Package 'BaSTA'

October 12, 2022

Type Package

Title Age-Specific Survival Analysis from Incomplete Capture-Recapture/Recovery Data

Version 1.9.5

Date 2022-10-05

Author Fernando Colchero, Owen Jones, Maren Rebke

Maintainer Fernando Colchero <colchero@imada.sdu.dk>

Depends R (>= 2.10)

Imports snowfall

Description Estimates survival and mortality with covariates from capture-recapture/recovery data in a Bayesian framework when many individuals are of unknown age. It includes tools for data checking, model diagnostics and outputs such as life-tables and plots.

License GPL

LazyLoad yes

LazyData yes

NeedsCompilation no

Repository CRAN

Date/Publication 2022-10-05 12:40:05 UTC

R topics documented:

`BaSTA-package` *BaSTA: a package for parametric Bayesian estimation of age-specific survival for truncated and censored capture-recapture/recovery records.*

Description

This package estimates survival trajectories with covariates from capture-recapture/recovery data in a Bayesian framework when many individuals are of unknown age. It includes tools for data checking, model diagnostics, and outputs such as life-tables and plots.

Details

Package:	BaSTA
Type:	Package
Version:	1.9.5
Date:	2022-10-05
License:	GNU General Public Licence
LazyLoad:	yes

This package estimates survival parameters from capture-recapture data where there are individuals with unknown birth/death dates.

Author(s)

Fernando Colchero <colchero@imada.sdu.dk>, Owen Jones <jones@biology.sdu.dk>, and Maren Rebke <maren.rebke@avitec-research.de>

References

Colchero, F. and J.S. Clark (2012) Bayesian inference on age-specific survival from capture-recapture data for censored and truncated data. *Journal of Animal Ecology*. 81, 139-149.

Colchero, F., O.R Jones and M. Rebke. (2012) BaSTA: an R package for Bayesian estimation of age-specific survival from incomplete mark-recapture/recovery data with covariates. *Methods in Ecology and Evolution*. 3, 466-470.

`basta` *Parametric Bayesian estimation of age-specific survival for left-truncated and right-censored capture-recapture/recovery data.*

Description

This function performs multiple Markov Chain Monte Carlo (MCMC) simulations for the Bayesian estimation of age-specific mortality and survival trends when a large proportion of (or all) records have unknown times of birth and/or death. Survival parameters and unknown (i.e. latent) birth and death times are estimated, allowing the user to test a range of mortality patterns, and to test the effect of continuous and/or discrete covariates following Colchero and Clark's (2012) approach.

Usage

```
basta(object, ...)

## Default S3 method:
basta(object, studyStart, studyEnd, minAge = 0, model = "GO",
    shape = "simple", covarsStruct = "fused", niter = 11000,
    burnin = 1001, thinning = 20, recaptTrans = studyStart,
    nsim = 1, parallel = FALSE, ncpus = 2, lifeTable = TRUE,
    updateJumps = TRUE, ...)
```

Arguments

object	A data.frame to be used as an input data file for BaSTA. The first column is a vector of individual unique IDs, the second and third columns are birth and death years respectively. Columns $4, \ldots, n_t - 1$ represent the observation window of n_t years. This is followed (optionally) by columns for categorical and continuous covariates (see details).
studyStart	The first year of the study.
studyEnd	The last year of the study.
minAge	Age at which the analysis should start (see details)
model	The underlying mortality model to be used. "EX" = exponential,"GO" = Gompertz, "WE" = Weibull and "LO" = logistic (see details).
shape	The overall shape of the model. Values are: simple = no extra parameters added; Makeham = a constant parameter is added to the mortality; and bathtub = a Gompertz declining mortality for early ages and a constant parameter are added to the mortality model (see details).
covarsStruct	Character string that indicates how covariates should be evaluated. The options are: "fused", which defines all categorical variables as covariates for each mortality parameter and all continuous covariates under a proportional hazards structure; "prop.haz", which puts all covariates under a proportional hazards structure; and "all.in.mort" puts all covariates as a multilevel function of the mortality parameters (see details).
niter	The total number of MCMC steps.
burnin	The number of iterations for the burn in (see details).
thinning	The number of skipped MCMC steps to minimize serial autocorrelation (see details).

