# Package 'HaDeX'

October 12, 2022

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

**Version** 1.2.2

**Description** Functions for processing, analysis and visualization of Hydrogen Deuterium eXchange monitored by Mass Spectrometry experiments (HDX-MS) (10.1093/bioinformatics/btaa587). 'HaDeX' introduces a new standardized and reproducible workflow for the analysis of the HDX-MS data, including novel uncertainty intervals. Additionally, it covers data exploration, quality control and generation of publication-quality figures. All functionalities are also available in the in-built 'Shiny' app.

**Depends** R (>= 3.0)

**License** GPL-3

**Encoding** UTF-8

**RoxygenNote** 7.1.1

**Imports** data.table, dplyr, DT, ggplot2, gsubfn, latex2exp, reshape2, readr, readxl, shiny, stringr, tidyr

**Suggests** spelling, covr, digest, gridExtra, knitr, pander, renv, rmarkdown, shinycssloaders, shinyhelper, shinyjs, testthat, vdiffr

**VignetteBuilder** knitr

**Language** en-US

**NeedsCompilation** no

**Author** Weronika Puchala [cre, aut] (<https://orcid.org/0000-0003-2163-1429>), Michal Burdukiewicz [aut] (<https://orcid.org/0000-0001-8926-582X>), Dominik Rafacz [ctb] (<https://orcid.org/0000-0003-0925-1909>)

**Maintainer** Weronika Puchala <puchala.weronika@gmail.com>

**Repository** CRAN

**Date/Publication** 2021-08-12 14:00:02 UTC

# R topics documented:



---

`HaDeX-package` *HaDeX*

---

## Description

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

## Author(s)

Weronika Puchala, Michal Burdukiewicz.

---

`add_stat_dependency` *Calculates confidence limits*

---

## Description

Returns relation with confidence limits for each peptide.

## Usage

```
add_stat_dependency(
  calc_dat,
  confidence_limit = 0.98,
  theoretical = FALSE,
  relative = TRUE
)
```

## Arguments

| | |
|---|---|
| calc_dat | processed data from DynamX file - using prepare_dataset |
| confidence_limit | confidence limit chosen by user - from range [0, 1]. |
| theoretical | logical value to determine if the plot is theoretical or not. |
| relative | logical value to determine if values are relative or absolute. |

## Details

...

## 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/relative)

## See Also

read_hdx prepare_dataset

## Examples

```
#load example data
dat <- read_hdx(system.file(package = "HaDeX",
                            "HaDeX/data/KD_180110_CD160_HVEM.csv"))

# prepate dataset for states `CD160` and `CD160_HVEM` in given time parameters
calc_dat <- prepare_dataset(dat,
                            in_state_first = "CD160_0.001",
                            chosen_state_first = "CD160_1",
                            out_state_first = "CD160_1440",
                            in_state_second = "CD160_HVEM_0.001",
                            chosen_state_second = "CD160_HVEM_1",
                            out_state_second = "CD160_HVEM_1440")

# add calculated confidence limits for prepared data
add_stat_dependency(calc_dat,
                    confidence_limit = 0.98,
                    theoretical = FALSE,
                    relative = TRUE)
```

`calculate_confidence_limit_values` *Calculate the value of confidence limit*

## Description

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

## Usage

```
calculate_confidence_limit_values(
  calc_dat,
  confidence_limit = 0.98,
  theoretical = FALSE,
  relative = TRUE
)
```

## Arguments

| | |
|---|---|
| `calc_dat` | processed data from DynamX file - using prepare_dataset |
| `confidence_limit` | confidence limit chosen by user - from range [0, 1]. |
| `theoretical` | logical value to determine if plot is theoretical or not. |
| `relative` | logical value to determine if values are relative or absolute. |

## Details

...

## 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` `prepare_dataset`

## Examples

```
# load example data
dat <- read_hdx(system.file(package = "HaDeX", "HaDeX/data/KD_180110_CD160_HVEM.csv"))

# prepare dataset for states `CD160` and `CD160_HVEM` in given time parameters
calc_dat <- prepare_dataset(dat,
                            in_state_first = "CD160_0.001",
                            chosen_state_first = "CD160_1",
                            out_state_first = "CD160_1440",
                            in_state_second = "CD160_HVEM_0.001",
                            chosen_state_second = "CD160_HVEM_1",
                            out_state_second = "CD160_HVEM_1440")

# calculates confidence limits for prepared data
calculate_confidence_limit_values(calc_dat = calc_dat,
                                  confidence_limit = 0.99,
                                  theoretical = FALSE,
                                  relative = TRUE)
```

