Package 'embryogrowth'

August 30, 2019

Type Package

Title Tools to Analyze the Thermal Reaction Norm of Embryo Growth

Version 7.6

Date 2019-08-30

Author Marc Girondot <`marc.girondot@u-psud.fr`>

Maintainer Marc Girondot <`marc.girondot@u-psud.fr`>

Description Tools to analyze the embryo growth and the sexualisation thermal reaction norms.

Depends deSolve, optimx, numDeriv, parallel, HelpersMG (>= 3.6.6),R (>= 3.5)

Suggests entropy, shiny, coda, polynom, car, lmf, gam

License GPL-2

LazyData yes

LazyLoad yes

Encoding UTF-8

RoxygenNote 6.1.1

NeedsCompilation no

Repository CRAN

Date/Publication 2019-08-30 07:10:02 UTC

R topics documented:

`embryogrowth-package` *The package embryogrowth*

Description

Tools to analyze the embryo growth and the sexualisation thermal reaction norms.
The lastest version of this package can always been installed using:
install.packages("http:///max2.ese.u-psud.fr/epc/conservation/CRAN/HelpersMG.tar.gz", repos=NULL, type="source")
install.packages("http:///max2.ese.u-psud.fr/epc/conservation/CRAN/embryogrowth.tar.gz", repos=NULL, type="source")

Details

Fit a parametric function that describes dependency of embryo growth to temperature

Package:	embryogrowth
Type:	Package
Version:	7.6 build 994
Date:	2019-08-30
License:	GPL (>= 2)
LazyLoad:	yes

Author(s)

Marc Girondot `<marc.girondot@u-psud.fr>`

References

Girondot, M. & Kaska, Y. 2014. A model to predict the thermal reaction norm for the embryo growth rate from field data. Journal of Thermal Biology. 45, 96-102.

Fuentes, M.M.P.B., Monsinjon, J., Lopez, M., Lara, P., Santos, A., dei Marcovaldi, M.A.G., Girondot, M., 2017. Sex ratio estimates for species with temperature-dependent sex determination differ

according to the proxy used. Ecological Modelling 365, 55-67.

Girondot, M., Monsinjon, J., Guillon, J.-M., 2018. Delimitation of the embryonic thermosensitive period for sex determination using an embryo growth model reveals a potential bias for sex ratio prediction in turtles. Journal of Thermal Biology 73, 32-40.

Monsinjon, J., Jribi, I., Hamza, A., Ouerghi, A., Kaska, Y., Girondot, M., 2017. Embryonic growth rate thermal reaction norm of Mediterranean Caretta caretta embryos from two different thermal habitats, Turkey and Libya. Chelonian Conservation and Biology 16, 172-179.

See Also

Delmas, V., Prevot-Julliard, A.-C., Pieau, C. & Girondot, M. 2008. A mechanistic model of temperature-dependent sex determination in a Chelonian, the European pond turtle. Functional Ecology, 22, 84-93.

Girondot, M., Ben Hassine, S., Sellos, C., Godfrey, M. & Guillon, J.-M. 2010. Modeling thermal influence on animal growth and sex determination in Reptiles: being closer of the target gives new views. Sexual Development, 4, 29-38.

Girondot, M. 1999. Statistical description of temperature-dependent sex determination using maximum likelihood. Evolutionary Ecology Research, 1, 479-486.

Girondot, M., & Kaska, Y. (2014). Nest temperatures in a loggerhead- nesting beach in Turkey is more determined by sea surface temperature than air temperature. Journal of Thermal Biology, 47, 13-18.

Examples

```
## Not run:
library("embryogrowth")
packageVersion("embryogrowth")
data(nest)
formated <- FormatNests(nest)
# The initial parameters value can be:
# "T12H", "DHA",  "DHH", "Rho25"
# Or
# "T12L", "DT", "DHA",  "DHH", "DHL", "Rho25"
x <- structure(c(115.758929130522, 428.649022170996, 503.687251738993,
12.2621455821612, 306.308841227278, 116.35048615105), .Names = c("DHA",
"DHH", "DHL", "DT", "T12L", "Rho25"))
# or
x <- structure(c(118.431040984352, 498.205702157603, 306.056280989839,
118.189669472381), .Names = c("DHA", "DHH", "T12H", "Rho25"))
# pfixed <- c(K=82.33) or rK=82.33/39.33
pfixed <- c(rK=2.093313)

################################################################################
#
# The values of rK=2.093313 and M0=1.7 were used in
# Girondot, M. & Kaska, Y. 2014. A model to predict the thermal
# reaction norm for the embryo growth rate from field data. Journal of
# Thermal Biology. 45, 96-102.
#
```

```
# Based on recent analysis on table of development for both Emys orbicularis and
# Caretta caretta, best value for pfixed should be 1.209 and M0 should be 0.34.
# Girondot, M., Monsinjon, J., Guillon, J.-M., Submitted. Delimitation of the
# embryonic thermosensitive period for sex determination using an embryo growth
# model reveals a potential bias for sex ratio prediction in turtles.
#
# See the example in the stages datasets
#
################################################################################

resultNest_4p_SSM4p <- searchR(parameters=x, fixed.parameters=pfixed,
temperatures=formated, derivate=dydt.Gompertz, M0=1.7,
test=c(Mean=39.33, SD=1.92))
data(resultNest_4p_SSM4p)
par(mar=c(4, 4, 1, 1))
plot(resultNest_4p_SSM4p$data[[1]][, 1]/60/24,resultNest_4p_SSM4p$data[[1]][, 2], bty="n", las=1,
     xlab="Days of incubation", ylab="Temperatures in °C",
     type="l", xlim=c(0,70),ylim=c(20, 35))
for (i in 2:resultNest_4p_SSM4p$data$IndiceT[3]) {
  par(new=TRUE)
  plot(resultNest_4p_SSM4p$data[[i]][, 1]/60/24,resultNest_4p_SSM4p$data[[i]][, 2],
  bty="n", las=1, xlab="", ylab="", type="l", xlim=c(0,70),ylim=c(20, 35), axes = FALSE)
}
par(mar=c(4, 4, 1, 1))
pMCMC <- TRN_MHmcmc_p(resultNest_4p_SSM4p, accept=TRUE)
# Take care, it can be very long, sometimes several days
resultNest_mcmc_4p_SSM4p <- GRTRN_MHmcmc(result=resultNest_4p_SSM4p,
parametersMCMC=pMCMC, n.iter=10000, n.chains = 1, n.adapt = 0,
thin=1, trace=TRUE)
data(resultNest_mcmc_4p_SSM4p)
out <- as.mcmc(resultNest_mcmc_4p_SSM4p)
# This out obtained after as.mcmc can be used with coda package
# plot() can use the direct output of GRTRN_MHmcmc() function.
plot(resultNest_mcmc_4p_SSM4p, parameters=1, xlim=c(0,550))
plot(resultNest_mcmc_4p_SSM4p, parameters=3, xlim=c(290,320))
# But rather than to use the SD for each parameter independantly, it is
# more logical to estimate the distribution of the curves
new_result <- ChangeSSM(resultmcmc = resultNest_mcmc_4p_SSM4p, result = resultNest_4p_SSM4p,
                        temperatures = seq(from = 20, to = 35, by = 0.1),
                        initial.parameters = NULL)
par(mar=c(4, 4, 1, 5)+0.4)

plotR(result = resultNest_4p_SSM4p, parameters = new_result$par,
           ylabH = "Temperatures\ndensity", ylimH=c(0, 0.3), atH=c(0, 0.1, 0.2),
           ylim=c(0, 3), show.hist=TRUE)

# Beautiful density plots

plotR(result = resultNest_4p_SSM4p,
             resultmcmc=resultNest_mcmc_4p_SSM4p,
             curves = "MCMC quantiles", show.density=TRUE)

plotR(resultNest_6p_SSM6p, resultmcmc=resultNest_mcmc_6p_SSM6p,
```

```
            ylim=c(0, 4), show.density=TRUE, show.hist=TRUE,
            curves = "MCMC quantiles",
            ylimH=c(0,0.5), atH=c(0, 0.1, 0.2))

## End(Not run)
```

`calibrate.datalogger` *Calibrate data loggers and correct time series of temperatures*

Description

Calibrate a time series of temperatures. Use or gam or glm. If no temperatures.series is given, it will use the read.temperatures.

Usage

```
calibrate.datalogger(control.temperatures = stop("Control temperatures is missing"),
 read.temperatures = stop("Read temperatures must be indicated"),
 temperatures.series = NULL, gam = TRUE, se.fit = TRUE)
```

Arguments

`control.temperatures`	The true temperatures during the calibration process
`read.temperatures`	The read temperatures during the calibration process
`temperatures.series`	The temperatures to be converted using calibration
`gam`	Does gam should be used (TRUE) or glm (FALSE).
`se.fit`	Do standard errors are to be returned

Details

calibrate.datalogger calibrates data loggers and correct time series of temperatures.

Value

The function will return a corrected time series of temperatures as a vector if se.fit is FALSE or a list if se.fit is TRUE.

Author(s)

Marc Girondot

See Also

Other Data loggers utilities: `uncertainty.datalogger`

Examples

```
## Not run:
library(embryogrowth)
calibrate.datalogger(control.temperatures=20:30,
                     read.temperatures=(20:30)+rnorm(11))

## End(Not run)
```

`ChangeSSM` *Generate set of parameters for different forms of thermal norm of reaction*

Description

Generate a set of parameters for thermal reaction norm model.
If initial.parameters is NULL and resultmcmc is not NULL, it will generate parameters and SE based on the average of the curves.

Usage

```
ChangeSSM(result = NULL, resultmcmc = NULL, temperatures = seq(from =
  20, to = 35, by = 0.1), parameters = NULL, initial.parameters = NULL,
  fixed.parameters = NULL, outmcmc = "quantiles", progressbar = TRUE,
  ...)
```

Arguments

`result`	A result obtained by searchR()
`resultmcmc`	A result obtained by GRTRN_MHmcmc()
`temperatures`	A vector with incubation temperatures in degrees Celsius
`parameters`	A vector of parameters for model to be converted. Not necessary if result is provided.
`initial.parameters`	NULL or a vector of parameters for initial model model to be fited
`fixed.parameters`	NULL of a vector of parameters to be used but fixed
`outmcmc`	What statistic will be estimated if a mcmc is provided. Can be "mean-sd" or "quantiles".
`progressbar`	If TRUE, a progressbar is shown
`...`	A control list to be used with optim, see ?optim

Details

ChangeSSM convert different forms of thermal norm of reaction

Value

A vector with parameters or a result object formatted with new parameters is result is non null

Author(s)

Marc Girondot

Examples

```
## Not run:
data(resultNest_6p_SSM6p)
x1 <- resultNest_6p_SSM6p$par
data(resultNest_4p_SSM4p)
x2 <- resultNest_4p_SSM4p$par
temperaturesC <- (200:350)/10
s <- ChangeSSM(temperatures=temperaturesC, parameters=x1, initial.parameters=x2)
sY <- plotR(resultNest_6p_SSM6p, ylim=c(0,3), col="black", curves = "ML")
plotR(resultNest_4p_SSM4p, col="red", scaleY=sY, new=FALSE)
plotR(s$par, col="green", scaleY=sY, new=FALSE, curves = "ML")
legend("topleft", legend=c("r function to mimic", "Initial new r function",
"Fitted new r function"), lty=c(1, 1, 1), col=c("black", "red", "green"))
# Other example to fit anchored parameters
data(resultNest_4p_SSM4p)
x0 <- resultNest_4p_SSM4p$par
t <- hist(resultNest_4p_SSM4p, plot=FALSE)
x <- c(3.4, 3.6, 5.4, 5.6, 7.6, 7.5, 3.2)
names(x) <- seq(from=range(t$temperatures)[1], to=range(t$temperatures)[2],
     length.out=7)
newx <- ChangeSSM(temperatures = (200:350)/10, parameters = x0,
       initial.parameters = x,
       control=list(maxit=5000))
 # Example on how to generate a set of SSM parameters from anchored parameters
 xanchor <- GenerateAnchor(nests=resultNest_4p_SSM4p)
 x <- resultNest_4p_SSM4p$par
 xanchor["294"] <- 0
 xanchor["308"] <- 2.3291035
 x <- ChangeSSM(parameters = xanchor,
                     initial.parameters = x, control=list(maxit=5000))
 sY <- plotR(resultNest_4p_SSM4p$par, ylim = c(0,3), curves="ML")
 plotR(xprime$par, col="red", scaleY=sY, new=FALSE, curves="ML")
 legend("topleft", legend=c("Fitted parameters", "Constrainted parameters"), lty=1,
        col=c("black", "red"))
 # Weibull model
 x <- ChangeSSM(temperatures = (200:350)/10,
                parameters = resultNest_4p_SSM4p$par,
                initial.parameters = structure(c(73, 300, 26),
                                     .Names = c("k", "lambda", "scale")),
                control=list(maxit=1000))
 # normal asymmetric model
 x <- ChangeSSM(temperatures = (200:350)/10,
               parameters = resultNest_4p_SSM4p$par,
               initial.parameters = structure(c(3, 10, 8, 32),
```

```
               .Names = c("Scale", "sdL", "sdH", "Peak")),
               control=list(maxit=1000))
 # trigonometric model
 x <- ChangeSSM(temperatures = (200:350)/10,
               parameters = resultNest_4p_SSM4p$par,
               initial.parameters = structure(c(3, 20, 40, 32),
               .Names = c("Max", "LengthB", "LengthE", "Peak")),
               control=list(maxit=1000))

 # example with a mcmc object, CI being 2.SD
 # Note the symmetric CI
data(resultNest_mcmc_4p_SSM4p)
new_result <- ChangeSSM(resultmcmc = resultNest_mcmc_4p_SSM4p, result = resultNest_4p_SSM4p,
                        temperatures = seq(from = 20, to = 35, by = 0.1),
                        outmcmc = "mean-sd",
                        initial.parameters = NULL)

plotR(new_result, ylim=c(0, 3), curves="ML")
 # example with a mcmc object, CI being defined by 2.5%-97.5% quantiles
 # Note the asymmetric CI
data(resultNest_mcmc_4p_SSM4p)
new_result <- ChangeSSM(resultmcmc = resultNest_mcmc_4p_SSM4p, result = resultNest_4p_SSM4p,
                        temperatures = seq(from = 20, to = 35, by = 0.1),
                        outmcmc = "quantiles",
                        initial.parameters = NULL)

plotR(new_result, ylim=c(0, 3), curves="ML")
plotR(new_result, ylim=c(0, 3), curves="ML quantiles")

## End(Not run)
```

`DatabaseNestingArea` *Database of RMU for marine turtles*

Description

Database of RMU for marine turtles

Usage

`DatabaseNestingArea`

Format

A dataframe with raw data.

Details

Database of RMU for marine turtles

Author(s)

Marc Girondot `<marc.girondot@u-psud.fr>`

Maria Sousa Martins `<maria.esmartins@gmail.com>`

References

Wallace, B. P., DiMatteo, A. D., Hurley, B. J., Finkbeiner, E. M., Bolten, A. B., Chaloupka, M. Y., Hutchinson, B. J., Abreu-Grobois, F. A., Amorocho, D., Bjorndal, K. A., Bourjea, J., Bowen, B. W., Duenas, R. B., Casale, P., Choudhury, B. C., Costa, A., Dutton, P. H., Fallabrino, A., Girard, A., Girondot, M., Godfrey, M. H., Hamann, M., Lopez-Mendilaharsu, M., Marcovaldi, M. A., Mortimer, J. A., Musick, J. A., Nel, R., Seminoff, J. A., Troeng, S., Witherington, B. & Mast, R. B. 2010. Regional management units for marine turtles: a novel framework for prioritizing conservation and research across multiple scales. Plos One, 5, e15465.

Examples

```
## Not run:
library(embryogrowth)
data(DatabaseNestingArea)

## End(Not run)
```

`DatabaseTSD` *Database of TSD information for reptiles*

Description

Database of TSD information for reptiles
The columns are:
* Species: Name de the species in binominal nommenclature
* Country: From which country the eggs come from
* Area: Name of the beach or region the eggs come from
* RMU: For marine turtles, name of the RMU for this population; see Wallace, B.P., DiMatteo, A.D., Hurley, B.J., Finkbeiner, E.M., Bolten, A.B., Chaloupka, M.Y., Hutchinson, B.J., Abreu-Grobois, F.A., Amorocho, D., Bjorndal, K.A., Bourjea, J., Bowen, B.W., Duenas, R.B., Casale, P., Choudhury, B.C., Costa, A., Dutton, P.H., Fallabrino, A., Girard, A., Girondot, M., Godfrey, M.H., Hamann, M., Lopez-Mendilaharsu, M., Marcovaldi, M.A., Mortimer, J.A., Musick, J.A., Nel, R., Seminoff, J.A., Troeng, S., Witherington, B., Mast, R.B., 2010. Regional management units for marine turtles: a novel framework for prioritizing conservation and research across multiple scales. Plos One 5, e15465.
* Incubation.temperature: Nominal incubation temperature
* Incubation.temperature.Accuracy: What is the accuracy of the measure of temperature
* Incubation.temperature.SD: Experimental SD of incubation temperatures
* Incubation.temperature.Amplitude: How much the temperature could fluctuate around nominal temperature
* Correction.factor: Difference between the incubator temperature and the eggs temperature
* IP.min: Shorter incubation period

* IP.max: Longer incubation period
* IP.mean: Mean incubation periods
* IP.SD: Standard deviation for incubation periods
* Total: Total number of eggs incubated
* Hatched: Number of hatchlings
* NotHatched: Number of embryos with development visible but dead during incubation
* Undeveloped: Number of embryos showing no development
* Intersexes: Number of individuals intersexes or ambiguous for sex phenotype
* Males: Number of individuals indentified as males
* Females: Number of individuals indentified as females
* Sexed: Number of sexed individuals
* Clutch: Identity of clutch
* Reference: Bibliographic reference
* Note: Diverse information for this incubation
* Version: Date of the version of this database
Initial version of this database has been built by Maria Sousa Martins.

Usage

`DatabaseTSD`

Format

A dataframe with raw data.

Details

Database of TSD information for marine turtles

Author(s)

Marc Girondot `<marc.girondot@u-psud.fr>`

References

Binckley C.A., Spotila J.R., Wilson K.S. and Paladino F.V. (1998) Sex determination and sex ratios of Pacific Leatherback Turtles, Dermochelys coriacea. Copeia, 1998(2), 291-300.

Castheloge, V.D., de D. dos Santos, M.R., de Castilhos, J.C., de J. Filho, P.R., de C. Gomes, L., Clemente-Carvalho, R.B.G., Ferreira, P.D., 2018. Pivotal temperature and hatcHling sex ratio of olive ridley sea turtles Lepidochelys olivacea from the South Atlantic coast of Brazil. Herpetological Conservation and Biology 13, 488-496.

Crastz F. (1982) Embryological stages of the marine turtle Lepidochelys olivacea (Eschscholtz). Rev. Biol. Trop., 30, 113-120.

De Souza, R.R., Vogt, R.C., 1994. Incubation temperature influences sex and hatchling size in the neotropical turtle Podocnemis unifilis. Journal of Herpetology 28, 453-464.

Dimond, M.T., 1985. Some effects of temperature on turtle egg incubation, in: Goel, S.C., Srivastava, C.B.L. (Eds.), Recent Advances in Developmental Biology of Animals. Indian Society of Developmental Biologists, Poona, India, pp. 35-39.

Emer, S.A., 2007. Temperature-dependent sex determination in Manouria emys emys, the asian forest tortoise, College of Arts and Sciences. Georgia State University, Atlanta, GE, p. 60.

Georges A., Limpus C.J. and Stoutjesdijk R. (1994) Hatchling sex in the marine turtle Caretta caretta is determined by proportion of development at a temperature, not daily duration of exposure. Journal of Experimental Zoology, 270(5), 432-444.

Girondot M. 1999 Statistical description of temperature-dependent sex determination using maximum likelihood. Evolutionary Ecology Research, 1(3), 479-486.

Gu, H.-X., Xia, Z.-R., Li, P.-P., Duan, J.-X., Ye, M.-B., Zhang, F.-Y., Chen, H.-L., Lin, R.-J., 2010. Sex identification and pivotal temperature for sex determination in hatchling green sea turtle. Chinese J. Zool. 45, 81-88.

Gutzke, W. H. N., & Paukstis, G. L. (1983). Influence of the hydric environment on sexual differentiation in turtles. J. Exp. Zool., 226, 467-469. [Chrysemys picta]

Harry, J.L., Limpus, C.J., 1989. Low temperature protection of marine turtle eggs during long-distance relocation. Aust. Wildl. Res. 16.

Hernández-Montoya V, Páez VP, Ceballos CP (2017) Effects of temperature on sex determination and embryonic development in the red-footed tortoise, Chelonoidis carbonarius. Chelonian Conservation and Biology 16:164-171

Hewavisenthi, S. (1999) Influence of incubation environment on the development of the flatback turtle (Natator depressus). Ph.D. Thesis, Central Queensland University.

Hewavisenthi, S. & Parmenter, C.J. (2000) Hydric environment and sex determination in the flatback turtle (Natator depressus Garman) (Chelonia : Cheloniidae). Australian Journal of Zoology, 48, 653-659.

Hulin, V., Delmas, V., Girondot, M., Godfrey, M. H. and Guillon, J.-M. 2009. Temperature-dependent sex determination and global change: are some species at greater risk? Oecologia, 160, 493-506.

LeBlanc, A.M., Wibbels, T., Shaver, D., Walker, J.S., 2012. Temperature-dependent sex determination in the Kemp's ridley sea turtle: effects of incubation temperatures on sex ratios. Endangered Species Research 19, 123-128.

Limpus C.J., Reed P.C. and Miller J.D. (1985) Temperature dependent sex determination in Queensland sea turtles: Intraspecific variation in Caretta caretta. In Grigg G., Shine R. and Ehmann H. (eds) Biology of Australian frogs and reptiles. New South Wales, Australia: Royal Zoological Society, pp 343-351.

Limpus C.J., Reed P. and Miller J.D. (1983) Islands and turtles: the influence of choice of nesting beach on sex ratio. In Baker J.J., Carter R.M., Saurmarco P.W. and Stark K.P. (eds.) Proceedings of the Inaugural Great Barrier Reef Conference, 29 August-2 September, 1983. James Cook University Press of North Queensland, pp. 397-402.

López Correa, J.Y., 2010. Diferenciacion gonadica en crias de Lepidochelys olivacea (Eschscholtz, 1829) (Testudinata: Cheloniidae), Instituto Politicnico Nacional, Centro Interdisciplinaria de Ciencias Marinas, La Paz, BCS, p. 108.

Lubiana, A.l., Júnior, P.D.F., 2009. Pivotal temperature and sexual dimorphism of Podocnemis expansa hatchlings (Testudines: Podocnemididae) from Bananal Island, Brazil. Zoologia 26, 527-533.

Marcovaldi M.A., Godfrey M.H. and Mrosovsky N. (1997) Estimating sex ratios of loggerhead turtles in Brazil from pivotal incubation temperatures. Canadian Journal of Zoology-Revue Canadienne De Zoologie, 75, 755-770.

McCoy C.J., Vogt R.C. and Censky E.J. (1983) Temperature-controlled sex determination in the sea turtle Lepidochelys olivacea. J. Herpetol., 17(4), 404-406.

Merchant-Larios, H, Diaz-Hernandez, V, Marmolejo-Valencia A. 2010. Gonadal morphogenesis and gene expression in reptiles with temperature-dependent sex determination. Sexual Development 4:50-61.

Merchant-Larios, H., Villalpando-Fierro, I. & Centeno-Urruiza, B. 1989. Gonadal morphogenesis under controlled temperature in the sea turtle Lepidochelys olivacea. Herpetol. Monographs, 3, 43-61.

Merchant-Larios, H., Ruiz-Ramirez, S., Moreno-Mendoza, N. & Marmolejo-Valencia, A. 1997. Correlation among thermosensitive period, estradiol response, and gonad differentiation in the sea turtle Lepidochelys olivacea. General and Comparative Endocrinology, 107, 373-385.

Michel-Morfinu J.E., Gomez Munoz V.M. and Navarro Rodriguez C. (2001) Morphometric model for sex assessment in hatchling olive ridley sea turtles. Chelonian Conservation and Biology, 40(1), 53-58.

Mohanty-Hejmadi, P., Behra, M., Dimond, M.T., 1985. Temperature dependent sex differentiation in the olive ridley Lepidochelys olivacea and its implications for conservation, Symposium on Endangered Marine Animals and Marine Parks. Marine Biological Association of India, Cochin., pp. 1-5.

Mrosovsky N., Kamel S., Rees A.F. and Margaritoulis D. (2002) Pivotal temperature for loggerhead turtles (Caretta caretta) from Kyparissia Bay, Greece. Canadian Journal of Zoology-Revue Canadienne De Zoologie, 80(12), 2118-2124.

Mrosovsky N., Dutton P.H. and Whitmore C.P. (1984) Sex ratios of two species of sea turtle nesting in Suriname. Can. J. Zool., 62, 2227-2239.

Mrosovsky N., Bass A., Corliss L.A., Richardson J.I. and Richardson T.H. (1992) Pivotal and beach temperature for hawksbill turtles nesting in Antigua. Can. J. Zool., 70, 1920-1925.

Rimblot F., Fretey J., Mrosovsky N., Lescure J. and Pieau C. (1985) Sexual differentiation as a function of the incubation temperature of eggs in the sea-turtle Dermochelys coriacea (Vandelli, 1761). Amphibia-Reptilia, 85(6), 83-92.

Rimblot-Baly F., Lescure J., Fretey J. and Pieau C. (1986-1987) Sensibilité à la température de la differenciation sexuelle chez la tortue Luth, Dermochelys coriacea (Vandelli, 1761); application des données de l'incubation artificielle à l'étude de la sex-ratio dans la nature. Annales des Sciences Naturelles, Zoologie, 8, 277-290.

Ruiz Garcia N.A. (2014) Efectos de la temperatura sobre el desarrollo embrionario y el desempeño de crías de la tortuga golfina, Lepidochelys olivacea. Tesis Nivel Maestria.

Shaver, D.J., 2007. Padre Island National Seashore, Texas sea turtle nesting and stranding 2006 report. U.S. Department of the Interior, National Park Service Publication, p. 30.

