Package ‘embryogrowth’

June 16, 2025

Type Package

Title Tools to Analyze the Thermal Reaction Norm of Embryo Growth

Version 10.2

Date 2025-06-16

Description Tools to analyze the embryo growth and the sexualisation thermal reaction norms. See <doi:10.7717/peerj.8451> for tsd functions; see <doi:10.1016/j.jtherbio.2014.08.005> for thermal reaction norm of embryo growth.

Depends deSolve, optimx, numDeriv, ggplot2, HelpersMG (>= 6.5),R (>= 4.1)

Suggests entropy, shiny, coda, polynom, car, gam, pbapply, cranlogs, parallel, mapdata

License GPL-2

LazyData yes

LazyLoad yes

Encoding UTF-8

RoxygenNote 7.3.2

NeedsCompilation no

Imports Rdpack

RdMacros Rdpack

Author Marc Girondot [aut, cre] (ORCID: <`https://orcid.org/0000-0001-6645-8530`>)

Maintainer Marc Girondot <`marc.girondot@gmail.com`>

Repository CRAN

Date/Publication 2025-06-16 13:30:02 UTC

Contents

`embryogrowth-package` *The package embryogrowth*

Description

Tools to analyze the embryo growth and the sexualisation thermal reaction norms.
The latest version of this package can always been installed using:
install.packages("http://marc.girondot.free.fr/CRAN/HelpersMG.tar.gz", repos=NULL, type="source")
install.packages("http://marc.girondot.free.fr/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:	10.2 build 2094
Date:	2025-06-16
License:	GPL (>= 2)
LazyLoad:	yes

Author(s)

Marc Girondot `<marc.girondot@gmail.com>`

References

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

Godfrey MH, Delmas V, Girondot M (2003). “Assessment of patterns of temperature-dependent sex determination using maximum likelihood model selection.” *Ecoscience*, **10**(3), 265-272.

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.

Girondot M, Kaska Y (2015). “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. doi:10.1016/j.jtherbio.2014.10.008.

Fuentes MM, Monsinjon J, Lopez M, Lara P, Santos A, dei Marcovaldi MA, Girondot M (2017). “Sex ratio estimates for species with temperature-dependent sex determination differ according to the proxy used.” *Ecological Modelling*, **365**, 55-67. doi:10.1016/j.ecolmodel.2017.09.022.

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**(2), 172-179. doi:10.2744/CCB1269.1.

Girondot M, Godfrey MH, Guillon J, Sifuentes-Romero I (2018). “Understanding and integrating resolution, accuracy and sampling rates of temperature data loggers used in biological and ecological studies.” *Engineering Technology Open Access Journal*, **2**(4), 55591.

Girondot M, Monsinjon J, Guillon J (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. doi:10.1016/j.jtherbio.2018.02.006.

Abreu-Grobois FA, Morales-Mérida BA, Hart CE, Guillon J, Godfrey MH, Navarro E, Girondot M (2020). “Recent advances on the estimation of the thermal reaction norm for sex ratios.” *PeerJ*, **8**, e8451. doi:10.7717/peerj.8451, `https://peerj.com/articles/8451/`.

Morales Mérida A, Helier A, Cortés-Gómes AA, Girondot M (2021). “Hatching success rather than temperature-dependent sex determination as the main driver of Olive Ridley (Lepidochelys olivacea) nest density in the Pacific Coast of Central America.” *Animals (Basel)*, **11**, 3168. doi:10.3390/ani11113168.

Monsinjon J, Guillon J, Wyneken J, Girondot M (2022). “Thermal reaction norm for sexualization: the missing link between temperature and sex ratio for temperature-dependent sex determination.” *Ecological Modelling*, **473**(110119), 1-7. doi:10.1016/j.ecolmodel.2022.110119.

Morales-Mérida BA, Morales-Cabrera A, Chúa C, Girondot M (2023). “Olive ridley sea turtle incubation in natural conditions is possible on Guatemalan beaches.” *Sustainability*, **15**, 14196. doi:10.3390/su151914196.

Hulin V, Delmas V, Girondot M, Godfrey MH, Guillon J (2009). “Temperature-dependent sex determination and global change: Are some species at greater risk?” *Oecologia*, **160**(3), 493-506.

Tello-Sahagún LA, Ley-Quiñonez CP, Abreu-Grobois FA, Monsinjon JR, Zavala-Norzagaray AA, Girondot M, Hart CE (2023). “Neglecting low season nest protection exacerbates female biased sea turtle hatchling production through the loss of male producing nests.” *Biological Conservation*, **277**, 109873. doi:10.1016/j.biocon.2022.109873.

Morales-Mérida BA, Contreras-Mérida MR, Girondot M (2019). “Pipping dynamics in marine turtle Lepidochelys olivacea nests.” *Trends in Developmental Biology*, **12**, 23-30.

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 rK should be 1.209 and M0 should be 0.34.
# 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
#
# See the example in the stages datasets
#
################################################################################

resultNest_4p_SSM <- searchR(parameters=x, fixed.parameters=pfixed,
temperatures=formated, integral=integral.Gompertz, M0=1.7,
hatchling.metric=c(Mean=39.33, SD=1.92))
data(resultNest_4p_SSM)
par(mar=c(4, 4, 1, 1))
plot(resultNest_4p_SSM$data, bty="n", las=1,
     xlab="Days of incubation", ylab="Temperatures in °C",
     series="all",
     type="l", xlim=c(0,70),ylim=c(20, 35))
par(mar=c(4, 4, 1, 1))
pMCMC <- TRN_MHmcmc_p(resultNest_4p_SSM, accept=TRUE)
# Take care, it can be very long, sometimes several days
resultNest_mcmc_4p_SSM <- GRTRN_MHmcmc(result=resultNest_4p_SSM,
parametersMCMC=pMCMC, n.iter=10000, n.chains = 1, n.adapt = 0,
thin=1, trace=TRUE)
data(resultNest_mcmc_4p_SSM)
out <- as.mcmc(resultNest_mcmc_4p_SSM)
# 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_SSM, parameters=1, xlim=c(0,550))
plot(resultNest_mcmc_4p_SSM, 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_SSM, result = resultNest_4p_SSM,
                        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_SSM, 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_SSM,
             resultmcmc=resultNest_mcmc_4p_SSM,
             ylim=c(0, 8),
             curve = "MCMC quantiles", show.density=TRUE)

plotR(resultNest_6p_SSM, resultmcmc=resultNest_mcmc_6p_SSM,
            ylim=c(0, 8), show.density=TRUE, show.hist=TRUE,
            curve = "MCMC quantiles",
            ylimH=c(0,0.5), atH=c(0, 0.1, 0.2))

# How many times this package has been download
library(cranlogs)
embryogrowth <- cran_downloads("embryogrowth", from = "2014-08-16",
                               to = Sys.Date() - 1)
sum(embryogrowth$count)
plot(embryogrowth$date, embryogrowth$count, type="l", bty="n")

## 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

References

Girondot M, Godfrey MH, Guillon J, Sifuentes-Romero I (2018). "Understanding and integrating resolution, accuracy and sampling rates of temperature data loggers used in biological and ecological studies." *Engineering Technology Open Access Journal*, **2**(4), 55591.

See Also

Other Data loggers utilities: `movement()`, `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_SSM)
x1 <- resultNest_6p_SSM$par
data(resultNest_4p_SSM)
x2 <- resultNest_4p_SSM$par
temperaturesC <- (200:350)/10
s <- ChangeSSM(temperatures=temperaturesC, parameters=x1, initial.parameters=x2)
sY <- plotR(resultNest_6p_SSM, ylim=c(0,3), col="black", curve = "ML")
plotR(resultNest_4p_SSM, col="red", scaleY=sY, new=FALSE)
plotR(s$par, col="green", scaleY=sY, new=FALSE, curve = "ML")
legend("topleft", legend=c("r function to mimic", "Initial new r function",
"Fitted new r function"), lty=c(1, 1, 1), col=c("black", "red", "green"))
# Other example to fit anchored parameters
data(resultNest_4p_SSM)
x0 <- resultNest_4p_SSM$par
t <- hist(resultNest_4p_SSM, 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_SSM)
 x <- resultNest_4p_SSM$par
 xanchor["294"] <- 0
 xanchor["308"] <- 2.3291035
 x <- ChangeSSM(parameters = xanchor,
                    initial.parameters = x, control=list(maxit=5000))
 sY <- plotR(resultNest_4p_SSM$par, ylim = c(0,3), curve="ML")
 plotR(xprime$par, col="red", scaleY=sY, new=FALSE, curve="ML")
 legend("topleft", legend=c("Fitted parameters", "Constrainted parameters"), lty=1,
        col=c("black", "red"))
 # Weibull model
 x <- ChangeSSM(temperatures = (200:350)/10,
                parameters = resultNest_4p_SSM$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_SSM$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_SSM$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_SSM)
new_result <- ChangeSSM(resultmcmc = resultNest_mcmc_4p_SSM, result = resultNest_4p_SSM,
                        temperatures = seq(from = 20, to = 35, by = 0.1),
                        outmcmc = "mean-sd",
                        initial.parameters = NULL)

plotR(new_result, ylim=c(0, 3), curve="ML")
 # example with a mcmc object, CI being defined by 2.5%-97.5% quantiles
 # Note the asymmetric CI
data(resultNest_mcmc_4p_SSM)
new_result <- ChangeSSM(resultmcmc = resultNest_mcmc_4p_SSM, result = resultNest_4p_SSM,
                        temperatures = seq(from = 20, to = 35, by = 0.1),
                        outmcmc = "quantiles",
                        initial.parameters = NULL)

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

# A little trick
# to convert SSM4 to SSM6, you can use:

x4 <- c('DHA' = 69.718935117894063,
         'DHH' = 497.81709040501079,
         'T12H' = 308.95543713889509,
         'Rho25' = 255.24186073771696)

x6 <- c(x4["DHA"],
.        x4["DHH"],
.        'DHL' = x4[["DHH"]],
.        'DT' = 0.5,
.        'T12L' = x4[["T12H"]],
.        x4['Rho25'])

## End(Not run)
```

`DatabaseNestingArea` *Database of RMU for marine turtles*

Description

Database of RMU for marine turtles - Version 2010

Usage

```
DatabaseNestingArea
```

Format

A dataframe with raw data.

Details

Database of RMU for marine turtles

Author(s)

Marc Girondot `<marc.girondot@universite-paris-saclay.fr>`

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

References

Wallace BP, DiMatteo AD, Hurley BJ, Finkbeiner EM, Bolten AB, Chaloupka MY, Hutchinson BJ, Abreu-Grobois FA, Amorocho D, Bjorndal KA, Bourjea J, Bowen BW, Dueñas RB, Casale P, Choudhury BC, Costa A, Dutton PH, Fallabrino A, Girard A, Girondot M, Godfrey MH, Hamann M, López-Mendilaharsu M, Marcovaldi MA, Mortimer JA, Musick JA, Nel R, Seminoff JA, Troëng S, Witherington B, Mast RB (2010). “Regional management units for marine turtles: a novel framework for prioritizing conservation and research across multiple scales.” *PLoS One*, **5**(12), e15465. doi:10.1371/journal.pone.0015465.

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 of the species in binominal nommenclature
- `Subspecies`: Name of the subspecies
- `Country`: From which country the eggs come from

- `Area`: Name of the beach or region the eggs come from
- `RMU.2010`: 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.
- `RMU.2023`: For marine turtles, name of the RMU for this population; see Wallace BP, Posnik ZA, Hurley BJ, DiMatteo AD, Bandimere A, Rodriguez I, Maxwell SM, Meyer L, Brenner H, Jensen MP, LaCasella E, Shamblin BM, Abreu Abreu-Grobois FA, Stewart KR, Dutton PH, Barrios-Garrido H, Dalleau M, Dell'amico F, Eckert KL, FitzSimmons NN, Garcia-Cruz M, Hays GC, Kelez S, Lagueux CJ, Madden Hof CA, Marco A, Martins SLT, Mobaraki A, Mortimer JA, Nel R, Phillott AD, Pilcher NJ, Putman NF, Rees AF, Rguez-Baron JM, Seminoff JA, Swaminathan A, Turkozan O, Vargas SM, Vernet PD, Vilaça S, Whiting SD, Hutchinson BJ, Casale P, Mast RB (2023) Marine turtle regional management units 2.0: an updated framework for conservation and research of wide-ranging megafauna species. Endangered Species Research 52:209-223.
- `Incubation.temperature.set`: Nominal incubation temperature
- `Incubation.temperature.recorded`: Nominal or real (if available) incubation temperature
- `Duplicated.data`: TRUE if these data are duplicated in database
- `Duplicate`: Unique code for the duplicate
- `Incubation.temperature.Constant`: Does the incubation temperature was set as constant or CTE was reported
- `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
- `Box`: Identity of the condition incubation
- `Clutch`: Identity or number of clutches
- `Reference`: Bibliographic reference
- `Note`: Diverse information for this incubation
- `Digital_Identifier`: A unique digital identifier
- `Version`: Date of the last modification for each record

The Incubation.temperature records are the incubation temperature of the incubator. If a correction factor was substracted in the publication to represent the temperature of the egg itself, it has been added here.

Usage

```
DatabaseTSD
```

Format

A dataframe with raw data.

Details

Database of TSD information for marine turtles

Author(s)

Marc Girondot `<marc.girondot@universite-paris-saclay.fr>`

See Also

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

Examples

```
## Not run:
library(embryogrowth)
data(DatabaseTSD)
DatabaseTSD.version()
totalIncubation_Lo <- subset(DatabaseTSD,
         Species=="Lepidochelys olivacea" & (!is.na(Sexed) & Sexed!=0),
         select=c("Males", "Females", "Incubation.temperature.set"))
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)
```

`DatabaseTSD.version` *Version of database of TSD information for reptiles*

Description

Return the date of the most recent update of the database.

Usage

```
DatabaseTSD.version()
```

Details

Database of information for incubation of turtles

Value

The date of the lastest updated version

Author(s)

Marc Girondot `<marc.girondot@universite-paris-saclay.fr>`

See Also

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

Examples

```
## Not run:
library(embryogrowth)
DatabaseTSD.version()

## 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,
hatchling.metric=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_SSM <- searchR(parameters=x, fixed.parameters=pfixed,
temperatures=formated, derivate=dydt.Gompertz, M0=1.7,
hatchling.metric=c(Mean=39.33, SD=1.92))
data(resultNest_4p_SSM)

## 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,
hatchling.metric=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).
The parameter LayingTime is also a vector of POSIXct time or POSIXlt time.
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.
The UnitTime should be "seconds", "minutes", "hours", or "days" to be understood by plot function.

Usage

```
FormatNests(
  data = stop("A dataset must be provided !"),
  Time.Format = NULL,
  Time.Zone = NULL,
  previous = NULL,
  LayingTime = NULL,
  UnitTime = "minutes",
  Longitude = NULL,
  Latitude = NULL,
  Informations = NULL,
  Males = NULL,
  Females = NULL,
  usemiddletime = FALSE,
  simplify = TRUE,
  weight = NULL,
  hatchling.metric.mean = NULL,
  hatchling.metric.sd = NULL,
  col.Time = "Time"
)
```

Arguments

`data`	Data to be newly formated.
`Time.Format`	Format of time. See description. If NULL, no time conversion is done.
`Time.Zone`	The format of time zone as obtained by OlsonNames(). See description.
`previous`	Data already formated.
`LayingTime`	Named POSIXct or POSIXlt time for each nest in data.
`UnitTime`	The units for time as a named list or vector
`Longitude`	The longitude of the nests as a named list or vector
`Latitude`	The latitude of the nests as a named list or vector
`Informations`	Some textual information about the nests as a named list or vector
`Males`	Number of sexed eggs being males
`Females`	Number of sexed eggs being females
`usemiddletime`	If TRUE, suppose that recorded temperatures are those at middle segment.
`simplify`	If TRUE, simply the time series by removing identical time series of temperatures.
`weight`	Named vector with weight used to estimate likelihood.
`hatchling.metric.mean`	The average size of hatchlings
`hatchling.metric.sd`	The standard deviation of size of hatchlings
`col.Time`	Name of the column with time.

