Package 'MDplot'

July 4, 2017

Version 1.0.1

Date 2017-07-04

Title Visualising Molecular Dynamics Analyses

Depends R (>= 3.0.0), methods, MASS, RColorBrewer, gplots, gtools

Suggests R.rsp

VignetteBuilder R.rsp

Description

Provides automatization for plot generation succeeding common molecular dynamics analyses. This includes straightforward plots, such as RMSD (Root-Mean-Square-Deviation) and RMSF (Root-Mean-Square-Fluctuation) but also more sophisticated ones such as dihedral angle maps, hydrogen bonds, cluster bar plots and DSSP (Definition of Secondary Structure of Proteins) analysis. Currently able to load GROMOS, GROMACS and AMBER formats, respectively.

License GPL-3

URL https://github.com/MDplot/MDplot

LazyLoad yes

Author Christian Margreitter [aut, cre], Chris Oostenbrink [aut]

Maintainer Christian Margreitter <christian.margreitter@gmail.com>

NeedsCompilation no

Repository CRAN

Date/Publication 2017-07-04 20:26:53 UTC

R topics documented:

`clusters` *Cluster bar plot*

Description

This function plots clusters over a set of trajectories as joint, coloured bar plots. The clusters are sorted beginning with the most populated one in descending order.

Usage

```
clusters( clusters,
          clustersNumber = NA,
          legendTitle = "trajectories",
          barePlot = FALSE,
          ... )
```

Arguments

`clusters`	Matrix with clusters: trajectories are given in row-wise, clusters in column-wise fashion as provided by `load_clusters()`, the associated loading function.
`clustersNumber`	When specified, only these first clusters are shown.
`legendTitle`	The title of the legend.

barePlot	Boolean, indicating whether the plot is to be made without any additional information.
...	Additional arguments (ellipsis).

Value

Returns a nxm-matrix with n being the number of input trajectories and m the number of different clusters. Each element in the matrix holds the number of snapshots, the respective cluster occurred in the respective trajectory.

Author(s)

Christian Margreitter

Examples

```
# GROMOS (see load_clusters() for other input possibilities)
clusters( load_clusters( system.file( "extdata/clusters_example.txt.gz",
                                      package = "MDplot" ) ) )
```

---

clusters_ts *Cluster timeseries plot*

Description

This function plots distributions between clusters over time. In the top sub-plot, the overall distribution is given, while the timeseries is given at the bottom. The clusters are sorted beginning with the most populated one and then in descending order. Selections can be made and clusters that are not selected do also not appear in the timeseries plot (white areas).

Usage

```
clusters_ts( clustersDataTS,
             clustersNumber = NA,
             selectTraj = NA,
             selectTime = NA,
             timeUnit = NA,
             snapshotsPerTimeInt = 1000,
             ... )
```

Arguments

clustersDataTS	List of cluster information as provided by load_clusters_ts(), the associated loading function.
clustersNumber	Integer, specifying the number of clusters that is to be plotted.
selectTraj	Vector of indices of trajectories that are to be plotted (as given in the input file).
selectTime	Range of time in snapshots, which is to be plotted.

`timeUnit` Abbreviation of time unit.

`snapshotsPerTimeInt` Specifies, how many snapshots make up one time unit.

`...` Additional arguments (ellipsis).

Value

Returns a summary `(n+1)xm`-matrix with `n` being the number of input trajectories and `m` the number of different clusters (which have been plotted). Each element in the matrix holds the number of snapshots, the respective cluster occured in the respective trajectory. In addition, the first line is the overall summary counted over all trajectories.

Author(s)

Christian Margreitter

Examples

```
# GROMOS (see load_clusters_ts() for other input possibilities)
clusters_ts( load_clusters_ts( system.file( "extdata/clusters_ts_example.txt.gz",
                                package = "MDplot" ),
                                lengths = c( 4000, 4000, 4000, 4000, 4000, 4000 ) ),
             clustersNumber = 7 )
```

---

dssp *DSSP plot for secondary structure elements (proteins)*

Description

Plots summary plot for secondary structure motifs based on the output of the widely used classification program DSSP, which uses hydrogen bonds for classification. The default order is: "3-Helix", "4-Helix", "5-Helix", "Bend", "Beta-Bridge", "Beta-Strand", "Turn" (depending on the input, not all types might be included).

Usage

```
dssp( dsspData,
      printLegend = FALSE,
      useOwnLegend = FALSE,
      elementNames = NA,
      colours = NA,
      showValues = NA,
      showResidues = NA,
      plotType = "dots",
      selectedElements = NA,
      barePlot = FALSE,
      ... )
```

Arguments

dsspData	Table containing the information on the secondary structure motifs.
printLegend	If TRUE, a legend is printed on the right hand side of the plot.
useOwnLegend	If FALSE, the names of the secondary structure elements are considered to be in default order.
elementNames	Vector of names for the secondary structure elements.
colours	A vector of colours, that can be specified to replace the default ones.
showValues	A vector of boundaries for the values (two elements).
showResidues	A vector of boundaries for the residues (two elements).
plotType	Either "dots", "curves" or "bars".
selectedElements	A vector of names of the elements selected for plotting.
barePlot	Boolean, indicating whether the plot is to be made without any additional information.
...	Additional arguments (ellipsis).

Value

Returns matrix, where the first column is the residue-number and the remaining ones denote secondary structure classes. Residues are given row-wise and values range from 0 to 100 percent.

Author(s)

Christian Margreitter

Examples

```
# GROMOS (see load_dssp() for other input possibilities)
dssp( load_dssp( system.file( "extdata/dssp_example.txt.gz",
                 package = "MDplot" ) ) )
```

---

dssp_ts *DSSP timeseries plot for secondary structure elements (proteins)*

Description

Plots time-series for secondary structure motifs in the context of the widely used DSSP algorithm. The default order is: "3-Helix", "4-Helix", "5-Helix", "Bend", "Beta-Bridge", "Beta-Strand", "Turn".

