Package 'sitar'

March 12, 2019

Type Package

Title Super Imposition by Translation and Rotation Growth Curve Analysis

Version 1.1.1

Maintainer Tim Cole <tim.cole@ucl.ac.uk>

Description Functions for fitting and plotting SITAR (Super Imposition by Translation And Rotation) growth curve models. SITAR is a shape-invariant model with a regression B-spline mean curve and subject-specific random effects on both the measurement and age scales. The model was first described by Lindstrom (1995) <doi:10.1002/sim.4780141807> and developed as the SITAR method by Cole et al (2010) <doi:10.1093/ije/dyq115>.

License GPL (>= 2)

URL https://github.com/statist7/sitar

Depends nlme, R (>= 3.0.0), splines

Imports dplyr, glue, purrr, quantreg, rlang, rsample, stats, tibble, tidyr

Suggests knitr, rmarkdown

VignetteBuilder knitr

Encoding UTF-8

LazyData yes

LazyLoad yes

RoxygenNote 6.1.1

NeedsCompilation no

Author Tim Cole [aut, cre] (<https://orcid.org/0000-0001-5711-8200>)

Repository CRAN

Date/Publication 2019-03-12 16:30:03 UTC

R topics documented:

`sitar-package` *SITAR (SuperImposition by Translation And Rotation) growth curve analysis*

Description

SITAR is a method of growth curve analysis, based on **nlme**, that estimates a single mean growth curve as a regression B-spline, plus a set of up to three fixed and random effects (a, b and c) defining how individual growth curves differ from the mean curve. SITAR stands for SuperImposition by Translation And Rotation.

Details

The package also contains some utility functions for the LMS method, as used to construct growth reference centiles (see **gamlss**).

Package:	sitar
Type:	Package
Version:	1.0
Date:	2013-09-23
License:	GPL-2

Effect a (or alpha) measures size, and is a random intercept relative to the spline curve intercept. Effect b (or beta) measures tempo, the timing of the growth process, and reflects a shift on the x scale relative to the mean. Effect c (or gamma) is velocity, and indicates how the x scale is stretched or shrunk reflecting the rate at which 'time' passes for individuals. The aim is for individual curves, adjusted for a, b and c, to lie on top of (i.e. be superimposed on) the mean curve.

The package creates an object of class `sitar`, based on **nlme**, representing the nonlinear mixed-effects model fit. Generic functions such as `print`, `plot` and `summary` have methods to show the results of the fit, along with `resid`, `coef`, `fitted`, `fixed.effects` and `random.effects` to extract some of its components. The functions `AICadj`, `BICadj` and `varexp` compare respectively the AIC, BIC and variance explained of a series of models, taking into account any transformations of the y variable. Functions `plotclean`, `velout`, `codeplot` and `zapvelout` are useful to clean the data file.

Author(s)

Tim Cole <tim.cole@ucl.ac.uk>

References

The idea of SITAR growth curve analysis arose from the paper by Beath (2007) and was first described in Cole et al (2010). The other references describe applications of SITAR to a variety of data forms.

Beath KJ. Infant growth modelling using a shape invariant model with random efffects. Stat Med 2007;26:2547-64.

Cole TJ, Cortina Borja M, Sandhu J, et al. Nonlinear growth generates age changes in the moments of the frequency distribution: the example of height in puberty. Biostatistics 2008;9:159-71.

Cole TJ, Donaldson MD, Ben-Shlomo Y. SITAR–a useful instrument for growth curve analysis. Int J Epidemiol 2010;39:1558-66.

Gault EJ, Perry RJ, Cole TJ, et al. Effect of oxandrolone and timing of pubertal induction on final height in Turner's syndrome: randomised, double blind, placebo controlled trial. BMJ 2011;\-342:d1980.

Johnson L, Llewellyn CH, van Jaarsveld CHM, et al. Genetic and environmental influences on infant growth: prospective analysis of the Gemini twin birth cohort. PLoS ONE 2011;6:e19918.

Prentice A, Dibba B, Sawo Y, et al. The effect of prepubertal calcium carbonate supplementation on the age of peak height velocity in Gambian adolescents. Am J Clin Nutr 2012;96:1042-50.

Dean MC, Cole TJ. Human life history evolution explains dissociation between the timing of tooth eruption and peak rates of root growth. PLoS ONE 2013;8:e54534.

Cole TJ, Statnikov Y, Santhakumaran S, et al. Birth weight and longitudinal growth in infants born below 32 weeks gestation: a UK population study. Arch Dis Child Fetal Neonatal Ed 2014;99:F34-F40.

Cole TJ, Rousham EK, Hawley NL, et al. Ethnic and sex differences in skeletal maturation among the Birth to Twenty cohort in South Africa. Arch Dis Child 2014;100:138-43.

Cole TJ, Pan H, Butler GE. A mixed effects model to estimate timing and intensity of pubertal growth from height and secondary sexual characteristics. Ann Hum Biol 2014;41:7683.

Johnson L, van Jaarsveld CHM, Llewellyn CH, et al. Associations between infant feeding and the size, tempo and velocity of infant weight gain: SITAR analysis of the Gemini twin birth cohort. Int J Obes 2014;38:980-7.

Pizzi C, Cole TJ, Richiardi L, et al. Prenatal influences on size, velocity and tempo of iInfant growth: findings from three contemporary cohorts. PLoS ONE 2014;9:e90291.

Ward KA, Cole TJ, Laskey MA, et al. The Effect of Prepubertal Calcium Carbonate Supplementation on Skeletal Development in Gambian Boys-A 12-Year Follow-Up Study. J C E M 2014;99:3169-76.

`anova.sitar` *Compare Likelihoods of Fitted SITAR Objects*

Description

anova method for `sitar` objects, based on `anova.lme`.

Usage

```
## S3 method for class 'sitar'
anova(object, ..., test = TRUE, type = c("sequential",
  "marginal"), adjustSigma = TRUE, Terms, L, verbose = FALSE)
```

Arguments

object	an object inheriting from class sitar.
...	other optional fitted model objects.
test	an optional logical value controlling whether likelihood ratio tests should be used.
type	an optional character string specifying the type of sum of squares to be used.
adjustSigma	see anova.lme.
Terms	see anova.lme.
L	see anova.lme.
verbose	an optional logical value.

Value

a data frame inheriting from class "anova.lme".

Author(s)

Tim Cole <tim.cole@ucl.ac.uk>

---

apv_se *Bootstrap standard errors for SITAR peak velocity and age at peak velocity*

Description

apv_se bootstraps a SITAR model to generate standard errors for age at peak velocity (apv) and peak velocity (pv).

Usage

```
apv_se(object, nboot = 10, seed = NULL, plot = FALSE, ...)
```

Arguments

object	SITAR model.
nboot	number of bootstrap samples (default 10).
seed	integer to initialize the random number generator (default NULL).
plot	logical to control plotting (default FALSE).
...	optional arguments defining the velocity curve to be bootstrapped, and the plot. See Details.

Details

The mean velocity curve to be bootstrapped can be modified with arguments `subset`, `abc`, `xfun`, `yfun` or `ns`.

If `plot` is TRUE, the original velocity curve is plotted along with each bootstrap sample's pv versus apv.

Value

a 2x2 array giving the mean and se of apv and pv, with attribute "bs" a tibble containing the bootstrap estimates of apv and pv, with NAs removed.

Author(s)

Tim Cole <tim.cole@ucl.ac.uk>

Examples

```
data(heights)
## fit sitar model for height
model <- sitar(x = age, y = height, id = id, data = heights, df = 4)

## bootstrap standard errors for age at peak velocity and peak velocity
output <- apv_se(model, nboot=3, seed=111, plot=TRUE)
```

---

berkeley *The Berkeley Child Guidance Study*

Description

The Berkeley Child Guidance Study dataset contains longitudinal anthropometry data for 136 children from birth to 21 years.

Usage

```
berkeley
```

Format

A data frame with 4884 observations on the following 10 variables:

id factor with levels 201-278 male and 301-385 female

age years, numeric vector

height cm, numeric vector

weight kg, numeric vector

stem.length cm, numeric vector

bi.acromial cm, numeric vector

bi.iliac cm, numeric vector

leg.circ cm, numeric vector

strength lb, numeric vector

sex factor with level 1 male and level 2 female

Details

The data are for 66 boys and 70 girls from Berkeley, California born in 1928-29 of north European ancestry, and followed from birth to 21 years. Measurements were at ages 0, 0.085, 0.25 to 2 (3-monthly), 2 to 8 (annually), and 8 to 21 (6-monthly) years.

