

# Package ‘HDXBoxeR’

April 10, 2024

**Type** Package

**Title** Analysis of Hydrogen-Deuterium Exchange Mass-Spectrometry Data

**Version** 0.0.1

**Author** Maria K. Janowska [aut, cre] (<<https://orcid.org/0000-0002-8232-461X>>),  
Katherine Reiter [ctb],  
Pearl Magala [ctb],  
Miklos Guttman [ctb],  
Rachel E. Klevit [ctb]

**Maintainer** Maria K. Janowska <mka.janowska@gmail.com>

**Description** A protocol that facilitates the processing and analysis of Hydrogen-Deuterium Exchange Mass Spectrometry data using p-value statistics and Critical Interval analysis.

It provides a pipeline for analyzing data from 'HDXExaminer' (Sierra Analytics, Trajan Scientific), automating matching and comparison of protein states through Welch's T-test and the Critical Interval statistical framework.

Additionally, it simplifies data export, generates 'Pymol' scripts, and ensures calculations meet publication standards.

'HDXBoxeR' assists in various aspects of hydrogen-deuterium exchange data analysis, including reprocessing data, calculating parameters, identifying significant peptides, generating plots, and facilitating comparison between protein states.

For details check papers by Hageman and Weis (2019) <[doi:10.1021/acs.analchem.9b01325](https://doi.org/10.1021/acs.analchem.9b01325)> and Masson et al. (2019) <[doi:10.1038/s41592-019-0459-y](https://doi.org/10.1038/s41592-019-0459-y)>.

**License** GPL (>= 2)

**Imports** dplyr, graphics, grDevices, RColorBrewer, stats, stringr,  
tidyr, utils, methods, wrapr

**Suggests** knitr, rmarkdown, testthat (>= 3.0.0)

**VignetteBuilder** knitr, rmarkdown

**Encoding** UTF-8

**RoxygenNote** 7.2.3

**Config/testthat/edition** 3

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2024-04-10 20:10:02 UTC

**R topics documented:**

all_summary . . . . .	4
arguments_call1 . . . . .	5
arguments_call2 . . . . .	5
arguments_call3 . . . . .	6
arg_df . . . . .	6
arg_UN_FD . . . . .	7
average_timecourse . . . . .	7
ave_timepoint . . . . .	8
av_tc . . . . .	8
av_tp . . . . .	9
backHX_calculations . . . . .	9
boxplot_tp . . . . .	10
CI_2pts . . . . .	10
CI_single . . . . .	11
CI_tc . . . . .	12
CI_tp . . . . .	12
color_ranges_Blue_Red_heat_map . . . . .	13
color_ranges_Spectral . . . . .	14
coverage_residue . . . . .	14
deuteration_woods_timecourse . . . . .	15
deuteration_woods_timepoints . . . . .	16
dif_ave . . . . .	17
dif_tp . . . . .	17
dif_tp_proc . . . . .	18
duplicate_sets . . . . .	18
extreme_input_gap . . . . .	19
extreme_input_undeut . . . . .	19
general_info . . . . .	20
getCoords1 . . . . .	21
heat_map_tc . . . . .	21
heat_map_tp . . . . .	22
heat_map_tp_maxuptake . . . . .	22
heat_map_tp_maxuptake_proc . . . . .	23
heat_map_tp_proc . . . . .	24
is.nan.data.frame . . . . .	25
lab_dif . . . . .	25
lab_dif_proc . . . . .	26
lab_vol . . . . .	26
legend_heat_map . . . . .	27
legend_heat_map_tc . . . . .	27
legend_heat_map_timecourse . . . . .	28
legend_heat_map_tp . . . . .	28
legend_heat_map_tp_proc . . . . .	29
legend_nm_bottom . . . . .	29
legend_raw_ave . . . . .	30
legend_raw_ave_proc . . . . .	30

legend_raw_ave_tc . . . . .	31
legend_sig_peptides . . . . .	31
legend_states_PerD_bottom . . . . .	32
legend_tc_bottom . . . . .	32
nm_states . . . . .	33
output_FD . . . . .	33
output_FD_proc . . . . .	34
output_prep . . . . .	34
output_tc . . . . .	35
output_tp . . . . .	36
output_UD . . . . .	38
output_UD_proc . . . . .	38
palette_legend . . . . .	39
palette_ll . . . . .	39
peptide_pv_tp . . . . .	40
peptide_pv_tp_proc . . . . .	40
plots_av_tcourse . . . . .	41
plots_av_tp . . . . .	42
plots_av_tp_proc . . . . .	43
plots_diff_tp . . . . .	43
plots_diff_tp_proc . . . . .	44
plots_vol_tp . . . . .	45
plot_heat_map_max_uptake_tp . . . . .	45
plot_heat_map_max_uptake_tp_proc . . . . .	46
plot_heat_map_tc . . . . .	47
plot_heat_map_tp . . . . .	48
plot_heat_map_tp_proc . . . . .	49
plot_peptide_sig_tp . . . . .	50
plot_peptide_sig_tp_proc . . . . .	50
pl_gen_ch2 . . . . .	51
pl_gen_uptake . . . . .	52
ppar . . . . .	52
pparLM . . . . .	53
ppar_bottom_legend . . . . .	53
ppar_wider . . . . .	54
prep_timecourse_plot_ave . . . . .	54
prep_timecourse_plot_sd . . . . .	55
pv_timecourse . . . . .	55
pv_timepoint . . . . .	56
pymol_script_average_residue . . . . .	56
pymol_script_significant_peptide . . . . .	57
pymol_script_significant_peptide_proc . . . . .	58
pymol_script_significant_residue . . . . .	59
pymol_script_significant_residue_proc . . . . .	60
pymol_str . . . . .	61
qpcr.cbind.na . . . . .	62
ranges_function . . . . .	63
ranges_function_tc . . . . .	63

rbind_na . . . . .	64
reset_par . . . . .	64
robot_2states_indexes . . . . .	65
robot_indexes . . . . .	66
robot_indexes_df . . . . .	67
robot_plot_All . . . . .	68
sd_timecourse . . . . .	69
sd_timecourse_proc . . . . .	69
sd_timepoint . . . . .	70
select_indices . . . . .	70
significant_peptide_uptake . . . . .	71
summary_sd_CI . . . . .	72
uptake_plots . . . . .	72
verbose_timecourse_output . . . . .	73
verbose_timepoint_output . . . . .	74
vol_tp . . . . .	75
woods_CI_plot . . . . .	75

## Index 77

---

all_summary	<i>Returns full summary table.</i>
-------------	------------------------------------

---

### Description

Returns summary data. Function returns: Protein states, timepoints, number of replicates, # peptides, % coverage, average peptide length and redundancy. backexchange calculations (average and range), Critical interval and standard deviation. Function requires undeuterated and Fully deuterated sets marked in Deut.time as 0s and FD respectively.

### Usage

```
all_summary(filepath, replicates = 3, Dfact = 0.85)
```

### Arguments

filepath	filepath to the input file. Input file is All_results table from HDX_Examiner, where all the fields are marked for export.
replicates	number of replicates. Default set to 3.
Dfact	Dfact is the fraction of D/H in the labeling buffer used. Default set up to 0.85

### Value

Returns summary table.

### Examples

```
file_nm<-system.file("extdata", "All_results_table.csv", package = "HDXBoxer")
a<- all_summary(file_nm, replicates=3, Dfact=0.85)
```

---

arguments_call1	<i>Returns default arguments for the output_tp functions. States</i>
-----------------	--

---

**Description**

Function used as internal function

**Usage**

```
arguments_call1(filepath)
```

**Arguments**

filepath	input file location
----------	---------------------

**Value**

The default arguments to output\_tp functions.

---

arguments_call2	<i>Returns default arguments for the output_tp functions. Deut.Time</i>
-----------------	---

---

**Description**

Function used as internal function

**Usage**

```
arguments_call2(filepath, states)
```

**Arguments**

filepath	input file location
states	states used

**Value**

The default arguments to output\_tp functions.

---

arguments_call3	<i>Returns default arguments for the output_tp functions. # replicates</i>
-----------------	--

---

**Description**

Function used as internal function

**Usage**

```
arguments_call3(filepath, states, times)
```

**Arguments**

filepath	input file location
states	states used
times	deuteration times

**Value**

The default arguments to output\_tp functions.

---

arg_df	<i>Returns initially processed data.frame from the export from the HDX-Examiner</i>
--------	---

---

**Description**

Function used as internal function

**Usage**

```
arg_df(filepath)
```

**Arguments**

filepath	input file location
----------	---------------------

**Value**

Data.frame for further processing

---

arg_UN_FD	<i>Returns initially processed data.frame from the export from the HDX-Examiner</i>
-----------	---

---

**Description**

Function used as internal function

**Usage**

```
arg_UN_FD(filepath)
```

**Arguments**

filepath          input file location

**Value**

Data.frame for further processing

---

average_timecourse	<i>Calculates average for time course data.</i>
--------------------	---

---

**Description**

Calculates average for time course data.

**Usage**

```
average_timecourse(filepath)
```

**Arguments**

filepath          filepath to the All\_results input file.

**Value**

data frame with average deuteration uptake data.

---

ave_timepoint	<i>Returns average value for either uptake of procent data.</i>
---------------	---

---

**Description**

Calculates average of uptake or procent data. Returns data frame with average values. Default for the number of replicates is 3.

**Usage**

```
ave_timepoint(df, replicates = 3)
```

**Arguments**

df	output from functions output_tp or output_tp_proc.
replicates	number of replicates used. Default is set to replicates=3

**Value**

Data.frame with average values

**Examples**

```
file_nm<-system.file("extdata", "All_results_table.csv", package = "HDXBoxeR")
a<- output_tp(file_nm)
ave<-ave_timepoint(df=a) ##if number of replicates is equal 3
ave<-ave_timepoint(df=a, replicates=4) ##if number of replicates is equal 4
```

---

av_tc	<i>Preparatory function for average plot for timecourses</i>
-------	--

---

**Description**

Returns plots with average deuteration at each peptide.

**Usage**

```
av_tc(df, cola)
```

**Arguments**

df	output from functions output_tp or output_tp or output_tp_proc.
cola	color palette for different Protein States. As default Paired palette from color.Brewer is used.

**Value**

plots of averages



---

av_tp	<i>Preparatory function for average plot</i>
-------	--

---

**Description**

Returns plots with average deuteration at each peptide.

**Usage**

```
av_tp(df, cola)
```

**Arguments**

df	output from functions output_tp or output_tp or output_tp_proc.
cola	color palette for different Protein States. As default Paired palette from color.Brewer is used.

**Value**

plots of averages

---

backHX_calculations	<i>Summary of backexchange summary</i>
---------------------	--

---

**Description**

Returns average and ranges of backexchange. Function calculates as:  $1 - (m_{100\%} - m_{0\%}) / N / D_{\text{fact}}$ .  $m_{0\%}$  is the non-deuterated peptide centroid mass,  $m_{100\%}$  is the maximally labeled peptide centroid mass,  $N$  is the theoretical number of backbone amides in the peptide and  $D_{\text{frac}}$  is the fraction of D/H in the labeling buffer used. Function requires undeuterated and Fully deuterated sets marked in Deut.time as 0s and FD respectively.