Shaver, D.J., 2008. Padre Island National Seashore, Texas sea turtle nesting and stranding 2007 report. U.S. Department of the Interior, National Park Service Publication, p. 33.

Shaver, D.J., 2009. Padre Island National Seashore, Texas sea turtle nesting and stranding 2008 report. U.S. Department of the Interior, National Park Service Publication, p. 38.

Shaver, D.J., Owens, D.W., Chaney, A.H., Caillouet, C.W., Burchfield Jr., P., Marquez, R., 1988. Styrofoam box and beach temperatures in relation to incubation and sex ratios of kemp's ridley sea turtles, in: Schroeder, B.A. (Ed.), Proceedings of the Eighth Annual Workshop on Sea Turtle

Conservation and Biology. NOAA Technical Memorandum NMFS-SEFC-214, Fort Fisher, North Carolina, USA, pp. 103-108.

Valenzuela, N., 2001. Constant, shift, and natural temperature effects on sex determination in Podocnemis expansa. Ecology 82, 3010-3024.

Tokunaga, S., Iwakiri, Y., Nakajima, Y., (1999). Temperature-dependent sex determination of a sea turtle, Caretta caretta, from Miyazaki,Japan. Bull. Kitakyushu Mus. Nat. Hist. 18, 147-156.

Wibbels T., Rostal D.C. and Byles R. (1998) High pivotal temperature in the sex determination of the olive ridley sea turtle, Lepidochelys olivacea, from Playa Nancite, Costa Rica. Copeia, 1998(4), 1086-1088.

Yntema C.L. and Mrosovsky N. (1980) Sexual differentiation in hatchling loggerheads (Caretta caretta) incubated at different controlled temperatures. Herpetologica, 36(1), 33-36.

Yntema C.L. and Mrosovsky N. (1982) Critical periods and pivotal temperatures for sexual differentiation in loggerhead sea turtles. Canadian Journal of Zoology-Revue Canadienne De Zoologie, 60(5), 1012-1016.

See Also

Other Functions for temperature-dependent sex determination: `P_TRT`, `TSP.list`, `plot.tsd`, `predict.tsd`, `stages`, `tsd_MHmcmc_p`, `tsd_MHmcmc`, `tsd`

Examples

```
## Not run:
library(embryogrowth)
data(DatabaseTSD)
DatabaseTSD$Version[1]
totalIncubation_Lo <- subset(DatabaseTSD,
         Species=="Lepidochelys olivacea" & (!is.na(Sexed) & Sexed!=0),
         select=c("Males", "Females", "Incubation.temperature"))
tot_Lo <- with(totalIncubation_Lo, tsd(males=Males, females=Females,
 temperatures=Incubation.temperature), parameters.initial = c(P=30.5, S=-0.4))
 predict(tot_Lo)

## End(Not run)
```

`dydt.exponential` *Return the derivative of the exponential function*

Description

Return the derivative of the exponential function
dydt.exponential(t, size, parms)

Usage

```
dydt.exponential(t, size, parms)
```

Arguments

`t`	The time in any unit
`size`	The current size
`parms`	A vector with alpha and K values being c(alpha=x1, K=x2). K is not used.

Details

dydt.exponential returns the derivative of the exponential function.

Value

A list with the derivative

Author(s)

Marc Girondot

Examples

```
## Not run:
library(embryogrowth)
data(nest)
formated <- FormatNests(nest)
# The initial parameters value can be:
# "T12H", "DHA",  "DHH", "Rho25"
# Or
# "T12L", "DT", "DHA",  "DHH", "DHL", "Rho25"
x <- structure(c(306.174998729436, 333.708348843241,
299.856306141849, 149.046870203155),
.Names = c("DHA", "DHH", "T12H", "Rho25"))
# K or rK are not used for dydt.linear or dydt.exponential
resultNest_4p_exponential <- searchR(parameters=x, fixed.parameters=NULL,
temperatures=formated, derivate=dydt.exponential, M0=1.7,
test=c(Mean=39.33, SD=1.92))

## End(Not run)
```

`dydt.Gompertz` *Return the derivative of the Gompertz function*

Description

Return the derivative of the Gompertz function
dydt.Gompertz(t, size, parms)

Usage

```
dydt.Gompertz(t, size, parms)
```

Arguments

`t`	The time in any unit
`size`	The current size
`parms`	A vector with alpha and K values being c(alpha=x1, K=x2)

Details

dydt.Gompertz returns the derivative of the Gompertz function.

Value

A list with the derivative

Author(s)

Marc Girondot

Examples

```
## Not run:
library(embryogrowth)
data(nest)
formated <- FormatNests(nest)
# The initial parameters value can be:
# "T12H", "DHA",  "DHH", "Rho25"
# Or
# "T12L", "DT", "DHA",  "DHH", "DHL", "Rho25"
x <- structure(c(118.768297442004, 475.750095909406, 306.243694918151,
116.055824800264), .Names = c("DHA", "DHH", "T12H", "Rho25"))
# pfixed <- c(K=82.33) or rK=82.33/39.33
pfixed <- c(rK=2.093313)
# K or rK are not used for dydt.linear or dydt.exponential
resultNest_4p_SSM4p <- searchR(parameters=x, fixed.parameters=pfixed,
temperatures=formated, derivate=dydt.Gompertz, M0=1.7,
test=c(Mean=39.33, SD=1.92))
data(resultNest_4p_SSM4p)

## End(Not run)
```

`dydt.linear` *Return the derivative of the linear function*

Description

Return the derivative of the linear function
dydt.linear(t, size, parms)

Usage

```
dydt.linear(t, size, parms)
```

Arguments

`t`	The time in any unit
`size`	The current size
`parms`	A vector with alpha being c(alpha=x1, K=x2). Only alpha is used.

Details

dydt.Linear returns the derivative of the linear function.

Value

A list with the derivative

Author(s)

Marc Girondot

Examples

```
## Not run:
library(embryogrowth)
data(nest)
formated <- FormatNests(nest)
# The initial parameters value can be:
# "T12H", "DHA",  "DHH", "Rho25"
# Or
# "T12L", "DT", "DHA",  "DHH", "DHL", "Rho25"
x <- structure(c(306.174998729436, 333.708348843241, 299.856306141849,
149.046870203155), .Names = c("DHA", "DHH", "T12H", "Rho25"))
# K or rK are not used for dydt.linear or dydt.exponential
resultNest_4p_linear <- searchR(parameters=x, fixed.parameters=NULL,
temperatures=formated, derivate=dydt.linear, M0=1.7,
test=c(Mean=39.33, SD=1.92))

## End(Not run)
```

`FormatNests` *Create a dataset of class Nests to be used with searchR*

Description

Will create a dataset of class Nests to be used with searchR
FormatNests(nest, previous=x) with x being a previously formated data.
The raw data must be organized being:
First column is the time in minutes since the beginning of incubation
Each column next is the trace of temperatures, one column for each nest.
For example, for two nests:
Time Nest1 Nest2
0 29.8 27.6
90 30.2 28.8
120 30.4 30.7
180 31.2 32.6
...
65800 30.8 32.6
65890 30.2
65950 30.4

The Nest1 ends incubation at 65800 minutes whereas Nest2 ends incubation at 65950 (last row with temperature for each).
The parameter Weight is a vector: weight=c(Nest1=1, Nest2=1.2)
It can be used to format database already formated with old format; in this case, just use data=xxx with xxx being the old format database.

Usage

```
FormatNests(data = stop("A dataset must be provided !"),
  previous = NULL, simplify = TRUE, weight = NULL)
```

Arguments

`data`	Data to be newly formated
`previous`	Data already formated
`simplify`	If TRUE, simply the time series by removing identical time series of temperatures
`weight`	Named vector with weight for likelihhod

Details

FormatNests creates a dataset of class "Nests" to be used with searchR

Value

A list with all the nests formated to be used with searchR.

Author(s)

Marc Girondot

Examples

```
## Not run:
library(embryogrowth)
data(nest)
formated <- FormatNests(nest, previous=NULL)
formated <- FormatNests(nest)

## End(Not run)
```

`GenerateAnchor` *Generate a set of anchored parameters*

Description

Generate a set of anchored parameters.
It is important that the anchors (i.e. the temperatures used as anchors) encompass the highest and lowest temperatures that are present in nests.
The value for each anchor is R * 1E5. The 1E5 factor allows to value to be close to unity.

Usage

```
GenerateAnchor(temperatures = NULL, nests = NULL, parameters = NULL,
  number.anchors = 7)
```

Arguments

`temperatures`	A vector with temperatures to serve as anchors
`nests`	Formated nest data or result object obtained from searchR()
`parameters`	A set of parameters value
`number.anchors`	Number of anchors

Details

GenerateAnchor Generate a set of anchored parameters

Value

A vector with parameters

Author(s)

Marc Girondot

Examples

```
## Not run:
# Example to generate anchored parameters
newp <- GenerateAnchor()
newp <- GenerateAnchor(temperatures=seq(from=20,
  to=35, length.out=7))
newp <- GenerateAnchor(number.anchors=7)
data(nest)
formated <- FormatNests(nest, previous=NULL)
newp <- GenerateAnchor(nests=formated)
newp <- GenerateAnchor(nests=formated, number.anchors=10)
data(resultNest_4p_SSM4p)
newp <- GenerateAnchor(nests=resultNest_4p_SSM4p, number.anchors=7)
newp <- GenerateAnchor(nests=resultNest_4p_SSM4p, temperatures=seq(from=20,
 to=35, length.out=10))
newp <- GenerateAnchor(nests=resultNest_4p_SSM4p, number.anchors=7)
newp <- c(newp, Scale=1)

## End(Not run)
```

`GenerateConstInc` *Generate a data.frame with constant incubation temperature and incubation duration*

Description

Generate a data.frame from constant incubation temperature and incubation duration

Usage

```
GenerateConstInc(durations = stop("At least one incubation length must be provided"),
 temperatures = stop("At least one incubation temperature must be provided"),
 names = NULL)
```

Arguments

`durations`	A vector with incubation durations
`temperatures`	A vector with incubation temperatures
`names`	A vector of column names

Details

GenerateConstInc generates a data.frame with constant incubation temperature and incubation duration

Value

A date.frame that can be used with FormatNests()

Author(s)

Marc Girondot

Examples

```
## Not run:
temp_cst <- GenerateConstInc(durations=c(150000, 100100, 100000),
temperatures=c(28, 30.5, 30.6),
names=c("T28", "T30.5", "T30.6"))

## End(Not run)
```

`GenerateTest` *Generate a data.frame that can be used as test value for searchR()*

Description

Generate a data.frame that can be used as test value for searchR()

Usage

```
GenerateTest(series = stop("A result object or names of series must be provided"),
 size = NULL, previous = NULL)
```

Arguments

`series`	Name of series or object from searchR()
`size`	Size or mass at hatching. Will be recycled if necessary
`previous`	Previous formated test data

Details

GenerateTest Generate a data.frame that can be used as test value for searchR()

Value

A data.frame with size or mass at hatching for each nest

Author(s)

Marc Girondot

Examples

```
## Not run:
library(embryogrowth)
data(resultNest_4p_SSM4p)
testsize1 <- GenerateTest(resultNest_4p_SSM4p)
testsize2 <- GenerateTest(series=resultNest_4p_SSM4p,
size=c(Mean=39.3, SD=1.92))

## End(Not run)
```

`GRTRN_MHmcmc` *Metropolis-Hastings algorithm for Embryo Growth Rate Thermal Reaction Norm*

Description

Run the Metropolis-Hastings algorithm for data.
The number of iterations is `n.iter+n.adapt+1` because the initial likelihood is also displayed.
I recommend that thin=1 because the method to estimate SE uses resampling.
If initial point is maximum likelihood, n.adapt = 0 is a good solution.
To get the SE of the point estimates from `result_mcmc <-GRTRN_MHmcmc(result=try)`, use:
`result_mcmc$SD`
coda package is necessary for this function.
The parameters `intermediate` and `filename` are used to save intermediate results every 'intermediate' iterations (for example 1000). Results are saved in a file named `filename`.
The parameter previous is used to indicate the list that has been save using the parameters intermediate and filename. It permits to continue a mcmc search.
These options are used to prevent the consequences of computer crash or if the run is very very long and processes with user limited time.

Usage

```
GRTRN_MHmcmc(result = NULL, n.iter = 10000, parametersMCMC = NULL,
  n.chains = 1, n.adapt = 0, thin = 1, trace = NULL,
  traceML = FALSE, parallel = TRUE, adaptive = FALSE,
  adaptive.lag = 500, adaptive.fun = function(x) {     ifelse(x >
  0.234, 1.3, 0.7) }, intermediate = NULL,
  filename = "intermediate.Rdata", previous = NULL)
```

Arguments

`result`	An object obtained after a SearchR fit
`n.iter`	Number of iterations for each step
`parametersMCMC`	A set of parameters used as initial point for searching with information on priors
`n.chains`	Number of replicates

`n.adapt`	Number of iterations before to store outputs
`thin`	Number of iterations between each stored output
`trace`	TRUE or FALSE or period, shows progress
`traceML`	TRUE or FALSE to show ML
`parallel`	If true, try to use several cores using parallel computing
`adaptive`	Should an adaptive process for SDProp be used
`adaptive.lag`	Lag to analyze the SDProp value in an adaptive content
`adaptive.fun`	Function used to change the SDProp
`intermediate`	Period for saving intermediate result, NULL for no save
`filename`	If intermediate is not NULL, save intermediate result in this file
`previous`	Previous result to be continued. Can be the filename in which intermediate results are saved.

Details

GRTRN_MHmcmc runs the Metropolis-Hastings algorithm for data (Bayesian MCMC)

Value

A list with resultMCMC being mcmc.list object, resultLnL being likelihoods and parametersMCMC being the parameters used

Author(s)

Marc Girondot

Examples

```
## Not run:
library(embryogrowth)
data(nest)
formated <- FormatNests(nest)
# The initial parameters value can be:
# "T12H", "DHA",  "DHH", "Rho25"
# Or
# "T12L", "T12H", "DHA",  "DHH", "DHL", "Rho25"
x <- structure(c(118.768297442004, 475.750095909406, 306.243694918151,
116.055824800264), .Names = c("DHA", "DHH", "T12H", "Rho25"))
# pfixed <- c(K=82.33) or rK=82.33/39.33
pfixed <- c(rK=2.093313)
resultNest_4p_SSM4p <- searchR(parameters=x, fixed.parameters=pfixed,
temperatures=formated, derivate=dydt.Gompertz, M0=1.7,
test=c(Mean=39.33, SD=1.92))
data(resultNest_4p_SSM4p)
pMCMC <- TRN_MHmcmc_p(resultNest_4p_SSM4p, accept=TRUE)
# Take care, it can be very long; several days
resultNest_mcmc_4p_SSM4p <- GRTRN_MHmcmc(result=resultNest_4p_SSM4p,
   adaptive = TRUE,
```

```
parametersMCMC=pMCMC, n.iter=10000, n.chains = 1,
n.adapt = 0, thin=1, trace=TRUE)
data(resultNest_mcmc_4p_SSM4p)
out <- as.mcmc(resultNest_mcmc_4p_SSM4p)
# This out can be used with coda package
# Test for stationarity and length of chain
require(coda)
heidel.diag(out)
raftery.diag(out)
# plot() can use the direct output of GRTRN_MHmcmc() function.
plot(resultNest_mcmc_4p_SSM4p, parameters=1, xlim=c(0,550))
plot(resultNest_mcmc_4p_SSM4p, parameters=3, xlim=c(290,320))
# summary() permits to get rapidly the standard errors for parameters
# They are store in the result also.
se <- result_mcmc_4p_SSM4p$SD
# the confidence interval is better estimated by:
apply(out[[1]], 2, quantile, probs=c(0.025, 0.975))
# The use of the intermediate method is as followed;
# Here the total mcmc iteration is 10000, but every 1000, intermediate
# results are saved in file intermediate1000.Rdata:
resultNest_mcmc_4p_SSM4p <- GRTRN_MHmcmc(result=resultNest_4p_SSM4p,
parametersMCMC=pMCMC, n.iter=10000, n.chains = 1,
n.adapt = 0, thin=1, trace=TRUE,
intermediate=1000, filename="intermediate1000.Rdata")
# If run has been stopped for any reason, it can be resumed with:
resultNest_mcmc_4p_SSM4p <- GRTRN_MHmcmc(previous="intermediate1000.Rdata")
# Example to use of the epsilon parameter to get confidence level
resultNest_4p_epsilon <- resultNest_4p
resultNest_4p_epsilon$fixed.parameters <- c(resultNest_4p_epsilon$par,
                   resultNest_4p_epsilon$fixed.parameters)
resultNest_4p_epsilon$par <- c(epsilon = 0)
pMCMC <- TRN_MHmcmc_p(resultNest_4p_epsilon, accept = TRUE)
resultNest_mcmc_4p_epsilon <- GRTRN_MHmcmc(result = resultNest_4p_epsilon,
           n.iter = 10000, parametersMCMC = pMCMC,
           n.chains = 1, n.adapt = 0, thin = 1, trace = TRUE, parallel = TRUE)
data(resultNest_mcmc_4p_epsilon)
plot(resultNest_mcmc_4p_epsilon, parameters="epsilon", xlim=c(-11, 11), las=1)
plotR(resultNest_4p_epsilon, SE=c(epsilon = unname(resultNest_mcmc_4p_epsilon$SD)),
           ylim=c(0, 3), las=1)

## End(Not run)
```

`HatchingSuccess.fit` *Fit a hatching success model to data using maximum likelihood*

Description

Set of functions to study the hatching success.

Usage

```
HatchingSuccess.fit(par, data, fixed.parameters = NULL,
  column.Incubation.temperature = "Incubation.temperature",
  column.Hatched = "Hatched", column.NotHatched = "NotHatched")
```

Arguments

`par`	A set of parameters.
`data`	A dataset in a data.frame with a least three columns: Incubation.temperature, Hatched and NotHatched
`fixed.parameters`	A set of parameters that must not be fitted.
`column.Incubation.temperature`	Name of the column with incubation temperatures
`column.Hatched`	Name of the column with hatched number
`column.NotHatched`	Name of the column with not hatched number

Details

HatchingSuccess.fit fits a hatching success model to data

Value

Return a object of class HatchingSuccess

Author(s)

Marc Girondot

See Also

Other Hatching success: `HatchingSuccess.MHmcmc_p`, `HatchingSuccess.MHmcmc`, `HatchingSuccess.lnL`, `HatchingSuccess.model`, `logLik.HatchingSuccess`, `nobs.HatchingSuccess`, `plot.HatchingSuccess`, `predict.HatchingSuccess`

Examples

```
## Not run:
library(embryogrowth)
totalIncubation_Cc <- subset(DatabaseTSD,
                             Species=="Caretta caretta" &
                               Note != "Sinusoidal pattern" &
                               !is.na(Total) & Total != 0)

par <- c(S.low=0.5, S.high=0.3,
         P.low=25, deltaP=10, MaxHS=logit(0.8))

HatchingSuccess.lnL(par=par, data=totalIncubation_Cc)
```

```
g <- HatchingSuccess.fit(par=par, data=totalIncubation_Cc)

HatchingSuccess.lnL(par=g$par, data=totalIncubation_Cc)

par(mar=c(4, 4, 1, 1), +0.4)
plot(g)

## End(Not run)
```

`HatchingSuccess.lnL` *Return -log likelihood of the data and the parameters*

Description

Set of functions to study the hatching success.

Usage

```
HatchingSuccess.lnL(par, data, fixed.parameters = NULL,
  column.Incubation.temperature = "Incubation.temperature",
  column.Hatched = "Hatched", column.NotHatched = "NotHatched")
```

Arguments

`par`	A set of parameters.
`data`	A dataset in a data.frame with a least three columns: Incubation.temperature, Hatched and NotHatched
`fixed.parameters`	A set of parameters that must not be fitted.
`column.Incubation.temperature`	Name of the column with incubation temperatures
`column.Hatched`	Name of the column with hatched number
`column.NotHatched`	Name of the column with not hatched number

Details

HatchingSuccess.lnL return -log likelihood of the data and the parameters

Value

Return -log likelihood of the data and the parameters

Author(s)

Marc Girondot

See Also

Other Hatching success: `HatchingSuccess.MHmcmc_p`, `HatchingSuccess.MHmcmc`, `HatchingSuccess.fit`, `HatchingSuccess.model`, `logLik.HatchingSuccess`, `nobs.HatchingSuccess`, `plot.HatchingSuccess`, `predict.HatchingSuccess`

Examples

```
## Not run:
library(embryogrowth)
totalIncubation_Cc <- subset(DatabaseTSD,
                             Species=="Caretta caretta" &
                               Note != "Sinusoidal pattern" &
                               !is.na(Total) & Total != 0)

par <- c(S.low=0.5, S.high=0.3,
         P.low=25, deltaP=10, MaxHS=logit(0.8))

HatchingSuccess.lnL(par=par, data=totalIncubation_Cc)

g <- HatchingSuccess.fit(par=par, data=totalIncubation_Cc)

HatchingSuccess.lnL(par=g$par, data=totalIncubation_Cc)

t <- seq(from=20, to=40, by=0.1)
CIq <- predict(g, temperature=t)

par(mar=c(4, 4, 1, 1), +0.4)
plot(g)

## End(Not run)
```

`HatchingSuccess.MHmcmc`

Metropolis-Hastings algorithm for hatching success

Description

Run the Metropolis-Hastings algorithm for hatching success.
The number of iterations is n.iter+n.adapt+1 because the initial likelihood is also displayed.
I recommend that thin=1 because the method to estimate SE uses resampling.
If initial point is maximum likelihood, n.adapt = 0 is a good solution.
To get the SE from result_mcmc <- HatchingSuccesss.MHmcmc(result=try), use:
result_mcmc$BatchSE or result_mcmc$TimeSeriesSE
The batch standard error procedure is usually thought to be not as accurate as the time series methods.
Based on Jones, Haran, Caffo and Neath (2005), the batch size should be equal to sqrt(n.iter).
Jones, G.L., Haran, M., Caffo, B.S. and Neath, R. (2006) Fixed Width Output Analysis for Markov chain Monte Carlo , Journal of the American Statistical Association, 101:1537-1547.

coda package is necessary for this function.
The parameters intermediate and filename are used to save intermediate results every 'intermediate' iterations (for example 1000). Results are saved in a file of name filename.
The parameter previous is used to indicate the list that has been save using the parameters intermediate and filename. It permits to continue a mcmc search.
These options are used to prevent the consequences of computer crash or if the run is very very long and processes at time limited.

Usage

```
HatchingSuccess.MHmcmc(result = stop("Give a result of HatchingSuccesss.fit()"),
 n.iter = 10000, parametersMCMC = NULL, n.chains = 1, n.adapt = 0,
 thin = 1, trace = FALSE, traceML = FALSE,
 batchSize = sqrt(n.iter), adaptive = FALSE, adaptive.lag = 500,
 adaptive.fun = function(x) {     ifelse(x > 0.234, 1.3, 0.7) },
 intermediate = NULL, filename = "intermediate.Rdata",
 previous = NULL)
```

Arguments

`result`	An object obtained after a SearchR fit
`n.iter`	Number of iterations for each step
`parametersMCMC`	A set of parameters used as initial point for searching with information on priors
`n.chains`	Number of replicates
`n.adapt`	Number of iterations before to store outputs
`thin`	Number of iterations between each stored output
`trace`	TRUE or FALSE or period, shows progress
`traceML`	TRUE or FALSE to show ML
`batchSize`	Number of observations to include in each batch fo SE estimation
`adaptive`	Should an adaptive process for SDProp be used
`adaptive.lag`	Lag to analyze the SDProp value in an adaptive content
`adaptive.fun`	Function used to change the SDProp
`intermediate`	Period for saving intermediate result, NULL for no save
`filename`	If intermediate is not NULL, save intermediate result in this file
`previous`	Previous result to be continued. Can be the filename in which intermediate results are saved.

Details

HatchingSuccess.MHmcmc runs the Metropolis-Hastings algorithm for hatching success (Bayesian MCMC)

Value

A list with resultMCMC being mcmc.list object, resultLnL being likelihoods and parametersMCMC being the parameters used

Author(s)

Marc Girondot

See Also

Other Hatching success: `HatchingSuccess.MHmcmc_p`, `HatchingSuccess.fit`, `HatchingSuccess.lnL`, `HatchingSuccess.model`, `logLik.HatchingSuccess`, `nobs.HatchingSuccess`, `plot.HatchingSuccess`, `predict.HatchingSuccess`

Examples

```
## Not run:
library(embryogrowth)
totalIncubation_Cc <- subset(DatabaseTSD,
                             Species=="Caretta caretta" &
                               Note != "Sinusoidal pattern" &
                               !is.na(Total) & Total != 0)

par <- c(S.low=0.5, S.high=0.3,
         P.low=25, deltaP=10, MaxHS=logit(0.8))

g <- HatchingSuccess.fit(par=par, data=totalIncubation_Cc)
pMCMC <- HatchingSuccess.MHmcmc_p(g, accept=TRUE)
mcmc <- HatchingSuccess.MHmcmc(result=g, parameters = pMCMC,
                  adaptive=TRUE, n.iter=100000, trace=1000)

## End(Not run)
```

`HatchingSuccess.MHmcmc_p` *Generates set of parameters to be used with HatchingSuccess.MHmcmc()*

Description

Interactive script used to generate set of parameters to be used with HatchingSuccess.MHmcmc().

Usage

```
HatchingSuccess.MHmcmc_p(result = NULL, parameters = NULL,
  fixed.parameters = NULL, accept = FALSE)
```

Arguments

`result`	An object obtained after a HatchingSuccess.fit() fit
`parameters`	A set of parameters. Replace the one from result
`fixed.parameters`	A set of fixed parameters. Replace the one from result
`accept`	If TRUE, the script does not wait user information

Details

HatchingSuccess.MHmcmc_p generates set of parameters to be used with HatchingSuccess.MHmcmc()

Value

A matrix with the parameters

Author(s)

Marc Girondot

See Also

Other Hatching success: `HatchingSuccess.MHmcmc`, `HatchingSuccess.fit`, `HatchingSuccess.lnL`, `HatchingSuccess.model`, `logLik.HatchingSuccess`, `nobs.HatchingSuccess`, `plot.HatchingSuccess`, `predict.HatchingSuccess`

Examples

```
## Not run:
library(embryogrowth)
totalIncubation_Cc <- subset(DatabaseTSD,
                             Species=="Caretta caretta" &
                               Note != "Sinusoidal pattern" &
                               !is.na(Total) & Total != 0)

par <- c(S.low=0.5, S.high=0.3,
         P.low=25, deltaP=10, MaxHS=logit(0.8))

g <- HatchingSuccess.fit(par=par, data=totalIncubation_Cc)
pMCMC <- HatchingSuccess.MHmcmc_p(g, accept=TRUE)
mcmc <- HatchingSuccesss.MHmcmc(result=g, parameters = pMCMC,
                  adaptive=TRUE, n.iter=100000, trace=1000)

## End(Not run)
```

`HatchingSuccess.model` *Return the hatching success according the set of parameters and temperatures*

Description

Set of functions to study the hatching success.

