Package ‘admtools’

March 1, 2024

Title Estimate and Manipulate Age-Depth Models

Version 0.2.0

Description Estimate age-depth models from stratigraphic and sedimentological data,
and transform data
between the time and stratigraphic domain.

URL https://github.com/MindTheGap-ERC/admtools,
https://mindthegap-erc.github.io/admtools/

BugReports https://github.com/MindTheGap-ERC/admtools/issues

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.2.3

Depends R (>= 2.10), ape

LazyData true

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

NeedsCompilation no

Author Niklas Hohmann [aut, cre] (<https://orcid.org/0000-0003-1559-1838>)

Maintainer Niklas Hohmann <N.H.Hohmann@uu.nl>

Repository CRAN

Date/Publication 2024-03-01 14:22:43 UTC

R topics documented:

`add_adm_to_multiadm` *add adm object ot multiadm object*

Description

add adm object ot multiadm object

Usage

```
add_adm_to_multiadm(x, ...)
```

Arguments

`x`	multiadm object
`...`	adm objects to be added to x

Value

a multiadm object

`CarboCATLite_data` *Example data from CarboCATLite*

Description

Data exported from CarboCATLite model run

Usage

```
CarboCATLite_data
```

Format

A list with the following fields:

- *time_myr* : time points in Myr from the model run
- *height_2_km_offshore_m* : sediment thickness accumulated 2 km from shore
- *height_4_km_offshore_m* : sediment thickness accumulated 4 km from shore
- *height_6_km_offshore_m* : sediment thickness accumulated 6 km from shore
- *height_8_km_offshore_m* : sediment thickness accumulated 8 km from shore
- *height_10_km_offshore_m* : sediment thickness accumulated 10 km from shore
- *height_12_km_offshore_m* : sediment thickness accumulated 12 km from shore
- *eustatic_SL_m* : eustatic sea level used for the model run.

Source

Elapsed model time, sea level, and accumulated sediment thickness taken from the scenario A model run here

References

- Burgess, Peter. "CarboCAT: A cellular automata model of heterogeneous carbonate strata." Computers & geosciences 53 (2013): 129-140. doi:10.1016/j.cageo.2011.08.026
- Burgess, Peter. (2023). CarboCATLite (v1.0.1). Zenodo. doi:10.5281/zenodo.8402578

`condensation` *condensation with height*

Description

returns (instantaneous) condensation (time preserved per length increment) for a section

Usage

```
condensation(x, h, mode = "rcll", ...)
```

Arguments

`x`	adm or multiadm object
`h`	numeric vector, positions where condensation is determined
`mode`	string, handed over to *sed_rate_t*, see ?sed_rate_t for details
`...`	parameters passed to *get_time*, see ?get_time for details

Value

if x is an adm object, a numeric vector of condensations. if x is a multiadm object, a list of condensations

condensation_fun *condensation function*

Description

returns a function that determines instantaneous condensation (time preserved per strat. increment)

Usage

```
condensation_fun(x, mode = "rcll", ...)
```

Arguments

x	adm object
mode	string, handed over to *sed_rate_t*, see ?sed_rate_t for details
...	parameters passed to *get_time*, see ?get_time for details

Value

a function

flux_const *constant deterministic tracer flux*

Description

For usage with *strat_cont_to_multiadm*; defines constant tracer flux in the time domain

Usage

```
flux_const()
```

Value

a function factory that takes no arguments

See Also

flux_linear(), flux_quad(), strat_cont_gen_from_tracer()

Examples

```
## Not run:
# see this vignette for an example
vignette("adm_from_trace_cont")

## End(Not run)
```

`flux_linear` *linear deterministic tracer flux*

Description

For usage with *strat_cont_to_multiadm* : defines linear tracer flux in the time domain Tracer flux is the linear function passing through the points (x0, y0) and (x1, y1)

Usage

```
flux_linear(x0 = 0, y0 = 1, x1 = 1, y1 = 2)
```

Arguments

`x0`	numeric, abscissa
`y0`	numeric, ordinate
`x1`	numeric, abscissa
`y1`	numeric, ordinate