**Usage**

```
backHX_calculations(filepath, Dfact = 0.85)
```

**Arguments**

filepath	filepath to the input file. Input file is All_results table from HDX_Examiner, where all the fields are marked for export.
Dfact	is the fraction of D/H in the labeling buffer used. Default set up to 0.85

**Value**

Returns summary table for backexchange.

**Examples**

```
file_nm<-system.file("extdata", "All_results_table.csv", package = "HDXBoxeR")
a<- backHX_calculations(filepath=file_nm, Dfact=0.85)
```

---

 boxplot\_tp

*Plots boxplots for all the averages in the set*


---

**Description**

Returns boxplots to compare sets between each other

**Usage**

```
boxplot_tp(df, replicates = 3, ...)
```

**Arguments**

df                    average data frame. Generated using ave\_timepoint() function.  
 replicates           number of replicates in sample. Default set to 3.  
 ...                   inherited boxplot parameters

**Value**

boxplots for average deuterium uptake per set.

**Examples**

```
file_nm<-system.file("extdata", "All_results_table.csv", package = "HDXBoxeR")
a<- output_tp(file_nm)
boxplot_tp(df=a, replicates=3)
```

---

 CI\_2pts

*Global confidence interval threshold from experimental standard deviation for 2 samples.*


---

**Description**

Calculation of global confidence interval using approach by: Reliable Identification of Significant Differences in Differential Hydrogen Exchange-Mass Spectrometry Measurements Using a Hybrid Significance Testing Approach Tyler S. Hageman and David D. Weis Analytical Chemistry 2019 91 (13), 8008-8016 DOI: 10.1021/acs.analchem.9b01325 calculations for alpha 0.99

**Usage**

```
CI_2pts(s1, s2, replicates = 3)
```

**Arguments**

s1                    standard deviation from one sample  
s2                    standard deviation from second sample  
replicates            number of replicates. Default set to 3.

**Value**

threshold for determining significance.

**Examples**

```
sd1<-data.frame(c(0.1, 0.12, 0.13, 0.09, 0.11, 0.10))  
sd2<-data.frame(c(0.18, 0.11, 0.13, 0.08, 0.11, 0.06))  
CI_2pts(s1=sd1, s2=sd2, replicates=3)
```

---

CI_single	<i>Global confidence interval threshold from experimental standard deviation for 1 sample</i>
-----------	---

---

**Description**

Calculation of global confidence interval using approach by: Reliable Identification of Significant Differences in Differential Hydrogen Exchange-Mass Spectrometry Measurements Using a Hybrid Significance Testing Approach Tyler S. Hageman and David D. Weis Analytical Chemistry 2019 91 (13), 8008-8016 DOI: 10.1021/acs.analchem.9b01325 calculations for alpha 0.99

**Usage**

```
CI_single(s1, replicates = 3)
```

**Arguments**

s1                    standard deviation from one sample  
replicates            number of replicates. Default set to 3.

**Value**

threshold for determining significance.

**Examples**

```
sd1<-data.frame(c(0.1, 0.12, 0.13, 0.09, 0.11, 0.10))  
CI_single(s1=sd1, replicates=3)
```

---

CI_tc	<i>Critical interval calculation two sets of timecourses</i>
-------	--

---

**Description**

Preparatory function for calculation of pvalue between sets.

**Usage**

```
CI_tc(sd_c, sd_v, replicates = 3, pv_cutoff = 0.01)
```

**Arguments**

sd_c	dataframe of control
sd_v	dataframe for variant
replicates	number of replicates. Default set to 3.
pv_cutoff	pvalue cutoff. Default set to 0.01

**Value**

Critical interval for 2 sets

---

CI_tp	<i>Global confidence interval treshold from experimental standard deviation</i>
-------	---

---

**Description**

Calculation of global confidence interval using approach by for all protein states compared to first state in the data.frame. Reliable Identification of Significant Differences in Differential Hydrogen Exchange-Mass Spectrometry Measurements Using a Hybrid Significance Testing Approach Tyler S. Hageman and David D. Weis Analytical Chemistry 2019 91 (13), 8008-8016 DOI: 10.1021/acs.analchem.9b01325

**Usage**

```
CI_tp(df, replicates = 3, alpha = 0.01)
```

**Arguments**

df	standard deviation dataframe.
replicates	number of replicates. Default set to 3.
alpha	significance level. Set as default to 0.01

**Value**

treshold for determining significance.

**Examples**

```
file_nm<-system.file("extdata", "All_results_table.csv", package = "HDXBoxeR")
a<- output_tc(file_nm, seq_match=FALSE)
sd<-sd_timepoint(df=a, replicates=3)
CI_tp(df=sd, replicates=3, alpha=0.01 )
CI_tp(sd)
```

---

color\_ranges\_Blue\_Red\_heat\_map

*Returns color pallete from red to blue with number of colors for defined ranges*

---

**Description**

Returns color pallete from red to blue with number of colors for defined ranges

**Usage**

```
color_ranges_Blue_Red_heat_map(ranges, colors_initial)
```

**Arguments**

**ranges** vector of numbers. Should have the same number of positive and negative values and contain 0.

**colors\_initial** additional color that should be first in the palette.

**Value**

color scheme for number

**Examples**

```
color_ranges_Blue_Red_heat_map(ranges=c(-Inf, -100, -50, 0, 50, 100, Inf), colors_initial="white")
```

---

color\_ranges\_Spectral *Returns Spectral palette with colors matching defined ranges*

---

**Description**

Spectral palette for timecourse data

**Usage**

```
color_ranges_Spectral(ranges, colors_initial)
```

**Arguments**

ranges                vector of numbers. Should have the same number of positive and negative values and contain 0.

colors\_initial      additional color that should be first in the palette.

**Value**

color scheme for number

**Examples**

```
color_ranges_Spectral(ranges=c(-Inf, -100, -50, 0, 50, 100, Inf), colors_initial="white")
```

---

coverage\_residue      *Returns coverage per residue*

---

**Description**

returns vector with coverage information

**Usage**

```
coverage_residue(df1, start_col, end_col)
```

**Arguments**

df1                    output from functions output\_tp or output\_tp\_proc.

start\_col             number of "Start" column in data.frame

end\_col                number of "Start" column in data.frame

**Value**

vector with coverage per residue

**Examples**

```
file_nm<-system.file("extdata", "All_results_table.csv", package = "HDXBoxeR")
a<- output_tp(file_nm)
coverage_residue(df1=a,start_col=2, end_col=3 )
```

---

deuteration\_woods\_timecourse

*Return woods plots for the timecourse*

---

**Description**

All the peptides are plotted based on their uptake.

**Usage**

```
deuteration_woods_timecourse(  
  input_data,  
  states,  
  replicates = 3,  
  ylim = c(0, 120),  
  ...  
)
```

**Arguments**

input_data	output from function output_tc(..., percent=TRUE)
states	states, if missing all states used
replicates	replicates
ylim	y axis limits
...	other parameters

**Value**

Woods plots for the timecourse

**Examples**

```
file_nm<-system.file("extdata", "All_results_table.csv", package = "HDXBoxeR")
a<- output_tc(file_nm, percent=TRUE)
deuteration_woods_timecourse(a)
```

---

`deuteration_woods_timepoints`*Return woods plots for the timepoints*

---

**Description**

All the peptides are plotted based on their uptake.

**Usage**

```
deuteration_woods_timepoints(  
  input_data,  
  times,  
  replicates = 3,  
  cola = NA,  
  ylim = c(0, 120),  
  ...  
)
```

**Arguments**

<code>input_data</code>	output from function <code>output_tp(..., percent=TRUE)</code>
<code>times</code>	Deuteration times, if missing all deuteration times used
<code>replicates</code>	replicates
<code>cola</code>	colors, default NA
<code>ylim</code>	y axis limits
<code>...</code>	other parameters

**Value**

Woods plots for the timepoints

**Examples**

```
file_nm<-system.file("extdata", "All_results_table.csv", package = "HDXBoxeR")  
a<- output_tp(file_nm, percent=TRUE)  
deuteration_woods_timepoints(a[1:12,])
```



---

dif_ave	<i>Returns data frame with difference of averages between State1 and other states provided.</i>
---------	---

---

### Description

Returns average difference data.frame. Sets are compared to the first state in the input file. If other order of the sets is required use Default for the number of replicates is 3.

### Usage

```
dif_ave(df)
```

### Arguments

df                    output from functions output\_tp, output\_tp\_proc, output\_tp\_states or output\_tp\_proc\_states.

### Value

Data.frame with difference values btw control and other protein states.

### Examples

```
file_nm<-system.file("extdata", "All_results_table.csv", package = "HDXBoxeR")
a<- output_tp(file_nm)
pv<-pv_timepoint(df=a) ##if number of replicates is equal 3
pv1<-pv_timepoint(df=a, replicates=3) ##if number of replicates is equal 4
#b<-output_tp_states(file_nm, states=c("4EHP", "State2", "State3" ))
#pv_states<-pv_timepoint(df=b) ### here means of State4, will be compared to State2 and State4
```

---

dif_tp	<i>Preparatory function for difference plot</i>
--------	---

---

### Description

Returns plots with difference deuteration at each peptide.

### Usage

```
dif_tp(df, cola)
```

### Arguments

df                    output from functions output\_tp or output\_tp\_proc.  
cola                   color palette for different Protein States. As default Paired palette from color.Brewer is used.

**Value**

plots of difference in average

---

dif_tp_proc	<i>Preparatory function for difference plot</i>
-------------	---

---

**Description**

Returns plots with difference deuteration at each peptide.

**Usage**

```
dif_tp_proc(df, cola)
```

**Arguments**

df	output from functions output_tp or output_tp_proc.
cola	color palette for different Protein States. As default Paired palette from color.Brewer is used.

**Value**

plots of difference in average

---

duplicate_sets	<i>Duplicate set function</i>
----------------	-------------------------------

---

**Description**

Internal function

**Usage**

```
duplicate_sets(df)
```

**Arguments**

df	dataframe
----	-----------

**Value**

duplicate sets

---

extreme\_input\_gap      *Makes input for Extreme for bimodal analysis.*

---

**Description**

Makes input for Extreme for bimodal analysis.

**Usage**

```
extreme_input_gap(hm_dir, replicates, timepoints, output_path = "NA")
```

**Arguments**

hm\_dir                directory in which all the folders which needs to be processed are  
replicates            number of replicates in sample  
timepoints            lists timepoints used in experiments.  
output\_path          directory where the output files will be saved, hm\_dir default

**Value**

Inputs for extreme for all data prepared.

**Examples**

```
path_to_folders<-system.file("extdata", package = "HDXBoxeR")  
  
extreme_input_gap(hm_dir =path_to_folders, replicates = 3,  
timepoints =c(3, 60, 1800, 72000), output_path=tempdir())
```

---

extreme\_input\_undeut      *Makes input for Extreme for bimodal analysis.*

---

**Description**

If data is missing it returns non-deuterated data in these columns.

**Usage**

```
extreme_input_undeut(hm_dir, replicates, timepoints, output_path = "NA")
```

**Arguments**

hm_dir	directory in which all the folders which needs to be processed are
replicates	number of replicates in sample
timepoints	lists timepoints used in experiments.
output_path	directory where output should be written

**Value**

Inputs for extreme for all data prepared.

