Package 'cardidates'

June 3, 2018

Type Package

Title Identification of Cardinal Dates in Ecological Time Series

Version 0.4.8

Depends R (>= 3.2), boot, pastecs, lattice

Imports graphics, grDevices, stats, methods, utils

Encoding latin1

Author Susanne Rolinski [aut],
René Sachse [aut],
Thomas Petzoldt [aut, cre]

Maintainer Thomas Petzoldt <thomas.petzoldt@tu-dresden.de>

Description Identification of cardinal dates
(begin, time of maximum, end of mass developments)
in ecological time series using fitted Weibull functions.

License GPL (>= 2)

URL http://cardidates.r-forge.r-project.org

LazyData yes

NeedsCompilation no

Repository CRAN

Date/Publication 2018-06-03 19:02:53 UTC

R topics documented:

`cardidates-package` *Identification of Cardinal Dates in Ecological Time Series*

Description

Identification of cardinal dates (begin, time of maximum, end of mass developments) in ecological time series using fitted Weibull functions.

Details

Phenology and seasonal succession in aquatic ecosystems are strongly dependent on physical factors. In order to promote investigations into this coupling, objective and relible methods of characterising annual time series are important.

The proposed methods were developed within the AQUASHIFT research program and used to determine the beginning, maximum and end of the spring mass development of phytoplankton in different lakes and water reservoirs. These time points, which we call "cardinal dates", can be analysed for temporal trends and relationships to climate variables.

The complete methodology is described in Rolinski et. al (2007). Until now we implemented only the most reliable approach using Weibull-Functions (Method B in the article), other algorithms may follow when required.

The methodology may also be useful for other ecological time series (e.g. small mammals or insects). Please don't hesitate to contact the package maintainer if you feel that this package should be generalized to other processes.

Author(s)

Susanne Rolinski (original algorithm), Thomas Petzoldt and René Sachse (package and documentation).

Maintainer: Thomas Petzoldt <thomas.petzoldt@tu-dresden.de>

References

Rolinski, S., Horn, H., Petzoldt, T. & Paul, L. (2007): Identification of cardinal dates in phytoplankton time series to enable the analysis of long-term trends. Oecologia **153**, 997 - 1008. doi: 10.1007/s0044200707832.

Wagner, A., Hülsmann, S., Paul, L., Paul, R. J., Petzoldt, T., Sachse, R., Schiller, T., Zeis, B., Benndorf, J. & Berendonk, T. U. (2012): A phenomenological approach shows a high coherence of warming patterns in dimictic aquatic systems across latitude- Marine Biology **159**(11), 2543-2559. doi: 10.1007/s0022701219345.

See Also

weibull4, weibull6, fitweibull, peakwindow, CDW, metaCDW

Examples

```
########## quick start for the impatient ###############
## create some test data
set.seed(123)
x <- seq(0, 360, length = 20)
y <- abs(rnorm(20, mean = 1, sd = 0.1))
y[5:10] <- c(2, 4, 7, 3, 4, 2)

## fit Weibull function with 6 free parameters
res <- fitweibull6(x, y)
plot(res)
summary(res)

################## more details ########################

## show some properties
res$r2
p <- res$p
o <- res$fit
f <- res$ymax

## identify cardinal dates from fitted curves
(smd  <- CDW(p))
(smda <- CDW(p, symmetric = FALSE))

## plot data, curve and cardinal dates
plot(x, y, ylim = c(0, 10), xlim = c(0, 365))
lines(o$x, o$f * f)
points(x, fweibull6(x, p) * f, col = "green")
points(smd$x, smd$y * f, col = "orange", pch = 16)

## or, alternatively:
points(smda$x, fweibull6(smda$x, p) * f, col = "red", pch = 1, cex = 1.2)

## for comparison: fit of a 4 parameter Weibull
res4  <- fitweibull4(x, y)
res4$r2
p <- res4$p
o <- res4$fit
f <- res4$ymax
smd  <- CDW(p)
lines(o$x, o$f * f, col = "blue")
points(smd$x, fweibull4(smd$x, p) * f, col = "blue", pch = 16)
```

carditest *Artificial Data Set: 3 Years of Phytoplankton*

Description

The data contains 3 years of an artificial phytoplankton data set which conforms to the metaCDW data structure.

