Package 'metadynminer'

October 13, 2022

Type Package

Title Tools to Read, Analyze and Visualize Metadynamics HILLS Files from 'Plumed'

Version 0.1.7

Date 2022-04-12

Depends R (>= 3.3.0)

LinkingTo Rcpp

Description Metadynamics is a state of the art biomolecular simulation technique. 'Plumed' Tribello, G.A. et al. (2014) <doi:10.1016/j.cpc.2013.09.018> program makes it possible to perform metadynamics using various simulation codes. The results of metadynamics done in 'Plumed' can be analyzed by 'metadynminer'. The package 'metadynminer' reads 1D and 2D metadynamics hills files from 'Plumed' package. It uses a fast algorithm by Hosek, P. and Spiwok, V. (2016) <doi:10.1016/j.cpc.2015.08.037> to calculate a free energy surface from hills. Minima can be located and plotted on the free energy surface. Transition states can be analyzed by Nudged Elastic Band method by Henkelman, G. and Jonsson, H. (2000) <doi:10.1063/1.1323224>. Free energy surfaces, minima and transition paths can be plotted to produce publication quality images.

LazyData true

License GPL-3

RoxygenNote 6.1.0

Imports Rcpp

Suggests testthat

URL https://metadynamics.cz/metadynminer/

NeedsCompilation yes

Author Vojtech Spiwok [aut, cre] (<https://orcid.org/0000-0001-8108-2033>)

Maintainer Vojtech Spiwok <spiwokv@vscht.cz>

Repository CRAN

Date/Publication 2022-04-14 11:02:29 UTC

R topics documented:

`acealanme` *Hills from 30 ns metadynamics of AceAlaNme in water with two collective variables*

Description

Hills from 30 ns metadynamics of AceAlaNme (Amber99SB-ILDN) in water (TIP3P) with Ramachandran dihedrals phi and psi as collective variables.

Usage

```
acealanme
```

Format

hillsfile object

Source

http://www.metadynamics.cz/metadynminer/data/HILLS2d

`acealanme1d` *Hills from 30 ns metadynamics of AceAlaNme in water with one collective variable*

Description

Hills from 30 ns metadynamics of AceAlaNme (Amber99SB-ILDN) in water (TIP3P) with a Ramachandran dihedral phi as the collective variable.

Usage

```
acealanme1d
```

Format

hillsfile object

Source

http://www.metadynamics.cz/metadynminer/data/HILLS1d

`emptyhills` *Generate empty HILLS from Plumed*

Description

'emptyhills' returns a hillsfile object with no hills. User can specify whether some collective variables are periodic.

Usage

```
emptyhills(dim=2, per=c(FALSE, FALSE), pcv1=c(-pi,pi), pcv2=c(-pi,pi))
```

Arguments

`dim`	dimensionality of the output.
`per`	logical vector specifying periodicity of collective variables.
`pcv1`	periodicity of CV1.
`pcv2`	periodicity of CV2.

Value

hillsfile object.

Examples

```
emptyhills(dim=2)
```

`feprof` *Calculate free energy profile for minima object (generic function for 'metadynminer' and 'metadynminer3d')*

Description

'feprof' calculates free energy profiles for free energy minima. It finds the global minimum at the 'imax' and calculates the evolution of free energies of a local vs. the global free energy minimum. The free energy of the global minimum is constant (zero).

Usage

```
feprof(minims, imax)
```

Arguments

`minims`	minima object.
`imax`	index of a hill from which summation stops (default the rest of hills).

feprof.minima *Calculate free energy profile for minima object*

Description

'feprof.minima' calculates free energy profiles for free energy minima. It finds the global minimum at the 'imax' and calculates the evolution of free energies of a local vs. the global free energy minimum. The free energy of the global minimum is constant (zero).

Usage

```
## S3 method for class 'minima'
feprof(minims, imax = NULL)
```

Arguments

minims	minima object.
imax	index of a hill from which summation stops (default the rest of hills).

Examples

```
tfes<-fes(acealanme, imax=5000)
minima<-fesminima(tfes)
prof<-feprof(minima)
prof
```

fes *Calculate free energy surface by Bias Sum algorithm (generic function for 'metadynminer' and 'metadynminer3d')*

Description

'fes' sums up hills using fast Bias Sum algorithm.

Usage

```
fes(hills, imin, imax, xlim, ylim, zlim, npoints)
```

Arguments

hills	hillsfile object.
imin	index of a hill from which summation starts (default 1).
imax	index of a hill from which summation stops (default the rest of hills).
xlim	numeric vector of length 2, giving the CV1 coordinates range.
ylim	numeric vector of length 2, giving the CV2 coordinates range.
zlim	numeric vector of length 2, giving the CV3 coordinates range.
npoints	resolution of the free energy surface in number of points.

Value

fes object.

`fes.hillsfile` *Calculate free energy surface by Bias Sum algorithm*

Description

'fes.hillsfile' sums up hills using fast Bias Sum algorithm.