**Examples**

```
path_to_folders<-system.file("extdata", package = "HDBoxeR")
extreme_input_undeut(hm_dir=path_to_folders, replicates = 3,
timepoints =c(3, 60, 1800, 72000), output_path=tempdir())
```

---

general_info	<i>Provides summary table for all data.sets.</i>
--------------	--

---

**Description**

Returns data frame sumamrizing general information about the data sets. Function returns: Protein states, timepoints, number of replicates, # peptides, % coveregae, average peptide length and redundancy.

**Usage**

```
general_info(filepath)
```

**Arguments**

filepath	filepath to the input file. Input file is All_results table from HDX_Examiner, where all the fields are marked for export.
----------	--

**Value**

Returns summary table.

**Examples**

```
file_nm<-system.file("extdata", "All_results_table.csv", package = "HDBoxeR")
a<- general_info(file_nm)
```

---

getCoords1                      *function from plotfunctions package*

---

**Description**

Margin coordinates

**Usage**

```
getCoords1(pos = 1.1, side = 1, input = "p")
```

**Arguments**

pos	position
side	side of plot
input	plot or figure position

**Value**

coordinates of margins

---

heat\_map\_tc                      *Plots heat maps for time courses.*

---

**Description**

Returns heat map on timecourses with raw data.

**Usage**

```
heat_map_tc(df, ranges = c(seq(0, 100, by = 10), Inf))
```

**Arguments**

df	timecourse input
ranges	ranges for coloring scheme. Default set to c(seq(0, 100, by=10), Inf)

**Value**

heat map for timecourses

---

heat\_map\_tp                      *Preparatory function for heat map*

---

### Description

Returns heat map

### Usage

```
heat_map_tp(
  df,
  pv,
  sd,
  ranges = c(-Inf, seq(-30, 30, by = 10), Inf),
  pv_cutoff = 0.01,
  replicates = 3
)
```

### Arguments

df	average data frame. Generated using ave_timepoint() function.
pv	pvalues dataframes calculated using pv_timepoint() function
sd	standard deviation data.frame generated using sd_timepoint function
ranges	ranges for coloring scheme. Default set to c(-Inf, seq(-30, 30, by=10), Inf)
pv_cutoff	p-value cutoff here set up to 0.01
replicates	number of replicates in sample. Default set to 3.

### Value

heat map for timepoints

---

heat\_map\_tp\_maxuptake    *Preparatory function for heat map of maximum uptake per residue.*

---

### Description

Returns heat map

**Usage**

```
heat_map_tp_maxuptake(  
  df,  
  pv,  
  sd,  
  ranges = c(-Inf, seq(-30, 30, by = 10), Inf),  
  pv_cutoff = 0.01,  
  replicates = 3  
)
```

**Arguments**

df	average data frame. Generated using ave_timepoint() function.
pv	pvalues dataframes calculated using pv_timepoint() function
sd	standard deviation data.frame generated using sd_timepoint function
ranges	ranges for coloring scheme. Default set to c(-Inf, seq(-30, 30, by=10), Inf)
pv_cutoff	p-value cutoff here set up to 0.01
replicates	number of replicates in sample. Default set to 3.

**Value**

maximum uptake heat map for timepoints

---

heat\_map\_tp\_maxuptake\_proc

*Preparatory function for heat map of maximum percent deuteration per residue.*

---

**Description**

Returns heat map

**Usage**

```
heat_map_tp_maxuptake_proc(  
  df,  
  dfup,  
  pv,  
  sd,  
  ranges = c(-Inf, seq(-30, 30, by = 10), Inf),  
  pv_cutoff = 0.01,  
  replicates = 3  
)
```

**Arguments**

df	average data frame for procent deuteration. Generated using ave_timepoint() function.
dfup	average data frame for deuteration uptake. Generated using ave_timepoint() function.
pv	pvalues dataframes calculated using pv_timepoint() function
sd	standard deviation data.frame generated using sd_timepoint function
ranges	ranges for coloring scheme. Default set to c(-Inf, seq(-30, 30, by=10), Inf)
pv_cutoff	p-value cutoff here set up to 0.01
replicates	number of replicates in sample. Default set to 3.

**Value**

Maximum uptake heat map for timepoints

---

heat_map_tp_proc	<i>Preparatory function for heat map for procent deuteration</i>
------------------	--

---

**Description**

Returns heat map

**Usage**

```
heat_map_tp_proc(
  df,
  dfup,
  pv,
  sd,
  ranges = c(-Inf, seq(-30, 30, by = 10), Inf),
  pv_cutoff = 0.01,
  replicates = 3
)
```

**Arguments**

df	average data frame for procent deuteration. Generated using ave_timepoint() function.
dfup	average data frame for deuteration uptake. Generated using ave_timepoint() function.
pv	pvalues dataframes calculated using pv_timepoint() function
sd	standard deviation data.frame generated using sd_timepoint function
ranges	ranges for coloring scheme. Default set to c(-Inf, seq(-30, 30, by=10), Inf)
pv_cutoff	p-value cutoff here set up to 0.01
replicates	number of replicates in sample. Default set to 3.



**Value**

heat map for timepoints

---

is.nan.data.frame	<i>Checks for NaN is data.frame</i>
-------------------	-------------------------------------

---

**Description**

Function by Hong Ooi; <https://stackoverflow.com/questions/18142117/how-to-replace-nan-value-with-zero-in-a-huge-data-frame>

**Usage**

```
## S3 method for class 'data.frame'
is.nan(x)
```

**Arguments**

x                      Data frame to be checked for NaN

**Value**

logical. Returns info if data.frame contains NaNs.

**Examples**

```
## this function will overwrite the is.nan function that works only on vectors and matrices
df<-data.frame(c(0,NaN), c(1, 2))
is.nan(df)
df[is.nan(df)]<- 0
```

---

lab_dif	<i>Legend for difference in averages plot.</i>
---------	--

---

**Description**

Returns legend for difference in average plots. Preparatory function.

**Usage**

```
lab_dif(df, cola)
```

**Arguments**

df                      output from functions average difference  
cola                     color palette for different Protein States. As default Paired palette from color.Brewer is used.

**Value**

legend for difference in average plot for time points

---

lab_dif_proc	<i>Preparatory function for difference plot for procent deuteration</i>
--------------	---

---

**Description**

Returns legends for plots procent deuteration at each peptide.

**Usage**

```
lab_dif_proc(df, cola)
```

**Arguments**

df	output from functions output_tp or output_tp_proc.
cola	color palette for different Protein States. As default Paired palette from RColorBrewer is used.

**Value**

legends for procent deuteration plots

---

lab_vol	<i>Preparatory function for volcano plot legends</i>
---------	--

---

**Description**

Returns volcano plots

**Usage**

```
lab_vol(df, cola)
```

**Arguments**

df	output from functions output_tp
cola	color palette for different Protein States. As default Paired palette from color.Brewer is used.

**Value**

legends for volcano plots

---

legend\_heat\_map      *Legend for the heatmaps prep function.*

---

**Description**

Returns names for legend for the heatmaps

**Usage**

```
legend_heat_map(ranges = c(-Inf, seq(-30, 30, by = 10), Inf))
```

**Arguments**

ranges      ranges that are to be colored in the legend. Default ranges=c(-Inf,seq(-30, 30, by=10), Inf )

**Value**

legend for the heatmap

---

legend\_heat\_map\_tc      *Legend for the heatmaps for timecourses.*

---

**Description**

Returns names for legend for the heatmaps. Extracts names from data.frame

**Usage**

```
legend_heat_map_tc(df)
```

**Arguments**

df      generated using output\_tcourse()

**Value**

legend for the heatmap

---

legend\_heat\_map\_timecourse

*Legend for the heatmaps prep function for timecourses.*

---

### Description

Returns names for legend for the heatmaps

### Usage

```
legend_heat_map_timecourse(ranges = c(-Inf, seq(0, 100, by = 10), Inf))
```

### Arguments

ranges            ranges that are to be colored in the legend. Default ranges=c(-Inf,seq(-30, 30, by=10), Inf )

### Value

legend for the heatmap

---

legend\_heat\_map\_tp

*Legend for the heatmaps.Extracts names from data.frame*

---

### Description

Returns names for legend for the heatmaps

### Usage

```
legend_heat_map_tp(df)
```

### Arguments

df                average data frame. Generated using ave\_timepoint() function.

### Value

legend for the heatmap

### Examples

```
file_nm<-system.file("extdata", "All_results_table.csv", package = "HDXBoxeR")
a<- output_tp(file_nm)
legend_heat_map_tp(df=a)
```

---

legend\_heat\_map\_tp\_proc

*Legend for the heatmaps percent.Extracts names from data.frame*

---

**Description**

Returns names for legend for the heatmaps

**Usage**

legend\_heat\_map\_tp\_proc(df)

**Arguments**

df                    average data frame.

**Value**

legend for the heatmap percent

---

legend\_nm\_bottom

*Legend, bottom of the plots*

---

**Description**

Internal function

**Usage**

legend\_nm\_bottom(names, cols)

**Arguments**

names                labels  
cols                 colors

**Value**

legend at the bottom of the plot

---

legend_raw_ave	<i>Legend for average plot.</i>
----------------	---------------------------------

---

**Description**

Returns legend with average plots. Preparatory function.

**Usage**

```
legend_raw_ave(df, cola)
```

**Arguments**

df	output from functions output_tp or output_tp_proc.
cola	color palette for different Protein States. As default Paired palette from color.Brewer is used.

**Value**

legend for average plot for time points

---

legend_raw_ave_proc	<i>Preparatory function to draw legends for average procent</i>
---------------------	---

---

**Description**

Returns legend with average procent deuteration at each peptide.

**Usage**

```
legend_raw_ave_proc(df, cola)
```

**Arguments**

df	output from functions output_tp or output_tp_proc.
cola	color palette for different Protein States. As default Paired palette from color.Brewer is used.

**Value**

legend for average deuteration procent for timepoints

---

legend\_raw\_ave\_tc      *Legend for average deuteration plot for timecourse.*

---

**Description**

Returns legend with average plots. Preparatory function.

**Usage**

```
legend_raw_ave_tc(df, cola)
```

**Arguments**

df                      output from functions output\_tp or output\_tp\_proc.  
cola                    color palette for different Protein States. As default Paired palette from color.Brewer is used.

**Value**

legend for average plot for time course

---

legend\_sig\_peptides      *Legend for the significant peptides*

---

**Description**

Returns names for legend for the significant peptides plots.

**Usage**

```
legend_sig_peptides(ranges = c(-Inf, seq(-30, 30, by = 10), Inf))
```

**Arguments**

ranges                 ranges that are to be colored in the legend. Default ranges=c(-Inf,seq(-30, 30, by=10), Inf )

**Value**

legend for the heatmap

legend\_states\_PerD\_bottom

*Legend, bottom of the plots*

---

**Description**

Internal function

**Usage**

```
legend_states_PerD_bottom(df, cols)
```

**Arguments**

df	dataframe
cols	colors

**Value**

legend at the bottom of the plot

---

legend\_tc\_bottom

*Preparatory function returns legends for the timecourses.*

---

**Description**

Preparatory function

**Usage**

```
legend_tc_bottom(df, cols)
```

**Arguments**

df	data frame from which names will be extracted
cols	colors to be used in legend

**Value**

legend at the bottom of the plot



---

nm_states	<i>Lists names of states in data sets</i>
-----------	---

---

**Description**

Returns vector with name of states used for choosing states for input functions generation.