recaptTrans	A vector (of length npi) defining the recapture probability transition times (RPTP). These are points (years) where the recapture probability is thought to change. The default setting is for the recapture probability to be constant throughout the study, so the recaptTrans is simply defined as a single element vector of the first year of the observation period (e.g. c(1985)). If recapture probabilities were known to change at year say, 1990, the RPTP should be defined as c(1985, 1990).
nsim	A numerical value for the number of simulations to be run.
parallel	A logical argument indicating whether the multiple simulations should be run in parallel or not. If TRUE, package **snowfall** is called and multiple simulations are run in parallel. If **snowfall** is not installed, the model is ran in series.
ncpus	a numerical value that indicates the number of cpus to be used if parallel is TRUE and package **snowfall** is installed. The default is 2 cpus. If package pkgsnowfall is not installed, the simulations are run in series.
lifeTable	A logical argument indicating whether or not to produce life tables. If TRUE, a cohort life table is calculated using function MakeLifeTable.
updateJumps	A logical argument indicating wheter to update jump standard deviations (adaptive independent Metropolis) until an update rate of 0.25 is achieved (see details).
...	Additional arguments to be passed to function basta (see details)

Details

To construct the input data object the function CensusToCaptHist can be used to build the capture-recapture matrix, while the covariate (design) matrix can be constructed with the MakeCovMat function.

basta uses parametric mortality functions to estimate age-specific mortality (survival) from capture-recapture/recovery data. The mortality function describes how the risk of mortality changes with age, and is defined as $\mu(x|\theta)$, where x corresponds to age and θ is a vector of parameters to be estimated.

The model argument allows the user to choose between four basic mortality functions, namely (a) Exponential ("EX"; Cox and Oakes 1974), with constant mortality with age specified as

$$\mu_b(x|b) = b,$$

where $b > 0$, (b) the Gompertz mortality function ("GO"; Gompertz 1925, Pletcher 1999), calculated as

$$\mu_b(x|b) = exp(b_0 + b_1 x),$$

where $-\infty < b_0, b_1 < \infty$, (c) the Weibull mortality model ("WE"; Pinder III *et al.* 1978) calculated as

$$\mu_b(x|b) = b_0 b_1^{b_0} x^{b_0 - 1},$$

where $b_0, b_1 > 0$, and (d) the logistic mortality model ("LO"; Pletcher 1999), calculated as

$$\mu_b(x|b) = exp(b_0 + b_1 x)/(1 + b_2 exp(b_0)/b_1(exp(b_1 x) - 1)),$$

where $b_0, b_1, b_2 > 0$.

The shape argument allows the user to extend these models in order to explore more complex mortality shapes. The default value is "simple" which leaves the model as defined above. With value "Makeham", a constant is added to the mortality, making the model equal to $\mu_0(x|\theta) = \mu_b(x|b) + c$, where $\theta = [c, b]$. With value "bathtub", concave shapes in mortality can be explored. This is achieved by adding a declining Gompertz term and a constant parameter to the basic mortality model, namely

$$\mu_0(x|\theta) = exp(a_0 - a_1 x) + c + \mu_b(x|b)$$

,

where $-\infty < a_0 < \infty$, $a_1 \leq 0$ and $c \leq 0$.

To incorporate covariates into the inference process, the mortality model is further extended by including a proportional hazards structure, of the form

$$\mu(x|\theta, \Gamma, Z_a, Z_c) = \mu_0(x|\theta, Z_a) exp(\Gamma Z_c)$$

,

where $\mu_0(x|\theta, Z_a)$ represents the mortality section as defined above, while the second term $exp(\Gamma Z_c)$ corresponds to the proportional hazards function. Z_a and Z_c are covariate (design) matrices for categorical and continuous covariates, respectively.

When covariates are included in the dataset, the basta function provides three different ways in which these can be evaluated by using argument covarsStruct:

1. "fused" will make the mortality parameters linear functions of all categorical covariates (analogous to a generalised linear model (GLM) structure) and will put all continuous covariates under a proportional hazards structure. Thus, for a simple exponential model with constant mortality of the form $\mu_0(x|b) = b$, the parameter is equal to $b = b_0 + b_1 z_1 + \ldots, b_k z_k$, where $[b_0, \ldots, b_k]$ are paramters that link the mortality parameter b with the categorical covariates $[z_1, \ldots, z_k]$.

2. "prop.haz" will put all covariates under a proportional hazards structure irrespective of the type of variable.

3. "all.in.mort" will put all covariates as linear functions of the survival parameters as explained above. Since most models require the lower bounds for the mortality parameters to be equal to 0, the only model that can be used for this test is Gompertz with shape set to "simple". In case these arguments are specified deferently, a warning message is printed noting that model will be forced to be "GO" and shape will be set to "simple".