The children were measured for height, weight (undressed), stem length, biacromial diameter, biiliac diameter, leg circumference, and dynamometric strength. The data were provided as an appendix to the book by Tuddenham and Snyder (1954), and a few transcription errors are corrected here. The `growth` dataset in the `fda` package uses heights from the same study.

References

Tuddenham RD, Snyder MM. Physical growth of California boys and girls from birth to eighteen years. University of California Publications in Child Development 1954;1:183-364.

Examples

```
data(berkeley)
## frequencies of age of measurement for each variable
## weight and length/height from birth, other variables from 6-8 years
## few measurements after 18 years
. <- as.factor(berkeley$age)
plot(levels(.), summary(.), type='s', las=1,
  xlab='age of measurement (years)', ylab='frequency of measurements')
points(levels(.), levels(.) < 0, pch=15)
for (i in 3:9) {
  .. <- .[!is.na(berkeley[, names(berkeley)[i]])]
  lines(levels(..), summary(..), type='s', col=i)
}
legend('topright', names(berkeley)[c(3:9)], text.col=c(3:9), bty='n', inset=0.04)
```

`BICadj` *Ways to compare SITAR models for fit*

Description

`BICadj` and `AICadj` calculate the BIC and AIC for SITAR models, adjusting the likelihood for Box-Cox transformed y variables. `varexp` calculates the variance explained by SITAR models, compared to the corresponding fixed effect models. `getL` is used by `[AB]ICadj` to find what power the y variable is raised to.

Usage

```
BICadj(..., pattern = NULL)

AICadj(..., k = 2, pattern = NULL)

varexp(..., pattern = NULL)

getL(expr)
```

Arguments

...	one or more SITAR models.
pattern	regular expression defining names of models.
k	numeric, the penalty per parameter to be used; the default k = 2 is the classical AIC.
expr	quoted or unquoted expression containing a single variable name.

Details

The deviance is adjusted if the y variable is power-transformed, using the formula

$$adjusteddeviance = deviance - 2n((\lambda - 1) * log(gm) + log(abs(\lambda)))$$

where λ is the power transform, and n and gm are the length and geometric mean of y.

The variance explained is given by

$$\%explained = 100 * (1 - (\sigma_2/\sigma_1)^2)$$

where σ_1 is the fixed effects RSD and σ_2 the SITAR random effects RSD.

BICadj and AICadj accept non-sitar models with a logLik class. varexp ignores objects not of class sitar.

getL does not detect if the variable in expr, or its log, contains a multiplying constant, so that the expressions log(x) and 1 + 2 * log(3 * x) both return 0.

Value

For BICadj and AICadj a named vector of deviances in increasing order. For varexp a named vector of percentages in decreasing order. For getL the power the variable in expr is raised to, or NA if expr is not a power of (a multiple of) the variable.

Author(s)

Tim Cole <tim.cole@ucl.ac.uk>

See Also

BIC, AIC

Examples

```
data(heights)
## fit sitar model for height
m1 <- sitar(x=age, y=height, id=id, data=heights, df=5)

## update it for log(height)
m2 <- update(m1, y=sqrt(height))

## compare variance explained in the two models
varexp(m1, m2)

## compare BIC adjusting for sqrt transform
## the pattern matches names starting with "m" followed by a digit
BICadj(pattern="^m[0-9]")

## find what power height is raised to
getL(quote(sqrt(sqrt(height))))
```

---

bupdate *Update the b fixed effect to minimise the b-c random effect correlation*

Description

A function to update the value of `bstart`, the starting value for the b fixed effect, to minimise the correlation between the random effects b and c.

Usage

```
bupdate(x)
```

Arguments

`x`	a `sitar` object.

Value

Returns an updated value of the b fixed effect, based on the random effect covariance matrix.

Author(s)

Tim Cole <tim.cole@ucl.ac.uk>

Examples

```
## fit sitar model with b fixed effect starting value defaulting to 'mean'
m1 <- sitar(x=age, y=height, id=id, data=heights, df=5)
print(fixef(m1)['b'])
```

```
## refit with starting value chosen to minimise b-c correlation and df increased
m2 <- update(m1, bstart=bupdate(m1), df=6)
print(fixef(m2)['b'])
```

---

cLMS *LMS conversion to and from z-scores*

Description

Routines to handle references constructed with the LMS method. Given a set of LMS values, the functions convert z-scores to measurement centiles and vice versa.

Usage

```
cLMS(z, L = 1, M, S)

zLMS(x, L = 1, M, S)
```

Arguments

z	vector or one-column matrix of z-scores to be converted to measurements.
L	vector of Box-Cox transformation (lambda) values, L in the LMS method.
M	vector of medians (mu), M in the LMS method.
S	vector of coefficients of variation (sigma), S in the LMS method.
x	vector or one-column matrix of measurements to be converted to z-scores.

Details

L, M and S – and if vectors then x and z – should all be the same length, recycled if necessary. The formulae converting x to z and vice versa are:

$$z = \frac{(x/M)^L - 1}{LS}$$

$$x = M(1 + LSz)^{1/L})$$

where L is reset to 10^-7 if it is zero. The LMS method is the same as the BCCG family in the gamlss package, except that lambda in LMS is referred to as nu in BCCG.

Value

If x and z are vectors zLMS and cLMS each return a vector, respectively of z-scores and measurement centiles, with length matching the length of (the longest of) x or z, L, M and S. If x or z are matrices zLMS and cLMS each return a matrix, the number of rows matching the length of (the longest of) L, M and S, and the number of columns matching the length of x or z.

Author(s)

Tim Cole <tim.cole@ucl.ac.uk>

See Also

z2cent, LMS2z, pdLMS

Examples

```
cLMS(z = as.matrix(-2:2), L = 1:-1, M = 5:7, S = rep(0.1, 3))
cLMS(z = 0:2, L = 1:-1, M = 7, S = 0.1)
cLMS(z = as.matrix(0:2), L = 1:-1, M = 7, S = 0.1)
zLMS(x = 6.5, L = 1:-1, M = 5:7, S = rep(0.1, 3))
```

---

codeplot *Plot and zap velocity outliers in growth curves*

Description

Handles output from velout function to display growth curves with outlying points, either plotting or zapping the outliers.

Usage

```
codeplot(outliers, icode = 4, ..., print = TRUE)

zapvelout(outliers, icode)
```

Arguments

outliers	Data frame returned from velout.
icode	The code number(s) defining the subset of curves to be displayed or zapped (between 1 and 6).
...	Optional plot parameters.
print	Option to print as well as plot information on each curve.

Details

The function velout identifies putative outliers for y in data, codeplot plots them, and zapvelout sets missing those confirmed as outliers. Codes range from 0 (normal) to 8, where 4 and 6 are conventional outliers (see velout).

Value

codeplot returns summary information on each curve with an outlier of the relevant code, and optionally plots the curve. zapvelout sets to NA values of y whose code is contained in icode, and returns the modified data frame.

Author(s)

Tim Cole <tim.cole@ucl.ac.uk>

See Also

velout

Examples

```
## identify outliers
outliers <- velout(age, height, id, heights, limit=2)

## plot outliers with code 4 or 6
codeplot(outliers, icode=c(4,6))

## set the 8 outliers missing
newheights <- zapvelout(outliers, icode=6)
```

---

dfpower *Tabulate BIC of SITAR models by degrees of freedom and xy power transformations*

Description

dfpower fits a series of SITAR models tabulated by specified degrees of freedom and power transformations of x and y, returning a table of function values (e.g. BIC) applied to each model.

Usage

```
dfpower(model, df, xpowers, ypowers, FUN = BICadj,
  maxIter = nlmeControl()$maxIter, verbose = FALSE)
```

Arguments

model	fitted **sitar** model to be updated.
df	vector of degrees of freedom to be fitted (defaults to df in model).
xpowers	vector of powers to apply to x (defaults to x power in model).
ypowers	vector of powers to apply to y (defaults to y power in model).
FUN	function to be tabulated (default BICadj, or e.g. AICadj or varexp).

maxIter	maximum number of iterations per fit.
verbose	logical controlling monitoring.

