abc

March 28, 2020

Version 1.3

Date 2011-05-09

Title Tools for Approximate Bayesian Computation (ABC)

Author Katalin Csillery, Michael Blum and Olivier Francois

Maintainer Katalin Csillery <kati.csillery@gmail.com> and Michael Blum <michael.blum@imag.fr>

Depends R (>= 2.10), nnet, quantreg, locfit

Description The package implements several ABC algorithms for performing parameter estimation and model selection. Cross-validation tools are also available for measuring the accuracy of ABC estimates, and to calculate the misclassification probabilities of different models.

Repository CRAN

License GPL (>= 3)

Date/Publication 2011-05-10 09:42:38

R topics documented:

abc *Parameter estimation with Approximate Bayesian Computation (ABC)*

Description

This function performs multivariate parameter estimation based on summary statistics using an ABC algorithm. The algorithms implemented are rejection sampling, and local linear or non-linear (neural network) regression. A conditional heteroscedastic model is available for the latter two algorithms.

Usage

```
abc(target, param, sumstat, tol, method, hcorr = TRUE, transf = "none",
logit.bounds, subset = NULL, kernel = "epanechnikov", numnet =
10, sizenet = 5, lambda = c(0.0001,0.001,0.01), trace = FALSE, maxit =
500, ...)
```

Arguments

`target` a vector of the observed summary statistics.

`param` a vector, matrix or data frame of the simulated parameter values, i.e. the dependent variable(s) when `method` is `"loclinear"` or `"neuralnet"`.

`sumstat` a vector, matrix or data frame of the simulated summary statistics, i.e. the independent variables when `method` is `"loclinear"` or `"neuralnet"`.

`tol` tolerance, the required proportion of points accepted nearest the target values.

`method` a character string indicating the type of ABC algorithm to be applied. Possible values are `"rejection"`, `"loclinear"`, and `"neuralnet"`. See also `Details`.

`hcorr` logical, the conditional heteroscedastic model is applied if `TRUE` (default).

`transf` a vector of character strings indicating the kind of transformation to be applied to the parameter values. The possible values are `"log"`, `"logit"`, and `"none"` (default), when no is transformation applied. See also `Details`.

`logit.bounds` a matrix of bounds if `transf` is `"logit"`. The matrix has as many lines as parameters (including the ones that are not `"logit"` transformed) and 2 columns. First column is the minimum bound and second column is the maximum bound.

`subset` a logical expression indicating elements or rows to keep. Missing values in `param` and/or `sumstat` are taken as `FALSE`.

`kernel` a character string specifying the kernel to be used when `method` is `"loclinear"` or `"neuralnet"`. Defaults to `"epanechnikov"`. See `density` for details.

`numnet` the number of neural networks when `method` is `"neuralnet"`. Defaults to 10. It indicates the number of times the function `nnet` is called.

`sizenet` the number of units in the hidden layer. Defaults to 5. Can be zero if there are no skip-layer units. See `nnet` for more details.

`lambda` a numeric vector or a single value indicating the weight decay when `method` is `"neuralnet"`. See `nnet` for more details. By default, 0.0001, 0.001, or 0.01 is randomly chosen for each of the networks.

`trace` logical, if `TRUE` switches on tracing the optimization of `nnet`. Applies only when `method` is `"neuralnet"`.

`maxit` numeric, the maximum number of iterations. Defaults to 500. Applies only when `method` is `"neuralnet"`. See also `nnet`.

`...` other arguments passed to `nnet`.

Details

These ABC algorithms generate random samples from the posterior distributions of one or more parameters of interest, $\theta_1, \theta_2, \ldots, \theta_n$. To apply any of these algorithms, (i) data sets have to be simulated based on random draws from the prior distributions of the θ_i's, (ii) from these data sets, a set of summary statistics have to be calculated, $S(y)$, (iii) the same summary statistics have to be calculated from the observed data, $S(y_0)$, and (iv) a tolerance rate must be chosen (`tol`). See `cv4abc` for a cross-validation tool that may help in choosing the tolerance rate.