Usage

```
data(carditest)
```

Format

A data frame with the following 4 columns:

`sample` time code

`x` day of year (interval 0 . . . 365)

`y` observed biovolume, abundance etc.

`flag` validity flag to switch single data records on (`TRUE`) or off (`FALSE`), defaults to `TRUE`

See Also

metaCDW

Examples

```
data(carditest)
head(carditest)
#View(carditest)
```

CDW *Cardinal Dates Using Fitted Weibull Curves*

Description

CDW (cardinal dates using Weibull curves) extracts "cardinal dates" from fitted four- and six-parametric Weibull curves.

Usage

```
CDW(p, xmin = 0, xmax = 365, quantile = 0.05, symmetric = FALSE)

CDWa(p, xmin = 0, xmax = 365, quantile = 0.05, symmetric = FALSE)
```

Arguments

p	object of class cardiFit returned by fitweibull6 or fitweibull4 or parameter vector of the fitted Weibull function,
xmin	left boundary (in day of year) of the integral under the curve,
xmax	right boundary (in day of year) of the integral under the curve,
quantile	two-sided quantile (percentage of integral) which defines beginning and end of the peak,
symmetric	if (TRUE), quantiles are calculated for the whole area under the curve, otherwise for each of the branches separately.

Details

CDW is a numerically improved version of the algorithm described in Rolinski et al. (2007). Version CDWa is an alternative, simplified version which sets the baseline before and after the peak to zero using appropriate offset parameters p[1] and p[4]. The original method described by Rolinski et al. 2007 (here called CDW) shifts the function for the left and right branch separately in the asymmetric case.

Value

A list with components:

x	x values of cardinal dates tMid, tBegin, tEnd,
y	the corresponding y values (divided by ymax),
p	parameters of the fitted Weibull function

See Also

weibull4, weibull6, fitweibull, peakwindow,

metaCDW, cardidates

Examples

```
## create some test data
set.seed(123)
x <- seq(0, 360, length = 20)
y <- abs(rnorm(20, mean = 1, sd = 0.1))
y[5:10] <- c(2, 4, 7, 3, 4, 2)

## fit Weibull function with 6 free parameters
res <- fitweibull6(x, y)

## show some properties
res$r2
p <- res$p
o <- res$fit
f <- res$ymax
```

```
## identify cardinal dates from fitted curves
(smd  <- CDW(p))
(smda <- CDW(p, symmetric = FALSE))

## plot data, curve and cardinal dates
plot(x, y, ylim=c(0, 10), xlim = c(0, 365))
lines(o$x, o$f * f)
points(x, fweibull6(x, p) * f, col = "green")
points(smd$x, fweibull6(smd$x, p) * f, col = "orange", pch = 16)
points(smda$x, fweibull6(smda$x, p) * f, col = "red", pch = 1, cex = 1.2)

## for comparison: additional fit of a 4 parameter Weibull
res4  <- fitweibull4(x, y)
res4$r2
p <- res4$p
o <- res4$fit
f <- res4$ymax
smd  <- CDW(p)
lines(o$x, o$f * f, col = "blue")
points(smd$x, fweibull4(smd$x, p) * f, col = "blue", pch = 16)
```

`fitweibull` *Fit Four or Six Parametric Weibull Functions*

Description

Fit a four- or six-parametric Weibull function to environmental data.

Usage

```
fitweibull6(x, y = NULL, p0 = NULL, linint = -1, maxit = 2000)

fitweibull4(x, y = NULL, p0 = c(0.1, 50, 5, 100), linint = -1, maxit = 1000)
```

Arguments

`x, y`	the x (in day of year) and y coordinates of a set of points. Alternatively, a single argument x can be provided.
`p0`	initial parameters for optimization. In case of `p0 = NA` a heuristic algorithm to derive initial values is used for `fitweibull6`.
`linint`	control parameter to select interpolation behavior. Negative values (default) specify automatic selection heuristic, zero disables interpolation. A positive value is interpreted as mandatory interpolation time step.
`maxit`	maximum number of iterations passed to the optimisation functions.