Details

The function provides a convenient way to optimise the model's degrees of freedom and explore transformations of x and y, based by default on adjusted BIC. The function value is returned with changed sign if there is a warning, or as NA if there is an error. The run-time can be shortened by reducing maxIter, as the models that converge do so in relatively few iterations, and much of the run-time is spent on models that fail to converge.

FUN can be any function returning a single numerical value.

The returned table can be rearranged using aperm.

Value

Table or vector of returned values.

Author(s)

Tim Cole <tim.cole@ucl.ac.uk>

Examples

```
data(heights)
m1 <- sitar(log(age), height, id, heights, 4)
dfpower(m1, df=4:5, xpowers=0:1, maxIter=4)
```

---

dfset — *Find degrees of freedom for a natural spline curve to minimise BIC or AIC*

Description

dfset fits a natural cubic spline for a range of degrees of freedom, and returns the df minimising the BIC or AIC.

Usage

```
dfset(x, y, data = parent.frame(), FUN = BIC, df = 1:15,
  plot = FALSE, ...)
```

Arguments

x	vector of x coordinates.
y	vector of y coordinates.
data	environment containing x and y.
FUN	function to be minimised (e.g. BIC or AIC).
df	vector of degrees of freedom to be searched.
plot	logical controlling plotting of FUN versus df.
...	parameters to pass to plot.

Value

degrees of freedom and value of FUN at minimum.

Author(s)

Tim Cole <tim.cole@ucl.ac.uk>

Examples

```
data(heights)
dfset(age, height, heights, FUN=BIC, plot=TRUE)
dfset(age, height, heights, FUN=function(a) AIC(a, k=1))
```

---

funcall *Function call with optional inverse*

Description

Applies an expression to vector v, optionally inverting the expression first. For example if the expression is log, funcall returns log(v) if inverse is FALSE, and exp(v) if inverse is TRUE.

Usage

```
funcall(v, vcall, inverse = FALSE)
```

Arguments

v	vector
vcall	expression
inverse	logical

Details

Inverse covers functions log, exp, sqrt, ^, *, /, +, -.

Value

Returns a vector of length v.

Author(s)

Tim Cole <tim.cole@ucl.ac.uk>

`getData.sitar` *Extract elements of fitted SITAR models*

Description

getData, getCovariate and getVarCov methods for `sitar` objects, based on `lme`.

Usage

```
## S3 method for class 'sitar'
getData(object)

## S3 method for class 'sitar'
getCovariate(object, ...)

## S3 method for class 'sitar'
getVarCov(obj, ...)
```

Arguments

`object, obj`	an object inheriting from class `sitar`.
`...`	other optional arguments.

Value

Respectively the data frame and `x` variable used in the fit, and the returned variance-covariance matrix.

Author(s)

Tim Cole <tim.cole@ucl.ac.uk>

`getPeakTrough` *Identify peaks and troughs of curve*

Description

Given vectors `x` and `y`, returns their values at the peak or trough of the curve, where dy/dx = 0.

Usage

```
getPeakTrough(x, y = NULL, peak = TRUE)
```

Arguments

`x`	vector.
`y`	vector.
`peak`	logical determining whether peak or trough is returned.

Value

A length-2 vector containing the values of `x` and `y` at the peak or trough. If no peak/trough is identified `x` and `y` are set to NA.

Author(s)

Tim Cole <tim.cole@ucl.ac.uk>

Examples

```
## create mean height velocity curve
data(heights)
m1 <- sitar(age, height, id, heights, 4)
x <- getCovariate(m1)
y <- fitted(m1, level=0)
y <- predict(smooth.spline(x, y), x, deriv=1)$y

## and plot it
. <- order(x)
plot(y[.] ~ x[.], type='l', xlab='age', ylab='height')
points(t(getPeakTrough(x, y)), pch=17)
points(t(getPeakTrough(x, y, peak=FALSE)), pch=25)
```

heights *Serial heights measured in 12 girls*

Description

Heights of 12 girls from the Chard Growth Study measured twice a year between 8 and 16 years of age.

Usage

```
heights
```

Format

A data frame with 124 observations on the following 4 variables:

id factor of subject ids (levels 1:12).

age vector of ages (years).

height vector of heights (cm).

men vector of ages at menarche (years), where negative values are right censored.

Examples

```
require(graphics)
data(heights)
coplot(height ~ age | id, data = heights, panel=panel.smooth,
  show.given=FALSE, xlab='age (years)', ylab='height (cm)', pch=19)
```

ifun *Invert an expression defining a data transformation*

Description

Given a transformed variable and the expression used to transform it, `ifun` creates a function containing the inverse expression that will back-transform the variable.

Usage

```
ifun(expr, verbose = FALSE)
```

Arguments

`expr`	a single-variable call or quoted expression to be inverted. The variable's name in `expr` is referred to here as `varname`.
`verbose`	a logical controlling printing of the intermediate functions $f(.)$, $g(.)$, $h(.)$ etc (see 'Details').

Details

ifun returns the inverting function such that ifun(expr)(eval(expr)) = varname, where expr can include any of the invertible functions in the 'Math' and 'Ops' groups.

To illustrate its use, consider variants of the sitar model height ~ age where age and/or height are transformed, e.g. height ~ log(age) or log(height) ~ sqrt(age). Each model is of the form y ~ x but the units of x and y vary.

The models are compared by plotting the fitted curves in their original units, by first applying suitable functions to back-transform x and y. For example with log(age), where expr = quote(log(age)), the function ifun = function(x) exp(x) back-transforms eval(expr) to give age. See the first example.

ifun generalises this process for increasingly complex expr, as the next two examples show.

The final example shows ifun in action with plot.sitar, which uses ifun as the default function for arguments xfun and yfun - they are used to back-transform x and y using the values of expr for x and y extracted from the model's sitar call.

Structuring expr suitably ensures it can be inverted - it should contain a single mention of a single variable (varname here), and possibly functions such as $f(.)$, $g(.)$, $h(.)$ etc such that expr = $f(g(h((varname))))$. The number of such functions is in principle unlimited. ifun returns function(x) $h^{-1}(g^{-1}(f^{-1}((x))))$, which ensures that expr is invertible so long as the individual functions are invertible.

Value

The required inverting function, with single argument x. Its "varname" attribute contains varname as a character string.

Author(s)

Tim Cole <tim.cole@ucl.ac.uk>

See Also

plot.sitar

Examples

```
## for best effect run all the code

## define varname variable
(age <- 1:9)

## simple case - age transformed to log(age)
(expr <- quote(log(age)))
## transformed age
eval(expr)
## inverting function, with "varname" attribute set to "age"
ifun(expr)
## inverted transformed age identical to age
all.equal(age, ifun(expr)(eval(expr)))
```

```
## more complex case - age transformed to log age since conception
(expr <- quote(log(age + 0.75)))
## inverting function
ifun(expr)
## inverted transformed age identical to age
all.equal(age, ifun(expr)(eval(expr)))

## ludicrously complex case involving exp, log10, ^, pi and trigonometry
(expr <- quote((exp(sin(pi * log10(age + 0.75)/2) - 1)^4)))
## inverting function, showing intermediate stages
ifun(expr, verbose=TRUE)
## identical to original
all.equal(age, ifun(expr)(eval(expr)))

## example of plot.sitar back-transforming transformed x and y in sitar models
## fit sitar models
m1 <- sitar(x=age, y=height^2, id=id, data=heights, df=6)
m2 <- update(m1, x=log(age+0.75), y=height)

## default plot options for xfun & yfun back-transform x & y to original scales
## xfun=ifun(x$call.sitar$x)
## yfun=ifun(x$call.sitar$y)
## compare mean curves for the two models where x & y are on the original scales
plot(m1, 'd', las=1)
lines(m2, 'd', col=2)
```

LMS2z *Convert to/from measurement from/to z-score with growth reference*

Description

A function to convert between measurements and z-scores using a growth reference previously fitted by the LMS method.