When `method` is `"rejection"`, the simple rejection algorithm is used. Parameter values are accepted if the Euclidean distance between $S(y)$ and $S(y_0)$ is sufficiently small. The percentage of accepted simulations is determined by `tol`. When `method` is `"loclinear"`, a local linear regression method corrects for the imperfect match between $S(y)$ and $S(y_0)$. The accepted parameter values are weighted by a smooth function (`kernel`) of the distance between $S(y)$ and $S(y_0)$, and corrected according to a linear transform: $\theta^* = \theta - b(S(y) - S(y_0))$. θ^*'s represent samples form the posterior distribution. This method calls the function `lsfit` from the `stats` library. The non-linear regression correction method (`"neuralnet"`) uses a non-linear regression to minimize the departure from non-linearity using the function `nnet`. The posterior samples of parameters based on the rejection algorithm are returned as well, even when one of the regression algorithms is used.

Several additional arguments can be specified when `method` is `"neuralnet"`. The method is based on the function `nnet` from the library `nnet`, which fits single-hidden-layer neural networks. `numnet` defines the number of neural networks, thus the function `nnet` is called `numnet` number of times. Predictions from different neural networks can be rather different, so the median of the predictions from all neural networks is used to provide a global prediction. The choice of the number of neural networks is a trade-off between speed and accuracy. The default is set to 10 networks. The number of units in the hidden layer can be specified via `sizenet`. Selecting the number of hidden units is similar to selecting the independent variables in a linear or non-linear regression. Thus, it corresponds to the complexity of the network. There is several rule of thumb to choose the number of hidden units, but they are often unreliable. Generally speaking, the optimal choice of `sizenet` depends on the dimensionality, thus the number of statistics in `sumstat`. It can be zero when there are no skip-layer units. See also `nnet` for more details. The `method` `"neuralnet"` is recommended when dealing with a large number of summary statistics.

If `method` is `"loclinear"` or `"neuralnet"`, a correction for heteroscedasticity is applied by default (`hcorr = TRUE`).

Parameters maybe transformed priori to estimation. The type of transformation is defined by `transf`. The length of `transf` is normally the same as the number of parameters. If only one value is given, that same transformation is applied to all parameters and the user is warned. When a parameter transformation used, the parameters are back-transformed to their original scale after the regression estimation. No transformations can be applied when `method` is `"rejection"`.

Using names for the parameters and summary statistics is strongly recommended. Names can be supplied as `names` or `colnames` to `param` and `sumstat` (and `target`). If no names are supplied, P1, P2, . . . is assigned to parameters and S1, S2, . . . to summary statistics and the user is warned.

Value

The returned value is an object of class `"abc"`, containing the following components:

`adj.values` The regression adjusted values, when `method` is `"loclinear"` or `"neuralnet"`.

`unadj.values` The unadjusted values that correspond to `"rejection"` `method`.

`ss` The summary statistics for the accepted simulations.

`weights` The regression weights, when `method` is `"loclinear"` or `"neuralnet"`.

`residuals` The residuals from the regression when `method` is `"loclinear"` or `"neuralnet"`. These are the "raw" residuals from `lsfit` or `nnet`, respectively, thus they are not on the original scale of the parameter(s).

`dist` The Euclidean distances for the accepted simulations.

`call` The original call.

`na.action` A logical vector indicating the elements or rows that were excluded, including both `NA`/`NaN`'s and elements/rows selected by `subset`.

`region` A logical expression indicting the elements or rows that were accepted.

`transf` The parameter transformations that have been used.

`logit.bounds` The bounds, if transformation was `"logit"`.

`kernel` The kernel used.

`method` Character string indicating the `method`, i.e. `"rejection"`, `"loclinear"`, or `"neuralnet"`.

`lambda` A numeric vector of length `numnet`. The actual values of the decay parameters used in each of the neural networks, when `method` is `"neuralnet"`. These values are selected randomly from the supplied vector of values.