Usage

```
HatchingSuccess.model(par, temperature)
```

Arguments

`par`	A set of parameters.
`temperature`	A vector of temperatures.

Details

HatchingSuccess.model returns the hatching success according the set of parameters and temperatures

Value

Return the hatching success according the set of parameters and temperatures

Author(s)

Marc Girondot

See Also

Other Hatching success: `HatchingSuccess.MHmcmc_p`, `HatchingSuccess.MHmcmc`, `HatchingSuccess.fit`, `HatchingSuccess.lnL`, `logLik.HatchingSuccess`, `nobs.HatchingSuccess`, `plot.HatchingSuccess`, `predict.HatchingSuccess`

Examples

```
## Not run:
library(embryogrowth)
totalIncubation_Cc <- subset(DatabaseTSD,
                             Species=="Caretta caretta" &
                               Note != "Sinusoidal pattern" &
                               !is.na(Total) & Total != 0)

par <- c(S.low=0.5, S.high=0.3,
         P.low=25, deltaP=10, MaxHS=logit(0.8))
```

```
HatchingSuccess.lnL(par=par, data=totalIncubation_Cc)

g <- HatchingSuccess.fit(par=par, data=totalIncubation_Cc)

HatchingSuccess.lnL(par=g$par, data=totalIncubation_Cc)

plot(g)

## End(Not run)
```

`hist.Nests` *Show the histogram of temperatures with set of nests*

Description

Show the histogram of temperatures with set of nests hist(data)

Usage

```
## S3 method for class 'Nests'
hist(x, series = "all", ...)
```

Arguments

`x`	Data formated using formatdata.
`series`	Series to be used, logical (TRUE ou FALSE), numbers or names. If "all", all series are used.
`...`	Parameters used by hist function

Details

hist.Nests shows the histogram of temperatures with set of nests

Value

A list with an histogram object with information on histogram or NULL if no series was selected and the complete set of temperatures used.

Author(s)

Marc Girondot `<marc.girondot@u-psud.fr>`

Examples

```
## Not run:
library(embryogrowth)
data(nest)
formated <- FormatNests(nest)
h <- hist(formated, series="all")

## End(Not run)
```

`hist.NestsResult` *Show the histogram of temperatures with set of nests*

Description

Show the histogram of temperatures with set of nests hist(data)

Usage

```
## S3 method for class 'NestsResult'
hist(x, series = "all", ...)
```

Arguments

`x`	Results obtained after searchR
`series`	Series to be used, logical (TRUE ou FALSE), numbers or names. If "all", all series are used.
`...`	Parameters used by hist function (example main="Title")

Details

hist.NestsResult shows the histogram of temperatures with set of nests

Value

A list with an histogram object with information on histogram or NULL if no series was selected and the complete set of temperatures used.

Author(s)

Marc Girondot `<marc.girondot@u-psud.fr>`

Examples

```
## Not run:
library(embryogrowth)
data(resultNest_4p_SSM4p)
h <- hist(resultNest_4p_SSM4p, series=c(1:5))

## End(Not run)
```

`info.nests` *Calculte statistics about nests*

Description

This function calculates many statistics about nests.
The embryo.stages is a named vector with relative size as compared to final size at the beginning of the stage. Names are the stages.
For example for SCL in Caretta caretta:
embryo.stages=structure(c(8.4, 9.4, 13.6, 13.8, 18.9, 23.5, 32.2, 35.2, 35.5, 38.5)/39.33),
.Names = c("21", "22", "23", "24", "25", "26", "27", "28", "29", "30", "31"))
indicates that the stages 21 begins at the relative size of 8.4/39.33.
Series can be indicated as the name of the series, their numbers or series or sucession of TRUE or FALSE. "all" indicates that all series must be analyzed.
The likelihood object is just the total likelihood of the data in the model.
If one parameter is named "pipping_emergence" it is used as the number of days between pipping and emergence to calculate the 1/3 and 2/3 of incubation.
The summary object is a data.frame composed of these elements:

- `Temperature.max` Maximum temperature recorded during incubation
- `TimeWeighted.temperature.mean` Average temperature weigthed by the time at each temperature for all incubation
- `TimeWeighted.temperature.se` Standard error for the average temperature weigthed by the time at each temperature for all incubation
- `MassWeighted.temperature.mean` Average temperature weigthed by the growth of embryo for all incubation
- `MassWeighted.temperature.se` Standard error for the average temperature weigthed by the growth of embryo for all incubation
- `TSP.TimeWeighted.temperature.mean` Average temperature during the TSP weigthed by the time at each temperature
- `TSP.TimeWeighted.temperature.se` Standard error for the average temperature during the TSP weigthed by the time at each temperature
- `TSP.MassWeighted.temperature.mean` Average temperature during the TSP weigthed by the mass increase at each temperature
- `TSP.MassWeighted.temperature.se` Standard error for the average temperature during the TSP weigthed by the mass increase at each temperature
- `TSP.STRNWeighted.temperature.mean` Averaged temperature during the TSP weighted by sexualisation thermal reaction norm
- `TSP.STRNWeighted.temperature.se` Standard error for temperature during the TSP weighted by sexualisation thermal reaction norm
- `TSP.MassWeighted.STRNWeighted.temperature.mean` Averaged temperature during the TSP weighted by sexualisation thermal reaction norm and embryo growth
- `TSP.MassWeighted.STRNWeighted.temperature.se` Standard error for temperature during the TSP weighted by sexualisation thermal reaction norm and embryo growth

- `TSP.length.mean` Average length of the TSP
- `TSP.length.se` Standard error for length of the TSP
- `TSP.begin.mean` Average time of the beginning of the TSP
- `TSP.begin.se` Standard error for the beginning of the TSP
- `TSP.end.mean` Average time of the endd of the TSP
- `TSP.end.se` Standard error for the end of the TSP
- `PM.mean` Proportion of males based on Massey et al. 2019
- `PM.se` Standard error for the proportion of males based on Massey et al. 2019
- `PM.TimeWeighted.mean` Proportion of males with time effect
- `PM.TimeWeighted.se` Standard error for the proportion of males with time effect
- `Incubation.length.mean` Average number of days for the incubation length
- `Incubation.length.se` Standard error for number of days for the incubation length
- `Middlethird.begin.mean` Average day at which the middle third incubation begins
- `Middlethird.begin.se` Standard error for day at which the middle third incubation begins
- `Middlethird.end.mean` Average day at which the middle third incubation ends
- `Middlethird.end.se` Standard error at which the middle third incubation ends
- `Middlethird.length.mean` Average days for second third of incubation
- `Middlethird.length.se` Standard error for second third of incubation
- `MiddleThird.TimeWeighted.temperature.mean` Average temperature during the middle third of incubation
- `MiddleThird.TimeWeighted.temperature.se` Standard error for the temperature during the middle third of incubation
- `MiddleThird.MassWeighted.temperature.mean` Average temperature during the middle third of incubation weighted by embryo growth
- `MiddleThird.MassWeighted.temperature.se` Standard error for temperature during the middle third of incubation weighted by embryo growth

The metric object is a list composed of data.frames which have the following attributes for each element of the list:

- `time.begin.tsp` Average time at the beginning of TSP
- `time.end.tsp` Average time at the end of TSP
- `time.begin.middlethird` Average time at the beginning of the middle third of incubation
- `time.begin.middlethird` Average time at the end of the middle third of incubation
- `time.begin.tsp.se` Standard error for the time at the beginning of TSP
- `time.end.tsp.se` Standard error for the time at the end of TSP
- `time.begin.middlethird.se` Standard error for the time at the beginning of the middle third of incubation
- `time.end.middlethird.se` Standard error for the time at the end of the middle third of incubation

- `stopattest` NA if stopattest was false, TRUE if at least one incubation series was longer than hatchling size and FALSE at contrary

If you indicate new set of temperatures, you must probably also indicate new test values.
Note: four species have predefined embryo stages. embryo.stages parameter can take the values:

- `Caretta caretta.SCL`
- `Chelonia mydas.SCL`
- `Emys orbicularis.SCL`
- `Emys orbicularis.mass`
- `Podocnemis expansa.SCL`
- `Generic.ProportionDevelopment`

But remember that mass is not the best proxy to describe the growth of an embryo because it can decrease if the subtrate becomes dry.
The progress bar is based on both replicates and timeseries.

Usage

```
info.nests(x = NULL, parameters = NULL, NestsResult = NULL,
  resultmcmc = NULL, hessian = NULL, CI = NULL,
  fixed.parameters = NULL, SE = NULL, temperatures = NULL,
  derivate = NULL, test = NULL, stopattest = FALSE, M0 = NULL,
  series = "all", TSP.borders = NULL,
  embryo.stages = "Generic.ProportionDevelopment", TSP.begin = 0,
  TSP.end = 0.5, replicate.CI = 100, weight = NULL,
  out = "likelihood", fill = NULL, SexualisationTRN = NULL,
  SexualisationTRN.mcmc = NULL, SexualisationTRN.CI = NULL,
  metric.end.incubation = NA, metabolic.heating = 0,
  temperature.heterogeneity = 0, progress = FALSE, warnings = TRUE,
  parallel = TRUE, tsd = NULL)
```

Arguments

`x`	A set of parameters to model the embryo growth thermal reaction norm or a NestsResult object.
`parameters`	A set of parameters to model the embryo growth thermal reaction norm. It will replace the parameters included in NestsResult (same as x).
`NestsResult`	A NestsResult object generated by searchR to model the embryo growth thermal reaction
`resultmcmc`	A mcmc result for embryo growth thermal reaction norm
`hessian`	An hessian matrix for embryo growth thermal reaction norm. It will replace the hessian matrix included in NestResult object.
`CI`	How to estimate CI for embryo growth thermal reaction norm; can be NULL, "SE", "MCMC", or "Hessian".

`fixed.parameters`	A set of fixed parameters to model the embryo growth thermal reaction norm. It will replace the fixed parameters included in NestsResult.
`SE`	Standard error for each parameter. It will replace the SE in NestsResult. Use SE=NA to remove SE from NestResult
`temperatures`	Timeseries of temperatures formatted using formatNests(). It will replace the one in NestsResult.
`derivate`	Function used to fit embryo growth: dydt.Gompertz, dydt.exponential or dydt.linear. It will replace the one in NestsResult.
`test`	Mean and SD of size of hatchlings. It will replace the one in NestsResult.
`stopattest`	TRUE or FALSE. If TRUE, the model stops when proxy of size reached the mean test size.
`M0`	Measure of hatchling size proxi at laying date. It will replace the one in NestsResult.
`series`	The name or number of the series to be estimated.
`TSP.borders`	The limits of TSP in stages. See embryo.stages parameter.
`embryo.stages`	The embryo stages. At least TSP.borders stages must be provided to estimate TSP borders. See note.
`TSP.begin`	Where TSP begin during the stage of beginning? In relative proportion of the stage.
`TSP.end`	Where TSP begin during the stage of ending? In relative proportion of the stage.
`replicate.CI`	Number of replicates to estimate CI. If <2, no CI is estimated.
`weight`	Weights of the different nests to estimate likelihood. It will replace the one in NestsResult.
`out`	Can take the values of "likelihood" or "metric" or "summary". "metric" and "summary" can be combined: c("metric", "summary")
`fill`	Number of minutes between two records. Create new one if they do not exist. NULL does not change the time of temperature recordings.
`SexualisationTRN`	A set of parameters used to model sexualisation thermal reaction norm during TSP or a result of STRN()
`SexualisationTRN.mcmc`	A mcmc object obtained from STRN_MHmcmc() to generate variability for sexualisation thermal reaction norm during TSP
`SexualisationTRN.CI`	How to estimate CI of sexualisation thermal reaction norm. Can be NULL, "SE", "MCMC", or "Hessian".
`metric.end.incubation`	The expected metric at the end of incubation. Used to calibrate TSP size. If NULL, take the maximum Mean of the test parameter. If NA, use the actual final size. Can be a vector and is recycled if necessary.
`metabolic.heating`	Degrees Celsius to be added at the end of incubation due to metabolic heating.

`temperature.heterogeneity`	SD of heterogeneity of temperatures. Can be 2 values, sd_low and sd_high and then HelpersMG::r2norm() is used.
`progress`	If FALSE, the progress bar is not shown (useful for using with sweave or knitr)
`warnings`	If FALSE, does not show warnings
`parallel`	If TRUE use parallel version for nests estimation
`tsd`	A object from tsd() that describe the thermal react norm of sex ratio at constant temperatures

Details

Calculate statistics about nests

Value

Return or the total likelihood or a list with $metric and $summary depending on out parameter

Author(s)

Marc Girondot

Examples

```
## Not run:
library(embryogrowth)
data(resultNest_4p_SSM4p)
summary.nests <- info.nests(resultNest_4p_SSM4p, out="summary",
  embryo.stages="Caretta caretta.SCL",
  replicate.CI=100,
  resultmcmc=resultNest_mcmc_4p_SSM4p,
  CI="MCMC",
  progress=TRUE)
  # Result is in summary.nests$summary
infoall <- info.nests(resultNest_4p_SSM4p)
  # Result is a value
infoall.df <- info.nests(resultNest_4p_SSM4p, out="metric",
  embryo.stages="Caretta caretta.SCL",
  replicate.CI=100,
  resultmcmc=resultNest_mcmc_4p_SSM4p,
  CI="MCMC",
  progress=TRUE)
# Result is in summary.nests$metric
infoall.both <- info.nests(resultNest_4p_SSM4p,
  embryo.stages="Caretta caretta.SCL",
  out=c("metric", "summary"), replicate.CI=100,
  resultmcmc=resultNest_mcmc_4p_SSM4p,
  CI="MCMC",
  progress=TRUE)
# Results are in summary.nests$summary and in summary.nests$metric
infoall.both <- info.nests(resultNest_4p_SSM4p,
  embryo.stages="Caretta caretta.SCL",
```

```
  out=c("metric", "summary"), replicate.CI=100,
  CI="Hessian",
  progress=TRUE)
infoall.both <- info.nests(resultNest_4p_SSM4p,
  embryo.stages="Caretta caretta.SCL",
  out=c("metric", "summary"), replicate.CI=100,
  CI="SE",
  progress=TRUE)
# Example of use of embryo.stages and TSP.borders:
  summary.nests <- info.nests(resultNest_4p_SSM4p, out="summary",
                             embryo.stages=c("10"=0.33, "11"=0.33, "12"=0.66, "13"=0.66),
                             TSP.borders = c(10, 12),
                             replicate.CI=100,
                             progress=TRUE)

#########################################
# Sex ratio using Massey et al. method PM
#########################################

# Massey, M.D., Holt, S.M., Brooks, R.J., Rollinson, N., 2019. Measurement
# and modelling of primary sex ratios for species with temperature-dependent
# sex determination. J Exp Biol 222, 1-9.

CC_Mediterranean <- subset(DatabaseTSD, RMU=="Mediterranean" &
Species=="Caretta caretta" & (!is.na(Sexed) & Sexed!=0))
tsdL <- with (CC_Mediterranean, tsd(males=Males, females=Females,
                                    temperatures=Incubation.temperature,
                                    equation="logistic", replicate.CI=NULL))

PM <- info.nests(resultNest_4p_SSM4p,
  embryo.stages="Caretta caretta.SCL", replicate.CI=100,
  CI="Hessian",
  out="summary", progress=TRUE, tsd=tsdL)

sr_TSP_SCLW <- predict(tsdL, temperatures=PM$summary[, "TSP.MassWeighted.temperature.mean"],
               temperature.se=PM$summary[, "TSP.MassWeighted.temperature.se"])

plot_errbar(x=sr_TSP_SCLW["50%", ], y=PM$summary[, "PM.mean"],
            x.minus=sr_TSP_SCLW["2.5%", ],
            x.plus=sr_TSP_SCLW["97.5%", ],
            errbar.y=2*PM$summary[, "PM.se"], xlab="CTE SCL growth",
            ylab="PM Massey et al. 2016", xlim=c(0, 1), ylim=c(0, 1), las=1)
segments(x0=0, y0=0, x1=1, y1=1, lty=2, col="red")

sr_TSP_TW <- predict(tsdL, temperatures=PM$summary[, "TSP.TimeWeighted.temperature.mean"],
               temperature.se=PM$summary[, "TSP.TimeWeighted.temperature.se"])

plot_errbar(x=sr_TSP_TW["50%", ], y=PM$summary[, "PM.mean"],
            x.minus=sr_TSP_TW["2.5%", ],
            x.plus=sr_TSP_TW["97.5%", ],
            errbar.y=2*PM$summary[, "PM.se"], xlab="CTE Time",
            ylab="PM Massey et al. 2016", xlim=c(0, 1), ylim=c(0, 1), las=1)
segments(x0=0, y0=0, x1=1, y1=1, lty=2, col="red")
```

```
plot_errbar(x=sr_TSP_TW["50%", ], y=sr_TSP_SCLW["50%", ],
            x.minus=sr_TSP_TW["2.5%", ],
            x.plus=sr_TSP_TW["97.5%", ],
            y.minus=sr_TSP_SCLW["2.5%", ],
            y.plus=sr_TSP_SCLW["97.5%", ], xlab="CTE Time",
            ylab="CTE SCL", xlim=c(0, 1), ylim=c(0, 1), las=1)
segments(x0=0, y0=0, x1=1, y1=1, lty=2, col="red")


## End(Not run)
```

`likelihoodR` *Estimate the likelihood of a set of parameters for nest incubation data*

Description

Estimate the likelihood of a set of parameters for nest incubation data

Usage

```
likelihoodR(result = NULL, parameters = NULL,
  fixed.parameters = NULL, temperatures = NULL, derivate = NULL,
  test = NULL, M0 = NULL, hessian = FALSE, weight = NULL,
  parallel = TRUE, echo = TRUE)
```

Arguments

`result`	A object obtained after searchR or likelihoodR
`parameters`	A set of parameters
`fixed.parameters`	A set of parameters that will not be changed
`temperatures`	Timeseries of temperatures
`derivate`	Function used to fit embryo growth: dydt.Gompertz, dydt.exponential or dydt.linear
`test`	Mean and SD of size of hatchlings
`M0`	Measure of hatchling size or mass proxi at laying date
`hessian`	If TRUE, the hessian matrix is estimated and the SE of parameters estimated.
`weight`	A named vector of the weight for each nest for likelihood estimation
`parallel`	If true, try to use several cores using parallel computing.
`echo`	If FALSE, does not display the result.

Details

likelihoodR estimates the likelihood of a set of parameters for nest incubation data

Value

A result object

Author(s)

Marc Girondot

Examples

```
## Not run:
library(embryogrowth)
data(nest)
formated <- FormatNests(nest)
# The initial parameters value can be:
# "T12H", "DHA",  "DHH", "Rho25"
# Or
# "T12L", "DT", "DHA",  "DHH", "DHL", "Rho25"
# K for Gompertz must be set as fixed parameter or being a constant K
# or relative to the hatchling size rK
x <- structure(c(118.768297442004, 475.750095909406, 306.243694918151,
116.055824800264), .Names = c("DHA", "DHH", "T12H", "Rho25"))
# pfixed <- c(K=82.33) or rK=82.33/39.33
pfixed <- c(rK=2.093313)
# K or rK are not used for dydt.linear or dydt.exponential
LresultNest_4p <- likelihoodR(parameters=x, fixed.parameters=pfixed,
temperatures=formated, derivate=dydt.Gompertz, M0=1.7,
test=c(Mean=39.33, SD=1.92))
data(resultNest_4p_SSM4p)
LresultNest_4p <- likelihoodR(result=resultNest_4p_SSM4p)

## End(Not run)
```

`logLik.HatchingSuccess` *Return -log L of a fit*

Description

Set of functions to study the hatching success.

Usage

```
## S3 method for class 'HatchingSuccess'
logLik(object, ...)
```

Arguments

`object`	The return of a fit done with fitHS.
`...`	Not used

Details

logLik.HatchingSuccess returns -log L of a fit

Value

Return -log L of a fit

Author(s)

Marc Girondot

See Also

Other Hatching success: `HatchingSuccess.MHmcmc_p`, `HatchingSuccess.MHmcmc`, `HatchingSuccess.fit`, `HatchingSuccess.lnL`, `HatchingSuccess.model`, `nobs.HatchingSuccess`, `plot.HatchingSuccess`, `predict.HatchingSuccess`

Examples

```
## Not run:
library(embryogrowth)
totalIncubation_Cc <- subset(DatabaseTSD,
                             Species=="Caretta caretta" &
                               Note != "Sinusoidal pattern" &
                               !is.na(Total) & Total != 0)

par <- c(S.low=0.5, S.high=0.3,
         P.low=25, deltaP=10, MaxHS=logit(0.8))

HatchingSuccesss.lnL(par=par, data=totalIncubation_Cc)

g <- HatchingSuccess.fit(par=par, data=totalIncubation_Cc)

HatchingSuccesss.lnL(par=g$par, data=totalIncubation_Cc)

t <- seq(from=20, to=40, by=0.1)
CIq <- predict(g, temperature=t)

par(mar=c(4, 4, 1, 1), +0.4)
plot(g)

## End(Not run)
```

`logLik.NestsResult` *Return Log Likelihood of a fit generated by searchR*

Description

Return Log Likelihood of a fit generated by searchR

Usage

```
## S3 method for class 'NestsResult'
logLik(object, ...)
```

Arguments

`object`	A result file generated by searchR
`...`	Not used

Details

logLik.NestsResult Return Log Likelihood of a fit

Value

The Log Likelihood value of the fitted model and data

Author(s)

Marc Girondot

Examples

```
## Not run:
library(embryogrowth)
data(resultNest_4p_SSM4p)
logLik(resultNest_4p_SSM4p)
AIC(resultNest_4p_SSM4p)

## End(Not run)
```

`logLik.STRN` *Return Log Likelihood of a fit generated by STRN*

Description

Return Log Likelihood of a fit generated by STRN

Usage

```
## S3 method for class 'STRN'
logLik(object, ...)
```

Arguments

`object`	A result file generated by STRN
`...`	Not used

Details

logLik.STRN Return Log Likelihood of a fit

Value

The Log Likelihood value of the fitted model and data

Author(s)

Marc Girondot

Examples

```
## Not run:
library(embryogrowth)
data(resultNest_4p_SSM4p)
logLik(resultNest_4p_SSM4p)
AIC(resultNest_4p_SSM4p)

## End(Not run)
```

`logLik.tsd` *Return Log Likelihood of a fit generated by tsd*

Description

Return Log Likelihood of a fit generated by tsd. The object has 3 attributes:
nall, and nobs the number of observations, df, the number of fitted parameters.

Usage

```
## S3 method for class 'tsd'
logLik(object, ...)
```

Arguments

`object`	A result file generated by tsd
`...`	Not used

Details

logLik.tsd Return Log Likelihood of a fit

Value

The Log Likelihood value of the fitted model and data

Author(s)

Marc Girondot

Examples

```
## Not run:
library(embryogrowth)
m <- c(10, 14, 7, 4, 3, 0, 0)
f <- c(0, 1, 2, 4, 15, 10, 13)
t <- c(25, 26, 27, 28, 29, 30, 31)
result <- tsd(males=m, females=f, temperatures=t)
logLik(result)
AIC(result)

## End(Not run)
```

`MovingIncubation` *Simulate incubation of a nest with the beginning of incubation varying*

Description

Simulate incubation of a nest with the beginning varying day by day
Temperatures must be in a data.frame with one column (Time) being the time and the second the temperatures (Temperature). A third columns can indicate the temperature at the end of incubation (Temperature.end.incubation). Do not use FormatNests() for this dataframe.