Usage

```
## S3 method for class 'hillsfile'
fes(hills, imin = 1, imax = NULL, xlim = NULL,
  ylim = NULL, zlim = NULL, npoints = 256)
```

Arguments

`hills`	hillsfile object.
`imin`	index of a hill from which summation starts (default 1).
`imax`	index of a hill from which summation stops (default the rest of hills).
`xlim`	numeric vector of length 2, giving the CV1 coordinates range.
`ylim`	numeric vector of length 2, giving the CV2 coordinates range.
`zlim`	numeric vector of length 2, giving the CV3 coordinates range.
`npoints`	resolution of the free energy surface in number of points.

Value

fes object.

Examples

```
tfes<-fes(acealanme, imax=5000)
```

fes2 *Calculate free energy surface by conventional algorithm (generic function for 'metadynminer' and 'metadynminer3d')*

Description

'fes2' sums up hills using slow conventional algorithm. It can be used as a reference or when hill widths are variable.

Usage

```
fes2(hills, imin, imax, xlim, ylim, zlim, npoints)
```

Arguments

hills	hillsfile object.
imin	index of a hill from which summation starts (default 1).
imax	index of a hill from which summation stops (default the rest of hills).
xlim	numeric vector of length 2, giving the CV1 coordinates range.
ylim	numeric vector of length 2, giving the CV2 coordinates range.
zlim	numeric vector of length 2, giving the CV3 coordinates range.
npoints	resolution of the free energy surface in number of points.

Value

fes object.

fes2.hillsfile *Calculate free energy surface by conventional algorithm*

Description

'fes2.hillsfile' sums up hills using slow conventional algorithm. It can be used as a reference or when hill widths are variable.

Usage

```
## S3 method for class 'hillsfile'
fes2(hills, imin = 1, imax = NULL, xlim = NULL,
  ylim = NULL, zlim = NULL, npoints = 256)
```

Arguments

hills	hillsfile object.
imin	index of a hill from which summation starts (default 1).
imax	index of a hill from which summation stops (default the rest of hills).
xlim	numeric vector of length 2, giving the CV1 coordinates range.
ylim	numeric vector of length 2, giving the CV2 coordinates range.
zlim	numeric vector of length 2, giving the CV3 coordinates range.
npoints	resolution of the free energy surface in number of points.

Value

fes object.

Examples

```
tfes<-fes2(acealanme, imax=1000)
```

---

fes2d21d *Calculate 1D free energy surface from hillsfile object*

Description

'fes2d21d' calculates 2D free energy surface, converts free energies to probabilities (exp(-F/kT)), sums them up along one collective variable and converts back to free energy (-kT log(P)).

Usage

```
fes2d21d(hills, remdim = 2, temp = 300, eunit = "kJ/mol", imin = 1,
  imax = NULL, xlim = NULL, ylim = NULL, npoints = 256)
```

Arguments

hills	hillsfile object.
remdim	dimension to be removed (1 for CV1, 2 for CV2, default 2).
temp	temperature in Kelvins (default 300).
eunit	energy units (kJ/mol or kcal/mol, kJ/mol is default).
imin	index of a hill from which summation starts (default 1).
imax	index of a hill from which summation stops (default the rest of hills).
xlim	numeric vector of length 2, giving the CV1 coordinates range.
ylim	numeric vector of length 2, giving the CV2 coordinates range.
npoints	resolution of the free energy surface in number of points.

Value

fes object.

Examples

```
tfes<-fes2d21d(acealanme, remdim=2, imax=5000)
```

`fesminima` *Find free energy minima in the fes object (generic function for 'metadynminer' and 'metadynminer3d')*

Description

'fesminima' finds free energy minima on 1D or 2D free energy surface. The surface is divided by a 1D or 2D grid and minima are found for each bin. Next the program determines whether the minimum of a bin is a local minimum of the whole free energy surface. Free energy minima are labeled constitutively by capital letters.

Usage

```
fesminima(inputfes, nbins)
```

Arguments

`inputfes`	fes object.
`nbins`	number of bins for each CV (default 8).

Value

minima object.

`fesminima.fes` *Find free energy minima in the fes object*

Description

'fesminima.fes' finds free energy minima on 1D or 2D free energy surface. The surface is divided by a 1D or 2D grid and minima are found for each bin. Next the program determines whether the minimum of a bin is a local minimum of the whole free energy surface. Free energy minima are labeled constitutively by capital letters.

Usage

```
## S3 method for class 'fes'
fesminima(inputfes, nbins = 8)
```

Arguments

inputfes	fes object.
nbins	number of bins for each CV (default 8).

Value

minima object.

Examples

```
tfes<-fes(acealanme, imax=5000)
minima<-fesminima(tfes)
minima
```

---

fespoint — *Calculate free energy at given point in the CV space (generic function for 'metadynminer' and 'metadynminer3d')*

Description

'fespoint' calculates free energy at given point in the CV space 'coord'. Hills are summed from 'imin' to 'imax'. Printed output can be suppressed by setting 'verb' to TRUE.