Details

Function `fitweibull6` uses extensive heuristics to derive initial parameters for the optimization. It is intended to work with data which are defined over an interval between 0 and 365, e.g. environmental data and especially for plankton blooms. Please note that the function does internal transformation:

$$y_{rel} = y_i / y_{max}$$

Note that additional data points are inserted between original measurements by linear interpolation with time step = 1 before curve fitting if the number of original data points is too low (currently n < 35). You can set `linint = 0` to switch interpolation off.

`fitweibull4` has only built-in heuristics for data interpolation but not for guessing initial parameters which must be supplied as vector `p0` in the call.

Value

A list with components:

`p`	vector of fitted parameters,
`ymax`	maximum y value used for transformation,
`r2`	coefficient of determination between transformed and fitted y values,
`fit`	data frame with the following columns:

- `x`~data (original or interpolated) used for curve fitting,
- `y`~data (transformed-original or interpolated) used for curve fitting,
- `f`~estimated function values.

Note

Note that the heuristics works optimal if unnecessary leading and trailing data are removed before the call.

Author(s)

Susanne Rolinski (original algorithm) and Thomas Petzoldt (package).

Maintainer: Thomas Petzoldt <thomas.petzoldt@tu-dresden.de>

References

Rolinski, S., Horn, H., Petzoldt, T., & Paul, L. (2007): Identification of cardinal dates in phytoplankton time series to enable the analysis of long-term trends. Oecologia **153**, 997 - 1008. http://dx.doi.org/10.1007/s00442-007-0783-2.

See Also

`weibull4`, `weibull6`,

`CDW` `peakwindow`, `cardidates`

Examples

```
## create some test data
set.seed(123)
x <- seq(0, 360, length = 20)
y <- abs(rnorm(20, mean = 1, sd = 0.1))
y[5:10] <- c(2, 4, 7, 3, 4, 2)

## fit Weibull function with 6 free parameters
res <- fitweibull6(x, y)

## show some properties
res$r2
p <- res$p
o <- res$fit
f <- res$ymax

## fit 6 parameter Weibull with user-provided start parameters

x <- seq(0, 150)
y <- fweibull6(x, c(0.8, 40, 5, 0.2, 80, 5)) + rnorm(x, sd = 0.1)
plot(x, y)
res <- fitweibull6(x, y, p0 = c(0, 40, 1, 1, 60, 0))
plot(res)
```

---

merge.cardiMetacdw *Merge Two Objects of Class 'cardiMetacdw'*

Description

This is a helper function that can be used to replace or add manually fitted cardidates objects to a set of cardidates objects fitted by metaCDW.

Usage

```
## S3 method for class 'cardiMetacdw'
merge(x, y, ...)
```

Arguments

x	the result of a call to metaCDW,
y	a second object resulting from a call to metaCDW, but which contains one sample only.
...	not implemented, reserved for future extensions.

Details

This is a helper function that can be used to add additional fits to an existing object or to replace single fits of a `metaCSW` object. Though the function is intended for fine-tuning of problematic samples it should not be abused for invalidating the general principles of objectivity and reproducibility.

Value

An object of class 'cardiMetacdw', i.e. a Weibull fit.

See Also

metaCDW

Examples

```
## artificial test data
data(carditest)

## identify all first peaks
fit <- metaCDW(carditest)

## plot it
plot(fit, carditest)

## detect second peak for 'Year2'
sample2 <- subset(carditest, sample == "Year 2")
sample2$sample <- factor(sample2$sample) # drop unused levels
fit2 <- metaCDW(sample2, xstart=150)

## merge results
merged.fit <- merge(fit, fit2)
plot(merged.fit, carditest)
```

---

metaCDW *Extract Cardinal Dates of Multiple Time Series at Once Using Fitted Weibull Curves*

Description

metaCDW determines the relevant peak of several time series and fits four- resp. six-parametric Weibull curves to these peaks of all series at once and extracts "cardinal dates" from the fitted curves.