Usage

```
MovingIncubation(NestsResult = NULL, resultmcmc = NULL,
  temperatures.df = stop("A data.frame with timeseries of temperatures must be provided"),
   metabolic.heating = 0, CI = "Hessian",
   temperature.heterogeneity = 0, average.incubation.duration = 60 *
   1440, skip = 1, parameters = NULL, fixed.parameters = NULL,
   SE = NULL, hessian = NULL, derivate = NULL, test = NULL,
   M0 = NULL, TSP.borders = c(21, 26),
   embryo.stages = "Caretta caretta.SCL", SexualisationTRN = NULL,
   SexualisationTRN.CI = "Hessian", replicate.CI = 1, parallel = TRUE,
   progressbar = TRUE)
```

Arguments

`NestsResult`	A result file generated by searchR
`resultmcmc`	A mcmc result. Will be used rather than SE if provided.
`temperatures.df`	A data.frame with 2 or 3 columns: Times, Temperatures and Temperatures.end.incubation (facultative)

`metabolic.heating`	Degrees Celsius to be added at the end of incubation due to metabolic heating
`CI`	How to estimate CI for embryo growth thermal reaction norm; can be NULL, "SE", "MCMC", or "Hessian".
`temperature.heterogeneity`	SD of heterogeneity of temperatures. Can be 2 values, sd_low and sd_high and then HelpersMG::r2norm() is used.
`average.incubation.duration`	The average time to complete incubation (not used if metabolic heating is setup)
`skip`	Number of data to skip between two runs
`parameters`	A set of parameters if result is not provided.
`fixed.parameters`	Another set of parameters if result is not provided.
`SE`	Standard error for each parameter if not present in result is not provided
`hessian`	A hessian matrix
`derivate`	Function used to fit embryo growth: dydt.Gompertz, dydt.exponential or dydt.linear
`test`	Mean and SD of size of hatchlings as a vector ie test=c(Mean=xx, SD=yy)
`M0`	Measure of hatchling size proxi at laying date
`TSP.borders`	The limits of TSP
`embryo.stages`	The embryo stages. At least TSP.borders stages must be provided to estimate TSP length
`SexualisationTRN`	A model for sexualisation thermal reaction norm during TSP obtained using STRN()
`SexualisationTRN.CI`	How to estimate CI of sexualisation thermal reaction norm. Can be NULL, "SE", "MCMC", or "Hessian".
`replicate.CI`	Number of randomizations to estimate CI
`parallel`	Should parallel computing be used. TRUE or FALSE
`progressbar`	Should a progress bar be shown ? TRUE or FALSE

Details

MovingIncubation simulate incubation of a nest with the beginning varying day by day

Value

A dataframe with informations about thermosensitive period length and incubation length day by day of incubation

Author(s)

Marc Girondot

Examples

```
## Not run:
library(embryogrowth)
data(resultNest_4p_SSM4p)
ti <- seq(from=0, to=(60*24*100), by=60)
temperatures <- rnorm(length(ti), 29, 5)
temperatures <- temperatures+ti/(60*24*100)/2
layout(mat=1:3)
parpre <- par(mar=c(4, 4, 1, 1)+0.4)
plot(ti/(60*24), temperatures, type="l", xlab="Days",
     ylab=expression("Nest temperature in "*degree*"C"), bty="n", las=1)
# The sexualisation thermal reaction norm is calculated for South Pacific RMU
out <- MovingIncubation(NestsResult=resultNest_4p_SSM4p,
     temperatures.df=data.frame(Time=ti, Temperature=temperatures),
     metabolic.heating = 0,
     SexualisationTRN = structure(c(71.922411148397, 613.773055147801,
     318.059753164125, 120.327257089974),
     .Names = c("DHA", "DHH", "T12H", "Rho25")))
with(out, plot(Time/(60*24), Incubation.length.mean/(60*24),
     xlab="Days along the season",
     ylab="Incubation duration",
     type="l", bty="n", las=1, ylim=c(74, 76)))
with(out, plot(Time/(60*24), TSP.MassWeighted.STRNWeighted.temperature.mean,
     xlab="Days along the season",
     ylab=expression("CTE for sex ratio in "*degree*"C"),
      type="l", bty="n", las=1, ylim=c(30, 31)))
par(mar=parpre)
layout(mat=c(1))

## End(Not run)
```

`nest` *Timeseries of temperatures for nests*

Description

Timeseries of temperatures for nests

Usage

```
nest
```

Format

A dataframe with raw data

Details

Timeseries of temperatures for nests

Author(s)

Marc Girondot `<marc.girondot@u-psud.fr>`

References

Girondot, M. & Kaska, Y. 2014. A model to predict the thermal reaction norm for the embryo growth rate from field data. Journal of Thermal Biology. 45, 96-102.

Examples

```
## Not run:
library(embryogrowth)
data(nest)

## End(Not run)
```

`nobs.HatchingSuccess` *Return number of observations of a fit*

Description

Set of functions to study the hatching success.

Usage

```
## S3 method for class 'HatchingSuccess'
nobs(object, ...)
```

Arguments

`object`	The return of a fit done with fitHS.
`...`	Not used

Details

nobs.NestsResult Return number of observations of a fit

Value

Return number of observations of a fit

Author(s)

Marc Girondot

See Also

Other Hatching success: `HatchingSuccess.MHmcmc_p`, `HatchingSuccess.MHmcmc`, `HatchingSuccess.fit`, `HatchingSuccess.lnL`, `HatchingSuccess.model`, `logLik.HatchingSuccess`, `plot.HatchingSuccess`, `predict.HatchingSuccess`

Examples

```
## Not run:
library(embryogrowth)
totalIncubation_Cc <- subset(DatabaseTSD,
                             Species=="Caretta caretta" &
                               Note != "Sinusoidal pattern" &
                               !is.na(Total) & Total != 0)

par <- c(S.low=0.5, S.high=0.3,
         P.low=25, deltaP=10, MaxHS=logit(0.8))

HatchingSuccesss.lnL(par=par, data=totalIncubation_Cc)

g <- HatchingSuccesss.fit(par=par, data=totalIncubation_Cc)

HatchingSuccesss.lnL(par=g$par, data=totalIncubation_Cc)

plot(g)

## End(Not run)
```

`nobs.NestsResult` *Return number of observations of a fit*

Description

Return number of observations of a fit.
This function is used for bbmle::ICtb().

Usage

```
## S3 method for class 'NestsResult'
nobs(object, ...)
```

Arguments

`object`	A result file generated by searchR
`...`	Not used

Details

nobs.NestsResult Return number of observations of a fit

Value

Return number of observations of a fit

Author(s)

Marc Girondot

Examples

```
## Not run:
library(embryogrowth)
data(resultNest_4p_SSM4p)
logLik(resultNest_4p_SSM4p)
AIC(resultNest_4p_SSM4p)
nobs(resultNest_4p_SSM4p)

## End(Not run)
```

`plot.HatchingSuccess` *Plot results of HatchingSuccess.fit() that best describe hatching success*

Description

Plot the estimates that best describe hatching success.
If resultmcmc is given and replicates is null, it will use all information to estimate confidence interval.

Usage

```
## S3 method for class 'HatchingSuccess'
plot(x, xlim = c(20, 40), ylim = c(0, 1),
  xlab = "Constant incubation temperatures", ylab = "Hatching success",
  bty = "n", las = 1, col.observations = "red",
  pch.observations = 19, cex.observations = 1,
  show.CI.observations = TRUE, col.mean = "black", lty.mean = 1,
  lwd.mean = 1, col.median = "black", lty.median = 2,
  lwd.median = 1, col.CI = "black", lty.CI = 3, lwd.CI = 1,
  replicates = NULL, resultmcmc = NULL, ...)
```

Arguments

`x`	A result file generated by HatchingSuccess.fit()
`xlim`	Range of temperatures
`ylim`	Hatching success range for y-axis
`xlab`	x label
`ylab`	y label

`bty`	bty graphival parameter
`las`	las graphical parameter
`col.observations`	Color of observations
`pch.observations`	Character used for observation (no observations if NULL)
`cex.observations`	Size of characters for observations
`show.CI.observations`	Should the confidence interval of the observations be shown ?
`col.mean`	Color of the mean model
`lty.mean`	Line type of the mean model (no line if NULL)
`lwd.mean`	Line width of the mean model
`col.median`	Color of the median model
`lty.median`	Line type of the median model (no line if NULL)
`lwd.median`	Line width of the mean model
`col.CI`	Color of the 95% confidence interval lines
`lty.CI`	Line type of the 95% confidence interval lines (no line if NULL)
`lwd.CI`	Line width of the 95% confidence interval lines
`replicates`	Number of replicates to estimate confidence interval
`resultmcmc`	Results obtained using HatchingSuccesss.MHmcmc()
`...`	Parameters for plot()

Details

plot.HatchingSuccess plot result of HatchingSuccess.fit() or HatchingSuccesss.MHmcmc() that best describe hatching success

Value

Nothing

Author(s)

Marc Girondot

See Also

Other Hatching success: `HatchingSuccess.MHmcmc_p`, `HatchingSuccess.MHmcmc`, `HatchingSuccess.fit`, `HatchingSuccess.lnL`, `HatchingSuccess.model`, `logLik.HatchingSuccess`, `nobs.HatchingSuccess`, `predict.HatchingSuccess`

Examples

```
## Not run:
library(embryogrowth)
totalIncubation_Cc <- subset(DatabaseTSD,
                             Species=="Caretta caretta" &
                               Note != "Sinusoidal pattern" &
                               !is.na(Total) & Total != 0)

par <- c(S.low=0.5, S.high=0.3,
         P.low=25, deltaP=10, MaxHS=logit(0.8))

HatchingSuccess.lnL(par=par, data=totalIncubation_Cc)

g <- HatchingSuccess.fit(par=par, data=totalIncubation_Cc)

HatchingSuccess.lnL(par=g$par, data=totalIncubation_Cc)

plot(g, replicates=0)
plot(g, replicates=10000)

pMCMC <- HatchingSuccess.MHmcmc_p(g, accept=TRUE)
mcmc <- HatchingSuccess.MHmcmc(result=g, parameters = pMCMC,
                                adaptive=TRUE, n.iter=100000, trace=1000)
plot(g, resultmcmc=mcmc)
plot(g, resultmcmc=mcmc, pch.observations=NULL, lty.mean=NULL)

## End(Not run)
```

`plot.NestsResult` *Plot the embryo growth*

Description

Plot the embryo growth from one or several nests.
The embryo.stages is a named vector with relative size as compared to final size at the beginning of the stage. Names are the stages.
For example for SCL in Caretta caretta:
embryo.stages=structure(c(8.4, 9.4, 13.6, 13.8, 18.9, 23.5, 32.2, 35.2, 35.5, 38.5)/39.33),
.Names = c("21", "22", "23", "24", "25", "26", "27", "28", "29", "30", "31"))
indicates that the stages 21 begins at the relative size of 8.4/39.33.
Series can be indicated as the name of the series, its number or succesion of TRUE or FALSE. "all" indicates that all series must be printed.
show.fioritures parameter does not affect show.test option.
Note: Four species have predefined embryo stages. embryo.stages parameter can take the values:

- `Caretta caretta.SCL`
- `Chelonia mydas.SCL`

- `Emys orbicularis.SCL`
- `Emys orbicularis.mass`
- `Podocnemis expansa.SCL`
- `Generic.ProportionDevelopment`

Usage

```
## S3 method for class 'NestsResult'
plot(x, ..., parameters = NULL,
  fixed.parameters = NULL, resultmcmc = NULL, hessian = NULL,
  SE = NULL, temperatures = NULL, derivate = NULL, test = NULL,
  stopattest = FALSE, M0 = NULL, weight = NULL, series = "all",
  TSP.borders = NULL, embryo.stages = NULL, TSP.begin = 0,
  TSP.end = 0.5, replicate.CI = 100, metric.end.incubation = NULL,
  col.stages = "blue", col.PT = "red", col.TSP = "gray",
  col.temperatures = "green", col.S = "black", lty.temperatures = 1,
  lwd.temperatures = 2, ylimT = NULL, ylimS = NULL, xlim = NULL,
  show.stages = TRUE, show.TSP = TRUE, show.third = TRUE,
  CI = NULL, show.metric = TRUE, show.fioritures = TRUE,
  show.temperatures = TRUE, show.PT = TRUE, PT = c(mean = NA, SE =
  NA), show.test = TRUE, add = FALSE,
  lab.third = "2nd third of incubation", at.lab.third = 10,
  lab.PT = "PT", lab.stages = "Stages", at.lab.TSP = 8,
  lab.TSP = "TSP", mar = c(4, 5, 4, 5) + 0.3,
  xlab = "Days of incubation", ylabT = expression("Temperature in " *
  degree * "C"), ylabS = "Embryo metric", progress = TRUE,
  parallel = TRUE)
```

Arguments

`x`	A result file generated by searchR
`...`	Parameters for plot()
`parameters`	A set of parameters if result is not provided.
`fixed.parameters`	Another set of parameters if result is not provided.
`resultmcmc`	A mcmc result. Will be used rather than SE if provided.
`hessian`	An Hessian matrix.
`SE`	Standard error for each parameter if result is not provided, or replace the one in NestsResult. Use SE=NA to remove SE from NestResult
`temperatures`	Timeseries of temperatures formatted using formatNests(). Will replace the one in result.
`derivate`	Function used to fit embryo growth: dydt.Gompertz, dydt.exponential or dydt.linear
`test`	Mean and SD of size of hatchlings
`stopattest`	TRUE or FALSE. If TRUE, the model stops when proxy of size reached the mean test size.

`M0`	Measure of hatchling size proxi at laying date
`weight`	Weights of the different nests to estimate likelihood
`series`	The name or number of the series to be displayed. Only one series can be displayed at a time.
`TSP.borders`	The limits of TSP in stages. See embryo.stages parameter.
`embryo.stages`	The embryo stages. At least TSP.borders stages must be provided to estimate TSP borders. See note.
`TSP.begin`	Where TSP begin during the stage of beginning? In relative proportion of the stage.
`TSP.end`	Where TSP begin during the stage of ending? In relative proportion of the stage.
`replicate.CI`	Number of replicates to estimate CI. If 1, no CI is estimated.
`metric.end.incubation`	The expected metric at the end of incubation. Used to calibrate TSP size. If NULL, take the maximum Mean of the test parameter. If NA, use the actual final size. Can be a vector and is recycled if necessary.
`col.stages`	The color of the stages
`col.PT`	The color of the pivotal temperature
`col.TSP`	The color of the TSP
`col.temperatures`	The color of the temperatures
`col.S`	The color of the size or mass. Can be a vector (useful when series="all" option).
`lty.temperatures`	Type of line for temperatures
`lwd.temperatures`	Width of line for temperatures
`ylimT`	Range of temperatures to be displayed
`ylimS`	Range of size to be displayed
`xlim`	Range of incubation days to be displayed
`show.stages`	TRUE or FALSE, does the embryo stages should be displayed?
`show.TSP`	TRUE or FALSE, does the TSP boders should be displayed?
`show.third`	TRUE or FALSE, does the first and second third boders should be displayed?
`CI`	How to estimate CI; can be NULL, "SE", "MCMC", or "Hessian"
`show.metric`	TRUE or FALSE, does the plot of embryo metric is shown?
`show.fioritures`	If FALSE, set show.PT, show.temperatures, show.stages, show.TSP, show.third to FALSE, CI to NULL
`show.temperatures`	TRUE or FALSE, does the temperatures should be displayed?
`show.PT`	TRUE or FALSE, does the pivotal temperature should be displayed?
`PT`	Value for pivotal temperature, mean and SE
`show.test`	TRUE or FALSE, does the hatchling size should be displayed

`add`	If TRUE, all the curves are shown on the same graph
`lab.third`	Label for 2nd third of incubation
`at.lab.third`	Position of Label for 2nd third of incubation [default=10]; y-lim is scaled by at.lab.third
`lab.PT`	Label for Pivotal Temperature
`lab.stages`	Label for Stages
`at.lab.TSP`	Position of Label for TSP [default=8]; y-lim is scaled by at.lab.third
`lab.TSP`	Label for the TSP
`mar`	Parameter mar used for plot
`xlab`	Label for axis
`ylabT`	Label for temperature axis
`ylabS`	Label for size axis
`progress`	If FALSE, the progress bar is not shown (useful for use with sweave or knitr)
`parallel`	Should parallel computing be used ? TRUE or FALSE
`NestsResult`	A NestsResult file generated by searchR

Details

plot.NestsResult Plot the embryo growth

Value

Nothing

Author(s)

Marc Girondot

Examples

```
## Not run:
library(embryogrowth)
data(resultNest_4p_SSM4p)
plot(resultNest_4p_SSM4p, xlim=c(0,70), ylimT=c(22, 32), ylimS=c(0,45), series=1,
    SE=c(DHA=1.396525, DHH=4.101217, T12H=0.04330405, Rho25=1.00479),
    CI = "SE", replicate.CI = 100,
    embryo.stages="Caretta caretta.SCL")
plot(resultNest_4p_SSM4p, xlim=c(0,70), ylimT=c(22, 32), ylimS=c(0,45), series=1,
    CI = "Hessian", replicate.CI = 100,
    embryo.stages="Caretta caretta.SCL")
plot(resultNest_4p_SSM4p, xlim=c(0,70), ylimT=c(22, 32), ylimS=c(0,45), series=1,
     resultmcmc = resultNest_mcmc_4p_SSM4p,
    CI = "MCMC", replicate.CI = 100,
     embryo.stages="Caretta caretta.SCL")
# to plot all the nest at the same time, use
plot(resultNest_4p_SSM4p, xlim=c(0,70), ylimT=c(22, 32), ylimS=c(0,45),
    series="all", show.fioritures=FALSE, add=TRUE,
```

```
    embryo.stages="Caretta caretta.SCL")
# to use color different for series
plot(resultNest_4p_SSM4p, xlim=c(0,70), ylimT=c(22, 32), ylimS=c(0,45), add=TRUE,
    series="all", show.fioritures=FALSE, col.S=c(rep("black", 5), rep("red", 6)),
    embryo.stages="Caretta caretta.SCL")

# to plot all the temperature profiles

par(mar=c(4, 4, 1, 1))
plot(resultNest_4p_SSM4p$data[[1]][, 1]/60/24,
     resultNest_4p_SSM4p$data[[1]][, 2], bty="n",
     las=1, xlab="Days of incubation",
     ylab=expression("Temperatures in "*degree*"C"),
     type="l", xlim=c(0,70),ylim=c(20, 35))

     for (i in 2:21) {
          par(new=TRUE)
          plot(resultNest_4p_SSM4p$data[[i]][, 1]/60/24,
          resultNest_4p_SSM4p$data[[i]][, 2], bty="n",
          las=1, xlab="", ylab="", type="l", xlim=c(0,70),
          ylim=c(20, 35), axes = FALSE)
     }

## End(Not run)
```

---

plot.tsd *Plot results of tsd() that best describe temperature-dependent sex determination*

Description

Plot the estimates that best describe temperature-dependent sex determination.

Usage

```
## S3 method for class 'tsd'
plot(x, ..., show.observations = TRUE, show.model = TRUE,
  males.freq = TRUE, show.PTRT = TRUE, las.x = 1, las.y = 1,
  lab.PT = paste("Pivotal ", x$type), resultmcmc = NULL, chain = 1,
  l = 0.05, replicate.CI = 10000, range.CI = 0.95, mar = c(4, 4, 4,
  1) + 0.4, temperatures.plot = seq(from = 25, to = 35, by = 0.1),
  durations.plot = seq(from = 40, to = 70, by = 0.1),
  lab.TRT = paste0("Transitional range of ", x$type, "s l=", x$l * 100,
  "%"), col.TRT = "gray", col.TRT.CI = rgb(0.8, 0.8, 0.8, 0.5),
  col.PT.CI = rgb(0.8, 0.8, 0.8, 0.5), show.CI = TRUE, warn = TRUE)
```

Arguments

x	A result file generated by tsd()

`...`	Parameters for plot()
`show.observations`	Should the observations be shown
`show.model`	Should the model be shown
`males.freq`	Should the graph uses males relative frequency [TRUE] or females [FALSE]
`show.PTRT`	Should the P and TRT information be shown
`las.x`	las parameter for x axis
`las.y`	las parameter for y axis
`lab.PT`	Label to describe pivotal temperature
`resultmcmc`	A result of tsd_MHmcmc()
`chain`	What chain to be used is resultmcmc is provided
`l`	Sex ratio limits to define TRT are l and 1-l
`replicate.CI`	replicate.CI replicates from the hessian matrix to estimate CI
`range.CI`	The range of confidence interval for estimation, default=0.95
`mar`	The par("mar") parameter
`temperatures.plot`	Temperatures used for showing curves of sex ratio
`durations.plot`	Durations used for showing curves of sex ratio
`lab.TRT`	Label to describe transitional range of temperature
`col.TRT`	The color of TRT
`col.TRT.CI`	The color of CI of TRT based on range.CI
`col.PT.CI`	The color of CI of PT based on range.CI
`show.CI`	Do the CI for the curve should be shown
`warn`	Do the warnings must be shown ? TRUE or FALSE

Details

plot.tsd plot result of tsd() that best describe temperature-dependent sex determination

Value

Nothing

Author(s)

Marc Girondot

References

Girondot, M. 1999. Statistical description of temperature-dependent sex determination using maximum likelihood. Evolutionary Ecology Research, 1, 479-486.

Godfrey, M.H., Delmas, V., Girondot, M., 2003. Assessment of patterns of temperature-dependent sex determination using maximum likelihood model selection. Ecoscience 10, 265-272.

Hulin, V., Delmas, V., Girondot, M., Godfrey, M.H., Guillon, J.-M., 2009. Temperature-dependent sex determination and global change: are some species at greater risk? Oecologia 160, 493-506.

Girondot M., Submited. On the concept of embryological thermosensitive period for sex determination in reptiles.

See Also

Other Functions for temperature-dependent sex determination: `DatabaseTSD`, `P_TRT`, `TSP.list`, `predict.tsd`, `stages`, `tsd_MHmcmc_p`, `tsd_MHmcmc`, `tsd`

Examples

```
## Not run:
CC_AtlanticSW <- subset(DatabaseTSD, RMU=="Atlantic, SW" &
                           Species=="Caretta caretta" & (!is.na(Sexed) & Sexed!=0))
tsdL <- with (CC_AtlanticSW, tsd(males=Males, females=Females,
                                  temperatures=Incubation.temperature-Correction.factor,
                                  equation="logistic"))
plot(tsdL)

## End(Not run)
```

`plotR` *Show the fitted growth rate dependent on temperature and its density*

Description

Show the fitted growth rate dependent on temperature and its density.
The curve "ML quantiles" is based on delta method.
The curve "ML" just shows the fitted model.
The curve "MCMC quantiles" uses the mcmc replicates to build the quantiles.
The curve "MCMC mean-SD" uses the mcmc replicates to build a symetric credibility interval.
The parameter curves is case insensitive. If only parameters is given, curves must be ML.

Usage

```
plotR(result = NULL, resultmcmc = NULL, parameters = NULL,
  fixed.parameters = NULL, temperatures = NULL,
  curves = "ML quantiles", set.par = 1, ylim = c(0, 5),
  xlim = c(20, 35), hessian = NULL, replicate.CI = 1000,
  cex.lab = par("cex"), cex.axis = par("cex"), scaleY = "auto",
  lty = 1, ltyCI = 3, lwd = 1, lwdCI = 1, col = "black",
```

```
  col.polygon = "grey", polygon = FALSE, probs = 0.95,
  colramp = colorRampPalette(c("white", rgb(red = 0.5, green = 0.5, blue
  = 0.5))), bandwidth = c(0.1, 0.01), pch = "", main = "",
  xlab = expression("Temperature in " * degree * "C"), ylab = NULL,
  bty = "n", las = 1, by.temperature = 0.1, show.density = FALSE,
  new = TRUE, show.hist = FALSE, ylimH = NULL, atH = NULL,
  ylabH = "Temperature density", breaks = "Sturges",
  log.hist = FALSE, mar = NULL)
```

Arguments

`result`	A result object or a list of result objects
`resultmcmc`	A result object from GRTN_MHmcmc() function
`parameters`	A set of parameters - Has the priority over result
`fixed.parameters`	A set of fixed parameters
`temperatures`	A set of temperatures - Has the priority over result
`curves`	What curves to show: "MCMC quantiles" or "MCMC mean-SD" based on mcmc or "ML" or "ML quantiles" for maximum-likelihood
`set.par`	1 or 2 to designate with set of parameters to show
`ylim`	Range of values for y-axis
`xlim`	Range of values for x-axis
`hessian`	An hessian matrix
`replicate.CI`	Number of replicates to estimate confidence interval with Hessian if delta method failed
`cex.lab`	cex value for axis
`cex.axis`	cex value for axis
`scaleY`	Scaling factor for y axis or "auto"
`lty`	The type of lines
`ltyCI`	The type of lines
`lwd`	The type of lines
`lwdCI`	The type of lines
`col`	The color of the lines
`col.polygon`	The color of the polygon
`polygon`	If TRUE, confidence interval is shown as a polygon with color
`probs`	Confidence or credibility interval to show
`colramp`	Ramp function accepting an integer as an argument and returning n colors.
`bandwidth`	numeric vector (length 1 or 2) of smoothing bandwidth(s). If missing, a more or less useful default is used. bandwidth is subsequently passed to function bkde2D.
`pch`	Character for outlayers

`main`	Title of the graph
`xlab`	Label for x axis
`ylab`	Label for y axis
`bty`	Box around the pot
`las`	Orientation for labels in y axis
`by.temperature`	Step to built the temperatures
`show.density`	TRUE or FALSE for use with Hessian or MCMC
`new`	Should the graphics be a new one (TRUE) or superimposed to a previous one (FALSE)
`show.hist`	TRUE or FALSE
`ylimH`	Scale of histogram using ylimH=c(min, max)
`atH`	Position of ticks for scale of histogram
`ylabH`	Label for histogram scale
`breaks`	See ?hist
`log.hist`	SHould the y scale for hist is log ?
`mar`	The value of par("mar"). If null, it will use default depending on show.dist. If NA, does not change par("mar").