Usage

```
fespoint(hills, coord, imin, imax, verb)
```

Arguments

hills	hillsfile object.
coord	coordinates of the point in the CV space.
imin	index of a hill from which calculation of difference starts (default 1).
imax	index of a hill from which summation stops (default the rest of hills).
verb	if TRUE, the output is verbose (default TRUE).

fespoint.hillsfile *Calculate free energy at given point in the CV space*

Description

'fespoint.hillsfile' calculates free energy at given point in the CV space 'coord'. Hills are summed from 'imin' to 'imax'. Printed output can be suppressed by setting 'verb' to TRUE.

Usage

```
## S3 method for class 'hillsfile'
fespoint(hills, coord = NULL, imin = 1,
  imax = NULL, verb = T)
```

Arguments

hills	hillsfile object.
coord	coordinates of the point in the CV space.
imin	index of a hill from which calculation of difference starts (default 1).
imax	index of a hill from which summation stops (default the rest of hills).
verb	if TRUE, the output is verbose (default TRUE).

Examples

```
fespoint(acealanme, c(0,0), imax=5000)
```

head.hillsfile *Print first n lines of hillsfile*

Description

'head.hillsfile' prints first n lines of a hillsfile object.

Usage

```
## S3 method for class 'hillsfile'
head(x, n = 10, ...)
```

Arguments

x	hillsfile object.
n	number of lines (default 10).
...	further arguments passed to or from other methods.

Examples

```
head(acealanme)
```

lines.fes *Plots 1D free energy surface object as lines*

Description

'lines.fes' plots 1D free energy surface as lines.

Usage

```
## S3 method for class 'fes'
lines(x, lwd = 1, col = "black", ...)
```

Arguments

x	fes object.
lwd	line width for drawing symbols see 'par'.
col	color code or name, see 'par'.
...	further arguments passed to or from other methods.

Examples

```
tfes<-fes(acealanme1d, imax=5000)
plot(tfes)
lines(tfes, lwd=4)
```

lines.hillsfile *Plot lines for hillsfile object*

Description

'lines.hillsfile' plots lines for hillsfile object. For a hillsfile with one collective variable it plots its evolution. For a hillsfile with two collective variables it plots CV1 vs. CV2.

Usage

```
## S3 method for class 'hillsfile'
lines(x, ignoretime = FALSE, lwd = 1,
  col = "black", ...)
```

Arguments

x	hillsfile object.
ignoretime	time in the first column of the HILLS file will be ignored.
lwd	line width for drawing symbols see 'par'.
col	color code or name, see 'par'.
...	further arguments passed to or from other methods.

Examples

```
plot(acealanme)
lines(acealanme, col="red")
```

`lines.nebpath` *Plot lines for Nudged Elastic Band*

Description

'lines.nebpath' plots lines for free energy profile calculated by Nudged Elastic Band.

Usage

```
## S3 method for class 'nebpath'
lines(x, col = "red", lwd = 1, ...)
```

Arguments

`x`	nebpath object.
`col`	color code or name, see 'par'.
`lwd`	line width for drawing symbols see 'par'.
`...`	further arguments passed to or from other methods.

Examples

```
tfes<-fes(acealanme, imax=5000)
minima<-fesminima(tfes)
nebAD<-neb(minima, min1="A", min2="D", nsteps=20)
plot(nebAD)
lines(nebAD, lwd=4)
```

`linesonfes` *Plot lines for Nudged Elastic Band projected onto free energy surface*

Description

'linesonfes' plots lines for free energy profile calculated by Nudged Elastic Band projected onto free energy surface.

Usage

```
linesonfes(x, col = "red", lwd = 1)
```

Arguments

x	nebpath object.
col	color code or name, see 'par'.
lwd	line width for drawing symbols see 'par'.

Examples

```
tfes<-fes(acealanme, imax=5000)
minima<-fesminima(tfes)
nebAD<-neb(minima, min1="A", min2="D", nsteps=20)
plot(minima)
linesonfes(nebAD)
```

---

max.fes *Calculate maximum of free energy surface*

Description

'max.fes' calculates maximum of free energy in a fes object.

Usage

```
## S3 method for class 'fes'
max(inputfes, na.rm = NULL, ...)
```

Arguments

inputfes	fes object.
na.rm	a logical indicating whether missing values should be removed.
...	further arguments passed to or from other methods.

Examples

```
tfes<-fes(acealanme, imax=5000)
max(tfes)
```

min.fes *Calculate minimum of free energy surface*

Description

'min.fes' calculates minimum of free energy in a fes object.

Usage

```
## S3 method for class 'fes'
min(inputfes, na.rm = NULL, ...)
```

Arguments

inputfes	fes object.
na.rm	a logical indicating whether missing values should be removed.
...	further arguments passed to or from other methods.