Usage

```
dssp_ts( tsData,
         printLegend = TRUE,
         timeBoundaries = NA,
         residueBoundaries = NA,
         timeUnit = NA,
         snapshotsPerTimeInt = 1000,
         barScaleFactor = 0.25,
         barePlot = FALSE,
         ... )
```

Arguments

tsData	List consisting of lists, which are composed of a name (string) and a values table (x ... snapshots, y ... residues). Can be generated by load_dssp_ts().
printLegend	If TRUE, a legend is printed on the right hand side of the plot.
timeBoundaries	A vector of boundaries for the time in snapshots.
residueBoundaries	A vector of boundaries for the residues.
timeUnit	If set, the snapshots are transformed into the respective time (depending on parameter snapshotsPerTime).
snapshotsPerTimeInt	Number of snapshots per respective timeUnit (only used when timeUnit is set).
barScaleFactor	Allows to manually overwrite the height of the bars.
barePlot	Boolean, indicating whether the plot is to be made without any additional information.
...	Additional arguments (ellipsis).

Value

This function does not return data.

Author(s)

Christian Margreitter

Examples

```
# GROMOS (see load_dssp_ts() for other input possibilities)
dssp_ts( load_dssp_ts( system.file( "extdata/dssp_ts_example",
                                    package = "MDplot" ) ) )
```

hbond *Plot hydrogen bond summary*

Description

This function plots the summary output of hydrogen bond calculations and allows selection of donor and acceptor residues and atoms. Occurrence over the whole trajectory is indicated by a colour scale. A legend indicating the colour scale can be printed. Note, that in case multiple hydrogen bond interactions between two particular residues take place, the one with prevalence will be used for colour-coding (and by default, this interaction is marked with a black circle, see below).

Usage

```
hbond( hbonds,
       plotMethod = "residue-wise",
       acceptorRange = NA,
       donorRange = NA,
       printLegend = TRUE,
       showMultipleInteractions = TRUE,
       barePlot = FALSE,
       ... )
```

Arguments

hbonds	Table containing the hydrogen bond information in its columns `hbondID`, `resDonor`, `resDonorName`, `resAcceptor`, `resAcceptorName`, `atomDonor`, `atomDonorName`, `atomH`, `atomAcceptor`, `AtomAcceptorName`, `percentage`. This kind of table is automatically generated by function `load_hbond()`.
plotMethod	Allows to set the detail of hbond information displayed. Options are currently: • `residue-wise`
acceptorRange	Vector, specifying which range of acceptor residues is used.
donorRange	Vector, specifying which range of donor residues is used.
printLegend	Boolean, used to print or hide the legend for the occurences.
showMultipleInteractions	If `TRUE`, this option causes multiple interactions between the same residues as being represented by a black circle around the coloured dot.
barePlot	Boolean, indicating whether the plot is to be made without any additional information.
...	Additional arguments (ellipsis).

Value

Returns a table containing the information used for the plot:

- `resDonor` Residue number (donor).

- resAcceptor Residue number (acceptor).
- percentage Percentage, that has been used for colour-coding.
- numberInteractions Number of hydrogen bond interactions taking place between the specified donor and acceptor residues.

Author(s)

Christian Margreitter

Examples

```
# GROMOS (see load_hbond() for other input possibilities)
hbond( load_hbond( system.file( "extdata/hbond_example.txt.gz",
                                package = "MDplot" ) ) )
```

---

hbond_ts *Plot hydrogen bond timeseries*

Description

Timeseries plot of hydrogen bonds (various selections possible). In addition to the timeseries file, depending on the MD engine format used, an additional summary file might also be necessary (see below for examples).

Usage

```
hbond_ts( timeseries,
          summary,
          acceptorRange = NA,
          donorRange = NA,
          plotOccurences = FALSE,
          scalingFactorPlot = NA,
          printNames = FALSE,
          namesToSingle = FALSE,
          printAtoms = FALSE,
          timeUnit = NA,
          snapshotsPerTimeInt = 1000,
          timeRange = NA,
          hbondIndices = NA,
          barePlot = FALSE,
          ... )
```

Arguments

timeseries	Table, containing the timeseries information (e.g. produced by load_hbond_ts()).
summary	Table, containing the summary information (e.g. produced by load_hbond()).
acceptorRange	Vector of acceptor resdiues, which are to be plotted.
donorRange	Vector of donor residues, which are to be plotted.
plotOccurences	Specifies, whether the overall summary should also be plotted in a subplot on the right hand side.
scalingFactorPlot	To manually set the scaling factor (if necessary).
printNames	Enables human readable names rather than the hbond identifiers.
namesToSingle	If printNames is TRUE, this flags enables one letter codes.
printAtoms	If printNames is TRUE, this flag adds the atom names in the hydrogen bond identification on the y-axis.
timeUnit	Specifies the time unit on the x-axis.
snapshotsPerTimeInt	Specifies, how many snapshots make up one time unit (see above).
timeRange	Selects a certain time range specified by a vector.
hbondIndices	List, containing vectors to select hbonds by their identifiers.
barePlot	Boolean, indicating whether the plot is to be made without any additional information.
...	Additional arguments (ellipsis).

Value

Returns a nx2 matrix, with the first column being the list of hydrogen bond identifiers plotted and the second one the occurrence (in percent) over the selected time range.