Value

a function factory that takes no arguments. Upon each evaluation, it returns a linear function passing through the points (x0, y0) and (x1, y1)

See Also

`flux_const()`, `flux_quad()`, `strat_cont_gen_from_tracer()`

`flux_quad` *quadratic deterministic tracer flux*

Description

For usage with *strat_cont_to_multiadm* : defines quadratic tracer flux in the time domain defined by the function f(x) = ax^2 + bx + c

Usage

```
flux_quad(a = 1, b = 1, c = 1)
```

Arguments

`a`	numeric
`b`	numeric
`c`	numeric

Value

a function factory that takes no arguments. Upon each evaluation, it returns the quadratic function f(x) = ax^2 + bx + c

See Also

`flux_linear()`, `flux_const()`, `strat_cont_gen_from_tracer()`

get_completeness *Determine stratigraphic (in)completeness*

Description

Determine stratigraphic (in)completeness

Usage

```
get_completeness(x)
get_incompleteness(x)
```

Arguments

x	an adm object

Details

Stratigraphic (in)completeness is expressed as a proportion, i.e. a number between 0 and 1

Value

Number between 0 and 1, the stratigraphic (in)completeness

Examples

```
my_adm = tp_to_adm(t = 1:4, h = c(1,2,2,4))
get_completeness(my_adm)
get_incompleteness(my_adm)
```

get_data_from_eTimeOpt *extract data from eTimeOpt results*

Description

Extracts data from eTimeOpt. The type of data extractecd dependes on the `output` setting used for eTimeOpt. If you want ot extract specific data, adjust the output parameter in eTimeOpt to return the correct data (e.g. 2 for r^2 envelope). See eTimeOpt documentation for details on this. Then call this function on the return variable.

Usage

```
get_data_from_eTimeOpt(res, index = 1)
```

Arguments

`res`	results generated by eTimeOpt
`index`	which output should be extracted? See description for details

Value

a list with three entries "sed_rate" : numeric vector, sedimentation rates "height" : numeric vector, heights "results" : matrix with length(height) rows and length(sed_rate) columns. results of eTimeOpt

See Also

`sed_rate_from_matrix()` to use define sedimentation rates based on this functions outputs, `sedrate_to_multiadm()` to estimate age-depth models from the outputs.

get_height *determine stratigraphic height deposited at specific time*

Description

Takes an adm object and a vector of times, and returns the stratigraphic heights deposited at said times

Usage

```
get_height(x, t, destructive = TRUE, out_dom_val_h = "default", ...)
```

Arguments

`x` an *adm* or *multiadm* object

`t` vector of times

`destructive` logical - should destructive intervals be considered? See Details

`out_dom_val_h` "strat_limits", "default", or a vector with one or two entries. What value is assigned to times that are not covered by the age-depth model?

`...` parameters handed over to is_destructive

Details

if destructive is true, NA is returned for times that coincide with destructive intervals. This is achieved by calling *is_destructive* with arguments passed by ....

out_dom_val specified the return value for times that are not covered by the age-depth model. For "default", NA is returned. For "strat_limits", the lowest resp. highest stratigraphic position is returned. For a vector of length one, this value is assigned to both sides. For a vector or length 2 or more, the first and second entries are assigned on the left (resp. right) side

Value

a vector with same length as t, containing the strat heights deposited

---

`get_hiat_duration` *extract hiatus duration*

Description

returns a vector of hiatus durations

Usage

```
get_hiat_duration(x)
```

Arguments

`x` an adm object

Value

a vector with one element per hiatus: the duration of the hiatus

See Also

- `get_hiat_pos()` to determine only stratigraphic position of hiatuses
- `get_hiat_no()` to determine number of hiatuses in an adm
- `get_hiat_list()` to get hiatus position, stat & end time

`get_hiat_list` *extract hiatus info*

Description

returns a list with hiatus position and timing (start & end)

Usage

```
get_hiat_list(x)
```

Arguments

`x`	an adm object

Value

a list with one element per hiatus. each element is a named vector with the following entries:

- "height" : stratigraphic position of hiatus
- "start" : time when hiatus begins
- "end" : time when hiatus ends