The burnin argument represents the number of steps at the begining of the MCMC run that is be discarded. This sequence commonly corresponds to the non-converged section of the MCMC sequence. Convergence and model selection measures are calculated from the remaining thinned parameter chains if multiple simulations are run, and all if all of them run to completion.

The thinning argument specifies the number of steps to be skipped in order to reduce serial autocorrelation. The thinned sequence, which only includes steps after burn in, is then used to calculate convergence statistics and model for selection.

The updateJumps argument specifies wether to run a simulation to find appropriate jump standard deviations for theta and gamma parameters. If argument "nsim" is set to 1, then the simulation runs with the update jumps routine active. If "nsim" is larger than 1, then an initial simulation is ran to find apropriate jumps before the main analysis is ran.

Additional arguments for priors, jumps and start values can be passed on the ... section. For instance, argument thetaStart can be specified as a vector defining the initial values for each parameter in the survival model. If this argument is not specified, a set of random parameters is generated for each simulation. Similarly, argument gammaStart can be specified for all parameters in the proportional hazards section of the model. Jump standard deviations (i.e. the standard error in the Metropolis step) can be specified with arguments thetaJumps and gammaJumps. As with thetaStart, default values are assigned if these arguments are not specified. Arguments thetaPriorMean, thetaPriorSd, gammaPriorMean and gammaPriorSd can be used to specify prior means and standard errors for each survival and proportional hazards parameters. Again, if not specified, default values are assigned.

The number of parameters in thetaStart, thetaJumps, thetaPriorMean and thetaPriorSd should be a vector or matrix for the parameters in the mortality function. The number of parameters will depend on the model chosen with model (see above). If the number of parameters specified does not match the number of parameters inherent to the model and shape selected, the function returns an error.

As described above, the number of parameters for gammaStart, gammaJumps, gammaPriorMean and gammaPriorSd arguments (i.e. section b), namely the proportional hazards section, will be a function of the number of continuous covariates if argument covarsStruct is "fused", or to the total number of covariates when covarsStruct is "prop.haz".

Another additional argument is returnAges, which outputs a value estAges, which is a matrix with all estimated ages after the burnin and thinning. This matrix also includes ages that were known, thus each row corresponds to the individual in the same row as the original dataset.

Value

coefficients — A matrix with estimated coefficients (i.e. mean values per parameter on the thinned sequences after burnin), which includes standard errors, upper and lower 95% credible intervals, update rates per parameter, serial autocorrelation on the thinned sequences and the potential scale reduction factor for convergence (see Convergence value below).

DIC — Basic deviance information criterion (DIC) calculations to be used for model selection (Spiegelhalter *et al.* 2002, Celeux *et al.* 2006). Small differences between values should only be used a reference (see comments in Spiegelhalter *et al.* 2002). If all or some of the simulations failed, then the returned value is "Not calculated".

convergence — If requested, a matrix with convergence coefficients based on potential scale reduction as described by Gelman *et al.* (2004). If only one simulation was ran, then the returned value is "Not calculated".

KullbackLeibler — If called by summary, list with Kullback-Leibler discrepancy matrices between pair of parameters for categorical covariates (McCulloch 1989, Burnham and Anderson 2001) and McCulloch's (1989) calibration measure. If only one sim-

ulation was ran or if no convergence is reached, then the returned value is "Not calculated".

`params` If requested, a matrix with the thinned, converged parameter traces of all runs. This matrix is used to calculate quantiles for parameters, survival probability and mortality (see below).

`settings` If called by `summary`, this is a vector indicating the number of iterations for each MCMC, the burn in sequence, the thinning interval, and the number of simulations that were run.

`modelSpecs` Model specifications inidicating the `model`, the `shape`, the covariate structure and the minimum age that were specified by the user.

`jumpPriors` If requested or called by functions `summary` or `summary.basta`, a matrix with the jump standard deviations and prior means and standard deviation for the parameters.

`birthQuant` If requested, summary matrix of estimated times of birth including median and upper and lower 95% predictive intervals.

`deathQuant` If requested, summary matrix of estimated times of birth including median and upper and lower 95% predictive intervals.

`agesQuant` If requested, summary matrix of estimated ages at death including median and upper and lower 95% predictive intervals.

`mortQuant` If requested or called by functions `plot` or `plot.basta` median and 95% predictive intervals for the estimated mortality rates.

`survQuant` If requested or called by functions `plot` or `plot.basta` median and 95% predictive intervals for the estimated survival probability.

`parsForPlot` If requested or called by functions `plot` or `plot.basta` thinned sequences, including burn-in, of parameter traces used to plot outputs.

