Package 'scuba'

July 2, 2015

Version 1.8-0

Date 2015-07-02

Title Diving Calculations and Decompression Models

Author Adrian Baddeley [aut, cre],
Vittorio Broglio [ctb, dtc],
Pedro Antonio Neves [ctb, dtc],
Andrew Bassom [ctb],
Peter Buzzacott [ctb]

Maintainer Adrian Baddeley <Adrian.Baddeley@uwa.edu.au>

Depends R (>= 3.0.0)

Imports utils, graphics, stats

Description Code for describing and manipulating scuba diving profiles
(depth-time curves) and decompression models,
for calculating the predictions of decompression models,
for calculating maximum no-decompression time and decompression tables,
and for performing mixed gas calculations.

License GPL (>= 2)

LazyData true

NeedsCompilation yes

Repository CRAN

Date/Publication 2015-07-02 16:07:25

R topics documented:

---

scuba-package *The Scuba Package*

Description

This is a summary of the features of scuba, a package in R that performs theoretical calculations about scuba diving — dive profiles, decompression models, gas toxicity and so on.

Details

scuba is a package for performing calculations in the theory of scuba diving. The package supports

- creation, manipulation and plotting of dive profiles
- gas diffusion models
- decompression calculations
- gas toxicity calculations.

The scuba package is intended only for use in research and education about the mathematical and statistical basis of decompression theory. It is emphatically not designed for actual use in scuba diving and related activities. See the detailed disclaimer in `scuba.disclaimer`.

Following is a summary of the main features of the package. For a more detailed explanation, with illustrations, see the vignette *Introduction to the Scuba package* which accompanies the package.

Dive profiles

A *dive profile* gives the diver's depth as a function of elapsed time during a scuba dive. The command `dive` creates an object representing a dive profile.

A dive profile is piecewise linear: it is a series of *stages* that join successive *waypoints*. Each waypoint is specified by the depth and elapsed time when it is reached. The stage between two waypoints is either a sojourn at a fixed depth, or an ascent or descent at a constant rate.

The function `dive` interprets its arguments as a sequence of actions or events occurring during the dive. If an argument is a vector of length 2, it is interpreted as `c(depth,time)` specifying the depth and duration of a stage of the dive. If the argument is a single number, it is interpreted as a depth, meaning that the diver ascends or descends to this depth.

For example, the command `d <- dive(c(18, 45))` specifies a dive to 18 metres for 45 minutes. The command `d <- dive(c(18, 45), c(5,3))` specifies a dive to 18 metres for 45 minutes followed by a safety stop at 5 metres for 3 minutes. Multilevel dives with any number of stages are specified in the same way. A dive object may include periods spent at the surface (depth zero) and may therefore represent a succession of dives separated by surface intervals. For example, `d <- dive(c(30,15),c(9,1),c(5,5),c(0,60),c(12,60),c(5,5))` represents two dives (with safety stops) separated by a one-hour surface interval.

The resulting object `d` is an object of class `"dive"`. It can be plotted as a conventional dive profile graph by executing the command `plot(d)`. It can be printed as a table of depths and times by typing its name `d`, or by executing `print(d,seconds=FALSE)` to print times to the nearest minute. A summary of the dive (with such information as the average depth, maximum depth and the main stages of the dive) can be printed by typing `summary(d)`.

By default, the function `dive` fills in some details about the dive. It assumes that the diver breathes compressed air; the dive starts and ends at the surface (depth zero); the diver descends at the default descent rate of 30 metres per minute; and the diver ascends at the default ascent rate of 18 metres per minute. These defaults can be changed by giving extra arguments to the function `dive`.

Dive profiles can also be modified after they are created: see below.

Real Dive Profiles

Dive profiles may also be uploaded from your dive computer and studied in the **scuba** package. First convert the uploaded profile data to a `data.frame` with two columns, the first column containing the elapsed time and the second column containing the depth (in metres) recorded at each time. The elapsed times can be either a vector of character strings in minutes-and-seconds format `mm:ss` or hours-minutes seconds `hh:mm:ss`, or a vector of integer times measured in *seconds* of elapsed time, or an object of class `difftime` containing the elapsed times in any time unit. Then pass this data frame as an argument to the function `dive`.

An example of such a data frame, uploaded from a dive computer, is provided in the `baron` dataset supplied with the package. See the help file for `baron` for an example of how to convert a data frame to a `"dive"` object.

The package also provides 12 real dive profiles that have already been converted to `"dive"` objects. See the help files for `pedro` and `deepmine`.

Decompression Calculations

The **scuba** package performs the mathematical calculations of decompression theory:

- the theoretical No Decompression Limit (maximum duration of a no-decompression dive to a specified depth) can be computed by `ndl(depth)`
- the quantity of nitrogen dissolved in the diver's body after a dive `d` can be computed by `haldane(d)`
- the quantity of nitrogen dissolved in the diver's body at each instant **during** a dive `d` can be computed by `haldane(d, progressive=TRUE)` or plotted interactively by `showstates(d)`.

These calculations are based on the classical theory of decompression originated by Haldane (Boycott et al, 1908). The diver's body is idealised as a set of independent compartments, each connected directly to the breathing gas, and governed by classical (exponential) diffusion.

The model parameters (the number of compartments, their diffusion rates, and the maximum tolerated nitrogen tension in each compartment) may be chosen by the user. By default, the model parameters are taken from the DSAT model which is the basis of the PADI Recreational Dive Planner. Alternatively, the user can choose from a variety of standard compartment models using the command `pickmodel`, or construct a new model using `hm`.

No-decompression limits (the maximum duration of a no-decompression dive to a given depth) can be calculated using the function `ndl`. For example `ndl(30)` gives the theoretical NDL for a dive to 30 metres, predicted by the DSAT model. To use the classical US Navy model instead, type `ndl(30, model="USN")` or `ndl(30, model=pickmodel("USN"))`.

The 'best' double no-decompression dive to given depths `d1` and `d2` can be calculated by `bestdoubledive` according to the algorithm of Baddeley and Bassom (2012).

The nitrogen tension (the quantity of dissolved nitrogen, in atmospheres absolute) in the diver's body after a dive, can be calculated using the function `haldane`. If `d` is a dive object then `haldane(d)` returns a data frame containing the nitrogen tissue tensions (ata) at the end of the dive, in each of the 8 tissue compartments of the DSAT model. To use the US Navy model instead, type `haldane(d, "USN")` or `haldane(d, pickmodel("USN"))`.

To compute the nitrogen tissue tensions at each waypoint during the dive, use `haldane(d, progressive=TRUE)`.

To visualise the nitrogen tissue tensions during the dive, use the interactive function showstates. This plots the dive and waits for you to click on a position in the graph. The tissue tensions at that instant are displayed as a bar plot.

The total oxygen toxicity incurred during a dive can be computed by oxtox.

Oxygen partial pressure at each stage of a dive is computed by ppO2.

Bubble theory calculations are not yet implemented.

Nitrox and trimix

A **breathing gas** is represented by an object of class "gas". The object air is a representation of compressed air (21% oxygen, 79% nitrogen) as an object of this class. (Don't reassign another value to this object!!!)

Nitrox mixtures (mixtures of oxygen and nitrogen) can be represented using the function nitrox. For example, EAN 32 is represented by nitrox(0.32).

Trimix (a mixture of oxygen, nitrogen and helium) can also be represented, using the command trimix. For example, Trimix 15/50 (containing 15% oxygen, 50% helium and 35% nitrogen) is represented by trimix(0.15, 0.5).

There are methods for print and summary for gas objects.

Decompression calculations (haldane, ndl, showstates) also work with nitrox and trimix.

Decompression calculations with trimix require a Haldane model that includes parameters for Helium diffusion. Use pickmodel("Z") to select the Buehlmann ZH-L16A model, or hm to create a new model that includes Helium diffusion.

Standard nitrox calculations are also available, for example

ead	equivalent air depth
mod	maximum operating depth
maxmix	richest nitrox mix for a given depth

The total oxygen toxicity incurred during a nitrox or trimix dive can be computed by oxtox. Oxygen partial pressure at each stage of a dive is computed by ppO2.

Diving on different gases

Every "dive" object contains information about the breathing gas or gases used in the dive. This information is determined when the "dive" object is created (by the function dive). The default breathing gas is air.

The function dive interprets its arguments as a sequence of actions or events occurring during the dive. If an argument is a vector of length 2, it is interpreted as c(depth,time) specifying the depth and duration of a stage of the dive. If the argument is a single number, it is interpreted as a depth, meaning that the diver ascends or descends to this depth.

Each argument to dive may also be a "gas" object, like nitrox(0.32), which means that the diver switches to this gas.

So, for example, dive(nitrox(0.32), c(30,20)) means a dive to 30 metres for 20 minutes conducted on EAN 32 (Nitrox 0.32) from start to finish. The command dive(c(30,20), 5, nitrox(0.36), c(5,3))

means a dive on air to 30 metres for 20 minutes, ascending to 5 metres while breathing air, then switching to EAN 36 for a safety stop at 5 metres for 3 minutes.

Alternatively you can use the argument `tanklist` to specify a list of tanks of breathing gas (with optional names like `"travel"` and `"deco"`) and change between tanks at different stages of the dive using an argument of the form `tank=number` or `tank=name`. The tank list of a dive object can be extracted using `tanklist` and modified using `tanklist<-`.

Manipulating dive profiles

Dive profiles can also be manipulated after they are created. This allows you, for example, to modify the deepest portion of a dive (diving to a deeper depth or for a longer duration), to abort a dive prematurely, to cut-and-paste several dives together, or to consider the tissue saturation incurred by a particular segment of a dive.

The commands `depths.dive` and `times.dive` extract the depths and elapsed times at each waypoint during the dive. The depths can be modified using `depths.dive<-`. For example `d <- dive(c(30,20))` creates a dive to 30 metres for 20 minutes, starting and finishing at the surface; to change the depth to 35 metres, type `depths.dive(d)[2:3] <- 35`. Similarly the elapsed times can be modified using `times.dive<-`. It may be more convenient to use the functions `durations.dive` and `durations.dive<-` which give the duration of each stage (the time between two successive waypoints). For example `durations.dive(d)[2] <- 25` would mean that the diver now spends 25 minutes at the bottom instead of 20 minutes.

To extract only part of a dive profile, use `chop.dive`.

To paste together two dive profiles or fragments of dive profiles, simply give them as arguments to `dive`.

Changing the breathing gases

A dive object has a *tank list* which is a list of the tanks of breathing gas that were used (or were available to be used) during the dive. The function `tanklist` returns this list, and the function `tanklist<-` changes the list.

For example, `d <- dive(c(30,20), c(5,5))` is a dive conducted using air. To modify it to a dive that used nitrox EANx 32, simply type `tanklist(d) <- list(nitrox(0.32))`. Again `d <- dive(air, c(30,40), 6, nitrox(0.5), c(6,3), c(3,3))` is a dive conducted using air (tank 1) for the deep section and EANx 50 (tank 2) for the decompression stops at 6 metres and 3 metres. To change the contents of tank 1 to EANx 32, type `tanklist(d) <- list(nitrox(0.32), nitrox(0.5))` or just `tanklist(d)[[1]] <- nitrox(0.32)`. To associate a name which each tank, give names to the entries in the tank list, for example `tanklist(d) <- list(deep=nitrox(0.32), deco=nitrox(0.5))` or just assign `names(tanklist(d)) <- c("deep", "deco")`.