---

`calculate_kinetics` *Calculate kinetic data*

---

## Description

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

## Usage

```
calculate_kinetics(
  dat,
  protein = dat[["Protein"]][1],
  sequence,
  state,
  start,
  end,
  time_in,
  time_out,
  deut_part = 1
)
```

## Arguments

| | |
|---|---|
| `dat` | dat data read by `read_hdx` |
| `protein` | protein value for chosen peptide |
| `sequence` | sequence of the peptide for which the kinetics is calculated |
| `state` | state of given sequence |

| | |
|---|---|
| start | end of given sequence |
| end | end of given sequence |
| time_in | time in for experimental calculations |
| time_out | time out for experimental calculations |
| deut_part | percentage of deuterium the protein was exposed to, value in range [0, 1] |

## Details

The function calculates deuteration data for all available data points for given peptide. All four variants (relative & theoretical combinations) of deuteration 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_kinetics function. The first version doesn't support filled Modification and Fragment columns.

## Value

data frame with deuteration calculated for all the data points between time_in and time_out. The chosen time point for which deuteration in all four variants is calculated is available in column 'time_chosen'. The rest of the returned structure is equivalent to structure returned by calculate_state_deuteration.

## See Also

read_hdx calculate_state_deuteration plot_kinetics

## Examples

```
# load example data
dat <- read_hdx(system.file(package = "HaDeX",
                            "HaDeX/data/KD_180110_CD160_HVEM.csv"))

# calculate data for sequence INITSSASQEGTRLN in state CD160
(kin1 <- calculate_kinetics(dat,
                    protein = "db_CD160",
                    sequence = "INITSSASQEGTRLN",
                    state = "CD160",
                    start = 1,
                    end = 15,
                    time_in = 0.001,
                    time_out = 1440))

# calculate data for sequence INITSSASQEGTRLN in state CD160_HVEM
(kin2 <- calculate_kinetics(dat,
                    protein = "db_CD160",
                    sequence = "INITSSASQEGTRLN",
                    state = "CD160_HVEM",
                    start = 1,
                    end = 15,
                    time_in = 0.001,
                    time_out = 1440))
```

```
# load extra libraries
library(dplyr)
library(ggplot2)

# plot example - experimental and relative
bind_rows(kin1, kin2) %>%
 plot_kinetics(theoretical = FALSE,
               relative = TRUE) +
 labs(title = "Kinetic plot for INITSSASQEGTRLN")

# plot example - theoretical and absolute
bind_rows(kin1, kin2) %>%
  plot_kinetics(theoretical = TRUE,
               relative = FALSE) +
  labs(title = "Theoretical kinetics plot for INITSSASQEGTRLN")
```

---

`calculate_state_deuteration` *Calculate deuteration*

---

## Description

Calculates deuteration uptake based on supplied parameters.

## Usage

```
calculate_state_deuteration(
  dat,
  protein,
  state,
  time_in,
  time_chosen,
  time_out,
  deut_part = 1
)
```

## Arguments

| | |
|---|---|
| `dat` | data as imported by the `read_hdx` function |
| `protein` | protein included in calculations |
| `state` | state included in calculations |
| `time_in` | experimental 'time_in' |
| `time_chosen` | chosen time point |
| `time_out` | experimental 'time_out' |
| `deut_part` | percentage of deuterium the protein was exposed to, value in range [0, 1] |

## Details

The function `calculate_state_deuteration` calculates deuteration for peptides in given protein in given state based on supplied parameters: 'time_in', 'time_out' and 'time_chosen'. All four variants (combinations of theoretical & relative) 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` `calculate_confidence_limit_values` `add_stat_dependency`

## Examples

```
# load example data
dat <- read_hdx(system.file(package = "HaDeX", "HaDeX/data/KD_180110_CD160_HVEM.csv"))

# calculate deuteration for state "CD160"
calculate_state_deuteration(dat, protein = "db_CD160", state = "CD160",
                            time_in = 0, time_chosen = 5.000, time_out = 1440.000)
```

---

# comparison_plot *Plot comparison plot*

---

## Description

Produces comparison_plot based on previously processed data - theoretical or experimental. User can change labels if needed.