Examples

```
tfes<-fes(acealanme, imax=5000)
min(tfes)
```

neb *Find transition path on free energy surface by Nudged Elastic Band method*

Description

'neb' finds a transition path on free energy surface for a given pair of minima. For a 1D surface it simply takes the free energy profile between the two minima. For 2D surface it calculates the transition path by Nudged Elastic

Usage

```
neb(minims = minims, min1 = "A", min2 = "B", nbins = 20,
  nsteps = 100, step = 1, k = 0.2)
```

Arguments

minims	minima object.
min1	starting minimum identifier (can be letter or index, default "A").
min2	final minimum identifier (can be letter or index, default "B").
nbins	number of bins along Nudged Elastic Band (default 20).
nsteps	number of Nudged Elastic Band iterations (default 100).
step	Nudged Elastic Band iteration step (default 1).
k	Nudged Elastic Band toughness (default 0.2).

Value

NEB path

Examples

```
tfes<-fes(acealanme, imax=5000)
minima<-fesminima(tfes)
nebAD<-neb(minima, min1="A", min2="D", nsteps=20)
nebAD
```

`oneminimum` *Creates one ad hoc free energy minimum for a fes object (generic function for 'metadynminer' and 'metadynminer3d')*

Description

'oneminimum' creates an ad hoc free energy minimum on free energy surface. This can be used to calculate free energy surface evolution at arbitrary point of free energy surface.

Usage

```
oneminimum(inputfes, cv1, cv2, cv3)
```

Arguments

`inputfes`	fes object.
`cv1`	the value of collective variable 1.
`cv2`	the value of collective variable 2.
`cv3`	the value of collective variable 3.

Value

minima object.

oneminimum.fes *Creates one ad hoc free energy minimum for a fes object*

Description

'oneminimum.fes' creates an ad hoc free energy minimum on free energy surface. This can be used to calculate free energy surface evolution at arbitrary point of free energy surface.

Usage

```
## S3 method for class 'fes'
oneminimum(inputfes, cv1, cv2, cv3)
```

Arguments

inputfes	fes object.
cv1	the value of collective variable 1.
cv2	the value of collective variable 2.
cv3	the value of collective variable 3.

Value

minima object.

Examples

```
tfes<-fes(acealanme1d)
minima<-fesminima(tfes)
minima<-minima+oneminimum(tfes, cv1=0, cv2=0)
minima
```

plot.fes *Plot free energy surface object*

Description

'plot.fes' plots free energy surface. For a fes with one collective variable it plots a 1D profile. For a fes with two collective variables it plots 2D free energy surface using image, contours or combination of both (default).

Usage

```
## S3 method for class 'fes'
plot(x, plottype = "both", colscale = F, xlim = NULL,
  ylim = NULL, zlim = NULL, main = NULL, sub = NULL, xlab = NULL,
  ylab = NULL, nlevels = 10, levels = NULL,
  col = rainbow(135)[100:1], labels = NULL, labcex = 0.6,
  drawlabels = TRUE, colscalelab = "free energy",
  method = "flattest", contcol = par("fg"), lty = par("lty"),
  lwd = 1, asp = NULL, axes = T, ...)
```

Arguments

x	fes object.
plottype	specifies whether 2D free energy surface will be plotted as image, contours or both (default "both").
colscale	specifies whether color scale will be plotted (default False).
xlim	numeric vector of length 2, giving the x coordinates range.
ylim	numeric vector of length 2, giving the y coordinates range.
zlim	numeric vector of length 2, giving the z coordinates range.
main	an overall title for the plot: see 'title'.
sub	a sub title for the plot: see 'title'.
xlab	a title for the x axis: see 'title'.
ylab	a title for the y axis: see 'title'.
nlevels	number of contour levels desired if 'levels' is not supplied.
levels	numeric vector of levels at which to draw contour lines.
col	color of the free energy surface. For 1D surface it is the color of the line. For 2D it is a list of colors such as that generated by 'rainbow', 'heat.colors', 'topo.colors', 'terrain.colors' or similar functions (default=rainbow(135)[100:1]).
labels	a vector giving the labels for the contour lines. If 'NULL' then the levels are used as labels, otherwise this is coerced by 'as.character'.
labcex	'cex' for contour labeling. This is an absolute size, not a multiple of 'par("cex")'.
drawlabels	logical. Contours are labeled if 'TRUE'.
colscalelab	color scale label (default "free energy").
method	character string specifying where the labels will be located. Possible values are '"simple"', '"edge"' and '"flattest"' (the default). See the 'Details' section.
contcol	contour color.
lty	line type for the lines drawn.
lwd	contour line width.
asp	the y/x aspect ratio, see 'plot.window'.
axes	a logical value indicating whether both axes should be drawn on the plot.
...	further arguments passed to or from other methods.

Examples

```
tfes2d<-fes(acealanme, imax=5000)
plot(tfes2d)
tfes1d<-fes(acealanme1d)
plot(tfes1d)
```

`plot.hillsfile` *Plot hillsfile object*

Description

'plot.hillsfile' plots hillsfile object. For a hillsfile with one collective variable it plots its evolution. For a hillsfile with two collective variables it plots CV1 vs. CV2.

