage

```
show_quality_control_data(qc_dat)
```

Arguments

qc_dat	data produced by create_quality_control_dataset function, scaled if necessary.

Details

This data frame presents the mean uncertainty in function of selected maximal exchange control time of measurement. This data is available in the GUI.

Value

a data.frame object.

See Also

create_quality_control_dataset plot_quality_control

Examples

```
qc_dat <- create_quality_control_dataset(alpha_dat)
show_quality_control_data(qc_dat)
```

---

show_replicate_histogram_data

Show replicate data

Description

Show histogram data on number of replicates per peptide in one or multiple time point of measurement.

Usage

```
show_replicate_histogram_data(rep_dat)
```

Arguments

rep_dat	replicate data, created by create_replicate_dataset function.

Details

The function shows the information about number of replicates for peptides in one or multiple time point of measurement, depends on supplied data.

Value

a data.frame object.

See Also

create_replicate_dataset plot_replicate_histogram

Examples

```
rep_dat <- create_replicate_dataset(alpha_dat)
show_replicate_histogram_data(rep_dat)
```

show_summary_data *Summary data*

Description

Show summary table

Usage

```
show_summary_data(
  dat,
  confidence_level = "",
  protein_length = max(dat[["End"]])
)
```

Arguments

dat	data imported by the read_hdx function
confidence_level	confidence level for the test, from range [0, 1].
protein_length	the length of the protein sequence

Details

The format in the table is suggested by the community and should be provided for experimental data.

Value

a data.table object

References

Masson, G.R., Burke, J.E., Ahn, N.G., Anand, G.S., Borchers, C., Brier, S., Bou-Assaf, G.M., Engen, J.R., Englander, S.W., Faber, J., et al. (2019). Recommendations for performing, interpreting and reporting hydrogen deuterium exchange mass spectrometry (HDX-MS) experiments. Nat Methods 16, 595–602

See Also

read_hdx

Examples

```
show_summary_data(alpha_dat)
```

show_uc_data *Deuterium uptake curve data*

Description

Present deuterium uptake curve data

Usage

```
show_uc_data(uc_dat, theoretical = FALSE, fractional = FALSE)
```

Arguments

uc_dat	calculated kinetic data by calculate_kinetics or calculate_peptide_kinetics or create_kinetic_dataset function
theoretical	logical, indicator if values are calculated using theoretical controls
fractional	logical, indicator if values are shown in fractional form

Details

The function show_uptake_data generates a subsets of the uc_dat based on selected parameters. The numerical values are rounded to 4 places. The names of columns are changed to user-friendly ones.

Value

a data.frame object

See Also

read_hdx calculate_kinetics calculate_peptide_kinetics

Examples

```
uc_dat <- calculate_kinetics(alpha_dat,
                             sequence = "GFGDLKSPAGL",
                             state = "ALPHA_Gamma",
                             start = 1, end = 11)
show_uc_data(uc_dat)
```

show_uptake_data *Deuterium uptake data*

Description

Present deuterium uptake values in selected form

Usage

```
show_uptake_data(
  uptake_dat,
  theoretical = FALSE,
  fractional = FALSE,
  renamed = TRUE
)
```

Arguments

uptake_dat	data produced by create_uptake_dataset function or create_state_uptake_dataset
theoretical	logical, indicator if values are calculated using theoretical controls
fractional	logical, indicator if values are shown in fractional form
renamed	logical, indicator if the names of the columns are renamed to user-friendly ones. Currently FALSE not implemented.

Details

The function show_uptake_data generates a subsets of the uptake_dat based on selected parameters. The numerical values are rounded to 4 places. The names of columns are changed to user-friendly ones.

Value

a data.frame object

See Also

read_hdx create_uptake_dataset

Examples

```
uptake_dat <- create_uptake_dataset(alpha_dat)
show_uptake_data(uptake_dat)
```

update_hdexaminer_file *Update HDeXaminer datafile*

Description

Update data from HDeXaminer file

Usage

```
update_hdexaminer_file(
  dat,
  fd_time,
  old_protein_name = NULL,
  new_protein_name = NULL,
  old_state_name = NULL,
  new_state_name = NULL,
  confidence = c("High", "Medium")
)
```

Arguments

dat	data imported by the read_hdx function
fd_time	time point [min] for fully deuterated sample
old_protein_name	protein name to be changed
new_protein_name	new name for old_protein_name
old_state_name	state names to be changed
new_state_name	new names for old_state_name
confidence	vector of accepted confidence values (internal flag from HDeXaminer). By default only accepted values are 'Medium' and 'High', with 'Low' excluded

Details

The function update_hdexaminer_file changes the data read from HDeXaminer file. Data from HDeXaminer is condensed and automated data retrieving may be corrected by the user. The original file has a mark "FD" for fully deuterated data instead of numerical value for time point (provided in minutes) that is not consistent for workflow and not enough for precise data description. Moreover, the data about both protein and state is included in one column and for detailed information function update_hdexaminer_file allows to change them.

Value

a data.frame object

See Also

read_hdx calculate_kinetics plot_coverage plot_position_frequency reconstruct_sequence

Index