`lifeTable` If requested and specified in the argument `lifeTable`, a cohort life table calculated from the estimated ages at death.

Author(s)

Fernando Colchero <colchero@imada.sdu.dk>, Owen R. Jones <jones@biology.sdu.dk>, and Maren Rebke <maren.rebke@avitec-research.de>

References

Burnham, K.P. and Anderson, D.R. (2001) Kullback-Leibler information as a basis for strong inference in ecological studies. *Widlife Research*, 28, 111-119.

Celeux, G., Forbes, F., Robert, C. P., and Titterington, D. M. (2006) Deviance information criteria for missing data models. *Bayesian Analysis*, 1(4), 651-673.

Colchero, F. and J.S. Clark (2012) Bayesian inference on age-specific survival from capture-recapture data for censored and truncated data. *Journal of Animal Ecology*. 81, 139-149.

Colchero, F., O.R. Jones and M. Rebke. (2012) BaSTA: an R package for Bayesian estimation of age-specific survival from incomplete mark-recapture/recovery data with covariates. *Method in Ecology and Evolution*. 3, 466-470.

Cox, D. R., and Oakes D. (1984) *Analysis of Survival Data*. Chapman and Hall, London.

Gelman, A., Carlin, J.B., Stern, H.S. and Rubin, D.B. (2004) *Bayesian data analysis*. 2nd edn. Chapman & Hall/CRC, Boca Raton, Florida, USA.

Gompertz, B. (1825) On the nature of the function expressive of the law of human mortality, and on a new mode of determining the value of life contingencies. *Philosophical Transactions of the Royal Society of London*, 115, 513-583.

King, R. and Brooks, S.P. (2002) Bayesian model discrimination for multiple strata capture-recapture data. *Biometrika*, 89, 785-806.

McCulloch, R.E. (1989) Local model influence. *Journal of the American Statistical Association*, 84, 473-478.

Pinder III, J.E., Wiener, J.G. and Smith, M.H. (1978) The Weibull distribution: a new method of summarizing survivorship data. *Ecology*, 59, 175-179.

Spiegelhalter, D.J., Best, N.G., Carlin, B.P. and van der Linde, A. (2002) Bayesian measures of model complexity and fit. *Journal of the Royal Statistical Society: Series B*, 64, 583-639.

See Also

`summary.basta`, `print.basta`, `plot.basta` to visualise summary outputs for objects of class "basta". `CensusToCaptHist` and `MakeCovMat` for raw data formatting.

Examples

```
## Load data:
data("sim1", package = "BaSTA")

## Check data consistency:
new.dat  <- DataCheck(sim1, studyStart = 51, studyEnd = 70, autofix = rep(1,7))

## Run short version of BaSTA on the data:
out <- basta(sim1, studyStart = 51, studyEnd = 70, niter = 100, burnin = 11,
             thinning = 10, updateJumps = FALSE)

## Print results:
summary(out, digits = 3)

## Plot traces for survival parameters:
plot(out)

## Plot traces for proportional hazards parameter:
plot(out, trace.name = "gamma")

## Plot survival and mortality curves:
plot(out, plot.trace = FALSE)
```

CensusToCaptHist *Constructs capture-history matrix from census matrix to be used in Bayesian Survival Trajectory Analysis (BaSTA).*

Description

This function takes a common census table, consisting of a pair of vectors with ID and observation date, and converts it into a capture-history matrix to be used as part of a BaSTA data input.

Usage

```
CensusToCaptHist(ID, d, dformat = "%Y", timeInt = "Y")
```

Arguments

ID	A vector with individual IDs.
d	A vector of dates when each individual was observed.
dformat	Defines the date format for d when d is of class character.
timeInt	A one character string specifying which time interval should be used between capture occasions. Arguments are "Y" for years, "M" for months, "W" for weeks and "D" for days.

Details

The d argument can be specified as an object of class POSIXct or POSIXlt, as a vector of integer time intervals or as a character string indicating the day, month and year. (e.g. dd/mm/yyyy, mmddyyyy, mm-dd-yyyy etc.). When d is of class character then argument dformat needs to be specified using the same conventions as in function format.POSIXct for objects of class POSIXct or POSIXlt.

Author(s)

Owen R. Jones <jones@demogr.mpg.de> and Maren Rebke <rebke@demogr.mpg.de>

See Also

MakeCovMat, which formats a covariate matrix compatible with this output.