Details

plotR shows the fitted growth rate dependent on temperature and the density of the mcmc

Value

The value of scaleY to be used with other plotR function

Author(s)

Marc Girondot

Examples

```
## Not run:
library(embryogrowth)
plotR(result = resultNest_4p_SSM4p,
            resultmcmc=resultNest_mcmc_4p_SSM4p,
            curves = "MCMC quantiles")
#################
plotR(resultmcmc=resultNest_mcmc_4p_SSM4p,
            curves = "MCMC quantiles", show.density=TRUE)
#################
plotR(resultmcmc=resultNest_mcmc_4p_SSM4p,
            curves = "MCMC quantiles", polygon=TRUE)
#################
plotR(resultmcmc=resultNest_mcmc_6p_SSM6p, ylim=c(0,4),
      curves = "MCMC quantiles", polygon=TRUE, col.polygon = rgb(0, 1, 0, 1))
```

```
plotR(resultmcmc=resultNest_mcmc_4p_SSM4p,
      curves = "MCMC quantiles", polygon=TRUE, col.polygon = rgb(1, 0, 0, 0.5), new=FALSE)
legend("topleft", legend=c("SSM 4 parameters", "SSM 6 parameters"),
        pch=c(15, 15), col=c(rgb(1, 0, 0, 0.5), rgb(0, 1, 0, 1)))
#################
sy <- plotR(resultmcmc=resultNest_mcmc_4p_SSM4p,
             curves = "MCMC quantiles", show.density=FALSE)
plotR(resultmcmc=resultNest_mcmc_6p_SSM6p, col="red",
             curves = "MCMC quantiles", show.density=FALSE,
             new=FALSE, scaleY=sy)
#################
sy <- plotR(result=resultNest_6p_SSM6p, curves="ML",
             show.hist = TRUE, new = TRUE)
plotR(result=resultNest_4p_SSM4p, curves="ML", scaleY=sy,
             show.hist = FALSE, new = FALSE, col="red")
#################
plotR(result=resultNest_6p_SSM6p, curves="ML",
             show.hist = TRUE, ylimH=c(0,1), atH=c(0, 0.1, 0.2))
################
plotR(result = resultNest_4p_SSM4p,
             resultmcmc=resultNest_mcmc_4p_SSM4p,
             show.density = TRUE,
             curves = "MCMC quantiles")
#################
plotR(result=resultNest_4p_SSM4p,
             ylim=c(0, 4), curves="ML quantiles", scaleY=1E5)
#################
plotR(result=resultNest_4p_SSM4p, show.hist = TRUE,
             ylim=c(0, 4), curves="ML quantiles", scaleY=1E5)
#################
plotR(resultmcmc=resultNest_mcmc_4p_SSM4p,
             ylim=c(0, 4), curves = "MCMC quantiles", show.density=TRUE, scaleY=1E5)

## End(Not run)
```

`plot_transition` *Show fonction used for transition*

Description

Plot the transition function

Usage

```
plot_transition(result = NULL, parameters = NULL, sizes = c(0, 40),
  ...)
```

Arguments

`result`	A result object
`parameters`	Set of parameters. If both result and parameters are indicated, parameters have priority.
`sizes`	The range of possible sizes
`...`	Parameters for plot() such as main= or ylim=

Details

plot_transition show fonction used for transition

Value

Nothing

Author(s)

Marc Girondot

Examples

```
## Not run:
library(embryogrowth)
data(nest)
formated <- FormatNests(nest)
data(resultNest_4p_SSM4p)
# Get a set of parameters without transition
x1 <- resultNest_4p_SSM4p$par
# Generate a set of parameters with transition
x2 <- switch.transition(x1)
x2 <- x2[names(x2)!="transition_P"]
x2["transition_S"] <- 4
pfixed <- c(rK=2.093313, transition_P=20)
resultNest_4p_transition <- searchR(parameters=x2, fixed.parameters=pfixed,
temperatures=formated, derivate=dydt.Gompertz, M0=1.7,
test=c(Mean=39.33, SD=1.92))
data(resultNest_4p_transition)
# show the model for smallest size
plotR(resultNest_4p_transition, ylim=c(0,0.3))
# show the model for larger sizes
plotR(resultNest_4p_transition, set.par=2, ylim=c(0,0.3))
# plot model for both together
plotR(resultNest_4p_transition, set.par=c(1,2), ylim=c(0,0.3),
       col=c("red", "black"), legend=list("Initial", "End"))
plot_transition(result=resultNest_4p_transition, las=1, sizes=c(0,40))
compare_AIC(one.model=list(resultNest_4p_SSM4p), two.models=list(resultNest_4p_transition))
# Note that the model with fitted transition_P is trivial. Embryos grow fast until
# they reach hatchling size and then growth rate becomes null!

## End(Not run)
```

`predict.HatchingSuccess` *Return prediction based on a model fitted with HatchingSuccess.fit()*

Description

Set of functions to study the hatching success.
If replicates is NULL or 0, it returns the fitted model.

Usage

```
## S3 method for class 'HatchingSuccess'
predict(object, ..., temperature = NULL,
  probs = c(0.025, 0.5, 0.975), replicates = NULL, resultmcmc = NULL)
```

Arguments

`object`	The return of a fit done with HatchingSuccesss.fit().
`...`	Not used
`temperature`	A vector of temperatures.
`probs`	Quantiles.
`replicates`	Number of replicates to estimate the confidence interval.
`resultmcmc`	Results obtained using HatchingSuccesss.MHmcmc()

Details

predict.HatchingSuccess returns prediction based on a model fitted with HatchingSuccessfit()

Value

Return a matrix with prediction based on a model fitted with HatchingSuccess.fit()

Author(s)

Marc Girondot

See Also

Other Hatching success: `HatchingSuccess.MHmcmc_p`, `HatchingSuccess.MHmcmc`, `HatchingSuccess.fit`, `HatchingSuccess.lnL`, `HatchingSuccess.model`, `logLik.HatchingSuccess`, `nobs.HatchingSuccess`, `plot.HatchingSuccess`

Examples

```
## Not run:
library(embryogrowth)
totalIncubation_Cc <- subset(DatabaseTSD,
                             Species=="Caretta caretta" &
                               Note != "Sinusoidal pattern" &
                               !is.na(Total) & Total != 0)

par <- c(S.low=0.5, S.high=0.3,
         P.low=25, deltaP=10, MaxHS=logit(0.8))

HatchingSuccess.lnL(x=par, data=totalIncubation_Cc)

g <- HatchingSuccess.fit(par=par, data=totalIncubation_Cc)

HatchingSuccess.lnL(par=g$par, data=totalIncubation_Cc)

plot(g)

## End(Not run)
```

`predict.tsd` *Estimate sex ratio according to constant incubation temperature*

Description

Estimate sex ratio according to constant incubation temperature
The data.frame has the temperatures or durations in columns and the quantiles in rows.
Note that incubation duration is a very bad proxy for sex ratio. See Georges, A., Limpus, C. J. & Stoutjesdijk, R. 1994. Hatchling sex in the marine turtle Caretta caretta is determined by proportion of development at a temperature, not daily duration of exposure. J. Exp. Zool., 270, 432-444.
If replicate.CI is 0 or NULL, point estimate for maximum likelihood is returned.

Usage

```
## S3 method for class 'tsd'
predict(object, temperatures = NULL, durations = NULL,
  SD.temperatures = NULL, SD.durations = NULL, resultmcmc = NULL,
  chain = 1, replicate.CI = 10000, probs = c(0.025, 0.5, 0.975), ...)
```

Arguments

`object`	A result file generated by tsd
`temperatures`	A vector of temperatures
`durations`	A vector of durations
`SD.temperatures`	SD of temperatures

`SD.durations`	SD of durations
`resultmcmc`	A result of tsd_MHmcmc()
`chain`	What chain to be used is resultmcmc is provided
`replicate.CI`	Number of replicates to estimate CI
`probs`	The quantiles to be returned, default=c(0.025, 0.5, 0.975)
`...`	Not used

Details

predict.tsd Estimate sex ratio according to constant incubation temperature

Value

A data.frame with informations about sex-ratio

Author(s)

Marc Girondot

See Also

Other Functions for temperature-dependent sex determination: `DatabaseTSD`, `P_TRT`, `TSP.list`, `plot.tsd`, `stages`, `tsd_MHmcmc_p`, `tsd_MHmcmc`, `tsd`

Examples

```
## Not run:
library(embryogrowth)
m <- c(10, 14, 7, 4, 3, 0, 0)
f <- c(0, 1, 2, 4, 15, 10, 13)
t <- c(25, 26, 27, 28, 29, 30, 31)
result <- tsd(males=m, females=f, temperatures=t)
plot(result)
predict(result, temperatures=c(25, 31), replicate.CI = 10000)
predict(result, temperatures=c(25, 31), SD.temperatures = c(1, 2), replicate.CI = 10000)
d <- c(72, 70, 65, 63, 62, 60, 59)
result <- tsd(males=m, females=f, durations=d)
predict(result, durations=c(67, 68), replicate.CI = 10000)

## End(Not run)
```

`P_TRT` *Estimate the transitional range of temperatures based on a set of parameters*

Description

Estimate the parameters that best describe the thermal reaction norm for sex ratio when temperature-dependent sex determination occurs.
It can be used also to evaluate the relationship between incubation duration and sex ratio.
The parameter l was defined in Girondot (1999). The TRT is defined from the difference between the two boundary temperatures giving sex ratios of l and 1 - l.
For logistic equation, exact value is used and precision iterations are used for other equations.
In Girondot (1999), l was 0.05 and then the TRT was defined as being the range of temperatures producing from 5 If l is null, TRT is not estimated and only sex ratio is estimated.

Usage

```
P_TRT(x = NULL, resultmcmc = NULL, chain = 1, equation = NULL,
  l = 0.05, replicate.CI = NULL, temperatures = NULL,
  durations = NULL, SD.temperatures = NULL, SD.durations = NULL,
  probs = c(0.025, 0.5, 0.975), warn = TRUE)
```

Arguments

`x`	Set of parameters or a result of tsd()
`resultmcmc`	A result of tsd_MHmcmc()
`chain`	What chain to be used is resultmcmc is provided
`equation`	What equation should be used. Must be provided if x is not a result of tsd()
`l`	Sex ratio limits to define TRT are l and 1-l. If NULL, TRT is not estimated.
`replicate.CI`	If a result of tsd() is provided, use replicate.CI replicates from the hessian matrix to estimate CI
`temperatures`	If provided returns the sex ratio and its quantiles for each temperature
`durations`	If provided returns the sex ratio and its quantiles for each duration
`SD.temperatures`	SD of temperatures
`SD.durations`	SD of durations
`probs`	Probabilities used to estimate quantiles
`warn`	Do the warnings must be shown ? TRUE or FALSE

Details

P_TRT estimates the transitional range of temperatures based on a set of parameters

Value

A list with a P_TRT object containing a matrix with lower and higher bounds for TRT, TRT and P and a P_TRT_quantiles object with quantiles for each and a sexratio_quantiles object

Author(s)

Marc Girondot

References

Girondot, M. 1999. Statistical description of temperature-dependent sex determination using maximum likelihood. Evolutionary Ecology Research, 1, 479-486.

Godfrey, M.H., Delmas, V., Girondot, M., 2003. Assessment of patterns of temperature-dependent sex determination using maximum likelihood model selection. Ecoscience 10, 265-272.

Hulin, V., Delmas, V., Girondot, M., Godfrey, M.H., Guillon, J.-M., 2009. Temperature-dependent sex determination and global change: are some species at greater risk? Oecologia 160, 493-506.

See Also

Other Functions for temperature-dependent sex determination: `DatabaseTSD`, `TSP.list`, `plot.tsd`, `predict.tsd`, `stages`, `tsd_MHmcmc_p`, `tsd_MHmcmc`, `tsd`

Examples

```
## Not run:
library("embryogrowth")
CC_AtlanticSW <- subset(DatabaseTSD, RMU=="Atlantic, SW" &
                          Species=="Caretta caretta" & Sexed!=0)
tsdL <- with (CC_AtlanticSW, tsd(males=Males, females=Females,
                                 temperatures=Incubation.temperature-Correction.factor,
                                 equation="logistic"))
P_TRT(tsdL)
P_TRT(tsdL, replicate.CI=1000)
P_TRT(tsdL, replicate.CI=1000, temperatures=20:35)
P_TRT_out <- P_TRT(tsdL, replicate.CI=1000, temperatures=c(T1=20, T2=35))
attributes(P_TRT_out$sexratio_quantiles)$temperatures
P_TRT(tsdL$par, equation="logistic")
pMCMC <- tsd_MHmcmc_p(tsdL, accept=TRUE)
# Take care, it can be very long
result_mcmc_tsd <- tsd_MHmcmc(result=tsdL,
parametersMCMC=pMCMC, n.iter=10000, n.chains = 1,
n.adapt = 0, thin=1, trace=FALSE, adaptive=TRUE)
P_TRT(result_mcmc_tsd, equation="logistic")

## End(Not run)
```

`resultNest_3p_Dallwitz`

Fit using the nest database

Description

Fit using the nest database

Usage

```
resultNest_3p_Dallwitz
```

Format

A list with fitted information about data(nest)

Details

Result of the fit using the nest database

Author(s)

Marc Girondot <marc.girondot@u-psud.fr>

References

Girondot, M., Monsinjon, J., Guillon, J.-M., Submitted. Delimitation of the embryonic thermosensitive period for sex determination using an embryo growth model reveals a potential bias for sex ratio prediction in turtles.

Examples

```
## Not run:
library(embryogrowth)
data(nest)
formated <- FormatNests(nest)
x <- structure(c(4.88677476830268, 20.4051904475743, 31.5173105860335),
.Names = c("Dallwitz_b1", "Dallwitz_b2", "Dallwitz_b3"))
pfixed <- c(rK=1.208968)
resultNest_3p_Dallwitz <- searchR(parameters=x, fixed.parameters=pfixed,
temperatures=formated, derivate=dydt.Gompertz, M0=0.3470893,
test=c(Mean=39.33, SD=1.92))
plotR(result=resultNest_3p_Dallwitz, show.hist = TRUE,
             ylim=c(0, 8), curves="ML quantiles")

## End(Not run)
```

`resultNest_3p_Weibull` *Result of the fit using the nest database using Weibull function*

Description

Fit using the nest database using Weibull function. The model is:
rT <- dweibull(T, shape=abs(parms["k"]),
scale=abs(parms["lambda"]))*parms["scale"]*1E-5

Usage

```
resultNest_3p_Weibull
```

Format

A list with fitted information about data(nest)

Details

Result of the fit using the nest database using Weibull function

Author(s)

Marc Girondot `<marc.girondot@u-psud.fr>`

References

Girondot, M., & Kaska, Y. (2014). A model to predict the thermal reaction norm for the embryo growth rate from field data. Journal of Thermal Biology, 45, 96-102. doi: 10.1016/j.jtherbio.2014.08.005

Examples

```
## Not run:
library(embryogrowth)
data(nest)
formated <- FormatNests(nest)
# Weibull model
 x <- ChangeSSM(temperatures = (200:350)/10,
                parameters = resultNest_4p_SSM4p$par,
                initial.parameters = structure(c(73.4009010417375, 304.142079511996,
                                           27.4671689276281),
                                      .Names = c("k", "lambda", "scale")),
                control=list(maxit=1000))
pfixed <- c(rK=2.093313)
resultNest_3p_Weibull <- searchR(parameters=x$par, fixed.parameters=pfixed,
                         temperatures=formated, derivate=dydt.Gompertz, M0=1.7,
                         test=c(Mean=39.33, SD=1.92))
plotR(resultNest_3p_Weibull, ylim=c(0, 3))
```

```
plotR(resultNest_3p_Weibull, ylim=c(0, 3), ylimH = c(0, 0.9), show.hist=TRUE)
compare_AIC(SSM=resultNest_4p_SSM4p, Weibull=resultNest_3p_Weibull)

## End(Not run)
```

`resultNest_4p_normal` *Result of the fit using the nest database using asymmetric normal function*

Description

Fit using the nest database using asymmetric normal function

Usage

```
resultNest_4p_normal
```

Format

A list with fitted information about data(nest)

Details

Result of the fit using the nest database using asymmetric normal function

Author(s)

Marc Girondot `<marc.girondot@u-psud.fr>`

References

Girondot, M., & Kaska, Y. (2014). A model to predict the thermal reaction norm for the embryo growth rate from field data. Journal of Thermal Biology, 45, 96-102. doi: 10.1016/j.jtherbio.2014.08.005

Examples

```
## Not run:
library(embryogrowth)
data(nest)
formated <- FormatNests(nest)
x <- ChangeSSM(temperatures = (200:350)/10,
               parameters = resultNest_4p_SSM4p$par,
               initial.parameters = structure(c(3, 7, 11, 32),
                                .Names = c("Scale", "sdL", "sdH", "Peak")),
               control=list(maxit=1000))
pfixed <- c(rK=2.093313)
resultNest_4p_normal <- searchR(parameters=x$par, fixed.parameters=pfixed,
                         temperatures=formated, derivate=dydt.Gompertz, M0=1.7,
                         test=c(Mean=39.33, SD=1.92))
plotR(resultNest_4p_normal, ylim=c(0, 3))
```

```
plotR(resultNest_4p_normal, ylim=c(0, 3), ylimH = c(0, 0.9), show.hist=TRUE)
compare_AIC(SSM=resultNest_4p_SSM4p, Asymmetric.normal=resultNest_4p_normal)

## End(Not run)
```

`resultNest_4p_SSM` *Fit using the nest database*

Description

Fit using the nest database

Usage

```
resultNest_4p_SSM
```

Format

A list with fitted information about data(nest)

Details

Result of the fit using the nest database

Author(s)

Marc Girondot <marc.girondot@u-psud.fr>

References

Girondot, M., Monsinjon, J., Guillon, J.-M., Submitted. Delimitation of the embryonic thermosensitive period for sex determination using an embryo growth model reveals a potential bias for sex ratio prediction in turtles.

Examples

```
## Not run:
library(embryogrowth)
data(nest)
formated <- FormatNests(nest)
x <- structure(c(109.683413821537, 614.969219372661, 306.386903812694,
 229.003478775323), .Names = c("DHA", "DHH", "T12H", "Rho25"))
pfixed <- c(rK=1.208968)
resultNest_4p_SSM <- searchR(parameters=x, fixed.parameters=pfixed,
temperatures=formated, derivate=dydt.Gompertz, M0=0.3470893,
test=c(Mean=39.33, SD=1.92))
plotR(result=resultNest_4p_SSM, show.hist = TRUE,
             ylim=c(0, 8), curves="ML quantiles")

## End(Not run)
```

`resultNest_4p_SSM4p` *Fit using the nest database*

Description

Fit using the nest database

Usage

```
resultNest_4p_SSM4p
```

Format

A list with fitted information about data(nest)

Details

Result of the fit using the nest database

Author(s)

Marc Girondot `<marc.girondot@u-psud.fr>`

References

Girondot, M., & Kaska, Y. (2014). A model to predict the thermal reaction norm for the embryo growth rate from field data. Journal of Thermal Biology, 45, 96-102. doi: 10.1016/j.jtherbio.2014.08.005

Examples

```
## Not run:
library(embryogrowth)
data(nest)
formated <- FormatNests(nest)
# The initial parameters value can be:
# "T12H", "DHA",  "DHH", "Rho25"
# Or
# "T12L", "DT", "DHA",  "DHH", "DHL", "Rho25"
x <- structure(c(118.431040984352, 498.205702157603, 306.056280989839,
118.189669472381), .Names = c("DHA", "DHH", "T12H", "Rho25"))
# pfixed <- c(K=82.33) or rK=82.33/39.33
pfixed <- c(rK=2.093313)
resultNest_4p_SSM4p <- searchR(parameters=x, fixed.parameters=pfixed,
temperatures=formated, derivate=dydt.Gompertz, M0=1.7,
test=c(Mean=39.33, SD=1.92))
plotR(result=resultNest_4p_SSM4p, show.hist = TRUE,
             ylim=c(0, 4), curves="ML quantiles")

## End(Not run)
```

`resultNest_4p_transition`

Result of the fit using the nest database using transition

Description

Fit using the nest database using transition

Usage

```
resultNest_4p_transition
```

Format

A list with fitted information about data(nest)

Details

Result of the fit using the nest database using transition

Author(s)

Marc Girondot `<marc.girondot@u-psud.fr>`

References

Girondot, M., & Kaska, Y. (2014). A model to predict the thermal reaction norm for the embryo growth rate from field data. Journal of Thermal Biology, 45, 96-102. doi: 10.1016/j.jtherbio.2014.08.005

Examples

```
## Not run:
library(embryogrowth)
data(nest)
formated <- FormatNests(nest)
data(resultNest_4p_transition)

## End(Not run)
```

`resultNest_4p_trigo` *Result of the fit using the nest database using trigonometric function*

Description

Fit using the nest database using trigonometric function

Usage

```
resultNest_4p_trigo
```

Format

A list with fitted information about data(nest)

Details

Result of the fit using the nest database using trigonometric function

Author(s)

Marc Girondot `<marc.girondot@u-psud.fr>`

References

Girondot, M., & Kaska, Y. (2014). A model to predict the thermal reaction norm for the embryo growth rate from field data. Journal of Thermal Biology, 45, 96-102. doi: 10.1016/j.jtherbio.2014.08.005

Examples

```
## Not run:
library(embryogrowth)
data(nest)
formated <- FormatNests(nest)
x <- ChangeSSM(temperatures = (200:350)/10,
               parameters = resultNest_4p_SSM4p$par,
               initial.parameters = structure(c(3, 20, 40, 32),
                          .Names = c("Max", "LengthB", "LengthE", "Peak")),
               control=list(maxit=1000))
pfixed <- c(rK=2.093313)
resultNest_4p_trigo <- searchR(parameters=x$par, fixed.parameters=pfixed,
                          temperatures=formated, derivate=dydt.Gompertz, M0=1.7,
                          test=c(Mean=39.33, SD=1.92))
plotR(resultNest_4p_trigo, ylim=c(0, 3))
plotR(resultNest_4p_trigo, ylim=c(0, 3), ylimH = c(0, 0.9), show.hist=TRUE)
compare_AIC(SSM=resultNest_4p_SSM4p, trigonometric=resultNest_4p_trigo)

## End(Not run)
```

`resultNest_4p_weight` *Fit using the nest database with weight*

Description

Fit using the nest database with weight

Usage

```
resultNest_4p_weight
```

Format

A list with fitted information about data(nest)

Details

Result of the fit using the nest database with weight

Author(s)

Marc Girondot `<marc.girondot@u-psud.fr>`

References

Girondot, M., & Kaska, Y. (2014). A model to predict the thermal reaction norm for the embryo growth rate from field data. Journal of Thermal Biology, 45, 96-102. doi: 10.1016/j.jtherbio.2014.08.005

Examples

```
## Not run:
library(embryogrowth)
data(nest)
formated <- FormatNests(nest)
w <- weightmaxentropy(formated, control_plot=list(xlim=c(20,36)))
x <- structure(c(118.768297442004, 475.750095909406, 306.243694918151,
116.055824800264), .Names = c("DHA", "DHH", "T12H", "Rho25"))
# pfixed <- c(K=82.33) or rK=82.33/39.33
pfixed <- c(rK=2.093313)
# K or rK are not used for dydt.linear or dydt.exponential
resultNest_4p_weight <- searchR(parameters=x,
fixed.parameters=pfixed, temperatures=formated,
derivate=dydt.Gompertz, M0=1.7, test=c(Mean=39.33, SD=1.92),
method = "BFGS", weight=w)
data(resultNest_4p_weight)
plotR(resultNest_4p_weight, ylim=c(0,0.50), xlim=c(15, 35))

## End(Not run)
```

`resultNest_5p_Dallwitz` *Fit using the nest database*

Description

Fit using the nest database

Usage

```
resultNest_5p_Dallwitz
```

Format

A list with fitted information about data(nest)

Details

Result of the fit using the nest database

Author(s)

Marc Girondot <`marc.girondot@u-psud.fr`>

References

Girondot, M., Monsinjon, J., Guillon, J.-M., Submitted. Delimitation of the embryonic thermosensitive period for sex determination using an embryo growth model reveals a potential bias for sex ratio prediction in turtles.

Examples

```
## Not run:
library(embryogrowth)
data(nest)
formated <- FormatNests(nest)
x <- structure(c(4.91191231405918, 12.7453211281394, 31.2670410811077,
 5.7449376569153, -0.825689964543813), .Names = c("Dallwitz_b1",
 "Dallwitz_b2", "Dallwitz_b3", "Dallwitz_b4", "Dallwitz_b5"))
pfixed <- c(rK=1.208968)
resultNest_5p_Dallwitz <- searchR(parameters=x, fixed.parameters=pfixed,
temperatures=formated, derivate=dydt.Gompertz, M0=0.3470893,
test=c(Mean=39.33, SD=1.92))
plotR(result=resultNest_5p_Dallwitz, show.hist = TRUE,
             ylim=c(0, 8), curves="ML quantiles")

## End(Not run)
```

`resultNest_6p_SSM` *Fit using the nest database*

Description

Fit using the nest database

Usage

```
resultNest_6p_SSM
```

Format

A list with fitted information about data(nest)

Details

Result of the fit using the nest database

Author(s)

Marc Girondot `<marc.girondot@u-psud.fr>`

References

Girondot, M., Monsinjon, J., Guillon, J.-M., Submitted. Delimitation of the embryonic thermosensitive period for sex determination using an embryo growth model reveals a potential bias for sex ratio prediction in turtles.