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 <marc.girondot@gmail.com>

Examples

```
## Not run:
library(embryogrowth)
data(nest)
formated <- FormatNests(data=nest, previous=NULL, col.Time="Time")
# If I try to add the same nest, I have an error
formated <- FormatNests(data=nest, previous=formated, col.Time="Time")
# I duplicate the database and change the names
nest_duplicate <- nest
colnames(nest_duplicate) <- paste0(colnames(nest_duplicate), "_essai")
```

```
formated <- FormatNests(data=nest_duplicate, previous=formated, col.Time="Time_essai")
# It is possible to add information about these nests
formated <- FormatNests(data=nest, previous=NULL, col.Time="Time")
formated <- UpdateNests(data=formated, Males=c(DY.1=10), Females=c(DY.1=2))
####################
Laying.Time <- matrix(c("DY.1", "15/05/2010",
                 "DY.17", "24/05/2010",
                 "DY.16", "24/05/2010",
                 "DY.18", "25/05/2010",
                 "DY.20", "25/05/2010",
                 "DY.21", "26/05/2010",
                 "DY.22", "26/05/2010",
                 "DY.23", "26/05/2010",
                 "DY.24", "27/05/2010",
                 "DY.25", "27/05/2010",
                 "DY.28", "28/05/2010",
                 "DY.26", "28/05/2010",
                 "DY.27", "28/05/2010",
                 "DY.146", "20/06/2010",
                 "DY.147", "20/06/2010",
                 "DY.172", "24/06/2010",
                 "DY.175", "24/06/2010",
                 "DY.170", "24/06/2010",
                 "DY.260", "06/07/2010",
                 "DY.282", "12/07/2010",
                 "DY.310", "18/07/2010",
                 "DY.309", "18/07/2010",
                 "DY.328", "25/07/2010",
                 "DY.331", "26/07/2010"), byrow=TRUE, ncol=2)
tz <- OlsonNames()[grepl("Asia/Istanbul", OlsonNames())]
Laying.Time_f <- setNames(as.POSIXlt.character(Laying.Time[, 2], format = "%d/%m/%Y", tz=tz),
                          Laying.Time[, 1])
formated <- FormatNests(data=nest, previous=NULL, col.Time="Time", LayingTime=Laying.Time_f)
####################
# Now when the data are with absolute dates that are already formatted
nest_ec <- data.frame(Time=as.POSIXlt("24/05/2010", format="%d/%m/%Y")+ nest[, 1]*60,
                      DY.1.x=nest[, 2])
formated <- FormatNests(data=nest_ec, previous=NULL, col.Time="Time")
####################
# Now when the data are with absolute date that are in text format for example after
#   reading a csv format
nest_ec <- data.frame(Time=format(as.POSIXlt("24/05/2010", format="%d/%m/%Y")+ nest[, 1]*60,
                                  format = "%d/%m/%Y %H:%M:%S"),
                      DY.1.x=nest[, 2])
formated <- FormatNests(data=nest_ec, previous=NULL, col.Time="Time",
                        Time.Format="%d/%m/%Y %H:%M:%S",
                        Time.Zone=OlsonNames()[grepl("Asia/Istanbul", OlsonNames())],
                        hatchling.metric.mean=39.33, hatchling.metric.sd=1.92)


## 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_SSM)
newp <- GenerateAnchor(nests=resultNest_4p_SSM, number.anchors=7)
newp <- GenerateAnchor(nests=resultNest_4p_SSM, temperatures=seq(from=20,
 to=35, length.out=10))
newp <- GenerateAnchor(nests=resultNest_4p_SSM, 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)
```

`Generate_hatchling_metric`

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

Description

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

Usage

```
Generate_hatchling_metric(
  series = stop("A result object or names of series must be provided"),
  hatchling.metric = NULL,
  previous = NULL
)
```

Arguments

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

Details

Generate_hatchling_metric Generate a data.frame that can be used as hatchling.metric value for searchR()

Value

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

Author(s)

Marc Girondot <`marc.girondot@gmail.com`>

Examples

```
## Not run:
library(embryogrowth)
data(resultNest_4p_SSM)
testsize1 <- Generate_hatchling_metric(series = resultNest_4p_SSM)
testsize2 <- Generate_hatchling_metric(series=resultNest_4p_SSM,
                                       hatchling.metric=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,
  WAIC = TRUE,
  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
`WAIC`	Should WAIC data been recorded?
`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"
#############################################################################
pfixed <- c(rK=1.208968)
M0 = 0.3470893
#############################################################################
# 4 parameters
#############################################################################
x c('DHA' = 109.31113503282113, 'DHH' = 617.80695919563857,
    'T12H' = 306.38890489505093, 'Rho25' = 229.37265815800225)

resultNest_4p_SSM <- searchR(parameters=x, fixed.parameters=pfixed,
temperatures=formated, integral=integral.Gompertz, M0=M0,
hatchling.metric=c(Mean=39.33, SD=1.92))
data(resultNest_4p_SSM)
plot(resultNest_4p_SSM, xlim=c(0,70), ylimT=c(22, 32), ylimS=c(0,45), series=1,
embryo.stages="Caretta caretta.SCL")
#############################################################################
pMCMC <- TRN_MHmcmc_p(resultNest_4p_SSM, accept=TRUE)
# Take care, it can be very long; several days
resultNest_mcmc_4p_SSM <- GRTRN_MHmcmc(result=resultNest_4p_SSM ,
                                       adaptive = TRUE          ,
                                       WAIC=TRUE                ,
                                    parametersMCMC=pMCMC     ,
                                    n.iter=10000            ,
                                    n.chains = 1             ,
                                    n.adapt = 0              ,
                                    thin=1                   ,
                                    trace=TRUE               )
# The SDProp in pMCMC at the beginning of the Markov chain can
# be considered as non-optimal. However, the values at the end
# of the Markoc chain are better due to the use of the option
# adaptive = TRUE. Then a good strategy is to run again the MCMC
# with this final set:
pMCMC[, "SDProp"] <- resultNest_mcmc_4p_SSM$parametersMCMC$SDProp.end
# Also, take the set of parameters fitted as maximim likelihood
# as initial set of value
pMCMC[, "Init"] <- as.parameters(resultNest_mcmc_4p_SSM)
# Then I run again the MCMC; it will ensure to get the optimal distribution
resultNest_mcmc_4p_SSM <- GRTRN_MHmcmc(result=resultNest_4p_SSM,
                                       adaptive = TRUE         ,
                                    parametersMCMC=pMCMC    ,
                                    n.iter=10               ,
                                    n.chains = 1            ,
                                    n.adapt = 0             ,
```

```
                                    thin=1                  ,
                                    trace=TRUE              )

data(resultNest_mcmc_4p_SSM)
out <- as.mcmc(resultNest_mcmc_4p_SSM)
# 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_SSM, parameters=1, xlim=c(0,550))
plot(resultNest_mcmc_4p_SSM, 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_SSM$SD
# the confidence interval is better estimated by:
apply(out[[1]], 2, quantile, probs=c(0.025, 0.975))
# The use of the intermediate method is as followed;
# Here the total mcmc iteration is 10000, but every 1000, intermediate
# results are saved in file intermediate1000.Rdata:
resultNest_mcmc_4p_SSM <- GRTRN_MHmcmc(result=resultNest_4p_SSM,
parametersMCMC=pMCMC, n.iter=10000, n.chains = 1,
n.adapt = 0, thin=1, trace=TRUE,
intermediate=1000, filename="intermediate1000.Rdata")
# If run has been stopped for any reason, it can be resumed with:
resultNest_mcmc_4p_SSM <- GRTRN_MHmcmc(previous="intermediate1000.Rdata")
# Example to use of the epsilon parameter to get confidence level
resultNest_4p_epsilon <- resultNest_4p
resultNest_4p_epsilon$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.
The first version of the model was published in:

Laloë, J.-O., Monsinjon, J., Gaspar, C., Touron, M., Genet, Q., Stubbs, J., Girondot, M. & Hays, G.C. (2020) Production of male hatchlings at a remote South Pacific green sea turtle rookery: conservation implications in a female-dominated world. Marine Biology, 167, 70.
The version available here is enhanced by using a double flexit model rather than a double logistic model. The flexit model is described here:
Abreu-Grobois, F.A., Morales-Mérida, B.A., Hart, C.E., Guillon, J.-M., Godfrey, M.H., Navarro, E. & Girondot, M. (2020) Recent advances on the estimation of the thermal reaction norm for sex ratios. PeerJ, 8, e8451.

Usage

```
HatchingSuccess.fit(
  par = NULL,
  data = stop("data must be provided"),
  fixed.parameters = NULL,
  column.Incubation.temperature = "Incubation.temperature",
  column.Hatched = "Hatched",
  column.NotHatched = "NotHatched",
  hessian = TRUE
)
```

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
`hessian`	Should Hessian matrix be estimated?

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()`, `HatchingSuccess.MHmcmc_p()`, `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=0.8)

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

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

plot(g.logistic)

par <- c(S.low=0.5, S.high=0.3,
         P.low=25, deltaP=10,
         K1.low=1, K2.low=-1, K1.high=1, K2.high=-1,
         MaxHS=0.8)

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

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

compare_AICc(logistic=g.logistic, flexit=g.flexit)
plot(x=g.logistic, what = c("observations", "ML", "CI"), replicates=10000)

pMCMC <- HatchingSuccess.MHmcmc_p(result = g.logistic, accept = TRUE)
MCMC <- HatchingSuccess.MHmcmc(result = g.logistic, parametersMCMC = pMCMC,
                            n.iter = 100000,
                           adaptive = TRUE)

plot(MCMC, parameters = "S.low")
plot(MCMC, parameters = "S.high")
plot(MCMC, parameters = "P.low")
plot(MCMC, parameters = "deltaP")
plot(MCMC, parameters = "MaxHS")

plot(x=g.logistic, what = c("observations", "ML", "CI"),
        replicates=10000, resultmcmc=MCMC)

####### Exemple with Chelonia mydas
totalIncubation_Cm <- subset(DatabaseTSD,
Species=="Chelonia mydas" &
  Note != "Sinusoidal pattern" &
```

```
    !is.na(Total) & Total != 0 &
    !is.na(NotHatched) & !is.na(Hatched))

    totalIncubation_Cm$NotHatched <- totalIncubation_Cm$NotHatched +
    ifelse(!is.na(totalIncubation_Cm$Undeveloped), totalIncubation_Cm$Undeveloped, 0)


    plot(x=totalIncubation_Cm$Incubation.temperature,
    y=totalIncubation_Cm$Hatched/totalIncubation_Cm$Total, bty="n", las=1,
    xlab="Constant incubation temperature", ylab="Proportion of hatching")

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

  g.logistic <- HatchingSuccess.fit(par=par, data=totalIncubation_Cm)
  plot(g.logistic)

  pMCMC <- HatchingSuccess.MHmcmc_p(g.logistic, accept=TRUE)
  mcmc <- HatchingSuccess.MHmcmc(result=g.logistic, parameters = pMCMC,
                    adaptive=TRUE, n.iter=100000, trace=1000)
  par <- as.parameters(mcmc)
  par <- as.parameters(mcmc, index="median")

  plot(mcmc, parameters=c("P.low"))
  plot(mcmc, parameters=c("deltaP"))
  plot(mcmc, parameters=c("S.low"))
  plot(mcmc, parameters=c("S.high"))
  plot(mcmc, parameters=c("MaxHS"))

  plot(g.logistic, resultmcmc=mcmc, what = c("observations", "CI"))

  ## 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()`, `HatchingSuccess.MHmcmc_p()`, `HatchingSuccess.fit()`, `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 &
                               !is.na(NotHatched) & !is.na(Hatched))

par <- c(S.low=0.5, S.high=0.3,
         P.low=25, deltaP=10, MaxHS=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 <- HatchingSuccess.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 HatchingSuccess.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()`, `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=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()`, `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=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.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()`, `HatchingSuccess.MHmcmc_p()`, `HatchingSuccess.fit()`, `HatchingSuccess.lnL()`, `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=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)
```

`HeterogeneityNests` *Model heterogeneity of temperatures.*

Description

Generate a model of heterogeneity of temperatures.

Usage

```
HeterogeneityNests(
  nests = stop("An object of class Nests2, NestsResult, or mcmcComposite."),
  control.legend.total = list(),
  control.legend.metabolicheating = list(),
  show.full.incubation = TRUE,
  show.first.half.incubation = TRUE,
  ...
)
```

Arguments

`nests`	An object of class Nests, Nests2, NestsResult, or mcmcComposite.
`control.legend.total`	A list of options for legend.
`control.legend.metabolicheating`	A list of options for legend.
`show.full.incubation`	Show the plot with full incubation?
`show.first.half.incubation`	Show the plot with first half incubation?
`...`	Parameters used for plot.

Details

HeterogeneityNests models heterogeneity of temperatures.

Value

Nothing.

Author(s)

Marc Girondot <`marc.girondot@gmail.com`>

Examples

```
## Not run:
library(embryogrowth)
data(nest)
Laying.Time <- matrix(c("DY.1", "15/05/2010",
                "DY.17", "24/05/2010",
                "DY.16", "24/05/2010",
                "DY.18", "25/05/2010",
                "DY.20", "25/05/2010",
                "DY.21", "26/05/2010",
                "DY.22", "26/05/2010",
                "DY.23", "26/05/2010",
                "DY.24", "27/05/2010",
```

```
                      "DY.25", "27/05/2010",
                      "DY.28", "28/05/2010",
                      "DY.26", "28/05/2010",
                      "DY.27", "28/05/2010",
                      "DY.146", "20/06/2010",
                      "DY.147", "20/06/2010",
                      "DY.172", "24/06/2010",
                      "DY.175", "24/06/2010",
                      "DY.170", "24/06/2010",
                      "DY.260", "06/07/2010",
                      "DY.282", "12/07/2010",
                      "DY.310", "18/07/2010",
                      "DY.309", "18/07/2010",
                      "DY.328", "25/07/2010",
                      "DY.331", "26/07/2010"), byrow=TRUE, ncol=2)
tz <- OlsonNames()[grepl("Asia/Istanbul", OlsonNames())]
Laying.Time_f <- as.POSIXlt.character(Laying.Time[, 2], format = "%d/%m/%Y", tz=tz)
names(Laying.Time_f) <- Laying.Time[, 1]
nests <- FormatNests(data=nest, previous=NULL, col.Time="Time", LayingTime=Laying.Time_f)
HeterogeneityNests(nests, ylim=c(0, 4))

## End(Not run)
```

`hist.Nests2` *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 'Nests2'
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.Nests2 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@universite-paris-saclay.fr`>

Examples

```
## Not run:
library(embryogrowth)
data(nest)
formated <- FormatNests(nest)
h <- hist(x=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@universite-paris-saclay.fr`>

Examples

```
## Not run:
library(embryogrowth)
data(resultNest_4p_SSM)
h <- hist(resultNest_4p_SSM, 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 as compared to the final size.
Series can be indicated as the name of the series, their numbers or series or succession 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 with the suffix .mean, .se or .quantile_x with x from the parameter probs.

- `Temperature.max` Maximum temperature recorded during incubation
- `TimeWeighted.temperature` Average temperature during all incubation
- `GrowthWeighted.temperature` Average temperature weighted by the actual growth during all incubation
- `TimeWeighted.GrowthRateWeighted.temperature` Average temperature weighted by the growth rate during all incubation
- `TSP.TimeWeighted.temperature` Average temperature during the TSP
- `TSP.GrowthWeighted.temperature` Average temperature weighted by the actual growth during the TSP
- `TSP.TimeWeighted.GrowthRateWeighted.temperature` Average temperature weighted by the growth rate during the TSP
- `TSP.TimeWeighted.STRNWeighted.temperature` Average temperature weighted by the thermal reaction norm of sexualization during the TSP
- `TSP.GrowthWeighted.STRNWeighted.temperature` Average temperature weighted by actual growth and the thermal reaction norm of sexualization during the TSP
- `TSP.TimeWeighted.GrowthRateWeighted.STRNWeighted.temperature` Average temperature weighted by growth rate and the thermal reaction norm of sexualization during the TSP

- `TSP.length` TSP duration
- `TSP.begin` Beginning of the TSP
- `TSP.end` End of the TSP
- `TSP.PM.GrowthWeighted` Average of male probability for each temperature weighted by actual growth during the TSP
- `TSP.PM.TimeWeighted.GrowthRateWeighted` Average of male probability for each temperature weighted by growth rate during the TSP
- `TSP.PM.TimeWeighted` Average of male probability for each temperature during the TSP
- `Incubation.length` Incubation length duration
- `MiddleThird.length` Middle third incubation duration
- `MiddleThird.begin` Beginning of the middle third incubation duration
- `MiddleThird.end` End of the middle third incubation duration
- `MiddleThird.TimeWeighted.temperature` Average temperature during the middle third incubation
- `MiddleThird.GrowthWeighted.temperature` Average temperature weighted by the actual growth during the middle third incubation
- `MiddleThird.TimeWeighted.GrowthRateWeighted.temperature` Average temperature weighted by the growth rate during the middle third incubation
- `TSP.TimeWeighted.sexratio` Sex ratio based on average temperature during the TSP
- `TSP.GrowthWeighted.sexratio` Sex ratio based on average temperature weighted by the actual growth during the TSP
- `TSP.TimeWeighted.GrowthRateWeighted.sexratio` Sex ratio based on average temperature weighted by the growth rate during the TSP
- `TSP.TimeWeighted.STRNWeighted.sexratio` Sex ratio based on average temperature weighted by the thermal reaction norm of sexualization during the TSP
- `TSP.GrowthWeighted.STRNWeighted.sexratio` Sex ratio based on average temperature weighted by the actual growth and thermal reaction norm of sexualization during the TSP
- `TSP.TimeWeighted.GrowthRateWeighted.STRNWeighted.sexratio` Sex ratio based on average temperature weighted by the growth rate and the thermal reaction norm of sexualization during the TSP
- `MiddleThird.TimeWeighted.sexratio` Sex ratio based on average temperature during the middle third incubation
- `MiddleThird.GrowthWeighted.sexratio` Sex ratio based on average temperature weighted by actual growth during the middle third incubation
- `MiddleThird.TimeWeighted.GrowthRateWeighted.sexratio` Sex ratio based on average temperature weighted by growth rate during the middle third incubation
- `TimeWeighted.sexratio` Sex ratio based on average temperature during all incubation
- `GrowthWeighted.sexratio` Sex ratio based on average temperature weighted by actual growth during all incubation
- `TimeWeighted.GrowthRateWeighted.sexratio` Sex ratio based on average temperature weighted by growth rate during all incubation

If `out` is equal to `summary`, the return is a list with:

- `summary` is a data.frame with statistics for each nest.
- `dynamic.metric` object is a list composed of data.frames with the dynamics of growth for each nest. It showed only temperatures from original dataset.
- `summary.dynamic.metric` is a data.frame with the following columns with the suffix .mean, .se or .quantile_x with x from the parameter probs.