Examples

```
id.vec <- sort(sample(1:5, size = 15, replace = TRUE))
d.vec <- rep(0, length(id.vec))
for(i in unique(id.vec)){
  svec <- which(id.vec == i)
  d.vec[svec] <- sort(sample(1990:1995, length(svec)))
}
Y <- CensusToCaptHist(ID = id.vec,  d = d.vec)
```

DataCheck *A function to check the input data file for a Bayesian Survival Trajectory Analysis (BaSTA) analysis.*

Description

This function performs some basic error checking on the input files for a BaSTA analysis. A number of checks are performed including; (1) individuals dying before the observation window starts; (2) individuals with no observations of any kind (i.e. phantom individuals); (3) individuals with birth date recorded as being after death date; (4) individuals with observations recorded after death; (5) individuals with observations before birth; (6) years of birth must appear as 0 in the observation matrix; (7) years of death must appear as 0 in the observation matrix.

Usage

```
DataCheck(object, studyStart, studyEnd, autofix = rep(0, 7),
       silent = TRUE)
```

Arguments

object	A data.frame to be used as an input data file for BaSTA. The first column is the individual's ID, the second and third columns are birth and death years respectively. Columns 4 to nt+3 represent the observation window of nt years. This is followed (optionally) by columns for covariate.
studyStart	The start year of the observation window.
studyEnd	The end year of the observation window.
autofix	A vector argument with a length of 7 indicating whether to automatically fix any errors (see details). This should be used with extreme caution. We recommend going back to the individual-based data and fixing each error "by hand".
silent	A logical argument indicating whether to print a detailed report to the screen or not.

Details

Argument autofix allows the user to fix the potential errors by specifying a code for each fix. Below are the descriptions of the actions that are taken depending on the error type and the fix code:

Type 1: 0 = do nothing; 1 = remove from dataframe.

Type 2: 0 = do nothing; 1 = remove from dataframe.

Type 3: 0 = do nothing; 1 = replace death records with 0; 2 = replace birth records with 0; 3 = replace both birth and death records with 0.

Type 4: 0 = do nothing; 1 = remove spurious post-death observations.

Type 5: 0 = do nothing; 1 = remove observations that pre-date year of birth.

Type 6: 0 = do nothing; 1 = replace birth year element of observation matrix with 0.

Type 7: 0 = do nothing; 1 = replace death year element of observation matrix with 0.

Value

ok	A logical indicator that indicates if the data are free of errors or not. i.e. TRUE = the data have no apparent errors, and FALSE = there is at leat one error.
newData	A corrected data frame.
type1	A vector of row numbers in the original data frame where there are deaths occurring before the study starts.
type2	A vector of row numbers in the original data frame where there are no birth/death AND no obervations.
type3	A vector of row numbers in the original data frame where there are births recorded after death.
type4	A vector of row numbers in the original data frame where there are observations (i.e. recaptures) after death.
type5	A vector of row numbers in the original data frame where there are observations (i.e. recaptures) before birth.
type6	A vector of row numbers in the original data frame where the year of birth is not a zero in the recapture matrix.
type7	A vector of row numbers in the original data frame where the year of death is not a zero in the recapture matrix.

Author(s)

Owen R. Jones <jones@biology.sdu.dk> and Fernando Colchero <colchero@imada.sdu.dk>

See Also

basta

MakeCovMat *Function to build a matrix of covariates (i.e. design matrix) for a Bayesian Survival Trajectory Analysis (BaSTA) analysis.*

Description

This function creates a matrix of covariates or design matrix appropriate for BaSTA from raw individual level covariate data. The function identifies categorical and continuous covariates and organizes them accordingly.

Usage

```
MakeCovMat(x, data)
```

Arguments

x	A character string vector or a numerical vector indicating the columns to be included or an object of class formula.
data	A data frame of n rows (n = number of individuals in dataset) and nz columns (number of general covariates) including categorical covariates as factors (e.g. sex with individual labels "f","m", location, etc.) and/or continuous individual covariates (e.g. weight at birth, general weather conditions at each location, etc.).

Details

The x argument can be of class character, numeric or formula as long as the elements described correspond to the column names in the data data frame. The data frame specified in argument data needs to explicitly differentiate between categorical and numerical variables. The elements in the column of a categorical variable must be coerced to be factors.

Value

The function returns a new covariate matrix to be collated to a matrix that includes a column for individual ID, a column for time of birth, and a column for times of death, plus the full recapture matrix.