Author(s)

Christian Margreitter

Examples

```
# the examples are valid and wrapped in the "dontrun{}" environment for efficiency purposes only
# GROMOS
## Not run:
hbond_ts( load_hbond_ts( system.file( "extdata/hbond_ts_example.txt.gz",
                                      package = "MDplot" ) ),
          load_hbond( system.file( "extdata/hbond_example.txt.gz",
                                   package = "MDplot" ) ) )
## End(Not run)

# GROMACS
## Not run:
hbond_ts( load_hbond_ts( system.file( "extdata/hbond_ts_example_GROMACS.xpm.gz",
                                      package = "MDplot" ),
```

```
                        mdEngine = "GROMACS" ),
          load_hbond( system.file( "extdata/hbond_ts_example_GROMACS.xpm.gz",
                                   package = "MDplot" ),
                      system.file( "extdata/hbond_example_GROMACS.txt.gz",
                                    package = "MDplot" ),
                      mdEngine = "GROMACS" ),
          plotOccurences = TRUE, namesToSingle = FALSE, printNames = TRUE,
          printAtoms = TRUE, hbondIndices = list( c( 1, 12 ) ),
          timeUnit = "ns", snapshotsPerTimeInt = 100 )
## End(Not run)

# AMBER
hbond_ts( load_hbond_ts( system.file( "extdata/hbond_ts_example_AMBER.txt.gz",
                                      package = "MDplot" ),
                         mdEngine = "AMBER" ),
          load_hbond( system.file( "extdata/hbond_ts_example_AMBER.txt.gz",
                                   package = "MDplot" ),
                      mdEngine = "AMBER" ),
          plotOccurences = TRUE, timeRange = c( 20, 60 ) )
```

`load_clusters` *Loading cluster information*

Description

This function loads clusters from a plain text file and stores them in a matrix. The trajectories can be named by the user. The output of this function can be used as input of function `clusters()`.

Usage

```
load_clusters( path,
               names = NA,
               lengths = NA,
               mdEngine = "GROMOS" )
```

Arguments

`path`	Specifies the path of the text input file.
`names`	Optional vector of trajectory names. If provided, needs to be of the same length as the number of clusters to be plotted.
`lengths`	When GROMACS input needs to be parsed, the lengths of the respective trajectories have to be given. This holds also in the case, that only one is used.
`mdEngine`	Argument distinguishes between input formats based on the molecular dynamics engine used. Currently available: `"GROMOS"`, `"GROMACS"` and `"AMBER"`. Note, that two different kinds of `AMBER` output may be loaded (see example input files).

Value

Returns a nxm-matrix with n being the number of input trajectories and m the number of different clusters. Each element in the matrix holds the number of snapshots, the respective cluster occured in the respective trajectory.

Author(s)

Christian Margreitter

Examples

```
# GROMOS
load_clusters( system.file( "extdata/clusters_example.txt.gz", package = "MDplot" ) )

# GROMACS
load_clusters( system.file( "extdata/clusters_example_GROMACS.txt.gz", package = "MDplot" ),
               mdEngine = "GROMACS", lengths = c( 1001, 1001 ) )

# AMBER (first)
load_clusters( system.file( "extdata/clusters_example_1_AMBER.txt.gz", package = "MDplot" ),
               mdEngine = "AMBER" )

# AMBER (second)
load_clusters( system.file( "extdata/clusters_example_2_AMBER.txt.gz", package = "MDplot" ),
               mdEngine = "AMBER" )
```

---

load_clusters_ts *Loading timeseries cluster information*

Description

This function loads the timeseries information for clusters from a plain text file and stores them in a list. The trajectories can be named by the user.

Usage

```
load_clusters_ts( path,
                  lengths,
                  names = NA,
                  mdEngine = "GROMOS" )
```

Arguments

path	Specifies the path of the text input file.
lengths	Mandatory vector holding the number of snapshots for the respective trajectories (e.g. when three trajectories of 3000 snapshots each have been analysed together: lengths = c(3000,3000,300)).

names	Optional vector of trajectory names. If provided, needs to be of the same length as the number of clusters to be plotted.
mdEngine	Argument distinguishes between input formats based on the molecular dynamics engine used. Currently available: "GROMOS", "GROMACS" and "AMBER".

Value

Returns a list of name-cluster lists, which consist of:

[1]	Trajectory name.
[2]	Vector of cluster numbers, where 0 indicates a structure not belonging to a cluster specified.

Author(s)

Christian Margreitter

Examples

```
# GROMOS
load_clusters_ts( system.file( "extdata/clusters_ts_example.txt.gz", package = "MDplot" ),
                  lengths = c( 4000, 4000, 4000, 4000, 4000, 4000 ) )

# GROMACS
load_clusters_ts( system.file( "extdata/clusters_example_GROMACS.txt.gz", package = "MDplot" ),
                  mdEngine = "GROMACS", lengths = c( 1001, 1001 ) )

# AMBER
load_clusters_ts( system.file( "extdata/clusters_ts_example_AMBER.txt.gz", package = "MDplot" ),
                  mdEngine = "AMBER", lengths = c( 50, 50 ) )
```

---

load_dssp *Load DSSP information*

Description

Loads DSSP summary output files from a specified file and combines it into a table. This table may be used as input for function dssp()

Usage

```
load_dssp( path,
           mdEngine = "GROMOS" )
```

Arguments

path	Path to the input file.
mdEngine	Argument distinguishes between input formats based on the molecular dynamics engine used. Currently available: "GROMOS", "GROMACS" and "AMBER".

Value

Returns matrix, where the first column is the residue-number and the remaining ones denote secondary structure classes. Residues are given row-wise and values range from `0` to `100` percent.