If `out` is equal to `details`, the return is a list with:

- The statistics for each replicate for each nest (one per element of the list)

If `out` is equal to `metric`, the return is a list with:

- `dynamic.metric` object is a list composed of data.frames with the dynamics of growth for each nest
- `indices.dynamic.metric` is a data.frame with the following columns.

The object `summary.dynamic.metric` or `indices.dynamic.metric` is a data.frame with the following columns:

- `series` Name of the series
- `metric.begin.tsp` Metric at the beginning of TSP
- `metric.end.tsp` Metric at the end of TSP
- `hatchling.metric.mean` Average expected size of hatchlings
- `hatchling.metric.sd` standard deviation of expected size of hatchlings
- `time.begin.tsp` Time at the beginning of TSP
- `time.end.tsp` Time at the end of TSP
- `time.begin.middlethird` Time at the beginning of the middle third incubation
- `time.end.middlethird` Time at the end of the middle third incubation
- `stop.at.hatchling.metric` Take the value of NA if stop.at.hatchling.metric was FALSE. TRUE if at least one incubation series was longer than hatchling size and FALSE at contrary
- `stop.at.hatchling.metric` Take the value of NA if stop.at.hatchling.metric was FALSE. TRUE if at least one incubation series was longer than hatchling size and FALSE at contrary
- `stop.at.hatchling.metric` Take the value of NA if stop.at.hatchling.metric was FALSE. TRUE if at least one incubation series was longer than hatchling size and FALSE at contrary
- `stop.at.hatchling.metric` Take the value of NA if stop.at.hatchling.metric was FALSE. TRUE if at least one incubation series was longer than hatchling size and FALSE at contrary
- `stop.at.hatchling.metric` Take the value of NA if stop.at.hatchling.metric was FALSE. TRUE if at least one incubation series was longer than hatchling size and FALSE at contrary
- `stop.at.hatchling.metric` Take the value of NA if stop.at.hatchling.metric 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 hatchling.metric 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`
- `Lepidochelys olivacea.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 substrate becomes dry.
The progress bar is based on both replicates and timeseries progress. It necessitates the pbapply package.
If replicate.CI is null or 0, only maximum likelihood is used and no confidence interval is calculated.
If replicate.CI is 1, one random value for the parameters is used but no confidence interval is calculated.
In other cases, replicate.CI random samples are used to estimate confidence interval.
About parallel computing:
Set options mc.cores ato tell what sort of parallel computing
Example:
options(mc.cores = detectCores())
If mc.cores is not defined, it will used detectCores() by default.

Usage

```
info.nests(
  x = NULL,
  parameters = NULL,
  NestsResult = NULL,
  resultmcmc = NULL,
  hessian = NULL,
  GTRN.CI = NULL,
  fixed.parameters = NULL,
  SE = NULL,
  temperatures = NULL,
  integral = NULL,
  derivate = NULL,
  hatchling.metric = NULL,
  stop.at.hatchling.metric = FALSE,
  M0 = NULL,
  series = "all",
  TSP.borders = NULL,
  embryo.stages = NULL,
  TSP.begin = 0,
  TSP.end = 0.5,
  replicate.CI = 0,
  weight = NULL,
```

```
  out = "likelihood",
  WAIC = TRUE,
  fill = NULL,
  probs = c(0.025, 0.5, 0.975),
  SexualisationTRN = NULL,
  SexualisationTRN.mcmc = NULL,
  SexualisationTRN.CI = NULL,
  metric.end.incubation = "observed",
  metabolic.heating = 0,
  temperature.heterogeneity = 0,
  progressbar = FALSE,
  warnings = TRUE,
  parallel = TRUE,
  tsd = NULL,
  tsd.CI = NULL,
  tsd.mcmc = NULL,
  zero = 1e-09,
  verbose = FALSE
)
```

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.
`GTRN.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.
`integral`	Function used to fit embryo growth: integral.Gompertz, integral.exponential or integral.linear. 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.
`hatchling.metric`	Mean and SD of size of hatchlings. It will replace the one in NestsResult.

`stop.at.hatchling.metric`	TRUE or FALSE. If TRUE, the model stops when proxy of size reached the mean hatchling.metric 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. See description
`weight`	Weights of the different nests to estimate likelihood. It will replace the ones in NestsResult.
`out`	Can take the values of "likelihood", "summary", "details", "metric" or "dynamic".
`WAIC`	Should WAIC data be returned?
`fill`	Number of minutes between two records. Create new one if they do not exist. NULL does not change the time of temperature recordings.
`probs`	Probabilities for metric quantiles.
`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 metric at the end of incubation used to calibrate TSP size. Can be "hatchling.metric", or "observed".
`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.
`progressbar`	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

`tsd.CI`	How to estimate CI for sex ratio thermal reaction norm; Can be NULL, "SE", "MCMC", or "Hessian".
`tsd.mcmc`	A object from tsd_MHmcmc() .
`zero`	Value to replace 0 or 1.
`verbose`	If TRUE, show more information.

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 `<marc.girondot@gmail.com>`

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.
Fuentes MM, Monsinjon J, Lopez M, Lara P, Santos A, dei Marcovaldi MA, Girondot M (2017). “Sex ratio estimates for species with temperature-dependent sex determination differ according to the proxy used.” *Ecological Modelling*, **365**, 55-67. doi:10.1016/j.ecolmodel.2017.09.022.
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**(2), 172-179. doi:10.2744/CCB1269.1.
Girondot M, Monsinjon J, Guillon J (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. doi:10.1016/j.jtherbio.2018.02.006.

Examples

```
## Not run:
library(embryogrowth)
data(resultNest_4p_SSM)
# Some basic calculations to show the advantage of parallel computing
system.time(summary.nests <- info.nests(x=resultNest_4p_SSM, out="summary",
  embryo.stages="Caretta caretta.SCL", replicate.CI=0, parallel=FALSE))
system.time(summary.nests <- info.nests(x=resultNest_4p_SSM, out="summary",
  embryo.stages="Caretta caretta.SCL", replicate.CI=0, parallel=TRUE))
system.time(summary.nests <- info.nests(x=resultNest_4p_SSM, out="summary",
  embryo.stages="Caretta caretta.SCL", replicate.CI=0, parallel=TRUE, progressbar=TRUE))
system.time(summary.nests <- info.nests(x=resultNest_4p_SSM, out="likelihood",
 embryo.stages="Caretta caretta.SCL", replicate.CI=0, parallel=TRUE, progressbar=FALSE))

# By default parallel computing is TRUE but progressbar is FALSE
```

```
# When out is "likelihood", it returns only the likelihood
# otherwise, it returns a list with 3 objects "summary",
#        "dynamic.metric", and "summary.dynamic.metric".

summary.nests <- info.nests(resultNest_4p_SSM, out="summary",
  embryo.stages="Caretta caretta.SCL",
  replicate.CI=100,
  resultmcmc=resultNest_mcmc_4p_SSM,
  GTRN.CI="MCMC",
  progressbar=TRUE)

summary.nests <- info.nests(resultNest_4p_SSM,
  embryo.stages="Caretta caretta.SCL",
  out="summary", replicate.CI=100,
  GTRN.CI="Hessian",
  progressbar=TRUE)

summary.nests <- info.nests(resultNest_4p_SSM,
  series = 1,
  embryo.stages="Caretta caretta.SCL",
  out="summary", replicate.CI=100,
  GTRN.CI="SE",
  progressbar=TRUE)

# Example of use of embryo.stages and TSP.borders:
  summary.nests <- info.nests(resultNest_4p_SSM, 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,
                             progressbar=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(x=resultNest_4p_SSM,
  GTRN.CI="Hessian", tsd.CI="Hessian",
  embryo.stages="Caretta caretta.SCL", replicate.CI=100,
  out="summary", progressbar=TRUE, tsd=tsdL)


plot_errbar(x=PM$summary$TimeWeighted.temperature.mean,
            y=PM$summary$TSP.PM.GrowthWeighted.mean,
```

```
            y.minus=PM$summary$TSP.PM.GrowthWeighted.quantile_0.025,
            y.plus=PM$summary$TSP.PM.GrowthWeighted.quantile_0.975,
            xlab="CTE SCL growth",
            ylab="PM Massey et al. 2016", xlim=c(26, 32), ylim=c(0, 1), las=1)

# Relationship between growth and growth rate

infoall.df <- info.nests(x=resultNest_4p_SSM, out="summary",
  embryo.stages="Caretta caretta.SCL",
  replicate.CI=100,
  resultmcmc=resultNest_mcmc_4p_SSM,
  GTRN.CI="MCMC",
  progressbar=TRUE)

  layout(1)
plot(x=infoall.df$dynamic.metric[[1]][, "Time"],
     y=infoall.df$dynamic.metric[[1]][, "Metric_50%"],
     type="l", las=1, bty="n",
     xlab="Time in minute", ylab="Growth", ylim=c(0, 39), xlim=c(0, 100000))
lines(x=infoall.df$dynamic.metric[[1]][, "Time"],
     y=infoall.df$dynamic.metric[[1]][, "Metric_2.5%"], lty=2)
lines(x=infoall.df$dynamic.metric[[1]][, "Time"],
     y=infoall.df$dynamic.metric[[1]][, "Metric_97.5%"], lty=2)

## End(Not run)
```

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

Description

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

Usage

```
integral.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

integral.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,
hatchling.metric=c(Mean=39.33, SD=1.92))

## End(Not run)
```

`integral.Gompertz` *Return the result of the Gompertz function*

Description

Return the result of the Gompertz function as a data.frame with two columns, time and metric integral.Gompertz(t, size, parms)

Usage

```
integral.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

integral.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_SSM <- searchR(parameters=x, fixed.parameters=pfixed,
temperatures=formated, integral=integral.Gompertz, M0=1.7,
hatchling.metric=c(Mean=39.33, SD=1.92))
data(resultNest_4p_SSM)

## End(Not run)
```

`integral.linear` *Return the linear function*

Description

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

Usage

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

Arguments

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

Details

integral.linear returns 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,
hatchling.metric=c(Mean=39.33, SD=1.92))

## 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,
  integral = NULL,
  derivate = NULL,
  hatchling.metric = 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
`integral`	Function used to fit embryo growth: integral.Gompertz, integral.exponential or integral.linear
`derivate`	Function used to fit embryo growth: dydt.Gompertz, dydt.exponential or dydt.linear. It will replace the one in NestsResult.
`hatchling.metric`	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 <`marc.girondot@gmail.com`>

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 integral.linear or integral.exponential
LresultNest_4p <- likelihoodR(parameters=x, fixed.parameters=pfixed,
temperatures=formated, integral=integral.Gompertz, M0=1.7,
hatchling.metric=c(Mean=39.33, SD=1.92))
data(resultNest_4p_SSM)
LresultNest_4p <- likelihoodR(result=resultNest_4p_SSM)

## 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()`, `HatchingSuccess.MHmcmc_p()`, `HatchingSuccess.fit()`, `HatchingSuccess.lnL()`, `HatchingSuccess.model()`, `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=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)
```

`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_SSM)
logLik(resultNest_4p_SSM)
AIC(resultNest_4p_SSM)

## 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_SSM)
logLik(resultNest_4p_SSM)
AIC(resultNest_4p_SSM)

## 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)
```

`movement` *Analyze movement recorded within a nest with an accelerometer datalogger*

Description

This function is used to evaluate significant movement within a nest.
The "quiet" period is the period without any expected move. It is used as a reference to detect the period with significant movements.
It returns a data.frame with the columns:
"Time", "x", "y", "z",
"mvt", "mvt_standardized", "peakmvt", "running", "mvt_MA_standardized",
"mvt_2", "mvt_2_standardized", "peakmvt_2", "running_2", "mvt_2_MA_standardized".
mvt and mvt_2 are two different methods. Often mvt_2 is better to descrobe movments.

Usage

```
movement(
  x = stop("data.frame must be provided"),
  col.time = "Time",
  col.x = "x",
  col.y = "y",
  col.z = "z",
  DaysQuiet = 40,
  SkipDays = 1,
  k = 4,
  Windowsize = 15
)
```

Arguments

`x`	A data.frame with 4 columns, one for time and three for x, y, and z position
`col.time`	Name of the column with time
`col.x`	Name of the column with x positions
`col.y`	Name of the column with y positions
`col.z`	Name of the column with z positions
`DaysQuiet`	Number of days in quiet period
`SkipDays`	Number of days to skip before being in quiet mode
`k`	Factor to multiply SD to prevent false positive detection
`Windowsize`	Number of records used for moving average

Details

movement is a function that permits to analyze movement datalogger

Value

The function will return a data.frame

Author(s)

Marc Girondot

References

Morales-Mérida BA, Contreras-Mérida MR, Girondot M (2019). "Pipping dynamics in marine turtle Lepidochelys olivacea nests." *Trends in Developmental Biology*, **12**, 23-30.

See Also

Other Data loggers utilities: `calibrate.datalogger()`, `uncertainty.datalogger()`

Examples

```
## Not run:
library(embryogrowth)
mv <- movement(x=dataf,
               col.time="Time",
               col.x="x", col.y="y", col.z="z")

## 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,
  GTRN.CI = "Hessian",
  tsd = NULL,
  tsd.CI = NULL,
  tsd.mcmc = NULL,
  SexualisationTRN = NULL,
  SexualisationTRN.CI = "Hessian",
  SexualisationTRN.mcmc = NULL,
  temperatures.df = stop("A data.frame must be provided"),
  temperature.heterogeneity = 0,
  metabolic.heating = 0,
  average.incubation.duration = 60 * 1440,
  max.time = 100 * 24 * 60,
  skip = 1,
  parameters = NULL,
  fixed.parameters = NULL,
  SE = NULL,
  hessian = NULL,
  integral = NULL,
  derivate = NULL,
  hatchling.metric = NULL,
  M0 = NULL,
  embryo.stages = "Caretta caretta.SCL",
  TSP.borders = c(21, 26),
  TSP.begin = 0,
  TSP.end = 0.5,
  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.
`GTRN.CI`	How to estimate CI for embryo growth thermal reaction norm; can be NULL, "SE", "MCMC", "pseudohessianfrommcmc" or "Hessian".
`tsd`	A object from tsd() that describe the thermal react norm of sex ratio at constant temperatures
`tsd.CI`	How to estimate CI for sex ratio thermal reaction norm; Can be NULL, "SE", "MCMC", "pseudohessianfrommcmc" or "Hessian".
`tsd.mcmc`	A object from tsd_MHmcmc()

`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", "pseudohessianfrommcmc" or "Hessian".
`SexualisationTRN.mcmc`	MCMC object for STRN.
`temperatures.df`	A data.frame with 2 or 3 columns: Times, Temperatures and Temperatures.end.incubation (facultative)
`temperature.heterogeneity`	SD of heterogeneity of temperatures. Can be 2 values, sd_low and sd_high and then HelpersMG::r2norm() is used.
`metabolic.heating`	Degrees Celsius to be added at the end of incubation due to metabolic heating
`average.incubation.duration`	The average time to complete incubation (not used if metabolic heating is setup)
`max.time`	Maximum time of incubation
`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
`integral`	Function used to fit embryo growth: integral.Gompertz, integral.exponential or integral.linear
`derivate`	Function used to fit embryo growth: dydt.Gompertz, dydt.exponential or dydt.linear. It will replace the one in NestsResult.
`hatchling.metric`	Mean and SD of size of hatchlings as a vector ie hatchling.metric=c(Mean=xx, SD=yy)
`M0`	Measure of hatchling size proxi at laying date
`embryo.stages`	The embryo stages. At least TSP.borders stages must be provided to estimate TSP length
`TSP.borders`	The limits of TSP
`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 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_SSM)
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_SSM,
     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(70, 80)))
with(out, plot(Time/(60*24), TSP.GrowthWeighted.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@universite-paris-saclay.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(), HatchingSuccess.MHmcmc_p(), HatchingSuccess.fit(), HatchingSuccess.lnL(), HatchingSuccess.model(), logLik.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=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)

nobs(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_SSM)
logLik(resultNest_4p_SSM)
AIC(resultNest_4p_SSM)
nobs(resultNest_4p_SSM)

## 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 replicates is 0, it returns only the fitted model.
If replicates is null and resultmcmc is not null, it will use all the mcmc data.
if replicates is lower than the number of iterations in resultmcmc, it will use sequence of data regularly thined.

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.ML = "black",
  lty.ML = 1,
  lwd.ML = 1,
  col.median = "black",
  lty.median = 2,
  lwd.median = 1,
  col.CI = "black",
  lty.CI = 3,
  lwd.CI = 1,
  replicates = NULL,
  resultmcmc = NULL,
  polygon = TRUE,
  color.polygon = rgb(red = 0.8, green = 0, blue = 0, alpha = 0.1),
  what = c("observations", "ML", "CI"),
  ...
)
```

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 graphical 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.ML`	Color of the maximum likelihood model
`lty.ML`	Line type of the maximum likelihood model (no line if NULL)
`lwd.ML`	Line width of the maximum likelihood 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 HatchingSuccess.MHmcmc()
`polygon`	If TRUE, confidence interval is shown as a polygon
`color.polygon`	The color used for polygon
`what`	Indicate what to plot: "observations", "ML", "CI"
`...`	Parameters for plot()

Details

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

Value

Nothing

Author(s)

Marc Girondot

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=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.Nests2` *Show the plot of temperatures with set of nests*

Description

Show the plot of temperatures with set of nests
If time is "absolute", LayingTime must be indicated in FormatNests()
xlim being auto-month used the closest begin and end of month and auto-year uses the clsest begin and end of year.