See Also

- `get_hiat_pos()` to determine only stratigraphic position of hiatuses
- `get_hiat_no()` to determine number of hiatuses in an adm
- `get_hiat_duration()` to determine duration of hiatuses

`get_hiat_no` *get no. of hiatuses*

Description

Determines the number of hiatuses in an age-depth model

Usage

```
get_hiat_no(x)
```

Arguments

`x`	an adm object

Value

An integer, no. of hiatuses in the age-depth model

See Also

- `get_hiat_pos()` to determine stratigraphic positions of hiatuses
- `get_hiat_list()` to determine position and timing of hiatuses
- `get_hiat_duration()` to determine duration of hiatuses

Examples

```
my_adm = tp_to_adm(t = 1:4, h = c(1,2,2,3)) # one hiatus
get_hiat_no(my_adm)
```

`get_hiat_pos` *get hiatus positions*

Description

Determines stratigraphic position of hiatuses

Usage

```
get_hiat_pos(x)
```

Arguments

x	an adm object

Value

numeric vector with stratigraphic positions of hiatuses

See Also

- `get_hiat_list()` to get hiatus positions and durations
- `get_hiat_no()` to determine number of hiatuses
- `get_hiat_duration()` to determine duration of hiatuses

Examples

```
my_adm = tp_to_adm(t = 1:4, h = c(1,2,2,3)) # one hiatus at height 2
get_hiat_pos(my_adm)
```

get_L_unit *extract length unit*

Description

extracts the length unit from adm or multiadm object

Usage

```
get_L_unit(x, ...)
```

Arguments

x	adm or multiadm object
...	other parameters

Value

character - the length unit of x

See Also

get_T_unit() set_L_unit()

get_time *Determine times based on age-depth model*

Description

Takes an age-depth model and vector of stratigraphic positions to determine the corresponding time of formation

Usage

```
get_time(x, h, hiat_mode = "start",
bdry_pts_hiat = "destructive", out_dom_val_t = "default")
```

Arguments

x	an *adm* or *multiadm* object
h	vector of stratigraphic positions
hiat_mode	"start", "end", or "destroy". If a stratigraphic position coincides with a hiatus, what should be returned?
bdry_pts_hiat	"consistent" or "destructive". How are hiatuses at the start/end of the adm treated?
out_dom_val_t	:"default", "time_limits", or a numeric value. What value is returned for heights not covered by the age-depth model?

Details

If a stratigraphic position coincides with a hiatus, should the start time or the end time of the hiatus be returned? Using "destroy" retruns NA If the adm starts/ends with a hiatus, should the time returned be consistent with *hiat_mode*, or should it be NA?

Value

numeric vector. Times of deposition of the provided heights in h

`get_total_duration` *Total duration covered*

Description

Total duration covered

Usage

```
get_total_duration(x)
```

Arguments

`x`	an adm object

Value

numeric vector, total duration covered by the age-depth models

`get_total_thickness` *get thickness*

Description

get thickness

Usage

```
get_total_thickness(x)
```

Arguments

`x`	an adm object

Value

numeric vector containing total sediment thickness accumulated

get_T_unit *extract Time unit*

Description

extracts the Time unit from adm or multiadm object

Usage

```
get_T_unit(x, ...)
```

Arguments

x	adm or multiadm object
...	other parameters

Value

character - the time unit of x

See Also

set_T_unit() get_L_unit()

is_adm *Is an adm object a valid age-depth model*

Description

Contructors for adm objects such as *tp_to_adm* do not check whether the inputs define a valid age-depth mode, e.g. one where the law of superposition holds. This function performs these checks

Usage

```
is_adm(x, quietly = TRUE)
```

Arguments

x	an object
quietly	logical. should descriptive warnings be shown?

Value

logical. Is the input a valid adm object?