## Usage

```
comparison_plot(
  calc_dat,
  theoretical = FALSE,
  relative = TRUE,
  state_first = "state_first",
  state_second = "state_second"
)
```

## Arguments

| | |
|---|---|
| calc_dat | processed data from DynamX file - using prepare_dataset |
| theoretical | logical value to determine if plot is theoretical or not. default : false |
| relative | logical value to determine if values are relative or absolute. default : true |
| state_first | first state name |
| state_second | second state name |

## Details

...

This is the first version - multi-state calculations are not supported.

## Value

a ggplot object.

## See Also

read_hdx prepare_dataset

## Examples

```
# load example data
dat <- read_hdx(system.file(package = "HaDeX", "HaDeX/data/KD_180110_CD160_HVEM.csv"))

# prepare dataset for states `CD160` and `CD160_HVEM` in given time parameters
calc_dat <- prepare_dataset(dat,
                            in_state_first = "CD160_0.001",
                            chosen_state_first = "CD160_1",
                            out_state_first = "CD160_1440",
                            in_state_second = "CD160_HVEM_0.001",
                            chosen_state_second = "CD160_HVEM_1",
                            out_state_second = "CD160_HVEM_1440")

# plot comparison plot - theoretical & relative
comparison_plot(calc_dat = calc_dat,
                theoretical = TRUE,
                relative = TRUE,
                state_first = "CD160",
                state_second = "CD160_HVEM")

# plot comparison plot - experimental & relative
comparison_plot(calc_dat = calc_dat,
                theoretical = FALSE,
                relative = TRUE,
                state_first = "CD160",
                state_second = "CD160_HVEM")

# plot comparison plot - theoretical & absolute
comparison_plot(calc_dat = calc_dat,
```

```
                theoretical = TRUE,
                relative = FALSE,
                state_first = "CD160",
                state_second = "CD160_HVEM")

# plot comparison plot - experimental & absolute
comparison_plot(calc_dat = calc_dat,
                theoretical = FALSE,
                relative = FALSE,
                state_first = "CD160",
                state_second = "CD160_HVEM")
```

---

HaDeX_gui *HaDeX Graphical User Interface*

---

## Description

Launches graphical user interface.

## Usage

```
HaDeX_gui(port = getOption("shiny.port"))
```

## Arguments

| | |
|---|---|
| port | The TCP port. See runApp. |

## Warning

Any ad-blocking software may cause malfunctions.

---

plot_coverage *Plot peptide coverage*

---

## Description

Plots the peptide coverage of the protein sequence.

## Usage

```
plot_coverage(
  dat,
  protein = dat[["Protein"]][1],
  chosen_state = dat[["State"]][1]
)
```

### Arguments

| | |
|---|---|
| dat | data as imported by the read_hdx function |
| protein | protein to be included in plot |
| chosen_state | sequence states to be included in plot |

### Details

The function plot_coverage plots sequence coverage based on experimental data for chosen protein in chosen state. Only non-duplicated peptides are shown on the plot, next to each other.

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

### Value

a ggplot object.

### See Also

read_hdx plot_position_frequency

### Examples

```
# load example data
dat <- read_hdx(system.file(package = "HaDeX",
                            "HaDeX/data/KD_180110_CD160_HVEM.csv"))

# plot coverage with default parameters
plot_coverage(dat)

# plot coverage with explicit parameters
plot_coverage(dat, protein = "db_CD160", chosen_state = "CD160_HVEM")
```

---

## plot_kinetics *Plot kinetics data*

---

### Description

Plots kinetics of the hydrogen-deuterium exchange for specific peptides.

### Usage

```
plot_kinetics(kin_dat, theoretical = FALSE, relative = TRUE)
```

## Arguments

| | |
|---|---|
| kin_dat | calculated kinetic data by calculate_kinetics function |
| theoretical | logical, determines if plot shows theoretical values |
| relative | logical, determines if values are relative or absolute |

## Details

This function visualises the output of the calculate_kinetics function. Based on supplied parameters appropriate columns are chosen for the plot. The uncertainty associated with each peptide is shown as a ribbon. Axis are labeled according to the supplied parameters but no title is provided.

If you want to plot data for more then one peptide in one state, join calculated data by using bind_rows from dplyr package and pass the result as kin_dat.

## Value

a ggplot object.