Author(s)

Christian Margreitter

Examples

```
# GROMOS
load_dssp( system.file( "extdata/dssp_example.txt.gz", package = "MDplot" ) )

# GROMACS
load_dssp( system.file( "extdata/dssp_example_GROMACS.txt.gz",
                        package = "MDplot" ), mdEngine = "GROMACS" )

# AMBER (same input as for load_dssp_ts())
load_dssp( system.file( "extdata/dssp_ts_example_AMBER.txt.gz",
                        package = "MDplot" ), mdEngine = "AMBER" )
```

---

load_dssp_ts *Load DSSP timeseries*

Description

Loads DSSP output files from a specified directory (GROMOS) or a specified file (GROMACS, AMBER) and combines it into a list suited as input for `dssp_ts()`.

Usage

```
load_dssp_ts( folder,
              filenames = NA,
              stride = 1,
              mdEngine = "GROMOS" )
```

Arguments

`folder`	Folder, in which the DSSP output files are located.
`filenames`	Vector with filenames. GROMOS: default file names are "3-Helix.out", "4-Helix.out", "5-Helix.out", "Bend.out", "Beta-Bridge.out", "Beta-Strand.out" and "Turn.out". Files not present are ignored. GROMACS and AMBER: just give the appropriate filename (see examples below).
`stride`	Allows to use only every xth snapshot.
`mdEngine`	Argument distinguishes between input formats based on the molecular dynamics engine used. Currently available: `"GROMOS"`, `"GROMACS"` and `"AMBER"`.

Value

Returns a list, where every element represents a secondary structure motif holding the following information:

`name`	The name of the secondary structure motif.
`values`	Holds one vector each for the `x` and `y` axis:

- `x` Snapshots, at which the respective motif has been identified.
- `y` Residues, for which the respective motif has been identified.

Author(s)

Christian Margreitter

Examples

```
# GROMOS
load_dssp_ts( system.file( "extdata/dssp_ts_example", package = "MDplot" ) )

# GROMACS
load_dssp_ts( folder = system.file( "extdata", package = "MDplot" ),
              filenames = "dssp_example_GROMACS.txt.gz",
              mdEngine = "GROMACS" )

# AMBER
load_dssp_ts( folder = system.file( "extdata", package = "MDplot" ),
              filenames = "dssp_ts_example_AMBER.txt.gz",
              mdEngine = "AMBER" )
```

---

load_hbond *Loading hydrogen bond data*

Description

This function loads hydrogen bond information from a text file and stores it in a table. See functions `hbond()` and `hbond_ts()` for usage of the return value.

Usage

```
load_hbond( path,
            GROMACShbondlogfile = NA,
            mdEngine = "GROMOS" )
```

Arguments

path	Specifies the path of the text input file.
GROMACShbondlogfile	Additional file path required in case GROMACS format is specified.
mdEngine	Argument distinguishes between input formats based on the molecular dynamics engine used. Currently available: "GROMOS", "GROMACS" and "AMBER". Note, that load_hbond() is able to load both summary and time-series data for the AMBER simulation package.

Value

Returns a table, where the hydrogen bonds are stored in a row-wise fashion and the columns hold the following information (note, that information not available e.g. because the MD analysis tool output does not provide it, is represented by NA values):

- hbondID The identifier number of the hydrogen bonds (if not given by the input, they are numbered ascendingly).
- resDonor Number of the donor residue.
- resDonorName Name of the donor residue.
- resAcceptor Number of the acceptor residue.
- resAcceptorName Name of the acceptor residue.
- atomDonor Number of atom, that is the hydrogen bond donor.
- atomDonorName Name of atom, that is the hydrogen bond donor.
- atomH Number of atom (proton) that is forming the hydrogen bond.
- atomAcceptor Number of atom, that is the hydrogen bond acceptor.
- atomAcceptorName Name of atom, that is the hydrogen bond acceptor.
- percentage Number between 0 and 100 in percent representing the occurence rate of a particular hydrogen bond over the trajectory.

Author(s)

Christian Margreitter

Examples

```
# GROMOS
load_hbond( system.file( "extdata/hbond_example.txt.gz", package = "MDplot" ) )

# GROMACS
load_hbond( system.file( "extdata/hbond_ts_example_GROMACS.xpm.gz",
                         package = "MDplot" ),
            system.file( "extdata/hbond_example_GROMACS.txt.gz",
                         package = "MDplot" ),
            mdEngine = "GROMACS" )

# AMBER (summary)
```

```
load_hbond( system.file( "extdata/hbond_example_AMBER.txt.gz",
            package = "MDplot" ), mdEngine = "AMBER" )

# AMBER (time-series)
load_hbond( system.file( "extdata/hbond_ts_example_AMBER.txt.gz",
            package = "MDplot" ), mdEngine = "AMBER" )
```

---

load_hbond_ts *Loading hydrogen bonds timeseries*

Description

This function loads hydrogen bond timeseries information from a text file and stores it in a table. See function hbond_ts() for usage of the return value. In case, AMBER format is used as input, this functions' return value might also be used for function hbond().

Usage

```
load_hbond_ts( path,
               mdEngine = "GROMOS" )
```

Arguments

path	Specifies the path of the input file.
mdEngine	Argument distinguishes between input formats based on the molecular dynamics engine used. Currently available: "GROMOS", "GROMACS" and "AMBER".

Value

Returns a nx2-matrix, where the first column holds the time in snapshots and the second one the respective hydrogen bond identifier. Note, that function hbond_ts() requires additional input provided by function load_hbond() and that hydrogen bond identifiers have to match.