The *selection* of tanks, i.e. which tank is actually used at each stage of the dive, is specified by the function `whichtank`. The command `whichtank(d)` returns a vector of integers or character strings, identifying which tank in the tank list is in use at each waypoint during the dive. That is, `whichtank(d)[i]` is the tank in use at the `i`th waypoint during the dive. The vector `whichtank(d)` has the same length as the vectors `depths.dive(d)` and `times.dive(d)`.

To change the selection of tanks at each stage during the dive, use the function `whichtank<-`. For example, `d <- dive(air, c(30,40), 6, nitrox(0.5), c(6,3), c(3,3))` is a dive conducted using air (tank 1) for the deep section and EANx 50 (tank 2) for the decompression stops at 6 metres and 3 metres. To change this so that the deco gas is only used at the 3-metre stop, type

`whichtank(d) <- ifelse(depths.dive(d) < 3, 1, 2)`. Alternatively `whichtank(d)[depths.dive(d) > 3] <- 1` would select tank 1 for all parts of the dive deeper than 3 metres. These manipulations are usually easier to understand if the tanks have names. For example typing `names(tanklist(d)) <- c("deep", "deco")` we could then type `whichtank(d) <- ifelse(depths.dive(d) < 3, "deep", "deco")` or `whichtank(d)[depths.dive(d) > 3] <- "deep"`.

Licence

This library and its documentation are usable under the terms of the "GNU General Public License", a copy of which is distributed with the package.

Author(s)

Adrian Baddeley <Adrian.Baddeley@uwa.edu.au> http://www.maths.uwa.edu.au/~adrian/ with contributions from Vittorio Broglio and Pedro Antonio Neves.

References

Baddeley, A. (2013) *Introduction to the scuba package*. Vignette accompanying this package.

Baddeley, A. and Bassom, A.P. (2011) Classical theory of decompression and the design of scuba diving tables. *The Mathematical Scientist* **36**, 75-88.

Bookspan, J. (1995) Diving physiology in plain English. Undersea and Hyperbaric Medicine Society, Kensington, Maryland (USA). ISBN 0-930406-13-3.

Boycott, A.E. Damant, G.C.C. and Haldane, J.S. (1908) The prevention of compressed air illness. *Journal of Hygiene* (London) **8**, 342–443.

Brubakk, A.O. and Neuman, T.S. (eds.) (2003) Bennett and Elliott's Physiology and Medicine of Diving. 5th Edition. Saunders. ISBN 0-7020-2571-2

Buehlmann, A.A. (1983) *Dekompression - Dekompressionskrankheit*. Springer-Verlag.

Buehlmann, A.A., Voellm, E.B. and Nussberger, P. (2002) *Tauchmedizin*. 5e Auflage. Springer-Verlag.

Tikvisis, P. and Gerth, W.A. (2003) Decompression Theory. In Brubakk and Neuman (2003), Chapter 10.1, pages 419-454.

Wienke, B.R. (1994) *Basic diving physics and applications*. Best Publishing Co.

Workman, R.D. (1965) Calculation of decompression schedules for nitrogen-oxygen and helium-oxygen dives. Research Report 6-65. US Navy Experimental Diving Unit. Washington DC.

`air` *Air*

Description

A dataset representing air as a breathing gas, and a function which tests whether a gas is air.

Usage

```
air
is.air(g)
```

Arguments

g	An object of class "gas".

Details

An object of class "gas" represents a breathing gas. Such objects are required for various calculations in the scuba library.

The dataset air represents compressed air (21% oxygen, 79% nitrogen) as a breathing gas. It is equivalent to nitrox(0.21).

The function is.air expects its argument g to be a gas object. It returns TRUE if g is equivalent to air, and FALSE otherwise.

Author(s)

Adrian Baddeley <Adrian.Baddeley@uwa.edu.au> http://www.maths.uwa.edu.au/~adrian/

See Also

nitrox

Examples

```
v <- air
is.air(v)
v <- nitrox(0.21)
is.air(v)
# both are TRUE
```

---

ascent *Ascent Rate or Time*

Description

Specify an Ascent Rate or Ascent Time.

Usage

```
ascent(speed=NULL, time=NULL)
```

Arguments

speed	Ascent rate in metres per minute. Incompatible with time.
time	Total ascent time in minutes. Incompatible with speed.

Details

An object of class "rate" represents a diver's rate of ascent or descent, or the total time taken to ascend or descend. Such objects are useful in describing dives, especially in the function dive.

The value returned by ascent represents an ascent rate or ascent time. Exactly one of the arguments speed and time should be given. If speed is given, this specifies a fixed rate of ascent, in metres per minute. If time is given, this specifies a fixed total time of ascent, in minutes.

Value

An object of class "rate" representing the ascent rate or ascent time.

Author(s)

Adrian Baddeley <Adrian.Baddeley@uwa.edu.au> http://www.maths.uwa.edu.au/~adrian/

See Also

dive, descent

Examples

```
# Ascend at 18 metres/minute
ascent(18)
# Ascend in exactly 3 minutes
ascent(time=3)
```

---

baron *Real Scuba Dive Profile*

Description

Contains the dive profile data for a scuba dive on the *Baron Gautsch* wreck, uploaded from a dive computer.

Format

A data frame with the following columns:

time	character	elapsed time (minutes:seconds)
depth	numeric	depth (metres)
temp	numeric	temperature (C)
bar	numeric	breathing gas pressure (bar)
RBT	integer	residual bottom time (min)
WL	integer	workload

Details

This dataset contains the dive profile data for a real scuba dive uploaded from a dive computer. It gives the depth, water temperature, breathing gas cylinder pressure, calculated residual bottom time, and calculated workload, recorded every 4 seconds throughout the dive.

The dive site was the wreck of the vessel *Baron Gautsch* in Croatia. The dive, to a maximum depth of 40 metres, with total dive time of 40 minutes, was conducted on nitrox (EAN 30) by Vittorio Broglio.

The examples show how to convert the depth-time data to a `dive` object.

Source

Vittorio Broglio

Examples

```
data(baron)
b <- dive(nitrox(0.30), baron[,1:2])
```

---

bestdoubledive *Find the Best Double Dive To Given Depths*

Description

For two no-decompression dives, to depths `d1` and `d2` metres respectively, separated by a given time interval, find the optimal durations of the two dives.

Usage

```
bestdoubledive(d1, d2, surface = 30, verbose = FALSE, model = "D", asdive=TRUE)
```

Arguments

`d1`	Depth of first dive in metres.
`d2`	Depth of second dive in metres.
`surface`	Length of surface interval between dives, in minutes.
`verbose`	Logical. If `TRUE`, print lots of extra information, and return extra information. If `FALSE` (the default), just return the optimal dive.
`model`	The decompression model. Either a character string, containing the name of a decompression model recognised by `pickmodel`, or an object of class `"hm"` (created by `hm`) representing a decompression model. Defaults to the DSAT model.
`asdive`	Logical. If `TRUE` (the default), the data for the optimal dive are converted into a dive object (object of class `"dive"`). If `FALSE`, the data are returned as a data frame.

Details

This command implements the algorithm described by Baddeley and Bassom (2012) which calculates the 'best' double dive to two given depths separated by a given surface interval, without exceeding the no-decompression limits.

Consider a no-decompression dive to depth d_1 metres for t_1 minutes, followed by a surface interval of s minutes, followed by a no-decompression dive to depth d_2 for t_2 minutes. The 'best' double dive is defined by Baddeley and Bassom (2012) as the one which maximises the integral of depth $\Phi = t_1 d_1 + t_2 d_2$.

Value

By default (when `verbose=FALSE` and `asdive=TRUE`) the result is a dive object (object of class `"dive"`).

Otherwise, the result is a data frame with columns `t1` and `t2` containing the dive durations in minutes, `phi` containing the value of Φ, and `case` specifying which of the cases specified in Baddeley and Bassom (2012) provided the optimum. If `verbose=FALSE` this data frame has only one row, giving the best double dive. If `verbose=TRUE` then the data frame has several rows giving the candidates for optimal dive in each case of the algorithm.

Author(s)

Adrian Baddeley <Adrian.Baddeley@uwa.edu.au> http://www.maths.uwa.edu.au/~adrian/

References

Baddeley, A. and Bassom, A.P. (2011) Classical theory of decompression and the design of scuba diving tables. *The Mathematical Scientist* **36**, 75-88.

See Also

`pickmodel`, `hm`, `ndl`, `haldane`

Examples

```
d <- bestdoubledive(40, 12, 15)
plot(d)
# Table 3 in Baddeley and Bassom (2012)
bestdoubledive(40, 12, 15, verbose=TRUE)
# Brief description of optimal dive
summary(bestdoubledive(40, 12, 15))
# Data for optimal dive
bestdoubledive(40, 12, 15, asdive=FALSE)
```

Bookspan *Tissue data from Bookspan's book*

Description

Data for several decompression models of Haldane type, giving the tissue halftimes for Nitrogen and the M-values (maximum tolerated partial pressure of Nitrogen for surfacing).

Format

The dataset Bookspan is a list with the following entries:

Halftimes Halftimes data. A list with entries

Haldane	halftimes for Haldane's original model
USN	halftimes for US Navy original model
DSAT	halftimes for DSAT model
OrcaEdge	halftimes for OrcaEdge computer
MicroBrain	halftimes for MicroBrain computer
ZHL12	halftimes for ZH-L12 model

Mvalues.ata surfacing M-values in ata (atmospheres absolute). A list with entries

Haldane	M-values for Haldane's original model
USN	M-values for US Navy original model
DSAT	M-values for DSAT model

Mvalues.fsw surfacing M-values in fsw (feet of seawater). A list with the same structure as Mvalues.fsw.

Details

Bookspan (1995) tabulates the parameters of several different decompression models of the classical Haldane type. The basic parameters for such a model are

halftimes: the Nitrogen diffusion halftime (in minutes) for each tissue in the model

M-values: the maximum partial pressure of Nitrogen in each tissue which would be tolerated without symptoms at an ambient pressure of 1 atmosphere (the "surfacing M-value").

The data tabulated in Bookspan (1995, page 16) purportedly give the halftimes for Haldane's original model, the original US Navy model, the DSAT model (basis of PADI repetitive dive planner), the OrcaEdge, MicroBrain and Aladin dive computers, and the Buehlmann ZH-L12 model.

The data tabulated in Bookspan (1995, page 23) give the corresponding M-values (in fsw) for the US Navy and DSAT models only.

These parameters are all contained in the dataset Bookspan.

Author(s)

Adrian Baddeley <Adrian.Baddeley@uwa.edu.au> http://www.maths.uwa.edu.au/~adrian/

Source

Bookspan (1995), page 16 (halftime data) and page 23 (M-values).

References

Bookspan, J. (1995) Diving physiology in plain English. Undersea and Hyperbaric Medicine Society, Kensington, Maryland (USA). ISBN 0-930406-13-3.

Examples

```
data(Bookspan)
Bookspan$Halftimes$DSAT
```

BuehlmannL16A *Decompression model ZH-L16A*

Description

The parameters of the decompression model ZH-L16A developed by Buehlmann.

Format

A data frame giving the following parameters for each compartment:

`id`	compartment name (1, 1b, 2, . . . , 16)
`halftime`	Nitrogen halftime (minutes)
`a`	parameter a in `ata` (atmospheres absolute)
`b`	parameter b (dimensionless)

Details

This dataset contains the *tissue parameters* of the decompression model ZH-L16A.