Usage

```
LMS2z(x, y, sex, measure, ref, toz = TRUE, LMStable = FALSE)
```

Arguments

x	vector of ages.
y	vector or one-column matrix of either measurements or z-scores, depending on the value of toz.
sex	vector where 1/2 = males/females = boys/girls = TRUE/FALSE, based on the uppercase first character of the string.
measure	measurement name, as character string, the choice depending on the choice of ref (see e.g. references uk90, who06 and ukwhopt).

ref	growth reference, either as name or character string, available as a data object or data frame.
toz	logical set to TRUE for conversion from measurement to z-score, or FALSE for the reverse.
LMStable	logical set to TRUE to return the associated LMS table as a data frame in attribute LMStable.

Details

Vectors of L, M and S corresponding to x and sex are extracted using cubic interpolation and passed to either cLMS or zLMS, depending on toz.

Value

A vector or matrix containing the transformed values. If y is a vector then a vector is returned, else if y is a one-column matrix then a matrix is returned, with length(x) rows and length(y) columns. The matrix row names are set to x, and the column names to either y or if toz is FALSE, z2cent(y). If LMStable is TRUE the associated LMS table is returned as a data frame in attribute LMStable.

Author(s)

Tim Cole <tim.cole@ucl.ac.uk>

See Also

z2cent. The LMS method can be fitted to data using the package gamlss with the BCCG or BCCGo family, where nu (originally lambda), mu and sigma correspond to L, M and S respectively.

Examples

```
## convert girls' heights data to UK 90 z-scores
data(heights)
data(uk90)
with(heights, LMS2z(age, height, sex = 2, measure = 'ht', ref = 'uk90'))

## construct table of boys weight centiles by age for WHO standard
data(who06)
zs <- -4:4*2/3 # z-scores for centiles
ages <- 0:12/4 # 3-month ages
LMS2z(ages, as.matrix(zs), sex = 'm', measure = 'wt', ref = who06, toz = FALSE)
```

LMSfit *Estimate LMS curves from tabulated growth reference centiles*

Description

A function to summarise an existing set of growth reference centiles as the L, M and S curves of the LMS method.

Usage

```
LMSfit(x, y, sex, data = parent.frame(), centiles = c(3, 10, 25, 50,
  75, 90, 97), df = c(6, 10, 8), L1 = FALSE, plot = TRUE, ...)
```

Arguments

x	vector of tabulated ages.
y	matrix of corresponding measurement centiles, e.g. of height or weight, with nrows = length(x) and ncols = length(centiles).
sex	two-level factor where level 1 corresponds to male and level 2 to female.
data	optional data frame containing x, y and sex.
centiles	vector of centiles corresponding to the columns of y, default c(3, 10, 25, 50, 75, 90, 97).
df	length-3 vector with the cubic smoothing spline equivalent degrees of freedom (edf) for the L, M and S curves, default c(6, 10, 8).
L1	logical constraining the L curve to 1, i.e. a Normal distribution, default FALSE.
plot	logical to plot the estimated L, M and S curves, default TRUE.
...	optional graphical parameters for the plots.

Details

At each age the optimal Box-Cox power Lopt is estimated to render the centiles closest to Normal, and the corresponding median Mopt and coefficient of variation Sopt are derived. The three sets of values are then smoothed across age to give L, M and S.

Value

A list with the results:

list("LMS") data frame of sex, x, L, M, S, Lopt, Mopt, Sopt.

list("ey") matrix of predicted values of y.

list("ez") matrix of predicted values of z.

list("fit") matrix of summary statistics for ey, giving for each column cmean the mean centile, zmean the mean z-score, zSD the SD of the z-score, and zmin and zmax the minimum and maximum z-scores.

Author(s)

Tim Cole <tim.cole@ucl.ac.uk>

See Also

LMS2z, z2cent. The LMS method can be fitted to data using the package gamlss with the BCCG family, where nu (originally lambda), mu and sigma correspond to L, M and S respectively.

Examples

```
## first construct table of boys weight centiles by age for WHO standard
data(who06)
zs <- -4:4*2/3 # z-scores for centiles
ages <- 0:12/4 # ages 0-3 years by 3 months
v <- vapply(as.list(zs), function(z)
 LMS2z(ages, z, sex = 1, measure = 'wt', ref = 'who06', toz = FALSE),
  rep(0, length(ages)))
round(v, 2)

## then back-calculate the original LMS curves and display summary statistics
LMSfit(x=ages, y=v, sex=1, centiles=pnorm(zs)*100, plot=FALSE)
```

---

mplot *Plot multiple growth curves*

Description

Function to plot multiple growth curves indexed by subject id.

Usage

```
mplot(x, y, id, data = parent.frame(), subset = NULL, add = FALSE,
  ...)
```

Arguments

x	vector of x coordinates.
y	vector of y coordinates.
id	factor denoting subject levels.
data	optional dataframe containing x, y and id.
subset	optional logical defining a subset of rows in data.
add	optional logical defining whether the plot is pre-existing (TRUE) or new (FALSE).
...	Further graphical parameters (see par) may also be supplied as arguments, particularly background colour bg, character expansion cex, colour col, line type lty, line width lwd and character pch.

Details

The arguments x, y and id can be given as character strings. The par parameters can be functions of vector variables in data, e.g. to colour curves separately by id use: col = id.

Author(s)

Tim Cole <tim.cole@ucl.ac.uk>

Examples

```
mplot(age, height, id, heights, col=id)
```

---

pdLMS *Plot frequency distributions(s) for given L, M and S values in LMS method*

Description

The LMS method defines frequency distributions in terms of L, M and S parameters. pdLMS plots one or more LMS distributions and optionally returns specified centiles on each distribution.

Usage

```
pdLMS(L = 1, M = 1, S = 0.2, zcent = NULL, zlim = 3.5,
  N = 1000, plot = TRUE, ...)
```

Arguments

L	vector of Box-Cox transformation (lambda) values, L in the LMS method (default 1 corresponding to the Normal distribution).
M	vector of medians (mu), M in the LMS method (default 1).
S	vector of coefficients of variation (sigma), S in the LMS method (default 0.2).
zcent	optional vector of z-scores for conversion to the measurement scale under each distribution.
zlim	scalar defining z-score limits underlying x-axis (default 3.5).
N	number of points per distribution curve (default 1000).
plot	logical for plotting (default TRUE).
...	Further graphical parameters (see par) may also be supplied as arguments, particularly colour col, line type lty, line width lwd and character pch.

Details

L, M and S should all be the same length, recycled if necessary.

Value

An invisible list with the following components:

x	vector of x values for plotting.
density	matrix of densities for each distribution.
centile	matrix of measurement centiles corresponding to zcent under each distribution.

The distributions can be plotted with matplot(x, density, type='l').

Author(s)

Tim Cole <tim.cole@ucl.ac.uk>

See Also

z2cent, LMS2z, cLMS

Examples

```
## plot normal distribution
pdLMS()
## compare variety of distributions
## with centiles corresponding to +3 z-scores
pdLMS(L=-2:3, M=2:3, S=1:3/10, zcent=3, lty=1)
```

---

plot.sitar *Plot SITAR model*

Description

plot and lines methods for objects of class sitar, providing various flavours of plot of the fitted growth curves. Also helper functions to return the data for plotting, e.g. with ggplot2.