Author(s)

Owen R. Jones <jones@demogr.mpg.de> and Fernando Colchero <colchero@demogr.mpg.de>

See Also

basta

Examples

```
## Simulated sex and weight data for 5 individuals:
sex        <- sample(c("f", "m"), 5, replace = TRUE)
weight     <- rnorm(5, mean = 10, sd = 1)
raw.mat    <- data.frame(sex, weight)
new.mat    <- MakeCovMat(~sex + weight, data = raw.mat)
```

MakeLifeTable *A function for calculating standard cohort-based life tables from age at death data.*

Description

This function calculates a standard cohort-based life table from death ages.

Usage

```
MakeLifeTable(DeathAges, ax = 0.5, n = 1)
```

Arguments

DeathAges	A vector of ages at death. These can be precise (e.g. 5.4 yrs) or rounded to the integer year (e.g. 5 yrs).
ax	A vector describing the average *proportion* of the interval lived by those dying in the interval. Note that this is slightly different from convention, which defines ax as the average length of time lived in the interval by those dying in the interval. This is so we can cope automatically with varying interval sizes. Further note that the ax in the life table produced by this code is expressed in units of time, rather than proportion of the interval. It is only necessary to put the first few values - further values are extrapolated to the correct dimensions. e.g. if ax is set as c(0.3, 0.5) and there are 5 rows in the life tables, ax will become c(0.3, 0.5, 0.5, 0.5, 0.5).
n	Defines the interval width. The default is 1 (yr), but it can be set to be any numerical value(s). For example, if you want small intervals for early ages, and larger intervals for older ages, you can define n as a sequence such as c(1,1,5) to indicate that the first 2 intervals are 1 year long, then the rest of the intervals are 5 years long.

Value

The function returns a data.frame in the form of a life table with the following column headings. Start and end of the interval, where start of the interval is defined as x, lx (number alive at age x (i.e. at the start of the interval)), nx the size of the interval, dx (number dying between ages x and $x+n$), mx (death rate in the cohort between ages x and $x+n$), ax (average time lived in the interval by those dying in the interval), qx (probability of dying between ages x and $x+n$), px (probability of surviving between ages x and $x+n$), Lx ('person' years lived between ages x and $x+n$), Tx ('person' years lived above age x), ex (life expectancy at age x).

Author(s)

Owen Jones <jones@biology.sdu.dk>

References

Preston, S.H., Heuveline, P. and Guillot, M. (2001) *Demography: Measuring and modeling population processes.* Blackwell Publishers, Oxford, UK.

See Also

basta

Examples

```
MakeLifeTable(rpois(100,3))
```

`multibasta` *Function to run multiple BaSTA models on the same dataset.*

Description

These functions allow users to run models with different functional forms on the same dataset, to perform model comparison and to visualize the results of the multiple runs.

Usage

```
multibasta(object, studyStart, studyEnd, models, shapes, ...)
## S3 method for class 'multibasta'
summary(object, ...)
## S3 method for class 'multibasta'
print(x, ...)
```

Arguments

`object`	For function `multibasta`, a `data.frame` to be used as an input data file for BaSTA. The first column is a vector of individual unique IDs, the second and third columns are birth and death years respectively. Columns 4-(nt-1) represent the observation window of nt years. This is followed (optionally) by columns for categorical and continuous covariates (see `details`). For function `summary`, a `multibasta` output object of class "multibasta."
`x`	a `multibasta` output object of class "multibasta."
`studyStart`	The first year of the study.
`studyEnd`	The last year of the study.
`models`	A character vector specifying the models to be tested (e.g. "GO", "WE", "LO")
`shapes`	A character vector specifying the shapes to be tested (e.g. "simple", "Makeham", "bathtub").
`...`	Additional arguments to be passed to function `basta` (see `details`)

Value

`runs`	A list with the `basta` outputs for the models and shapes tested.
`DICs`	A matrix with the DIC values for each model, sorted from the model with lowest DIC to the model with highest DIC.
`models`	A summary table showing the models and shaped tested.

Author(s)

Fernando Colchero <colchero@imada.sdu.dk>, Owen R. Jones <jones@biology.sdu.dk>, and Maren Rebke <maren.rebke@avitec-research.de>

See Also

basta, as well as CensusToCaptHist and MakeCovMat for raw data formatting.