Usage

```
## S3 method for class 'hillsfile'
plot(x, ignoretime = FALSE, xlab = NULL,
  ylab = NULL, xlim = NULL, ylim = NULL, main = NULL, sub = NULL,
  pch = 1, col = "black", bg = "red", cex = 1, asp = NULL,
  lwd = 1, axes = TRUE, ...)
```

Arguments

x	hillsfile object.
ignoretime	time in the first column of the HILLS file will be ignored.
xlab	a title for the x axis: see 'title'.
ylab	a title for the y axis: see 'title'.
xlim	numeric vector of length 2, giving the x coordinates range.
ylim	numeric vector of length 2, giving the y coordinates range.
main	an overall title for the plot: see 'title'.
sub	a sub title for the plot: see 'title'.
pch	plotting 'character', i.e., symbol to use. See 'points'.
col	color code or name, see 'par'.
bg	background (fill) color for the open plot symbols given by 'pch = 21:25'.
cex	character (or symbol) expansion: a numerical vector. This works as a multiple of 'par("cex")'.
asp	the y/x aspect ratio, see 'plot.window'.
lwd	line width for drawing symbols see 'par'.
axes	a logical value indicating whether both axes should be drawn on the plot.
...	further arguments passed to or from other methods.

Examples

```
plot(acealanme)
```

plot.minima *Plot minima object*

Description

'plot.minima' plots free energy surface with minima. The free energy surface is plotted the same way as by plot.fes with additional minima labels.

Usage

```
## S3 method for class 'minima'
plot(x, plottype = "both", xlim = NULL, ylim = NULL,
  zlim = NULL, colscale = F, colscalelab = "free energy",
  main = NULL, sub = NULL, xlab = NULL, ylab = NULL,
  nlevels = 10, levels = NULL, col = rainbow(135)[100:1],
  labels = NULL, labcex = 0.6, drawlabels = TRUE,
  method = "flattest", textcol = "black", pch = 1, bg = "red",
  cex = 1, contcol = par("fg"), lty = par("lty"), lwd = par("lwd"),
  asp = NULL, axes = TRUE, ...)
```

Arguments

x	minima object.
plottype	specifies whether 2D free energy surface will be plotted as image, contours or both (default "both").
xlim	numeric vector of length 2, giving the x coordinates range.
ylim	numeric vector of length 2, giving the y coordinates range.
zlim	numeric vector of length 2, giving the z coordinates range.
colscale	specifies whether color scale will be plotted (default False).
colscalelab	color scale label (default "free energy").
main	an overall title for the plot: see 'title'.
sub	a sub title for the plot: see 'title'.
xlab	a title for the x axis: see 'title'.
ylab	a title for the y axis: see 'title'.
nlevels	number of contour levels desired if 'levels' is not supplied.
levels	numeric vector of levels at which to draw contour lines.
col	color of the free energy surface. For 1D surface it is the color of the line. For 2D it is a list of colors such as that generated by 'rainbow', 'heat.colors', 'topo.colors', 'terrain.colors' or similar functions (default=rainbow(135)[100:1]).
labels	a vector giving the labels for the contour lines. If 'NULL' then the levels are used as labels, otherwise this is coerced by 'as.character'.
labcex	'cex' for contour labeling. This is an absolute size, not a multiple of 'par("cex")'.

drawlabels	logical. Contours are labeled if 'TRUE'.
method	character string specifying where the labels will be located. Possible values are '"simple"', '"edge"' and '"flattest"' (the default). See the 'Details' section.
textcol	color of minima labels.
pch	plotting 'character', i.e., symbol to use. See 'points'
bg	background (fill) color for the open plot symbols given by 'pch = 21:25'.
cex	character (or symbol) expansion: a numerical vector. This works as a multiple of 'par("cex")'.
contcol	contour color.
lty	line type for the lines drawn.
lwd	contour line width.
asp	the y/x aspect ratio, see 'plot.window'.
axes	a logical value indicating whether both axes should be drawn on the plot.
...	further arguments passed to or from other methods.

Examples

```
tfes<-fes(acealanme, imax=5000)
minima<-fesminima(tfes)
plot(minima)
```

`plot.nebpath` *Plot Nudged Elastic Band*

Description

'plot.nebpath' plots free energy profile calculated by Nudged Elastic Band.

Usage

```
## S3 method for class 'nebpath'
plot(x, xlim = NULL, ylim = NULL, main = NULL,
  sub = NULL, xlab = "bin", ylab = "free energy", col = "red",
  lwd = 1, asp = NULL, cex = 1, axes = T, ...)
```

Arguments

x	nebpath object.
xlim	numeric vector of length 2, giving the x coordinates range.
ylim	numeric vector of length 2, giving the y coordinates range.
main	an overall title for the plot: see 'title'.
sub	a sub title for the plot: see 'title'.
xlab	a title for the x axis: see 'title'.

ylab	a title for the y axis: see 'title'.
col	color code or name, see 'par'.
lwd	line width for drawing symbols see 'par'.
asp	the y/x aspect ratio, see 'plot.window'.
cex	text expansion.
axes	a logical value indicating whether both axes should be drawn on the plot.
...	further arguments passed to or from other methods.