`is_destructive` *Is deposition destructive?*

Description

Determines whether specified time is destrucive or not

Usage

```
is_destructive(x, t, mode = "rcll",
bdry_pts_hiat = "destructive", out_dom_mode = "default")
```

Arguments

`x`	an *adm* or *multiadm* object
`t`	vector of times
`mode`	string, either "rcll", "lcrl", "open", or "closed"
`bdry_pts_hiat`	string, "destructive" or "consistent". If the adm starts/ends with a hiatus, should the start/end be removed, or treated consistently with mode?
`out_dom_mode`	""default", "destructive", or "conservative"

Value

logical vector of same length as t. Is deposition at time t destructive?

`is_multiadm` *is valid multiadm object?*

Description

is valid multiadm object?

Usage

```
is_multiadm(x, quietly = TRUE)
```

Arguments

`x`	object to be tested
`quietly`	logical, should a descriptive warning be returned?

Value

Logical. Is the object a valid multiadm object?

is_sac *is valid sac objects*

Description

checks if the object is a valid sac object

Usage

```
is_sac(x)
```

Arguments

x	the object to check

Value

logical. Is x a valid sac object?

L_axis_lab *plot height axis label*

Description

plot height axis label

Usage

```
L_axis_lab(
  label = "Height",
  unit = TRUE,
  sep = " ",
  brac = c("[", "]"),
  line = 2,
  outer = FALSE,
  at = NA,
  adj = NA,
  padj = NA,
  cex = NA,
  col = NA,
  font = NA,
  ...
)
```

Arguments

`label`	Axis label
`unit`	Logical or character, should unit be plotted
`sep`	separator between label and unit
`brac`	brackets surrounding unit
`line`	parameter passed to *mtext*, see ?mtext for details
`outer`	parameter passed to *mtext*, see ?mtext for details
`at`	parameter passed to *mtext*, see ?mtext for details
`adj`	parameter passed to *mtext*, see ?mtext for details
`padj`	parameter passed to *mtext*, see ?mtext for details
`cex`	parameter passed to *mtext*, see ?mtext for details
`col`	parameter passed to *mtext*, see ?mtext for details
`font`	parameter passed to *mtext*, see ?mtext for details
`...`	further graphical parameters passed to *mtext*, see ?mtext for details

Value

invisible NULL

`make_legend` *plot legend*

Description

plots a legend for the multiadm plot

Usage

```
make_legend()
```

Value

invisible NULL

merge_adm_to_multiadm *combine multiple adm ojects into multiadm object*

Description

combine multiple adm ojects into multiadm object

Usage

```
merge_adm_to_multiadm(...)
```

Arguments

...	adm objects

Value

object of class multiadm

merge_multiadm *merge multiple multiadm objects*

Description

merge multiple multiadm objects

Usage

```
merge_multiadm(...)
```

Arguments

...	adm objects

Value

multiadm object

plot.adm *plotting adm objects*

Description

plotting adm objects

Usage

```
## S3 method for class 'adm'
plot(
  x,
  lwd_destr = 1,
  lwd_acc = 1,
  lty_destr = 3,
  lty_acc = 1,
  col_destr = "black",
  col_acc = "black",
  ...
)
```

Arguments

x	an adm object
lwd_destr	line width of hiatuse
lwd_acc	line width of conservative intervals
lty_destr	linetype of hiatuses
lty_acc	line type of conservative intervals
col_destr	color of erosive intervals
col_acc	color of conservative intervals
...	arguments passed to plot

plot.multiadm *plot multiadm object*

Description

plot multiadm object

Usage

```
## S3 method for class 'multiadm'
plot(x, ...)
```

Arguments

`x`	multiadm object
`...`	parameters passed to `plot`

Value

a plot of the multiadm object

`plot.sac` *plot sediment accumulation curve*

Description

plot sediment accumulation curve

Usage

```
## S3 method for class 'sac'
plot(x, ...)
```

Arguments

`x`	object of class *sac*
`...`	further parameters (currently ignored)

Value

invisible NULL

`plot_condensation` *plot condensation in height*

Description

plots condensation (time per stratigraphic increment) throughout the section

Usage

```
plot_condensation(x, h = "default", mode = "rcll", ...)
```

Arguments

`x`	an adm object
`h`	"default" or a numeric vector of height where the sed rate is evaluated
`mode`	string, handed over to *sed_rate_t*, see ?sed_rate_t for details
`...`	parameters passed to *get_time*, see ?get_time for details