Buehlmann's decompression model ZH-L16A is a pure diffusion (Haldane-type) model consisting of 17 tissue compartments. Nitrogen in the breathing gas diffuses in and out of each compartment at a rate governed by the halftime for that compartment. The maximum tolerable nitrogen tension in a compartment, when the ambient pressure is P atmospheres absolute, is

$$P_{\text{tol}} = a + \frac{P}{b}$$

where a and b are values intrinsic to the tissue. Thus, for a no-decompression dive at sea level, the nitrogen tension in each compartment must not exceed the value $a + 1/b$ for that compartment.

The dataset `BuehlmannL16A` is a table giving the values of a and b for each of the 17 tissue compartments.

Decompression calculations

The dataset BuehlmannL16A is just a table of numbers. It cannot be used directly for decompression calculations, because the **scuba** package does not recognise it as a decompression model (object of class "hm").

To perform decompression calculations with the ZH-L16A model, use the functions haldane or showstates with the argument model="ZH-L16A" (or just model="Z").

To convert a table of a and b values, such as the table above, into an object of class "hm" representing a decompression model, first calculate the M-values M0 = a + 1/b and dM = 1/b, and then pass the values M0, dM to the function hm.

Author(s)

Adrian Baddeley <Adrian.Baddeley@uwa.edu.au> http://www.maths.uwa.edu.au/~adrian/

Source

Buehlmann et al (2003), Table 25, page 158.

References

Buehlmann, A.B., Voellm, E.B. and Nussberger, P. (2002) Tauchmedizin. 5th edition. Springer-Verlag, Berlin. ISBN 3-540-42979-4.

See Also

Workman65

---

chop.dive *Extract Part of a Dive Profile*

Description

Extracts only part of a dive profile.

Usage

```
chop.dive(d, t0=0, t1=max(times.dive(d)))
```

Arguments

d	The dive. An object of class "dive".
t0, t1	Elapsed times (in minutes) of the start and end of the period which should be extracted.

Details

The argument d should be an object of class "dive" representing a dive profile.

This command extracts the part of the dive profile that starts at time t0 minutes and ends at time t1 minutes, and returns it as an object of class "dive". The clock is adjusted so that the new dive profile starts at time 0 and ends at time t1-t0.

Note that the resulting dive profile does not start and end at the surface: the dive profile is simply chopped off at elapsed time t1. If you want the dive to start and end at the surface, execute dive(chop.dive(d, t0, t1)).

Value

An object of class "dive" starting at time 0 and ending at time t1-t0.

Author(s)

Adrian Baddeley <Adrian.Baddeley@uwa.edu.au> http://www.maths.uwa.edu.au/~adrian/

See Also

dive

Examples

```
d <- dive(c(30,20), c(5,5))
# fragment of dive up to 10 minutes
chop.dive(d, 0, 10)
# dive aborted at 10 minutes
dive(chop.dive(d, 0, 10))
```

---

deco.ceiling *Decompression Ceiling for a Diver*

Description

Given the state of a diver's tissues, find the diver's decompression ceiling depth (minimum permissible depth) or decompression ceiling pressure (minimum permissible ambient pressure).

Usage

```
deco.ceiling(state, model, what = c("depth", "pressure"))
```

Arguments

state	Diver's tissue saturations. See Details.
model	Haldane type decompression model (object of class "hm") or one of the standard model names recognised by pickmodel.
what	What to calculate. See Details.

Details

Given the current state of a diver's tissues, this command calculates the diver's decompression ceiling: the shallowest depth (or lowest ambient pressure) to which it is safe to ascend.

The argument `state` gives the diver's current state or the diver's progressive state during a dive. It has typically been calculated using `haldane`. It should be either:

- A numeric vector giving the diver's current nitrogen tensions for each tissue. The length of the vector matches the number of tissues in the `model`.
- A matrix giving the diver's current nitrogen and (if present) helium tensions for each tissue. The rows of the matrix correspond to tissues in the model. The first column (or the column labelled `"N2"`) contains the nitrogen tensions, and if present, the second column (or column labelled `"He"`) contains the helium tensions.
- A three-dimensional array giving the progressive states of the diver over time. The first index corresponds to time. The second index corresponds to tissues in the model. The third coordinate corresponds to inert gases `"N2"` and `"He"`.

Entries in the vector, matrix or array `state` are pressures in atmospheres absolute (ata).

The minimum tolerated ambient pressure `Pamb.tol` is calculated using the equivalent of Buehlmann's formula (Buehlmann et al, 1995, section 6.5, page 117). If `what="pressure"`, this minimum tolerated ambient pressure is returned (negative pressures are reset to zero). If `what="depth"` this pressure is converted to diving depth in metres assuming that the surface pressure is 1 ata (negative depths are reset to zero).

Value

A numeric vector, matrix or array of the same type as `state` and with the same dimensions as `state`, giving the decompression ceiling for each tissue, each inert gas, and each time point as given. If `what="pressure"` the entries in the result are pressures in ata. If `what="depth"` the entries are depths in metres.

Author(s)

Adrian Baddeley <Adrian.Baddeley@uwa.edu.au> http://www.maths.uwa.edu.au/~adrian/

References

Buehlmann, A.B., Voellm, E.B. and Nussberger, P. (2002) Tauchmedizin. 5th edition. Springer-Verlag, Berlin. ISBN 3-540-42979-4.

See Also

`hm` for decompression models.

`haldane` to calculate tissue tensions during or after a dive.

Examples

```
# diver state after saturation diving on EANx30 at 20 metres
s <- saturated.state("Z", 20, nitrox(0.30))

# decompression ceiling for each compartment
depthceiling <- deco.ceiling(s, "Z")
round(depthceiling, 2)
round(max(depthceiling), 2)
# decompress at 8 metres or deeper

# flying-after-diving with cabin pressure 0.8 ata
pressureceiling <- deco.ceiling(s, "Z", "pressure")
round(pressureceiling, 2)
any(pressureceiling > 0.8)
# immediate flying-after-diving is not safe
```

---

deepmine *Extremely Deep Decompression Dive*

Description

Contains the dive profile for a long, extremely deep, decompression dive, using mixed gases, in a flooded mine.

Format

An object of class `"dive"`.

Details

This dataset contains the dive profile data for an extreme dive in a flooded mine.

This was a dive to 110 metres for 7 minutes, followed by 147 minutes of staged decompression, including 40 minutes in a dry habitat at 4 metres. Five different mixed gases (trimix and nitrox) were breathed.

The diver suffered decompression sickness after this dive. An analysis of the dive profile was presented by Buzzacott et al (2013).

The dataset is an object of class `"dive"` containing the dive profile. It can be plotted and printed (using `plot.dive` and `print.dive`). The nitrogen saturation can be computed using `haldane` and the cumulative oxygen toxicity using `oxtox`.

Source

Peter Buzzacott.

References

Buzzacott, P., Papadopoulou, V., Baddeley, A., Petri, N. and Lind, F. (2013) Exceptional diving exposure with deep stops and DCS (case study). Conference Poster, European Underwater and Baromedical Society/ South Pacific Underwater Medicine Society/ Southern African Underwater and Hyperbaric Medicine Association, Tricontinental Scientific Meeting on Diving and Hyperbaric Medicine, La Reunion, September 23-28, 2013.

Examples

```
data(deepmine)
plot(deepmine, legendpos="right")
```

`depths.dive` *Depths at each waypoint of a dive*

Description

Extracts, or alters, the depth at each waypoint during a dive.

Usage

```
depths.dive(d)
depths.dive(d) <- value
```

Arguments

`d`	A dive (object of class `"dive"`).
`value`	A numeric vector containing depths in metres.

Details

An object of class `"dive"` represents a scuba dive profile. It is created by the function `dive`. A dive is defined as a series of waypoints occurring at specified depths and times. The depth at each waypoint is returned by `depths.dive`. The assignment `depths.dive(d) <- value` alters these depths, provided the new vector `value` has the same length as the old one.

Value

`depths.dive` returns a numeric vector containing the depths at each waypoint, in metres.

Author(s)

Adrian Baddeley <Adrian.Baddeley@uwa.edu.au> http://www.maths.uwa.edu.au/~adrian/

See Also

`times.dive`, `durations.dive`, `dive`.

Examples

```
d <- dive(c(30,20), c(5,5))
d
depths.dive(d)
# what if we had dived to 35 metres?
depths.dive(d)[2:3] <- 35
d
```

`descent` *Descent Rate or Time*

Description

Specify an Descent Rate or Descent Time.

Usage

```
descent(speed=NULL, time=NULL)
```

Arguments

`speed`	Descent rate in metres per minute. Incompatible with `time`.
`time`	Total descent time in minutes. Incompatible with `speed`.

Details

An object of class `"rate"` represents a diver's rate of ascent or descent, or the total time taken to ascend or descend. Such objects are useful in describing dives, especially in the function `dive`.

The value returned by `descent` represents a descent rate or descent time. Exactly one of the arguments `speed` and `time` should be given. If `speed` is given, this specifies a fixed rate of descent, in metres per minute. If `time` is given, this specifies a fixed total time of descent, in minutes.

Value

An object of class `"rate"` representing the descent rate or descent time.

Author(s)

Adrian Baddeley <Adrian.Baddeley@uwa.edu.au> http://www.maths.uwa.edu.au/~adrian/

See Also

`dive`, `descent`

Examples

```
# Descend at 30 metres/minute
descent(30)
# Descend in exactly 3 minutes
descent(time=3)
```

`dive` *Define a Dive Profile*

Description

Define a dive profile.

Usage

```
dive(..., begin=0, end=0, tanklist=NULL)
```

Arguments

`...`	Any number of arguments, specifying a sequence of events that make up the dive. The arguments may specify depths, time spent at each depth, ascent or descent rates, and gas switches. See Details.
`begin, end`	The depths at the start and finish of the dive (normally zero indicating that the dive starts and finishes at the surface) or `NA`.
`tanklist`	Optional list of the gases contained in each tank, for a dive with multiple tanks.

Details

This function creates an object of class `"dive"` which represents a scuba dive. The dive profile is assumed to be piecewise linear, that is, the graph of depth against time is a broken straight line. Dives are assumed to start and finish at the surface.

The arguments `...` specify a succession of events that make up the dive. Each argument can be

a vector of length 2 interpreted as `c(depth, duration)` where `depth` gives the depth in metres and `duration` the length of stay at this depth, in minutes.

a single number interpreted as a waypoint depth in metres. The diver will ascend or descend to this depth, at the current default rate of ascent or descent.

an object of class `"gas"` specifying a breathing gas. Such objects are created by the functions `nitrox` and `trimix`. The diver switches to this gas.

an argument of the form `tank=n` specifying a switch to another tank or cylinder of breathing gas. This is available only when the tanks used in the dive have been specified by the argument `tanklist`. The value `n` should be a valid index for this list (either a serial number in the tank list or a character string matching one of the names in the tank list).

an ascent/descent rate object created by the functions `ascent` or `descent`. This resets the default rate of ascent or descent.

a data frame with 2 columns containing a dive profile, usually uploaded from a dive computer. The first column should contain the *elapsed* times, and the second column contains the depths (in metres) measured at these times. The column of elapsed times can be either a character vector containing times in minutes-and-seconds format `mm:ss`, or an integer vector containing elapsed times in **seconds**, or a vector of class `difftime` representing elapsed times in any time unit. In this case you probably want to set the arguments `begin=NA` and `end=NA`.

an object of class `"dive"` representing a dive profile. This allows the user to paste dive profiles together.