Examples

```
tfes<-fes(acealanme, imax=5000)
minima<-fesminima(tfes)
nebAD<-neb(minima, min1="A", min2="D", nsteps=20)
plot(nebAD)
```

---

plot.profiles *Plot free energy profile*

Description

'plot.profiles' plots evolution of free energy differences between minima. They are colored by rainbow colors from the global one (blue) to the highest (red).

Usage

```
## S3 method for class 'profiles'
plot(x, which = NULL, ignoretime = FALSE,
  xlim = NULL, ylim = NULL, main = NULL, sub = NULL, xlab = NULL,
  ylab = NULL, col = NULL, asp = NULL, lwd = 1, axes = T, ...)
```

Arguments

x	profiles object.
which	vector of indexes of profiles to be plotted (default all).
ignoretime	time in the first column of the HILLS file will be ignored.
xlim	numeric vector of length 2, giving the x coordinates range.
ylim	numeric vector of length 2, giving the y coordinates range.
main	an overall title for the plot: see 'title'.
sub	a sub title for the plot: see 'title'.
xlab	a title for the x axis: see 'title'.
ylab	a title for the y axis: see 'title'.
col	color code or name, see 'par'.
asp	the y/x aspect ratio, see 'plot.window'.
lwd	line width.
axes	a logical value indicating whether both axes should be drawn on the plot.
...	further arguments passed to or from other methods.

Examples

```
tfes<-fes(acealanme, imax=5000)
minima<-fesminima(tfes)
prof<-feprof(minima)
plot(prof)
```

`plotheights` *Plot evolution of heights of hills (generic function for 'metadynminer' and 'metadynminer3d')*

Description

'plotheights' plots evolution of heights of hills. In well tempered metadynamics hill heights decrees with flooding of the free energy surface. Evolution of heights may be useful to evaluate convergence of the simulation.

Usage

```
plotheights(hills, ignoretime, xlab, ylab, xlim, ylim, main, sub, col, asp,
  lwd, axes)
```

Arguments

`hills`	hillsfile object.
`ignoretime`	time in the first column of the HILLS file will be ignored.
`xlab`	a title for the x axis: see 'title'.
`ylab`	a title for the y axis: see 'title'.
`xlim`	numeric vector of length 2, giving the x coordinates range.
`ylim`	numeric vector of length 2, giving the y coordinates range.
`main`	an overall title for the plot: see 'title'.
`sub`	a sub title for the plot: see 'title'.
`col`	color code or name, see 'par'.
`asp`	the y/x aspect ratio, see 'plot.window'.
`lwd`	line width for drawing symbols see 'par'.
`axes`	a logical value indicating whether both axes should be drawn on the plot.

`plotheights.hillsfile` *Plot evolution of heights of hills in hillsfile object*

Description

'plotheights.hillsfile' plots evolution of heights of hills. In well tempered metadynamics hill heights decrees with flooding of the free energy surface. Evolution of heights may be useful to evaluate convergence of the simulation.

Usage

```
## S3 method for class 'hillsfile'
plotheights(hills, ignoretime = FALSE, xlab = NULL,
  ylab = NULL, xlim = NULL, ylim = NULL, main = NULL, sub = NULL,
  col = "black", asp = NULL, lwd = 1, axes = TRUE)
```

Arguments

`hills`	hillsfile object.
`ignoretime`	time in the first column of the HILLS file will be ignored.
`xlab`	a title for the x axis: see 'title'.
`ylab`	a title for the y axis: see 'title'.
`xlim`	numeric vector of length 2, giving the x coordinates range.
`ylim`	numeric vector of length 2, giving the y coordinates range.
`main`	an overall title for the plot: see 'title'.
`sub`	a sub title for the plot: see 'title'.
`col`	color code or name, see 'par'.
`asp`	the y/x aspect ratio, see 'plot.window'.
`lwd`	line width for drawing symbols see 'par'.
`axes`	a logical value indicating whether both axes should be drawn on the plot.

Examples

```
plotheights(acealanme)
```

points.fes *Plots 1D free energy surface object as points*

Description

'points.fes' plots 1D free energy surface as points.

Usage

```
## S3 method for class 'fes'
points(x, pch = 1, col = "black", bg = "red",
  cex = 1, lwd = 1, ...)
```

Arguments

x	fes object.
pch	plotting 'character', i.e., symbol to use. See 'points'
col	color code or name, see 'par'.
bg	background (fill) color for the open plot symbols given by 'pch = 21:25'.
cex	character (or symbol) expansion: a numerical vector. This works as a multiple of 'par("cex")'.
lwd	line width for drawing symbols see 'par'.
...	further arguments passed to or from other methods.

Examples

```
tfes<-fes(acealanme1d, imax=5000)
plot(tfes)
points(tfes)
```

points.hillsfile *Plot points for hillsfile object*

Description

'points.hillsfile' plots points for hillsfile object. For a hillsfile with one collective variable it plots its evolution. For a hillsfile with two collective variables it plots CV1 vs. CV2.