Usage

```
## S3 method for class 'Nests2'
plot(
  x,
  series = "all",
  time = "relative",
  show.heterogeneity = TRUE,
  probs.heterogeneity = c(0.025, 0.5, 0.975),
  col.heterogeneity = rgb(red = 0.2, green = 0.2, blue = 0.2, alpha = 0.6),
  show.legend.names = TRUE,
  show.legend.heterogeneity = TRUE,
  control.legend.heterogeneity = list(),
  control.legend.names = list(),
  col = "black",
  cex.axis = 1,
  xlim = "auto",
  xlab.axis = NULL,
  ...
)
```

Arguments

`x`	Data formated using formatdata.
`series`	Series to be used, logical (TRUE ou FALSE), numbers or names. If "all", all series are used.
`time`	Can be relative or absolute.
`show.heterogeneity`	If TRUE, show the 95% heterogeneity in grey.
`probs.heterogeneity`	Quantiles of heterogeneity.

`col.heterogeneity`	Color of heterogeneity.
`show.legend.names`	Show a legend for names.
`show.legend.heterogeneity`	Show the heterogeneity legend.
`control.legend.heterogeneity`	The list of parameters for legend of heterogeneity.
`control.legend.names`	The list of parameters for legend of names.
`col`	A recycled vector of colors.
`cex.axis`	The size of x-axis labels
`xlim`	The xlim parameter of plot or can be "auto", "auto-month", or "auto-year"
`xlab.axis`	Labels of x-axis
`...`	Parameters used by plot function

Details

plot.Nests2 shows the plot of temperatures for set of nests

Value

The position of labels if xaxt="n" is used.

Author(s)

Marc Girondot `<marc.girondot@universite-paris-saclay.fr>`

Examples

```
## Not run:
library(embryogrowth)
data(nest)
formated <- FormatNests(nest)
plot(x=formated, series=”all”, col=rainbow(21))
plot(x=formated, series=1, main=”DY.1”)
Laying.Time <- matrix(c(”DY.1”, ”15/05/2010”,
                ”DY.17”, ”24/05/2010”,
                ”DY.16”, ”24/05/2010”,
                ”DY.18”, ”25/05/2010”,
                ”DY.20”, ”25/05/2010”,
                ”DY.21”, ”26/05/2010”,
                ”DY.22”, ”26/05/2010”,
                ”DY.23”, ”26/05/2010”,
                ”DY.24”, ”27/05/2010”,
                ”DY.25”, ”27/05/2010”,
                ”DY.28”, ”28/05/2010”,
                ”DY.26”, ”28/05/2010”,
                ”DY.27”, ”28/05/2010”,
```

```
                 "DY.146", "20/06/2010",
                 "DY.147", "20/06/2010",
                 "DY.172", "24/06/2010",
                 "DY.175", "24/06/2010",
                 "DY.170", "24/06/2010",
                 "DY.260", "06/07/2010",
                 "DY.282", "12/07/2010",
                 "DY.310", "18/07/2010",
                 "DY.309", "18/07/2010",
                 "DY.328", "25/07/2010",
                 "DY.331", "26/07/2010"), byrow=TRUE, ncol=2)
tz <- OlsonNames()[grepl("Asia/Istanbul", OlsonNames())]
Laying.Time_f <- as.POSIXlt.character(Laying.Time[, 2], format = "%d/%m/%Y", tz=tz)
names(Laying.Time_f) <- Laying.Time[, 1]
formated <- FormatNests(data=nest, previous=NULL, col.Time="Time", LayingTime=Laying.Time_f)
plot(x=formated, time="absolute", ylim=c(20, 35),
     col= rainbow(21, alpha = 1), control.legend.heterogeneity=list(cex.0.5))

## 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.hatchling.metric 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`
- `Lepidochelys olivacea.SCL`
- `Generic.ProportionDevelopment`

Usage

```
## S3 method for class 'NestsResult'
plot(
  x,
  ...,
  parameters = NULL,
  fixed.parameters = NULL,
  resultmcmc = NULL,
  hessian = NULL,
  SE = NULL,
  temperatures = NULL,
  integral = NULL,
  derivate = NULL,
  hatchling.metric = NULL,
  stop.at.hatchling.metric = FALSE,
  M0 = NULL,
  weight = NULL,
  series = "all",
  TSP.borders = NULL,
  embryo.stages = NULL,
  STRN = NULL,
  TSP.begin = 0,
  TSP.end = 0.5,
  replicate.CI = 100,
  metric.end.incubation = "observed",
  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,
  GTRN.CI = NULL,
  show.metric = TRUE,
  show.fioritures = TRUE,
  show.temperatures = TRUE,
  show.PT = TRUE,
  PT = c(mean = NA, SE = NA),
  show.hatchling.metric = 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.
`integral`	Function used to fit embryo growth: integral.Gompertz, integral.exponential or integral.linear
`derivate`	Function used to fit embryo growth: dydt.Gompertz, dydt.exponential or dydt.linear. It will replace the one in NestsResult.
`hatchling.metric`	Mean and SD of size of hatchlings
`stop.at.hatchling.metric`	TRUE or FALSE. If TRUE, the model stops when proxy of size reached the mean hatchling.metric 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.
`STRN`	An object obtained from STRN()
`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 hatchling.metric 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?
`GTRN.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, GTRN.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.hatchling.metric`	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 <`marc.girondot@gmail.com`>

Examples

```
## Not run:
library(embryogrowth)
data(resultNest_4p_SSM)
plot(resultNest_4p_SSM, 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),
    GTRN.CI = "SE", replicate.CI = 100,
    embryo.stages="Caretta caretta.SCL")
plot(resultNest_4p_SSM, xlim=c(0,70), ylimT=c(22, 32), ylimS=c(0,45), series=1,
    GTRN.CI = "Hessian", replicate.CI = 100,
    embryo.stages="Caretta caretta.SCL")
plot(resultNest_4p_SSM, xlim=c(0,70), ylimT=c(22, 32), ylimS=c(0,45), series=1,
     resultmcmc = resultNest_mcmc_4p_SSM,
    GTRN.CI = "MCMC", replicate.CI = 100,
     embryo.stages="Caretta caretta.SCL")
# to plot all the nest at the same time, use
plot(resultNest_4p_SSM, 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_SSM, 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
nests <- resultNest_4p_SSM$data
plot(nests, series="all", col=rainbow(21))
```

```
plot(nests, series="all", col=rainbow(21), time="Absolute", ylim=c(20, 35))

## 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.
Girondot M (1999). “Statistical description of temperature-dependent sex determination using maximum likelihood.” *Evolutionary Ecology Research*, **1**(3), 479-486.
Godfrey MH, Delmas V, Girondot M (2003). “Assessment of patterns of temperature-dependent sex determination using maximum likelihood model selection.” *Ecoscience*, **10**(3), 265-272.
Abreu-Grobois FA, Morales-Mérida BA, Hart CE, Guillon J, Godfrey MH, Navarro E, Girondot M (2020). “Recent advances on the estimation of the thermal reaction norm for sex ratios.” *PeerJ*, **8**, e8451. doi:10.7717/peerj.8451, `https://peerj.com/articles/8451/`.
Hulin V, Delmas V, Girondot M, Godfrey MH, Guillon J (2009). “Temperature-dependent sex determination and global change: Are some species at greater risk?” *Oecologia*, **160**(3), 493-506.

Usage

```
## S3 method for class 'tsd'
plot(
  x,
  ...,
  show.observations = TRUE,
  show.observations.sd = TRUE,
  show.model = TRUE,
  males.freq = TRUE,
  show.PTRT = TRUE,
  las.x = 1,
  las.y = 1,
  lab.PT = paste0("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, 0.8),
```

```
  col.PT.CI = rgb(0.8, 0.8, 0.8, 0.8),
  show.CI = TRUE,
  warn = TRUE,
  use.ggplot = TRUE
)
```

Arguments

`x`	A result file generated by tsd()
`...`	Parameters for plot()
`show.observations`	Should the observations be shown
`show.observations.sd`	Should the observations SD 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
`use.ggplot`	Use ggplot graphics (experimental). TRUE or FALSE

Details

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

Value

Nothing

Author(s)

Marc Girondot

See Also

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

Examples

```
## Not run:
library(embryogrowth)
CC_AtlanticSW <- subset(DatabaseTSD, RMU.2010=="Atlantic, SW" &
                          Species=="Caretta caretta" & (!is.na(Sexed) & Sexed!=0))
tsdL <- with (CC_AtlanticSW, tsd(males=Males, females=Females,
                                 temperatures=Incubation.temperature.set,
                                 equation="logistic"))
# By default, it will return a ggplot object
# Here I show the advantage of using a ggplot object
g <- plot(tsdL)
# You can remove named layers. For example:
names(g$layers)
g$layers["Observations"] <- NULL; plot(g)
# And add some
# Due to a bug in ggplot, it is necessary to remove all names to obtain correct legends
names(g$layers) <- NULL
g + geom_point(data=CC_AtlanticSW, aes(x=Incubation.temperature.set, y=Males/Sexed,
               size = Sexed), inherit.aes = FALSE, show.legend = TRUE, shape=19)
# Force to use the original plot
plot(tsdL, use.ggplot = FALSE)

## End(Not run)
```

`plotR` *Plot 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 curve is case insensitive. If only parameters is given, curve must be ML.

Usage

```
plotR(
  result = NULL,
  resultmcmc = NULL,
  chain = 1,
  parameters = NULL,
  fixed.parameters = NULL,
  temperatures = NULL,
  curve = "ML quantiles",
  set.par = 1,
  ylim = c(0, 5),
  xlim = c(20, 35),
  xlimR = NULL,
  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,
  yaxt = "s",
  bty = "n",
  las = 1,
  xaxt = "s",
  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
`chain`	The chain to use in resultmcmc
`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
`curve`	What curve to show: "MCMC quantiles" or "MCMC mean-SD" based on mcmc or "ML" or "ML quantiles" or "ML mean-SE" for maximum-likelihood. Or "none"
`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
`xlimR`	description to show the curve
`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
`yaxt`	The yaxt parameter of y-axis

`bty`	Box around the pot
`las`	Orientation for labels in y axis
`xaxt`	The xaxt parameter of x-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 plots the fitted growth rate dependent on temperature and the density of the mcmc

Value

A list with the value of scaleY to be used with other plotR function and the plot data in xy list element

Author(s)

Marc Girondot `<marc.girondot@gmail.com>`

Examples

```
## Not run:
library(embryogrowth)
# Note that the confidence interval is not the same for mcmc and ml quantiles
plotR(result = resultNest_4p_SSM,
            resultmcmc=resultNest_mcmc_4p_SSM,
            curve = "MCMC quantiles", ylim=c(0, 8))
plotR(resultNest_4p_SSM, curve="ML quantiles", ylim=c(0, 6))
#################
plotR(resultmcmc=resultNest_mcmc_4p_SSM, ylim=c(0, 10),
            curve = "MCMC quantiles", show.density=TRUE)
#################
plotR(resultmcmc=resultNest_mcmc_4p_SSM,
            curve = "MCMC quantiles", polygon=TRUE, ylim=c(0, 10))
#################
plotR(resultmcmc=resultNest_mcmc_6p_SSM, ylim=c(0,8),
      curve = "MCMC quantiles", polygon=TRUE, col.polygon = rgb(0, 1, 0, 1))
```

```
plotR(resultmcmc=resultNest_mcmc_4p_SSM, ylim=c(0,8),
       curve = "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_SSM, ylim=c(0, 8),
             curve = "MCMC quantiles", show.density=FALSE)
plotR(resultmcmc=resultNest_mcmc_6p_SSM, col="red", ylim=c(0, 8),
             curve = "MCMC quantiles", show.density=FALSE,
             new=FALSE, scaleY=sy$scaleY)
#################
sy <- plotR(result=resultNest_6p_SSM, curve="none",
             scaleY=1E5,
             ylim=c(0, 8),
             show.hist = TRUE, new = TRUE, mar=c(4, 4, 1, 4))
#################
plotR(result=resultNest_6p_SSM, curve="ML",
             ylim=c(0, 8),
             show.hist = TRUE, ylimH=c(0,1), atH=c(0, 0.1, 0.2))
################
plotR(result = resultNest_4p_SSM, ylim=c(0, 8),
             resultmcmc=resultNest_mcmc_4p_SSM,
             show.density = TRUE,
             curve = "MCMC quantiles")
#################
plotR(resultmcmc=resultNest_mcmc_4p_SSM, ylim=c(0, 8),
             curve = "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_SSM)
# Get a set of parameters without transition
x1 <- resultNest_4p_SSM$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, integral=integral.Gompertz, M0=1.7,
hatchling.metric=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_SSM), 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 0, it returns only the fitted model.
If replicates is null and resultmcmc is not null, it will use all the mcmc data.
if replicates is lower than the number of iterations in resultmcmc, it will use sequence of data regularly thined.

Usage

```
## S3 method for class 'HatchingSuccess'
predict(
  object,
  ...,
  temperature = NULL,
  probs = c(0.025, 0.5, 0.975),
  replicates = NULL,
  resultmcmc = NULL,
  chain = 1
)
```

Arguments

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

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()`, `HatchingSuccess.MHmcmc_p()`, `HatchingSuccess.fit()`, `HatchingSuccess.lnL()`, `HatchingSuccess.model()`, `logLik.HatchingSuccess()`, `nobs.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=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,
  l = 0.05,
  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
`l`	Limit for TRT in sex ratio
`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`, `DatabaseTSD.version()`, `P_TRT()`, `ROSIE`, `ROSIE.version()`, `TSP.list`, `plot.tsd()`, `stages`, `tsd()`, `tsd_MHmcmc()`, `tsd_MHmcmc_p()`

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.
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.
If l is null, TRT is not estimated and only sex ratio is estimated.
if replicate.CI is null or 0, no replicate is used and only point estimate is done.
Standard error is calculated using resampling based on the Hessian matrix with Cholesky decomposition or using MCMC chain if resultmcmc is provided. In the former case, a replicate.CI random sample of the MCMC results will be used.

Usage

```
P_TRT(
  x = NULL,
  resultmcmc = NULL,
  fixed.parameters = 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()
`fixed.parameters`	Set of fixed parameters
`chain`	What chain to be used if 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 `<marc.girondot@gmail.com>`

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`, `DatabaseTSD.version()`, `ROSIE`, `ROSIE.version()`, `TSP.list`, `plot.tsd()`, `predict.tsd()`, `stages`, `tsd()`, `tsd_MHmcmc()`, `tsd_MHmcmc_p()`

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,
                                  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@universite-paris-saclay.fr`>

References

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

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, integral=integral.Gompertz, M0=0.3470893,
hatchling.metric=c(Mean=39.33, SD=1.92))
plotR(result=resultNest_3p_Dallwitz, show.hist = TRUE,
             ylim=c(0, 8), curve="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@universite-paris-saclay.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_SSM$par,
                initial.parameters = structure(c(73.4009010417375, 304.142079511996,
                                           27.4671689276281),
                                       .Names = c("k", "lambda", "scale")),
                control=list(maxit=1000))
pfixed <- c(rK=2.093313)
resultNest_3p_Weibull <- searchR(parameters=x$par, fixed.parameters=pfixed,
                         temperatures=formated, integral=integral.Gompertz, M0=1.7,
                         hatchling.metric=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_SSM, 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@universite-paris-saclay.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_SSM$par,
               initial.parameters = structure(c(3, 7, 11, 32),
                                .Names = c("Scale", "sdL", "sdH", "Peak")),
               control=list(maxit=1000))
pfixed <- c(rK=2.093313)
resultNest_4p_normal <- searchR(parameters=x$par, fixed.parameters=pfixed,
                         temperatures=formated, integral=integral.Gompertz, M0=1.7,
                         hatchling.metric=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_SSM, 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@universite-paris-saclay.fr`>

References

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

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, integral=integral.Gompertz, M0=0.3470893,
hatchling.metric=c(Mean=39.33, SD=1.92))
plotR(result=resultNest_4p_SSM, show.hist = TRUE,
             ylim=c(0, 8), curve="ML quantiles")

## End(Not run)
```

`resultNest_4p_SSM_Linear`

Fit using the nest database

Description

Fit using the nest database

Usage

```
resultNest_4p_SSM_Linear
```

Format

A list with fitted information about data(nest)

Details

Result of the fit using the nest database with linear progression

Author(s)

Marc Girondot <`marc.girondot@universite-paris-saclay.fr`>

Examples

```
## Not run:
library(embryogrowth)
data(nest)
formated <- FormatNests(nest)
pfixed <- NULL
M0 = 0
##############################################################################
```