Dives are assumed to start and finish at the surface. This can be changed by specifying values for `begin` and `end`. See the section on Start and End of Dive.

Initially the descent rate is set to 30 metres per minute, the ascent rate is 18 metres per minute, and the breathing gas is air. These settings may be changed during the dive by the `...` arguments.

A dive object may include periods spent at the surface (depth zero) and may therefore represent a succession of dives separated by surface intervals.

Once an object of class `"dive"` has been created, it can be plotted and printed (using `plot.dive` and `print.dive`). The nitrogen saturation can be computed using `haldane` and the cumulative oxygen toxicity using `oxtox`.

Value

An object of class `"dive"` describing the dive profile and the breathing gases used.

Start and End of Dive

To make it easy to specify simple dive profiles, the algorithm fills in some information. If the diving depths specified in the arguments `...` do not start at the surface (depth zero), then by default, the algorithm assumes that the dive did start at the surface, and it adds an extra dive segment at the beginning of the dive. The diver is assumed to descend from the surface, at the standard descent rate, to the first depth specified. Thus `dive(c(10,25))` means that the diver will first descend from the surface to 10 metres at a rate of 30 metres per minute, remain there for 25 minutes, then ascend to the surface at a rate of 18 metres per minute.

To change this behaviour, set a different value for the argument `begin`. If the dive really started at a nonzero depth, e.g. a dry habitat at 3 metres, set `begin=3`. If one of the arguments `...` is a data frame uploaded from a dive computer, then set `begin=NA`, so that that the dive will start at the first depth specified in this data frame.

Similarly, if the last depth specified by the arguments `...` is a depth below the surface (depth greater than zero), then by default the diver is assumed to ascend from this last depth to the surface, at the standard ascent rate. To suppress this altogether, set `end=NA`. To specify a different final depth for the dive, set a different value for `end`.

Modifying a dive object

The depths and elapsed times at each waypoint during the dive can be extracted and changed using `depths.dive` and `times.dive`.

Dives can be cut into pieces using `chop.dive` and pasted together using `dive`.

It is possible to alter the *breathing gases* used in a dive d, yielding a new dive object. This makes it possible to study the effect of conducting the same dive profile with different breathing gases. You can change the specification of the breathing gas (or gases) by tanklist(d) <- value. In a multiple-tank dive you can change the periods when each tank was used, by whichtank(d) <- value.

Warnings

Not suitable for representing altitude dives.

Author(s)

Adrian Baddeley <Adrian.Baddeley@uwa.edu.au> http://www.maths.uwa.edu.au/~adrian/

See Also

nitrox, ascent, descent, haldane, plot.dive, times.dive, durations.dive, depths.dive, chop.dive, tanklist, whichtank.

Examples

```
# Dive to 25 m for 20 min followed by safety stop at 5 metres for 3 min
d <- dive(c(25,20),c(5,3))
plot(d)

# Bounce dive to 20 metres
d <- dive(20)

# Two dives separated by a one-hour surface interval
d <- dive(c(30,15),c(9,2),c(5,5),c(0,60),c(12,60),c(5,5))

# ASCENT RATES
# Ascent rate 18 m/min below 9 metres, 6m/min above 9 metres
d <- dive(c(30, 12), ascent(18), 9, ascent(6), c(5,3))

# UPLOADED DIVE PROFILE
data(baron)
pro <- baron[, 1:2]
d <- dive(pro)
d <- dive(pro, begin=NA, end=NA)
plot(d)

# GAS USE
# 30-metre dive on Nitrox 32
d <- dive(nitrox(0.32), c(30,20), c(5,5))

# GAS SWITCHING
# Dive to 18 m for 30 min on air,
# switch to Nitrox 36, ascend to 5 metres, safety stop
d <- dive(c(18, 30), nitrox(0.36), c(5,3))
# Same as above, but ascend to 5 m on air, then switch gas
d <- dive(c(18, 30), 5, nitrox(0.36), c(5,3))
```

```
# SWITCHING TO SPECIFIC TANKS
d <- dive(tanklist=list(main=air, deco=nitrox(0.50)),
          tank="main", c(30, 20), 5, tank="deco", c(5,10))

# Descend to 5 metres on pure oxygen, switch to Trimix,
# descend to 30 metres, remain 40 minutes, ascend to 6 metres,
# switch to pure oxygen, ascend to 5 metres, decompress 10 minutes,
# surface and continue breathing pure oxygen for 10 minutes

d <- dive(tanklist=list(travel=trimix(0.15, 0.5), deco=nitrox(1)),
          tank="deco", 5, tank="travel", c(30,40), 6, tank="deco",
          c(5,10), c(0,10))
```

`durations.dive` *Durations of time between each waypoint of a dive*

Description

Extracts, or alters, the durations of each interval between waypoints during a dive.

Usage

```
durations.dive(d)
durations.dive(d) <- value
```

Arguments

`d`	A dive (object of class `"dive"`).
`value`	A numeric vector containing durations in minutes.

Details

An object of class `"dive"` represents a scuba dive. It is created by the function `dive`. A dive is defined as a series of waypoints occurring at specified depths and times.

The duration of the time interval between each successive pair of waypoints is returned by `durations.dive`. The assignment `durations.dive(d) <- value` alters these durations, provided the new vector `value` has the same length as the old one.

Value

`durations.dive` returns a numeric vector containing the durations of intervals between each waypoint, in minutes.

Author(s)

Adrian Baddeley <Adrian.Baddeley@uwa.edu.au> http://www.maths.uwa.edu.au/~adrian/

See Also

depths.dive, times.dive, dive.

Examples

```
d <- dive(c(30,20), c(5,5))
d
durations.dive(d)
# what if we stayed at the bottom for 25 minutes instead of 20?
durations.dive(d)[2] <- 25
d
```

---

ead *Equivalent Air Depth*

Description

Computes the Equivalent Air Depth for a Nitrox mixture at given depths.

Usage

```
ead(depth, g)
eadtable(g, ppO2max=1.4)
```

Arguments

depth	depth of dive, in metres.
g	The breathing gas. An object of class "gas" or a number specifying the fraction (between 0 and 1) of oxygen in the nitrox mixture.
ppO2max	maximum permitted partial pressure of oxygen in ata.

Details

Applies the standard formula for equivalent air depth.

Value

For ead, the Equivalent Air Depth in metres for each value of depth; for eadtable, a data frame containing depths and Equivalent Air Depth values, for a range of depths up to the maximum permitted by ppO2max.

Warnings

This function does not check whether the breathing gas would be safe (it could be hypoxic or toxic at the depth in question).

ead returns negative values for sufficiently shallow depths. In eadtable these values are blank.

Not applicable to altitude dives. Not applicable to gas mixtures other than nitrox (oxygen-nitrogen mixtures).

Author(s)

Adrian Baddeley <Adrian.Baddeley@uwa.edu.au> http://www.maths.uwa.edu.au/~adrian/

See Also

mod, maxmix

Examples

```
# Nitrox I (32% oxygen) at 20 metres
ead(20, 0.32)
# Nitrox I table of EAD's
eadtable(0.32)
# Nitrox II (36% oxygen) at a range of depths
ead(10:25, 0.36)
```

END *Equivalent Air Depth*

Description

Computes the Equivalent Narcotic Depth for a Trimix mixture at given depth.

Usage

```
END(depth, g)
```

Arguments

depth	depth of dive, in metres. A numeric value or a vector of values.
g	The breathing gas. An object of class "gas".

Details

Applies the standard formula for equivalent narcotic depth: the depth at which compressed air would have the same partial pressure of narcotic gases as the specified gas g at the specified depth (assuming that both Nitrogen and Oxygen are narcotic).

Value

Numeric value or vector giving the Equivalent Narcotic Depth in metres for each value of depth.

Warnings

This function does not check whether the breathing gas would be safe (it could be hypoxic or toxic at the depth in question). It is not applicable to altitude dives.

Author(s)

Adrian Baddeley <Adrian.Baddeley@uwa.edu.au> http://www.maths.uwa.edu.au/~adrian/

See Also

ead

Examples

```
# Trimix 18/50
#            (18% oxygen, 50% helium)
END(30, trimix(0.18, 0.5))
```

---

haldane *Tissue Saturation by Haldane Model*

Description

Computes a diver's tissue saturation during and after a dive, as predicted by a Haldane model.

Usage

```
 haldane(d,
model=pickmodel("DSAT"),
prevstate=NULL,
        progressive=FALSE,
        relative=FALSE,
        deco=FALSE,
        derived=FALSE)
```

Arguments

d	The dive profile. An object of class dive.
model	The decompression model. An object of class "hm". Defaults to the DSAT (PADI) model.
prevstate	Optional. Initial state of the diver. A data frame containing the tissue saturations for each tissue compartment in the model, at the start of the dive. Defaults to the state of a diver with no previous dive history.
progressive	Logical flag. If TRUE, the tissue saturations are computed at every time point during the dive. If FALSE (the default), only the final tissue saturation at the end of the dive is computed.
relative	Logical flag indicating whether to compute relative tissue saturations. If FALSE (the default), tissue saturations are expressed as absolute pressures in atmospheres absolute (ata). If TRUE, the tissue saturation for each compartment is expressed as a fraction of the surfacing M-value for the compartment. (Alternatively if deco=TRUE then tissue saturation is expressed as a fraction of the M-value at current depth.)

deco — Logical flag indicating whether to calculate relative saturations for a decompression dive. If deco=FALSE, then relative tissue saturations are computed by dividing the absolute tissue saturation by the surfacing M-value, as appropriate for a no-decompression dive. If deco=TRUE, then relative tissue saturations are computed by dividing the absolute tissue saturation by the M-value at the current depth, as appropriate for a decompression dive. This argument applies only when relative=TRUE.

derived — Logical flag indicating whether to calculate additional quantities such as the decompression ceiling.

Details

This command computes a diver's nitrogen saturation during and after a dive, as predicted by a Haldane model (Boycott et al, 1908).

A Haldane-type decompression model describes the diver's body as a set of independent compartments connected directly to the breathing gas and governed by classical diffusion.

Henry's Law is applied to predict the on- and off-gassing of inert gas in each tissue (compartment) of the model. The resulting differential equations are solved analytically (the solution is often called the 'Schreiner equation' in the decompression literature).

The argument prevstate represents the tissue saturation of the diver at the start of the dive. It should be a data frame, with one row for each compartment of the decompression model, and one column for each inert gas (N2 and/or He) in the model. Such data frames are usually generated by saturated.state or haldane.

If progressive=FALSE, the diver's tissue saturation at the end of the dive is calculated.

If progressive=TRUE, the tissue saturations are calculated at each waypoint during the dive. The corresponding times (extracted by times.dive(d)) are *not equally spaced* over time.

If derived=TRUE then additional quantities are computed including the washout (the difference between the current tissue tension of inert gas and the partial pressure of inert gas in the breathing gas), the decompression ceiling depth (minimum tolerable diving depth) and decompression ceiling pressure (minimum tolerable ambient pressure). These quantities are returned as an attribute of the result: attr(result, "derived"). This is a list containing the components Dceiling (depth ceiling), Pceiling (pressure ceiling) and washout (washout), each of which is a vector, matrix or array of the same format as the result.

To compute the tissue saturation at an arbitrary instant of time during the dive, tim, use haldane(chop.dive(d, 0, tim)). To view the tissue saturation at arbitrary instants of time using interactive graphics, use showstates.