Author(s)

Christian Margreitter

Examples

```
# the examples are valid and wrapped in the "dontrun{}" environment for efficiency purposes only
# GROMOS
## Not run:
load_hbond_ts( system.file( "extdata/hbond_ts_example.txt.gz", package = "MDplot" ) )
## End(Not run)

# GROMACS
## Not run:
load_hbond_ts( system.file( "extdata/hbond_ts_example_GROMACS.xpm.gz",
                            package = "MDplot" ),
```

```
               mdEngine = "GROMACS" )
## End(Not run)

# AMBER
## Not run:
load_hbond_ts( system.file( "extdata/hbond_ts_example_AMBER.txt.gz",
                            package = "MDplot" ),
               mdEngine = "AMBER" )
## End(Not run)
```

---

load_noe *Loading NOE violations*

Description

This function load one or more Nuclear-Overhauser-Effect (NOE) violation files. Its output can be feeded into function noe(). Note, that in case the number of used bins differ between files, the resulting matrix is automatically expanded to cover all bins.

Usage

```
load_noe( files,
          mdEngine = "GROMOS" )
```

Arguments

files	Vector of file paths to be loaded.
mdEngine	Argument distinguishes between input formats based on the molecular dynamics engine used. Currently available: "GROMOS".

Value

Returns a matrix, in which the first column holds the bin boundaries and the following columns represent the data from the respective input files, i.e. the number of violations in the respective bin.

Author(s)

Christian Margreitter

Examples

```
# GROMOS
load_noe( c( system.file( "extdata/noe_example_1.txt.gz",
                          package = "MDplot" ),
             system.file( "extdata/noe_example_2.txt.gz",
                          package = "MDplot" ) ) )
```

load_ramachandran *Load dihedral information (Ramachandran plot input)*

Description

Loads a textfile with dihedral angles, which are to be stored in a matrix. By default, the first column is phi and the second psi. Angles can be shifted by a constant value (in order to transform them from 0 to 360 to the usually used -180 to 180).

Usage

```
load_ramachandran( path,
                   angleColumns = c(1,2),
                   shiftAngles = NA,
                   mdEngine = "GROMOS" )
```

Arguments

path	Path to input file. At least two columns of the same length are expected.
angleColumns	If more columns are present, the angle columns can be chosen by this vector.
shiftAngles	In order to shift the values by a constant factor (e.g. -180).
mdEngine	Argument distinguishes between input formats based on the molecular dynamics engine used. Currently available: "GROMOS", "GROMACS" and "AMBER".

Value

A nx2-matrix with phi and psi angles in the respective columns.

Author(s)

Christian Margreitter

Examples

```
# GROMOS
load_ramachandran( system.file( "extdata/ramachandran_example.txt.gz", package = "MDplot" ) )

# GROMACS
load_ramachandran( system.file( "extdata/ramachandran_example_GROMACS.txt.gz",
                   package = "MDplot" ), mdEngine = "GROMACS" )

# AMBER
load_ramachandran( system.file( "extdata/ramachandran_example_AMBER.txt.gz",
                   package = "MDplot" ), mdEngine = "AMBER" )
```

load_rmsd *Loading function for* rmsd()

Description

Returns a list of vector pairs of datapoint indices and RMSD values.

Usage

```
load_rmsd( files,
           mdEngine = "GROMOS" )
```

Arguments

files	Vector of paths to input text files.
mdEngine	Argument distinguishes between input formats based on the molecular dynamics engine used. Currently available: "GROMOS", "GROMACS" and "AMBER".

Value

A list of vectors, alternately holding indices and their respective values.

Author(s)

Christian Margreitter

Examples

```
# GROMOS
load_rmsd( c( system.file( "extdata/rmsd_example_1.txt.gz", package = "MDplot" ),
              system.file( "extdata/rmsd_example_2.txt.gz", package = "MDplot" ) ) )

# GROMACS
load_rmsd( c( system.file( "extdata/rmsd_example_GROMACS.txt.gz",
                           package = "MDplot" ) ),
           mdEngine = "GROMACS" )

# AMBER
load_rmsd( c( system.file( "extdata/rmsd_example_AMBER.txt.gz",
                           package = "MDplot" ) ),
           mdEngine = "AMBER" )
```

load_rmsf *Loading function for* rmsf()

Description

Returns a list of vector pairs of datapoint indices and RMSF values.

Usage

```
load_rmsf( files,
           mdEngine = "GROMOS" )
```

Arguments

files	Vector of paths to input text files.
mdEngine	Argument distinguishes between input formats based on the molecular dynamics engine used. Currently available: "GROMOS", "GROMACS" and "AMBER".

Value

A list of vectors, alternately holding atom indices and their respective values.

Author(s)

Christian Margreitter

Examples

```
# GROMOS
load_rmsf( c( system.file( "extdata/rmsf_example_1.txt.gz", package = "MDplot" ),
              system.file( "extdata/rmsf_example_2.txt.gz", package = "MDplot" ) ) )

# GROMACS
load_rmsf( c( system.file( "extdata/rmsf_example_GROMACS.txt.gz",
                           package = "MDplot" ) ),
          mdEngine = "GROMACS" )

# AMBER
load_rmsf( c( system.file( "extdata/rmsf_example_AMBER.txt.gz",
                           package = "MDplot" ) ),
          mdEngine = "AMBER" )
```

load_TIcurve *Loading function for thermodynamic integration function* TIcurve()

Description

Returns a list of matrices holding three columns (lambda state point, value and error) for every file.

Usage

```
load_TIcurve( files,
              mdEngine = "GROMOS" )
```

Arguments

files	Vector of files (up to two) to be loaded.)
mdEngine	Argument distinguishes between input formats based on the molecular dynamics engine used. Currently available: "GROMOS".

Value

Returns a list of (at least) nx3-matrices, each holding three columns:

1. lambda-points
2. partial derivative of the Hamiltonian in respect to lambda at respective lambda
3. error associated with partial derivative at respective lambda

Author(s)

Christian Margreitter

Examples

```
# GROMOS
load_TIcurve( c( system.file( "extdata/TIcurve_example.txt.gz",
                              package = "MDplot" ) ) )
```

load_timeseries *Loading function for* timeseries()

Description

Returns a list of vector pairs of datapoint indices and values.