```
# 4 parameters SSM
##############################################################################
x <- c('DHA' = 64.868697530424186, 'DHH' = 673.18292743646771,
       'T12H' = 400.90952554047749, 'Rho25' = 82.217237723502123)
resultNest_4p_SSM_Linear <- searchR(parameters=x, fixed.parameters=pfixed,
temperatures=formated, integral=integral.linear, M0=M0,
hatchling.metric=c(Mean=39.33, SD=1.92)/39.33)

## 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@universite-paris-saclay.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@universite-paris-saclay.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_SSM$par,
               initial.parameters = structure(c(3, 20, 40, 32),
                          .Names = c("Max", "LengthB", "LengthE", "Peak")),
               control=list(maxit=1000))
pfixed <- c(rK=2.093313)
resultNest_4p_trigo <- searchR(parameters=x$par, fixed.parameters=pfixed,
                          temperatures=formated, integral=integral.Gompertz, M0=1.7,
                          hatchling.metric=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_SSM, 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@universite-paris-saclay.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 integral.linear or integral.exponential
resultNest_4p_weight <- searchR(parameters=x,
fixed.parameters=pfixed, temperatures=formated,
integral=integral.Gompertz, M0=1.7, hatchling.metric=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@universite-paris-saclay.fr>

References

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

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, integral=integral.Gompertz, M0=0.3470893,
hatchling.metric=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@universite-paris-saclay.fr>`

References

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

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, integral=integral.Gompertz, M0=0.3470893,
hatchling.metric=c(Mean=39.33, SD=1.92))
plotR(result=resultNest_6p_SSM, show.hist = TRUE,
             ylim=c(0, 8), curve="ML")

## End(Not run)
```

`resultNest_mcmc_4p_SSM`

Result of the mcmc using the nest database

Description

Fit using the nest database

Usage

```
resultNest_mcmc_4p_SSM
```

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@universite-paris-saclay.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_SSM <- searchR(parameters=x, fixed.parameters=pfixed,
temperatures=formated, integral=integral.Gompertz, M0=1.7,
test=c(Mean=39.33, SD=1.92))
data(resultNest_4p_SSM)
pMCMC <- TRN_MHmcmc_p(resultNest_4p_SSM, accept=TRUE)
```

```
# Take care, it can be very long, sometimes several days
resultNest_mcmc_4p_SSM <- GRTRN_MHmcmc(result=resultNest_4p_SSM,
 adaptive = TRUE,
parametersMCMC=pMCMC, n.iter=10000, n.chains = 1, n.adapt = 0,
thin=1, trace=TRUE)
data(resultNest_mcmc_4p_SSM)
1-rejectionRate(as.mcmc(resultNest_mcmc_4p_SSM))
as.parameters(resultNest_mcmc_4p_SSM)
layout(mat=matrix(1:4, nrow = 2))
plot(resultNest_mcmc_4p_SSM, parameters = "all", scale.prior = TRUE, las = 1)
layout(mat=1)
plotR(resultNest_4p_SSM, resultmcmc=resultNest_mcmc_4p_SSM, ylim=c(0,8),
      curve = "MCMC quantiles", show.hist = TRUE, atH = c(0, 0.1, 0.2), ylimH = c(0, 1),
         main="Schoolfield, Sharpe & Magnuson 4-parameters", show.density=FALSE)
plot(resultNest_4p_SSM, resultmcmc=resultNest_mcmc_4p_SSM, series=1, GTRN.CI = "MCMC",
     replicate.CI=1000,
     embryo.stages="Caretta caretta.SCL", show.stages=TRUE, show.TSP=FALSE,
     show.third = FALSE, xlim=c(0, 70), las=1, ylimT = c(20, 35), ylab="SCL in mm")

## End(Not run)
```

---

resultNest_mcmc_4p_SSM_Linear

Result of the mcmc using the nest database

Description

Fit using the nest database

Usage

```
resultNest_mcmc_4p_SSM_Linear
```

Format

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

Details

Result of the mcmc using the nest database

Author(s)

Marc Girondot <marc.girondot@universite-paris-saclay.fr>

Examples

```
## Not run:
library(embryogrowth)
data(nest)
formated <- FormatNests(nest)
pfixed <- NULL
M0 = 0
#############################################################################
# 4 parameters SSM
#############################################################################
x <- c('DHA' = 64.868697530424186, 'DHH' = 673.18292743646771,
       'T12H' = 400.90952554047749, 'Rho25' = 82.217237723502123)
resultNest_4p_SSM_Linear <- searchR(parameters=x, fixed.parameters=pfixed,
temperatures=formated, integral=integral.linear, M0=M0,
hatchling.metric=c(Mean=39.33, SD=1.92)/39.33)
data(resultNest_4p_SSM_Linear)
pMCMC <- TRN_MHmcmc_p(resultNest_4p_SSM_Linear, accept=TRUE)
# Take care, it can be very long, sometimes several days
resultNest_mcmc_4p_SSM_Linear <- GRTRN_MHmcmc(result=resultNest_4p_SSM_Linear,
 adaptive = TRUE,
parametersMCMC=pMCMC, n.iter=10000, n.chains = 1, n.adapt = 0,
thin=1, trace=TRUE)
data(resultNest_mcmc_4p_SSM_Linear)
1-rejectionRate(as.mcmc(resultNest_mcmc_4p_SSM_Linear))
as.parameters(resultNest_mcmc_4p_SSM_Linear)
layout(mat=matrix(1:4, nrow = 2))
plot(resultNest_mcmc_4p_SSM_Linear, parameters = "all", scale.prior = TRUE, las = 1)
layout(mat=1)
plotR(resultNest_4p_SSM_Linear, resultmcmc=resultNest_mcmc_4p_SSM_Linear, ylim=c(0,2),
      curve = "MCMC quantiles", show.hist = TRUE, atH = c(0, 0.1, 0.2), ylimH = c(0, 1),
         main="Schoolfield, Sharpe & Magnuson 4-parameters", show.density=FALSE)
plot(resultNest_4p_SSM_Linear, resultmcmc=resultNest_mcmc_4p_SSM_Linear, series=1,
     GTRN.CI = "MCMC", replicate.CI=100,
     embryo.stages="Caretta caretta.SCL", show.stages=FALSE, show.TSP=FALSE,
     show.third = FALSE, xlim=c(0, 70), las=1, ylimT = c(20, 35), ylab="SCL in mm")

## End(Not run)
```

`resultNest_mcmc_6p_SSM`

Result of the mcmc using the nest database

Description

Fit using the nest database

Usage

```
resultNest_mcmc_6p_SSM
```

Format

A list of class mcmcComposite with mcmc result for data(nest) with 6 parameters and Gompertz model of growth

Details

Result of the mcmc using the nest database

Author(s)

Marc Girondot `<marc.girondot@universite-paris-saclay.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(115.758929130522, 428.649022170996, 503.687251738993,
12.2621455821612, 306.308841227278, 116.35048615105), .Names = c("DHA",
"DHH", "DHL", "DT", "T12L", "Rho25"))
pfixed <- c(rK=2.093313)
resultNest_6p_SSM <- searchR(parameters=x, fixed.parameters=pfixed,
temperatures=formated, integral=integral.Gompertz, M0=1.7,
hatchling.metric=c(Mean=39.33, SD=1.92))
data(resultNest_6p)
pMCMC <- TRN_MHmcmc_p(resultNest_6p_SSM, accept=TRUE)
# Take care, it can be very long, sometimes several days
resultNest_mcmc_6p_SSM <- GRTRN_MHmcmc(result=resultNest_6p_SSM,
 adaptive = TRUE,
parametersMCMC=pMCMC, n.iter=10000, n.chains = 1, n.adapt = 0,
thin=1, trace=TRUE)
data(resultNest_mcmc_6p_SSM)
plot(resultNest_mcmc_6p_SSM, parameters="T12L", main="", xlim=c(290, 320), bty="n")

## End(Not run)
```

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

Description

Fit using the nest database with anchored parameters

Usage

```
resultNest_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@universite-paris-saclay.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, integral=integral.Gompertz, M0=1.7,
  hatchling.metric=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
resultNest_mcmc_newp <- GRTRN_MHmcmc(result=resultNest_newp,
  parametersMCMC=pMCMC, n.iter=10000, n.chains = 1, n.adapt = 0,
thin=1, trace=TRUE)
data(resultNest_mcmc_newp)
data(resultNest_4p_SSM)
newp <- GenerateAnchor(nests=resultNest_4p_SSM, number.anchors=7)
# Here the confidence interval is built based on anchored parameters
```

```
plotR(resultNest_4p_SSM, parameters=newp, SE=resultNest_mcmc_newp$SD,
 ylim=c(0,4), ylimH=c(0,0.4), show.hist=TRUE, curve="ML quantiles")
# Here the confidence interval is built based on parametric SSM equation
data(resultNest_4p_SSM)
plotR(resultNest_4p_SSM, SE=resultNest_mcmc_4p_SSM$SD,
 ylim=c(0,4), ylimH=c(0,0.4), show.hist=TRUE, curve="ML quantiles")
plot(resultNest_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(resultNest_mcmc_newp, las=1, xlim=c(0,30), parameters="296.333333333333")
plot(resultNest_mcmc_newp, las=1, xlim=c(0,30), parameters=3)

## 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@universite-paris-saclay.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, integral=integral.Gompertz, M0=1.7,
hatchling.metric=c(Mean=39.33, SD=1.92))
data(resultNest_newp)
plotR(resultNest_newp)

## End(Not run)
```

`ROSIE` *Database of TSD information for turtles*

Description

Database of incubation information with sex ratio for turtles

Usage

```
ROSIE
```

Format

A dataframe with raw data.

Details

Database of TSD information for turtles

Author(s)

Caleb Krueger <`kruegeca@gmail.com`>

References

Krueger CJ, Janzen FJ (2022). "ROSIE, a database of reptilian offspring sex ratios and sex-determining mechanisms, beginning with Testudines." *Scientific Data*, **9**(1). ISSN 2052-4463, doi:10.1038/s41597021011081.

See Also

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

Examples

```
## Not run:
library(embryogrowth)
data(ROSIE)
ROSIE.version()
ROSIE <- cbind(ROSIE, RMU.2023=NA)
ROSIE[(ROSIE$Species == "Lepidochelys_olivacea") &
      (grepl("Orissa", ROSIE$Location)),
      "RMU.2023"] <- "Northeast Indian"
ROSIE[(ROSIE$Species == "Lepidochelys_olivacea") &
      (grepl("Oaxaca", ROSIE$Location) | grepl("Nancite", ROSIE$Location) |
      grepl("Destiladeras", ROSIE$Location) | grepl("Baja California", ROSIE$Location) |
      grepl("Sinaloa", ROSIE$Location) | grepl("Chocó, Colombia", ROSIE$Location) |
      grepl("Jalisco, Mexico", ROSIE$Location)),
      "RMU.2023"] <- "East Pacific"
ROSIE[(ROSIE$Species == "Lepidochelys_olivacea") &
      (grepl("Brazil", ROSIE$Location)),
      "RMU.2023"] <- "West Atlantic"

ROSIE[(ROSIE$Species == "Dermochelys_coriacea") &
      (grepl("Suriname", ROSIE$Location) | grepl("French Guiana", ROSIE$Location)),
      "RMU.2023"] <- "Northwest Atlantic"
ROSIE[(ROSIE$Species == "Dermochelys_coriacea") &
      (grepl("Playa Grande", ROSIE$Location)),
      "RMU.2023"] <- "East Pacific"
ROSIE[(ROSIE$Species == "Dermochelys_coriacea") &
      (grepl("Malaysia", ROSIE$Location)),
      "RMU.2023"] <- "West Pacific"

ROSIE[(ROSIE$Species == "Eretmochelys_imbricata") &
      (grepl("Florida", ROSIE$Location) | grepl("Antigua", ROSIE$Location) |
      grepl("Virgin Islands", ROSIE$Location) | grepl("Puerto Rico", ROSIE$Location) |
      grepl("Campeche", ROSIE$Location) | grepl("Yucatán", ROSIE$Location) |
      grepl("St. Kitts and Nevis", ROSIE$Location)),
      "RMU.2023"] <- "Northwest Atlantic"
ROSIE[(ROSIE$Species == "Eretmochelys_imbricata") &
      (grepl("Queensland", ROSIE$Location)),
      "RMU.2023"] <- "Southwest Pacific"
ROSIE[(ROSIE$Species == "Eretmochelys_imbricata") &
      (grepl("Brazil", ROSIE$Location)),
      "RMU.2023"] <- "Southwest Atlantic"


ROSIE[(ROSIE$Species == "Caretta_caretta") &
      (grepl("Georgia", ROSIE$Location) | grepl("South Carolina", ROSIE$Location) |
      grepl("North Carolina", ROSIE$Location) | grepl("Florida", ROSIE$Location) |
```

```
        grepl("Texas", ROSIE$Location)),
        "RMU.2023"] <- "Northwest Atlantic"
  ROSIE[(ROSIE$Species == "Caretta_caretta") &
        (grepl("Cyprus", ROSIE$Location) | grepl("Greece", ROSIE$Location) |
        grepl("Turkey", ROSIE$Location)),
        "RMU.2023"] <- "Mediterranean"
  ROSIE[(ROSIE$Species == "Caretta_caretta") &
        (grepl("Queensland", ROSIE$Location)),
        "RMU.2023"] <- "South Pacific"
  ROSIE[(ROSIE$Species == "Caretta_caretta") &
        (grepl("Western Australia", ROSIE$Location)),
        "RMU.2023"] <- "Southeast Indian"
  ROSIE[(ROSIE$Species == "Caretta_caretta") &
        (grepl("South Africa", ROSIE$Location)),
        "RMU.2023"] <- "Southwest Indian"
  ROSIE[(ROSIE$Species == "Caretta_caretta") &
        (grepl("Japan", ROSIE$Location)),
        "RMU.2023"] <- "North Pacific"
  ROSIE[(ROSIE$Species == "Caretta_caretta") &
        (grepl("Brazil", ROSIE$Location)),
        "RMU.2023"] <- "Southwest Atlantic"

  ROSIE[(ROSIE$Species == "Chelonia_mydas") &
        (grepl("Queensland", ROSIE$Location)),
        "RMU.2023"] <- "Southwest Pacific"
  ROSIE[(ROSIE$Species == "Chelonia_mydas") &
       (grepl("Tortuguero", ROSIE$Location) | grepl("British West Indies", ROSIE$Location)),
        "RMU.2023"] <- "North Atlantic"
  ROSIE[(ROSIE$Species == "Chelonia_mydas") &
        (grepl("Suriname", ROSIE$Location) | grepl("Ascension Island", ROSIE$Location) |
        grepl("Guinea-Bissau", ROSIE$Location)),
        "RMU.2023"] <- "South Atlantic"
  ROSIE[(ROSIE$Species == "Chelonia_mydas") &
        (grepl("Malaysia", ROSIE$Location) | grepl("Phillipines", ROSIE$Location) |
        grepl("China", ROSIE$Location) | grepl("Taiwan", ROSIE$Location) |
        grepl("Western Australia", ROSIE$Location)),
        "RMU.2023"] <- "East Indian and Southeast Asia"
  ROSIE[(ROSIE$Species == "Chelonia_mydas") &
        (grepl("Japan", ROSIE$Location)),
        "RMU.2023"] <- "West Central Pacific"
  ROSIE[(ROSIE$Species == "Chelonia_mydas") &
        (grepl("Cyprus", ROSIE$Location) | grepl("Turkey", ROSIE$Location)),
        "RMU.2023"] <- "Mediterranean"
  ROSIE[(ROSIE$Species == "Chelonia_mydas") &
        (grepl("Oman", ROSIE$Location)),
        "RMU.2023"] <- "Northwest Indian"
  ROSIE[(ROSIE$Species == "Chelonia_mydas") &
        (grepl("Hawaii", ROSIE$Location)),
        "RMU.2023"] <- "North Central Pacific"

  ROSIE[(ROSIE$Species == "Natator_depressus"),
        "RMU.2023"] <- "nd"
```

```
ROSIE[(ROSIE$Species == "Lepidochelys_kempii"),
      "RMU.2023"] <- "Northwest Atlantic"


totalIncubation_Lo <- subset(ROSIE,
         (Species == "Lepidochelys_olivacea") & (!is.na(Total_Sexed) & Total_Sexed!=0) &
         (Incubation_Setup == "Constant"),
      select=c("Males", "Females", "Mean_Temp", "Latitude", "Longitude", "Location", "RMU.2023"))

library(mapdata)
map()
scale <- 50
title(bquote("Species name: "*italic(.("Lepidochelys olivacea"))))
for (l in unique(totalIncubation_Lo$Location)) {
  SR_sub <- subset(totalIncubation_Lo, subset = Location == l)
  points(x=SR_sub$Longitude[1], y=SR_sub$Latitude[1], pch=19,
  col= 1 + which(l == unique(totalIncubation_Lo$Location)),
  cex=sum(SR_sub$Males + SR_sub$Females, na.rm = TRUE)/scale)
}

tot_Lo <- with(totalIncubation_Lo, tsd(males=Males, females=Females,
 temperatures=Mean_Temp), parameters.initial = c(P=30.5, S=-0.4))
 plot(tot_Lo, xlim=c(20, 40))
 plot(tot_Lo, xlim=c(20, 40), use.ggplot=FALSE)
 predict(tot_Lo)

## End(Not run)
```

`ROSIE.version` *Version of database of TSD information for reptiles*

Description

Return the date of the most recent update of the ROSIE database.

Usage