Usage

```
metaCDW(dat, method = "weibull6", xstart = 55, xmin = 0, xmax = 365,
        minpeak = 0.1, mincut = 0.382,
        quantile = 0.05, symmetric = FALSE, p0 = NULL, linint = -1,
        findpeak = TRUE, maxit = 2000)
## S3 method for class 'cardiMetacdw'
summary(object, file="", ...)
```

Arguments

dat	a data.frame containig the columns: sample (description of the timeseries such as year, water body, or any other code (should be numeric or factor), x (the independend variable, e.g. Julian day of the year), y (the dependend variable, e.g. phytoplankton biovolume) and an optional column flag (of type boolean indicating whether a data point should be included in analysis),
method	either "weibull6" or "weibull4",
xstart	offset (day of year) for the "spring" peak; either a single numeric value for all years or a vector of the same length as number of samples,
xmin	left boundary (in day of year) of the integral under the curve,
xmax	right boundary (in day of year) of the integral under the curve,
quantile	two-sided quantile (percentage of integral) which defines beginning and end of the peak,
minpeak	minimum value of the total maximum which is regarded as peak (default value is derived from golden section),
mincut	minimum relative height of a pit compared to the lower of the two neighbouring maxima at which these maxima are regarded as separate peaks.
symmetric	if (TRUE), quantiles are calculated for the whole area under the curve, otherwise for each of the branches separately.
p0	initial parameters for optimization. In case of p0 = NULL a heuristic algorithm to derive initial values is used for fitweibull6,
linint	control parameter to select interpolation behavior. Negative values (default) specify automatic selection heuristic, zero disables interpolation. A positive value is interpreted as mandatory interpolation time step.
maxit	maximum number of iterations passed to the optimisation functions,
findpeak	a logical value indicating whether the relevant peaks of the time series should be identified automatically with peakwindow before the computation proceeds,
object	a result from a call to metaCDW,
file	file name where the data are to be written to, defaults to screen,
...	other parameters of summary passed to write.

Details

This is a top-level function which calls peakwindow, fitweibull and CDW for a series of data sets and returns a table (data frame) of all results.

Value

A list with components:

metares	data frame with cardinal dates and fitted parameters, see CDW for details,
weibullfits	list of fit details for all fits, see fitweibull for details.

See Also

weibull4, weibull6, fitweibull, peakwindow, CDW,

cardidates

Examples

```
## open test data set (3 years) with 4 columns
## sample, x, y, flag
data(carditest)
dat <- carditest

## alternatively: import data from spreadsheet via the clipboard
# dat <- read.table("clipboard", sep = "\t", header = TRUE)

## or, for languages with comma as decimal separator:
# dat <- read.table("clipboard", sep = "\t", header = TRUE, dec = ",")

## Note: as.numeric recodes factor year to numeric value
plot(as.numeric(dat$sample)*365 + dat$x, dat$y, type = "b")

## do the analysis
tt <- metaCDW(dat, xstart = 55)

## plot results
par(mfrow=c(1, 3))
lapply(tt$weibullfits, plot)

## return table of results
summary(tt)

## Not run:
## copy to clipboard in spreadsheet compatible format
summary(tt, file = "clipboard", sep = "\t", quote = FALSE, row.names = FALSE)

## or, for languages with comma as decimal separator:
#summary(tt, file = "clipboard", sep = "\t", dec = ",",
#  quote = FALSE, row.names = FALSE)

## End(Not run)
```

---

peakwindow *Identify Peaks in Time Series*

Description

This function identifies peaks in time series and helps to identify the time window of the first maximum according to given rules.

Usage

```
peakwindow(x, y = NULL, xstart = 0, xmax = max(x), minpeak = 0.1, mincut = 0.382)
```

Arguments

x, y	the x (in day of year) and y coordinates of a set of points. Alternatively, a single argument x can be provided.
xstart	x value (e.g. time of ice-out) before the maximum value of the searched peak (this is a "weak" limit),
xmax	maximum of the end of the searched peak (this is a "hard" maximum,
minpeak	minimum value of the total maximum which is regarded as peak,
mincut	minimum relative height of a pit compared to the lower of the two neighbouring maxima at which these maxima are regarded as separate peaks (default value is derived from golden section).