**Usage**

```
nm_states(filepath)
```

**Arguments**

filepath            filepath to the input file. Input file is All\_results table from HDX\_Examiner, where all the fields are marked for export.

**Value**

list of Protein States.

**Examples**

```
file_nm<-system.file("extdata", "All_results_table.csv", package = "HDXBoxer")
names_states<- nm_states(file_nm)
```

---

output_FD	<i>Prepares output for HDX-MS Full deuteration data</i>
-----------	---

---

**Description**

Returns a data frame for Full deuteration set

**Usage**

```
output_FD(filepath)
```

**Arguments**

filepath            filepath to the input file. Input file is All\_results table from HDX\_Examiner, where all the fields are marked for export.

**Value**

data frame with reorganized data where in columns is uptake data for Protein States.

**Examples**

```
file_nm<-system.file("extdata", "All_results_table.csv", package = "HDXBoxeR")
a<-output_FD(file_nm)
```

---

output_FD_proc	<i>Prepares output for HDX-MS Full deuteration data for procent deuteration.</i>
----------------	--

---

**Description**

Returns a data frame for Full deuteration set

**Usage**

```
output_FD_proc(filepath)
```

**Arguments**

filepath	filepath to the input file. Input file is All_results table from HDX_Examiner, where all the fields are marked for export.
----------	--

**Value**

data frame with reorganized data where in columns is procent deuteration for Protein States.

**Examples**

```
file_nm<-system.file("extdata", "All_results_table.csv", package = "HDXBoxeR")
a<- output_FD_proc(file_nm)
```

---

output_prep	<i>Prepares output with HDX-MS data for publications</i>
-------------	--

---

**Description**

Format prepared based of example from: Masson, G.R., Burke, J.E., Ahn, N.G. et al. Recommendations for performing, interpreting and reporting hydrogen deuterium exchange mass spectrometry (HDX-MS) experiments. Nat Methods 16, 595–602 (2019). <https://doi.org/10.1038/s41592-019-0459-y> It generates csv file in format ready for publication of the data.

**Usage**

```
output_prep(filepath, output_name, states, replicates, times, percent = FALSE)
```

**Arguments**

filepath	filepath to the input file. Input file is All_results table from HDX_Examiner, where all the fields are marked for export.
output_name	Name of output file. It has to be csv file
states	function allows to choose what states should be used for analysis. Default all states are used.
replicates	number of replicates to be used in analysis. The function takes number of replicates up to specified number. If no argument provided number maximal common number of replicates it used.
times	lists the deuteration times to be used in analysis. Default all states used.
percent	return either uptake or percent deuteration, default=FALSE, return uptake

**Value**

Returns&saves data.frame in format that is accepted for the publications.

**Examples**

```
file_nm<-system.file("extdata", "All_results_table.csv", package = "HDXBoxeR")
output_prep(filepath=file_nm, output_name=tempfile())
```

---

output_tc	<i>Prepares output for HDX-MS for the deuteration uptake or percent deuteration for the time courses.</i>
-----------	---

---

**Description**

Returns a data frame organized for additional analysis. In columns are deuteration uptake or percent deuteration data for the given protein states. Function allows for writing csv with data, matching sequences of peptide. Protein.States, Deut.times, or number of replicates can be specified.

**Usage**

```
output_tc(
  filepath,
  replicates,
  states,
  times,
  seq_match = FALSE,
  csv = "NA",
  percent = FALSE
)
```

**Arguments**

filepath	filepath to the input file. Input file is All_results table from HDX_Examiner, where all the fields are marked for export.
replicates	number of replicates to be used in analysis. The function takes number of replicates up to specified number. If no argument provided number maximal common number of replicates it used.
states	function allows to choose what states should be used for analysis. Default all states are used.
times	lists the deuteration times to be used in analysis. Default all states used.
seq_match	Flag allows to choose if the peptide sequences should be matched between states. seq_match=FALSE signifies no sequence matching, seq_match=TRUE states that the sequences are matched between the sets.
csv	Flag allowing saving the output as csv. With default csv="NA", data is not saved. If csv output is decided, provide output name.
percent	Flag allowing to choose output as deuterium uptake (FALSE) or percent deuteration (TRUE). Default deuteration uptake.

**Value**

data frame with reorganized data where in columns is the deuteration uptake for Protein States.

**Examples**

```
file_nm<-system.file("extdata", "All_results_table.csv", package = "HDXBoxeR")
a<- output_tc(filepath=file_nm) ###all default parameters used

# all possible flags listed & percent deuteration output,
#with sequences matching for protein states.

a<-output_tc(filepath=file_nm, replicates=3, states=c("bound", "Unbound"),
times=c("3.00s", "72000.00s"), seq_match=TRUE, csv="NA", percent=TRUE)
```

---

output\_tp

*Prepares output for HDX-MS for the deuteration uptake or percent deuteration for the time points.*

---

**Description**

Returns a data frame organized for additional analysis. In columns are deuteration uptake or percent deuteration data for the given protein states. Function allows for writing csv with data, matching sequences of peptide. Protein.States, Deut.times, or number of replicates can be specified.

**Usage**

```
output_tp(
  filepath,
  replicates,
  states,
  times,
  seq_match = FALSE,
  csv = "NA",
  percent = FALSE
)
```

**Arguments**

filepath	filepath to the input file. Input file is All_results table from HDX_Examiner, where all the fields are marked for export.
replicates	number of replicates to be used in analysis. The function takes number of replicates up to specified number. If no argument provided number maximal common number of replicates it used.
states	function allows to choose what states should be used for analysis. Default all states are used.
times	lists the deuteration times to be used in analysis. Default all states used.
seq_match	Flag allows to choose if the peptide sequences should be matched between states. seq_match=FALSE signifies no sequence matching, seq_match=T states that the sequences are matched between the sets.
csv	Flag allowing saving the output as csv. With default csv="NA", data is not saved. If csv output is decided, provide output name.
percent	Flag allowing to choose output as deuteration uptake (FALSE) or percent deuteration (TRUE). Default deuteration uptake.

**Value**

data frame with reorganized data where in columns is the deuteration uptake for Protein States.

**Examples**

```
file_nm<-system.file("extdata", "All_results_table.csv", package = "HDXBoxeR")
a<- output_tp(filepath=file_nm) ###all default parameters used

# all possible flags listed & percent deuteration output,
# with sequences matching for protein states.

a<-output_tp(filepath=file_nm, replicates=3, states=c("bound", "Unbound"),
times=c("3.00s", "72000.00s"), seq_match=TRUE, csv="NA", percent=TRUE)
```

---

output_UD	<i>Prepares output for HDX-MS Undeuterated sample data.</i>
-----------	---

---

**Description**

Returns a data frame for Full deuteration set

**Usage**

```
output_UD(filepath)
```

**Arguments**

filepath            filepath to the input file. Input file is All\_results table from HDX\_Examiner, where all the fields are marked for export.

**Value**

data frame with reorganized data where in columns is uptake data for Protein States.

**Examples**

```
file_nm<-system.file("extdata", "All_results_table.csv", package = "HDXBoxeR")  
a<- output_UD(file_nm)
```

---

output_UD_proc	<i>Prepares output for HDX-MS Undeuterated data for procent deuteration.</i>
----------------	--

---

**Description**

Returns a data frame for Undeuterated control set

**Usage**

```
output_UD_proc(filepath)
```

**Arguments**

filepath            filepath to the input file. Input file is All\_results table from HDX\_Examiner, where all the fields are marked for export.

**Value**

data frame with reorganized data where in columns is procent deuteration for Protein States.

**Examples**

```
file_nm<-system.file("extdata", "All_results_table.csv", package = "HDXBoxeR")  
a<- output_UD_proc(file_nm)
```

---

palette_legend	<i>Color scheme using heatmap. Legend Extracts names from data.frame</i>
----------------	--

---

**Description**

Returns names for legend for the heatmaps

**Usage**

```
palette_legend(col_palette)
```

**Arguments**

col\_palette     palette to be used in the heat map

**Value**

legend for the heatmap

---

palette_ll	<i>Color scheme using heatmap. Legend extracts names from data frame</i>
------------	--

---

**Description**

Returns names for legend for the heatmaps

**Usage**

```
palette_ll(palette, lab)
```

**Arguments**

palette     palette to be used in the heat map  
lab         labels to be used in palette

**Value**

legend for the heatmap

---

peptide_pv_tp	<i>Preparatory function for significant peptide plots</i>
---------------	---

---

### Description

Returns plot where significant peptides are colored in blue-red scheme.

### Usage

```
peptide_pv_tp(
  df,
  pv,
  sd,
  nb_row,
  ranges = c(-Inf, seq(-30, 30, by = 10), Inf),
  pv_cutoff = 0.01,
  replicates = 3
)
```

### Arguments

df	average data frame. Generated using ave_timepoint() function.
pv	pvalues dataframes calculated using pv_timepoint() function
sd	standard deviation data.frame generated using sd_timepoint function
nb_row	number of peptides in each row. Plotting parameter.
ranges	ranges for coloring scheme. Default set to c(-Inf, seq(-30, 30, by=10), Inf)
pv_cutoff	p-value cutoff here set up to 0.01
replicates	number of replicates in sample. Default set to 3.

### Value

plot with peptides which are significantly different between sets.

---

peptide_pv_tp_proc	<i>Preparatory function for showing peptides with significant differences between sets.</i>
--------------------	---

---

### Description

Returns plot where significantly different peptides are colored in blue-red scheme.



**Usage**

```
peptide_pv_tp_proc(
  df,
  dfup,
  pv,
  sd,
  nb_row = 100,
  ranges = c(-Inf, seq(-30, 30, by = 10), Inf),
  pv_cutoff = 0.01,
  replicates = 3
)
```

**Arguments**

df	average data frame for procent deuteration. Generated using ave_timepoint() function.
dfup	average data frame for deuteration uptake. Generated using ave_timepoint() function.
pv	pvalues dataframes calculated using pv_timepoint() function
sd	standard deviation data.frame generated using sd_timepoint function
nb_row	number of peptides in each row. Plotting parameter.
ranges	ranges for coloring scheme. Default set to c(-Inf, seq(-30, 30, by=10), Inf)
pv_cutoff	p-value cutoff here set up to 0.01
replicates	number of replicates in sample. Default set to 3.

**Value**

plot with peptides which are significantly different between sets.

---

plots\_av\_tcourse      *Generates average deuteration plot for the time-course.*

---

**Description**

Returns plots with average deuteration at each peptide.

**Usage**

```
plots_av_tcourse(df, replicates = 3, cola)
```

**Arguments**

df	output from functions output_tcourse or output_tcourse_proc.
replicates	number of replicates in set as default set to 3.
cola	color palette for different Protein States. As default Paired palette from RColorBrewer is used.

**Value**

average deuteration plots

**Examples**

```
file_nm<-system.file("extdata", "All_results_table.csv", package = "HDXBoxeR")
a<- output_tc(file_nm)
plots_av_tcourse(df=a, replicates=3, cola=c(1:4))
plots_av_tcourse(df=a)
```

---

plots\_av\_tp

*Returns average deuteration plot for timepoints in the data frame*

---

**Description**

Returns plots with average deuteration at each peptide.

**Usage**

```
plots_av_tp(df, replicates = 3, cola)
```

**Arguments**

df	output from functions output_tp or output_tp_proc.
replicates	number of replicates in set as default set to 3.
cola	color palette for different Protein States. As default Paired palette from color.Brewer is used.