Usage

```
## S3 method for class 'hillsfile'
points(x, ignoretime = FALSE, pch = 1,
  col = "black", bg = "red", cex = 1, lwd = 1, ...)
```

Arguments

x	hillsfile object.
ignoretime	time in the first column of the HILLS file will be ignored.
pch	plotting 'character', i.e., symbol to use. See 'points'.
col	color code or name, see 'par'.
bg	background (fill) color for the open plot symbols given by 'pch = 21:25'.
cex	character (or symbol) expansion: a numerical vector. This works as a multiple of 'par("cex")'.
lwd	line width for drawing symbols see 'par'.
...	further arguments passed to or from other methods.

Examples

```
plot(acealanme)
points(acealanme, col="red")
```

---

points.nebpath *Plot points for Nudged Elastic Band*

Description

'points.nebpath' plots points for free energy profile calculated by Nudged Elastic Band.

Usage

```
## S3 method for class 'nebpath'
points(x, pch = NULL, cex = 1, bg = NULL,
  col = "red", lwd = 1, ...)
```

Arguments

x	nebpath object.
pch	plotting 'character', i.e., symbol to use. See 'points'.
cex	character (or symbol) expansion: a numerical vector. This works as a multiple of 'par("cex")'.
bg	background (fill) color for the open plot symbols given by 'pch = 21:25'.
col	color code or name, see 'par'.
lwd	line width for drawing symbols see 'par'.
...	further arguments passed to or from other methods.

Examples

```
tfes<-fes(acealanme, imax=5000)
minima<-fesminima(tfes)
nebAD<-neb(minima, min1="A", min2="D", nsteps=20)
plot(nebAD)
points(nebAD)
```

`pointsonfes` *Plot points for Nudged Elastic Band projected onto free energy surface*

Description

'pointsonfes' plots points for free energy profile calculated by Nudged Elastic Band projected onto free energy surface.

Usage

```
pointsonfes(x, pch = NULL, cex = 1, bg = NULL, col = "red",
  lwd = 1)
```

Arguments

`x`	nebpath object.
`pch`	plotting 'character', i.e., symbol to use. See 'points'.
`cex`	character (or symbol) expansion: a numerical vector. This works as a multiple of 'par("cex")'.
`bg`	background (fill) color for the open plot symbols given by 'pch = 21:25'.
`col`	color code or name, see 'par'.
`lwd`	line width for drawing symbols see 'par'.

Examples

```
tfes<-fes(acealanme, imax=5000)
minima<-fesminima(tfes)
nebAD<-neb(minima, min1="A", min2="D", nsteps=20)
plot(minima)
pointsonfes(nebAD)
```

print.fes *Print dimensionality, minimum and maximum of free energy surface*

Description

'print.fes' prints dimensionality, minimum and maximum of free energy in a fes object

Usage

```
## S3 method for class 'fes'
print(x, ...)
```

Arguments

x	fes object
...	further arguments passed to or from other methods.

Examples

```
tfes<-fes(acealanme, imax=5000)
tfes
```

print.hillsfile *Print hillsfile*

Description

'print.hillsfile' prints dimensionality and size of a hillsfile object.

Usage

```
## S3 method for class 'hillsfile'
print(x, ...)
```

Arguments

x	hillsfile object.
...	further arguments passed to or from other methods.

Examples

```
acealanme
```

print.minima *Print minima object*

Description

'print.minima' prints free energy minima (identifier, values of bins and collective variables and free energy).

Usage

```
## S3 method for class 'minima'
print(x, ...)
```

Arguments

x	minima object.
...	further arguments passed to or from other methods.

Examples

```
tfes<-fes(acealanme, imax=5000)
minima<-fesminima(tfes)
minima
```

print.nebpath *Print Nudged Elastic Band minima*

Description

'print.nebpath' prints the list minima for Nudged Elastic Band

Usage

```
## S3 method for class 'nebpath'
print(x, ...)
```

Arguments

x	nebpath object
...	further arguments passed to or from other methods.

Examples

```
tfes<-fes(acealanme, imax=5000)
minima<-fesminima(tfes)
nebAD<-neb(minima, min1="A", min2="D", nsteps=20)
nebAD
```

print.profiles *Print profiles object*

Description

'print.profiles' prints free energy profile.

Usage

```
## S3 method for class 'profiles'
print(x, ...)
```

Arguments

x	minima object.
...	further arguments passed to or from other methods.

Examples

```
tfes<-fes(acealanme, imax=5000)
minima<-fesminima(tfes)
prof<-feprof(minima)
prof
```

prob *Calculate probability of free energy surface*

Description

'prob' calculates probability from free energy in a fes object.

Usage

```
prob(inputfes, temp = 300, eunit = "kJ/mol")
```

Arguments

inputfes	fes object.
temp	temperature in Kelvins.
eunit	energy units (kJ/mol or kcal/mol, kJ/mol is default).

Examples

```
tfes<-fes(acealanme, imax=5000)
print(prob(tfes))
```

read.hills *Read HILLS from Plumed*

Description

'read.hills' reads a HILLS file generated by Plumed and returns a hillsfile object. User can specify whether some collective variables are periodic.