Details

This is a heuristic peak detection algorithm. It can be used for two related purposes, (i) to identify all relevant peaks within a time-series and (ii) to identify the time window which belongs to one single peak (smd = specified mass development, e.g. spring maximum in phytoplankton time series).

Value

A list with the following elements:

peaks	a data frame with the characteristics (index, xleft, x, xright and y) of all identified peaks,
data	the original data set (x, y),
smd.max.index	index of the maximum value of the "specified" peak,
smd.max.x	x-value of the maximum of the "specified" peak,
smd.indices	indices (data window) of all data belonging to the "specified" peak,
smd.x	x-values (time window) of all data belonging to the "specified" peak,
smd.y	corresponding y-values of all data belonging to the "specified" peak,
peakid	vector with peak-id-numbers for all data.

See Also

weibull4, weibull6, fitweibull, CDW plot.cardiPeakwindow cardidates

Examples

```
## generate test data with 3 peaks
set.seed(123)
x <- seq(0, 360, length = 20)
y <- abs(rnorm(20, mean = 1, sd = 0.1))
y[5:10] <- c(2, 4, 7, 3, 4, 2)
```

```
y <- c(y, 0.8 * y, 1.2 * y)
x <- seq(0, 360, along = y)
y[6] <- y[7]   # test case with 2 neighbouring equal points

## plot the test data
plot(x, y, type="b")

## identify the "spring mass development"
peaks <- peakwindow(x, y)
ind <- peaks$smd.indices
lines(x[ind], y[ind], col="red", lwd=2)

## now fit the cardinal dates
fit <- fitweibull6(peaks$smd.x, peaks$smd.y)
CDW(fit)
plot(fit)

## some more options ...
peaks <- peakwindow(x, y, xstart=150, mincut = 0.455)
ind <- peaks$smd.indices
lines(x[ind], y[ind], col = "blue")
points(x, y, col = peaks$peakid +1, pch = 16) # all peaks

## work with indices only
peaks <- peakwindow(y)

## test case with disturbed sinus
x<- 1:100
y <- sin(x/5) +1.5 + rnorm(x, sd = 0.2)
peaks <- peakwindow(x, y)
plot(x, y, type = "l", ylim = c(0, 3))
points(x, y, col = peaks$peakid + 2, pch = 16)

## test case: only one peak
yy <- c(1:10, 11:1)
peakwindow(yy)

## error handling test case: no turnpoints
# yy <- rep(1, length(x))
# peakwindow(x, yy)
```

`plot.cardiFit` *Plot Method for cardiFit Objects*

Description

This function is a top-level function to visualize cardinal date objects fitted with `fitweibull4` and `fitweibull6`.

Usage

```
## S3 method for class 'cardiFit'
plot(x, y = NULL, xmin = 0, xmax = 365, quantile = 0.05, symmetric = FALSE, ...)
```

Arguments

x	object of class cardiFit resulting from a call to fitweibull6 or fitweibull4,
y	not used, for compatibility with plot only,
xmin	left boundary (in day of year) of the integral under the curve,
xmax	right boundary (in day of year) of the integral under the curve,
quantile	two-sided quantile (percentage of integral) which defines beginning and end of the peak,
symmetric	if (TRUE), quantiles are calculated for the whole area under the curve, otherwise for each of the branches separately,
...	other arguments passed to plot.default.

Details

See CDW for a detailed description of parameters.

See Also

weibull4, weibull6, fitweibull, peakwindow,

cardidates

Examples

```
## create some test data
set.seed(123)
x <- seq(0, 360, length = 20)
y <- abs(rnorm(20, mean = 1, sd = 0.1))
y[5:10] <- c(2, 4, 7, 3, 4, 2)

## fit Weibull function with 6 free parameters
res <- fitweibull6(x, y)

## see the results
plot(res)
summary(res)
```

plot.cardiMetacdw *Plot Method for cardiPeakwindow Objects*

Description

This function is intended to visualize peaks identified by the metaCDW function.