Value

If relative=FALSE and progressive=FALSE, a data frame giving the diver's inert gas saturation state at the end of the dive. Each row of the data frame corresponds to a tissue compartment. The column N2 gives the nitrogen tension (in atmospheres absolute) of each compartment. The column He, if present, gives the Helium tension (in atmospheres absolute) in each compartment.

If relative=FALSE and progressive=TRUE, a three-dimensional array giving the diver's inert gas saturation state at each time point during the dive. The first dimension of the array corresponds to successive time points during the dive (the times can be extracted by times.dive). The second

dimension corresponds to the tissue compartments. The third dimension corresponds to the inert gases (N2 and/or He). The entries are gas tensions (in atmospheres absolute).

If relative=TRUE and progressive=FALSE, a vector giving the diver's relative saturation of inert gas at the end of the dive. Entries in the vector correspond to tissue compartments. The entries are relative gas tensions, that is, the total inert gas (Nitrogen plus Helium) tissue saturation divided by the appropriate M-value for that compartment: either the surfacing M-value (if deco=FALSE) or the M-value at current depth (if deco=TRUE).

If relative=TRUE and progressive=TRUE, a matrix giving the diver's relative saturation at each time point during the dive. Rows of the array correspond to successive time points during the dive. Columns correspond to the tissue compartments. The entries are relative gas tensions, that is, the total inert gas (Nitrogen plus Helium) tissue saturation divided by the appropriate M-value for that compartment: either the surfacing M-value (if deco=FALSE) or the M-value at current depth (if deco=TRUE).

If derived=TRUE then additional quantities are returned as an attribute of the result. This is extracted by: attr(result, "derived"). It is a list containing the components Dceiling (depth ceiling), Pceiling (pressure ceiling) and washout (washout), each of which is a vector, matrix or array of the same format as the result.

Warnings

Not applicable to altitude dives. Not suitable for dive planning.

No constraints of any kind are checked. In particular it is not guaranteed that the model accepts the dive profile as a no-decompression dive.

Author(s)

Adrian Baddeley <Adrian.Baddeley@uwa.edu.au> http://www.maths.uwa.edu.au/~adrian/

References

Bookspan, J. (1995) Diving physiology in plain English. Undersea and Hyperbaric Medicine Society, Kensington, Maryland (USA). ISBN 0-930406-13-3.

Boycott, A.E. Damant, G.C.C. and Haldane, J.B. (1908) The prevention of compressed air illness. *Journal of Hygiene* (London) **8**, 342–443.

Brubakk, A.O. and Neuman, T.S. (eds.) (2003) Bennett and Elliott's Physiology and Medicine of Diving. 5th Edition. Saunders. ISBN 0-7020-2571-2

Buehlmann, A.A. (1983) *Dekompression - Dekompressionskrankheit*. Springer-Verlag.

Buehlmann, A.A., Voellm, E.B. and Nussberger, P. (2002) *Tauchmedizin*. 5e Auflage. Springer-Verlag.

Tikvisis, P. and Gerth, W.A. (2003) Decompression Theory. In Brubakk and Neuman (2003), Chapter 10.1, pages 419-454.

Wienke, B.R. (1994) *Basic diving physics and applications*. Best Publishing Co.

Workman, R.D. (1965) Calculation of decompression schedules for nitrogen-oxygen and helium-oxygen dives. Research Report 6-65. US Navy Experimental Diving Unit. Washington DC.

See Also

dive, oxtox, pickmodel, showstates, Mmix, M0mix.

Examples

```
# First dive to 25 m for 20 min with safety stop
d1 <- dive(c(25,20),c(5,5))
# Evaluate saturation according to DSAT model
s1 <- haldane(d1)
s1
# Look at saturation (in ata)
barplot(s1$N2, ylab="Saturation (ata)")
# Look at relative saturation
M0 <- param(pickmodel("D"), "N2", "M0")
barplot(100 * s1$N2/M0, ylab="Saturation (percent)")

# Evaluate saturation after 2 hour surface interval
s2 <- haldane(dive(c(0,120)), prevstate=s1)
# Then after another dive to 18 m for 30 min with safety stop
s3 <- haldane(dive(c(18, 30),c(5,3)), prevstate=s2)
# Assess effect of breathing 80% oxygen at safety stop
s3o <- haldane(dive(c(18, 30),5, nitrox(0.8), c(5,3)), prevstate=s2)

# Inspect saturation during dive d1 at time 10 minutes
s10 <- haldane(chop.dive(d1, 0, 10))

# Progressive saturation during dive
# A real dive
plot(deepmine, col=1, key.gases="none")
# compute saturations during dive
hmine <- haldane(deepmine, model="Z", progressive=TRUE)
# show N2 saturations during dive
# Image plot
image(x=times.dive(deepmine), y=1:17, z=hmine[,,"N2"],
    xlab="Time (min)", ylab="Compartment", axes=FALSE)
axis(1)
axis(2, at=1:17, labels=dimnames(hmine)[[2]])
# Perspective plot
persp(x=times.dive(deepmine), y=1:17, z=hmine[,,"N2"],
    xlab="Time (min)", ylab="Compartment", zlab="Saturation (atm)",
    col="green3", shade=0.6, border=NA,
    theta=20, phi=30, ltheta=120, lphi=20)

#....  Derived quantities .....
hmine <- haldane(deepmine, model="Z", progressive=TRUE, derived=TRUE)
der <- attr(hmine, "derived")
names(der)

# Decompression ceiling depth (time x compartment x gas)
dcd <- der$Dceiling
# Overall decompression ceiling at each time point
dc <- apply(dcd, 1, max)
```

```
# plot dive with deco ceiling
plot(deepmine, key.gases="none", col=1)
lines(times.dive(deepmine), -dc, lty=3, lwd=2)
legend(100, -60, lty=c(1,3), lwd=2,
    legend=c("dive profile", "deco ceiling"))

# Nitrogen washout for tissue 1b (positive values indicate off-gassing)
plot(times.dive(deepmine), der$washout[,"1b", "N2"],
      type="l", xlab="Time (min)", ylab="Washout")
```

---

hm *Haldane Type Model*

Description

Creates a Haldane-type decompression model.

Usage

```
hm(HalfT, M0=NULL, dM=NULL, ...,
             N2 = list(HalfT=HalfT, M0=M0, dM=dM),
             He = NULL,
             title="user-defined model",
             cnames=NULL,
             mixrule=NULL)
```

Arguments

HalfT	Vector of nitrogen halftimes (in minutes) for each compartment.
M0	Optional vector of surfacing M-values (in ata) of nitrogen for each compartment.
dM	Optional vector of gradients for M-values (dimensionless) for each nitrogen compartment.
...	Ignored.
N2	An alternative way of specifying all the data for Nitrogen. A list with elements labelled "HalfT", "M0" and "dM" giving the halftimes, surfacing M-values, and M-value gradients for each compartment.
He	Data for Helium, if available. A list with elements labelled "HalfT", "M0" and "dM" giving the Helium halftimes, surfacing M-values, and M-value gradients for each compartment.
title	Optional name of model. A character string.
cnames	Optional names of compartments. A vector of character strings.
mixrule	Mixing rule for M-values. Either "N2", "interpolate", or NULL (representing a sensible default).

Details

This function creates an object of class "hm", which represents a Haldane-type decompression model. Objects of this class are needed by the commands ndl, haldane and others.

A Haldane-type decompression model describes the diver's body as a set of independent compartments connected directly to the breathing gas and governed by classical diffusion.

The argument halfT specifies the half-times of the nitrogen compartments, in minutes. The length of halfT implicitly determines the number of nitrogen compartments.

The argument M0, if present, specifies the surfacing M-values for the nitrogen compartments. These are the maximum values of nitrogen tissue tension that are tolerated at the surface. These values are required in order to plan a no-decompression dive.

The argument dM, if present, specifies the rate of increase in nitrogen M-values with pressure. The maximum nitrogen tissue tension tolerated at a pressure P atmospheres is M0 + (P-1) * dM. These values are required in order to plan a decompression dive.

Optionally the model may also allow calculation with Helium diffusion. In that case, the argument He should be a list, with components halfT, M0 and dM, specifying the Helium halftimes, maximum surfacing Helium tensions, and Helium gradients, respectively.

If Helium parameters are included, so that diving with *trimix* (Oxygen/Nitrogen/Helium mixture) is permitted, then the model must also specify a rule for combining the parameters for Helium and Nitrogen to obtain the parameters for any trimix gas. This rule is specified by the argument mixrule. Current options are:

"N2" Ignore the Helium parameters; pretend that Helium is Nitrogen. Combine Nitrogen and Helium into a single inert gas, and take the parameters M0, dM for this gas to be the parameters M0, dM for Nitrogen.

"interpolate" Apply Buehlmann's (1983, 2002) interpolation rule. Convert the parameters M0, dM to the Buehlmann parameters a = M0 - dM and b = 1/dM. For a mixture of Nitrogen and Helium, calculate the a,b values by linear interpolation between the values for Nitrogen and Helium according to the gas fractions. Then convert from a,b back to M0, dM.

The default is mixrule="interpolate" whenever the Helium parameters are specified.

Note that this mixture calculation applies only to the saturation parameters M0, dM which tell us whether a dive is staying within the no-decompression limits. This mixture calculation does not affect the Haldane calculations of the gas tensions in the diver's body.

The class "hm" has methods for print and as.data.frame.

Value

An object of class "hm", representing the decompression model.

Author(s)

Adrian Baddeley <Adrian.Baddeley@uwa.edu.au> http://www.maths.uwa.edu.au/~adrian/

References

Bookspan, J. (1995) Diving physiology in plain English. Undersea and Hyperbaric Medicine Society, Kensington, Maryland (USA). ISBN 0-930406-13-3.

Boycott, A.E. Damant, G.C.C. and Haldane, J.B. (1908) The prevention of compressed air illness. *Journal of Hygiene* (London) **8**, 342–443.

Brubakk, A.O. and Neuman, T.S. (eds.) (2003) Bennett and Elliott's Physiology and Medicine of Diving. 5th Edition. Saunders. ISBN 0-7020-2571-2

Buehlmann, A.A. (1983) *Dekompression - Dekompressionskrankheit*. Springer-Verlag.

Buehlmann, A.A., Voellm, E.B. and Nussberger, P. (2002) *Tauchmedizin*. 5e Auflage. Springer-Verlag.

Tikvisis, P. and Gerth, W.A. (2003) Decompression Theory. In Brubakk and Neuman (2003), Chapter 10.1, pages 419-454.

Wienke, B.R. (1994) *Basic diving physics and applications*. Best Publishing Co.

Workman, R.D. (1965) Calculation of decompression schedules for nitrogen-oxygen and helium-oxygen dives. Research Report 6-65. US Navy Experimental Diving Unit. Washington DC.

See Also

`pickmodel` for some standard models of Haldane type; `ndl`, `haldane`, `showstates`

Examples

```
hm(c(10,20,60), rep(2 * 0.79, 3))
hm(c(10,20,60), rep(2 * 0.79, 3), rep(2 * 0.79, 3))
hm(c(10,20,60), rep(2 * 0.79, 3), rep(2 * 0.79, 3),
   He=list(HalfT=c(5,10,30), M0=c(1,1,1), dM=c(1,1,1)))
```

`is.nitrox` *Recognise a Nitrox Gas*

Description

Determines whether a given gas is nitrox.

Usage

```
is.nitrox(g)
```

Arguments

`g`	An object of class `"gas"`.

Details

An object of class "gas" represents a breathing gas. Such objects are required for various calculations in the scuba library.