Usage

```
load_timeseries( files,
                 mdEngine = "GROMOS" )
```

Arguments

files	Vector of paths to input text files.
mdEngine	Argument distinguishes between input formats based on the molecular dynamics engine used. Currently available: "GROMOS", "GROMACS" and "AMBER". Note, that for GROMACS input, mulit-column data is supported.

Value

List of vectors, holding alternately indices and values.

Author(s)

Christian Margreitter

Examples

```
# GROMOS
load_timeseries( c( system.file( "extdata/timeseries_example_1.txt.gz",
                                 package = "MDplot" ),
                    system.file( "extdata/timeseries_example_2.txt.gz",
                                 package = "MDplot" ) ) )

# GROMACS
load_timeseries( c( system.file( "extdata/timeseries_example_GROMACS.txt.gz",
                                 package = "MDplot" ) ),
                 mdEngine = "GROMACS" )
```

load_xrmsd *Loading function for* xrmsd()

Description

Loads matrix information from the specified file.

Usage

```
load_xrmsd(  path,
             factor = 1,
             removeLowerHalf = TRUE,
             mdEngine = "GROMOS"  )
```

Arguments

path	Specifies the input file.
factor	In case the RMSD values are given in $$nm * factor$$, the factor can be specified. If the unit is already nanometers, 1 is the appropriate value.
removeLowerHalf	If TRUE, the lower half of the plot will be white.
mdEngine	Argument distinguishes between input formats based on the molecular dynamics engine used. Currently available: "GROMOS", "GROMACS" and "AMBER".

Value

Returns a nx3-matrix, with the first two columns holding the position (x- and y-axis) and the third the respective RMSD value.

Author(s)

Christian Margreitter

Examples

```
# GROMOS
load_xrmsd( system.file( "extdata/xrmsd_example.txt.gz",
                         package = "MDplot" ),
            factor = 10000 )

# GROMACS
load_xrmsd( system.file( "extdata/xrmsd_example_GROMACS.xpm.gz",
                         package = "MDplot" ),
            mdEngine = "GROMACS" )
```

```
# AMBER
load_xrmsd( system.file( "extdata/xrmsd_example_AMBER.txt.gz",
                         package = "MDplot" ),
            mdEngine = "AMBER" )
```

MDplot_argument-class *Arguments for bash script interface*

Description

Container for bash arguments with "key" <> "value" pairs.

Slots

key: Object of class "character".

value: Object of class "character".

Author(s)

Christian Margreitter

---

noe *Plot NOE violations*

Description

This function plots Nuclear-Overhauser-Effect (NOE) violations. Note, that negative violations are not considered, in case they are part of the input.

Usage

```
noe( noeData,
     printPercentages = TRUE,
     colours = NA,
     lineTypes = NA,
     names = NA,
     plotSumCurves = TRUE,
     maxYAxis = NA,
     printLegend = FALSE,
     ... )
```

Arguments

noeData	Input matrix. Can be generated by using function load_noe().
printPercentages	If TRUE, the violations will be reported in a relative manner (percent) instead absolute numbers.
colours	Vector of colours to be used for the bars.
lineTypes	If plotSumCurves is TRUE, this vector might be used to specify the types of curves plotted.
names	Vector to name the input columns (legend).
plotSumCurves	If TRUE, the violations are summed up from left to right to show the overall behaviour.
maxYAxis	Can be used to manually set the y-axis of the plot.
printLegend	Boolean, which triggers plotting of the legend.
...	Additional arguments (ellipsis).

Value

Returns a matrix, in which the first column holds the bin boundaries used and the following columns represent either the percentage or absolute numbers of the violations per bin, depending on the specification.

Author(s)

Christian Margreitter

Examples

```
# GROMOS
noe( load_noe( c( system.file( "extdata/noe_example_1.txt.gz",
                               package = "MDplot" ),
                  system.file( "extdata/noe_example_2.txt.gz",
                               package = "MDplot" ) ) ),
     names = c( "run1", "run2" ), printLegend = TRUE )
```

---

ramachandran *Ramachandran plot for two dihedral angles*

Description

This plotting function divides a full rotation (360 degrees) into x- and y- bins and colors them according to the number of angle pairs that are provided in the input, a so-called Ramachandran plot.

Usage

```
ramachandran( dihedrals,
              xBins = 150,
              yBins = 150,
              heatFun = "norm",
              structureAreas = c(),
              plotType = "sparse",
              printLegend = FALSE,
              plotContour = FALSE,
              barePlot = FALSE,
              ... )
```

Arguments

dihedrals	Matrix with angles (two columns).
xBins	Number of x-axis bins to be plotted.
yBins	Number of y-axis bins to be plotted.
heatFun	Function selector for calculation of the colour, possible are either: • "norm" (default) for linear calculation, or • "log" for logarithmic calculation.
structureAreas	List of areas, which are plotted as black lines.
plotType	Type of plot to be used, either "sparse" (default, using function hist2d()), "comic" (own binning, use for very few datapoints) or "fancy" (3D, using function persp()).
printLegend	Boolean, specifying whether a heat legend is to be plotted or not.
plotContour	Boolean, specifying whether a contour should be plotted or no.
barePlot	Boolean, indicating whether the plot is to be made without any additional information.
...	Additional arguments (ellipsis).

Value

Returns a list of binned dihedral angle occurrences.

Author(s)

Christian Margreitter

Examples

```
# GROMOS (see load_rmsd() for other input possibilities)
ramachandran( load_ramachandran( system.file( "extdata/ramachandran_example.txt.gz",
                                              package = "MDplot" ) ) )
```

rmsd *Root-mean-square-deviation (RMSD) plot*

Description

Plot (multiple) RMSD file(s) as produced by molecular dynamics packages.