Usage

```
## S3 method for class 'sitar'
plot(x, opt = "dv", labels, apv = FALSE, xfun = NULL,
  yfun = NULL, subset = NULL, ns = 101, abc = NULL, trim = 0,
  add = FALSE, nlme = FALSE, returndata = FALSE, ..., xlab = NULL,
  ylab = NULL, vlab = NULL, xlim = c(NA, NA), ylim = c(NA, NA),
  vlim = c(NA, NA), legend = list(x = "topleft", inset = 0.04, bty =
  "o"))

## S3 method for class 'sitar'
lines(x, ...)
```

```
plot_d(x, ...)

plot_v(x, ...)

plot_D(x, ...)

plot_V(x, ...)

plot_u(x, ...)

plot_a(x, ...)

plot_c(x, ...)
```

Arguments

x	object of class sitar.
opt	character string containing a subset of letters corresponding to the options: 'd' for fitted Distance curve, 'v' for fitted Velocity curve, 'c' for fitted Crosssectional distance curve, 'D' for individual fitted Distance curves, 'V' for individual fitted Velocity curves, 'u' for Unadjusted individual growth curves, and 'a' for Adjusted individual growth curves. Options 'dvcDV' give spline curves, while 'ua' give data curves made up as line segments. If both distance and velocity curves are specified, the axis for the velocity curve appears on the right side of the plot (y2), and a legend identifying the distance and velocity curves is provided.
labels	optional character vector containing plot labels for x, y and y velocity from the original SITAR model. The three elements can alternatively be provided via parameters xlab, ylab and vlab. The latter take precedence. Default labels are the names of x and y, and "y velocity", suitably adjusted to reflect any back-transformation via xfun and yfun.
apv	optional logical specifying whether or not to calculate the age at peak velocity from the velocity curve. If TRUE, age at peak velocity is calculated as the age when the second derivative of the fitted curve changes sign (after applying xfun and/or yfun). Age at peak velocity is marked in the plot with a vertical dotted line, and its value, along with peak velocity, is printed and returned. NB their standard errors can be obtained using the bootstrap with the function apv_se.
xfun	optional function to be applied to the x variable prior to plotting. Defaults to NULL, which translates to ifun(x$call.sitar$x) and inverts any transformation applied to x in the original SITAR model call. To plot on the transformed scale set xfun to I.
yfun	optional function to be applied to the y variable prior to plotting. Defaults to NULL, which translates to ifun(x$call.sitar$y) and inverts any transformation applied to y in the original SITAR model call. To plot on the transformed scale set yfun to I.
subset	optional logical vector of length x defining a subset of data rows to be plotted, for x and data in the original sitar call.
ns	scalar defining the number of points for spline curves (default 101).

abc	vector of named values of random effects a, b and c used to define an individual growth curve, e.g. abc=c(a=1, c=-0.1). Alternatively a single character string defining an id level whose random effect values are used. If abc is set, level is ignored. If abc is NULL (default), or if a, b or c values are missing, values of zero are assumed.
trim	number (default 0) of long line segments to be excluded from plot with option 'u' or 'a'. See Details.
add	optional logical defining if the plot is pre-existing (TRUE) or new (FALSE). TRUE is equivalent to using lines.
nlme	optional logical which set TRUE plots the model as an nlme object, using plot.nlme arguments.
returndata	logical defining whether to plot the data (default FALSE) or just return the data for plotting (TRUE).
...	Further graphical parameters (see par) may also be supplied as arguments, e.g. line type lty, line width lwd, and colour col. For the velocity (y2) plot y2par can be used (see Details).
xlab	optional label for x axis
ylab	optional label for y axis
vlab	optional label for v axis (velocity)
xlim	optional x axis limits
ylim	optional y axis limits
vlim	optional v axis limits
legend	optional list of arguments for legend with distance-velocity plots

Details

For options involving both distance curves (options 'dcDua') and velocity curves (options 'vV') the velocity curve plot (with right axis) can be annotated with par parameters given as a named list called y2par. To suppress the legend that comes with it set legend = NULL.

The helper functions plot_d, plot_v, plot_D, plot_V, plot_u, plot_a and plot_c correspond to the seven plot options defined by their last letter, and return the data for plotting, e.g. for use with ggplot2.

The trim option allows unsightly long line segments to be omitted from plots with options 'a' or 'u'. It ranks the line segments on the basis of the age gap (dx) and the distance of the midpoint of the line from the mean curve (dy) using the formula abs(dx)/mad(dx) + abs(dy)/mad(dy) and omits those with the largest values.

Value

If returndata is FALSE returns invisibly a list of (up to) three objects:

usr	value of par('usr') for the main plot.
usr2	the value of par('usr') for the velocity (y2) plot.

apv — if argument apv is TRUE a named list giving the age at peak velocity (apv) and peak velocity (pv) from the fitted velocity curve, either overall or (with options D or V, invisibly) for all subjects.

If returndata is TRUE (which it is with the helper functions) returns invisibly either a tibble or named list of tibbles, containing the data to be plotted. The helper functions each return a tibble. The variable names are '.x', '.y' and (for curves grouped by subject) '.id'. Note that '.x' and '.y' are returned after applying xfun and yfun. Hence if for examplex x = log(age) in the original sitar call then '.x' corresponds by default to age.

Author(s)

Tim Cole <tim.cole@ucl.ac.uk>

See Also

mplot, plotclean, ifun, apv_se

Examples

```
## fit sitar model
m1 <- sitar(x=age, y=height, id=id, data=heights, df=4)

## draw fitted distance and velocity curves
## with velocity curve in blue
## adding age at peak velocity (apv)
plot(m1, y2par=list(col='blue'), apv=TRUE)

## bootstrap standard errors for apv and pv
## Not run:
res <- apv_se(m1, nboot=20, plot=TRUE)

## End(Not run)
## draw individually coloured growth curves adjusted for random effects
## using same x-axis limits as for previous plot
plot(m1, opt='a', col=id, xlim=xaxsd())

## add mean curve in red
lines(m1, opt='d', col='red', lwd=2)

## add mean curve for a, b, c = -1 SD
lines(m1, opt='d', lwd=2, abc=-sqrt(diag(getVarCov(m1))))

## draw fitted height distance curves coloured by subject, using ggplot
## Not run:
require(ggplot2)
ggplot(plot_D(m1), aes(.x, .y, colour=.id)) +
labs(x='age', y='height') +
geom_line(show.legend=FALSE)

## End(Not run)
```

plotclean *Plot multiple growth curves to identify outliers*

Description

A version of mplot to plot growth curves and identify outliers. When outliers are clicked on, and if id is specified, the corresponding growth curve is highlighted. If id is not specified the selected point is highlighted. Use right-click to exit.

Usage

```
plotclean(x, y, id = NULL, data = parent.frame(), n = length(x),
  par.out = list(pch = 20), ...)
```

Arguments

x	vector of x coordinates.
y	vector of y coordinates.
id	factor of subject levels indexing each growth curve.
data	optional dataframe containing x, y and id.
n	maximum number of points to be identified.
par.out	list of optional graphical parameters to control appearance of selected outlying points and lines.
...	Further graphical parameters (see par) may also be supplied as arguments for lines and points, particularly line type, lty, line width, lwd and color, col.

Value

plotclean returns either a vector rows (if data is not specified) or a list:

rows	a vector of row numbers corresponding to the selected points.
data	a subset of data consisting of rows rows, and columns id, x and y.

Author(s)

Tim Cole <tim.cole@ucl.ac.uk>

Examples

```
if (interactive()) plotclean(age, height, id, heights)
```

predict.sitar *Predict SITAR model*

Description

Predict method for sitar objects, based on predict.lme.

Usage

```
## S3 method for class 'sitar'
predict(object, newdata = getData(object), level = 1L,
  ..., deriv = 0L, abc = NULL, xfun = function(x) x,
  yfun = function(y) y)
```

Arguments

object	an object inheriting from class sitar.
newdata	an optional data frame to be used for obtaining the predictions, defaulting to the data used to fit object. It requires named columns for x, and for id if level = 1, matching the names in object. Variables with the reserved names x=.x or id=.id take precedence over the model x and id variables. Any covariates in a.formula, b.formula or c.formula can also be included. By default their values are set to the mean, so when level = 0 the prediction represents the mean curve.
level	an optional integer vector giving the level(s) of grouping to be used in obtaining the predictions, level 0 corresponding to the population predictions. Defaults to level 1, and level = 0:1 fits both levels.
...	other optional arguments: asList, na.action and naPattern.
deriv	an optional integer specifying predictions corresponding to either the fitted curve or its derivative. deriv = 0 (default) specifies the distance curve, deriv = 1 the velocity curve and deriv = 2 the acceleration curve.
abc	an optional named vector containing values of a subset of a, b and c, default NULL. Ignored if level = 0. It gives predictions for a single subject with the specified values of a, b and c, where missing values are set to 0. Alternatively abc can contain the value for a single id.
xfun	an optional function to apply to x to convert it back to the original scale, e.g. if x = log(age) then xfun = exp. Only relevant if deriv > 0 - see Details.
yfun	an optional function to apply to y to convert it back to the original scale, e.g. if y = sqrt(height) then yfun = function(z) z^2.