```
ROSIE.version()
```

Details

Database of information for incubation of turtles

Value

The date of the latest updated version

Author(s)

Marc Girondot `<marc.girondot@universite-paris-saclay.fr>`

See Also

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

Examples

```
## Not run:
library(embryogrowth)
ROSIE.version()

## End(Not run)
```

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

Description

Fit the parameters that best represent data.
hatchling.metric 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 model 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"),
  temperatures = stop("Formated temperature must be provided"),
  fixed.parameters = c(rK = 1.208968),
  integral = integral.Gompertz,
  derivate = NULL,
  hatchling.metric = c(Mean = 39.33, SD = 1.92),
  M0 = 0.3470893,
  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
`temperatures`	Timeseries of temperatures after formated using FormatNests()
`fixed.parameters`	A set of parameters that will not be changed
`integral`	Function used to fit embryo growth: integral.Gompertz, integral.exponential or integral.linear
`derivate`	Function used to fit embryo growth: derivate.Gompertz, derivate.exponential or derivate.linear
`hatchling.metric`	A vector with Mean and SD of size of hatchlings, ex. hatchling.metric=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 `<marc.girondot@gmail.com>`

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.

Fuentes MM, Monsinjon J, Lopez M, Lara P, Santos A, dei Marcovaldi MA, Girondot M (2017). “Sex ratio estimates for species with temperature-dependent sex determination differ according to the proxy used.” *Ecological Modelling*, **365**, 55-67. doi:10.1016/j.ecolmodel.2017.09.022.

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**(2), 172-179. doi:10.2744/CCB1269.1.

Girondot M, Monsinjon J, Guillon J (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. doi:10.1016/j.jtherbio.2018.02.006.

Angilletta MJ (2006). “Estimating and comparing thermal performance curves.” *Journal of Thermal Biology*, **31**(7), 541-545. ISSN 03064565, doi:10.1016/j.jtherbio.2006.06.002.

Georges A, Beggs K, Young JE, Doody JS (2005). “Modelling development of reptile embryos under fluctuating temperature regimes.” *Physiological and Biochemical Zoology*, **78**, 18-30.

Dallwitz MJ, Higgins JP (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.

Examples

```
## Not run:
library(embryogrowth)
data(nest)
Laying.Time <- matrix(c("DY.1", "15/05/2010",
                 "DY.17", "24/05/2010",
                 "DY.16", "24/05/2010",
                 "DY.18", "25/05/2010",
                 "DY.20", "25/05/2010",
                 "DY.21", "26/05/2010",
                 "DY.22", "26/05/2010",
                 "DY.23", "26/05/2010",
                 "DY.24", "27/05/2010",
                 "DY.25", "27/05/2010",
                 "DY.28", "28/05/2010",
                 "DY.26", "28/05/2010",
                 "DY.27", "28/05/2010",
                 "DY.146", "20/06/2010",
                 "DY.147", "20/06/2010",
                 "DY.172", "24/06/2010",
```

```
                  "DY.175", "24/06/2010",
                  "DY.170", "24/06/2010",
                  "DY.260", "06/07/2010",
                  "DY.282", "12/07/2010",
                  "DY.310", "18/07/2010",
                  "DY.309", "18/07/2010",
                  "DY.328", "25/07/2010",
                  "DY.331", "26/07/2010"), byrow=TRUE, ncol=2)
tz <- OlsonNames()[grepl("Asia/Istanbul", OlsonNames())]
Laying.Time_f <- setNames(as.POSIXlt.character(Laying.Time[, 2], format = "%d/%m/%Y", tz=tz),
                          Laying.Time[, 1])
formated <- FormatNests(data=nest, previous=NULL, col.Time="Time",
                        LayingTime=Laying.Time_f,
                        hatchling.metric.mean=39.33,
                        hatchling.metric.sd=1.92)
plot(formated, series=c(1, 2), lwd=3)
# 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
########################################################################
# From 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
#  rK = 1.208968
#  M0 = 0.3470893
########################################################################
pfixed <- c(rK=1.208968)
M0 = 0.3470893
########################################################################
# 4 parameters SSM
########################################################################
x4 <- c('DHA' = 109.31113503282113, 'DHH' = 617.80695919563857,
    'T12H' = 306.38890489505093, 'Rho25' = 229.37265815800225)

resultNest_4p_SSM <- searchR(parameters=x4, fixed.parameters=pfixed,
                            temperatures=formated,
                            integral=integral.Gompertz, M0=M0)

data(resultNest_4p_SSM)
plot(resultNest_4p_SSM, xlim=c(0,70), ylimT=c(22, 32),
     ylimS=c(0,45), series=1,
     embryo.stages="Caretta caretta.SCL")
plotR(resultNest_4p_SSM, ylim=c(0,6))

########################################################################
# 6 parameters SSM
########################################################################
x6 <- structure(c(106.567809092008, 527.359011254683, 614.208632495199,
2720.94506457237, 306.268259715624, 120.336791245212), .Names = c("DHA",
```

```
"DHH", "DHL", "DT", "T12L", "Rho25"))

###############################################################################
# example of data.frame for hatchling.metric
###############################################################################
thatchling.metric <- data.frame(Mean = rep(39.33, formated$IndiceT["NbTS"]),
                     SD = rep(1.92, formated$IndiceT["NbTS"]),
                     row.names = formated$Names)
# It is sometimes difficult to find a good starting point for
# SSM 6 parameters model. This function helps to find it based on a previoulsy
# fitted model.

 x <- ChangeSSM(temperatures = (200:350)/10,
                parameters = resultNest_4p_SSM$par,
                initial.parameters = x6,
                control=list(maxit=1000))

resultNest_6p_SSM <- searchR(parameters=x$par, fixed.parameters=pfixed,
                        temperatures=formated, integral=integral.Gompertz,
                        M0=M0,
                        hatchling.metric=thatchling.metric)

plotR(resultNest_6p_SSM, curve = "ML", ylim=c(0, 8))

data(resultNest_6p_SSM)
pMCMC <- TRN_MHmcmc_p(resultNest_6p_SSM, accept=TRUE)
# Take care, it can be very long, sometimes several days
resultNest_mcmc_6p_SSM <- GRTRN_MHmcmc(result=resultNest_6p_SSM,
                                      parametersMCMC=pMCMC,
                                      n.iter=10000,
                                      n.chains = 1,
                                      n.adapt = 0,
                                      thin=1,
                                      trace=TRUE)

###############################################################################
# compare_AIC() is a function from the package "HelpersMG"
compare_AIC(test1=resultNest_4p_SSM, test2=resultNest_6p_SSM)
###############################################################################

###############################################################################
############# Example as linear progression of development
############# The development progress goes from 0 to 1
###############################################################################

pfixed <- NULL
M0 = 0

###############################################################################
# 4 parameters SSM
###############################################################################
x4 <- c('DHA' = 64.868697530424186, 'DHH' = 673.18292743646771,
       'T12H' = 400.90952554047749, 'Rho25' = 82.217237723502123)
```

```
resultNest_4p_SSM_Linear <- searchR(parameters=x4, fixed.parameters=pfixed,
temperatures=formated, integral=integral.linear, M0=M0,
hatchling.metric=c(Mean=39.33, SD=1.92)/39.33)
plotR(resultNest_4p_SSM_Linear, ylim=c(0, 2), scaleY= 100000, curve = "ML")
plot(resultNest_4p_SSM_Linear, xlim=c(0,70), ylimT=c(22, 32), ylimS=c(0,1.1),
series=1, embryo.stages="Generic.ProportionDevelopment")

tc <- GenerateConstInc(duration=600*24*60, temperatures = 28)
tc_f <- FormatNests(tc)

plot(x=resultNest_4p_SSM_Linear, xlim=c(0,70), ylimT=c(22, 32), ylimS=c(0,1.1),
     series=1, embryo.stages="Generic.ProportionDevelopment",
     stop.at.hatchling.metric=TRUE, metric.end.incubation="hatchling.metric",
     temperatures=tc_f, hatchling.metric=c(Mean=39.33, SD=1.92)/39.33,
     show.TSP=FALSE)

############################################################################
############ with new parametrization based on anchor
############ This is a non-parametric version
############################################################################

data(resultNest_4p_SSM)
x0 <- resultNest_4p_SSM$par
t <- range(hist(resultNest_4p_SSM, plot=FALSE)$temperatures)

x <- getFromNamespace(".SSM", ns="embryogrowth")(T=seq(from=t[1],
                                                       to=t[2],
                                                       length.out=7),
                                            parms=x0)[[1]]*1E5
names(x) <- as.character(seq(from=t[1],
                          to=t[2],
                          length.out=7))

M0 <- 0.3470893
pfixed <- c(rK=1.208968)
resultNest_newp <- searchR(parameters=x, fixed.parameters=pfixed,
                           temperatures=formated,
                           integral=integral.Gompertz, M0=M0,
                           hatchling.metric=c(Mean=39.33, SD=1.92))
plotR(resultNest_newp, ylim=c(0, 2),
      xlim=c(23, 34), ylimH=c(0, 3), show.hist=TRUE)
compare_AIC(test4p=resultNest_4p_SSM,
            test6p=resultNest_6p_SSM,
            testAnchor=resultNest_newp)

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

 x <- ChangeSSM(temperatures = (200:350)/10,
                parameters = resultNest_4p_SSM$par,
                initial.parameters = structure(c(73.4009010417375, 304.142079511996,
                                               27.4671689276281),
```

```
                                        .Names = c("k", "lambda", "scale")),
                control=list(maxit=1000))
M0 <- 0.3470893
pfixed <- c(rK=1.208968)
resultNest_3p_Weibull <- searchR(parameters=x$par, fixed.parameters=pfixed,
                         temperatures=formated, integral=integral.Gompertz, M0=M0,
                         hatchling.metric=c(Mean=39.33, SD=1.92))
plotR(resultNest_3p_Weibull, ylim=c(0,6), col="Black")
compare_AIC(SSM=resultNest_4p_SSM, Weibull=resultNest_3p_Weibull)

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

x <- ChangeSSM(temperatures = (200:350)/10,
               parameters = resultNest_4p_SSM$par,
               initial.parameters = structure(c(3, 7, 11, 32),
                               .Names = c("Scale", "sdL", "sdH", "Peak")),
               control=list(maxit=1000))
M0 <- 0.3470893
pfixed <- c(rK=1.208968)
resultNest_4p_normal <- searchR(parameters=x$par, fixed.parameters=pfixed,
                         temperatures=formated, integral=integral.Gompertz, M0=M0,
                         hatchling.metric=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_SSM$par,
               initial.parameters = structure(c(3, 20, 40, 32),
               .Names = c("Max", "LengthB", "LengthE", "Peak")),
               control=list(maxit=1000))
M0 <- 0.3470893
pfixed <- c(rK=1.208968)
resultNest_4p_trigo <- searchR(parameters=x$par, fixed.parameters=pfixed,
                         temperatures=formated, integral=integral.Gompertz, M0=M0,
                         hatchling.metric=c(Mean=39.33, SD=1.92))

###############################################################
# Example with thermal reaction norm fitted from Dallwitz model
###############################################################
# See: 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.

# Note that Dallwitz model has many problems and I recommend to not use it:
# - The 3-parameters is too highly constraint
# - The 5 parameters produced infinite outputs for some sets of parameters that
#   can be generated while using delta method.

x <- c('Dallwitz_b1' = 4.8854060791241816,
```

```
       'Dallwitz_b2' = 20.398366565842029,
       'Dallwitz_b3' = 31.510995256647092)
M0 <- 0.3470893
pfixed <- c(rK=1.208968)
resultNest_3p_Dallwitz <- searchR(parameters=x, fixed.parameters=pfixed,
                         temperatures=formated, integral=integral.Gompertz, M0=M0,
                         hatchling.metric=c(Mean=39.33, SD=1.92))
plotR(resultNest_3p_Dallwitz, ylim=c(0,6))

x <- c('Dallwitz_b1' = 4.9104386262684656,
       'Dallwitz_b2' = 7.515425231891359,
       'Dallwitz_b3' = 31.221784599026638,
       'Dallwitz_b4' = 7.0354467023505682,
       'Dallwitz_b5' = -1.5955717975708577)
pfixed <- c(rK=1.208968)
resultNest_5p_Dallwitz <- searchR(parameters=x, fixed.parameters=pfixed,
                        temperatures=formated, integral=integral.Gompertz, M0=0.3470893,
                         hatchling.metric=c(Mean=39.33, SD=1.92))
plotR(resultNest_5p_Dallwitz, ylim=c(0,3), scaleY=10000)

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

xp <- ChangeSSM(parameters = resultNest_3p_Dallwitz$par,
                initial.parameters = resultNest_4p_SSM$par)
pfixed <- c(rK=1.208968)
resultNest_4p_SSM <- searchR(parameters=xp$par, fixed.parameters=pfixed,
                        temperatures=formated, integral=integral.Gompertz, M0=0.3470893,
                         hatchling.metric=c(Mean=39.33, SD=1.92))
plotR(resultNest_4p_SSM, ylim=c(0,6))

compare_AIC(Dallwitz3p=resultNest_3p_Dallwitz, Dallwitz5p=resultNest_5p_Dallwitz,
             SSM=resultNest_4p_SSM, SSM=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, integral=integral.linear, M0=0,
                         hatchling.metric=c(Mean=1, SD=NA))
 plotR(resultNest_PropDev_3p_Dallwitz, ylim=c(0, 1.5), curve="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, integral=integral.linear, M0=0,
                          hatchling.metric=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), curve="ML")
 plotR(resultNest_PropDev_5p_Dallwitz, ylim=c(0, 1.5), curve="ML", new=FALSE, col="red")
 compare_AICc(Dallwitz3p=resultNest_PropDev_3p_Dallwitz,
              Dallwitz5p=resultNest_PropDev_5p_Dallwitz)

############################################################################
# Dalwitz model with proportion of development and fitted SD for final size
############################################################################

x <- c('Dallwitz_b1' = 1.4886497996404355,
       'Dallwitz_b2' = 10.898310418085916,
       'Dallwitz_b3' = 31.263224721068056,
       'Dallwitz_b4' = 6.1624623077734535,
       'Dallwitz_b5' = -1.0027132357973265,
       'SD' = 0.041829475961912894)
resultNest_PropDev_5p_Dallwitz <- searchR(parameters=x, fixed.parameters=NULL,
                          temperatures=formated, integral=integral.linear, M0=0,
                          hatchling.metric=c(Mean=1))
 plotR(resultNest_PropDev_5p_Dallwitz, ylim=c(0, 1.5), curve="ML")
 # Note that the standard error of the curve cannot be estimated with delta method.
 # MCMC should be used
 plot(x=resultNest_PropDev_5p_Dallwitz, ylimS=c(0,1), xlim=c(0,60), series=2,
         embryo.stages="Generic.ProportionDevelopment")

###############################################################################
# Parameters Threshold_Low and Threshold_High are used to truncate growth rate
###############################################################################

plotR(result=resultNest_PropDev_5p_Dallwitz,
     fixed.parameters=c(Threshold_Low=26,
                        Threshold_High=33),
     ylim=c(0, 1.5), curve="ML")

## 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@universite-paris-saclay.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`, `DatabaseTSD.version()`, `P_TRT()`, `ROSIE`, `ROSIE.version()`, `TSP.list`, `plot.tsd()`, `predict.tsd()`, `tsd()`, `tsd_MHmcmc()`, `tsd_MHmcmc_p()`

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 sexratio parameter is a character string which can be:

- `TSP.TimeWeighted.sexratio.mean` Sex ratio based on average temperature during the TSP
- `TSP.GrowthWeighted.sexratio.mean` Sex ratio based on average temperature weighted by the actual growth during the TSP
- `TSP.TimeWeighted.GrowthRateWeighted.sexratio.mean` Sex ratio based on average temperature weighted by the growth rate during the TSP
- `TSP.TimeWeighted.STRNWeighted.sexratio.mean` Sex ratio based on average temperature weighted by the thermal reaction norm of sexualization during the TSP
- `TSP.GrowthWeighted.STRNWeighted.sexratio.mean` Sex ratio based on average temperature weighted by the actual growth and thermal reaction norm of sexualization during the TSP
- `TSP.TimeWeighted.GrowthRateWeighted.STRNWeighted.sexratio.mean` Sex ratio based on average temperature weighted by the growth rate and the thermal reaction norm of sexualization during the TSP
- `MiddleThird.TimeWeighted.sexratio.mean` Sex ratio based on average temperature during the middle third incubation
- `MiddleThird.GrowthWeighted.sexratio.mean` Sex ratio based on average temperature weighted by actual growth during the middle third incubation
- `MiddleThird.TimeWeighted.GrowthRateWeighted.sexratio.mean` Sex ratio based on average temperature weighted by growth rate during the middle third incubation
- `TimeWeighted.sexratio.mean` Sex ratio based on average temperature during all incubation

- `GrowthWeighted.sexratio.mean` Sex ratio based on average temperature weighted by actual growth during all incubation
- `TimeWeighted.GrowthRateWeighted.sexratio.mean` Sex ratio based on average temperature weighted by growth rate during all incubation
- `TSP.PM.TimeWeighted.mean` Average sex ratio based on temperature during the TSP
- `TSP.PM.GrowthWeighted.mean` Average sex ratio based on temperature weighted by the actual growth during the TSP
- `TSP.PM.TimeWeighted.GrowthRateWeighted.mean` Average sex ratio based on temperature weighted by the growth rate during the TSP

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`
- `Lepidochelys olivacea.SCL`
- `Generic.ProportionDevelopment`

A fifth name `fitted` must be used when limits of TSP are fitted using `BeginTSP` and `EndTSP` parameters.
The parameters that can be used in STRN are:
`BeginTSP`, `EndTSP` are the logit of the proportion of development;
To ensure that `BeginTSP` < `EndTSP`, it is better to use:
`BeginTSP`, `LengthTSP` and then `EndTSP` is estimated using `BeginTSP` + `abs(LengthTSP)`
`DHA`, `DHH`, `T12H` are the SSM parameters of sexualisation thermal reaction norm;
`dbeta_mu`, `dbeta_v` are the beta mean and variance of the impact of sexualisation according to TSP progress.
Or any parameter that can be used in a TSD model.