**Value**

average deuteration plots

**Examples**

```
file_nm<-system.file("extdata", "All_results_table.csv", package = "HDXBoxeR")
a<- output_tp(file_nm)
plots_av_tp(df=a, replicates=3, cola=c(1:4))
plots_av_tp(df=a)
```

---

plots_av_tp_proc	<i>Returns average procent deuteration plot for time points</i>
------------------	---

---

**Description**

Returns plots with average procent deuteration at each peptide.

**Usage**

```
plots_av_tp_proc(df, replicates = 3, cola)
```

**Arguments**

df	output from functions output_tp_proc.
replicates	number of replicates in set as default set to 3.
cola	color pallete for different Protein States. As default Paired pallete from RColorBrewer is used.

**Value**

average deuteration plots

**Examples**

```
file_nm<-system.file("extdata", "All_results_table.csv", package = "HDXBoxeR")
a<- output_tp(file_nm, percent=TRUE)
plots_av_tp_proc(df=a, replicates=3, cola=c(1:4))
plots_av_tp_proc(df=a)
```

---

plots_diff_tp	<i>Returns difference in average plot for timepoints in the data frame</i>
---------------	--

---

**Description**

Returns plots with difference in avarage for each peptide.

**Usage**

```
plots_diff_tp(df, replicates = 3, cola)
```

**Arguments**

df	output from functions output_tp or output_tp_proc.
replicates	number of replicates in set as default set to 3.
cola	color pallete for different Protein States. As default Paired pallete from color.Brewer is used.

**Value**

plots of difference of averages

**Examples**

```
file_nm<-system.file("extdata", "All_results_table.csv", package = "HDXBoxeR")
a<- output_tp(file_nm)
plots_diff_tp(df=a, replicates=3, cola=c(1:4))
plots_diff_tp(df=a)
```

---

plots_diff_tp_proc	<i>Returns difference in average percent deuteration plot for timepoints in the data frame</i>
--------------------	--

---

**Description**

Returns plots with difference in percent deuteration for each peptide.

**Usage**

```
plots_diff_tp_proc(df, replicates = 3, cola)
```

**Arguments**

df	output from functions output_tp_proc.
replicates	number of replicates in set as default set to 3.
cola	color palette for different Protein States. As default Paired palette from color.Brewer is used.

**Value**

plots of difference of average percent deuteration

**Examples**

```
file_nm<-system.file("extdata", "All_results_table.csv", package = "HDXBoxeR")
a<- output_tp(file_nm, percent=TRUE)
plots_diff_tp_proc(df=a, replicates=3, cola=c(1:4))
plots_diff_tp_proc(df=a)
```

---

plots_vol_tp	<i>Returns volcano plots for timepoints in the data frame</i>
--------------	---

---

**Description**

Returns volcano plots for each peptide. Critical interval is calculated according to #' Reliable Identification of Significant Differences in Differential Hydrogen Exchange-Mass Spectrometry Measurements Using a Hybrid Significance Testing Approach Tyler S. Hageman and David D. Weis Analytical Chemistry 2019 91 (13), 8008-8016 DOI: 10.1021/acs.analchem.9b01325 calculations for alpha 0.99 pvalues calculated using Welch t-test.

**Usage**

```
plots_vol_tp(df, replicates = 3, pv_cutoff = 0.01, cola)
```

**Arguments**

df	output from functions output_tp
replicates	number of replicates in set as default set to 3.
pv_cutoff	p-value cutoff here set up to 0.01
cola	color palette for different Protein States. As default Paired palette from color.Brewer is used.

**Value**

volcano plots

**Examples**

```
file_nm<-system.file("extdata", "All_results_table.csv", package = "HDXBoxeR")
a<- output_tp(file_nm)
plots_vol_tp(df=a, replicates=3, cola=c(1:4), pv_cutoff=0.01 )
plots_vol_tp(df=a, pv_cutoff=0.05)
```

---

plot_heat_map_max_uptake_tp	<i>Plots heat maps for maximum uptake per residue.</i>
-----------------------------	--

---

**Description**

Returns heat map with maximum uptake per residue.

**Usage**

```
plot_heat_map_max_uptake_tp(
  df,
  replicates = 3,
  mar_x = 3.5,
  ranges = c(-Inf, seq(-30, 30, by = 10), Inf),
  pv_cutoff = 0.01
)
```

**Arguments**

df	average data frame. Generated using ave_timepoint() function.
replicates	number of replicates in sample. Default set to 3.
mar_x	margin x width. Default=3.5
ranges	ranges for coloring scheme. Default set to c(-Inf, seq(-30, 30, by=10), Inf)
pv_cutoff	p-value cutoff here set up to 0.01

**Value**

heat map for maximum uptake per residue

**Examples**

```
file_nm<-system.file("extdata", "All_results_table.csv", package = "HDXBoxeR")
a<- output_tp(file_nm)
plot_heat_map_max_uptake_tp(df=a, replicates=3, pv_cutoff=0.01,
  ranges=c(-Inf,-40, -30,-20,-10, 0,10, 20,30,40, Inf) )
plot_heat_map_max_uptake_tp(df=a)
```

---

plot\_heat\_map\_max\_uptake\_tp\_proc

*Plots heat maps for maximum percent deuteration per residue.*

---

**Description**

Returns heat map with maximum percent deuteration per residue.

**Usage**

```
plot_heat_map_max_uptake_tp_proc(
  input_proc,
  input_up,
  mar_x = 3.5,
  ranges = c(-Inf, seq(-30, 30, by = 10), Inf),
  pv_cutoff = 0.01,
  replicates = 3
)
```

**Arguments**

input_proc	Dataframe with organized procent deuteration data. Input generated using output_tp_proc() function.
input_up	Dataframe with organized deuteration uptake. Input generated using output_tp() function.
mar_x	margin x width. Default=3.5
ranges	ranges for coloring scheme. Default set to c(-Inf, seq(-30, 30, by=10), Inf)
pv_cutoff	p-value cutoff here set up to 0.01
replicates	number of replicates in sample. Default set to 3.

**Value**

heat map for average uptake per residue for significant peptides.

**Examples**

```
file_nm<-system.file("extdata", "All_results_table.csv", package = "HDBoxeR")
a_up<- output_tp(file_nm)
a_proc<- output_tp(file_nm, percent=TRUE)
plot_heat_map_max_uptake_tp_proc(input_proc=a_proc, input_up=a_up, replicates=3, pv_cutoff=0.01,
ranges=c(-Inf,-40, -30,-20,-10, 0,10, 20,30,40, Inf) )
plot_heat_map_max_uptake_tp_proc(input_proc=a_proc, input_up=a_up)
```

---

plot\_heat\_map\_tc      *Plots heat maps for time courses.*

---

**Description**

Returns heat map on timecourses with raw data.

**Usage**

```
plot_heat_map_tc(
  df,
  replicates = 3,
  mar_x = 3.5,
  ranges = c(-Inf, seq(0, 100, by = 10), Inf)
)
```

**Arguments**

df	output from function output_tcourse
replicates	number of replicates in sample. Default set to 3.
mar_x	margin x width. Default=3.5
ranges	ranges for coloring scheme. Default set to c(seq(0, 100, by=10), Inf)

**Value**

heat map for time courses

**Examples**

```
file_nm<-system.file("extdata", "All_results_table.csv", package = "HDXBoxeR")
a<- output_tc(file_nm)
plot_heat_map_tc(df=a, replicates=3, ranges=c(seq(0, 100, by=5), Inf))
plot_heat_map_tc(df=a)
```

---

plot_heat_map_tp	<i>Plots heat maps for significant peptides.</i>
------------------	--

---

**Description**

Returns heat map with average values for significant uptake per residue.

**Usage**

```
plot_heat_map_tp(
  df,
  mar_x = 3.5,
  ranges = c(-Inf, seq(-30, 30, by = 10), Inf),
  pv_cutoff = 0.01,
  replicates = 3
)
```

**Arguments**

df	average data frame. Generated using ave_timepoint() function.
mar_x	margin x width. Default=3.5
ranges	ranges for coloring scheme. Default set to c(-Inf, seq(-30, 30, by=10), Inf)
pv_cutoff	p-value cutoff here set up to 0.01
replicates	number of replicates in sample. Default set to 3.

**Value**

heat map for average uptake per residue for significant peptides.

**Examples**

```
file_nm<-system.file("extdata", "All_results_table.csv", package = "HDXBoxeR")
a<- output_tp(file_nm)
plot_heat_map_tp(df=a, replicates=3, pv_cutoff=0.01,
  ranges=c(-Inf,-40, -30,-20,-10, 0,10, 20,30,40, Inf) )
plot_heat_map_tp(df=a)
```



---

plot\_heat\_map\_tp\_proc *Plots heat maps for significant peptides.*

---

## Description

Returns heat map with average values for significant uptake per residue.

## Usage

```
plot_heat_map_tp_proc(  
  input_proc,  
  input_up,  
  mar_x = 3.5,  
  ranges = c(-Inf, -3, -2, -1, 0, 1, 2, 3, Inf),  
  pv_cutoff = 0.01,  
  replicates = 3  
)
```

## Arguments

input_proc	Dataframe with organized procent deuteration data. Input generated using output_tp_proc() function.
input_up	Dataframe with organized deuteration uptake. Input generated using output_tp() function.
mar_x	margin x width. Default=3.5
ranges	ranges for coloring scheme. Default set to c(-Inf, seq(-30, 30, by=10), Inf)
pv_cutoff	p-value cutoff here set up to 0.01
replicates	number of replicates in sample. Default set to 3.

## Value

heat map for average uptake per residue for significant peptides.

## Examples

```
file_nm<-system.file("extdata", "All_results_table.csv", package = "HDXBoxeR")  
a_up<- output_tp(file_nm)  
a_proc<- output_tp(file_nm, percent=TRUE)  
plot_heat_map_tp_proc(input_proc=a_proc, input_up=a_up, replicates=3, pv_cutoff=0.01,  
  ranges=c(-Inf,-40, -30,-20,-10, 0,10, 20,30,40, Inf) )  
plot_heat_map_tp_proc(input_proc=a_proc, input_up=a_up)
```

---

plot\_peptide\_sig\_tp     *Significant peptide plots.*

---

**Description**

Returns plot where significant peptides are colored in blue-red scheme.

**Usage**

```
plot_peptide_sig_tp(  
  df1,  
  replicates = 3,  
  nb_pep_row = 100,  
  ranges = c(-Inf, seq(-30, 30, by = 10), Inf),  
  pv_cutoff = 0.01  
)
```

**Arguments**

df1	average data frame. Generated using ave_timepoint() function.
replicates	number of replicates in sample. Default set to 3.
nb_pep_row	number of peptides in each row. Plotting parameter. Default set to 100.
ranges	ranges for coloring scheme. Default set to c(-Inf, seq(-30, 30, by=10), Inf)
pv_cutoff	p-value cutoff here set up to 0.01

**Value**

plot with peptides which are significantly different between sets.

---

plot\_peptide\_sig\_tp\_proc  
                          *Draws peptides with significant difefrences between sets.*

---

**Description**

Returns plot where significant peptides are colored in blue-red scheme.