Value

invisible null

`plot_erosive_intervals` *mark erosive time intervals*

Description

mark erosive time intervals

Usage

```
plot_erosive_intervals(
  density = NULL,
  angle = 45,
  col = "azure3",
  border = NA,
  lty = 1,
  lwd = 1
)
```

Arguments

`density`	parameter passed to *rect*, see ?rect for details
`angle`	parameter passed to *rect*, see ?rect for details
`col`	parameter passed to *rect*, see ?rect for details
`border`	parameter passed to *rect*, see ?rect for details
`lty`	parameter passed to *rect*, see ?rect for details
`lwd`	parameter passed to *rect*, see ?rect for details

Value

invisible NULL

`plot_sed_rate_l` *plot sed. rate in height*

Description

plot sed. rate in height

Usage

```
plot_sed_rate_l(x, h = "default", mode = "rcll", ...)
```

Arguments

`x` an adm object

`h` "default" or a numeric vector of height where the sed rate is evaluated

`mode` string, handed over to *sed_rate_t*, see ?sed_rate_t for details

`...` parameters passed to *get_time*, see ?get_time for details

Value

invisible null

`plot_sed_rate_t` *plot sedimentation rate in time*

Description

plot sedimentation rate in time

Usage

```
plot_sed_rate_t(x, mode = "rcll")
```

Arguments

`x` adm object

`mode` string, "rcll" or "lcrl". Should the sedimentation rate be Right Continuous with Left Limits (rcll) or Left Continuous with Right Limits (lcrl)

Value

invisible NULL

sac_to_adm *turn sed. acc curve into adm*

Description

turn sed. acc curve into adm

Usage

```
sac_to_adm(x)
```

Arguments

x	object of class *sac*

Value

object of class *adm*

See Also

`tp_to_adm()` for the generator of *adm*

sedrate_to_multiadm *Estimate age-depth model from sedimentation rates & tie points*

Description

Combines information on tie points and sedimentation rates to estimate age-depth models and their associated uncertainty. For an example, see `vignette("adm_from_sedrate")`.

Usage

```
sedrate_to_multiadm(
  h_tp,
  t_tp,
  sed_rate_gen,
  h,
  no_of_rep = 100L,
  subdivisions = 100L,
  stop.on.error = TRUE,
  T_unit = NULL,
  L_unit = NULL
)
```

Arguments

`h_tp` : function, returns stratigraphic positions of tie points

`t_tp` : function, returns times of tie points

`sed_rate_gen` : function, returns function describing sedimentation rate

`h` : numeric, heights where the adm is calculated

`no_of_rep` : integer, number of repetitions

`subdivisions` maximum no of subintervals used in numeric integration. passed to *integrate*, see ?stats::integrate for details

`stop.on.error` logical passed to *integrate*, see ?stats::integrate for details

`T_unit` time unit

`L_unit` length unit

Value

object of class multiadm

Examples

```
## Not run:
# see this vignette for an example
vignette("adm_from_sedrate")

## End(Not run)
```

`sed_rate_from_matrix` *make sed rate gen from matrix*

Description

at height `height[i]`, the sedimentation rate is specified by the pdf `approxfun(sedrate, matrix[i,])`

Usage

```
sed_rate_from_matrix(height, sedrate, matrix, rate = 1)
```

Arguments

`height` vector of heights

`sedrate` vector of sed. rates x values

`matrix` matrix of sed rate y values

`rate` numeric, rate of the Poisson point process determining frequency of sedimentation rate changes.

Value

a function factory for usage with `sedrate_to_multiadm`

See Also

`sedrate_to_multiadm()` for estimating sedimentation rates based on the outputs, `get_data_from_eTimeOpt()` for extracting data from the `eTimeOpt` function of the astrochron package.