The function is.nitrox expects its argument g to be a gas object. It returns TRUE if g is a nitrox gas (a mixture of oxygen and nitrogen), and FALSE otherwise.

Author(s)

Adrian Baddeley <Adrian.Baddeley@uwa.edu.au> http://www.maths.uwa.edu.au/~adrian/

See Also

nitrox

Examples

```
v <- nitrox(0.50)
is.nitrox(v)
is.nitrox(air)
```

---

maxmix *Optimal Nitrox Mixture For Given Depth*

Description

Computes the optimal nitrox mixture for a given maximum depth.

Usage

```
maxmix(depth, ppO2max=1.4)
```

Arguments

depth	the maximum depth, in metres
ppO2max	maximum permitted partial pressure of oxygen in atmospheres absolute

Details

Computes the maximum fraction of oxygen in a nitrox mixture subject to the constraint that the partial pressure of oxygen does not exceed ppO2max atmospheres.

Value

The optimal nitrox mixture. An object of class "gas".

Warnings

Not applicable to altitude dives. Not applicable to gas mixtures other than nitrox (oxygen-nitrogen mixtures).

Author(s)

Adrian Baddeley <Adrian.Baddeley@uwa.edu.au> http://www.maths.uwa.edu.au/~adrian/

See Also

mod, ead, eadtable

Examples

```
# 30 metres
maxmix(30)
```

Mmix *Compute M-values for a Mixture of Inert Gases*

Description

Computes the maximum tolerated inert gas tension at depth (M-value) or at the surface (surfacing M-value, M0) for a given gas.

Usage

```
M0mix(model, fN2, fHe, mixrule=NULL)

Mmix(model, depth, fN2, fHe, mixrule=NULL)
```

Arguments

model	The decompression model. Either an object of class "hm" (see hm) or a character string giving the name of one of the existing models (see pickmodel).
depth	Numeric value or numeric vector giving the depth or depths in metres of seawater.
fN2	Fraction of nitrogen in the breathing gas. Number between 0 and 1. Alternatively this can be a gas object, and the values of fN2, fHe will be extracted from it.
fHe	Fraction of helium in the breathing gas. Number between 0 and 1.
mixrule	Optional string specifying the mixture rule: either "N2" or "interpolate". See Details. If this argument is absent, the mixture rule is taken from the model. If this argument is present, it overrides the mixture rule in model.

Details

M0mix computes the maximum tolerated inert gas tension, in each tissue compartment, for a diver returning to the surface (known as the surfacing M-value M0). This is used for planning no-decompression dives.

Mmix computes the maximum tolerated inert gas tension, in each tissue compartment, for a diver at a specified depth, called the M-value. This is used for planning decompression dives.

A Haldane-type decompression model only specifies these M-values for a single inert gas (i.e. only nitrogen or only helium). When the breathing gas is trimix, the inert gas is a mixture of nitrogen and helium, and the M-values for nitrogen and helium must somehow be combined to obtain M-values relevant to the mixed gas.

The rule for combining the M-values is specified by the argument mixrule. Current options are:

"N2" Ignore the Helium parameters; pretend that Helium is Nitrogen. Combine Nitrogen and Helium into a single inert gas, and take the parameters M0, dM for this gas to be the parameters M0, dM for Nitrogen.

"interpolate" Apply Buehlmann's (1983, 2002) interpolation rule. Convert the parameters M0, dM to the Buehlmann parameters a = M0 - dM and b = 1/dM. For a mixture of Nitrogen and Helium, calculate the a,b values by linear interpolation between the values for Nitrogen and Helium according to the gas fractions. Then convert from a,b back to M0, dM.

If the argument mixrule is missing or NULL, then the default mixture rule is taken from the model (since every Haldane model has a mixture rule, as explained in hm).

Value

A matrix, with one column for each compartment in the model, and one row for each entry of fN2 (and fHe and depth). Entries in the matrix are M-values in atmospheres absolute (ata).

Author(s)

Adrian Baddeley <Adrian.Baddeley@uwa.edu.au> http://www.maths.uwa.edu.au/~adrian/

See Also

hm, pickmodel for specifying models.

haldane for computing tissue saturations.

Examples

```
# Trimix 18/50, surfacing M-values
M0mix("Z", trimix(0.18, 0.5))

# Trimix 18/50, M-values at 0, 10, 20 metres
Mmix("Z", c(0,10,20), trimix(0.18, 0.5))
```

mod *Maximum Operating Depth*

Description

Computes the Maximum Operating Depth for a given gas mixture.

Usage

```
mod(g, ppO2max=1.4)
```

Arguments

g	Breathing gas. An object of class "gas" or a number giving the fraction (between 0 and 1) of oxygen in a nitrox mixture.
ppO2max	maximum permitted partial pressure of oxygen in atmospheres absolute

Details

Computes the maximum depth at which the partial pressure of oxygen does not exceed ppO2max.

Value

The maximum operating depth for this gas.

Warnings

Not applicable to altitude dives. Does not check whether the gas would be hypoxic at the surface.

Author(s)

Adrian Baddeley <Adrian.Baddeley@uwa.edu.au> http://www.maths.uwa.edu.au/~adrian/

See Also

ead, eadtable, maxmix

Examples

```
# Nitrox I (32% oxygen)
mod(0.32)
# Nitrox II (36% oxygen)
mod(nitrox(0.36))
# Trimix 15/50
mod(trimix(0.15, 0.5))
```

ndl *No-Decompression Limit*

Description

Compute the No-Decompression Limit (NDL) for a dive to a specified depth, under a Haldane-type decompression model.

Usage

```
ndl(depth, g=air, prevstate=NULL, model = "DSAT")
```

Arguments

depth	Depth (in metres) for which the NDL should be computed. A single number or a numeric vector.
g	Breathing gas for the dive. An object of class "gas".
prevstate	Optional. Initial state of the diver. A data frame containing the tissue saturations for each tissue compartment in the model, at the start of the dive. Defaults to the state of a diver with no previous dive history.
model	The decompression model. Either an object of class "hm", or a character string giving the name of one of the standard Haldane-type models. Options are "DSAT", "USN", "Workman" and "ZH-L16A".

Details

The No-Decompression Limit (NDL) for a given depth d is the maximum duration of a No-Decompression dive to depth d.

A No-Decompression dive is one which can (theoretically) be aborted at any time without requiring staged decompression. Equivalently, in a No-Decompression Dive, the Nitrogen saturation in each of the diver's tissue compartments never exceeds the maximum Nitrogen saturation tolerated at sea level (known as the surfacing M-value for that compartment).

This algorithm computes the NDL for a dive to the given depth, or for several possible alternative dives to the given depths, based on the specified decompression model. The NDL is determined by the halftimes and the surfacing M-values for all the compartments in the model.

If the breathing gas g contains Helium, then model must include compartments for Helium. Similar calculations apply in this case.

The argument prevstate represents the tissue saturation of the diver at the start of the dive. It should be a data frame, with one row for each compartment of the decompression model, and one column for each inert gas (N2 and/or He). Such data frames are usually generated by saturated.state or haldane.

Value

Numeric vector, of the same length as `depth`, giving the No-Decompression Limit in minutes for each depth. Infinite values are possible.

The value also has an attribute called `"controlling"` which identifies the controlling tissue (the tissue for which the M-value is reached).

Warning

This is not intended for use as a scuba diving table! The NDL calculated here is not the same as the NDL indicated in any published scuba diving table. For example, the NDL calculated by this algorithm using the halftimes and M-values of the DSAT model, may be longer than the NDL published in the DSAT Recreational Dive Planner. Infinite values of NDL may be returned in some cases.

Author(s)

Adrian Baddeley <Adrian.Baddeley@uwa.edu.au> http://www.maths.uwa.edu.au/~adrian/

See Also

`scuba.disclaimer`, `haldane`, `showstates`, `saturated.state`

Examples

```
# NDL for a dive to 15 metres
ndl(15)

# NDL for a dive to 24 metres on EANx 32
ndl(15, g=nitrox(0.32))

# NDL for a dive to 60 metres on Trimix 15/50 under ZH-L16A model
ndl(70, g=trimix(0.15, 0.5), model="Z")

# NDL for a repetitive dive on air to 15 metres, following an
# 18-minute air dive to 30 metres and half-hour surface interval
firstdive <- dive(c(30,18), c(5,3), c(0,30))
ndl(15, prevstate=haldane(firstdive))
```

`nitrox` *Nitrox Mixture*

Description

Create a Nitrox gas mixture.

Usage

```
nitrox(fO2)
```

Arguments

fO2	Fraction (between 0 and 1) of oxygen in the nitrox mixture.

Details

An object of class "gas" represents a breathing gas. Such objects are required for various calculations in the scuba library.

The value returned by nitrox represents Nitrox (oxygen-nitrogen mixture) with the fraction of oxygen specified by the argument fO2.

For the standard Nitrox calculations, see ead and mod. To determine whether a gas is nitrox, use is.nitrox.

There are methods for print and summary for objects of class "gas".

Value

An object of class "gas" representing Nitrox with the specified fraction of oxygen.

Author(s)

Adrian Baddeley <Adrian.Baddeley@uwa.edu.au> http://www.maths.uwa.edu.au/~adrian/

See Also

dive, ead, mod, air, is.nitrox

Examples

```
# Nitrox I (32% oxygen)
nitrox(0.32)
# Nitrox II (36% oxygen)
nitrox(0.36)
#
g <- nitrox(0.50)
is.nitrox(g)
```

---

oxtox *Pulmonary Oxygen Toxicity*

Description

Computes pulmonary oxygen toxicity dose for a given dive profile and breathing gas.

Usage

```
oxtox(d, progressive=FALSE, warn=TRUE)
```

Arguments

d	The dive profile. An object of class "dive".
progressive	Logical flag. If FALSE, the total oxygen toxicity from the dive is calculated. If TRUE, the cumulative oxygen toxicity at each time point during the dive is calculated.
warn	Logical flag indicating whether to issue a warning if the partial pressure of oxygen exceeds a threshold (1.4, 1.5 or 1.6 ata).

Details

Computes the total dose of pulmonary oxygen toxicity from the given dive profile, by

$$\int_0^T \left(\frac{\mathrm{ppO}_2 - 0.5}{0.5} \right)^{0.83} dt$$

The maximum tolerable dose per day is usually reckoned as 1500 OTU. Allowing 650 OTU for recompression therapy implies a working maximum of 850 OTU per day.

Value

If progressive=FALSE, a single numeric value, the total pulmonary oxygen toxicity dose in OTU.

If progressive=TRUE, a numeric vector containing the cumulative pulmonary oxygen toxicity dose in OTU at each time point during the dive. The corresponding time points can be extracted by times.dive.

Warnings

Not applicable to altitude dives.

Author(s)

Adrian Baddeley <Adrian.Baddeley@uwa.edu.au> http://www.maths.uwa.edu.au/~adrian/

References

Bookspan, J. (1995) Diving physiology in plain English. Undersea and Hyperbaric Medicine Society, Kensington, Maryland (USA). ISBN 0-930406-13-3.

Brubakk, A.O. and Neuman, T.S. (eds.) (2003) Bennett and Elliott's Physiology and Medicine of Diving. 5th Edition. Saunders. ISBN 0-7020-2571-2

Buehlmann, A.A., Voellm, E.B. and Nussberger, P. (2002) *Tauchmedizin*. 5e Auflage. Springer-Verlag.

Wienke, B.R. (1994) *Basic diving physics and applications*. Best Publishing Co.