**Usage**

```
plot_peptide_sig_tp_proc(  
  input_proc,  
  input_up,  
  nb_pep_row = 100,  
  ranges = c(-Inf, seq(-30, 30, by = 10), Inf),  
  pv_cutoff = 0.01,  
  replicates = 3  
)
```

**Arguments**

input_proc	Dataframe with organized procent deuteration data. Input generated using output_tp_proc() function.
input_up	Dataframe with organized deuteration uptake. Input generated using output_tp() function.
nb_pep_row	number of peptides in each row. Plotting parameter. Default set to 100.
ranges	ranges for coloring scheme. Default set to c(-Inf, seq(-30, 30, by=10), Inf)
pv_cutoff	p-value cutoff here set up to 0.01
replicates	number of replicates in sample. Default set to 3.

**Value**

plot with peptides which are significantly different between sets.

**Examples**

```
file_nm<-system.file("extdata", "All_results_table.csv", package = "HDXBoxeR")
a_up<- output_tp(file_nm)
a_proc<- output_tp(file_nm, percent=TRUE)
plot_peptide_sig_tp_proc(input_proc=a_proc, input_up=a_up, replicates=3, pv_cutoff=0.01,
ranges=c(-Inf,-40, -30,-20,-10, 0,10, 20,30,40, Inf), nb_pep_row=40 )
```

---

pl\_gen\_ch2

*Prepares the plot window for the woods functions*

---

**Description**

Internal function

**Usage**

```
pl_gen_ch2(df, ddlab = 1, ...)
```

**Arguments**

df	dataframe
ddlab	label
...	other

**Value**

Plot window

---

pl_gen_uptake	<i>Prepares the plot window for the woods functions</i>
---------------	---

---

**Description**

Internal function

**Usage**

```
pl_gen_uptake(df, timepoints, ddlab = 1, ...)
```

**Arguments**

df	dataframe
timepoints	deuteration times used
ddlab	label
...	other

**Value**

Plot window

---

ppar	<i>Preparation of figure window.</i>
------	--------------------------------------

---

**Description**

Prepares a plotting window with specified margins with specific number of figure row and columns.

**Usage**

```
ppar(mfrow2)
```

**Arguments**

mfrow2	mfrow: number of Multiple Figures (use ROW-wise).
--------	---

**Value**

modified par function with adjusted parameters

**Examples**

```
ppar(c(2,1))
```

---

pparLM	<i>Preparation of figure window. small margins</i>
--------	--

---

**Description**

Prepares a plotting window with specified margins with specific number of figure row and columns.

**Usage**

```
pparLM(mfrow2)
```

**Arguments**

mfrow2            mfrow: number of Multiple Figures (use ROW-wise).

**Value**

modified par function with adjusted parameters

**Examples**

```
pparLM(c(2,1))
```

---

ppar_bottom_legend	<i>Preparation of figure window with area for figure at the bottom.</i>
--------------------	---

---

**Description**

Prepares a plotting window with specified margins with specific number of figure row and columns.

**Usage**

```
ppar_bottom_legend(mfrow2)
```

**Arguments**

mfrow2            mfrow: number of Multiple Figures (use ROW-wise).

**Value**

modified par function with adjusted parameters

**Examples**

```
ppar_bottom_legend(c(2,3))
```

---

ppar_wider	<i>Preparation of figure window with more area on west side of plot.</i>
------------	--

---

**Description**

Prepares a plotting window with specified margins with specific number of figure row and columns.

**Usage**

```
ppar_wider(mfrow2)
```

**Arguments**

mfrow2            mfrow: number of Multiple Figures (use ROW-wise).

**Value**

default plotting window

**Examples**

```
ppar_wider(c(2,1))
```

---

prep_timecourse_plot_ave	<i>Prepares function for plotting averages in timecourse</i>
--------------------------	--

---

**Description**

Preparatory function

**Usage**

```
prep_timecourse_plot_ave(control_df, variant_df, replicates = 3)
```

**Arguments**

control\_df        dataframe of control  
variant\_df        dataframe for variant  
replicates        number of replicates. Default set to 3.

**Value**

dataframes with matched peptides in time course

---

```
prep_timecourse_plot_sd
```

*Prepares function for Critical interval for timecourses*

---

### Description

Preparatory function

### Usage

```
prep_timecourse_plot_sd(
  control_df_up,
  variant_df_up,
  replicates = 3,
  pv_cutoff = 0.01
)
```

### Arguments

control_df_up	dataframe of control
variant_df_up	dataframe for variant
replicates	number of replicates. Default set to 3.
pv_cutoff	cut off of pvalue used in calculation of critical interval. Default set to 0.01

### Value

Critical interval for all sets

---

```
pv_timecourse
```

*pvalue calculation between two sets of the data at certain timepoint*

---

### Description

Preparatory function for calculation of pvalue between sets.

### Usage

```
pv_timecourse(df_c, df_v, replicates = 3)
```

### Arguments

df_c	dataframe of control
df_v	dataframe for variant
replicates	number of replicates. Default set to 3.

**Value**

pvalue comparisons between two sets.

---

pv_timepoint	<i>Calculation of pvalue between first protein state and any other state from all_states file</i>
--------------	---

---

**Description**

Compares means of sets of uptake data and return dataframe with pvalues. Welch t.test is used for analysis. Sets are compared to the first state in the input file. If other order of the sets is required use Default for the number of replicates is 3.

**Usage**

```
pv_timepoint(df, replicates = 3)
```

**Arguments**

df	output from functions output_tp or output_tp_proc.
replicates	number of replicates used. Default is set to replicates=3

**Value**

Data.frame with p-values

**Examples**

```
file_nm<-system.file("extdata", "All_results_table.csv", package = "HDXBoxeR")
a<- output_tp(file_nm)
pv<-pv_timepoint(df=a) ##if number of replicates is equal 3
# pv1<-pv_timepoint(df=a, replicates=4) ##if number of replicates is equal 4
#b<-output_tp_states(file_nm, states=c("State4", "State2", "State3" ))
#pv_states<-pv_timepoint(df=b) ### here means of State4, will be compared to State2 and State4
```

---

pymol\_script\_average\_residue

*Writes a text files with pymol scripts to list significant residues.*

---

**Description**

Function write a script that can be used in pymol to color structure. Number of colors and corresponding to them ranges can be defined by user. Residues are being colored by average uptake values from the significant peptides per residues.



**Usage**

```
pymol_script_average_residue(  
  df,  
  path = "",  
  ranges = c(-Inf, seq(-30, 30, by = 10), Inf),  
  pv_cutoff = 0.01,  
  replicates = 3  
)
```

**Arguments**

df	output from functions output_tp
path	output folder location
ranges	ranges for coloring scheme. Default set to c(-Inf, seq(-30, 30, by=10), Inf)
pv_cutoff	p-value cutoff here set up to 0.01
replicates	number of replicates in sample. Default set to 3.

**Value**

pymol script with residues colored based on average of uptake per residue.

**Examples**

```
file_nm<-system.file("extdata", "All_results_table.csv", package = "HDXBoxeR")  
a<- output_tp(file_nm)  
pymol_script_average_residue(df=a, replicates=3, pv_cutoff=0.01,  
  ranges=c(-Inf,-40, -30,-20,-10, 0,10, 20,30,40, Inf), path=tempdir() )  
pymol_script_average_residue(df=a, path=tempdir())
```

---

pymol\_script\_significant\_peptide

*Writes a text files with pymol scripts to list significant peptides*

---

**Description**

Function write a script that can be used in pymol to color structure. Number of colors and corresponding to them ranges can be defined by user.

**Usage**

```
pymol_script_significant_peptide(  
  df,  
  path = "",  
  ranges = c(-Inf, seq(-30, 30, by = 10), Inf),  
  pv_cutoff = 0.01,  
  replicates = 3,  
  order.pep = TRUE  
)
```

**Arguments**

df	output from functions output_tp
path	location where the scripts will be saved
ranges	ranges for coloring scheme. Default set to c(-Inf, seq(-30, 30, by=10), Inf)
pv_cutoff	p-value cutoff here set up to 0.01
replicates	number of replicates in sample. Default set to 3.
order.pep	flag allowing to either order peptide according to the peptide length (default), or to position in the protein sequence.

**Value**

pymol script with colors assigned per peptide

**Examples**

```
file_nm<-system.file("extdata", "All_results_table.csv", package = "HDXBoxeR")  
a<- output_tp(file_nm)  
pymol_script_significant_peptide(df=a, replicates=3, path=tempdir(), pv_cutoff=0.01,  
  ranges=c(-Inf,-40, -30,-20,-10, 0,10, 20,30,40, Inf), order.pep=TRUE )  
pymol_script_significant_peptide(df=a, path=tempdir())
```

---

pymol\_script\_significant\_peptide\_proc

*Writes a text files with pymol scripts to list significant peptides*

---

**Description**

Function write a script that can be used in pymol to color structure. Number of colors and corresponding to them ranges can be defined by user.

**Usage**

```
pymol_script_significant_peptide_proc(
  input_proc,
  input_up,
  path = "",
  ranges = c(-Inf, seq(-30, 30, by = 10), Inf),
  pv_cutoff = 0.01,
  replicates = 3,
  order.pep = TRUE
)
```

**Arguments**

input_proc	Dataframe with organized procent deuteration data. Input generated using output_tp(, percent=T) function.
input_up	Dataframe with organized deuteration uptake. Input generated using output_tp() function.
path	location where the Pymol scripts will be saved
ranges	ranges for coloring scheme. Default set to c(-Inf, seq(-30, 30, by=10), Inf)
pv_cutoff	p-value cutoff here set up to 0.01
replicates	number of replicates in sample. Default set to 3.
order.pep	flag allowing to either order peptide according to the peptide length (default), or to position in the protein sequence.

**Value**

pymol script with colors assigned per peptide

**Examples**

```
file_nm<-system.file("extdata", "All_results_table.csv", package = "HDXBoxeR")
a_up<- output_tp(file_nm)
a_proc<- output_tp(file_nm, percent=TRUE)
pymol_script_significant_peptide_proc(input_proc=a_proc,
  input_up=a_up, path=tempdir(),replicates=3, pv_cutoff=0.01,
  ranges=c(-Inf,-40, -30,-20,-10, 0,10, 20,30,40, Inf), order.pep=TRUE)
```

---

pymol\_script\_significant\_residue

*Writes a text files with pymol scripts to list significant residues.*

---

**Description**

Function write a script that can be used in pymol to color structure. Number of colors and corresponding to them ranges can be defined by user. Residues are being colored by maximum uptake from significant peptides per residues.

**Usage**

```
pymol_script_significant_residue(
  df,
  path = "",
  ranges = c(-Inf, seq(-30, 30, by = 10), Inf),
  pv_cutoff = 0.01,
  replicates = 3
)
```

**Arguments**

df	average data frame. Generated using ave_timepoint() function.
path	location where the Pymol scripts will be saved
ranges	ranges for coloring scheme. Default set to c(-Inf, seq(-30, 30, by=10), Inf)
pv_cutoff	p-value cutoff here set up to 0.01
replicates	number of replicates in sample. Default set to 3.

**Value**

pymol script with colors assigned per residues by maximum uptake per residue

**Examples**

```
file_nm<-system.file("extdata", "All_results_table.csv", package = "HDXBoxeR")
a<- output_tp(file_nm)
pymol_script_significant_residue(df=a, path=tempdir(), replicates=3, pv_cutoff=0.01,
ranges=c(-Inf,-40, -30,-20,-10, 0,10, 20,30,40, Inf) )
pymol_script_significant_residue(df=a, path=tempdir())
```

---

pymol\_script\_significant\_residue\_proc

*Writes a text files with pymol scripts to list significant residues.*

---

**Description**

Function write a script that can be used in pymol to color structure. Number of colors and corresponding to them ranges can be defined by user. Residues are colored by average percent\_deuteration from the significant peptides per residues.