## See Also

calculate_kinetics

## Examples

```
# load example data
dat <- read_hdx(system.file(package = "HaDeX", "HaDeX/data/KD_180110_CD160_HVEM.csv"))

# calculate data for the sequence INITSSASQEGTRLN in the CD160 state
(kin1 <- calculate_kinetics(dat,
                            protein = "db_CD160",
                            sequence = "INITSSASQEGTRLN",
                            state = "CD160",
                            start = 1,
                            end = 15,
                            time_in = 0.001,
                            time_out = 1440))

# calculate data for the sequence INITSSASQEGTRLN in the CD160_HVEM state
(kin2 <- calculate_kinetics(dat,
                            protein = "db_CD160",
                            sequence = "INITSSASQEGTRLN",
                            state = "CD160_HVEM",
                            start = 1,
                            end = 15,
                            time_in = 0.001,
                            time_out = 1440))

# load extra packages
library(dplyr)

# plot a single peptide - theoretical and relative
```

```
plot_kinetics(kin_dat = kin1,
              theoretical = TRUE,
              relative = TRUE)

# plot joined data - experimental and absolute
bind_rows(kin1, kin2) %>%
  plot_kinetics(theoretical = FALSE,
                relative = FALSE)
```

---

`plot_position_frequency` *Plot position frequency*

---

## Description

Plots the frequency of coverage of protein sequence.

## Usage

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

## Arguments

| | |
|---|---|
| `dat` | data as imported by the `read_hdx` function |
| `protein` | protein to be included in plot |
| `chosen_state` | sequence states to be included in plot |

## Details

The function `plot_position_frequency` plots a histogram of the coverage frequency based on experimental data. The aim of this plot is to see how many times each position of the sequence was covered (by different peptides).

## Value

a `ggplot` object.

## See Also

`read_hdx` `plot_coverage`

## Examples

```
# load example data
dat <- read_hdx(system.file(package = "HaDeX",
                            "HaDeX/data/KD_180110_CD160_HVEM.csv"))

# function call with default parameters
plot_position_frequency(dat)

# function call with explicit parameters
plot_position_frequency(dat, chosen_state = "CD160_HVEM")
```

---

`prepare_dataset` *Calculate data*

---

## Description

Calculates values for visualization from input data file - both experimental and theoretical. All parameters are needed.

## Usage

```
prepare_dataset(
  dat,
  in_state_first,
  chosen_state_first,
  out_state_first,
  in_state_second,
  chosen_state_second,
  out_state_second
)
```

## Arguments

| | |
|---|---|
| `dat` | data frame with data from DynamX file |
| `in_state_first` | string in form "state_time" for first state in in time |
| `chosen_state_first` | string in form "state_time" for chosen state in in time |
| `out_state_first` | string in form "state_time" for first state in out time |
| `in_state_second` | string in form "state_time" for second state in in time |
| `chosen_state_second` | string in form "state_time" for second state in chosen time |
| `out_state_second` | string in form "state_time" for second state in out time |

## Details

...

This is the first version - multi-state calculations are not supported.

## Value

data frame with calculated values

## See Also

read_hdx

## Examples

```
# load example data
dat <- read_hdx(system.file(package = "HaDeX", "HaDeX/data/KD_180110_CD160_HVEM.csv"))

# prepare dataset for states `CD160` and `CD160_HVEM` in given time parameters
prepare_dataset(dat,
                in_state_first = "CD160_0.001",
                chosen_state_first = "CD160_1",
                out_state_first = "CD160_1440",
                in_state_second = "CD160_HVEM_0.001",
                chosen_state_second = "CD160_HVEM_1",
                out_state_second = "CD160_HVEM_1440")
```

---

# quality_control *Experiment quality control*

---

## Description

Checks how the uncertainty changes in a function of ‘out_time‘.

## Usage

```
quality_control(dat, state_first, state_second, chosen_time, in_time)
```

## Arguments

| | |
|---|---|
| dat | data read by read_hdx |
| state_first | state of the first peptide |
| state_second | state of the second peptide |
| chosen_time | chosen time point |
| in_time | ‘in‘ time |

## Details

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

## Value

`data.frame` with mean uncertainty per different ‘out_time‘ value

## See Also

`read_hdx`

## Examples

```
# load example data
dat <- read_hdx(system.file(package = "HaDeX", "HaDeX/data/KD_180110_CD160_HVEM.csv"))

# calculate mean uncertainty
(result <- quality_control(dat = dat,
                           state_first = "CD160",
                           state_second = "CD160_HVEM",
                           chosen_time = 1,
                           in_time = 0.001))

# load extra libraries
library(ggplot2)
library(tidyr)
library(dplyr)

# example of data visualization
gather(result, 2:7, key = 'type', value = 'value') %>%
filter(startsWith(type, "avg")) %>%
  ggplot(aes(x = factor(out_time), y = value, group = type)) +
  geom_line(aes(color = type)) +
  labs(x = "Out time",
       y = "Mean uncertainty")
```

---

`read_hdx` *Read HDX-MS data file*

---

## Description

Imports data from a HDX-MS file and validates its content.