Examples

```
## Not run:
library(embryogrowth)
data(nest)
formated <- FormatNests(nest)
x <- structure(c(104.954347370542, 3447.10062406071, 661.269363920423,
 96.3871849546537, 306.456389026151, 232.105840347154), .Names = c("DHA",
 "DHH", "DHL", "DT", "T12L", "Rho25"))
pfixed <- c(rK=1.208968)
resultNest_6p_SSM <- searchR(parameters=x, fixed.parameters=pfixed,
temperatures=formated, derivate=dydt.Gompertz, M0=0.3470893,
test=c(Mean=39.33, SD=1.92))
plotR(result=resultNest_6p_SSM, show.hist = TRUE,
             ylim=c(0, 8), curves="ML")

## End(Not run)
```

`resultNest_6p_SSM6p` *Fit using the nest database*

Description

Fit using the nest database

Usage

```
resultNest_6p_SSM6p
```

Format

A list with fitted information about data(nest)

Details

Result of the fit using the nest database

Author(s)

Marc Girondot <marc.girondot@u-psud.fr>

References

Girondot, M., & Kaska, Y. (2014). A model to predict the thermal reaction norm for the embryo growth rate from field data. Journal of Thermal Biology, 45, 96-102. doi: 10.1016/j.jtherbio.2014.08.005

Examples

```
## Not run:
library(embryogrowth)
data(nest)
formated <- FormatNests(nest)
# The initial parameters value can be:
# "T12H", "DHA",  "DHH", "Rho25"
# Or
# "T12L", "DT", "DHA",  "DHH", "DHL", "Rho25"
x <- structure(c(115.770032186805, 428.649144244891, 503.716700005443,
12.2641939339981, 306.305498098871, 116.380086027401),
.Names = c("DHA", "DHH", "DHL", "DT", "T12L", "Rho25"))
pfixed <- c(rK=2.093313)
resultNest_6p_SSM6p <- searchR(parameters=x, fixed.parameters=pfixed,
temperatures=formated, derivate=dydt.Gompertz, M0=1.7,
test=c(Mean=39.33, SD=1.92))

## End(Not run)
```

`resultNest_mcmc_4p_SSM4p` *Result of the mcmc using the nest database*

Description

Fit using the nest database

Usage

```
resultNest_mcmc_4p_SSM4p
```

Format

A list of class mcmcComposite with mcmc result for data(nest) with 4 parameters and Gompertz model of growth

Details

Result of the mcmc using the nest database

Author(s)

Marc Girondot `<marc.girondot@u-psud.fr>`

References

Girondot, M., & Kaska, Y. (2014). A model to predict the thermal reaction norm for the embryo growth rate from field data. Journal of Thermal Biology, 45, 96-102. doi: 10.1016/j.jtherbio.2014.08.005

Examples

```
## Not run:
library(embryogrowth)
data(nest)
formated <- FormatNests(nest)
# The initial parameters value can be:
# "T12H", "DHA",  "DHH", "Rho25"
# Or
# "T12L", "DT", "DHA",  "DHH", "DHL", "Rho25"
x <- structure(c(118.431040984352, 498.205702157603, 306.056280989839,
118.189669472381), .Names = c("DHA", "DHH", "T12H", "Rho25"))
# pfixed <- c(K=82.33) or rK=82.33/39.33
pfixed <- c(rK=2.093313)
resultNest_4p_SSM4p <- searchR(parameters=x, fixed.parameters=pfixed,
temperatures=formated, derivate=dydt.Gompertz, M0=1.7,
test=c(Mean=39.33, SD=1.92))
data(resultNest_4p_SSM4p)
pMCMC <- TRN_MHmcmc_p(resultNest_4p_SSM4p, accept=TRUE)
```

```
# Take care, it can be very long, sometimes several days
resultNest_mcmc_4p_SSM4p <- GRTRN_MHmcmc(result=resultNest_4p_SSM4p,
 adaptive = TRUE,
parametersMCMC=pMCMC, n.iter=10000, n.chains = 1, n.adapt = 0,
thin=1, trace=TRUE)
data(resultNest_mcmc_4p_SSM4p)
1-rejectionRate(as.mcmc(resultNest_mcmc_4p_SSM4p))
as.parameters(resultNest_mcmc_4p_SSM4p)
layout(mat=matrix(1:4, nrow = 2))
plot(resultNest_mcmc_4p_SSM4p, parameters = "all", scale.prior = TRUE, las = 1)
layout(mat=1)
plotR(resultNest_4p_SSM4p, resultmcmc=resultNest_mcmc_4p_SSM4p, ylim=c(0,4),
         main="Schoolfield, Sharpe & Magnuson 4-parameters", show.density=TRUE)

## End(Not run)
```

`resultNest_newp` *Fit using the nest database with anchored parameters*

Description

Fit using the nest database with anchored parameters

Usage

```
resultNest_newp
```

Format

A list with fitted information from data(nest) with anchored parameters

Details

Result of the fit using the nest database with anchored parameters

Author(s)

Marc Girondot `<marc.girondot@u-psud.fr>`

References

Girondot, M., & Kaska, Y. (2014). A model to predict the thermal reaction norm for the embryo growth rate from field data. Journal of Thermal Biology, 45, 96-102. doi: 10.1016/j.jtherbio.2014.08.005

Examples

```
## Not run:
library(embryogrowth)
data(nest)
formated <- FormatNests(nest)
newp <- GenerateAnchor(nests=formated, number.anchors=7)
pfixed <- c(rK=2.093313)
resultNest_newp <- searchR(parameters=newp, fixed.parameters=pfixed,
  temperatures=formated, derivate=dydt.Gompertz, M0=1.7,
test=c(Mean=39.33, SD=1.92))
data(resultNest_newp)
plotR(resultNest_newp)

## End(Not run)
```

`result_mcmc_newp` *Result of the mcmc using the nest database with anchored parameters*

Description

Fit using the nest database with anchored parameters

Usage

```
result_mcmc_newp
```

Format

A list of class mcmcComposite with mcmc result for data(nest) with anchored parameters

Details

Result of the mcmc using the nest database with anchored parameters

Author(s)

Marc Girondot `<marc.girondot@u-psud.fr>`

References

Girondot, M., & Kaska, Y. (2014). A model to predict the thermal reaction norm for the embryo growth rate from field data. Journal of Thermal Biology, 45, 96-102. doi: 10.1016/j.jtherbio.2014.08.005

Examples

```
## Not run:
library(embryogrowth)
data(nest)
formated <- FormatNests(nest)
newp <- GenerateAnchor(nests=formated, number.anchors=7)
pfixed <- c(rK=2.093313)
resultNest_newp <- searchR(parameters=newp, fixed.parameters=pfixed,
  temperatures=formated, derivate=dydt.Gompertz, M0=1.7,
  test=c(Mean=39.33, SD=1.92))
data(resultNest_newp)
pMCMC <- TRN_MHmcmc_p(resultNest_newp, accept=TRUE)
# Take care, it can be very long, sometimes several days
result_mcmc_newp <- GRTRN_MHmcmc(result=resultNest_newp,
  parametersMCMC=pMCMC, n.iter=10000, n.chains = 1, n.adapt = 0,
thin=1, trace=TRUE)
data(result_mcmc_newp)
data(resultNest_4p_SSM4p)
newp <- GenerateAnchor(nests=resultNest_4p_SSM4p, number.anchors=7)
# Here the confidence interval is built based on anchored parameters
plotR(resultNest_4p_SSM4p, parameters=newp, SE=result_mcmc_newp$SD,
 ylim=c(0,4), ylimH=c(0,0.4), show.hist=TRUE, curves="ML quantiles")
# Here the confidence interval is built based on parametric SSM equation
data(resultNest_4p_SSM4p)
plotR(resultNest_4p_SSM4p, SE=resultNest_mcmc_4p_SSM4p$SD,
 ylim=c(0,4), ylimH=c(0,0.4), show.hist=TRUE, curves="ML quantiles")
plot(result_mcmc_newp, las=1, xlim=c(0,30), parameters="294",
breaks=c(0, 1.00095, 2.0009, 3.00085, 4.0008, 5.00075, 6.0007, 7.00065, 8.0006, 9.00055,
10.0005, 11.00045, 12.0004, 13.00035, 14.0003, 15.00025, 16.0002, 17.00015, 18.0001,
19.00005, 20))
plot(result_mcmc_newp, las=1, xlim=c(0,30), parameters="296.333333333333")
plot(result_mcmc_newp, las=1, xlim=c(0,30), parameters=3)

## End(Not run)
```

`searchR` *Fit the parameters that best represent nest incubation data.*

Description

Fit the parameters that best represent data.
test can be a data.frame with two columns Mean and SD and rownames with the nest name.
If SD is na, then least squarre criteria is used for fitting.
Function to fit thermal reaction norm can be expressed as :

- a 4-parameters Schoolfield, Sharpe, and Magnuson model (1981) with `DHH`, `DHA`, `T12H`, and `Rho25`;

- a 6-parameters Schoolfield, Sharpe, and Magnuson model (1981) with `T12L`, `DT`, `DHH`, `DHL`, `DHA`, and `Rho25`;
- Each of these two first models can be combined as low and high sets of parameters by adding the `_L` suffix to one set. Then you must add also `transition_S` and `transition_P` parameters and then the growth rate is 1/(1+exp((1/transition_S)*(P-transition_P))) with P being the proportion of development;
- The `Rho25_b` control the effect of hygrometry (or `Rho25_b_L`) (It is not fully functional still);
- a Weibull function with `k` (shape), `lambda` (scale) and `theta` parameters;
- a normal function with `Peak`, `Scale`, and `sd` parameters;
- an asymmetric normal fuction with `Peak`, `Scale`, `sdH` and `sdL` parameters;
- a symmetric trigonometric function with `Length`, `Peak`, and `Max`;
- an asymmetric trigonometric function with `LengthB`, `LengthE`, `Peak`, and `Max`.
- Dallwitz-Higgins model (1992) can be used using `Dallwitz_b1`, `Dallwitz_b2`, `Dallwitz_b3`, `Dallwitz_b4` and `Dallwitz_b5` parameters.
- If `Dallwitz_b4` is not included, `Dallwitz_b4` = 6 will be used.
- If `Dallwitz_b5` is not included, `Dallwitz_b5` = 0.4 will be used.
- It is possible also to add the parameter `epsilon` and then the model becomes `X + epsilon` with X being any of the above model;
- It is possible also to add the parameter `epsilon_L` and then the model becomes `X_L + epsilon_L` with `X_L` being any of the above model with suffix `_L`;
- If the name of the parameter is a number, then the mdel is a polynom anchored with the rate being the parameter value at this temperature (the name). see `ChangeSSM()` function.

Usage

```
searchR(parameters = stop("Initial set of parameters must be provided"),
  fixed.parameters = NULL,
  temperatures = stop("Formated temperature must be provided !"),
  derivate = dydt.Gompertz, test = c(Mean = 39.33, SD = 1.92),
  M0 = 1.7, saveAtMaxiter = FALSE, fileName = "intermediate",
  weight = NULL, control = list(trace = 1, REPORT = 100, maxit = 500))
```

Arguments

`parameters`	A set of parameters used as initial point for searching
`fixed.parameters`	A set of parameters that will not be changed
`temperatures`	Timeseries of temperatures after formated using FormatNests()
`derivate`	Function used to fit embryo growth: dydt.Gompertz, dydt.exponential or dydt.linear
`test`	A vector with Mean and SD of size of hatchlings, ex. test=c(Mean=39, SD=3). Can be a data.frame also. See description
`M0`	Measure of hatchling size or mass proxi at laying date
`saveAtMaxiter`	If True, each time number of interation reach maxiter, current data are saved in file with filename name

`fileName`	The intermediate results are saved in file with fileName.Rdata name
`weight`	A named vector of the weight for each nest for likelihood estimation
`control`	List for control parameters for optimx

Details

searchR fits the parameters that best represent nest incubation data.

Value

A result object

Author(s)

Marc Girondot

References

Angilletta, M.J., 2006. Estimating and comparing thermal performance curves. Journal of Thermal Biology 31, 541-545.

Dallwitz, M.J., Higgins, J.P., 1992. User's guide to DEVAR. A computer program for estimating development rate as a function of temperature. CSIRO Aust Div Entomol Rep 2, 1-23.

Georges, A., Beggs, K., Young, J.E., Doody, J.S., 2005. Modelling development of reptile embryos under fluctuating temperature regimes. Physiological and Biochemical Zoology 78, 18-30.

Girondot, M., Kaska, Y., 2014. A model to predict the thermal reaction norm for the embryo growth rate from field data. Journal of Thermal Biology 45, 96-102.

Schoolfield, R.M., Sharpe, P.J., Magnuson, C.E., 1981. Non-linear regression of biological temperature-dependent rate models based on absolute reaction-rate theory. Journal of Theoretical Biology 88, 719-731.

Examples

```
## Not run:
library(embryogrowth)
data(nest)
formated <- FormatNests(nest)
# The initial parameters value can be:
# "T12H", "DHA",  "DHH", "Rho25"
# Or
# "T12L", "DT", "DHA",  "DHH", "DHL", "Rho25"
# K for Gompertz must be set as fixed parameter or being a constant K
# or relative to the hatchling size rK
x <- structure(c(105.966881676793, 613.944134764125, 306.449533440186,
                118.193882815108), .Names = c("DHA", "DHH", "T12H", "Rho25"))
# pfixed <- c(K=82.33) or rK=82.33/39.33
pfixed <- c(rK=2.093313)
resultNest_4p_SSM4p <- searchR(parameters=x, fixed.parameters=pfixed,
temperatures=formated, derivate=dydt.Gompertz, M0=1.7,
test=c(Mean=39.33, SD=1.92))
```

```
data(resultNest_4p_SSM4p)
plot(resultNest_4p_SSM4p, xlim=c(0,70), ylimT=c(22, 32), ylimS=c(0,45), series=1,
embryo.stages="Caretta caretta.SCL")
x <- structure(c(106.567809092008, 527.359011254683, 614.208632495199,
2720.94506457237, 306.268259715624, 120.336791245212), .Names = c("DHA",
"DHH", "DHL", "DT", "T12L", "Rho25"))
pfixed <- c(rK=2.093313)

# exemple of data.frame for test
ttest <- data.frame(Mean=rep(25.5, formated$IndiceT["NbTS"]),
                     SD=rep(0.75, formated$IndiceT["NbTS"]),
                     row.names=names(formated)[1:formated$IndiceT["NbTS"]])
resultNest_6p_SSM6p <- searchR(parameters=x, fixed.parameters=pfixed,
                        temperatures=formated, derivate=dydt.Gompertz, M0=1.7,
                        test=ttest)

data(resultNest_6p_SSM6p)
pMCMC <- TRN_MHmcmc_p(resultNest_6p_SSM6p, accept=TRUE)
# Take care, it can be very long, sometimes several days
resultNest_mcmc_6p_SSM6p <- GRTRN_MHmcmc(result=resultNest_6p_SSM6p,
parametersMCMC=pMCMC, n.iter=10000, n.chains = 1, n.adapt = 0,
thin=1, trace=TRUE)
data(resultNest_mcmc_6p_SSM6p)
# compare_AIC() is a function from the package "HelpersMG"
compare_AIC(test1=resultNest_4p_SSM4p, test2=resultNest_6p_SSM6p)
############ with new parametrization
data(resultNest_4p_SSM4p)
x0 <- resultNest_4p_SSM4p$par
t <- hist(resultNest_4p_SSM4p, plot=FALSE)
x <- c(3.4, 3.6, 5.4, 5.6, 7.6, 7.5, 3.2)
names(x) <- seq(from=range(t$temperatures)[1], to=range(t$temperatures)[2],
     length.out=7)
newx <- ChangeSSM(temperatures = (200:350)/10, parameters = x0,
       initial.parameters = x,
       control=list(maxit=5000))
pfixed <- c(rK=2.093313)
resultNest_newp <- searchR(parameters=newx, fixed.parameters=pfixed,
 temperatures=formated, derivate=dydt.Gompertz, M0=1.7,
 test=c(Mean=39.33, SD=1.92))
plotR(resultNest_newp, ylim=c(0, 3), xlimR=c(23, 34), ylimH=c(0, 0.3), show.hist.TRUE)
compare_AIC(test4p=resultNest_4p_SSM4p,
            test6p=resultNest_6p_SSM6p,
            testAnchor=resultNest_newp)

############################################
# example with thermal reaction norm fitted from Weibull function
############################################

 x <- ChangeSSM(temperatures = (200:350)/10,
                parameters = resultNest_4p_SSM4p$par,
                initial.parameters = structure(c(73.4009010417375, 304.142079511996,
                                                 27.4671689276281),
                                        .Names = c("k", "lambda", "scale")),
```

```
                control=list(maxit=1000))
pfixed <- c(rK=2.093313)
resultNest_3p_Weibull <- searchR(parameters=x$par, fixed.parameters=pfixed,
                         temperatures=formated, derivate=dydt.Gompertz, M0=1.7,
                         test=c(Mean=39.33, SD=1.92))
plotR(list(resultNest_4p_SSM4p, resultNest_3p_Weibull), ylim=c(0,3), col=c("Black", "red"))
compare_AIC(SSM4p=resultNest_4p_SSM4p, Weibull=resultNest_3p_Weibull)

###########################################
# example with thermal reaction norm fitted from asymmetric normal function
############################################

x <- ChangeSSM(temperatures = (200:350)/10,
               parameters = resultNest_4p_SSM4p$par,
               initial.parameters = structure(c(3, 7, 11, 32),
                               .Names = c("Scale", "sdL", "sdH", "Peak")),
               control=list(maxit=1000))
pfixed <- c(rK=2.093313)
resultNest_4p_normal <- searchR(parameters=x$par, fixed.parameters=pfixed,
                         temperatures=formated, derivate=dydt.Gompertz, M0=1.7,
                         test=c(Mean=39.33, SD=1.92))

###########################################
# example with thermal reaction norm fitted from trigonometric model
############################################

 x <- ChangeSSM(temperatures = (200:350)/10,
               parameters = resultNest_4p_SSM4p$par,
               initial.parameters = structure(c(3, 20, 40, 32),
               .Names = c("Max", "LengthB", "LengthE", "Peak")),
               control=list(maxit=1000))
pfixed <- c(rK=2.093313)
resultNest_4p_trigo <- searchR(parameters=x$par, fixed.parameters=pfixed,
                         temperatures=formated, derivate=dydt.Gompertz, M0=1.7,
                         test=c(Mean=39.33, SD=1.92))

###########################################
# example with thermal reaction norm fitted from Dallwitz model
# From Girondot, M., Monsinjon, J., Guillon, J.-M., Submitted. Delimitation of the
# embryonic thermosensitive period for sex determination using an embryo growth model
# reveals a potential bias for sex ratio prediction in turtles.
#  rK = 1.208968
#  M0 = 0.3470893
# See the example in stages datasets
############################################

x <- structure(c(4.88677476830268, 20.4051904475743, 31.5173105860335),
.Names = c("Dallwitz_b1", "Dallwitz_b2", "Dallwitz_b3"))
pfixed <- c(rK=1.208968)
resultNest_3p_Dallwitz <- searchR(parameters=x, fixed.parameters=pfixed,
                         temperatures=formated, derivate=dydt.Gompertz, M0=0.3470893,
                         test=c(Mean=39.33, SD=1.92))
plotR(resultNest_3p_Dallwitz, ylim=c(0,8))
```

```
x <- structure(c(4.91191231405918, 12.7453211281394, 31.2670410811077,
5.7449376569153, -0.825689964543813), .Names = c("Dallwitz_b1",
"Dallwitz_b2", "Dallwitz_b3", "Dallwitz_b4", "Dallwitz_b5"))
pfixed <- c(rK=1.208968)
resultNest_5p_Dallwitz <- searchR(parameters=x, fixed.parameters=pfixed,
                         temperatures=formated, derivate=dydt.Gompertz, M0=0.3470893,
                         test=c(Mean=39.33, SD=1.92))
plotR(resultNest_5p_Dallwitz, ylim=c(0,8))

xp <- resultNest_6p_SSM6p$par
xp["Rho25"] <- 233
pfixed <- c(rK=1.208968)
resultNest_6p_SSM <- searchR(parameters=xp, fixed.parameters=pfixed,
                         temperatures=formated, derivate=dydt.Gompertz, M0=0.3470893,
                         test=c(Mean=39.33, SD=1.92))
plotR(resultNest_6p_SSM, ylim=c(0,8))

xp <- ChangeSSM(parameters = resultNest_3p_Dallwitz$par,
                initial.parameters = resultNest_4p_SSM4p$par)
pfixed <- c(rK=1.208968)
resultNest_4p_SSM <- searchR(parameters=xp$par, fixed.parameters=pfixed,
                         temperatures=formated, derivate=dydt.Gompertz, M0=0.3470893,
                         test=c(Mean=39.33, SD=1.92))
plotR(resultNest_4p_SSM, ylim=c(0,8))

compare_AIC(Dallwitz3p=resultNest_3p_Dallwitz, Dallwitz5p=resultNest_5p_Dallwitz,
             SSM4p=resultNest_4p_SSM, SSM6p=resultNest_6p_SSM)

###########################################
# Example with thermal reaction norm of proportion of development
# fitted from Dallwitz model
# see Woolgar, L., Trocini, S., Mitchell, N., 2013. Key parameters describing
# temperature-dependent sex determination in the southernmost population of loggerhead
# sea turtles. Journal of Experimental Marine Biology and Ecology 449, 77-84.
############################################

x <- structure(c(1.48207559695689, 20.1100310234046, 31.5665036287242),
 .Names = c("Dallwitz_b1", "Dallwitz_b2", "Dallwitz_b3"))
resultNest_PropDev_3p_Dallwitz <- searchR(parameters=x, fixed.parameters=NULL,
                         temperatures=formated, derivate=dydt.linear, M0=0,
                         test=c(Mean=1, SD=NA))
 plotR(resultNest_PropDev_3p_Dallwitz, ylim=c(0, 1.5), curves="ML")
 plot(x=resultNest_PropDev_3p_Dallwitz, ylimS=c(0,1), xlim=c(0,60), series=2,
         embryo.stages="Generic.ProportionDevelopment")

x <- structure(c(1.48904182113431, 10.4170365155993, 31.2591665490154,
6.32355497589913, -1.07425378667104), .Names = c("Dallwitz_b1",
"Dallwitz_b2", "Dallwitz_b3", "Dallwitz_b4", "Dallwitz_b5"))
resultNest_PropDev_5p_Dallwitz <- searchR(parameters=x, fixed.parameters=NULL,
                         temperatures=formated, derivate=dydt.linear, M0=0,
                         test=c(Mean=1, SD=NA))
 plotR(resultNest_PropDev_5p_Dallwitz, ylim=c(0, 1.5))
```

```
plot(x=resultNest_PropDev_5p_Dallwitz, ylimS=c(0,1), xlim=c(0,60), series=2,
        embryo.stages="Generic.ProportionDevelopment")

plotR(resultNest_PropDev_3p_Dallwitz, ylim=c(0, 1.5), curves="ML")
plotR(resultNest_PropDev_5p_Dallwitz, ylim=c(0, 1.5), curves="ML", new=FALSE, col="red")
compare_AICc(Dallwitz3p=resultNest_PropDev_3p_Dallwitz,
             Dallwitz5p=resultNest_PropDev_5p_Dallwitz)


## End(Not run)
```

`stages` *Database of of embryonic development and thermosensitive period of development for sex determination*

Description

Database of embryonic development and thermosensitive period of development for sex determination.

Usage

```
stages
```

Format

A list with dataframes including attributes

Details

Database of embryonic development and thermosensitive period of development for sex determination

Author(s)

Marc Girondot `<marc.girondot@u-psud.fr>`

References

Pieau, C., Dorizzi, M., 1981. Determination of temperature sensitive stages for sexual differentiation of the gonads in embryos of the turtle, Emys orbicularis. Journal of Morphology 170, 373-382.

Yntema, C.L., Mrosovsky, N., 1982. Critical periods and pivotal temperatures for sexual differentiation in loggerhead sea turtles. Canadian Journal of Zoology-Revue Canadienne de Zoologie 60, 1012-1016.

Kaska, Y., Downie, R., 1999. Embryological development of sea turtles (Chelonia mydas, Caretta caretta) in the Mediterranean. Zoology in the Middle East 19, 55-69.

Greenbaum, E., 2002. A standardized series of embryonic stages for the emydid turtle Trachemys scripta. Canadian Journal of Zoology-Revue Canadienne de Zoologie 80, 1350-1370.

Magalhães, M.S., Vogt, R.C., Sebben, A., Dias, L.C., de Oliveira, M.F., de Moura, C.E.B., 2017. Embryonic development of the Giant South American River Turtle, Podocnemis expansa (Testudines: Podocnemididae). Zoomorphology.

See Also

Other Functions for temperature-dependent sex determination: `DatabaseTSD`, `P_TRT`, `TSP.list`, `plot.tsd`, `predict.tsd`, `tsd_MHmcmc_p`, `tsd_MHmcmc`, `tsd`