**Usage**

```
pymol_script_significant_residue_proc(
  input_up,
  input_proc,
  path = "",
  ranges = c(-Inf, seq(-30, 30, by = 10), Inf),
  pv_cutoff = 0.01,
  replicates = 3
)
```

**Arguments**

input_up	Dataframe with organized deuteration uptake. Input generated using output_tp() function.
input_proc	Dataframe with organized percent deuteration data. Input generated using output_tp_proc() function.
path	location where the Pymol scripts will be saved
ranges	ranges for coloring scheme. Default set to c(-Inf, seq(-30, 30, by=10), Inf)
pv_cutoff	p-value cutoff here set up to 0.01
replicates	number of replicates in sample. Default set to 3.

**Value**

pymol script with residues colored based on average of percent deuteration per residue.

**Examples**

```
file_nm<-system.file("extdata", "All_results_table.csv", package = "HDXBoxeR")
a_up<- output_tp(file_nm)
a_proc<- output_tp(file_nm, percent=TRUE)
pymol_script_significant_residue_proc(input_proc=a_proc,
input_up=a_up, path=tempdir(), replicates=3, pv_cutoff=0.01,
ranges=c(-Inf,-40, -30,-20,-10, 0,10, 20,30,40, Inf))
```

---

pymol\_str

*Preparatory function writing pymol scripts*


---

**Description**

Function rearrange vector to string by adding + sign between the numbers.

**Usage**

```
pymol_str(ind1)
```

**Arguments**

ind1                vector of numbers (residues)

**Value**

string with + as a separator.

**Examples**

```
res<-c(1,5, 19, 100, 109)
pymol_str(res)
```

---

qpcr.cbind.na                *Hidden function from qpcR package, typical usage as qpcR:::cbind.na*

---

**Description**

Combine data of unequal row length avoiding repetition or errors by filling with NAs. In contrast to classical cbind, cbind.na can be used to combine data such as

**Usage**

```
qpcr.cbind.na(..., deparse.level = 1)
```

**Arguments**

...                vectors  
deparse.level    set to 1 as default

**Value**

data frame with NA

**Examples**

```
qpcr.cbind.na(1:10, 1:3)
```

---

ranges_function	<i>Gives ranges for the averages</i>
-----------------	--------------------------------------

---

**Description**

Function used as internal function to get ranges in the function.

**Usage**

```
ranges_function(df_ave, values_df)
```

**Arguments**

df_ave	average per residues
values_df	data frame with values.

**Value**

ranges per set

---

ranges_function_tc	<i>Gives ranges for the averages for time course analysis</i>
--------------------	---

---

**Description**

Function used as internal function to get ranges in the function.

**Usage**

```
ranges_function_tc(df_ave, values_df)
```

**Arguments**

df_ave	average per residues
values_df	data frame with values.

**Value**

ranges per set

---

rbind_na	<i>bind non equal row</i>
----------	---------------------------

---

**Description**

kmezhou/canceR: A Graphical User Interface for accessing and modeling the Cancer Genomics Data of MSKCC <https://rdrr.io/github/kmezhou/canceR/src/R/rbind.na.R>

**Usage**

```
rbind_na(..., deparse.level = 1)
```

**Arguments**

... (generalized) vectors or matrices.  
deparse.level integer controlling the construction of labels in the case of non-matrix-like arguments (for the default method): deparse.level = 0 constructs no labels; the default, deparse.level = 1 or 2 constructs labels from the argument names.

**Value**

a data frame with merged rows

**Examples**

```
row1 <- c("a", "b", "c", "d")  
row2 <- c("A", "B", "C")  
row3 <- rbind_na(row1, row2)
```

---

reset_par	<i>Reset plotting window parameters to default</i>
-----------	--

---

**Description**

function by Farid Cheraghi, <https://stackoverflow.com/questions/9292563/reset-the-graphical-parameters-back-to-default-values-without-use-of-dev-off> function resets plotting window parameters

**Usage**

```
reset_par()
```

**Value**

default plotting window parameters

**Examples**

```
reset_par()
```



---

robot\_2states\_indexes *Returns a robot plot for selected peptides for 2 protein states.*

---

### Description

Modification of butterfly plot. x axis residues. y axis % deuteration for one variant above the axis and for second peptide below the axis. Peptides are compared between the sets for the significance change between sets. If there is significant change between sets peptides are plotted for all time-points. Significantly different timepoints for the peptides are colored. Peptides ranges are plotted as a line at corresponding % deuteration values.

### Usage

```
robot_2states_indexes(  
  thP,  
  th,  
  indexes,  
  states,  
  replicates = 3,  
  pvalue = 0.01,  
  ylim,  
  xlim,  
  CI_factor = 1  
)
```

### Arguments

thP	output of output_tcourse_proc() function. Raw data for percent deuteration for time courses
th	output of output_tcourse() function. Raw data for uptake deuteration for time courses
indexes	indexes of peptides to be drawn.
states	Need to choose only two protein states
replicates	number of replicates in sample. Default set to 3.
pvalue	p-value cutoff here set up to 0.01
ylim	y-axis range
xlim	x-axis range. Set as default from max and minimum residues for the protein
CI_factor	Multiplication factor for Critical Interval. Allows for more restrictive selection of Critical interval.

### Value

Robot maps for timecourses for 2 protein states and selected indexes.

**Examples**

```

file_nm<-system.file("extdata", "All_results_table.csv", package = "HDXBoxeR")
tm_df<-output_tc(filepath=file_nm)
tmP_df<-output_tc(filepath=file_nm, percent=TRUE)
names_states<- nm_states(file_nm) ### returns states names
ind1<-robot_indexes(thP = tmP_df, th=tm_df, pvalue=0.001, CI_factor=3, states=names_states[1:2])
robot_2states_indexes(thP = tmP_df, th=tm_df,
  states=names_states[1:2], indexes =ind1, pvalue=0.001, CI_factor=3)

```

---

robot_indexes	<i>Returns indexes for peptides with significant difference between two sets</i>
---------------	--

---

**Description**

Function to help decide which peptides will be drawn on Robot plots.

**Usage**

```
robot_indexes(thP, th, replicates = 3, pvalue = 0.01, states, CI_factor = 1)
```

**Arguments**

thP	output of output_tcourse_proc() function. Raw data for percent deuteration for time courses
th	output of output_tcourse() function. Raw data for uptake deuteration for time courses
replicates	number of replicates in sample. Default set to 3.
pvalue	p-value cutoff. Default set up to 0.01
states	Protein states from the set. As default all states are chosen.
CI_factor	Multiplication factor for Critical Interval. Allows for more restrictive selection of Critical interval.

**Value**

Returns indexes of significant peptides

**Examples**

```

file_nm<-system.file("extdata", "All_results_table.csv", package = "HDXBoxeR")
tm_df<-output_tc(filepath=file_nm)
tmP_df<-output_tc(filepath=file_nm, percent=TRUE)

# more restrictive peptide selection
robot_indexes(thP = tmP_df, th=tm_df, pvalue=0.001, CI_factor=3)

```

---

robot_indexes_df	<i>Returns dataframe with peptides which exhibit significant difference between two sets</i>
------------------	--

---

## Description

Function to help decide which peptides will be drawn on Robot plots.

## Usage

```
robot_indexes_df(thP, th, replicates = 3, pvalue = 0.01, states, CI_factor = 1)
```

## Arguments

thP	output of output_tcourse_proc() function. Raw data for percent deuteration for time courses
th	output of output_tcourse() function. Raw data for uptake deuteration for time courses
replicates	number of replicates in sample. Default set to 3.
pvalue	p-value cutoff. Default set up to 0.01
states	Protein states from the set. As default all states are chosen.
CI_factor	Multiplication factor for Critical Interval. Allows for more restrictive selection of Critical interval.

## Value

Returns dataframe listing peptides that are significantly different between sets.

## Examples

```
file_nm<-system.file("extdata", "All_results_table.csv", package = "HDXBoxeR")
tm_df<-output_tc(filepath=file_nm)
tmP_df<-output_tc(filepath=file_nm, percent=TRUE)

# more restrictive peptide selection
robot_indexes_df(thP = tmP_df, th=tm_df, pvalue=0.001, CI_factor=3)
```

---

robot_plot_All	<i>Returns a robot plot for comparisons of the timepoints samples</i>
----------------	---

---

### Description

Modification of butterfly plot. x axis residues. y axis % deuteration for one variant above the axis and for second peptide below the axis. Peptides are compared between the sets for the significance change between sets. If there is significant change between sets peptides are plotted for all time-points. Significantly different timepoints for the peptides are colored. Peptides ranges are plotted as a line at corresponding % deuteration values.

### Usage

```
robot_plot_All(
  thP,
  th,
  replicates = 3,
  pv_cutoff = 0.01,
  states,
  CI_factor = 1
)
```

### Arguments

thP	output of output_tcourse_proc() function. Raw data for procent deuteration for time courses
th	output of output_tcourse() function. Raw data for uptake deuteration for time courses
replicates	number of replicates in sample. Default set to 3.
pv_cutoff	p-value cutoff here set up to 0.01
states	Protein states from the set. As default all states are chosen.
CI_factor	Multiplication factor for Critical Interval. Allows for more restrictive selection of Critical interval.

### Value

Robot maps for timecourses

### Examples

```
file_nm<-system.file("extdata", "All_results_table.csv", package = "HDXBoxeR")
tm_df<-output_tc(filepath=file_nm)
tmP_df<-output_tc(filepath=file_nm, percent=TRUE)
robot_plot_All(thP = tmP_df, th=tm_df, pv_cutoff=0.001)

# more restrictive peptide selection
robot_plot_All(thP = tmP_df, th=tm_df, pv_cutoff=0.001, CI_factor=3)
```

---

sd_timecourse	Returns standard deviation for uptake data for timecourses.
---------------	---

---

**Description**

Calculates standard deviation for timecourse data.

**Usage**

```
sd_timecourse(filepath)
```

**Arguments**

filepath          filepath to the All\_results input file.

**Value**

Data.frame with standard deviation.

**Examples**

```
file_nm<-system.file("extdata", "All_results_table.csv", package = "HDXBoxeR")
sd_timecourse(filepath=file_nm)
```

---

sd_timecourse_proc	Returns standard deviation for percent deuteration data for timecourses.
--------------------	--

---

**Description**

Calculates standard deviation for time course data.

**Usage**

```
sd_timecourse_proc(filepath)
```

**Arguments**

filepath          filepath to the All\_results input file.

**Value**

Data.frame with standard deviation.

**Examples**

```
file_nm<-system.file("extdata", "All_results_table.csv", package = "HDXBoxeR")
sd_timecourse(filepath=file_nm)
```

---

sd_timepoint	<i>Returns standard deviation for dataframe.</i>
--------------	--

---

**Description**

Calculates standard deviation for the number of replicates in the function.

**Usage**

```
sd_timepoint(df, replicates = 3)
```

**Arguments**

df	output from functions output_tp or output_tp_proc.
replicates	number of replicates used. Default is set to replicates=3

**Value**

Data.frame with standard deviation.