`numparam` Number of parameters used.

`numstat` Number of summary statistics used.

`names` A list with two elements: `parameter.names` and `statistics.names`. Both contain a vector of character strings with the parameter and statistics names, respectively.

Author(s)

Katalin Csillery, Olivier Francois and Michael Blum with some initial code from Mark Beaumont (`http://www.rubic.rdg.ac.uk/~mab/`).

References

Pritchard, J.K., and M.T. Seielstad and A. Perez-Lezaun and M.W. Feldman (1999) Population growth of human Y chromosomes: a study of Y chromosome microsatellites. *Molecular Biology and Evolution*, **16**, 1791–1798.

Beaumont, M.A., Zhang, W., and Balding, D.J. (2002) Approximate Bayesian Computation in Population Genetics, *Genetics*, **162**, 2025-2035.

Blum, M.G.B. and Francois, O. (2010) Non-linear regression models for Approximate Bayesian Computation. *Statistics and Computing* **20**, 63-73.

Csillery, K., M.G.B. Blum, O.E. Gaggiotti and O. Francois (2010) Approximate Bayesian Computation (ABC) in practice. *Trends in Ecology and Evolution*, **25**, 410-418.

See Also

`summary.abc`, `hist.abc`, `plot.abc`, `lsfit`, `nnet`, `cv4abc`

Examples

```
data(musigma2)
## this data set contains five R objects, see ?musigma2 for
## details

## The rejection algorithm
##
rej <- abc(target=stat.obs, param=par.sim, sumstat=stat.sim, tol=.1, method =
"rejection")

## ABC with local linear regression correction without/with correction
## for heteroscedasticity
##
lin <- abc(target=stat.obs, param=par.sim, sumstat=stat.sim, tol=.1, hcorr =
FALSE, method = "loclinear", transf=c("none","log"))
linhc <- abc(target=stat.obs, param=par.sim, sumstat=stat.sim, tol=.1, method =
"loclinear", transf=c("none","log"))

## ABC with neural networks with correction for heteroscedasticity
##
net <- abc(target=stat.obs, param=par.sim, sumstat=stat.sim,
tol=.2, method="neuralnet", transf=c("none","log"))

## posterior summaries
##
linsum <- summary(linhc, intvl = .9)
linsum
## compare with the rejection sampling
summary(linhc, unadj = TRUE, intvl = .9)

## posterior histograms
##
hist(linhc, breaks=30, caption=c(expression(mu),
expression(sigma^2)))
```

```
## or send histograms to a pdf file
hist(linhc, file="linhc", breaks=30, caption=c(expression(mu),
expression(sigma^2)))

## diagnostic plots: compare the 3 'abc' objects: "loclinear",
## "loclinear" with correction for heteroscedasticity, and "neuralnet"
## with correction for heteroscedasticity
##
plot(lin, param=par.sim)
plot(linhc, param=par.sim)
plot(net, param=par.sim)

## example illustrates how to add "true" parameter values to a plot
##
postmod <- c(post.mu[match(max(post.mu[,2]), post.mu[,2]),1],
             post.sigma2[match(max(post.sigma2[,2]), post.sigma2[,2]),1])
plot(net, param=par.sim, true=postmod)


## artificial example to show how to use the logit tranformations
##
myp <- data.frame(par1=runif(1000,-1,1),par2=rnorm(1000),par3=runif(1000,0,2))
mys <- myp+rnorm(1000,sd=.1)
myt <- c(0,0,1.5)
lin2 <- abc(target=myt, param=myp, sumstat=mys, tol=.1, method =
"loclinear", transf=c("logit","none","logit"),logit.bounds = rbind(c(-1,
1), c(NA, NA), c(0, 2)))
summary(lin2)
```

cv4abc *Cross validation for Approximate Bayesian Computation (ABC)*

Description

This function performs a leave-one-out cross validation for ABC via subsequent calls to the function abc. A potential use of this function is to evaluate the effect of the choice of the tolerance rate on the quality of the estimation with ABC.