See Also

ppO2, ead, eadtable, mod, maxmix

Examples

```
# Nitrox II (36% oxygen) at 30 metres for 27 minutes
d <- dive(nitrox(0.36), c(30,27))
oxtox(d)

# Same as above, followed by safety stop on 100% oxygen
d <- dive(nitrox(0.36), c(30,27),5, nitrox(1), c(5,5))
oxtox(d)
```

---

param *Extract parameters from Haldane model*

Description

Extracts the tissue compartment parameters from a Haldane type decompression model.

Usage

```
param(model, species = "N2", what = "HalfT")
```

Arguments

model	The decompression model. An object of class "hm".
species	Character string. The inert gas for which parameters are required: "N2" for nitrogen or "He" for helium.
what	Character string. The parameter required: "HalfT" for half-times, "M0" for surfacing M-values, or "dM" for slopes of M-values.

Details

This command extracts the tissue compartment parameters from a Haldane-type decompression model.

The argument model should be an object of class "hm" (created by hm or obtained using pickmodel).

See hm for explanation of the parameters.

Value

A vector containing the values of the specified parameter for each tissue compartment. If the model does not contain this data, NULL is returned.

Author(s)

Adrian Baddeley <Adrian.Baddeley@uwa.edu.au> http://www.maths.uwa.edu.au/~adrian/

See Also

hm, pickmodel

Examples

```
# halftimes for the DSAT model
param(pickmodel("DSAT"), "N2", "HalfT")
```

---

pedro *Real Scuba Dive Profiles*

Description

Dive profiles for 11 different scuba dives.

Format

Each of the datasets pedro902, pedro903, etc, is an object of class "dive".

Details

The command data(pedro) gives access to 11 datasets named pedro902, pedro903, pedro904, pedro922, pedro943, pedro944, pedro945, pedro946, pedro948, pedro949 and pedro950. These are the dive profiles for 11 real scuba dives, uploaded from a dive computer.

Each of the datasets is an object of class "dive" containing the dive profile for one dive. It can be plotted and printed (using plot.dive and print.dive). The nitrogen saturation can be computed using haldane and the cumulative oxygen toxicity using oxtox.

The dives were conducted by Pedro Antonio Neves. Details are as follows:

Dive	*Max depth*	*Gases*	*Type*	*Location*
902	19 m	EAN 32	single	Armacao de Pera, Portugal
903	27 m	EAN 32	single	Farilhoes, Portugal
904	30 m	EAN 32	single	Berlenga, Portugal
922	15 m	EAN 32	multiple	Sines, Portugal
943	21 m	EAN 32	single	Armacao de Pera, Portugal
944	19 m	EAN 32	single	Armacao de Pera, Portugal
945	22 m	EAN 32	single	Armacao de Pera, Portugal
946	20 m	EAN 32	single	Armacao de Pera, Portugal
949	16 m	EAN 32	single	Finisterra,Galicia,Spain
950	40 m	EAN 50, Trimix	single	Finisterra,Galicia,Spain

All dives were conducted on Nitrox 32, except pedro950 which was conducted using Trimix 18/45 and Nitrox 50. All dive profiles consist of a single descent and ascent, except for pedro922 which consists of 5 short bounce dives separated by short surface intervals.

Source

Pedro Antonio Neves

Examples

```
data(pedro)
summary(pedro950)
plot(pedro950)
```

`pickmodel` *Standard Decompression Models*

Description

Selects one of a list of standard Haldane-type decompression models, identified by the name of the model.

Usage

```
pickmodel(model)
```

Arguments

`model`	The name of the model. A character string.

Details

A Haldane-type decompression model describes the diver's body as a set of independent compartments connected directly to the breathing gas and governed by classical diffusion.

This function is a convenient way to select one of the commonly used decompression models of Haldane type.

The argument `model` is partially matched to one of the following:

"Haldane" The original Haldane (1908) model with 5 compartments.

"USN" The first model used by the US Navy, with 6 compartments.

"DSAT" The DSAT model, with 8 compartments, underlying the PADI recreational dive planner tables.

"Workman65" The model of Workman (1965) with 9 compartments.

"ZH-L16A" The 'theoretical' model ZH-L16A of Buehlmann with 17 compartments.

The result is an object (of class `"hm"`) in the format required for use by commands such as `haldane`, `ndl` and `showstates`.

The data for the USN and Haldane models are taken from Wienke (1994), page 127 ff. The DSAT model is taken from Bookspan (1995), pp 16 and 23. The Workman 65 model is taken from Tikvisis and Gerth (2003), Table 10.1.1, page 440. The ZH-L16A parameters were taken from Buehlmann et al (2002) Table 25, page 158.

Value

An object of class `"hm"` representing a Haldane-type decompression model.

Note

If the argument `model` is missing, then the list of available options for `model` is printed.

Author(s)

Adrian Baddeley <Adrian.Baddeley@uwa.edu.au> http://www.maths.uwa.edu.au/~adrian/

References

Bookspan, J. (1995) Diving physiology in plain English. Undersea and Hyperbaric Medicine Society, Kensington, Maryland (USA). ISBN 0-930406-13-3.

Boycott, A.E. Damant, G.C.C. and Haldane, J.B. (1908) The prevention of compressed air illness. *Journal of Hygiene* (London) **8**, 342–443.

Brubakk, A.O. and Neuman, T.S. (eds.) (2003) Bennett and Elliott's Physiology and Medicine of Diving. 5th Edition. Saunders. ISBN 0-7020-2571-2

Buehlmann, A.A. (1983) *Dekompression - Dekompressionskrankheit*. Springer-Verlag.

Buehlmann, A.A., Voellm, E.B. and Nussberger, P. (2002) *Tauchmedizin*. 5e Auflage. Springer-Verlag.

Tikvisis, P. and Gerth, W.A. (2003) Decompression Theory. In Brubakk and Neuman (2003), Chapter 10.1, pages 419-454.

Wienke, B.R. (1994) *Basic diving physics and applications*. Best Publishing Co.

Workman, R.D. (1965) Calculation of decompression schedules for nitrogen-oxygen and helium-oxygen dives. Research Report 6-65. US Navy Experimental Diving Unit. Washington DC.

See Also

`ndl`, `haldane`, `showstates`

Examples

```
pickmodel("Z")
ndl(30, model=pickmodel("Z"))
```

`plot.dive` *Plot a Dive Profile*

Description

Plot a dive profile.

Usage

```
  ## S3 method for class 'dive'
plot(x, ..., main=deparse(substitute(x)),
                        key.gases=c("text", "legend", "none"),
                        text.cex=1,
                        text.verticals=TRUE,
                        col.gases=1:length(tanklist(x)),
                        legendpos=c("top","bottom", "left", "right",
                        "topleft", "topright", "bottomleft", "bottomright",
                        "center"))
```

Arguments

x	The dive profile. An object of class "dive" created by dive.
...	Arguments passed to plot.default.
main	Overall label for plot.
key.gases	String indicating whether the plot should be annotated with labels indicating what gas is breathed, and how to do the annotation.
text.cex	Numeric value controlling the size of text when key.gases="text".
text.verticals	Logical flag indicating whether the plot should indicate what gas is breathed during ascent and descent between stages, as well as during the main stages of the dive.
col.gases	Colours to be used to represent each of the breathing gases used in the dive. Either a numeric vector of colour codes, or a vector of character strings identifying colours.
legendpos	Character string determining the position of the legend, when key.gases="legend". Default is "top".

Details

A representation of the dive profile is plotted.

Value

NULL.

Author(s)

Adrian Baddeley <Adrian.Baddeley@uwa.edu.au> http://www.maths.uwa.edu.au/~adrian/

See Also

dive

Examples

```
# Dive to 25 m for 20 min with safety stop
d1 <- dive(c(25,20),c(5,5))
# plot it
plot.dive(d1)
# Dive to 18 m for 30 min with safety stop, on Nitrox EANx 36
d3 <- dive(nitrox(0.36), c(18,30), c(5,3))
plot.dive(d3)
# Real dive profile
data(baron)
d4 <- dive(nitrox(0.30), baron[, 1:2])
plot(d4, main="Baron Gautsch dive")
```

---

ppO2 *Oxygen Partial Pressures*

Description

Computes the partial pressure of oxygen at each stage during a dive.

Usage

```
ppO2(d)
```

Arguments

d	The dive profile. An object of class "dive".

Details

This function computes the partial pressure of oxygen at each stage of the dive profile. The result is a data frame with two columns: time giving the elapsed time, and ppO2 giving the partial pressure of oxygen (in atmospheres) in the current breathing gas at the current depth.

Value

A data frame with columns time and ppO2.

Author(s)

Adrian Baddeley <Adrian.Baddeley@uwa.edu.au> http://www.maths.uwa.edu.au/~adrian/

See Also

oxtox, ead, mod, maxmix

Examples

```
data(pedro)
plot(ppO2(pedro902), type="l")
```

print.dive *Print a Dive Profile*

Description

Print a dive profile.

Usage

```
  ## S3 method for class 'dive'
print(x, ..., seconds=TRUE)
```

Arguments

x	The dive profile. An object of class "dive" created by dive.
...	Arguments passed to print.default.
seconds	Flag indicating whether to print elapsed time to the nearest second (if TRUE) or to the nearest minute (if FALSE).

Details

A representation of the dive profile is printed.

Value

NULL.

Author(s)

Adrian Baddeley <Adrian.Baddeley@uwa.edu.au> http://www.maths.uwa.edu.au/~adrian/

See Also

dive, plot.dive

Examples

```
# Dive to 25 m for 20 min with safety stop
dive(c(25,20),c(5,5))
# Dive to 18 m for 30 min with safety stop, on Nitrox EANx 36
dive(nitrox(0.36), c(18,30), c(5,3))
```

`saturated.state` *Saturated Tissue State*

Description

Computes the amount of inert gas in each compartment of a diver who has reached equilibrium with the breathing gas at the specified depth. The default is a fresh diver with no diving history.

Usage

```
saturated.state(model="D", depth = 0, g = air)
```

Arguments

`model`	The decompression model. An object of class `"hm"`. Defaults to the DSAT (PADI) model.
`depth`	The diving depth at which saturation is achieved. A single number.
`g`	The breathing gas. An object of class `"gas"`.

Details

A Haldane-type decompression model describes the diver's body as a set of independent compartments connected directly to the breathing gas and governed by classical diffusion.

This command computes the amount of inert gas in each compartment of a diver who has reached equilibrium with the breathing gas at the specified depth.

The default is a fresh diver (in equilibrium with air at the surface at sea level).

Value

A data frame, with one row for each compartment of the model, and one column for each inert gas in the model (`N2` and/or `He`). Entries in the data frame are tissue tensions in ata.

Author(s)

Adrian Baddeley <Adrian.Baddeley@uwa.edu.au> http://www.maths.uwa.edu.au/~adrian/

See Also

`hm`, `haldane`, `nitrox`, `trimix`

Examples

```
fresh <- saturated.state()
deep  <- saturated.state("ZH", 80, trimix(0.15, 0.5))
```

scuba.constants *Constants for Use in Scuba Package*

Description

A list of various useful physical constants needed in calculations in the scuba package.

Format

`scuba.constants` is a list containing the following entries:

`fsw.per.atm`	factor to convert atm (atmospheres absolute) to fsw (feet of seawater)
`fO2air`	the fraction of oxygen in dry air at 25 C
`fN2air`	the fraction of *nitrogen plus argon* in dry air at 25 C

Author(s)

Adrian Baddeley <Adrian.Baddeley@uwa.edu.au> http://www.maths.uwa.edu.au/~adrian/

scuba.disclaimer *Disclaimer for Scuba Library*

Description

This page contains important warnings.