`sed_rate_gen_from_bounds` *seg rate gen from upper/lower bounds*

Description

seg rate gen from upper/lower bounds

Usage

```
sed_rate_gen_from_bounds(h_l, s_l, h_u, s_u, rate = 1)
```

Arguments

`h_l`	height values for lower bounds
`s_l`	sed rate values for lower bounds
`h_u`	height values for upper bounds
`s_u`	sed rate values for upper bounds
`rate`	rate of poisson point process

Value

a function factory for usage with `sedrate_to_multiadm`

See Also

`sedrate_to_multiadm()` for estimating age-depth models using the outputs, `sed_rate_from_matrix()` for other means of defining sedimentation rates

sed_rate_l *sedimentation rate in stratigraphic height*

Description

determines instantaneous sedimentation rate at a specified stratigraphic position

Usage

```
sed_rate_l(x, h, mode = "rcll", ...)
```

Arguments

`x`	adm object
`h`	numeric vector, stratigraphic positions
`mode`	string, handed over to *sed_rate_t*, see ?sed_rate_t for details
`...`	parameters passed to *get_time*, see ?get_time for details

Value

a vector of sed rates (if x is an adm object), or a list of sedimentation rates

sed_rate_l_fun *sed rate in height function*

Description

returns a function that determines sed. rates in height

Usage

```
sed_rate_l_fun(x, mode = "rcll", ...)
```

Arguments

`x`	an adm object
`mode`	string, handed over to *sed_rate_t*, see ?sed_rate_t for details
`...`	parameters passed to *get_time*, see ?get_time for details

Value

a function

sed_rate_t *sedimentation rate in time domain*

Description

infers the instantaneous sedimentation rate from adm objects

Usage

```
sed_rate_t(x, t, mode = "rcll")
```

Arguments

x adm or multiadm object

t vector of times at which sedimentation rates are determined

mode string, "rcll" or "lcrl". at non-differential points, is the sed rate left or right continuous?

Value

for adm objects, a vector giving sed. accumulation rates at time t. For multiadm objects, a list with accumulation rates

sed_rate_t_fun *sedimentation rate function*

Description

returns a function that retruns sedimentation rate

Usage

```
sed_rate_t_fun(x, mode = "rcll")
```

Arguments

x an adm object

mode string, "rcll" or "lcrl". Should the sedimentation rate be Right Continuous with Left Limits (rcll) or Left Continuous with Right Limits (lcrl)

Value

a function

`set_L_unit` *set length units*

Description

set length units for adm and multiadm objects

Usage

```
set_L_unit(x, L_unit, ...)
```

Arguments

`x`	adm or multiadm object
`L_unit`	time unit
`...`	further parameters

Value

an adm or multiadm object with the L unit assigned

See Also

`set_T_unit()` `get_L_unit()`

`set_T_unit` *set time units*

Description

set time units for adm and multiadm objects

Usage

```
set_T_unit(x, T_unit, ...)
```

Arguments

`x`	adm or multiadm object
`T_unit`	time unit
`...`	further parameters

Value

an adm or multiadm object with the time unit assigned

See Also

set_L_unit() get_T_unit()

split_multiadm *split multiadm objects into adm*

Description

split multiadm objects into adm

Usage

```
split_multiadm(x)
```

Arguments

x	a multiadm object

Value

list with objects of class adm

strat_cont_gen_from_tracer *proxy values in strat domain*

Description

Generates a function factory for usage with *strat_cont_to_multiadm* based on empirical tracer measurements in the section

Usage

```
strat_cont_gen_from_tracer(
  bin_borders,
  df,
  distribution = "normal",
  cap = TRUE,
  cap_val = 0
)
```

Arguments

bin_borders	borders of sampling bins
df	data frame with proxy records
distribution	character, currently only "normal" implemented. Specifies the distribution of proxies
cap	logical. Should values below cap_valbe replaced?
cap_val	numeric. If cap = TRUE, values below cap_valwill be replaced by cap_val

Value

a functional for usage with strat_cont_to_multiadm

See Also

flux_const(), flux_linear(), flux_quad() to define tracer fluxes

Examples

```
## Not run:
# see this vignette for a use case
vignette("adm_from_trace_cont")

## End(Not run)
```

---

strat_cont_to_multiadm

estimate age-depth model from tracer

Description

Estimates age-depth models by comparing observed tracer values in a section with assumptions on tracer flux in time. See vignette("adm_from_trace_cont") for a full example.