Usage

```
cv4abc(param, sumstat, abc.out = NULL, nval, tols, statistic = "median",
prior.range = NULL, method, hcorr = TRUE, transf = "none", logit.bounds
= c(0,0), subset = NULL, kernel = "epanechnikov", numnet = 10, sizenet =
5, lambda = c(0.0001,0.001,0.01), trace = FALSE, maxit = 500, ...)
```

Arguments

param	a vector, matrix or data frame of the simulated parameter values.

`sumstat` a vector, matrix or data frame of the simulated summary statistics.

`abc.out` an object of class `"abc"`, optional. If supplied, all arguments passed to `abc` are extracted from this object, except for `sumstat`, `param`, and `tol`, which always have to be supplied as arguments.

`nval` size of the cross-validation sample.

`tols` a single tolerance rate or a vector of tolerance rates.

`statistic` a character string specifying the statistic to calculate a point estimate from the posterior distribution of the parameter(s). Possible values are `"median"` (default), `"mean"`, or `"mode"`.

`prior.range` a range to truncate the prior range.

`method` a character string indicating the type of ABC algorithm to be applied. Possible values are `"rejection"`, `"loclinear"`, and `"neuralnet"`. See also `abc`.

`hcorr` logical, if `TRUE` (default) the conditional heteroscedastic model is applied.

`transf` a vector of character strings indicating the kind of transformation to be applied to the parameter values. The possible values are `"log"`, `"logit"`, and `"none"` (default), when no is transformation applied. See also `abc`.

`logit.bounds` a vector of bounds if `transf` is `"logit"`. These bounds are applied to all parameters that are to be logit transformed.

`subset` a logical expression indicating elements or rows to keep. Missing values in `param` and/or `sumstat` are taken as `FALSE`.

`kernel` a character string specifying the kernel to be used when `method` is `"loclinear"` or `"neuralnet"`. Defaults to `"epanechnikov"`. See `density` for details.

`numnet` the number of neural networks when `method` is `"neuralnet"`. Defaults to 10. It indicates the number of times the function `nnet` is called.

`sizenet` the number of units in the hidden layer. Defaults to 5. Can be zero if there are no skip-layer units. See `nnet` for more details.

`lambda` a numeric vector or a single value indicating the weight decay when `method` is `"neuralnet"`. See `nnet` for more details. By default, 0.0001, 0.001, or 0.01 is randomly chosen for each of the networks.

`trace` logical, `TRUE` switches on tracing the optimization of `nnet`. Applies only when `method` is `"neuralnet"`.

`maxit` numeric, the maximum number of iterations. Defaults to 500. Applies only when `method` is `"neuralnet"`. See also `nnet`.

`...` other arguments passed to `nnet`.

Details

A simulation is selected repeatedly to be a validation simulation, while the other simulations are used as training simulations. Each time the function `abc` is called to estimate the parameter(s). A total of `nval` validation simulations are selected.

The arguments of the function `abc` can be supplied in two ways. First, simply give them as arguments when calling this function, in which case `abc.out` can be `NULL`. Second, via an existing object of class `"abc"`, here `abc.out`. WARNING: when `abc.out` is supplied, the same `sumstat` and `param` objects have to be used as in the original call to `abc`. Column names of `sumstat` and `param` are checked for match.

See `summary.cv4abc` for calculating the prediction error from an object of class `"cv4abc"`.

Value

An object of class `"cv4abc"`, which is a list with the following elements

`call` The original calls to `abc` for each tolerance rates.

`cvsamples` Numeric vector of length `nval`, indicating which rows of the `param` and `sumstat` matrices were used as validation values.

`tols` The tolerance rates.

`true` The parameter values that served as validation values.

`estim` The estimated parameter values.

`names` A list with two elements: `parameter.names` and `statistics.names`. Both contain a vector of character strings with the parameter and statistics names, respectively.