Usage

```
rmsd( rmsdData,
      printLegend = TRUE,
      snapshotsPerTimeInt = 1000,
      timeUnit = "ns",
      rmsdUnit = "nm",
      colours = NA,
      names = NA,
      legendPosition = "bottomright",
      barePlot = FALSE,
      ... )
```

Arguments

rmsdData	List of (alternating) indices and RMSD values, as produced e.g. by load_rmsd().
printLegend	Boolean, which triggers plotting of the legend.
snapshotsPerTimeInt	Number, specifying how many snapshots are comprising one timeUnit.
timeUnit	Specifies, which unit the x-axis is given in.
rmsdUnit	Specifies, which unit the y-axis is given in.
colours	Vector of colours used for plotting.
names	Vector of the names of the trajectories.
legendPosition	Indicates position of legend: either "bottomright", "bottomleft", "topleft" or "topright".
barePlot	Boolean, indicating whether the plot is to be made without any additional information.
...	Additional arguments (ellipsis).

Value

Returns a list of lists, where each sub-list represents a RMSD curve and contains:

minValue	The minimum value over the whole time range.
maxValue	The maximum value over the whole time range.
meanValue	The mean value calculated over the whole time range.
sd	The standard deviation calculated over the whole time range.

Author(s)

Christian Margreitter

Examples

```
# GROMOS (see load_rmsd() for other input possibilities)
rmsd( load_rmsd( c( system.file( "extdata/rmsd_example_1.txt.gz", package = "MDplot" ),
                 system.file( "extdata/rmsd_example_2.txt.gz", package = "MDplot" ) ) ) )
```

---

rmsd_average *Root-mean-square-deviation (RMSD) average plot*

Description

Combines several RMSD index-value pairs and computes and plots the mean value and the spread (the respective minimum and maximum values) at every timepoint. This function is particularly useful, when multiple identical simulation runs (replicates) need to be analysed since it shows a 'corridor' which allows interpretation e.g. of the overall stability.

Usage

```
rmsd_average( rmsdInput,
              levelFactor = NA,
              snapshotsPerTimeInt = 1000,
              timeUnit = "ns",
              rmsdUnit = "nm",
              maxYAxis = NA,
              barePlot = FALSE,
              ... )
```

Arguments

`rmsdInput`	List of input tables (which are provided by function `load_rmsd()`).
`levelFactor`	If there are many datapoints, this parameter may be used to use only the `levelFactor`th datapoints to obtain a nicer graph.
`snapshotsPerTimeInt`	Number, specifying how many snapshots are comprising one `timeUnit`.
`timeUnit`	Specifies, which unit the x-axis is given in.
`rmsdUnit`	Specifies, which unit the y-axis is given in.
`maxYAxis`	Can be used to manually set the y-axis of the plot.
`barePlot`	Boolean, indicating whether the plot is to be made without any additional information.
`...`	Additional arguments (ellipsis).

Value

Returns a nx4-matrix, with the rows representing different snapshots and the columns the respective values as follows:

- snapshot Index of the snapshot.
- minimum The minimum RMSD value over all input sources at a given time.
- mean The mean RMSD value over all input sources at a given time.
- maximum The maximum RMSD value over all input sources at a given time.

Author(s)

Christian Margreitter

Examples

```
# GROMOS (see load_rmsd() for other input possibilities)
rmsd_average( list( load_rmsd( system.file( "extdata/rmsd_example_1.txt.gz",
                                            package = "MDplot" ) ),
                    load_rmsd( system.file( "extdata/rmsd_example_2.txt.gz",
                                            package = "MDplot" ) ) ),
              snapshotsPerTimeInt = 2000, maxYAxis = 0.445 )
```

---

rmsf *Root-mean-square-fluctuation (RMSF) plot*

Description

Plot (multiple) RMSF file(s) as produced by molecular dynamics packages.

Usage

```
rmsf( rmsfData,
      printLegend = TRUE,
      rmsfUnit = "nm",
      colours = NA,
      residuewise = FALSE,
      atomsPerResidue = NA,
      names = NA,
      rangeAtoms = NA,
      legendPosition = "topright",
      barePlot = FALSE,
      ... )
```

Arguments

rmsfData	List of (alternating) indices and RMSF values, as produced e.g. by load_rmsf().
printLegend	Boolean, which triggers plotting of the legend.
rmsfUnit	Specifies, which unit the y-axis is given in.
colours	Vector of colours used for plotting.
residuewise	Boolean, specifying whether atoms or residues are plotted on the x-axis.
atomsPerResidue	If residuewise is TRUE, this parameter can be used to specify the number of atoms per residue for plotting.
names	Vector of the names of the trajectories.
rangeAtoms	Range of atoms to be plotted.
legendPosition	Indicate position of legend: either "bottomright", "bottomleft", "topleft" or "topright".
barePlot	Boolean, indicating whether the plot is to be made without any additional information.
...	Additional arguments (ellipsis).

Value

A list of vectors, alternately holding atom indices and their respective values.

Author(s)

Christian Margreitter

Examples

```
# GROMOS (see load_rmsf() for other input possibilities)
rmsf( load_rmsf( c( system.file( "extdata/rmsf_example_1.txt.gz", package = "MDplot" ),
                 system.file( "extdata/rmsf_example_2.txt.gz", package = "MDplot" ) ) ) )
```

TIcurve *Thermodynamic integration plot*

Description

Plot the thermodynamic integration(s) (TIs) specified in the input files. Files have to have at least three columns (lambda point, free energy and error) in order to be valid. In addition, the delta free energiy (to a precision dependent on the error) are calculated. In case, two data input series are provided, the hysteresis is calculated.