Usage

```
## S3 method for class 'cardiMetacdw'
plot(x, y, type = "lattice", scale = TRUE, col.poly = "black", ...)
```

Arguments

x	an object of class cardiMetacdw resulting from a call to CDW,
y	the original data which were supplied to CDW,
type	either "lattice" or "polygon",
scale	a logical value indicating whether the height of polygons should be scaled relative to the highest peak,
col.poly	color of polygons, either a single numeric value for all years or a vector of the same length as number of samples,
...	other arguments passed to xyplot or plot.

Details

This is a top-level function to plot a complete set of fits resurned by metaCDW together with the original data points. The function requires the **lattice** package.

See Also

peakwindow, CDW metaCDW cardidates

Examples

```
## artificial test data
data(carditest)

## identify all peaks
tt <- metaCDW(carditest)

## plot it;
plot(tt, carditest)

## or with alternate layout:
plot(tt, carditest, layout = c(1, 3))

## plot polygons
plot(tt, carditest, type = "polygon")
```

`plot.cardiPeakwindow` *Plot Method for cardiPeakwindow Objects*

Description

This function is intended to visualize peaks identified by the peakwindow function.

Usage

```
## S3 method for class 'cardiPeakwindow'
plot(x, y, add=FALSE, ...)
```

Arguments

x	an object of class `cardiPeakwindow` resulting from a call to `peakwindow`,
y	not used, for compatibility with plot only,
add	if TRUE, highlight additional peaks in the plot,
...	other arguments passed to `plot.default`.

Details

Peaks identified by `peakwindow` are labelled with their coresponding peak numbers. The first peak which occurs after `xstart` (e.g. spring peak in biological time series) is highlighted with red color.

See Also

`weibull4`, `weibull6`, `fitweibull`, `peakwindow`, `CDW` `cardidates`

Examples

```
## generate artificial test data
set.seed(123)
x <- seq(1, 365 * 3, 18)
y <- rlnorm(x, sd = 0.6) +  5e-5 * exp(1e-4 * ((x - 100) %% 365)^2) +
                            1e-4 * exp(3e-4 * ((x - 220) %% 200)^2)

## identify peaks and mark first peak after a certain time x
peaks <- peakwindow(x, y, xstart = 55)

## plot it
plot(peaks)

# highlight peaks of other years
peaks2 <- peakwindow(x, y, xstart = 420)
peaks3 <- peakwindow(x, y, xstart = 785)
plot(peaks2, add = TRUE)
plot(peaks3, add = TRUE)
```

```
## mark years
abline(v = seq(0, 365 * 3, 365), col = "grey")
```

---

summary.cardiFit *Summary Method for cardiFit Objects*

Description

This function is a top-level function to extract main results from cardinal date objects fitted with `fitweibull4` or `fitweibull6`.

Usage

```
## S3 method for class 'cardiFit'
summary(object, xmin = 0, xmax = 365, quantile = 0.05, symmetric = FALSE, ...)
```

Arguments

`object`	an object resulting from a call to `fitweibull6` or `fitweibull4`,
`xmin`	left boundary (in day of year) of the integral under the curve
`xmax`	right boundary (in day of year) of the integral under the curve
`quantile`	two-sided quantile (percentage of integral) which defines beginning and end of the peak,
`symmetric`	if (`TRUE`), quantiles are calculated for the whole area under the curve, otherwise for each of the branches separately,
`...`	for compatibility only.

Details

See `CDW` fro a detailed description of function parameters.

See Also

`weibull4`, `weibull6`, `fitweibull`, `peakwindow`,

`cardidates`

Examples

```
## create some test data
set.seed(123)
x <- seq(0, 360, length = 20)
y <- abs(rnorm(20, mean = 1, sd = 0.1))
y[5:10] <- c(2, 4, 7, 3, 4, 2)

## fit Weibull function with 6 free parameters
res <- fitweibull6(x, y)
```

```
## see the results
plot(res)
summary(res)
```

`weibull4` *Four-Parametric Weibull Function*

Description

Four-parametric Weibull function and its definite integral.