`seed` The value of `.Random.seed` when `cv4abc` is called.

See Also

`abc`, `plot.cv4abc`, `summary.cv4abc`

Examples

```
data(musigma2)
## this data set contains five R objects, see ?musigma2 for
## details

## cv4abc() calls abc(). Here we show two ways for the supplying
## arguments of abc(). 1st way: passing arguments directly. In this
## example only 'param', 'sumstat', 'tol', and 'method', while default
## values are used for the other arguments.
##
cv.rej <- cv4abc(param=par.sim, sumstat=stat.sim, nval=50,
tols=c(.1,.2,.3), method=”rejection”)

## 2nd way: first creating an object of class 'abc', and then using it
## to pass its arguments to abc().
##
lin <- abc(target=stat.obs, param=par.sim, sumstat=stat.sim, tol=.2,
method=”loclinear”, transf=c(”none”,”log”))
cv.lin <- cv4abc(param=par.sim, sumstat=stat.sim, abc.out=lin, nval=50,
tols=c(.1,.2,.3))

## using the plot method. Different tolerance levels are plotted with
## different heat.colors. Smaller the tolerance levels correspond to
## ”more red” points.
## !!! consider using the argument 'exclude' (plot.cv4abc) to supress
## the plotting of any outliers that mask readibility !!!
plot(cv.lin, log=c("xy", "xy"), caption=c(expression(mu),
expression(sigma^2)))

## comparing with the rejection sampling
plot(cv.rej, log=c("", "xy"), caption=c(expression(mu), expression(sigma^2)))

## or printing results directly to a postscript file...
plot(cv.lin, log=c("xy", "xy"), caption=c(expression(mu),
expression(sigma^2)), file="CVrej", postscript=TRUE)

## using the summary method to calculate the prediction error
summary(cv.lin)
## compare with rejection sampling
summary(cv.rej)
```

cv4postpr *Leave-one-our cross validation for model selection ABC*

Description

This function performs a leave-one-out cross validation for model selection with ABC via subsequent calls to the function postpr.

Usage

```
cv4postpr(index, sumstat, postpr.out = NULL, nval, tols, method,
subset = NULL, kernel = "epanechnikov", numnet = 10, sizenet = 5, lambda
= c(0.0001,0.001,0.01), trace = FALSE, maxit = 500, ...)
```

Arguments

index — a vector of model indices. It can be character or numeric and will be coerced to factor. It must have the same length as the number of rows in sumstat to indicate which row of sumstat belong to which model.

sumstat — a vector, matrix or data frame of the simulated summary statistics.

postpr.out — an object of class "postpr", optional. If supplied, all arguments passed to postpr are extracted from this object, except for sumstat, index, and tols, which always have to be supplied as arguments.

nval — the size of the cross-validation sample for each model.

tols — a single tolerance rate or a vector of tolerance rates.

method — a character string indicating the type of simulation required. Possible values are "rejection", "mnlogistic", "neuralnet". See postpr for details.

subset — a logical expression indicating elements or rows to keep. Missing values in index and/or sumstat are taken as FALSE.

kernel — a character string specifying the kernel to be used when method is "loclinear" or "neuralnet". Defaults to "epanechnikov". See density for details.

`numnet` the number of neural networks when `method` is `"neuralnet"`. Defaults to 10. It indicates the number of times the function `nnet` is called.

`sizenet` the number of units in the hidden layer. Defaults to 5. Can be zero if there are no skip-layer units. See `nnet` for more details.

`lambda` a numeric vector or a single value indicating the weight decay when `method` is `"neuralnet"`. See `nnet` for more details. By default, 0.0001, 0.001, or 0.01 is randomly chosen for each of the networks.

`trace` logical, `TRUE` switches on tracing the optimization of `nnet`. Applies only when `method` is `"neuralnet"`.

`maxit` numeric, the maximum number of iterations. Defaults to 500. Applies only when `method` is `"neuralnet"`. See also `nnet`.