Usage

```
TIcurve( lambdas,
         invertedBackwards = FALSE,
         energyUnit = "kJ/mol",
         printValues = TRUE,
         printErrors = TRUE,
         errorBarThreshold = 0,
         barePlot = FALSE,
         ... )
```

Arguments

lambdas	List of matrices (automatically generated by load_TIcurve()) holding the thermodynamic integration information.
invertedBackwards	In case a forward and backward TI have been performed and the lambda points are enumerated reversely (i.e. 0.3 of one TI is equivalent to 0.7 of the other), this flag can be set to be TRUE in order to automatically mirror the values appropriately.
energyUnit	Defines the energy unit used for the plot.
printValues	Boolean, indicating whether the computed integration and error values are to be plotted.
printErrors	Boolean, indicating whether error bars are to be plotted.
errorBarThreshold	If the error at a given lambda point is below this theshold, it is not plotted.
barePlot	Boolean, indicating whether the plot is to be made without any additional information.
...	Additional arguments (ellipsis).

Value

Returns a list with the the following information:

- lambdapoints A list containing a (at least) nx3-matrix for every data input series.
- integrationresults A matrix containing one row of "deltaG" and "error" columns from the integration for every data input series.
- hysteresis If two (i.e. forward and backward) data input series are provided, the resulting hysteresis is reported (and set to be NA otherwise).

Author(s)

Christian Margreitter

Examples

```
# GROMOS (forward integration)
TIcurve( load_TIcurve( system.file( "extdata/TIcurve_example.txt.gz",
                                    package = "MDplot" ) ) )

# GROMOS (forward and backward integration)
TIcurve( load_TIcurve( c( system.file( "extdata/TIcurve_fb_forward_example.txt.gz",
                                       package = "MDplot" ),
                          system.file( "extdata/TIcurve_fb_backward_example.txt.gz",
                          package = "MDplot" ) ) ),
         invertedBackwards = TRUE )
```

---

timeseries *General timeseries plotting function*

Description

Plot one (or more) timeseries plots.

Usage

```
timeseries( tsData,
            printLegend = TRUE,
            snapshotsPerTimeInt = 1000,
            timeUnit = "ns",
            valueName = NA,
            valueUnit = NA,
            colours = NA,
            names = NA,
            legendPosition = "bottomright",
            barePlot = FALSE,
            ... )
```

Arguments

tsData	List of (alternating) indices and response values, as also produced by load_timeseries() for example.
printLegend	Boolean, which triggers plotting of the legend.
snapshotsPerTimeInt	Number, specifying how many snapshots are within one timeUnit.
timeUnit	Specifies, which unit the x-axis is given in.
valueName	Name of response variable.
valueUnit	Specifies, which unit the y-axis is given in.
colours	Vector of colours used for plotting.
names	Vector of the names of the trajectories.

legendPosition	Indicate position of legend: either "bottomright", "bottomleft", "topleft" or "topright".
barePlot	Boolean, indicating whether the plot is to be made without any additional information.
...	Additional arguments (ellipsis).

Value

Returns a list of list, the latter each holding for every data input series:

- minValue The minimum value over the whole set.
- maxValue The maximum value over the whole set.
- meanValue The mean value over the whole set.
- sd The standard deviation over the whole set.

Author(s)

Christian Margreitter

Examples

```
# GROMOS
timeseries( load_timeseries( c( system.file( "extdata/timeseries_example_1.txt.gz",
                                             package = "MDplot" ),
                                system.file( "extdata/timeseries_example_2.txt.gz",
                                             package = "MDplot" ) ) ),
            snapshotsPerTimeInt = 100 )

# GROMACS
timeseries( load_timeseries( c( system.file( "extdata/timeseries_example_GROMACS.txt.gz",
                                             package = "MDplot" ) ),
                             mdEngine = "GROMACS" ),
            ylim = c( 0.545, 0.7 ), valueName = "Area per lipid", valueUnit = "nm^2" )
```

---

translate_aminoacids *Function to translate between canonical and GROMOS amino acid abbreviations*

Description

Converts an aminoacid naming scheme in the context of GROMOS (e.g. containing HISH) into canonical three- or one-letter codes.

Usage

```
translate_aminoacids( input,
                      switchMode )
```

Arguments

input	Vector of GROMOS abbreviations.
switchMode	Type "1" results in single-letter, type "2" in three-letter code.

Author(s)

Christian Margreitter

Examples

```
translate_aminoacids( c( "ALA", "HISA", "SER" ),
                      switchMode = 2 )
```

---

xrmsd *XRMSD plot in heatmap style*

Description

Plots an heatmap according to the RMSD values for a 2D snapshot matrix, based on molecular dynamics trajectories.

Usage

```
xrmsd( xrmsdValues,
       printLegend = TRUE,
       xaxisRange = NA,
       yaxisRange = NA,
       colours = NA,
       rmsdUnit = "nm",
       barePlot = FALSE,
       ... )
```

Arguments

xrmsdValues	Input matrix (three rows: x-values, y-values, RMSD-value). Can be generated by function load_xrmsd().
printLegend	If TRUE, a legend is printed on the right hand side of the plot.
xaxisRange	A vector of boundaries for the x-snapshots.
yaxisRange	A vector of boundaries for the y-snapshots.
colours	A vector with colour names for the span palette.
rmsdUnit	Specifies, in which unit the RMSD values are given.
barePlot	Boolean, indicating whether the plot is to be made without any additional information.
...	Additional arguments (ellipsis).

Value

This function does not return data.

Author(s)

Christian Margreitter

Examples

```
# GROMOS (see load_xrmsd() for other input possibilities)
xrmsd( load_xrmsd( system.file( "extdata/xrmsd_example.txt.gz",
                                package = "MDplot" ),
                   factor = 10000 ) )
```

Index