Details

When deriv = 1 the returned velocity is in units of yfun(y) per xfun(x). So if x and/or y are transformed, velocity in units of y per x can be obtained by specifying xfun and/or yfun to back-transform them appropriately.

Value

A vector of the predictions, or a list of vectors if `asList = TRUE` and `level == 1`, or a data frame if `length(level) > 1`.

Author(s)

Tim Cole <tim.cole@ucl.ac.uk>

See Also

`ifun` for a way to generate the functions `xfun` and `yfun` automatically from the `sitar` model call.

Examples

```
data(heights)
## fit model
m1 <- sitar(x=age, y=height, id=id, data=heights, df=5)

## predictions at level 0
predict(m1, newdata=data.frame(age=9:16), level=0)

## predictions at level 1 for subject 5
predict(m1, newdata=data.frame(age=9:16, id=5), level=1)

## velocity predictions for subjects with early and late puberty
vel1 <- predict(m1, deriv=1, abc=c(b=-1))
mplot(age, vel1, id, heights, col=id)
vel1 <- predict(m1, deriv=1, abc=c(b=1))
mplot(age, vel1, id, heights, col=id, add=TRUE)
```

---

print.sitar *Print SITAR model*

Description

Print method for `sitar` objects, based on `print.lme`.

Usage

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

Arguments

`x`	an object inheriting class `sitar`.
`...`	other optional arguments.

Author(s)

Tim Cole <tim.cole@ucl.ac.uk>

`print.summary.sitar` *Print summary of SITAR model*

Description

A `print.summary` method for `sitar` objects.

Usage

```
## S3 method for class 'summary.sitar'
print(x, verbose = FALSE, ...)
```

Arguments

`x`	an object inheriting from class `summary.sitar`.
`verbose`	a logical to control the amount of output.
`...`	to specify extra arguments.

Value

A formatted summary of the object.

Author(s)

Tim Cole <tim.cole@ucl.ac.uk>

`recalib` *Recalibrate x, y data using SITAR random effects*

Description

A function to recalibrate x,y data using SITAR random effects

Usage

```
recalib(xc, yc, id = NULL, data, xcnew = NULL, ycnew = NULL, model,
  from, to)
```

Arguments

xc	character vector defining column name(s) of x data to be recalibrated.
yc	character vector defining column name(s) of y data to be recalibrated.
id	factor defining from and to rows. If NULL then recalibrate all rows.
data	dataframe containing xc, yc and id.
xcnew	column names for replacement columns xc. If default NULL then use names xcnew1... .
ycnew	column names for replacement columns yc. If default NULL then use names ycnew1... .
model	sitar model defining the random effects to be used for recalibration.
from	level of id defining existing data (must be a single row in coef{model}).
to	level of id defining data to be recalibrated (a single row in coef{model}).

Details

recalib recalibrates the values of xc and yc based on model. xc values are changed to:

(xc-c(coef[from,'b']))*exp(coef[from,'c']-coef[to,'c'])+coef[to,'b'].

yc values are changed to: yc-coef[from,'a']+coef[to,'a'].

Value

Returns the dataframe data with the from rows of xc and yc recalibrated.

Author(s)

Tim Cole <tim.cole@ucl.ac.uk>

---

sitar *Fit SITAR growth curve model*

Description

SITAR is a method of growth curve analysis, based on **nlme**, that summarises a set of growth curves with a mean growth curve as a regression spline, plus a set of up to three fixed and random effects (a, b and c) defining how individual growth curves differ from the mean curve.

Usage

```
sitar(x, y, id, data, df, knots, fixed = random, random = "a+b+c",
  a.formula = ~1, b.formula = ~1, c.formula = ~1, bounds = 0.04,
  start, xoffset = "mean", bstart = xoffset, returndata = FALSE,
  verbose = FALSE, correlation = NULL, weights = NULL,
  subset = NULL, method = "ML", na.action = na.fail,
  control = nlmeControl(msMaxIter = 100, returnObject = TRUE))

## S3 method for class 'sitar'
update(object, ..., evaluate = TRUE)
```

Arguments

x	vector of ages.
y	vector of measurements.
id	factor of subject identifiers.
data	data frame containing variables x, y and id.
df	degrees of freedom for cubic regression spline (2 or more).
knots	vector of values for knots (default df quantiles of x distribution).
fixed	character string specifying a, b, c fixed effects (default random).
random	character string specifying a, b, c random effects (default "a+b+c").
a.formula	formula for fixed effect a (default ~ 1).
b.formula	formula for fixed effect b (default ~ 1).
c.formula	formula for fixed effect c (default ~ 1).
bounds	span of x for regression spline, or fractional extension of range (default 0.04).
start	optional numeric vector of initial estimates for the fixed effects, or list of initial estimates for the fixed and random effects (see nlme).
xoffset	optional value of offset for x (either "mean" (default), "apv" or value).
bstart	optional starting value for fixed effect b (either "mean", "apv" or value (default xoffset)).
returndata	logical which if TRUE causes the model matrix to be returned, or if FALSE (default) the fitted model. Setting returndata TRUE is useful in conjunction with subset and subsample for simulation purposes.
verbose	optional logical value to print information on the evolution of the iterative algorithm (see nlme).
correlation	optional corStruct object describing the within-group correlation structure (see nlme).
weights	optional varFunc object or one-sided formula describing the within-group heteroscedasticity structure (see nlme).
subset	optional expression indicating the subset of the rows of data that should be used in the fit (see nlme).
method	character string, either "REML" or "ML" (default) (see nlme).
na.action	function for when the data contain NAs (see nlme).
control	list of control values for the estimation algorithm (see nlme) (default nlmeControl(returnObject=TRUE)).
object	object of class sitar.
...	further parameters for update consisting of any of the above sitar parameters.
evaluate	logical to control evaluation. If TRUE (default) the expanded update call is passed to sitar for evaluation, while if FALSE the expanded call itself is returned.

Details

xoffset allows the origin of x to be varied, while bstart specifies the starting value for b, both of which can affect the model fit and particularly b. The values of bstart, knots and bounds are offset by xoffset for fitting purposes, and similarly for fixed effect b.

The formulae a.formula, b.formula and c.formula can include functions and interactions, but make.names is used to ensure that the names of the corresponding model terms are valid. The modified not the original names need to be specified in predict.sitar.

update updates the model by taking the object call, adding any new parameters and replacing changed ones. Where feasible the fixed and random effects of the model being updated are suitably modified and passed via the start argument.

Value

An object inheriting from class sitar representing the nonlinear mixed-effects model fit, with all the components returned by nlme (see nlmeObject for a full description) plus the following components:

fitnlme	the function returning the predicted value of y.
call.sitar	the internal sitar call that produced the object.
xoffset	the value of xoffset.
ns	the lm object providing starting values for the B-spline curve.

Generic functions such as print, plot, anova and summary have methods to show the results of the fit. The functions resid, coef, fitted, fixed.effects, random.effects, predict, getData, getGroups, getCovariate and getVarCov can be used to extract some of its components.

Note that versions of sitar prior to 1.0.4 did not return fitnlme. Both plot and predict may require it, in which case they update the SITAR object on the fly, with a message. Also version 1.0.5 altered the defaults for xoffset and bstart. Models fitted with versions prior to 1.0.5 need refitting.

Author(s)

Tim Cole <tim.cole@ucl.ac.uk>

Examples

```
data(heights)
##  fit simple model
(m1 <- sitar(x=age, y=height, id=id, data=heights, df=5))

##  relate random effects to age at menarche (with censored values +ve)
##  both a (size) and b (tempo) are positively associated with age at menarche
amen <- abs(heights$men)
(m2 <- update(m1, a.form=~amen, b.form=~amen, c.form=~amen))
```

subsample *Sample from SITAR dataset*

Description

A function to sample from a SITAR dataset for experimental design purposes. Two different sampling schemes are offered, based on the values of `id` and `x`.

Usage

```
subsample(x, id, data, prob = 1, xlim = NULL)
```

Arguments

`x`	vector of age.
`id`	factor of subject identifiers.
`data`	dataframe containing `x` and `id`.
`prob`	scalar defining sampling probability. See Details.
`xlim`	length 2 vector defining range of `x` to be selected. See Details.