Examples

```
## Load data:
data("sim1", package = "BaSTA")

## Run short version of BaSTA on the data:
multiout <- multibasta(sim1, studyStart = 51, studyEnd = 70,
                 models = c("GO", "WE"), shapes = "simple",
                 niter = 100, burnin = 11, thinning = 10,
                 nsim = 1, updateJumps = FALSE, parallel = FALSE)

## Print results:
summary(multiout, digits = 3)
```

---

sim1 *Simulated dataset to demonstrate Bayesian Survival Trajectory Analysis (BaSTA)*

Description

This dataset was created by stochastically simulating a hypothetical population with different mortality patterns between males and females and with proportional decreases in mortality as a function of a hypothetical continuous covariate (e.g. birth weight, average adult weight, etc.). The population was simulated for 100 years, at each one of which 100 individuals were born. The number of females per generation was randomly drawn from a binomial distribution using function rbinom with probability of 0.5 (i.e. 1:1 sex ratio). The individual continuous covariate was randomly drawn from a random normal distribution (with function rnorm) with mean parameter equal to 0 (e.g. anomaly of weights) and standard deviation equal to 1. The time of death for each individual was inversed sampled from a Gompertz CDF of ages at death. The Gompertz parameters for females were: $\alpha = -4$ and $\beta = 0.15$; and for males at $\alpha = -3$ and $\beta = 0.15$. The gamma parameter for the continuous covariate was $\gamma = 0.2$.

The study was assumed to start at year 51 and to finish at year 70. Recapture probability was set to 0.6 and thus each observation per individual was randomly drawn from a Bernoulli trial with parameter $p = 0.6$. Captures at birth and recoveries were randomly drawn from a Bernoulli trial with parameters $p = 0.5$ and $p = 0.2$, respectively.

Therefore, the resulting dataset includes individuals where the data are left-truncated and/or right-censored. This is typical of capture mark recovery datasets.

Usage

```
sim1
```

Format

RData file, data frame.

sim1Out — *Output from a Bayesian Survival Trajectory Analysis (BaSTA) analysis on a simulated dataset*

Description

This dataset is the output of a BaSTA analysis on the simulated dataset sim1. The analysis consisted of four independent simulations run in parallel. Each simulation was run for 20,000 iterations. The model chosen was Gompertz ("GO") with the shape argument set to "simple" and covarStruct set to "fused".

Usage

```
sim1Out
```

Format

RData file, list, basta.

summary.basta — *Summarizing and plotting Bayesian Survival Trajectory Analysis (BaSTA) model outputs.*

Description

These functions are all generic methods for class basta.

Usage

```
## S3 method for class 'basta'
summary(object, ...)
## S3 method for class 'basta'
print(x, ...)
## S3 method for class 'basta'
plot(x, plot.trace = TRUE, trace.name = "theta",
    fancy = FALSE,...)
```

Arguments

object	An object of class basta.
x	An object of class basta.
plot.trace	A logical argument. If TRUE the raw parameter traces are plotted, else, the predictive intervals for the resulting survival probability and mortality rates are plotted.

`trace.name` Character string indicating the set of parameters or posteriors to be plotted. The options are: "`theta`" to plot the survival model parameters; "`gamma`" to plot the proportional hazards parameters (if it applies, else `plot.basta` returns an error); "`pi`" to plot the recapture probabilities; and "post" to plot the conditional posteriors for the parameters and for the latent ages at death, and the full posterior for the model.

`fancy` A logical argument indicating to plot a combined plot of trace densities and estimated vital rates.

`...` Additional arguments passed to functions `print`, `summary` and `plot` (see details).

Details

For objects of class `basta`, the `print` function returns three summary elements describing the model and its results, namely: `call`, `run`, `coefficients` and, if convergence was reached, the `DIC` values for model fit. `call` describes the basic model used (i.e. exponential, Gompertz, Weibull or logistic), the shape chosen, "`simple`", "`Makeham`" or "`bathtub`", the covariate structure chosen, "`fused`", "`prop.haz`" or "`all.in.mort`" and which covariates (if any) were categorical and which continuous. Argument `digits` can be used for number formatting (see `summary`() or `signif`() for details).

The summary element `coefficients` prints out the estimated coefficients for all parameters in the model, as well as their standard errors and 95% upper and lower credible intervals. It also includes a measure of serial autocorrelation for each parameter calculated from the thinned parameter chains, an update rate per parameter, and the potential scale reduction factor for each parameter as a measure of convergence (Gelman *et al.* 2004).

Function `summary` includes all the previous elements, as well as a summary description of the priors and jump standard deviations for all survival parameters, a calibration of the Kullback-Leibler discrepancy as a measure of parameter similarities for those parameters associated to categorical covariates (McCulloch 1989), and a measure of model fit based on the deviance information criterion (DIC) (Spiegelhalter *et al.* 2002).