Usage

```
read.hills(file = "HILLS", per = c(FALSE, FALSE), pcv1 = c(-pi, pi),
  pcv2 = c(-pi, pi), ignoretime = FALSE)
```

Arguments

file	HILLS file from Plumed.
per	logical vector specifying periodicity of collective variables.
pcv1	periodicity of CV1.
pcv2	periodicity of CV2.
ignoretime	time in the first column of the HILLS file will be ignored.

Value

hillsfile object.

Examples

```
l1<-"1 -1.409  2.808 0.3 0.3 1.111 10"
l2<-"2 -2.505  2.791 0.3 0.3 1.111 10"
l3<-"3 -2.346  2.754 0.3 0.3 1.069 10"
l4<-"4 -1.198  2.872 0.3 0.3 1.074 10"
fourhills<-c(l1,l2,l3,l4)
tf <- tempfile()
writeLines(fourhills, tf)
read.hills(tf, per=c(TRUE,TRUE))
```

read.plumed *Read 1D or 2D free energy surface from PLUMED sum_hills*

Description

'read.plumed' reads 1D or 2D free energy surface from PLUMED sum_hills. The grid in the (2D) inputfile must contain the same number of points for CV1 and CV2. It does not use the header of the file. Instead, user must specify the dimensionality (1 or 2). Periodicity must be specified as well.

Usage

```
read.plumed(file = "fes.dat", dim = 2, per = c(F, F, F))
```

Arguments

file	input file from PLUMED sum_hills.
per	logical vector specifying periodicity of collective variables.
dim	dimension (1 or 2, default 2).

Value

fes object.

Examples

```
l1<-"-3.142 -124.8 -44.76"
l2<-"-3.117 -125.9 -43.05"
l3<-"-3.092 -126.9 -41.22"
l4<-"-3.068 -127.9 -39.36"
l5<-"-3.043 -128.8 -37.45"
fourpoints<-c(l1,l2,l3,l4)
tf <- tempfile()
writeLines(fourpoints, tf)
read.plumed(tf, dim=1, per=c(TRUE,TRUE))
```

---

summary.fes *Print summary of free energy surface*

Description

'summary.fes' prints dimensionality, minimum and maximum of free energy in a fes object.

Usage

```
## S3 method for class 'fes'
summary(object, ...)
```

Arguments

object	fes object.
...	further arguments passed to or from other methods.

Examples

```
tfes<-fes(acealanme, imax=5000)
summary(tfes)
```

summary.hillsfile *Print summary for hillsfile*

Description

'summary.hillsfile' prints dimensionality, size and collective variable ranges of a hillsfile object.

Usage

```
## S3 method for class 'hillsfile'
summary(object, ...)
```

Arguments

object	hillsfile object.
...	further arguments passed to or from other methods.

Examples

```
summary(acealanme)
```

summary.minima *Print minima object summary*

Description

'summary.minima' prints summary for free energy minima (identifier, values of bins and collective variables, free energy and equilibrium populations).

Usage

```
## S3 method for class 'minima'
summary(object, temp = 300, eunit = "kJ/mol", ...)
```

Arguments

object	minima object
temp	temperature in Kelvins
eunit	energy units (kJ/mol or kcal/mol, kJ/mol is default)
...	further arguments passed to or from other methods.

Examples

```
tfes<-fes(acealanme, imax=5000)
minima<-fesminima(tfes)
summary(minima)
```

summary.nebpath *Print summary for Nudged Elastic Band*

Description

'print.nebpath' prints the list minima for Nudged Elastic Band, activation energies and half lives calculated by Eyring equation (https://doi.org/10.1063/1.1749604).

Usage

```
## S3 method for class 'nebpath'
summary(object, temp = 300, eunit = "kJ/mol", ...)
```

Arguments

object	nebpath object.
temp	temperature in Kelvins.
eunit	energy units (kJ/mol or kcal/mol, kJ/mol is default).
...	further arguments passed to or from other methods.

Examples

```
tfes<-fes(acealanme, imax=5000)
minima<-fesminima(tfes)
nebAD<-neb(minima, min1="A", min2="D", nsteps=20)
summary(nebAD)
```

summary.profiles *Print summary for free energy profile*

Description

'summary.profiles' prints the list of free energy minima with maximal and minimal free energy differences.

Usage

```
## S3 method for class 'profiles'
summary(object, imind = 1, imaxd = NULL, ...)
```

Arguments

object	profiles object.
imind	index of a hill from which calculation of difference starts (default 1).
imaxd	index of a hill from which calculation of difference stops (default the rest of hills).
...	further arguments passed to or from other methods.

Examples

```
tfes<-fes(acealanme, imax=5000)
minima<-fesminima(tfes)
prof<-feprof(minima)
summary(prof)
```

`tail.hillsfile` *Print last n lines of hillsfile*

Description

'tail.hillsfile' prints last n lines of a hillsfile object.

Usage

```
## S3 method for class 'hillsfile'
tail(x, n = 10, ...)
```

Arguments

`x`	hillsfile object.
`n`	number of lines (default 10).
`...`	further arguments passed to or from other methods.

Examples

```
tail(acealanme)
```

Index