Details

With the first sampling scheme `xlim` is set to `NULL` (default), and rows of `data` are sampled with probability `prob` without replacement. With the second sampling scheme `xlim` is set to a range within `range(x)`. Subjects `id` are then sampled with probability `prob` without replacement, and all their rows where `x` is within `xlim` are selected. The second scheme is useful for testing the power of the model to predict later growth when data only up to a certain age are available. Setting `xlim` to `range(x)` allows data to be sampled by subject. The returned value can be used as the `subset` argument in `sitar` or `update.sitar`.

Value

Returns a logical the length of `x` where `TRUE` indicates a sampled value.

Author(s)

Tim Cole <tim.cole@ucl.ac.uk>

See Also

`sitar`

Examples

```
## draw 50% random sample
s50 <- subsample(age, id, heights, prob=0.5)

## truncate age range to 7-12 for 50% of subjects
t50 <- subsample(age, id, heights, prob=0.5, xlim=c(7, 12))
```

---

summary.sitar *Create summary of SITAR model*

Description

A summary method for sitar objects based on summary.lme.

Usage

```
## S3 method for class 'sitar'
summary(object, adjustSigma = TRUE, verbose = FALSE,
  ...)
```

Arguments

object	object inheriting from class sitar.
adjustSigma	optional logical (see summary.lme).
verbose	optional logical to control the amount of output in print.summary.sitar.
...	some methods for this generic require additional arguments. None are used in this method.

Value

an object inheriting from class summary.sitar with all components included in object (see lmeObject for a full description of the components) plus the components for summary.lme and the following components:

x.adj	vector of length x in object with x values adjusted for subject-specific random effects b and c.
y.adj	vector of length y in object with y values adjusted for subject-specific random effects a.
apv	length 2 vector giving respectively age at peak velocity and peak velocity based on the fitted distance curve (using transformed x and y where specified).

Author(s)

Tim Cole <tim.cole@ucl.ac.uk>

timegap *Select equally spaced ages from a vector of ages*

Description

timegap indexes elements in a vector of ages such that the indexed ages are spaced integer multiples of a time interval apart, to within a given tolerance. timegap.id is a wrapper to apply timegap within levels of factor id. The selected ages can then be split into age groups the specified time interval wide, ensuring that (virtually) every subject has at most one measurement per interval.

Usage

```
timegap(age, gap, tol = 0.1 * gap, multiple = FALSE)

timegap.id(age, id, data = parent.frame(), gap, tol = 0.1 * gap,
  multiple = FALSE)

diffid(age, id, data = parent.frame(), lag = 1, differences = 1,
  sort = FALSE, keepNA = FALSE)
```

Arguments

age	vector of ages.
gap	numeric, the required positive time gap between selected ages.
tol	numeric, the positive tolerance around the gap (default 0.1 * gap).
multiple	logical, whether or not to return multiple solutions when found (default FALSE).
id	factor of subject ids.
data	data frame optionally containing age and id.
lag	an integer indicating which lag to use.
differences	an integer indicating the order of the difference.
sort	a logical indicating whether to first sort by id and age.
keepNA	a logical indicating whether to keep generated NAs.

Details

timegap calculates all possible differences between pairs of ages, expresses them as integer multiples of gap, restricts them to those within tolerance and identifies those providing the longest sequences. For sequences of the same length, those with the smallest standard deviation of successive differences (modulo the time interval) are selected.

Value

With timegap, for unique solutions, or multiple solutions with multiple FALSE, a vector of indices named with age. With timegap.id the subject vectors are returned invisibly, concatenated.

With multiple TRUE, where there are multiple solutions they are returned as a named matrix.

diffid returns diff(age) applied within id. With keepNA TRUE a suitable number of NAs are added at the end, while if FALSE all NAs are omitted.

Author(s)

Tim Cole <tim.cole@ucl.ac.uk>

Examples

```
data(heights)

## bin age into 1-year groups by id
## gives multiple measurements per id per year
with(heights, table(floor(age), id))

## now select heights measured multiples of 1 year apart
(tg1 <- timegap.id(age, id, heights, 1))

## no more than one measurement per id per year
with(heights[tg1, ], table(floor(age), id))

## most time intervals close to 1 year
summary(diffid(age, id, heights[tg1, ], lag=1))
```

---

uk90 *UK 1990 growth reference*

Description

The UK 1990 growth reference (Freeman et al 1995, Cole et al 1998) for height, weight, body mass index, circumferences and percent body fat, fitted by the LMS method and summarised by values of L, M and S by sex from 23 weeks gestation to 23 years.

Usage

```
uk90
```

Format

A data frame with 588 observations on the following 26 variables:

years numeric vector

L.ht numeric vector

M.ht numeric vector

S.ht numeric vector

L.wt numeric vector

M.wt numeric vector

S.wt numeric vector

L.bmi numeric vector

M.bmi numeric vector

S.bmi numeric vector

L.head numeric vector

M.head numeric vector

S.head numeric vector

L.sitht numeric vector

M.sitht numeric vector

S.sitht numeric vector

L.leglen numeric vector

M.leglen numeric vector

S.leglen numeric vector

L.waist numeric vector

M.waist numeric vector

S.waist numeric vector

L.bfat numeric vector

M.bfat numeric vector

S.bfat numeric vector

sex two-level factor with level 1 male and level 2 female

Details

The L, M and S values for each measurement correspond respectively to the Box-Cox power, median and coefficient of variation of the distribution by age and sex (Cole & Green 1992). The short names for each measurement (see LMS2z) are as follows: height (ht), weight (wt), body mass index (bmi), head circumference (head), sitting height (sitht), leg length (leglen), waist circumference (waist) and percent body fat (fat).

Source

The values are tabulated in the spreadsheet British1990.xls provided with the Excel add-in LMSgrowth from:

http://www.healthforallchildren.com/shop-base/software/lmsgrowth/.

References

Cole TJ, Green PJ. Smoothing reference centile curves: the LMS method and penalized likelihood. Stat Med 1992;11:1305-19.

Cole TJ, Freeman JV, Preece MA. British 1990 growth reference centiles for weight, height, body mass index and head circumference fitted by maximum penalized likelihood. Stat Med 1998;17:407-29.

Freeman JV, Cole TJ, Chinn S, et al. Cross sectional stature and weight reference curves for the UK, 1990. Arch Dis Child 1995;73:17-24.

Examples

```
data(uk90)
## calculate median BMI in girls from birth to 10 years
LMS2z(x = 0:10, y = 0, sex = 2, measure = 'bmi', ref = 'uk90', toz = FALSE)
```

---

ukwhopt *UK-WHO growth reference including preterm*

Description

The UK 1990 revised growth reference (Cole et al 2011) for birth length, weight and head circumference, fitted by the LMS method and summarised by values of L, M and S by sex from 23 to 42 weeks gestation.

Usage

```
ukwhopt
```

Format

A data frame with 40 observations on the following 12 variables:

weeks numeric vector

years numeric vector

L.ht numeric vector

M.ht numeric vector

S.ht numeric vector

L.wt numeric vector

M.wt numeric vector

S.wt numeric vector

L.head numeric vector

M.head numeric vector

S.head numeric vector

sex two-level factor with level 1 male and level 2 female

Details

The growth reference is the birth section of the UK-WHO growth reference (see Wright et al 2010).

The L, M and S values for each measurement correspond respectively to the Box-Cox power, median and coefficient of variation of the distribution by age and sex (Cole & Green 1992). The short names for each measurement (see LMS2z) are as follows: height (ht), weight (wt) and head circumference (head).

Age is measured in `weeks` gestation and `years` post-term, where 0 years corresponds to 40 weeks gestation.

Source

The values are tabulated in the Excel spreadsheet UK_WHO_preterm.xls provided with the Excel add-in LMSgrowth from http://www.healthforallchildren.com/?product_cat=software.

References

Cole TJ, Green PJ. Smoothing reference centile curves: the LMS method and penalized likelihood. Stat Med 1992;11:1305-19.

Cole TJ, Williams AF, Wright CM, et al. Revised birth centiles for weight, length and head circumference in the UK-WHO growth charts. Ann Hum Biol 2011;38:7-11.

Wright CM, Williams AF, Elliman D, et al. Using the new UK-WHO growth charts. BMJ 2010;\-340:c1140.