Examples

```
## Not run:
library(embryogrowth)
data(stages)
names(stages)
levels(as.factor(stages$Species))
# Version of database
stages$Version[1]
kaska99.SCL <- subset(stages, subset=(Species == "Caretta caretta"),
         select=c("Stage", "SCL_Mean_mm", "SCL_SD_mm", "Days_Begin", "Days_End"))

kaska99.SCL[kaska99.SCL$Stage==31, "Days_Begin"] <- 51
kaska99.SCL[kaska99.SCL$Stage==31, "Days_End"] <- 62
kaska99.SCL <- na.omit(kaska99.SCL)
kaska99.SCL[which(kaska99.SCL$Stage==31), "Stage"] <- c("31a", "31b", "31c")
kaska99.SCL <- cbind(kaska99.SCL,
                    Days_Mean=(kaska99.SCL[, "Days_Begin"]+kaska99.SCL[, "Days_End"])/2)
kaska99.SCL <- cbind(kaska99.SCL,
                     Days_SD=(kaska99.SCL[, "Days_End"]-kaska99.SCL[, "Days_Begin"])/4)
Gompertz <- function(x, par) {
   K <- par["K"]
   rT <- par["rT"]
   X0 <- par["X0"]
   y <- abs(K)*exp(log(abs(X0)/abs(K))*exp(-rT*x))
   return(y)
 }

ML.Gompertz <- function(x, par) {
  par <- abs(par)
  y <- Gompertz(x, par)
  return(sum(-dnorm(y, mean=kaska99.SCL[, "SCL_Mean_mm"],
                    sd=kaska99.SCL[, "SCL_SD_mm"], log=TRUE)))
}

parIni <- structure(c(48.66977358, 0.06178453, 0.38640902),
                   .Names = c("K", "rT", "X0"))

fitsize.SCL <- optim(parIni, ML.Gompertz, x=kaska99.SCL[, "Days_Mean"], hessian = TRUE)

# Estimation of standard error of parameters using Hessian matrix
sqrt(diag(solve(fitsize.SCL$hessian)))
```

```
# Estimation of standard error of parameters using Bayesian  concept and MCMC
pMCMC <- structure(list(Density = c("dunif", "dunif", "dunif"),
                        Prior1 = c(0, 0, 0), Prior2 = c(90, 1, 2),
                        SDProp = c(1, 1, 1),
                        Min = c(0, 0, 0), Max = c(90, 1, 2),
                        Init = fitsize.SCL$par),
                .Names = c("Density", "Prior1", "Prior2", "SDProp", "Min", "Max", "Init"),
                   row.names = c("K", "rT", "X0"), class = "data.frame")

Bayes.Gompertz <- function(data, x) {
  x <- abs(x)
  y <- Gompertz(data, x)
  return(sum(-dnorm(y, mean=kaska99.SCL[, "SCL_Mean_mm"],
                    sd=kaska99.SCL[, "SCL_SD_mm"], log=TRUE)))
}

mcmc_run <- MHalgoGen(n.iter=50000, parameters=pMCMC, data=kaska99.SCL[, "Days_Mean"],
                    likelihood=Bayes.Gompertz, n.chains=1, n.adapt=100, thin=1, trace=1,
                      adaptive = TRUE)

plot(mcmc_run, xlim=c(0, 90), parameters="K")
plot(mcmc_run, xlim=c(0, 1), parameters="rT")
plot(mcmc_run, xlim=c(0, 2), parameters="X0")

1-rejectionRate(as.mcmc(mcmc_run))

par <- mcmc_run$resultMCMC[[1]]

outsp <- t(apply(par, MARGIN = 1, FUN=function(x) Gompertz(0:70, par=x)))

rangqtiles <- apply(outsp, MARGIN=2, function(x) {quantile(x, probs=c(0.025, 0.5, 0.975))})

par(mar=c(4, 4, 2, 1))
plot_errbar(x=kaska99.SCL[, "Days_Mean"], y=kaska99.SCL[, "SCL_Mean_mm"],
            errbar.y = 2*kaska99.SCL[, "SCL_SD_mm"], bty="n", las=1,
            ylim=c(0, 50), xlab="Days", ylab="SCL mm",
           xlim=c(0, 70), x.plus = kaska99.SCL[, "Days_End"],
            x.minus = kaska99.SCL[, "Days_Begin"])

lines(0:70, rangqtiles["2.5%", ], lty=2)
lines(0:70, rangqtiles["97.5%", ], lty=2)
lines(0:70, rangqtiles["50%", ], lty=3)

text(x=50, y=10, pos=4, labels=paste("K=", format(x = fitsize.SCL$par["K"], digits = 4)))
text(x=50, y=12.5, pos=4,
   labels=paste("rK=", format(x = fitsize.SCL$par["K"]/39.33, digits = 4)))
text(x=50, y=15, pos=4, labels=paste("X0=", format(x = fitsize.SCL$par["X0"], digits = 4)))
title("Univariate normal distribution")

# Using a multivariate normal distribution

library(mvtnorm)
```

```
 ML.Gompertz.2D <- function(x, par) {
   par <- abs(par)
  y <- Gompertz(x, par)
  L <- 0
  for (i in seq_along(y)) {
    sigma <- matrix(c(kaska99.SCL$SCL_SD_mm[i]^2, 0, 0, kaska99.SCL$Days_SD[i]^2),
                    nrow=2, byrow=TRUE,
                    dimnames=list(c("SCL_SD_mm", "Days_SD"), c("SCL_SD_mm", "Days_SD")))
    L <- L -dmvnorm(x=c(SCL_SD_mm=kaska99.SCL$SCL_Mean_mm[i],
                    Days_SD=kaska99.SCL$Days_Mean[i]),
                    mean= c(SCL_SD_mm=y[i], Days_SD=kaska99.SCL$Days_Mean[i]),
                            sigma=sigma, log=TRUE)
  }
  return(L)
}

parIni <- structure(c(48.66977358, 0.06178453, 0.38640902),
                    .Names = c("K", "rT", "X0"))

fitsize.SCL.2D <- optim(parIni, ML.Gompertz.2D, x=kaska99.SCL[, "Days_Mean"], hessian = TRUE)

# Estimation of standard error of parameters using Hessian matrix
sqrt(diag(solve(fitsize.SCL.2D$hessian)))

# Estimation of standard error of parameters using Bayesian  concept and MCMC
Bayes.Gompertz.2D <- function(data, x) {
  x <- abs(x)
  y <- Gompertz(data, x)
  L <- 0
  for (i in seq_along(y)) {
    sigma <- matrix(c(kaska99.SCL$SCL_SD_mm[i]^2, 0, 0, kaska99.SCL$Days_SD[i]^2),
                    nrow=2, byrow=TRUE,
                    dimnames=list(c("SCL_SD_mm", "Days_SD"), c("SCL_SD_mm", "Days_SD")))
    L <- L - dmvnorm(x=c(SCL_SD_mm=kaska99.SCL$SCL_Mean_mm[i],
                         Days_SD=kaska99.SCL$Days_Mean[i]),
                    mean= c(SCL_SD_mm=y[i], Days_SD=kaska99.SCL$Days_Mean[i]),
                    sigma=sigma, log=TRUE)
  }
  return(L)
}

pMCMC <- structure(list(Density = c("dunif", "dunif", "dunif"),
                        Prior1 = c(0, 0, 0), Prior2 = c(90, 1, 2),
                        SDProp = c(1, 1, 1),
                        Min = c(0, 0, 0), Max = c(90, 1, 2),
                        Init = fitsize.SCL.2D$par),
               .Names = c("Density", "Prior1", "Prior2", "SDProp", "Min", "Max", "Init"),
                   row.names = c("K", "rT", "X0"), class = "data.frame")
mcmc_run.2D <- MHalgoGen(n.iter=50000, parameters=pMCMC, data=kaska99.SCL[, "Days_Mean"],
                  likelihood=Bayes.Gompertz.2D, n.chains=1, n.adapt=100, thin=1, trace=1,
                      adaptive = TRUE)

plot(mcmc_run.2D, xlim=c(0, 90), parameters="K")
```

```
plot(mcmc_run.2D, xlim=c(0, 1), parameters="rT")
plot(mcmc_run.2D, xlim=c(0, 2), parameters="X0")

1-rejectionRate(as.mcmc(mcmc_run.2D))

par <- mcmc_run.2D$resultMCMC[[1]]

outsp <- t(apply(par, MARGIN = 1, FUN=function(x) Gompertz(0:70, par=x)))

rangqtiles <- apply(outsp, MARGIN=2, function(x) {quantile(x, probs=c(0.025, 0.5, 0.975))})

par(mar=c(4, 4, 2, 1))
plot_errbar(x=kaska99.SCL[, "Days_Mean"], y=kaska99.SCL[, "SCL_Mean_mm"],
            errbar.y = 2*kaska99.SCL[, "SCL_SD_mm"], bty="n", las=1,
            ylim=c(0, 50), xlab="Days", ylab="SCL mm",
           xlim=c(0, 70), x.plus = kaska99.SCL[, "Days_End"],
            x.minus = kaska99.SCL[, "Days_Begin"])

lines(0:70, rangqtiles["2.5%", ], lty=2)
lines(0:70, rangqtiles["97.5%", ], lty=2)
lines(0:70, rangqtiles["50%", ], lty=3)

text(x=50, y=10, pos=4,
     labels=paste("K=", format(x = fitsize.SCL.2D$par["K"], digits = 4)))
text(x=50, y=12.5, pos=4,
     labels=paste("rK=", format(x = fitsize.SCL.2D$par["K"]/39.33, digits = 4)))
text(x=50, y=15, pos=4,
     labels=paste("X0=", format(x = fitsize.SCL.2D$par["X0"], digits = 4)))
title("Multivariate normal distribution")


## End(Not run)
```

`STRN` *Estimate the parameters that best describe the sexualisation thermal reaction norm within the TSP*

Description

Estimate the parameters that best describe the sexualisation thermal reaction norm within the TSP. The Temperatures parameter is a character string which can be:

- `TimeWeighted.temperature.mean`
- `TSP.TimeWeighted.temperature.mean`
- `TSP.MassWeighted.temperature.mean`
- `TSP.STRNWeighted.temperature.mean`
- `TSP.MassWeighted.STRNWeighted.temperature.mean`

- `MiddleThird.TimeWeighted.temperature.mean`

If information for sex is not known for some timeseries, set NA for Sexed.
Sexed, Males and Females must be vectors with names. The names must be the same as the names of timeseries of temperatures in EmbryoGrowthTRN.
Only two of these 3 parameters are required: Males, Females and Sexed
Note: four species have predefined embryo stages. embryo.stages parameter can take the values:

- `Caretta caretta.SCL`
- `Chelonia mydas.SCL`
- `Emys orbicularis.SCL`
- `Emys orbicularis.mass`
- `Podocnemis expansa.SCL`
- `Generic.ProportionDevelopment`

Usage

```
STRN(Initial_STRN = NULL, fixed.parameters = NULL,
  EmbryoGrowthTRN = stop("Embryo Growth Thermal Reaction Norm must be provided"),
  TSP.borders = NULL, embryo.stages = "Generic.ProportionDevelopment",
  tsd = NULL, equation = "logistic", Sexed = NULL, Males = NULL,
  Females = NULL,
  Temperatures = "TSP.MassWeighted.STRNWeighted.temperature.mean",
  SE = TRUE, parallel = TRUE, itnmax = 1000,
  method = c("Nelder-Mead", "BFGS"), control = list(trace = 1, REPORT =
  10), zero = 1e-09)
```

Arguments

`Initial_STRN`	Values for initial model of Sexualisation Thermal Reaction Norm
`fixed.parameters`	Value for Sexualisation Thermal Reaction Norm model that will not be changed
`EmbryoGrowthTRN`	The Embryo Growth Thermal Reaction Norm obtained with searchR()
`TSP.borders`	The limits of TSP in stages. See embryo.stages parameter.
`embryo.stages`	The embryo stages. At least TSP.borders stages must be provided to estimate TSP borders. See note.
`tsd`	The model used to predict sex ratio, obtained from tsd()
`equation`	If tsd parameter is not provided, equation and parameters for tsd model must be provided.
`Sexed`	The number of sexed embryos with names identifying timeseries
`Males`	The number of males embryos with names identifying timeseries
`Females`	The number of females embryos with names identifying timeseries
`Temperatures`	The temperature from out of info.nests to be used

`SE`	Should standard error of parameters and Hessian matrix be estimated ? TRUE or FALSE
`parallel`	Should parallel computing for info.nests() be used
`itnmax`	Maximum number of iterations for each method
`method`	Methods to be used with optimx
`control`	List for control parameters for optimx
`zero`	The value to replace a null sex ratio

Details

STRN estimates the parameters that best describe the sexualisation thermal reaction norm within the TSP

Value

The list with object return by optim()

Author(s)

Marc Girondot

Examples

```
## Not run:
library(embryogrowth)
MedIncubation_Cc <- subset(DatabaseTSD, Species=="Caretta caretta" &
RMU=="Mediterranean" & Sexed!=0)
Med_Cc <- with(MedIncubation_Cc, tsd(males=Males, females=Females,
 temperatures=Incubation.temperature, par=c(P=29.5, S=-0.01)))
plot(Med_Cc, xlim=c(25, 35))
males <- c(7, 0, 0, 0, 0, 5, 6, 3, 5, 3, 2, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0)
names(males) <- rev(rev(names(resultNest_4p_SSM4p$data))[-(1:2)])
sexed <- rep(10, length(males))
names(sexed) <- rev(rev(names(resultNest_4p_SSM4p$data))[-(1:2)])
Initial_STRN <- resultNest_4p_SSM4p$par[c("DHA_STRN", "DHH_STRN", "T12H_STRN")]
Initial_STRN <- structure(c(582.567096666926, 2194.0806711639, 3475.28414940385),
                          .Names = c("DHA_STRN", "DHH_STRN", "T12H_STRN"))
fp <- c(Rho25_STRN=100)
fitSTRN <- STRN(Initial_STRN=Initial_STRN,
                EmbryoGrowthTRN=resultNest_4p_SSM4p, tsd=Med_Cc,
                embryo.stages="Caretta caretta.SCL",
                Sexed=sexed, Males=males,
                fixed.parameters=fp,
                SE=TRUE,
                Temperatures="TSP.MassWeighted.STRNWeighted.temperature.mean")
plotR(fitSTRN, curves ="ML quantiles", ylim=c(0,2))
CTE <- info.nests(NestsResult=resultNest_4p_SSM4p,
                  SexualisationTRN=fitSTRN,
                  SexualisationTRN.CI="Hessian",
                  embryo.stages="Caretta caretta.SCL",
```

```
                  CI="Hessian",
                  replicate.CI=100,
                  progress=TRUE,
                  warnings=TRUE,
                  out="summary")$summary
# CTE with growth-weighted temperature average
plot(Med_Cc, xlim=c(25, 35))
points(x=CTE$TSP.MassWeighted.temperature.mean, y=males/sexed,
         col="red", pch=19)
legend("topright", legend=c("CTE with growth-weighted TRN"),
         pch=19, col=c("red"))
# CTE with sexualisation TRN and growth-weighted temperature average
plot(Med_Cc, xlim=c(25, 35))
points(x=CTE$TSP.MassWeighted.STRNWeighted.temperature.mean, y=males/sexed,
         col="red", pch=19)
legend("topright", legend=c("CTE with growth-weighted TRN and Sex. TRN"),
       pch=19, col=c("red"))
xx <- seq(from=20, to=35, by=0.1)
plot(x=xx,
         y=log10(getFromNamespace(".SSM", ns="embryogrowth")(xx,
                           c(fitSTRN$par, fitSTRN$fixed.parameters))[[1]]),
         type="l", bty="n", xlim=c(20, 35), ylim=c(-20, 20),
         xlab="Temperature", ylab="Sexualisation thermal reaction norm (log10)")

# Using only the sexualisation thermal reaction norm within TSP to calculate CTE

Initial_STRN <- resultNest_4p_SSM4p$par[c("DHA_STRN", "DHH_STRN", "T12H_STRN")]
Initial_STRN <- structure(c(3678.94960547096, -301.436485427701, 912.595953854977),
                         .Names = c("DHA_STRN", "DHH_STRN", "T12H_STRN"))
fp <- c(Rho25_STRN=100)
fitSTRN_2 <- STRN(Initial_STRN=Initial_STRN,
                EmbryoGrowthTRN=resultNest_4p_SSM4p, tsd=Med_Cc,
                embryo.stages="Caretta caretta.SCL",
                Sexed=sexed, Males=males,
                fixed.parameters=fp,
                Temperatures="TSP.STRNWeighted.temperature.mean")
CTE <- info.nests(NestsResult=resultNest_4p_SSM4p,
                  SexualisationTRN=fitSTRN_2,
                  SexualisationTRN.CI="Hessian",
                  embryo.stages="Caretta caretta.SCL",
                  CI="Hessian",
                  replicate.CI=100,
                  progress=TRUE,
                  warnings=TRUE,
                  out="summary")$summary
# CTE with sexualisation TRN
plot(Med_Cc, xlim=c(25, 35))
points(x=CTE$TSP.STRNWeighted.temperature.mean, y=males/sexed,
         col="red", pch=19)
legend("topright", legend=c("CTE with Sexualisation TRN"),
       pch=19, col=c("red"))
xx <- seq(from=20, to=35, by=0.1)
plot(x=xx,
```

```
        y=getFromNamespace(".SSM", ns="embryogrowth")(xx,
                          c(fitSTRN$par, fitSTRN$fixed.parameters))[[1]],
        type="l", bty="n", xlim=c(20, 35), ylim=c(0, 1E-18),
        xlab="Temperature", ylab="Sexualisation thermal reaction norm")

## End(Not run)
```

`STRN_MHmcmc` *Metropolis-Hastings algorithm for Sexualisation Thermal Reaction Norm*

Description

Run the Metropolis-Hastings algorithm for Sexualisation Thermal Reaction Norm.
The number of iterations is n.iter+n.adapt+1 because the initial likelihood is also displayed.
I recommend that thin=1 because the method to estimate SE uses resampling.
If initial point is maximum likelihood, n.adapt = 0 is a good solution.
To get the SE of the point estimates from `result_mcmc <-STRN_MHmcmc(result=try)`, use: `result_mcmc$SD`
coda package is necessary for this function.
The dataSTRN is a named list with the following objects:

- EmbryoGrowthTRN= result of `searchR`
- tsd= result of `tsd`
- sexed= vector with number of sexed embryos
- males= vector with number of males (could be also females=)
- Temperatures= a text of the temperatures name used as CTE

The Temperatures text for CTE can be:

- `TimeWeighted.temperature.mean`
- `TSP.TimeWeighted.temperature.mean`
- `TSP.MassWeighted.temperature.mean`
- `TSP.STRNWeighted.temperature.mean`
- `TSP.MassWeighted.STRNWeighted.temperature.mean`
- `MiddleThird.TimeWeighted.temperature.mean`

They are explained in the `info.nests` function.
This function is not still fully described as it has not been published still.
The parameters intermediate and filename are used to save intermediate results every 'intermediate' iterations (for example 1000). Results are saved in a file of name filename.
The parameter previous is used to indicate the list that has been save using the parameters intermediate and filename. It permits to continue a mcmc search.
These options are used to prevent the consequences of computer crash or if the run is very very long and processes at time limited.

Usage

```
STRN_MHmcmc(result = NULL, n.iter = 10000, parametersMCMC = NULL,
  n.chains = 1, n.adapt = 0, thin = 1, trace = NULL,
  traceML = FALSE, batchSize = sqrt(n.iter), dataSTRN = NULL,
  adaptive = FALSE, adaptive.lag = 500, adaptive.fun = function(x) {
     ifelse(x > 0.234, 1.3, 0.7) }, parallel = TRUE,
  intermediate = NULL, filename = "intermediate.Rdata",
  previous = NULL)
```

Arguments

`result`	An object obtained after a STRN fit
`n.iter`	Number of iterations for each step
`parametersMCMC`	A set of parameters used as initial point for searching with information on priors
`n.chains`	Number of replicates
`n.adapt`	Number of iterations before to store outputs
`thin`	Number of iterations between each stored output
`trace`	TRUE or FALSE or period, shows progress
`traceML`	TRUE or FALSE to show ML
`batchSize`	Number of observations to include in each batch fo SE estimation
`dataSTRN`	A named list data used to estimate likelihoods (see further in description)
`adaptive`	Should an adaptive process for SDProp be used
`adaptive.lag`	Lag to analyze the SDProp value in an adaptive content
`adaptive.fun`	Function used to change the SDProp
`parallel`	Should parallel computing for info.nests() be used
`intermediate`	Period for saving intermediate result, NULL for no save
`filename`	If intermediate is not NULL, save intermediate result in this file
`previous`	Previous result to be continued. Can be the filename in which intermediate results are saved.

Details

STRN_MHmcmc runs the Metropolis-Hastings algorithm for STRN (Bayesian MCMC)

Value

A list with resultMCMC being mcmc.list object, resultLnL being likelihoods and parametersMCMC being the parameters used

Author(s)

Marc Girondot

Examples

```
## Not run:
library(embryogrowth)
MedIncubation_Cc <- subset(DatabaseTSD, Species=="Caretta caretta" &
RMU=="Mediterranean" & Sexed!=0)
Med_Cc <- with(MedIncubation_Cc, tsd(males=Males, females=Females,
 temperatures=Incubation.temperature, par=c(P=29.5, S=-0.01)))
plot(Med_Cc, xlim=c(25, 35))
males <- c(7, 0, 0, 0, 0, 5, 6, 3, 5, 3, 2, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0)
names(males) <- rev(rev(names(resultNest_4p_SSM4p$data))[-(1:2)])
sexed <- rep(10, length(males))
names(sexed) <- rev(rev(names(resultNest_4p_SSM4p$data))[-(1:2)])
Initial_STRN <- resultNest_4p_SSM4p$par[c("DHA", "DHH", "T12H")]
Initial_STRN <- structure(c(3460.21379985491, 588.062535503578, 2347.70617453574),
                          .Names = c("DHA", "DHH", "T12H"))
fp <- c(Rho25=100)
fitSTRN <- STRN(Initial_STRN, EmbryoGrowthTRN=resultNest_4p_SSM4p, tsd=Med_Cc,
                Sexed=sexed, Males=males,
                fixed.parameters=fp,
                Temperatures="TSP.MassWeighted.STRNWeighted.temperature.mean")
pMCMC <- TRN_MHmcmc_p(fitSTRN, accept=TRUE)
pMCMC[, "Density"] <- "dunif"
pMCMC[, "Prior2"] <- pMCMC[, "Max"]<- 10000
pMCMC[, "Prior1"] <- pMCMC[, "Min"] <- 1
outMCMC <- STRN_MHmcmc(result = fitSTRN, n.iter = 50000, parametersMCMC = pMCMC,
                n.chains = 1, n.adapt = 0, thin = 1, trace = TRUE,
                dataSTRN = list(EmbryoGrowthTRN = resultNest_4p_SSM4p,
                   Temperatures = "TSP.STRNWeighted.temperature.mean",
                   fixed.parameters=fitSTRN$fixed.parameters,
                   zero=1E-9,
                   tsd=Med_Cc,
                   Sexed=sexed, Males=males),
                adaptive = TRUE, adaptive.lag = 500,
                intermediate = 1000,
                filename = "intermediate_mcmcSTRN.Rdata")
plot(outMCMC, parameters=1)
plot(outMCMC, parameters=2)
plot(outMCMC, parameters=3)
1-rejectionRate(as.mcmc(x = outMCMC))
# Take care,you computer will be 100% busy as all cores will be used intensively
outMCMC_parallel <- parallel::mclapply(1:detectCores(), function(x) {
                STRN_MHmcmc(result = fitSTRN, n.iter = 50000/detectCores(),
                parametersMCMC = pMCMC,
                n.chains = 1, n.adapt = 0, thin = 1, trace = TRUE,
                dataSTRN = list(EmbryoGrowthTRN = resultNest_4p_SSM4p,
                   Temperatures = "TSP.STRNWeighted.temperature.mean",
                   fixed.parameters=fitSTRN$fixed.parameters,
                   tsd=Med_Cc, zero=1E-9,
                   Sexed=sexed, Males=males),
                parallel=FALSE,
                adaptive = TRUE, adaptive.lag = 500,
                intermediate = NULL)
```

```
})
outMCMC_parallel_merge <- outMCMC_parallel[[1]]
for (i in 2:length(outMCMC_parallel)) {
  outMCMC_parallel_merge <- merge(outMCMC_parallel_merge, outMCMC_parallel[[i]])
}
plot(outMCMC_parallel_merge, parameters=1)
plot(outMCMC_parallel_merge, parameters=2)
plot(outMCMC_parallel_merge, parameters=3)

plotR(parameters = fitSTRN$par, fixed.parameters=fitSTRN$fixed.parameters,
      curves = "MCMC quantiles", ylim=c(0, 5), resultmcmc = outMCMC_parallel_merge,
      ylab="Relative contribution to sexualisation", xlim=c(28, 29))


## End(Not run)
```

`summary.Nests` *Summarize the information from a Nests object.*

Description

Summarize the information from a Nests object:
- Name of the nests, total incubation length and average temperature

Usage

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

Arguments

`object`	A object obtained after FormatNests()
`...`	Not used

Details

summary.Nests Summarize the information from a Nests object

Value

None

Author(s)

Marc Girondot

Examples

```
## Not run:
library(embryogrowth)
data(nest)
formated <- FormatNests(nest, previous=NULL)
summary(formated)

## End(Not run)
```

`switch.transition` *Add a transition parameter on a set of parameters or remove it*

Description

Add a transition parameter on a set of parameters or remove it

Usage

```
switch.transition(parameters = stop("A set of parameters must be supplied"))
```

Arguments

`parameters`	A vector with parameters

Details

switch.transition Add a transition parameter on a set of parameters or remove it

Value

A vector with parameters

Author(s)

Marc Girondot

Examples

```
## Not run:
data(resultNest_6p_SSM6p)
# Get a set of parameters without transition
x1 <- resultNest_6p_SSM6p$par
# Generate a set of parameters with transition
x2 <- switch.transition(x1)
# Generate a set of parameters without transition
x3 <- switch.transition(x3)

## End(Not run)
```

`tempConst` *Timeseries of constant temperatures for nests*

Description

Timeseries of temperatures for nests

Usage

```
tempConst
```

Format

A dataframe with raw data.

Details

Timeseries of constant temperatures for nests

Author(s)

Marc Girondot `<marc.girondot@u-psud.fr>`

References

Girondot, M., Kaska, Y., 2014. A model to predict the thermal reaction norm for the embryo growth rate from field data. Journal of Thermal Biology 45, 96-102.

Examples

```
## Not run:
library(embryogrowth)
# Same as:
# GenerateConstInc(durations = rep(104*60*24, 11),
# temperatures = 25:35,
# names = paste0("T",25:35))
data(tempConst)
tempConst_f <- FormatNests(tempConst)

data(nest)
formated <- FormatNests(nest)
x <- structure(c(109.683413821537, 614.969219372661, 306.386903812694,
 229.003478775323), .Names = c("DHA", "DHH", "T12H", "Rho25"))

 # See the stages dataset examples for justification of M0 and rK

pfixed <- c(rK=1.208968)
resultNest_4p_SSM <- searchR(parameters=x, fixed.parameters=pfixed,
temperatures=formated, derivate=dydt.Gompertz, M0=0.3470893,
```

```
test=c(Mean=39.33, SD=1.92))

plotR(result=resultNest_4p_SSM, show.hist = TRUE,
             ylim=c(0, 8), curves="ML quantiles")

# Now use the fited parameters from resultNest_4p_SSM with
# the constant incubation temperatures:

plot(resultNest_4p_SSM, temperatures=tempConst_f,
stopattest=TRUE, series="T30", xlim=c(0,50),
ylimT=c(22, 32), test=c(Mean=39.33, SD=1.92),
embryo.stages="Caretta caretta.SCL")

plot(resultNest_4p_SSM, temperatures=tempConst_f,
stopattest=TRUE, series="T25", xlim=c(0,120),
ylimT=c(22, 32), test=c(Mean=39.33, SD=1.92),
embryo.stages="Caretta caretta.SCL")


## End(Not run)
```

`test.parallel` *Estimate the likelihood of a set of parameters for nest incubation data with or without parallel computing option*

Description

Estimate the likelihood of a set of parameters for nest incubation data with or without parallel computing option. It uses the user time from the print result of system.time() function.

Usage

```
test.parallel(result = stop("A ResultNest object must be provided"))
```

Arguments

`result`	A object obtained after searchR or likelihoodR

Details

test.parallel estimates the likelihood of a set of parameters for nest incubation data with or without parallel computing option

Value

The gain or loss of computing time using parallel version

Author(s)

Marc Girondot

Examples

```
## Not run:
library(embryogrowth)
data(resultNest_4p_SSM4p)
test.parallel(resultNest_4p_SSM4p)

## End(Not run)
```

`TRN_MHmcmc_p` *Generates set of parameters to be used with GRTRN_MHmcmc() or STRN_MHmcmc()*

Description

Interactive script used to generate set of parameters to be used with GRTRN_MHmcmc() or STRN_MHmcmc().