### Usage

```
read_hdx(filename)
```

### Arguments

| | |
|---|---|
| filename | a file supplied by the user. Formats allowed: .csv, .xlsx and .xls. |

### Details

First version accepts files produced by DynamX 3.0 and 2.0 in 'cluster data' format. The function checks if all necessary columns are provided in correct format. The file must include at least two repetitions of the measurement for the uncertainty to be calculated.

### Value

dat - a data.frame with validated content.

### See Also

calculate_kinetics calculate_state_deuteration plot_coverage plot_position_frequency prepare_dataset quality_control reconstruct_sequence

### Examples

```
# read example data
head(read_hdx(system.file(package = "HaDeX",
                      "HaDeX/data/KD_180110_CD160_HVEM.csv")))
```

---

## reconstruct_sequence *Reconstruct protein sequence*

---

### Description

Reconstructs protein sequence from supplied file.

### Usage

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

### Arguments

| | |
|---|---|
| dat | data read by read_hdx |
| protein | the protein of which the structure is to be reconstructed |

### Details

The function reconstructs protein sequence from supplied experimental data. If a position is not covered, x is shown. First version doesn't support manual sequence length correction.

## Value

reconstructed sequence - `character` object.

## See Also

`read_hdx`

## Examples

```
dat <- read_hdx(system.file(package = "HaDeX", "HaDeX/data/KD_180110_CD160_HVEM.csv"))
reconstruct_sequence(dat)
```

---

`woods_plot` *Plot Woods' plot*

---

## Description

Produces Woods' plot based on theoretical or experimental HDX-MS data.

## Usage

```
woods_plot(
  calc_dat,
  theoretical = FALSE,
  relative = TRUE,
  confidence_limit = 0.98,
  confidence_limit_2 = 0.99
)
```

## Arguments

| | |
|---|---|
| `calc_dat` | data as imported by the `read_hdx` function and processed by the `prepare_dataset` function. |
| `theoretical` | `logical`, determines if plot shows theoretical values. |
| `relative` | `logical`, determines if values are relative or absolute. |
| `confidence_limit` | confidence limit. |
| `confidence_limit_2` | confidence limit 2. |

## Details

...

This is the first version - multi-state calculations are not supported.

## Value

a ggplot object.

## References

Woods, V.L., and Hamuro, Y. (2001). High resolution, high-throughput amide deuterium exchange-mass spectrometry (DXMS) determination of protein binding site structure and dynamics: utility in pharmaceutical design. J. Cell. Biochem. Suppl. Suppl 37, 89–98.

## See Also

read_hdx prepare_dataset

## Examples

```
# load example data
dat <- read_hdx(system.file(package = "HaDeX",
                            "HaDeX/data/KD_180110_CD160_HVEM.csv"))

# prepare dataset for states `CD160` and `CD160_HVEM`
# in given time parameters
calc_dat <- prepare_dataset(dat,
                            in_state_first = "CD160_0.001",
                            chosen_state_first = "CD160_1",
                            out_state_first = "CD160_1440",
                            in_state_second = "CD160_HVEM_0.001",
                            chosen_state_second = "CD160_HVEM_1",
                            out_state_second = "CD160_HVEM_1440")

# plot Woods plot - theoretical & relative
woods_plot(calc_dat = calc_dat,
           theoretical = TRUE,
           relative = TRUE,
           confidence_limit = 0.98,
           confidence_limit_2 = 0.99)

# plot Woods plot - experimental  & relative
woods_plot(calc_dat = calc_dat,
           theoretical = FALSE,
           relative = TRUE,
           confidence_limit = 0.98,
           confidence_limit_2 = 0.99)

# plot Woods plot - theoretical & absolute
woods_plot(calc_dat = calc_dat,
           theoretical = TRUE,
           relative = FALSE,
           confidence_limit = 0.98,
           confidence_limit_2 = 0.99)

# plot Woods plot - experimental & absolute
woods_plot(calc_dat = calc_dat,
```

```
theoretical = FALSE,
relative = FALSE,
confidence_limit = 0.98,
confidence_limit_2 = 0.99)
```

# Index