Examples

```
data(ukwhopt)
## calculate median birth weight in girls from 23 to 42 weeks gestation
LMS2z(x = (23:42-40) * 7 / 365.25, y = 0, sex = 2, measure = 'wt', ref = 'ukwhopt', toz = FALSE)
```

---

velout *Identify outliers with abnormal velocity in growth curves*

Description

Quickly identifies putative outliers in a large number of growth curves.

Usage

```
velout(x, y, id, data, lag = 1, velpower = 0.5, limit = 5,
  linearise = FALSE)
```

Arguments

x	age vector.
y	outcome vector, typically weight or height.
id	factor identifying each subject.
data	data frame containing x, y and id.
lag	lag between measurements for defining growth velocity.
velpower	a value, typically between 0 and 1, defining the power of delta x to use when calculating velocity as delta(y)/delta(x)^velpower. The default of 0.5 is midway between velocity and increment.
limit	the number of standard deviations beyond which a velocity is deemed to be an outlier.
linearise	if TRUE y is converted to a residual about the median curve of y versus x.

Details

The algorithm works by viewing serial measurements in each growth curve as triplets (A-B-C) and comparing the velocities between them. Velocity is calculated as

diff(y, lag = lag) / diff(x, lag = lag) ^ velpower

Missing values for x or y are ignored. If any of the AB, BC or AC velocities are abnormal (more than `limit` SDs in absolute value from the median for the dataset) the code for B is non-zero.

Value

Returns a data frame with columns: id, x, y (from the call), code (as described below), vel1, vel2 and vel3 (corresponding to the velocities AB, BC and AC above). The 'data' attribute contains the name of 'data'.

Code is a factor taking values between 0 and 8, with 0 normal (see table below). Values 1-6 depend on the pattern of abnormal velocities, while 7 and 8 indicate a duplicate age (7 for the first in an individual and 8 for later ones). Edge outliers, i.e. first or last for an individual, have just one velocity. Code 4 indicates a conventional outlier, with both AB and BC abnormal and AC normal. Code 6 is an edge outlier. Other codes are not necessarily outliers, e.g. codes 1 or 3 may be adjacent to a code 4. Use `codeplot` to look at individual curves, and `zapvelout` to delete outliers.

code	AB+BC	AC	interpretation
0	0	0	no outlier
0	0	NA	no outlier
1	0	1	rare pattern
2	1	0	complicated - look at curve
3	1	1	adjacent to simple outlier
4	2	0	single outlier
5	2	1	double outlier
6	1	NA	edge outlier
7	-	-	first duplicate age
8	-	-	later duplicate age

Author(s)

Tim Cole <tim.cole@ucl.ac.uk>

See Also

codeplot, zapvelout

Examples

```
outliers <- velout(age, height, id, heights, limit=3)
```

---

who06 *The WHO 2006 growth standard*

Description

The WHO growth standard (WHO 2006) for height, weight, body mass index, circumferences and skinfold thicknesses, fitted by the LMS method and summarised by values of L, M and S by sex from birth to 5 years.

Usage

```
who06
```

Format

A data frame with 150 observations on the following 23 variables:

years age from 0 to 5 years

L.ht numeric vector

M.ht numeric vector

S.ht numeric vector

L.wt numeric vector

M.wt numeric vector

S.wt numeric vector

L.bmi numeric vector

M.bmi numeric vector

S.bmi numeric vector

L.head numeric vector

M.head numeric vector

S.head numeric vector

L.arm numeric vector

M.arm numeric vector

S.arm numeric vector

L.subscap numeric vector

M.subscap numeric vector

S.subscap numeric vector

L.tricep numeric vector

M.tricep numeric vector

S.tricep numeric vector

sex two-level factor with level 1 male and level 2 female

Details

The L, M and S values for each measurement correspond respectively to the Box-Cox power, median and coefficient of variation of the distribution by age and sex (Cole & Green 1992). The short names for each measurement (see LMS2z) are as follows: height (ht), weight (wt), body mass index (bmi), head circumference (head), arm circumference (arm), subscapular skinfold (subscap), and tricep skinfold (tricep).

Source

http://www.who.int/childgrowth/en/

References

World Health Organization. WHO Child Growth Standards: Methods and development: Length/height-for-age, weight-for-age, weight-for-length, weight-for-height and body mass index-for-age. Geneva: WHO; 2006.

Cole TJ, Green PJ. Smoothing reference centile curves: the LMS method and penalized likelihood. Stat Med 1992;11:1305-19.

Examples

```
data(who06)
## calculate z-score for length 60 cm in boys at age 0:12 months
LMS2z(x = 0:12/12, y = 60, sex = 1, measure = 'ht', ref = 'who06')
```

---

xaxsd *Par args xaxs and yaxs option d*

Description

Implements par('xaxs') and par('yaxs') option 'd'.

Usage

```
xaxsd(usr = par()$usr[1:2])

yaxsd(usr = par()$usr[3:4])
```

Arguments

usr	a length-2 vector defining the length of the x-axis or y-axis.

Details

Implements par('xaxs') and par('yaxs') option 'd', i.e. uses previous axis scales in a new plot.

Value

By default returns xlim/ylim args to match current setting of par()$usr, i.e. previous plot scales. Specifying usr gives scales with the usr args at the extremes. If par('xlog') or par('ylog') are set the returned limits are antilogged (to base 10).

Author(s)

Tim Cole <tim.cole@ucl.ac.uk>

Examples

```
## generate and plot 100 data points
x <- rnorm(100)
y <- rnorm(100)
plot(x, y, pch=19)

## generate and plot 10 more
## constraining axis scales to be as before
x <- rnorm(10)
y <- rnorm(10)
plot(x, y, pch=19, xlim=xaxsd(), ylim=yaxsd())

## force axis extremes to be -3 and 3
plot(x, y, pch=19, xlim=xaxsd(c(-3,3)), ylim=yaxsd(c(-3,3)))
```

---

xyadj *Adjust x and y variables for SITAR random effects*

Description

xyadj Adjusts x and y values for subject-specific random effects from a SITAR model.

Usage

```
xyadj(object, x, y = NULL, id, abc = NULL, tomean = TRUE)
```

Arguments

object	a SITAR model.
x	a vector of x coordinates. If missing, x and y and id are obtained from object.
y	a vector of y coordinates (default NULL).
id	a factor denoting the subject levels corresponding to x and y.
abc	a data frame containing random effects for a, b and c (default ranef(object)[id,]).
tomean	a logical defining the direction of adjustment. TRUE (default) indicates that individual curves are translated and rotated to match the mean curve, while FALSE indicates the reverse, the mean curve being translated and rotated to match individual curves.

Details

When tomean = TRUE the x and y values are adjusted to

$$(x - xoffset - b < fixed > - b < random >) * exp(c < random >) + xoffset + b < fixed >$$

$$y - a < random >$$

When tomean = FALSE they are adjusted to

$$(x - xoffset - b < fixed >)/exp(c < random >) + xoffset + b < fixed > + b < random >$$

$$y + a < random >$$

In each case missing values of the fixed or random effects are set to zero.

Value

The list of adjusted values:

x	numeric vector.
y	numeric vector the same length as x, or NULL.

Author(s)

Tim Cole <tim.cole@ucl.ac.uk>

Examples

```
data(heights)
## fit sitar model for height
m1 <- sitar(x=age, y=height, id=id, data=heights, df=5)

## plot unadjusted data as growth curves
```

```
plot(m1, opt='u')

## overplot with adjusted data as points
with(heights, points(xyadj(m1), col='red', pch=19))
```

---

z2cent *Express z-scores as centile character strings for plotting*

Description

Converts z-scores, typically defining centiles in a growth chart, to character strings that can be used to label the centile curves.

Usage

```
z2cent(z)
```

Arguments

z	a scalar or vector of z-scores.

Value

A character string is returned, the same length as z. Z-scores between the 1st and 99th centile are converted to centiles with one or two significant figures (lower tail) or to their complement (upper tail). For larger z-scores in absolute value the character consists of "SDS" appended to the z-score rounded to one decimal place.

Author(s)

Tim Cole <tim.cole@ucl.ac.uk>

See Also

cLMS

Examples

```
z2cent(-4:4)
z2cent(qnorm(0:100/100))
```

Index