Usage

```
STRN(
  EmbryoGrowthTRN = stop("Embryo Growth Thermal Reaction Norm must be provided"),
  Initial_STRN = NULL,
  fixed.parameters = NULL,
  TSP.borders = NULL,
  embryo.stages = NULL,
  TSP.begin = 0,
  TSP.end = 0.5,
  tsd = NULL,
```

```
  equation = "logistic",
  Sexed = NULL,
  Males = NULL,
  Females = NULL,
  sexratio = "TSP.TimeWeighted.GrowthRateWeighted.STRNWeighted.sexratio.mean",
  fill = 60,
  parallel = TRUE,
  itnmax = 1000,
  method = c("Nelder-Mead", "BFGS"),
  control = list(trace = 1, REPORT = 10),
  zero = 1e-09,
  verbose = FALSE,
  hessian = TRUE
)
```

Arguments

`EmbryoGrowthTRN`	The Embryo Growth Thermal Reaction Norm obtained with searchR()
`Initial_STRN`	Values for initial model of Sexualisation Thermal Reaction Norm or tsd model
`fixed.parameters`	Value for Sexualisation Thermal Reaction Norm or tsd model that will not be changed
`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.
`tsd`	The model used to predict sex ratio, obtained from tsd()
`equation`	If tsd parameter is not provided, equation and parameters in Initial_STRN 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
`sexratio`	The sex ratio to be used
`fill`	See info.nests()
`parallel`	Should parallel computing for info.nests() be used
`itnmax`	Maximum number of iterations for each method; if 0, just return the likelihood
`method`	Methods to be used with optimx
`control`	List for control parameters for optimx
`zero`	The value to replace a null sex ratio
`verbose`	If TRUE, will show all intermediate parameters during fit
`hessian`	If TRUE, the Hessian approximation is estimated atthe end of the fit.

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 <marc.girondot@gmail.com>

Examples

```
## Not run:
library(embryogrowth)
MedIncubation_Cc <- subset(DatabaseTSD, Species=="Caretta caretta" &
RMU=="Mediterranean" & Sexed!=0)
Med_Cc <- tsd(males=MedIncubation_Cc$Males,
             females=MedIncubation_Cc$Females,
             temperatures=MedIncubation_Cc$Incubation.temperature,
             par=c(P=29.5, S=-0.1))
plot(Med_Cc, xlim=c(25, 35))
males <- c(7, 0, 0, 0, 0, 5, 6, 3, 5, 3, 2, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0)
names(males) <- rev(rev(names(resultNest_4p_SSM$data))[-(1:2)])
sexed <- rep(10, length(males))
names(sexed) <- rev(rev(names(resultNest_4p_SSM$data))[-(1:2)])

Initial_STRN <- c('DHA' = 1174.6461503413307,
                  'DHH' = 2001.0619192107047,
                  'T12H' = 3731.353104743393)
fp <- c(Rho25=100)
fitSTRN <- STRN(Initial_STRN=Initial_STRN,
                EmbryoGrowthTRN=resultNest_4p_SSM,
                tsd=Med_Cc,
                embryo.stages="Caretta caretta.SCL",
                Sexed=sexed, Males=males,
                fixed.parameters=fp,
                sexratio="TSP.GrowthWeighted.STRNWeighted.sexratio.mean")
plotR(fitSTRN, curve ="ML", ylim=c(0,2))
plotR(fitSTRN)
out <- info.nests(NestsResult=resultNest_4p_SSM,
                  SexualisationTRN=fitSTRN,
                  SexualisationTRN.CI="Hessian",
                  embryo.stages="Caretta caretta.SCL",
                  GTRN.CI="Hessian",
                  tsd=Med_Cc,
                  tsd.CI="Hessian",
                  replicate.CI=100,
                  progressbar=TRUE,
                  warnings=TRUE,
                  out="summary")$summary
```

```
# CTE with growth-weighted temperature average
plot(Med_Cc, xlim=c(25, 35))
points(x=out[, "TSP.GrowthWeighted.STRNWeighted.temperature.mean"], y=males/sexed,
         col="red", pch=19)
legend("topright", legend=c("CTE with growth-weighted and Sexualization TRN"),
         pch=19, col=c("red"))

#  Fit the beginning and end of TSP

Initial_STRN <- c('BeginTSP' = invlogit(0.33),
                  'EndTSP' = invlogit(0.66))
fp <- NULL
fitSTRN <- STRN(Initial_STRN=Initial_STRN,
                EmbryoGrowthTRN=resultNest_4p_SSM,
                tsd=Med_Cc,
                embryo.stages="fitted",
                Sexed=sexed, Males=males,
                fixed.parameters=fp,
              sexratio="TSP.TimeWeighted.GrowthRateWeighted.STRNWeighted.sexratio.mean")
invlogit(fitSTRN$par)
invlogit(fitSTRN$par-2*fitSTRN$SE)
invlogit(fitSTRN$par+2*fitSTRN$SE)

Initial_STRN <- c('dbeta_mu' = logit(0.5),
                  'dbeta_v' = 1/12)
fp <- NULL
fitSTRN <- STRN(Initial_STRN=Initial_STRN,
                EmbryoGrowthTRN=resultNest_4p_SSM,
                tsd=Med_Cc,
                embryo.stages="Caretta caretta.SCL",
                Sexed=sexed, Males=males,
                fixed.parameters=fp,
              sexratio="TSP.TimeWeighted.GrowthRateWeighted.STRNWeighted.sexratio.mean")
                mu <- invlogit(fitSTRN$par["dbeta_mu"]),
                v <- abs(fitSTRN$par["dbeta_v"])
                shape1 <- mu * (((mu * (1 - mu))/v) - 1)
                shape2 <- shape1 * (1 - mu)/mu
plot(seq(from=0, to=1, length.out=100),
     dbeta(seq(from=0, to=1, length.out=100),
               shape1=shape1, shape2=shape2),
     type="l", xlab="Progress of TSP",
     ylab="Force of sexualisation", bty="n", ylim=c(0, 6), las=1)

 Initial_STRN <- c('dbeta_mu' = 7.2194053298953236,
                   'dbeta_v' = 0.00050390986089928467)
fp <- NULL
fitSTRN <- STRN(Initial_STRN=Initial_STRN                                              ,
               EmbryoGrowthTRN=resultNest_4p_SSM                                       ,
               tsd=Med_Cc                                                              ,
               embryo.stages="Caretta caretta.SCL"                                     ,
               Sexed=sexed                                                             ,
               Males=males                                                             ,
               fixed.parameters=fp                                                     ,
```

```
             sexratio="TSP.TimeWeighted.GrowthRateWeighted.STRNWeighted.sexratio.mean" )

                mu <- invlogit(fitSTRN$par["dbeta_mu"]),
                v <- abs(fitSTRN$par["dbeta_v"])
                shape1 <- mu * (((mu * (1 - mu))/v) - 1)
                shape2 <- shape1 * (1 - mu)/mu

plot(seq(from=0, to=1, length.out=100),
     dbeta(seq(from=0, to=1, length.out=100),
               shape1=shape1, shape2=shape2),
     type="l", xlab="Progress of TSP",
     ylab="Force of sexualisation", bty="n", ylim=c(0, 0.04), las=1)

Initial_STRN <- c('dbeta_mu' = logit(0.5),
                  'dbeta_v' = 1/12)
L <- STRN(Initial_STRN=NULL                                                        ,
          fixed.parameters=Initial_STRN                                            ,
          EmbryoGrowthTRN=resultNest_4p_SSM                                        ,
          tsd=Med_Cc                                                               ,
        embryo.stages="Caretta caretta.SCL"                                            ,
          Sexed=sexed                                                              ,
          Males=males                                                              ,
          sexratio="TSP.TimeWeighted.GrowthRateWeighted.STRNWeighted.sexratio.mean")
Initial_STRN <- c('dbeta_mu' = logit(0.6),
                  'dbeta_v' = 1/12)
L <- STRN(Initial_STRN=NULL                                                        ,
          fixed.parameters=Initial_STRN                                            ,
          EmbryoGrowthTRN=resultNest_4p_SSM                                        ,
          tsd=Med_Cc                                                               ,
        embryo.stages="Caretta caretta.SCL"                                            ,
          Sexed=sexed                                                              ,
          Males=males                                                              ,
          sexratio="TSP.TimeWeighted.GrowthRateWeighted.STRNWeighted.sexratio.mean")
 Initial_STRN <- c('dbeta_mu' = 7.2192972077000004,
                    'dbeta_v' = 0.00050396969999999997)
L <- STRN(Initial_STRN=NULL                                                        ,
          fixed.parameters=Initial_STRN                                            ,
          EmbryoGrowthTRN=resultNest_4p_SSM                                        ,
          tsd=Med_Cc                                                               ,
          embryo.stages="Caretta caretta.SCL"                                      ,
          Sexed=sexed                                                              ,
          Males=males                                                              ,
          sexratio="TSP.TimeWeighted.GrowthRateWeighted.STRNWeighted.sexratio.mean")
                mu <- invlogit(fitSTRN$par["dbeta_mu"]),
                v <- abs(fitSTRN$par["dbeta_v"])
                shape1 <- mu * (((mu * (1 - mu))/v) - 1)
                shape2 <- shape1 * (1 - mu)/mu

   tsp_progress <- seq(from=0, to=1, length.out=100)
  plot(tsp_progress,
     dbeta(tsp_progress,
           shape1=shape1, shape2=shape2),
       type="l", xlab="Progress of TSP",
```

```
      ylab="Force of sexualisation", bty="n", ylim=c(0, 0.04), las=1)
 segments(x0=0, x1=1, y0=0, y1=0, lty=2)

## 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 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.
If fill is NA, it will use the stored fill value in result.

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),
  adaptive = FALSE,
  adaptive.lag = 500,
  adaptive.fun = function(x) {
     ifelse(x > 0.234, 1.3, 0.7)
 },
  parallel = TRUE,
  WAIC = TRUE,
  intermediate = NULL,
```

```
  filename = "intermediate.Rdata",
  previous = NULL,
  fill = NA
)
```

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
`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
`WAIC`	Prepare the output for loo() and waic().
`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.
`fill`	Parameters to be sent to STRN().

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 <`marc.girondot@gmail.com`>

Examples

```
## Not run:
library(embryogrowth)
MedIncubation_Cc <- subset(DatabaseTSD, Species=="Caretta caretta" &
RMU=="Mediterranean" & Sexed!=0)
Med_Cc <- tsd(males=MedIncubation_Cc$Males,
             females=MedIncubation_Cc$Females,
             temperatures=MedIncubation_Cc$Incubation.temperature,
             par=c(P=29.5, S=-0.1))
plot(Med_Cc, xlim=c(25, 35))
males <- c(7, 0, 0, 0, 0, 5, 6, 3, 5, 3, 2, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0)
names(males) <- rev(rev(names(resultNest_4p_SSM$data))[-(1:2)])
sexed <- rep(10, length(males))
names(sexed) <- rev(rev(names(resultNest_4p_SSM$data))[-(1:2)])
Initial_STRN <- resultNest_4p_SSM$par[c("DHA", "DHH", "T12H")]
Initial_STRN <- structure(c(582.567096666926, 2194.0806711639, 3475.28414940385),
                          .Names = c("DHA", "DHH", "T12H"))
fp <- c(Rho25=100)
fitSTRN <- STRN(Initial_STRN=Initial_STRN,
                EmbryoGrowthTRN=resultNest_4p_SSM,
                tsd=Med_Cc,
                embryo.stages="Caretta caretta.SCL",
                Sexed=sexed, Males=males,
                fixed.parameters=fp,
                sexratio="TSP.GrowthWeighted.STRNWeighted.sexratio")

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 = 10000, parametersMCMC = pMCMC,
                n.chains = 1, n.adapt = 0, thin = 1, trace = TRUE,
                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))


## End(Not run)
```

`summary.Nests2` *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 'Nests2'
summary(object, ...)
```

Arguments

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

Details

summary.Nests2 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_SSM)
# Get a set of parameters without transition
x1 <- resultNest_6p_SSM$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@universite-paris-saclay.fr>`

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, integral=integral.Gompertz, M0=0.3470893,
hatchling.metric=c(Mean=39.33, SD=1.92))

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

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

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

plot(resultNest_4p_SSM, temperatures=tempConst_f,
stop.at.hatchling.metric=TRUE, series="T25", xlim=c(0,120),
ylimT=c(22, 32), hatchling.metric=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_SSM)
test.parallel(resultNest_4p_SSM)

## 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"
##############################################################################
pfixed <- c(rK=1.208968)
M0 = 0.3470893
##############################################################################
# 4 parameters
##############################################################################
x <- structure(c(105.966881676793, 613.944134764125, 306.449533440186,
                118.193882815108), .Names = c("DHA", "DHH", "T12H", "Rho25"))
resultNest_4p_SSM <- searchR(parameters=x, fixed.parameters=pfixed,
temperatures=formated, integral=integral.Gompertz, M0=M0,
hatchling.metric=c(Mean=39.33, SD=1.92))
data(resultNest_4p_SSM)
plot(resultNest_4p_SSM, xlim=c(0,70), ylimT=c(22, 32), ylimS=c(0,45), series=1,
embryo.stages="Caretta caretta.SCL")
##############################################################################
pMCMC <- TRN_MHmcmc_p(resultNest_4p_SSM, 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

$$TRT_l = abs\,(S\,K_l)$$

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\ The default model is named logistic. This model, as well as the logit one, have the particularity to have a symmetric shape around P.
The *logistic* model is:

$$SR(T) = 1/(1 + exp((1/S) * (P - T))))$$

The *logit* model is:

$$SR(T) = 1/(1 + exp(4 * S * (P - T))))$$

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.
If asymmetric model is selected, it is always better to use *flexit* model.

$$if\ \ T < P\ \ then\ \ (1 + (2^{K_1} - 1) * exp(4 * S_1 * (P - T)))^{(-1/K_1)}$$

$$if\ \ T > P\ \ then\ \ 1 - ((1 + (2^{K_2} - 1) * exp(4 * S_2 * (T - P)))^{(-1/K_2)}$$

with:

$$S_1 = S/((4/K_1) * (2^{(-K_1)})^{(1/K_1+1)} * (2^{K_1} - 1))$$

$$S_2 = S/((4/K_2) * (2^{(-K_2)})^{(1/K_2+1)} * (2^{K_2} - 1))$$

The *flexit** model is defined as (QBT is the Quasi-Binary Threshold):

$$QBT = 1/(1 + exp(100 * (P - T)))$$

$$SR(T) = 1/(1 + exp(4 * (SL * QBT + SH * (1 - QBT)) * (P - T))))$$

The flexit** model has special interest because the parameter

$$SL + SH$$

are directly the TRT and then the unit of

$$SL + SH$$

are the same as unit of

$$P$$

.

$$SR(T) = 1/(1 + exp((log((1 - l)/l))/(SL * QBT + SH * (1 - QBT)) * (P - T))))$$

The *Stairs* model uses *TRTL*, *TRTH*, and *SRTRT*. The sex ratio is 1 for temperatures below *TRTL* and 0 for temperatures above *TRTH*. It is *logit(SRTRT)* between *TRTL* and *TRTH*.

Usage

```
tsd(
  df = NULL,
  males = NULL,
  females = NULL,
  N = NULL,
  temperatures = NULL,
  durations = NULL,
  l = 0.05,
  parameters.initial = c(P = 30, S = -2, K = 0, K1 = 1, K2 = 0, SL = -1, SH = -1),
  males.freq = TRUE,
  fixed.parameters = NULL,
  equation = "logistic",
  replicate.CI = 10000,
  range.CI = 0.95,
  SE = TRUE,
  replicate.NullDeviance = 1000,
  control = list(maxit = 1000),
  print = TRUE,
  method = "BFGS"
)
```

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`	Can be "logistic", "Hill", "A-logistic", "Hulin", "Double-A-logistic", "flexit", "flexit*", "flexit**", "GSD", "logit", "probit"
`replicate.CI`	Number of replicates to estimate confidence intervals
`range.CI`	The range of confidence interval for estimation, default=0.95
`SE`	If FALSE, does not estimate SE of parameters. Can be use when something wrong happens.
`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
`method`	method used for optim. Can be "BFGS", the most rapid or "Nelder-Mead" for special cases using n parameter.

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 <`marc.girondot@gmail.com`>

References

Girondot M (1999). “Statistical description of temperature-dependent sex determination using maximum likelihood.” *Evolutionary Ecology Research*, **1**(3), 479-486.
Godfrey MH, Delmas V, Girondot M (2003). “Assessment of patterns of temperature-dependent sex determination using maximum likelihood model selection.” *Ecoscience*, **10**(3), 265-272.
Abreu-Grobois FA, Morales-Mérida BA, Hart CE, Guillon J, Godfrey MH, Navarro E, Girondot M (2020). “Recent advances on the estimation of the thermal reaction norm for sex ratios.” *PeerJ*, **8**, e8451. doi:10.7717/peerj.8451, `https://peerj.com/articles/8451/`.
Hulin V, Delmas V, Girondot M, Godfrey MH, Guillon J (2009). “Temperature-dependent sex determination and global change: Are some species at greater risk?” *Oecologia*, **160**(3), 493-506.

See Also

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

Examples