Usage

```
strat_cont_to_multiadm(
  h_tp,
  t_tp,
  strat_cont_gen,
  time_cont_gen,
  h,
  no_of_rep = 100L,
  subdivisions = 100L,
  stop.on.error = TRUE,
  T_unit = NULL,
  L_unit = NULL
)
```

Arguments

`h_tp`	function, returning tie point heights
`t_tp`	function, returning tie points times
`strat_cont_gen`	function, describing tracer data observed in the section
`time_cont_gen`	function, describing tracer changes in time
`h`	numeric vector, heights where the age depth model is described
`no_of_rep`	integer, number of age depth models generated
`subdivisions`	integer, max no. of subintervals used by integration procedure. passed to *integrate*, see ?stats::integrate for details
`stop.on.error`	logical passed to *integrate*, see ?stats::integrate for details
`T_unit`	NULL or character, time unit
`L_unit`	NULL or character, length unit

Value

Object of class multiadm

Examples

```
## Not run:
# see this vignette for an example
vignette("adm_from_trace_cont")

## End(Not run)
```

`strat_to_time` *transform objects from strat. to time domain*

Description

Takes an object and transforms it from the time domain into the stratigraphic domain using the provided age-depth model. Currently implemented for the "phylo" class.

Usage

```
strat_to_time(obj, x, ...)
```

Arguments

`obj`	the object to be transformed
`x`	age-depth model
`...`	other parameters

Value

an object of the same type as obj

See Also

`time_to_strat()` to transform data from the time to the stratigraphic domain

---

`strat_to_time.list` *transform list from height to time domain*

Description

Lists are useful to keep data closely associated. This function transforms a list that contains observations associated with a stratigraphic position (recorded in the element with name "h") into a list where the observations are associated with time.

Usage

```
## S3 method for class 'list'
strat_to_time(obj, x, ...)
```

Arguments

`obj`	a list with one element named "h", which will be interpreted as stratigraphic positions
`x`	an *adm* object
`...`	options passed to *get_time*

Value

a list with one named element "t" instead of the element "h". This element contains the times of the stratigraphic positions in "h".

See Also

`time_to_strat.list()` for the transformation from time to height domain

Examples

```
# see vignette("admtools") for an example
```

`summary.adm` *summary of age-depth model*

Description

Displays some summary numbers of an age-depth models

Usage

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

Arguments

`object`	an adm object
`...`	other variables, are ignored

Value

Invisible NULL, prints summary to the console

Examples

```
my_adm = tp_to_adm(t = 1:5, h = c(2,2,3), L_unit = "m", T_unit = "Myr" )
summary(my_adm)
```

`summary.multiadm` *summary of age-depth model*

Description

Displays some summary numbers of an age-depth models

Usage

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

Arguments

`object`	a multiadm object
`...`	other variables, are ignored

Value

Invisible NULL, prints summary to the console

time_to_strat *transform objects from time domain to strat. domain*

Description

Takes an object and transforms it from the time domain into the stratigraphic domain using the provided age-depth model. Currently implemented for the "phylo" class.

Usage

```
time_to_strat(obj, x, ...)
```

Arguments

obj	the object to be transformed
x	age-depth model for the transformation
...	other parameters

Value

an object of the same type as obj

See Also

strat_to_time() to transform data from the stratigraphic domain to the time domain

time_to_strat.list *transform list from time to height domain*

Description

Lists are useful to keep data closely associated. This function transforms a list that contains observations associated with a time (recorded in the element with name "t") into a list where the observations are associated with stratigraphic position.