`...` other arguments passed to `nnet`.

Details

For each model, a simulation is selected repeatedly to be a validation simulation, while the other simulations are used as training simulations. Each time the function `postpr` is called to estimate the parameter(s).

Ideally, we want `nval` to be equal to the number of simulations for each model, however, this might take too much time. Users are warned not to choose a too large number of simulations (especially when the neural networks are used). Beware that the actual number of cross-validation estimation steps that need to be performed is `nval`*the number of models.

The arguments for the function `postpr` can be supplied in two ways. First, simply give them as arguments when calling this function, in which case `postpr.out` can be `NULL`. Second, via an existing object of class `"postpr"`, here `postpr.out`. WARNING: when `postpr.out` is supplied, the same `sumstat` and `param` objects have to be used as in the original call to `postpr`. Column names of `sumstat` and `param` are checked for match.

See `summary.cv4postpr` for calculating the prediction error from an object of class `"cv4postpr"` and `plot.cv4postpr` for visualizing the misclassification of the models using barplots.

Value

An object of class `"cv4postpr"`, which is a list with the following elements

`call` The original calls to `postpr` for each tolerance rates.

`cvsamples` Numeric vector of length `nval`*the number of models, indicating which rows of `sumstat` were used as validation values.

`tols` The tolerance rates.

`true` The true models.

`estim` The estimated model probabilities.

`method` The method used.

`names` A list of two elements: `model` contains the model names, and `statistics.names` the names of the summary statistics.

`seed` The value of `.Random.seed` when `cv4postpr` is called.

See Also

postpr, summary.cv4postpr, plot.cv4postpr

Examples

```
data(human)
africa <- postpr(tajima.obs["Hausa",], models, tajima.sim, tol=.1,
method="mnlogistic")
summary(africa)
cv.africa <- cv4postpr(models, tajima.sim, postpr.out=africa, nval=20,
tols=c(.01, .02))
summary(cv.africa)
class(cv.africa)
plot(cv.africa, names.arg=c("Bottleneck", "Constant", "Exponential"))
```

expected.deviance *Expected deviance*

Description

Model selection criterion based on posterior predictive distributions and approximations of the expected deviance.

Usage

```
expected.deviance(target, postsumstat, kernel = "gaussian", subset=NULL,
print=TRUE)
```

Arguments

target a vector of the observed summary statistics.

postsumstat a vector, matrix or data frame of summary statistics simulated a posteriori.

kernel a character string specifying the kernel to be used when. Defaults to "gaussian". See density for details.

subset a logical expression indicating elements or rows to keep. Missing values in postsumstat are taken as FALSE.

print prints out what percent of the distances have been zero.

Details

This function implements an approximation for the expected deviance based on simulation performed a posteriori. Thus, after the posterior distribution of parameters or the posterior model probabilities have been determined, users need to re-simulate data using the posterior. The Monte-Carlo estimate of the expected deviance is computed from the simulated data as follows: $D = -\frac{2}{n}\sum_{j=1}^{n} \log(K_\epsilon(\parallel s^j - s_0 \parallel))$, where n is number of simulations, K is the statistical kernel, ϵ is the error, i.e. difference between the observed and simulated summary statistics below which simualtions were accepted in the original call to postpr, the s^j's are the summary statistics obtained from the posterior predictive simualtions, and s_0 are the observed values of the summary statistics. The expected devaince averaged over the posterior distribution to compute a deviance information criterion (DIC).

Value

A list with the following components:

`expected.deviance`
: The approximate expected deviance.

`dist`
: The Euclidean distances for summary statistics simulated a posteriori.

References

Francois O, Laval G (2011) Deviance information criteria for model selection in approximate Bayesian computation *arXiv*:**0240377**.