Details

The scuba software library is intended for use in research and education about the mathematical and statistical basis of decompression theory. It is not designed for actual use in scuba diving and related activities. It is emphatically not suitable for use in actual diving.

Scuba diving is a dangerous activity with inherent risks of death and serious injury. No-one should attempt scuba diving without professional training, certification, supervision and regular medical assessment.

It is also dangerous for trained scuba divers to exceed the limitations of their training. Diving at altitudes above sea level, and breathing mixed gases other than air, carry increased risk and additional types of risk. Divers should seek additional, professional training and certification for such activities.

This software is not suitable for use in actual scuba diving. The software will yield numerical results for any diving activity, without giving any warning if the activity would be dangerous or fatal. Each function in the scuba library calculates the predictions of one theoretical model (a law of physics, a decompression model or another empirical relationship). In doing so, it does not take account of safety restrictions, other physical laws, or other important information.

The software is provided for academic interest only. It should not be used to generate diving tables or protocols related to diving. No output from this software should be misconstrued as a diving table. Only persons qualified to supervise diving activities or qualified in hyperbaric medicine should attempt to design diving tables. Although existing published diving tables are based on theoretical models, such tables have been extensively field-tested and modified before approval. Existing tables are more conservative than the models from which they were originally derived.

The author does not warrant that the software is correct in any sense whatsoever. Even if correctly computed, the predictions of a theoretical physical model may not be correct predictions.

Author(s)

Adrian Baddeley <Adrian.Baddeley@uwa.edu.au> http://www.maths.uwa.edu.au/~adrian/

---

showstates *Interactive Display of Diver Saturation*

Description

Plots a dive profile and interactively displays the diver's nitrogen saturation state at any intermediate stage indicated by the mouse.

Usage

```
showstates(d, model="DSAT", relative=TRUE, deco=FALSE, ...)
```

Arguments

d	The dive. An object of class "dive".
model	The decompression model. Either an object of class "hm" or a character string identifying a model.
relative	Logical flag indicating whether to compute relative tissue saturations. If FALSE, tissue saturations are expressed as absolute pressures in atmospheres absolute (ata). If TRUE (the default), the tissue saturation for each compartment is expressed as a fraction of the surfacing M-value for the compartment. (Alternatively if deco=TRUE then tissue saturation is expressed as a fraction of the M-value at current depth.)
deco	Logical flag indicating whether to calculate relative saturations for a decompression dive. If deco=FALSE, then relative tissue saturations are computed by dividing the absolute tissue saturation by the surfacing M-value, as appropriate for a no-decompression dive. If deco=TRUE, then relative tissue saturations are computed by dividing the absolute tissue saturation by the M-value at the current depth, as appropriate for a decompression dive. This argument applies only when relative=TRUE.
...	Arguments passed to plot.dive to control the plot of the dive profile.

Details

An object of class "dive" represents a scuba dive. It is created by the command dive.

This function plots the dive using plot.dive then waits for the user to click on the plot. This click selects a time during the dive. The algorithm computes the nitrogen tensions in the tissues at the indicated time, using haldane, and plots them as a bar graph. The cumulative oxygen toxicity is also computed using oxtox.

Note that (by default) the bar graph shows the *relative* nitrogen tensions in each compartment, that is, the tissue nitrogen tension divided by the "surfacing M-value" (tissue maximum nitrogen tension for a no-decompression dive at sea level).

The argument model determines the decompression model. It should be either an object of class "hm" (created by hm) or a character string matching one of the options in pickmodel.

Value

The vector of absolute nitrogen tensions (in ata) in the most recently clicked stage.

Author(s)

Adrian Baddeley <Adrian.Baddeley@uwa.edu.au> http://www.maths.uwa.edu.au/~adrian/

See Also

dive, haldane, oxtox

Examples

```
## Not run:
  showstates(dive(c(30,20), c(5,3)), "ZH")

## End(Not run)
```

---

tanklist *Extract or Change the Breathing Gas Tanks in a Dive*

Description

Extracts or modifies the list of breathing gases in a dive object.

Usage

```
tanklist(d)
tanklist(d) <- value
```

Arguments

d	The dive (an object of class "dive").
value	The new tank list. A list, whose entries are gases (objects of class "gas")

Details

An object of class "dive" represents a scuba dive, including information about depth, time and gas breathed at each stage of the dive. These objects are created by the function dive.

The tank list of a dive object is a list of the tanks of breathing gas that were used (or available to be used) during the dive. The function tanklist returns this list.

If a new value is assigned to the tank list of a dive d, then d is changed. The new dive d is conducted to the same depths and times as the old d, but with different gases.

Value

The value of tanklist(d) is a list whose elements are gases (objects of class "gas").

Author(s)

Adrian Baddeley <Adrian.Baddeley@uwa.edu.au> http://www.maths.uwa.edu.au/~adrian/

See Also

dive, nitrox

Examples

```
d <- dive(air, c(30,4), 5, nitrox(0.5), c(5,10))
d
tanklist(d)
tanklist(d) <- list(air, nitrox(0.8))
d
tanklist(d) <- list(travel=air, deco=nitrox(0.8))
d
```

---

times.dive *Elapsed times at each waypoint of a dive*

Description

Extracts, or alters, the elapsed time at each waypoint during a dive.

Usage

```
times.dive(d)
times.dive(d) <- value
```

Arguments

d	A dive (object of class "dive").
value	A numeric vector containing elapsed times in minutes.

Details

An object of class "dive" represents a scuba dive. It is created by the function dive. A dive is defined as a series of waypoints occurring at specified depths and times. The elapsed time at each waypoint is returned by times.dive. The assignment times.dive(d) <- value alters these elapsed times, provided the new vector value has the same length as the old one.

Value

times.dive returns a numeric vector containing the elapsed times at each waypoint, in minutes.

Author(s)

Adrian Baddeley <Adrian.Baddeley@uwa.edu.au> http://www.maths.uwa.edu.au/~adrian/

See Also

depths.dive, durations.dive, dive.

Examples

```
d <- dive(c(30,20), c(5,5))
d
times.dive(d)
# stretch time by 10 percent
times.dive(d) <- 1.1 * times.dive(d)
d
```

---

trimix *Trimix Gas*

Description

Create a Trimix gas.

Usage

```
trimix(fO2, fHe, fN2)
```

Arguments

fO2, fHe, fN2	Fractions (between 0 and 1) of oxygen, helium and nitrogen, respectively, in the trimix gas.

Details

An object of class "gas" represents a breathing gas. Such objects are required for various calculations in the scuba library.

The value returned by trimix represents Trimix (oxygen-nitrogen-helium mixture) with the fractions of oxygen, nitrogen and helium specified by the arguments fO2, fN2, fHe respectively. These fractions should sum to 1.

If only two of the gas fractions are given, the missing fraction will be calculated so that the three values sum to 1.

There are methods for print and summary for objects of class "gas".

Value

An object of class "gas" representing Trimix with the specified composition.

Author(s)

Adrian Baddeley <Adrian.Baddeley@uwa.edu.au> http://www.maths.uwa.edu.au/~adrian/

See Also

nitrox, air

Examples

```
# Trimix 18/50
#            (18% oxygen, 50% helium)
trimix(0.18, 0.5)
```

---

whichtank *Which Tanks are Used during a Dive*

Description

Determine which tank of breathing gas is used at each time point during a dive.

Usage

```
whichtank(d)
whichtank(d) <- value
```

Arguments

d	The dive (an object of class "dive").
value	Vector of integers or character strings identifying which tank of breathing gas is used at each time point.

Details

An object of class "dive" represents a scuba dive, including information about depth, time and gas breathed at each stage of the dive. These objects are created by the function dive.

A dive object has a *tank list* which is a list of the tanks of breathing gas that were used (or were available to be used) during the dive. The function tanklist returns this list.

The selection of tanks, i.e. which tank is actually used at each stage of the dive, is specified by whichtank. The command whichtank(d) returns a vector of integers or character strings, identifying which tank in the tank list is in use at each waypoint during the dive. That is, whichtank(d)[i] is the tank in use at the ith waypoint during the dive.

The command whichtank(d) <- value will change the selection of tanks used at each stage during the dive. Here value should be a vector of integers or character strings identifying tanks in the tank list, and the length of value should be equal to the length of the vectors depths.dive(d) and depths.dive(d).

A common use of whichtank is to specify that a particular gas should be used only in a particular range of depths. This is done by applying ifelse to depths.dive as shown in the Examples.

Value

The value of whichtank(d) is a vector of integers or character strings.

Author(s)

Adrian Baddeley <Adrian.Baddeley@uwa.edu.au> http://www.maths.uwa.edu.au/~adrian/

See Also

dive, tanklist

Examples

```
# tanks are numbered
d <- dive(air, c(30,40), 6, nitrox(0.5), c(6,3), c(3,3))
d
tanklist(d)
whichtank(d)
# change choice of tank at 6 metre deco stop
# The Hard Way:
whichtank(d) <- c(1,1,1,1,1,2,2,2)
d

# The Snazzy Way:
# if shallower than 3 metres, then tank 2, else tank 1
whichtank(d) <- ifelse(depths.dive(d) <= 3, 2, 1)
d

# tanks with names
dd <- dive(tanklist=list(travel=trimix(0.18, 0.5), deco=nitrox(0.8)),
          tank="travel", c(30,40), c(20, 10), 9, tank="deco",
          c(9,10), c(6,5), c(3,5))
```

```
dd
# if shallower than 6 metres, then deco gas, else travel gas
whichtank(dd) <- ifelse(depths.dive(dd) <= 6, "deco", "travel")
dd
```

`Workman65` *Decompression model of Workman 1965*

Description

The parameters of the decompression model due to Workman, 1965.

Format

A data frame giving the following parameters for each compartment:

`halftime`	Nitrogen halftime (minutes)
`M0`	surfacing M-value parameter M_0 in msw (metres of seawater)
`A`	slope parameter A (dimensionless)

Details

The decompression model developed for the US Navy by Workman (1965) is a pure diffusion (Haldane-type) model consisting of 9 tissue compartments. Nitrogen in the breathing gas diffuses in and out of each compartment at a rate governed by the halftime for that compartment. The maximum tolerable nitrogen tension in a compartment, when the diver is at depth D msw (metres of sea water), is $M = M_0 + A \times D$ msw, where M_0 and A are values intrinsic to the tissue. The value M_0 is called the 'surfacing M-value' since it is the maximum nitrogen tension that is allowed in each compartment during a no-decompression dive at sea level.

Author(s)

Adrian Baddeley <Adrian.Baddeley@uwa.edu.au> http://www.maths.uwa.edu.au/~adrian/

Source

Tikvisis and Gerth (2003), Table 10.1.1, page 440.

References

Brubakk, A.O. and Neuman, T.S. (eds.) (2003) Bennett and Elliott's Physiology and Medicine of Diving. 5th Edition. Saunders. ISBN 0-7020-2571-2

Tikvisis, P. and Gerth, W.A. (2003) Decompression Theory. In Brubakk and Neuman (2003), Chapter 10.1, pages 419-454.

Workman, R.D. (1965) Calculation of decompression schedules for nitrogen-oxygen and helium-oxygen dives. Research Report 6-65. US Navy Experimental Diving Unit. Washington DC.

See Also

BuehlmannL16A

Index