**Examples**

```
file_nm<-system.file("extdata", "All_results_table.csv", package = "HDXBoxeR")
a<- output_tp(file_nm)
sd<-sd_timepoint(df=a, replicates=3)
```

---

select_indices	<i>Allows for selecting some peptide from input data</i>
----------------	--

---

**Description**

Function allows for picking indices from the inputs based on: peptide start or end residue, length, state or timepoint. If parameters set to NA, condition is skipped.

**Usage**

```
select_indices(df, start = NA, end = NA, length = NA, times = NA, states = NA)
```

**Arguments**

df	input file (output of output_tc or output_tp)
start	provide number for the starting residue, default NA
end	provide number for the end residue, default NA
length	provide max length of the peptide
times	timepoints, only for the output_tp functions
states	states, only for the output_tc functions

**Value**

Row indices of the peptides that are fulfilling the conditions required.

**Examples**

```
file_nm<-system.file("extdata", "All_results_table.csv", package = "HDXBoxeR")
a<- output_tp(file_nm)
indb<-select_indices(a,length=12, start=100, end=200)
smaller_df<-a[indb,]
```

---

significant\_peptide\_uptake

*Function returns which peptides are significantly based of pv\_cutoff and Critical interval*

---

**Description**

Returns data frame with significant peptides.

**Usage**

```
significant_peptide_uptake(df_av, pv, sd, pv_cutoff = 0.01, replicates = 3)
```

**Arguments**

df_av	data.frame with averages created using ave_timepoint() function
pv	data.frame with pvalues created using pv_timepoint() function
sd	data.frame with standard deviations created using sd_timepoint() function
pv_cutoff	cutoff for Critical interval. Default=0.01
replicates	number of replicates as default set to 3.

**Value**

ranges per set

---

summary_sd_CI	<i>Provides summary table with Critical interval and standard deviation within the set.</i>
---------------	---

---

### Description

Returns summary data. Function returns: Protein states, timepoints, number of replicates, # peptides, % coverage, average peptide length and redundancy.

### Usage

```
summary_sd_CI(filepath, replicates = 3)
```

### Arguments

filepath	filepath to the input file. Input file is All_results table from HDX_Examiner, where all the fields are marked for export.
replicates	number of replicates. Default set to 3.

### Value

Returns summary table.

### Examples

```
file_nm<-system.file("extdata", "All_results_table.csv", package = "HDXBoxeR")  
a<- summary_sd_CI(file_nm, replicates=3)
```

---

uptake_plots	<i>Uptake plots</i>
--------------	---------------------

---

### Description

Uptake plots per peptide

### Usage

```
uptake_plots(  
  input_data,  
  timepoints,  
  replicates = 3,  
  cola = NA,  
  seq_match = TRUE  
)
```



**Arguments**

input_data	output from function output_tp(..., percent=T)
timepoints	the labeling times
replicates	replicates
cola	colors, default NA
seq_match	Flag TRUE or FALSE, default TRUE, match sequence of the protein states

**Value**

Uptake plots

**Examples**

```
file_nm<-system.file("extdata", "All_results_table.csv", package = "HDBoxerR")
a<- output_tc(file_nm, percent=TRUE)
x=c(3,60, 1800, 72000)
uptake_plots(a, x)
```

---

verbose\_timecourse\_output

*Returns csv with averages from analysis for procent deuteration file, standard deviation for time courses.*

---

**Description**

Returns information from analysis and save it as csv file. Sets are compared to the first state in the input file.

**Usage**

```
verbose_timecourse_output(filepath, output_name, replicates = 3, ...)
```

**Arguments**

filepath	path to All.Data.csv input from HDX-Examiner.
output_name	name of the output in csv format.
replicates	number of replicates used
...	other variables for output_tc

**Value**

csv with analysis for procent deuteration: standard deviation, for all protein states for time courses.

**Examples**

```
file_nm<-system.file("extdata", "All_results_table.csv", package = "HDXBoxeR")
verbose_timecourse_output(file_nm,tempfile(), replicates=3)
names_states<- nm_states(file_nm)
verbose_timecourse_output(file_nm, tempfile(), seq_match=TRUE, percent=TRUE,
states=names_states, replicates=3, times="3.00s")
```

---

```
verbose_timepoint_output
```

*Returns csv with averages from analysis for uptake file, standard deviation, p-values.*

---

**Description**

Returns information from analysis and save it as csv file. Sets are compared to the first state in the input file.

**Usage**

```
verbose_timepoint_output(filepath, output_name, replicates = 3, ...)
```

**Arguments**

filepath	path to All.Data.csv input from HDX-Examiner.
output_name	name of the output in csv format.
replicates	number of replicates used
...	other variables for output_tp

**Value**

csv with analysis for uptake file, standard deviation, p-values for all protein states.

**Examples**

```
file_nm<-system.file("extdata", "All_results_table.csv", package = "HDXBoxeR")
verbose_timepoint_output(file_nm, tempfile())
names_states<- nm_states(file_nm)
verbose_timepoint_output(file_nm, tempfile(), seq_match=TRUE, percent=TRUE,
states=names_states, replicates=3, times="3.00s")
```

---

vol_tp	<i>Preparatory function for volcano plot</i>
--------	--

---

**Description**

Returns volcano plots

**Usage**

```
vol_tp(df1, pv, CI, pv_cutoff = 0.01, cola)
```

**Arguments**

df1	differences in averages data.frame calculated using diff_ave function
pv	pvalues dataframes calculated using pv_timepoint function
CI	critical interval, here is multiple sets are using maximun CI is used.
pv_cutoff	p-value cutoff here set up to 0.01
cola	color palette for different Protein States. As default Paired palette from color.Brewer is used.

**Value**

volcano plots

---

woods_CI_plot	<i>Returns a woods plot for comparisons of the timepoints samples</i>
---------------	---

---

**Description**

Modification of butterfly plot. x axis residues. y axis % deuteration for Peptides are compared between the sets for the significance change between sets. If there is significant change between sets peptides are plotted for all timepoints. Significantly different timepoints for the peptides are colored. Peptides ranges are plotted as a line at corresponding % deuteration values.

**Usage**

```
woods_CI_plot(
  thP,
  th,
  replicates = 3,
  pv_cutoff = 0.01,
  states,
  CI_factor = 1,
  ylim = c(0, 120),
  ...
)
```

**Arguments**

thP	output of output_tcourse_proc() function. Raw data for percent deuteration for time courses
th	output of output_tcourse() function. Raw data for uptake deuteration for time courses
replicates	number of replicates in sample. Default set to 3.
pv_cutoff	p-value cutoff here set up to 0.01
states	Protein states from the set. As default all states are chosen.
CI_factor	Multiplication factor for Critical Interval. Allows for more restrictive selection of Critical interval.
ylim	y axis limit
...	other variables

**Value**

Woods plots with chosen statistically different peptides

**Examples**

```
file_nm<-system.file("extdata", "All_results_table.csv", package = "HDXBoxerR")
a<- output_tc(file_nm)
b<-output_tc(file_nm, percent=TRUE)
woods_CI_plot(thP=b, th=a, pv_cutoff = 0.001, CI_factor = 1, replicates=3)
```

# Index

all\_summary, 4  
arg\_df, 6  
arg\_UN\_FD, 7  
arguments\_call1, 5  
arguments\_call2, 5  
arguments\_call3, 6  
av\_tc, 8  
av\_tp, 9  
ave\_timepoint, 8  
average\_timecourse, 7  
  
backHX\_calculations, 9  
boxplot\_tp, 10  
  
CI\_2pts, 10  
CI\_single, 11  
CI\_tc, 12  
CI\_tp, 12  
color\_ranges\_Blue\_Red\_heat\_map, 13  
color\_ranges\_Spectral, 14  
coverage\_residue, 14  
  
deuteration\_woods\_timecourse, 15  
deuteration\_woods\_timepoints, 16  
dif\_ave, 17  
dif\_tp, 17  
dif\_tp\_proc, 18  
duplicate\_sets, 18  
  
extreme\_input\_gap, 19  
extreme\_input\_undeut, 19  
  
general\_info, 20  
getCoords1, 21  
  
heat\_map\_tc, 21  
heat\_map\_tp, 22  
heat\_map\_tp\_maxuptake, 22  
heat\_map\_tp\_maxuptake\_proc, 23  
heat\_map\_tp\_proc, 24  
  
is.nan.data.frame, 25  
  
lab\_dif, 25  
lab\_dif\_proc, 26  
lab\_vol, 26  
legend\_heat\_map, 27  
legend\_heat\_map\_tc, 27  
legend\_heat\_map\_timecourse, 28  
legend\_heat\_map\_tp, 28  
legend\_heat\_map\_tp\_proc, 29  
legend\_nm\_bottom, 29  
legend\_raw\_ave, 30  
legend\_raw\_ave\_proc, 30  
legend\_raw\_ave\_tc, 31  
legend\_sig\_peptides, 31  
legend\_states\_PerD\_bottom, 32  
legend\_tc\_bottom, 32  
  
nm\_states, 33  
  
output\_FD, 33  
output\_FD\_proc, 34  
output\_prep, 34  
output\_tc, 35  
output\_tp, 36  
output\_UD, 38  
output\_UD\_proc, 38  
  
palette\_legend, 39  
palette\_ll, 39  
peptide\_pv\_tp, 40  
peptide\_pv\_tp\_proc, 40  
pl\_gen\_ch2, 51  
pl\_gen\_uptake, 52  
plot\_heat\_map\_max\_uptake\_tp, 45  
plot\_heat\_map\_max\_uptake\_tp\_proc, 46  
plot\_heat\_map\_tc, 47  
plot\_heat\_map\_tp, 48  
plot\_heat\_map\_tp\_proc, 49  
plot\_peptide\_sig\_tp, 50

plot\_peptide\_sig\_tp\_proc, 50  
plots\_av\_tcourse, 41  
plots\_av\_tp, 42  
plots\_av\_tp\_proc, 43  
plots\_diff\_tp, 43  
plots\_diff\_tp\_proc, 44  
plots\_vol\_tp, 45  
ppar, 52  
ppar\_bottom\_legend, 53  
ppar\_wider, 54  
pparLM, 53  
prep\_timecourse\_plot\_ave, 54  
prep\_timecourse\_plot\_sd, 55  
pv\_timecourse, 55  
pv\_timepoint, 56  
pymol\_script\_average\_residue, 56  
pymol\_script\_significant\_peptide, 57  
pymol\_script\_significant\_peptide\_proc,  
58  
pymol\_script\_significant\_residue, 59  
pymol\_script\_significant\_residue\_proc,  
60  
pymol\_str, 61  
  
qpcr.cbind.na, 62  
  
ranges\_function, 63  
ranges\_function\_tc, 63  
rbind\_na, 64  
reset\_par, 64  
robot\_2states\_indexes, 65  
robot\_indexes, 66  
robot\_indexes\_df, 67  
robot\_plot\_All, 68  
  
sd\_timecourse, 69  
sd\_timecourse\_proc, 69  
sd\_timepoint, 70  
select\_indices, 70  
significant\_peptide\_uptake, 71  
summary\_sd\_CI, 72  
  
uptake\_plots, 72  
  
verbose\_timecourse\_output, 73  
verbose\_timepoint\_output, 74  
vol\_tp, 75  
  
woods\_CI\_plot, 75