Examples

```
## Function definitions
skewness <- function(x) {
sk <- mean((x-mean(x))^3)/(sd(x)^3)
return(sk)
}
kurtosis <- function(x) {
k <- mean((x-mean(x))^4)/(sd(x)^4) - 3
return(k)
}

## Observed summary statistics
obs.sumstat <- c(2.004821, 3.110915, -0.7831861, 0.1440266)

## Model 1 (Gaussian)
## ##################
## Simulate data
theta <- rnorm(10000, 2, 10)
zeta <- 1/rexp(10000, 1)
param <- cbind(theta, zeta)
y <- matrix(rnorm(200000, rep(theta, each = 20), sd = rep(sqrt(zeta),
each = 20)), nrow = 20, ncol = 10000)

## Calculate summary statistics
s <- cbind(apply(y, 2, mean), apply(y, 2, sd), apply(y, 2, skewness),
apply(y, 2, kurtosis))

## ABC inference
gaus <- abc(target=obs.sumstat, param = param, sumstat=s, tol=.1, hcorr =
FALSE, method = "loclinear")
param.post <- gaus$adj.values

## Posterior predictive simulations
postpred.gaus <- matrix(rnorm(20000, rep(param.post[,1], each = 20), sd
= rep(sqrt(param.post[,2]), each = 20)), nrow = 20, ncol = 1000)
statpost.gaus <- cbind(apply(postpred.gaus, 2,
mean),apply(postpred.gaus, 2, sd),apply(postpred.gaus,
2,skewness),apply(postpred.gaus, 2,kurtosis))

# Computation of the expected deviance
expected.deviance(obs.sumstat, statpost.gaus)$expected.deviance
expected.deviance(obs.sumstat, statpost.gaus, kernel =
"epanechnikov")$expected.deviance

## Modele 2 (Laplace)
## ##################
## Simulate data
zeta <- rexp(10000)
param <- cbind(theta, zeta)
y <- matrix(theta + sample(c(-1,1),200000, replace = TRUE)*rexp(200000,
rep(zeta, each = 20)), nrow = 20, ncol = 10000)

## Calculate summary statistics
s <- cbind( apply(y, 2, mean), apply(y, 2, sd), apply(y, 2, skewness),
apply(y, 2, kurtosis))

## ABC inference
lapl <- abc(target=obs.sumstat, param = param, sumstat=s, tol=.1, hcorr =
FALSE, method = "loclinear")
param.post <- lapl$adj.values

## Posterior predictive simulations
postpred.lapl <- matrix(param.post[,1] + sample(c(-1,1),20000, replace =
TRUE)*rexp(20000, rep(param.post[,2], each = 20)), nrow = 20, ncol =
1000)
statpost.lapl <- cbind(apply(postpred.lapl, 2,
mean),apply(postpred.lapl, 2, sd),apply(postpred.lapl,
2,skewness),apply(postpred.lapl, 2,kurtosis))

## Computation of the expected deviance
expected.deviance(obs.sumstat, statpost.lapl)$expected.deviance
expected.deviance(obs.sumstat, statpost.lapl, kernel =
"epanechnikov")$expected.deviance
```

hist.abc *Posterior histograms*

Description

Histograms of posterior samples from objects of class "abc".

Usage

```
## S3 method for class 'abc'
hist(x, unadj = FALSE, true = NULL, file = NULL,
postscript = FALSE, onefile = TRUE, ask =
!is.null(deviceIsInteractive()), col.hist = "grey", col.true = "red",
caption = NULL, ...)
```

Arguments

`x` an object of class `"abc"`.

`unadj` logical, if `TRUE` the unadjusted values are plotted even if method is `"loclinear"` or `"neuralnet"`.

`true` the true parameter value(s), if known. Vertical bar(s) are drawn at the true value(s). If more than one parameters were estimated, a vector of the true values have to be supplied.

`file` a character string giving the name of the file. See `postscript` for details on accepted file names. If `NULL` (the default) histograms are printed to the null device (e.g. `X11`). If not `NULL` histograms are printed on a `pdf` device. See also `postscript`.

`postscript` logical; if `FALSE` (default) histograms are printed on a `pdf` device, if `TRUE` on a `postscript` device.