Usage

```
## S3 method for class 'list'
time_to_strat(obj, x, ...)
```

Arguments

obj	a list with one element named "t", which will be interpreted as time
x	an *adm* object
...	options passed to *get_height*

Value

a list with one named element "h" instead of the element "t", containing the stratigraphic positions corresponding to the times inf "t"

See Also

`strat_to_time.list()` for the transformation from height to time domain

Examples

```
# see vignette("admtools") for an example
```

---

tp_height_det *deterministic tie points height domain*

Description

defines deterministic stratigraphic tie points

Usage

```
tp_height_det(heights)
```

Arguments

`heights`	numeric vector. Stratigraphic positions of the tie points

Value

a function for usage with *strat_cont_to_multiadm* and *sedrate_to_multiamd* as t_tp input

See Also

`tp_time_norm()` for tie points following a normal distribution, `tp_time_floating_scale()` for tie points for a floating scale,

tp_time_floating_scale *tie points for floating time scale*

Description

Defines tie points for a floating time sclae for usage with *sedrate_to_multiadm* and *strat_cont_to_multiadm*

Usage

```
tp_time_floating_scale()
```

Value

function for usage with *strat_cont_to_multiadm* and *sedrate_to_multiamd* as t_tp input

See Also

`tp_time_norm()` for tie points following a normal distribution, `tp_height_det()` for deterministic height tie points

Examples

```
## Not run:
# see this vignette for an example
vignette("adm_from_trace_cont")

## End(Not run)
```

tp_time_norm *time tie points with normal distribution*

Description

defines a function factory that returns normally distributed times. FOr usage with `sedrate_to_multiadm` and `strat_cont_to_multiadm`.

Usage

```
tp_time_norm(mean, sd, force_order = TRUE)
```

Arguments

`mean`	numeric vector, mean age of tie points
`sd`	numeric vector, standard deviation of tie points
`force_order`	logical, enforce strictly increasing times

Value

function for usage with *strat_cont_to_multiadm* and *sedrate_to_multiamd* as t_tp input

See Also

`tp_time_floating_scale()` for tie points for a floating scale, `tp_height_det()` for deterministic height tie points

tp_to_adm — *Construct age-depth model from tie points*

Description

Turns tie points into an `adm` object that represents an age-depth model

Usage

```
tp_to_adm(t, h, T_unit = NULL, L_unit = NULL)
```

Arguments

`t`	Vector, tie points in time
`h`	Vector, tie points in height
`T_unit`	time unit
`L_unit`	length unit

Details

by default, intervals with no sediment accumulation are marked as destructive. `tp_to_adm` does not check whether the inputs define a valid age-depth model. For this, use `is_adm`

Value

object of class `adm`

See Also

`is_adm()` to check validity of `adm` objects

Examples

```
## Not run:
my_adm = tp_to_adm(t = 1:4, h = c(1,2,2,3), T_unit = "kyr", L_unit = "m")
plot(my_adm)
# see vignette("admtools") for other examples

## End(Not run)
```

tp_to_sac *define sed. acc. curve*

Description

defines *sac* (sediment accumulation curve) object from tie points

Usage

```
tp_to_sac(t, h, T_unit = NULL, L_unit = NULL)
```

Arguments

`t`	numeric vector, time coordinates of tie points
`h`	numeric vector, height coordinates of tie points
`T_unit`	time unit
`L_unit`	length unit

Value

a *sac* object reflecting a sediment accumulation curve

T_axis_lab *plot time axis label*

Description

plot time axis label

Usage

```
T_axis_lab(
  label = "Time",
  unit = TRUE,
  sep = " ",
  brac = c("[", "]"),
  line = 2,
  outer = FALSE,
  at = NA,
  adj = NA,
  padj = NA,
  cex = NA,
  col = NA,
  font = NA,
  ...
)
```

Arguments

`label` Axis label

`unit` Logical or character, should unit be plotted

`sep` separator between label and unit

`brac` brackets surrounding unit

`line` parameter passed to *mtext*, see ?mtext for details

`outer` parameter passed to *mtext*, see ?mtext for details

`at` parameter passed to *mtext*, see ?mtext for details

`adj` parameter passed to *mtext*, see ?mtext for details

`padj` parameter passed to *mtext*, see ?mtext for details

`cex` parameter passed to *mtext*, see ?mtext for details

`col` parameter passed to *mtext*, see ?mtext for details

`font` parameter passed to *mtext*, see ?mtext for details

`...` further graphical parameters passed to *mtext*, see ?mtext for details

Value

invisible NULL

Index