Usage

```
fweibull4(x, p)

aweibull4(p, lower, upper)
```

Arguments

`x`	vector of function arguments,
`p`	vector of function parameters with: `p[1]` vertical shift (zero offset), `p[2]` vertical scaling, `p[3]` shape parameter a of `dweibull`, and `p[4]` scale parameter b of `dweibull`,
`lower`	lower limit of the cumulative (integrated) function,
`upper`	upper limit of the cumulative (integrated) function.

Details

The four-parametric Weibull function is essentially based on the Weibull density function `dweibull` and its integral by the Weibull distribution function `pweibull` with two additional parameters for y scaling and zero offset. It can be given by:

$$f(x) = p_1 + p_2(p_3/p_4)(x/p_4)^{p_3-1}\exp(-(x/b)^{p_3})$$

for $x \geq 0$.

Value

`fweibull4` gives the Weibull function and `aweibull4` its definite integral (cumulative sum or area under curve).

See Also

`dweibull`,

`weibull6`, `fitweibull`, `peakwindow`, `CDW`, `cardidates`

Examples

```
x <- seq(0, 5, 0.02)
plot(x, fweibull4(x, c(0, 1, 2, 1)), type = "l", ylim = c(0, 2))
points(x, dweibull(x, 2, 1), pch = "+") ## identical to former

## shape
lines(x, fweibull4(x, c(0, 2, 1.5, 1)), type = "l", col = "orange")
## horizontal scaling
lines(x, fweibull4(x, c(0, 2, 2, 2)), type = "l", col = "green")
## shifting
lines(x, fweibull4(x, c(1, 1, 2, 1)), type = "l", col = "blue")
## vertical scaling
lines(x, fweibull4(x, c(0, 2, 2, 1)), type = "l", col = "red")


## definite integral
p <- c(0, 1, 2, 2)
plot(x, aweibull4(p, lower = 0, upper = x))

p <- c(0.1, 1, 2, 2)
plot(x, aweibull4(p, lower = 0, upper = x))
```

`weibull6` *Six-Parametric Weibull Function*

Description

Six-parametric Weibull function and its definite integral.

Usage

```
fweibull6(x, p)

aweibull6(p, lower = 0, upper = 365)
```

Arguments

`x`	vector of function arguments
`p`	vector of function parameters with: `p[1]` determines the offset before increase $ofs = (p[4] + 1) * (1 - p[1])$, `p[2]` inflexion point of increasing branch, `p[3]` steepness of increasing branch, `p[4]` offset after the peak, `p[5]` inflexion point of decreasing branch, `p[6]` steepness of decreasing branch,
`lower`	lower limit of the cumulative (integrated) function,
`upper`	upper limit of the cumulative (integrated) function.

Details

The six-parametric Weibull function is more flexible than the four-parametric version. It is possible to have different offsets before and after the peak. The function can be given by:

$$f(x) = p_4 + \exp(-(x/p_5)^{p_6}))(1 - p_1 * \exp(-(x/p_2)^{p_3}))$$

for $x \geq 0$.

Value

`fweibull6` gives the function and `aweibull6` its definite integral (cumulative function or area under curve). Note that in contrast to `aweibull4`, the integral is solved numerically and that the function returns a scalar, not a vector.

See Also

`weibull4`,

`fitweibull`, `CDW`, `peakwindow`, `cardidates` `Vectorize`

Examples

```
x <- seq(0, 150)
plot(x, fweibull6(x, c(0.833, 40, 5, 0.2, 80, 5)), type = "l", ylim = c(0,2))

## interpretation of offsets
ofs1 <- 0.1
ofs2 <- 0.3
p1 <- 1-ofs1/(ofs2 + 1)

lines(x, fweibull6(x, c(p1, 20, 5, ofs2, 60, 5)), col = "red")

## definite integratel from zero to 150, returns scalar
aweibull6(c(p1, 20, 5, ofs2, 60, 5), lower = 0, upper = 150)

## use Vectorize to create vectorized functions
vec.aweibull6 <- Vectorize(aweibull6, "upper")
plot(x, vec.aweibull6(c(p1, 20, 5, ofs2, 60, 5), lower = 0, upper = x))
```

Index