Usage

```
TRN_MHmcmc_p(result = NULL, parameters = NULL,
  fixed.parameters = NULL, accept = FALSE)
```

Arguments

`result`	An object obtained after a SearchR fit
`parameters`	A set of parameters. Replace the one from result
`fixed.parameters`	A set of fixed parameters. Replace the one from result
`accept`	If TRUE, the script does not wait user information

Details

TRN_MHmcmc_p generates set of parameters to be used with GRTRN_MHmcmc() or STRN_MHmcmc()

Value

A matrix with the parameters

Author(s)

Marc Girondot

Examples

```
## Not run:
library(embryogrowth)
data(nest)
formated <- FormatNests(nest)
# The initial parameters value can be:
# "T12H", "DHA",  "DHH", "Rho25"
# Or
# "T12L", "T12H", "DHA",  "DHH", "DHL", "Rho25"
x <- structure(c(118.768297442004, 475.750095909406, 306.243694918151,
116.055824800264), .Names = c("DHA", "DHH", "T12H", "Rho25"))
# pfixed <- c(K=82.33) or rK=82.33/39.33
pfixed <- c(rK=2.093313)
resultNest_4p_SSM4p <- searchR(parameters=x, fixed.parameters=pfixed,
temperatures=formated, derivate=dydt.Gompertz, M0=1.7,
test=c(Mean=39.33, SD=1.92))
data(resultNest_4p_SSM4p)
pMCMC <- TRN_MHmcmc_p(resultNest_4p_SSM4p, accept=TRUE)

## End(Not run)
```

`tsd` *Estimate the parameters that best describe temperature-dependent sex determination*

Description

Estimate the parameters that best describe the thermal reaction norm for sex ratio when temperature-dependent sex determination occurs.
It can be used also to evaluate the relationship between incubation duration and sex ratio.
The parameter l was defined in Girondot (1999). The TRT is defined from the difference between the two boundary temperatures giving sex ratios of l and $1 - l$, respectively:
For logistic model (Girondot, 1999), it follows

$$TRT_l = abs\left(S\ K_l\right)$$

where K_l is a constant equal to $2\ log\left(\frac{l}{1-l}\right)$.
In Girondot (1999), l was 0.05 and then the TRT was defined as being the range of temperatures producing from 5% to 95% of each sex.
For other models, TRT is calculated numerically.
The basic model is logistic one. This model has the particularity to have a symmetric shape around P.
The other models have been built to alleviate this constraint. Hill and A-logistic models can be asymmetric, but it is impossible to control independently the low and high transitions.
Hulin model is assymmetric but the control of asymmetry is difficult to manage./cr If asymmetric model is selected, it is always better to use flexit model.

$$ifdose < Pthen(1 + (2^K1 - 1) * exp(4 * S1 * (P - x)))^{(} - 1/K1)$$

$$ifdose > Pthen1 - ((1 + (2^K2 - 1) * exp(4 * S2 * (x - P)))^(- 1/K2)$$

with:

$$S1 = S/((4/K1) * (2^(- K1))^(1/K1 + 1) * (2^K1 - 1))$$
$$S2 = S/((4/K2) * (2^(- K2))^(1/K2 + 1) * (2^K2 - 1))$$

Usage

```
tsd(df = NULL, males = NULL, females = NULL, N = NULL,
  temperatures = NULL, durations = NULL, l = 0.05,
  parameters.initial = c(P = NA, S = -2, K = 0, K1 = 1, K2 = 0),
  males.freq = TRUE, fixed.parameters = NULL, equation = "logistic",
  replicate.CI = 10000, range.CI = 0.95,
  replicate.NullDeviance = 1000, control = list(maxit = 1000),
  print = TRUE)
```

Arguments

`df`	A dataframe with at least two columns named males, females or N and temperatures, Incubation.temperature or durations column
`males`	A vector with male numbers
`females`	A vector with female numbers
`N`	A vector with total numbers
`temperatures`	The constant incubation temperatures used to fit sex ratio
`durations`	The duration of incubation or TSP used to fit sex ratio
`l`	Sex ratio limits to define TRT are l and 1-l (see Girondot, 1999)
`parameters.initial`	Initial values for P, S or K search as a vector, ex. c(P=29, S=-0.3)
`males.freq`	If TRUE data are shown as males frequency
`fixed.parameters`	Parameters that will not be changed
`equation`	Could be "logistic", "Hill", "A-logistic", "Hulin", "Double-A-logistic", "flexit" or "GSD"
`replicate.CI`	Number of replicates to estimate confidence intervals
`range.CI`	The range of confidence interval for estimation, default=0.95
`replicate.NullDeviance`	Number of replicates to estimate null distribution of deviance
`control`	List of parameters used in optim.
`print`	Should the results be printed at screen? TRUE (default) or FALSE

Details

tsd estimates the parameters that best describe temperature-dependent sex determination

Value

A list the pivotal temperature, transitional range of temperatures and their SE

Author(s)

Marc Girondot

References

Girondot, M. 1999. Statistical description of temperature-dependent sex determination using maximum likelihood. Evolutionary Ecology Research, 1, 479-486.

Godfrey, M.H., Delmas, V., Girondot, M., 2003. Assessment of patterns of temperature-dependent sex determination using maximum likelihood model selection. Ecoscience 10, 265-272.

Hulin, V., Delmas, V., Girondot, M., Godfrey, M.H., Guillon, J.-M., 2009. Temperature-dependent sex determination and global change: are some species at greater risk? Oecologia 160, 493-506.

See Also

Other Functions for temperature-dependent sex determination: `DatabaseTSD`, `P_TRT`, `TSP.list`, `plot.tsd`, `predict.tsd`, `stages`, `tsd_MHmcmc_p`, `tsd_MHmcmc`

Examples

```
## Not run:
library(embryogrowth)
CC_AtlanticSW <- subset(DatabaseTSD, RMU=="Atlantic, SW" &
                          Species=="Caretta caretta" & (!is.na(Sexed) & Sexed!=0) &
                          !is.na(Correction.factor))
tsdL <- with (CC_AtlanticSW, tsd(males=Males, females=Females,
                                 temperatures=Incubation.temperature-Correction.factor,
                                 equation="logistic", replicate.CI=NULL))
tsdH <- with (CC_AtlanticSW, tsd(males=Males, females=Females,
                                 temperatures=Incubation.temperature-Correction.factor,
                                 equation="Hill", replicate.CI=NULL))
tsdR <- with (CC_AtlanticSW, tsd(males=Males, females=Females,
                                 temperatures=Incubation.temperature-Correction.factor,
                                 equation="A-logistic", replicate.CI=NULL))
tsdF <- with (CC_AtlanticSW, tsd(males=Males, females=Females,
                                 temperatures=Incubation.temperature-Correction.factor,
                                 equation="Flexit", replicate.CI=NULL))
tsdDR <- with (CC_AtlanticSW, tsd(males=Males, females=Females,
                                 temperatures=Incubation.temperature-Correction.factor,
                                 equation="Double-A-logistic", replicate.CI=NULL))
gsd <- with (CC_AtlanticSW, tsd(males=Males, females=Females,
                                 temperatures=Incubation.temperature-Correction.factor,
                                 equation="GSD", replicate.CI=NULL))
compare_AIC(Logistic_Model=tsdL, Hill_model=tsdH, Alogistic_model=tsdR,
               flexit=tsdF,
               DoubleAlogistic_model=tsdDR, GSD_model=gsd)
compare_AICc(Logistic_Model=tsdL, Hill_model=tsdH, Alogistic_model=tsdR,
```

```
               DoubleAlogistic_model=tsdDR, GSD_model=gsd, factor.value = -1)
compare_BIC(Logistic_Model=tsdL, Hill_model=tsdH, Alogistic_model=tsdR,
               DoubleAlogistic_model=tsdDR, GSD_model=gsd, factor.value = -1)
##############
eo <- subset(DatabaseTSD, Species=="Emys orbicularis", c("Males", "Females",
                                       "Incubation.temperature"))

eo_Hill <- with(eo, tsd(males=Males, females=Females,
                                       temperatures=Incubation.temperature,
                                       equation="Hill"))
eo_Hill <- tsd(df=eo, equation="Hill", replicate.CI=NULL)
eo_logistic <- tsd(eo, replicate.CI=NULL)
eo_Alogistic <- with(eo, tsd(males=Males, females=Females,
                                 temperatures=Incubation.temperature,
                                 equation="a-logistic", replicate.CI=NULL))
### The Hulin model is a modification of A-logistic (See Hulin et al. 2009)

########## Caution
### It should not be used anymore as it can produce unexpected results
par <- eo_Alogistic$par
names(par)[which(names(par)=="K")] <- "K2"
par <- c(par, K1=0)
eo_Hulin <- with(eo, tsd(males=Males, females=Females,
                                 parameters.initial=par,
                                 temperatures=Incubation.temperature,
                                 equation="Hulin", replicate.CI=NULL))

### The Double-A-logistic model is a A-logistic model with K1 and K2 using respectively
### below and above P

########## Caution
### The curve is not smooth at pivotal temperature

par <- eo_Alogistic$par
names(par)[which(names(par)=="K")] <- "K2"
par <- c(par, K1=as.numeric(par["K2"])*0.8)
par["K2"] <- par["K2"]*0.8
eo_Double_Alogistic <- with(eo, tsd(males=Males, females=Females,
                                 parameters.initial=par,
                                 temperatures=Incubation.temperature,
                                 equation="Double-a-logistic", replicate.CI=NULL))

### The flexit model is modeled with K1 and K2 using respectively
### below and above P and smooth transition at P; S is the slope at P

par <- c(eo_logistic$par["P"], 1/4*eo_logistic$par["S"], K1=1, K2=1)
eo_flexit <- with(eo, tsd(males=Males, females=Females,
                                 parameters.initial=par,
                                 temperatures=Incubation.temperature,
                                 equation="flexit", replicate.CI=NULL))

compare_AIC(Logistic=eo_logistic, Hill=eo_Hill, Alogistic=eo_Alogistic,
             Hulin=eo_Hulin, Double_Alogistic=eo_Double_Alogistic,
```

```
             flexit=eo_flexit)
## Note that SE for lower limit of TRT is wrong
plot(eo_flexit)
## To get correct confidence interval, check \code{tsd_MHmcmc()}.

### Note the asymmetry of the Double-A-logistic and flexit models
predict(eo_Double_Alogistic,
     temperatures=c(eo_Double_Alogistic$par["P"]-0.2, eo_Double_Alogistic$par["P"]+0.2))
predict(eo_Double_Alogistic)

(p <- predict(eo_flexit,
       temperatures=c(eo_flexit$par["P"]-0.3, eo_flexit$par["P"]+0.3)))
p["50%", 1]-0.5; 0.5-p["50%", 2]
predict(eo_flexit)

### It can be used also for incubation duration
CC_AtlanticSW <- subset(DatabaseTSD, RMU=="Atlantic, SW" &
                          Species=="Caretta caretta" & Sexed!=0)
tsdL_IP <- with (CC_AtlanticSW, tsd(males=Males, females=Females,
                                 durations=IP.mean,
                                 equation="logistic", replicate.CI=NULL))
plot(tsdL_IP, xlab="Incubation durations in days")
# Example with Chelonia mydas
cm <- subset(DatabaseTSD, Species=="Chelonia mydas" & !is.na(Sexed), c("Males", "Females",
                                         "Incubation.temperature", "RMU"))
tsd(subset(cm, subset=RMU=="Pacific, SW"))
tsd(subset(cm, subset=RMU=="Pacific, Northwest"))
tsd(subset(cm, subset=RMU=="Atlantic, S Caribbean"))


## End(Not run)
```

`tsd_MHmcmc` *Metropolis-Hastings algorithm for Sex ratio*

Description

Run the Metropolis-Hastings algorithm for tsd.
Deeply modified from a MCMC script by Olivier Martin (INRA, Paris-Grignon).
The number of iterations is n.iter+n.adapt+1 because the initial likelihood is also displayed.
I recommend that thin=1 because the method to estimate SE uses resampling.
If initial point is maximum likelihood, n.adapt = 0 is a good solution.
To get the SE from result_mcmc <- tsd_MHmcmc(result=try), use:
result_mcmc$BatchSE or result_mcmc$TimeSeriesSE
The batch standard error procedure is usually thought to be not as accurate as the time series methods.
Based on Jones, Haran, Caffo and Neath (2005), the batch size should be equal to sqrt(n.iter).
Jones, G.L., Haran, M., Caffo, B.S. and Neath, R. (2006) Fixed Width Output Analysis for Markov chain Monte Carlo , Journal of the American Statistical Association, 101:1537-1547.

coda package is necessary for this function.
The parameters intermediate and filename are used to save intermediate results every 'intermediate' iterations (for example 1000). Results are saved in a file of name filename.
The parameter previous is used to indicate the list that has been save using the parameters intermediate and filename. It permits to continue a mcmc search.
These options are used to prevent the consequences of computer crash or if the run is very very long and processes at time limited.

Usage

```
tsd_MHmcmc(result = stop("A result of tsd() fit must be provided"),
  n.iter = 10000, parametersMCMC = NULL, n.chains = 1, n.adapt = 0,
  thin = 1, trace = FALSE, traceML = FALSE,
  batchSize = sqrt(n.iter), adaptive = FALSE, adaptive.lag = 500,
  adaptive.fun = function(x) {     ifelse(x > 0.234, 1.3, 0.7) },
  intermediate = NULL, filename = "intermediate.Rdata",
  previous = NULL)
```

Arguments

`result`	An object obtained after a SearchR fit
`n.iter`	Number of iterations for each step
`parametersMCMC`	A set of parameters used as initial point for searching with information on priors
`n.chains`	Number of replicates
`n.adapt`	Number of iterations before to store outputs
`thin`	Number of iterations between each stored output
`trace`	TRUE or FALSE or period, shows progress
`traceML`	TRUE or FALSE to show ML
`batchSize`	Number of observations to include in each batch fo SE estimation
`adaptive`	Should an adaptive process for SDProp be used
`adaptive.lag`	Lag to analyze the SDProp value in an adaptive content
`adaptive.fun`	Function used to change the SDProp
`intermediate`	Period for saving intermediate result, NULL for no save
`filename`	If intermediate is not NULL, save intermediate result in this file
`previous`	Previous result to be continued. Can be the filename in which intermediate results are saved.

Details

tsd_MHmcmc runs the Metropolis-Hastings algorithm for tsd (Bayesian MCMC)

Value

A list with resultMCMC being mcmc.list object, resultLnL being likelihoods and parametersMCMC being the parameters used

Author(s)

Marc Girondot

See Also

Other Functions for temperature-dependent sex determination: `DatabaseTSD`, `P_TRT`, `TSP.list`, `plot.tsd`, `predict.tsd`, `stages`, `tsd_MHmcmc_p`, `tsd`

Examples

```
## Not run:
library(embryogrowth)
eo <- subset(DatabaseTSD, Species=="Emys orbicularis", c("Males", "Females",
                                       "Incubation.temperature"))
eo_logistic <- tsd(eo)
pMCMC <- tsd_MHmcmc_p(eo_logistic, accept=TRUE)
# Take care, it can be very long
result_mcmc_tsd <- tsd_MHmcmc(result=eo_logistic,
parametersMCMC=pMCMC, n.iter=10000, n.chains = 1,
n.adapt = 0, thin=1, trace=FALSE, adaptive=TRUE)
# summary() permits to get rapidly the standard errors for parameters
summary(result_mcmc_tsd)
plot(result_mcmc_tsd, parameters="S", scale.prior=TRUE, xlim=c(-3, 3), las=1)
plot(result_mcmc_tsd, parameters="P", scale.prior=TRUE, xlim=c(25, 35), las=1)

plot(eo_logistic, resultmcmc = result_mcmc_tsd)

1-rejectionRate(as.mcmc(result_mcmc_tsd))
raftery.diag(as.mcmc(result_mcmc_tsd))
heidel.diag(as.mcmc(result_mcmc_tsd))
library(car)
o <- P_TRT(x=eo_logistic, resultmcmc=result_mcmc_tsd)
outEo <- dataEllipse(x=o$P_TRT[, "PT"],
                     y=o$P_TRT[, "TRT"],
                     levels=c(0.95),
                     draw=FALSE)
plot(x = o$P_TRT[, "PT"],
     y=o$P_TRT[, "TRT"],
     pch=".", las=1, bty="n",
     xlab="Pivotal temperature",
     ylab=paste0("TRT ", as.character(100*eo_logistic$l), "%"),
     xlim=c(28.4, 28.6),
     ylim=c(0.8, 1.8))
lines(outEo[, 1], outEo[, 2], col="green", lwd=2)
legend("topleft", legend = c("Emys orbicularis", "95% confidence ellipse"),
       pch=c(19, NA), col=c("black", "green"), lty=c(0, 1), lwd=c(0, 2))

logistic <- function(x, P, S) {
   return(1/(1+exp((1/S)*(P-x))))
}

q <- as.quantile(result_mcmc_tsd, fun=logistic,
```

```
                 xlim=seq(from=25, to=35, by=0.1), nameparxlim="x")
plot(x=seq(from=25, to=35, by=0.1), y=q[1, ], type="l", las=1,
     xlab="Temperatures", ylab="Male proportion", bty="n")
lines(x=seq(from=25, to=35, by=0.1), y=q[2, ])


## End(Not run)
```

`tsd_MHmcmc_p` *Generates set of parameters to be used with tsd_MHmcmc()*

Description

Interactive script used to generate set of parameters to be used with tsd_MHmcmc().

Usage

```
tsd_MHmcmc_p(result = stop("An output from tsd() must be provided"),
  accept = FALSE)
```

Arguments

`result`	An object obtained after a tsd fit
`accept`	If TRUE, the script does not wait user information

Details

tsd_MHmcmc_p generates set of parameters to be used with tsd_MHmcmc()

Value

A matrix with the parameters

Author(s)

Marc Girondot

See Also

Other Functions for temperature-dependent sex determination: `DatabaseTSD`, `P_TRT`, `TSP.list`, `plot.tsd`, `predict.tsd`, `stages`, `tsd_MHmcmc`, `tsd`

Examples

```
## Not run:
library(embryogrowth)
eo <- subset(DatabaseTSD, Species=="Emys orbicularis", c("Males", "Females",
                                        "Incubation.temperature"))
eo_logistic <- with(eo, tsd(males=Males, females=Females,
                                  temperatures=Incubation.temperature))
pMCMC <- tsd_MHmcmc_p(eo_logistic, accept=TRUE)
# Take care, it can be very long
result_mcmc_tsd <- tsd_MHmcmc(result=eo_logistic,
parametersMCMC=pMCMC, n.iter=10000, n.chains = 1,
n.adapt = 0, thin=1, trace=FALSE, adaptive=TRUE)
# summary() permits to get rapidly the standard errors for parameters
summary(result_mcmc_tsd)
plot(result_mcmc_tsd, parameters="S", scale.prior=TRUE, xlim=c(-3, 3), las=1)
plot(result_mcmc_tsd, parameters="P", scale.prior=TRUE, xlim=c(25, 35), las=1)

eo_flexit <- with(eo, tsd(males=Males, females=Females,
                                  parameters.initial=c(eo_logistic$par["P"],
                                  1/(4*eo_logistic$par["S"]),
                                  K1=1, K2=1),
                                  temperatures=Incubation.temperature,
                                  equation="flexit", replicate.CI=NULL))
pMCMC <- tsd_MHmcmc_p(eo_flexit, accept=TRUE)
result_mcmc_tsd <- tsd_MHmcmc(result=eo_flexit,
parametersMCMC=pMCMC, n.iter=10000, n.chains = 1,
n.adapt = 0, thin=1, trace=FALSE, adaptive=TRUE)
# summary() permits to get rapidly the standard errors for parameters
summary(result_mcmc_tsd)
plot(result_mcmc_tsd, parameters="S", scale.prior=TRUE, xlim=c(-3, 3), las=1)
plot(result_mcmc_tsd, parameters="P", scale.prior=TRUE, xlim=c(25, 35), las=1)
plot(result_mcmc_tsd, parameters="K1", scale.prior=TRUE, xlim=c(-10, 10), las=1)
plot(result_mcmc_tsd, parameters="K2", scale.prior=TRUE, xlim=c(-10, 10), las=1)

plot(eo_flexit, resultmcmc = result_mcmc_tsd)

## End(Not run)
```

`TSP.list` *Database of thermosensitive period of development for sex determination*

Description

Database of thermosensitive period of development for sex determination.
This database can be used with the functions plot() or info.nests().
The attributes TSP.begin.stages and TSP.end.stages for each dataframe give respectively the first and the last stages for TSP. Then the metrics for the limits of TSP are the average sizes before and after the TSP (see example, below).

If the metric for the stages before the TSP or after the TSP is not known, it will use the available information.

Usage

```
TSP.list
```

Format

A list with dataframes including attributes

Details

Database of thermosensitive period of development for sex determination

Author(s)

Marc Girondot `<marc.girondot@u-psud.fr>`

References

Mrosovsky, N., Pieau, C., 1991. Transitional range of temperature, pivotal temperatures and thermosensitive stages for sex determination in reptiles. Amphibia-Reptilia 12, 169-179.

See Also

Other Functions for temperature-dependent sex determination: `DatabaseTSD`, `P_TRT`, `plot.tsd`, `predict.tsd`, `stages`, `tsd_MHmcmc_p`, `tsd_MHmcmc`, `tsd`

Examples

```
## Not run:
library(embryogrowth)
data(TSP.list)
names(TSP.list)
reference <- "Emys_orbicularis.mass"
metric <- TSP.list[[reference]]
TSP.begin <- attributes(TSP.list[[reference]])$TSP.begin.stages
TSP.end <- attributes(TSP.list[[reference]])$TSP.end.stages
# Metric at the beginning of the TSP
del <- ifelse(all(metric$stages == TSP.begin - 1)==FALSE, 0, 1)
(metric$metric[metric$stages == TSP.begin - del] +
    metric$metric[metric$stages == TSP.begin]) / 2
# Metric at the end of the TSP
del <- ifelse(all(metric$stages == TSP.begin + 1)==FALSE, 0, 1)
(metric$metric[metric$stages == TSP.end] +
        metric$metric[metric$stages == del + TSP.end]) / 2

## End(Not run)
```

`uncertainty.datalogger`

Uncertainty of average temperatures obtained using temperature data logger

---

Description

Calculate the uncertainty of average temperature dependent on the characteristics of a data logger and sampling rate.
The temperature is supposed to be uniformaly distributed with min and max being -accuracy and +accuracy.

Usage

```
uncertainty.datalogger(max.time = 0, sample.rate = 0, accuracy = 0.5,
  resolution = 1, replicates = 10000, method = function(x) {     2 *
  qnorm(0.975) * sd(x) })
```

Arguments

`max.time`	being the maximum time to record in minutes
`sample.rate`	The sample rates in minutes
`accuracy`	The accuracy of the data logger in °C
`resolution`	The resolution of the data logger in °C
`replicates`	The number of replicates to estimate uncertainty.
`method`	The fonction that will be used to return the uncertainty.

Details

uncertainty.datalogger Calculate the uncertainty of the average temperature calculated using data gathered by a data logger.

Value

The function will return the uncertainty of the average temperature for the considered period as being the 95

Author(s)

Marc Girondot

See Also

Other Data loggers utilities: `calibrate.datalogger`

Examples

```
## Not run:
library(embryogrowth)
# Exemple using the hypothesis of Gaussian distribution
uncertainty.datalogger(sample.rate=30, accuracy=1, resolution=0.5,
                method=function(x) {2*qnorm(0.975)*sd(x)})
# Example without hypothesis about distribution, using quantiles
uncertainty.datalogger(sample.rate=30, accuracy=1, resolution=0.5,
                method=function(x) {quantile(x, probs=c(0.975))-
                                              quantile(x, probs=c(0.025))})
par(mar=c(4, 4, 1, 1))
plot(x=10:120, uncertainty.datalogger(sample.rate=10:120,
                                      accuracy=0.5,
                                      resolution=1),
     las=1, bty="n", type="l",
     xlab="Sample rate in minutes",
     ylab=expression("Uncertainty in "*degree*"C"),
     ylim=c(0, 0.15), xlim=c(0, 120))
lines(x=10:120, uncertainty.datalogger(sample.rate=10:120,
                                            accuracy=1,
                                            resolution=0.5), col="red")
lines(x=10:120, uncertainty.datalogger(sample.rate=10:120,
                                       accuracy=1,
                                       resolution=1), col="blue")
lines(x=10:120, uncertainty.datalogger(sample.rate=10:120,
                                       accuracy=0.5,
                                       resolution=0.5), col="yellow")
legend("topleft", legend=c("Accuracy=0.5, resolution=0.5",
                           "Accuracy=0.5, resolution=1",
                           "Accuracy=1, resolution=0.5",
                           "Accuracy=1, resolution=1"), lty=1,
       col=c("yellow", "black", "red", "blue"),
       cex=0.6)

## End(Not run)
```

`web.tsd` *Run a shiny application for basic functions of tsd function*

Description

Run a shiny application for basic functions of tsd function.

Usage

```
web.tsd()
```

Details

web.tsd runs a shiny application for basic functions of tsd function

Value

Nothing

Author(s)

Marc Girondot

Examples

```
## Not run:
library(embryogrowth)
web.tsd()

## End(Not run)
```

`weightmaxentropy` *Search for the weights of the nests which maximize the entropy of nest temperatures distribution*

Description

Search for the weights of the nests which maximize the entropy of nest temperatures distribution.
Entropy is measured by Shanon index.
Entropy method must be entropy.empirical because it is the only method insensitive to scaling.
If no weight is given, the initial weight is uniformly distributed.
Use control_optim=list(trace=0) for not show progress of search report.

Usage

```
weightmaxentropy(temperatures = stop("Temperature data must be provided !"),
weight = NULL, entropy.method = entropy::entropy.empirical,
plot = TRUE, control_optim = list(trace = 0, maxit = 500),
control_plot = NULL, control_entropy = NULL, col = c("black",
"red"))
```

Arguments

`temperatures`	Timeseries of temperatures formated using FormatNests()
`weight`	A named vector of the initial weight search for each nest for likelihood estimation
`entropy.method`	Entropy function, for example entropy::entropy.empirical. See package entropy for description
`plot`	Do the plot of temperatures before and after weight must be shown ? TRUE or FALSE
`control_optim`	A list with control paramaters for optim function

`control_plot`	A list with control paramaters for plot function
`control_entropy`	A list with control paramaters for entropy function
`col`	Colors for unweighted and weighted distributions

Details

Search for the weights of the nests which maximize the entropy of nest temperatures distribution

Value

A named vector of weights

Author(s)

Marc Girondot

Examples

```
## Not run:
library(embryogrowth)
data(nest)
formated <- FormatNests(nest)
w <- weightmaxentropy(formated, control_plot=list(xlim=c(20,36)))
x <- structure(c(120.940334922916, 467.467455887442,
306.176613681557, 117.857995419495),
.Names = c("DHA", "DHH", "T12H", "Rho25"))
# pfixed <- c(K=82.33) or rK=82.33/39.33
pfixed <- c(rK=2.093313)
# K or rK are not used for dydt.linear or dydt.exponential
resultNest_4p_weight <- searchR(parameters=x,
fixed.parameters=pfixed, temperatures=formated,
derivate=dydt.Gompertz, M0=1.7, test=c(Mean=39.33, SD=1.92),
method = "BFGS", weight=w)
data(resultNest_4p_weight)
plotR(resultNest_4p_weight, ylim=c(0,0.50), xlim=c(15, 35))
# Standard error of parameters can use the GRTRN_MHmcmc() function

## End(Not run)
```

Index