Function `plot` takes objects of class `basta` to create trace plots or, if the argument for `plot.trace` is set to `FALSE`, it plots estimated survival probabilities and mortality rates with their 95% predictive intervals. If argument `plot.trace` is set to `FALSE`, argument `xlim` can be used to define a range of ages to visualize survival and mortality trends. Also, if logical argument `noCI` is set to `TRUE`, credible intervals around survival and mortality are not plotted, leaving only the mean trends. This can be handy when several categorical covariates have been evaluated and the plots get too crowded.

Other arguments for `plot` include `names.legend` to indicate alternative names for the legend of vital rates plots. Also, when `plot.trace` is `FALSE`, argument `xlim` can be changed to display only a subest of the support. When argument `noCI` is `TRUE`, then the credible intervals around the vital rates are not displayed.

Value

Function `summary`() outputs the folowing values:

`coefficients` A matrix with estimated coefficients (i.e. mean values per parameter on the thinned sequences after burnin), which includes standard errors, upper and lower

95% credible intervals, update rates per parameter (commonly the same for all survival and proportional hazards parameters), serial autocorrelation on the thinned sequences and the potential scale reduction factor for convergence (see `Convergence` value below).

`DIC` Basic deviance information criterion (DIC) calculations to be used for model selection (Spiegelhalter *et al.* 2002).

`KullbackLeibler`
List with Kullback-Leibler discrepancy matrices between pair of parameters for categorical covariates (McCulloch 1989, Burnham and Anderson 2001) and McCulloch's (1989) calibration measure. If only one simulation was ran or if no convergence was reached, then the returned value is "`Not calculated`".

`convergence` A matrix with convergence coefficients based on potential scale reduction as described by Gelman *et al.* (2004). If only one simulation was ran, then the returned value is "`Not calculated`".

`modelSpecs` Model specifications inidicating the `model`, the `shape` and the covariate structure that were specified by the user.

`settings` A vector indicating the number of iterations for each MCMC, the burn in sequence, the thinning interval, and the number of simulations that were run.

Author(s)

Fernando Colchero <colchero@imada.sdu.dk>, Owen R. Jones <jones@biology.sdu.dk> and Maren Rebke <maren.rebke@avitec-research.de>

References

Gelman, A., Carlin, J.B., Stern, H.S. and Rubin, D.B. (2004) *Bayesian data analysis*. 2nd edn. Chapman & Hall/CRC, Boca Raton, Florida, USA.

McCulloch, R.E. (1989) Local model influence. *Journal of the American Statistical Association*, 84, 473-478.

Spiegelhalter, D.J., Best, N.G., Carlin, B.P. and Van Der Linde, A. (2002) Bayesian measures of model complexity and fit. *Journal of the Royal Statistical Society: Series B* 64, 583-639.

See also:

Colchero, F. and J.S. Clark (2012) Bayesian inference on age-specific survival from capture-recapture data for censored and truncated data. *Journal of Animal Ecology*. 81(1):139-149.

Colchero, F., O.R. Jones and M. Rebke. (2012) BaSTA: an R package for Bayesian estimation of age-specific survival from incomplete mark-recapture/recovery data with covariates. *Method in Ecology and Evolution*. DOI: 10.1111/j.2041-210X.2012.00186.x

See Also

`basta`

Examples

```
## Load BaSTA output:
data("sim1Out", package = "BaSTA")

## Print summary output:
summary(sim1Out)

## Plot traces for mortality parameters (theta):
plot(sim1Out)

## Plot traces for proportional hazards parameters (gamma):
plot(sim1Out, trace.name = "gamma")

## Plot traces for recapture probability(ies) (pi):
plot(sim1Out, trace.name = "pi")

## Plot predicted mortality and survival:
plot(sim1Out, plot.trace = FALSE)

## Change the color for each covariate on
## the predicted vital rates:
plot(sim1Out, plot.trace = FALSE,
     col = c("dark green", "dark blue"))

## Change the color and the legend text:
plot(sim1Out, plot.trace = FALSE,
     col = c("dark green", "dark blue"),
     names.legend = c("Females", "Males"))

## Plot predicted mortality and survival
## between 2 and 8 years of age:
plot(sim1Out, plot.trace = FALSE, xlim = c(2, 8))

## Plot predicted mortality and survival
## between 2 and 8 years of age without
## credible intervals:
plot(sim1Out, plot.trace = FALSE, xlim = c(2, 8),
     noCI = TRUE)

## Plot parameter densities and predicted vital
## rates in the same plot (i.e. fancy):
plot(sim1Out, fancy = TRUE)

## Change colors and legend names for the
## "fancy" plot:
plot(sim1Out, fancy = TRUE, col = c("dark green", "dark blue"),
     names.legend = c("Females", "Males"))
```

Index