```
## Not run:
library(embryogrowth)
CC_AtlanticSW <- subset(DatabaseTSD, RMU.2010=="Atlantic, SW" &
                          Species=="Caretta caretta" & (!is.na(Sexed) & Sexed!=0))
tsdL <- with (CC_AtlanticSW, tsd(males=Males, females=Females,
                                 temperatures=Incubation.temperature.set,
                                 equation="logistic", replicate.CI=NULL))
tsdH <- with (CC_AtlanticSW, tsd(males=Males, females=Females,
                                 temperatures=Incubation.temperature.set,
                                 equation="Hill", replicate.CI=NULL))
tsdR <- with (CC_AtlanticSW, tsd(males=Males, females=Females,
                                 temperatures=Incubation.temperature.set,
                                 equation="A-logistic", replicate.CI=NULL))
tsdF <- with (CC_AtlanticSW, tsd(males=Males, females=Females,
                                 temperatures=Incubation.temperature.set,
                                 equation="Flexit", replicate.CI=NULL))
tsdF1 <- with (CC_AtlanticSW, tsd(males=Males, females=Females,
                                 temperatures=Incubation.temperature.set,
                                 equation="Flexit*", replicate.CI=NULL))
tsdF2 <- with (CC_AtlanticSW, tsd(males=Males, females=Females,
                                 temperatures=Incubation.temperature.set,
                                 equation="Flexit**", replicate.CI=NULL))
tsdDR <- with (CC_AtlanticSW, tsd(males=Males, females=Females,
                                 temperatures=Incubation.temperature.set,
                                 equation="Double-A-logistic", replicate.CI=NULL))
gsd <- with (CC_AtlanticSW, tsd(males=Males, females=Females,
                                 temperatures=Incubation.temperature.set,
                                 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)

##############

tsdF2 <- with (CC_AtlanticSW, tsd(males=Males, females=Females,
                                 temperatures=Incubation.temperature.set,
                                 parameters.initial=c(P=30, SL=-2, SH=-2),
                                 equation="Flexit**", replicate.CI=NULL))
plot(tsdF2)

priors <- tsd_MHmcmc_p(
result = tsdF2,
default = "dunif",
accept = TRUE
)
priors["SL", "Prior2"] <- priors["SL", "Max"] <- -0.1
priors["SH", "Prior2"] <- priors["SH", "Max"] <- -0.1
```

```
out_mcmc <- tsd_MHmcmc(result=tsdF2, n.iter = 10000, parametersMCMC = priors)
plot(out_mcmc, parameters="SL")
plot(out_mcmc, parameters="SH")
plot(out_mcmc, parameters="P")

plot(tsdF2, resultmcmc=out_mcmc, use.ggplot = FALSE)

tsdF2p <- with (CC_AtlanticSW, tsd(males=Males, females=Females,
                                 temperatures=Incubation.temperature.set,
                                 parameters.initial=c(P=30, S=-2),
                                 equation="Flexit**", replicate.CI=NULL))
tsdL <- with (CC_AtlanticSW, tsd(males=Males, females=Females,
                                 temperatures=Incubation.temperature.set,
                                 equation="logistic", replicate.CI=NULL))


##############
eo <- subset(DatabaseTSD, Species=="Emys orbicularis", c("Males", "Females",
                                       "Incubation.temperature"))

eo_Hill <- with(eo, tsd(males=Males, females=Females,
                                       temperatures=Incubation.temperature.set,
                                       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.set,
                                 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.set,
                                 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.set,
                                  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.set,
                                  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.2010=="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.2010"))
tsd(subset(cm, subset=RMU.2010=="Pacific, SW"))
tsd(subset(cm, subset=RMU.2010=="Pacific, Northwest"))
tsd(subset(cm, subset=RMU.2010=="Atlantic, S Caribbean"))

### Eretmochelys imbricata
Ei_PacificSW <- subset(DatabaseTSD, RMU.2010=="Pacific, SW" &
                       Species=="Eretmochelys imbricata")
Ei_AtlanticW <- subset(DatabaseTSD, RMU.2010=="Atlantic, W (Caribbean and E USA)" &
                       Species=="Eretmochelys imbricata")
Ei_AtlanticSW <- subset(DatabaseTSD, RMU.2010=="Atlantic, SW" &
                       Species=="Eretmochelys imbricata")
Ei_PacSW <- tsd(Ei_PacificSW)
Ei_AtlW <- tsd(Ei_AtlanticW)
Ei_AtlSW <- tsd(Ei_AtlanticSW)
```

```
plot(Ei_PacSW, xlim=c(27, 33), show.PTRT = FALSE, main=expression(italic("Eretmochelys imbricata")))
par(new=TRUE)
plot(Ei_AtlW, xlim=c(27, 33), col="red", xlab="", ylab="",
     axes=FALSE, xaxt="n", show.PTRT = FALSE, errbar.col="red")
par(new=TRUE)
plot(Ei_AtlSW, xlim=c(27, 33), col="blue", xlab="", ylab="", axes=FALSE,
     xaxt="n", show.PTRT = FALSE, errbar.col="blue")
legend("topright", legend=c("Pacific, SW", "Atlantic, W", "Atlantic, SW"), lty=1,
col=c("black", "red", "blue"))

### Chelonia mydas
Cm_PacificSW <- subset(DatabaseTSD, RMU.2010=="Pacific, SW" & !is.na(Sexed) &
                       Species=="Chelonia mydas")
Cm_PacificNW <- subset(DatabaseTSD, RMU.2010=="Pacific, NW" &  !is.na(Sexed) &
                       Species=="Chelonia mydas")
Cm_AtlanticSC <- subset(DatabaseTSD, RMU.2010=="Atlantic, S Caribbean" &  !is.na(Sexed) &
                       Species=="Chelonia mydas")
Cm_IndianSE <- subset(DatabaseTSD, RMU.2010=="Indian, SE" &  !is.na(Sexed) &
                       Species=="Chelonia mydas")
Cm_PacSW <- tsd(Cm_PacificSW)
Cm_PacNW <- tsd(Cm_PacificNW)
Cm_IndSE <- tsd(Cm_IndianSE)
Cm_AtlSC <- tsd(Cm_AtlanticSC)

plot(Cm_PacSW, xlim=c(24, 34), show.PTRT = FALSE, main=expression(italic("Chelonia mydas")))
par(new=TRUE)
plot(Cm_PacNW, xlim=c(24, 34), col="red", xlab="", ylab="",
     axes=FALSE, xaxt="n", show.PTRT = FALSE, errbar.col="red")
par(new=TRUE)
plot(Cm_IndSE, xlim=c(24, 34), col="blue", xlab="", ylab="",
     axes=FALSE, xaxt="n", show.PTRT = FALSE, errbar.col="blue")
par(new=TRUE)
plot(Cm_AtlSC, xlim=c(24, 34), col="green", xlab="", ylab="",
     axes=FALSE, xaxt="n", show.PTRT = FALSE, errbar.col="green")

# To fit a TSDII or FMF TSD pattern, you must indicate P_low, S_low, P_high, and S_high
# for logistic model and P_low, S_low, K1_low, K2_low, P_high, S_high, K1_high, and K2_high for
# flexit model
# The model must be 0-1 for low and 1-0 for high with P_low < P_high

Chelydra_serpentina <- subset(DatabaseTSD, !is.na(Sexed) & (Sexed != 0) &
                       Species=="Chelydra serpentina")

model_TSDII <- tsd(Chelydra_serpentina, males.freq=FALSE,
                   parameters.initial=c(P_low=21, S_low=0.3, P_high=28, S_high=-0.4),
                   equation="logistic")
plot(model_TSDII, lab.TRT = "TRT l = 5 %")
priors <- tsd_MHmcmc_p(result=model_TSDII, accept=TRUE)
out_mcmc <- tsd_MHmcmc(result=model_TSDII, n.iter=10000, parametersMCMC=priors)
plot(model_TSDII, resultmcmc=out_mcmc, lab.TRT = "TRT l = 5 %")
predict(model_TSDII, temperatures=25:35)
```

```
# Podocnemis expansa
Podocnemis_expansa <- subset(DatabaseTSD, !is.na(Sexed) & (Sexed != 0) &
                        Species=="Podocnemis expansa")
Podocnemis_expansa_Valenzuela_2001 <- subset(Podocnemis_expansa,
                   Reference=="Valenzuela, 2001")
PeL2001 <- tsd(df=Podocnemis_expansa_Valenzuela_2001)
# The pivotal temperature is 32.133 °C (CI 95% 31.495;32.766)
# In Valenzuela, 2001: "Using data from the present study alone,
# the critical temperature was 32.2 °C by both methods and the 95%
# confidence limits were 31.4 °C and 32.9 °C."
# Data are close but not identical to what was published.

# The pivotal temperature calculated by maximum likelihood and by inverse
# prediction from logistic regression, was 32.6°C using raw data from
# 1991 (N. Valenzuela, unpublished data) and from this study. The lower
# and upper 95% confidence limits of the pivotal temperature were 32.2°C
# and 33.2°C,

Podocnemis_expansa_Valenzuela_1997 <- subset(Podocnemis_expansa,
                   subset=(((Reference=="Lance et al., 1992; Valenzuela et al., 1997") |
                   (Reference=="Valenzuela, 2001")) &
                   (!is.na(Sexed)) & (Sexed != 0)))

PeL1997 <- tsd(df=Podocnemis_expansa_Valenzuela_1997)

# Gekko japonicus

Gekko_japonicus <- subset(DatabaseTSD, !is.na(Sexed) & (Sexed != 0) &
Species=="Gekko japonicus")
model_TSDII_gj <- tsd(Gekko_japonicus, males.freq=TRUE,
                   parameters.initial=c(P_low=26, S_low=1.5,
                                        P_high=31, S_high=-1.5),
                   equation="logistic")
plot(model_TSDII_gj, lab.TRT = "TRT l = 5 %")
print(model_TSDII_gj)
prior <- tsd_MHmcmc_p(result = model_TSDII_gj, accept = TRUE)
prior <- structure(list(
  Density = c("dnorm", "dnorm", "dnorm", "dnorm"),
  Prior1 = c(26, 0.3, 31, -0.4),
  Prior2 = c(2, 1, 2, 1),
  SDProp = c(2, 0.5, 2, 0.5),
  Min = c(25, -2, 25, -2),
  Max = c(35, 2, 35, 2),
  Init = c(26, 0.3, 31, -0.4)),
  row.names = c("P_low",  "S_low", "P_high", "S_high"),
  class = "data.frame")

result_mcmc_tsd_gj <- tsd_MHmcmc(result=model_TSDII_gj,
parametersMCMC=prior, n.iter=10000, n.chains = 1,
n.adapt = 0, thin=1, trace=FALSE, adaptive=TRUE)
summary(result_mcmc_tsd_gj)
plot(result_mcmc_tsd_gj, parameters="P_low", scale.prior=TRUE, xlim=c(20, 30), las=1)
plot(result_mcmc_tsd_gj, parameters="P_high", scale.prior=TRUE, xlim=c(25, 35), las=1)
```

```
plot(model_TSDII_gj, resultmcmc = result_mcmc_tsd_gj)

# Trachemys scripta elegans
# Take care, the pattern reflects large population variation

Tse <- subset(DatabaseTSD, Species=="Trachemys scripta" & Subspecies == "elegans" & !is.na(Sexed))
Tse_logistic <- tsd(Tse)
plot(Tse_flexit)
compare_AICc(logistic=Tse_logistic, flexit=Tse_flexit)
plot(Tse_flexit)

# Exemple when only proportion is known; experimental
Ei_PacificSW <- subset(DatabaseTSD, RMU.2010=="Pacific, SW" &
                       Species=="Eretmochelys imbricata")
males <- Ei_PacificSW$Males/(Ei_PacificSW$Males+Ei_PacificSW$Females)*100
females <- 100-(Ei_PacificSW$Males/(Ei_PacificSW$Males+Ei_PacificSW$Females)*100)
temperatures <- Ei_PacificSW$Incubation.temperature
Ei_PacSW <- tsd(Ei_PacificSW)
par <- c(Ei_PacSW$par, n=10)
embryogrowth:::.tsd_fit(par=par, males=males, N=males+females, temperatures=temperatures,
                        equation="logistic")
Ei_PacSW_NormalApproximation <- tsd(males=males, females=females,
                                    temperatures=temperatures,
                                    parameters.initial=par)
Ei_PacSW_NormalApproximation$par
Ei_PacSW$par
# The data looks like only n=0.01 observations were done
# This is the reason of the large observed heterogeneity
plot(Ei_PacSW_NormalApproximation)

# Example of Flexit** model
temperatures <- seq(from=20, to=35, by=0.1)
l <- 0.05
SL <- 1
SH <- 2
P <- 29
# QBT is a threshold function
# It can be a piecewise function
QBT <- ifelse(temperatures < P, 1, 0)
# Or threshold using sign
QBT <- (sign( P - temperatures) + 1) / 2
# Or a logistic function
QBT <- (1+ exp(100*(temperatures - P)))^-1
# The advantage of logistic threshold is that the resulting function can be derivated
SR <- 1/(
        1+exp(
           ((-log((1-l)/l))/(SL*QBT+SH*(1-QBT)))*((P-temperatures))
             )
         )
# The pivotal temperature: P
temperatures[which.min(abs(SR - 0.5))]
# The TRT: SH + SL
temperatures[which.min(abs(SR - l))] - temperatures[which.min(abs(SR - (1 -l)))]
```

```
## 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 <`marc.girondot@gmail.com`>

See Also

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

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"),
  default = "dnorm",
  accept = TRUE
)
```

Arguments

`result`	An object obtained after a tsd fit
`default`	The default distribution for priors; can be dnorm only at that time
`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`, `DatabaseTSD.version()`, `P_TRT()`, `ROSIE`, `ROSIE.version()`, `TSP.list`, `plot.tsd()`, `predict.tsd()`, `stages`, `tsd()`, `tsd_MHmcmc()`

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@universite-paris-saclay.fr>`

References

Mrosovsky N, Pieau C (1991). "Transitional range of temperature, pivotal temperatures and thermosensitive stages for sex determination in reptiles." *Amphibia-Reptilia*, **12**(2), 169-179.
Monsinjon J, Guillon J, Wyneken J, Girondot M (2022). "Thermal reaction norm for sexualization: the missing link between temperature and sex ratio for temperature-dependent sex determination." *Ecological Modelling*, **473**(110119), 1-7. doi:10.1016/j.ecolmodel.2022.110119.
Girondot M, Monsinjon J, Guillon J (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. doi:10.1016/j.jtherbio.2018.02.006.

See Also

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

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% range where the true average temperature should be.

Author(s)

Marc Girondot

References

Girondot M, Godfrey MH, Guillon J, Sifuentes-Romero I (2018). "Understanding and integrating resolution, accuracy and sampling rates of temperature data loggers used in biological and ecological studies." *Engineering Technology Open Access Journal*, **2**(4), 55591.

See Also

Other Data loggers utilities: `calibrate.datalogger()`, `movement()`

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)
```

`UpdateNests` *Create a dataset of class Nests2 from an object of class Nests*

Description

Will create a dataset of class Nests2 to be used with searchR
This function is used to convert Nests or Nests2 format into the new one, Nests2, and add information.

Usage

```
UpdateNests(
  data = stop("An object with nests must be provided !"),
  weight = NULL,
  LayingTime = NULL,
  UnitTime = NULL,
  Longitude = NULL,
  Latitude = NULL,
  Informations = NULL,
  Males = NULL,
  Females = NULL,
  hatchling.metric.mean = NULL,
  hatchling.metric.sd = NULL
)
```

Arguments

`data`	An object of class Nests or Nests2.
`weight`	The weight of different nests for likelihood estimation.
`LayingTime`	Named POSIXct or POSIXlt time for each nest in data.
`UnitTime`	The units for time as a named list or vector
`Longitude`	The longitude of the nests as a named list or vector
`Latitude`	The latitude of the nests as a named list or vector
`Informations`	Some textual information about the nests as a named list or vector
`Males`	Number of sexed eggs being males.
`Females`	Number of sexed eggs being females.
`hatchling.metric.mean`	The average size of hatchlings.
`hatchling.metric.sd`	The standard deviation of size of hatchlings.

Details

UpdateNests creates a dataset of class "Nests2" to be used with searchR

Value

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

Author(s)

Marc Girondot <`marc.girondot@gmail.com`>

Examples

```
## Not run:
library(embryogrowth)
data(nest)
nest <- FormatNests(nest)
nest2 <- UpdateNests(data=nest)
nest2 <- UpdateNests(data=nest2, Males=c(DY.1=10), Females=c(DY.1=20))

## 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(temperatures=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,
integral=integral.Gompertz, M0=1.7, hatchling.metric=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