`onefile` logical, if `TRUE` (the default) allow multiple figures in one file. If `FALSE`, generate a file name containing the page number for each page. See `postscript` for further details.

`ask` logical; if `TRUE` (the default), the user is asked before each plot, see `par(ask=.)`.

`col.hist` the colour of the histograms.

`col.true` the colour of the vertical bar at the true value.

`caption` captions to appear above the histogram(s); `character` vector of valid graphics annotations, see `as.graphicsAnnot` for details. When `NULL` (default), `parnames` are used, which are extracted from `x` (see `abc`). Can be set to `NA` to suppress all captions.

`...` other parameters passed to `hist`.

Value

A list of length equal to the number of parameters, the elements of which are objects of class `"histogram"`. See `hist` for details.

See Also

`abc`, `plot.abc`

Examples

```
## see ?abc for examples
```

`human` *A set of R objects used to illustrate model selection in an ABC framework*

Description

`data(human)` loads in three R objects: `tajima.obs` is a data frame with 3 rows and 2 columns and contains the observed summary statistics, `tajima.sim` is also a data frame with 150,000 rows and 2 columns and contains the simulated summary statistics, and `models` is a vector of character strings of length 150,000 and contains the model indices.

Usage

```
data(human)
```

Format

The `tajima.obs` data frame contains the following columns:

`D.mean` The mean of Tajima's D statistic over 50 loci in 3 human populations, Hausa, Italian, and Chinese.

`D.var` The variance of Tajima's D statistic over 50 loci in 3 human populations, Hausa, Italian, and Chinese.

The `tajima.sim` data frame contains the following columns:

`D.mean` The mean of Tajima's D statistic over 50 simulated loci under 3 demographic scenarios: constant size population, population bottleneck, and population expansion.

`D.var` The variance of Tajima's D statistic over 50 simulated loci under 3 demographic scenarios: constant size population, population bottleneck, and population expansion.

Each row represents a simulation. Under each model 50,000 simulations were performed. Row names indicate the type of demographic model.

`models` contains the names of the demographic models.

Details

Data is provided to estimate the posterior probabilities of classical demographic scenarios in three human populations: Hausa, Italian, and Chinese. These three populations represent the three continents: Africa, Europe, Asia, respectively.

It is generally believed that African human populations are expanding, while human populations from outside of Africa have gone through a population bottleneck. Tajima's D statistic has been classically used to detect changes in historical population size. A negative Tajima's D signifies an excess of low frequency polymorphisms, indicating population size expansion. While a positive Tajima's D indicates low levels of both low and high frequency polymorphisms, thus a sign of a population bottleneck. In constant size populations, Tajima's D is expected to be zero.

With the help of the `human` data one can reach these expected conclusions for the three human population samples, in accordance with the conclusions of Voight et al. (2005) (where the observed statistics was taken from), but using ABC.

Source

The observed statistics were taken from Voight et al. 2005 (Table 1.). Also, the same input parameters were used as in Voight et al. 2005 to simulate data under the three demographic models. Simulations were performed using the software *ms* and the summary statistics were calculated using *sample_stats* (Hudson 1983).

References

B. F. Voight, A. M. Adams, L. A. Frisse, Y. Qian, R. R. Hudson and A. Di Rienzo (2005) Interrogating multiple aspects of variation in a full resequencing data set to infer human population size changes. *PNAS* **102**, 18508-18513.

Hudson, R. R. (2002) Generating samples under a Wright-Fisher neutral model of genetic variation. *Bioinformatics* **18** 337-338.

musigma2 *A set of objects used to estimate the population mean and variance in a Gaussian model with ABC.*

Description

musigma2 loads in five R objects: par.sim is a data frame and contains the parameter values of the simulated data sets, stat is a data frame and contains the simulated summary statistics, stat.obs is a data frame and contains the observed summary statistics, post.mu and post.sigma2 are data frames and contain the true posterior distributions for the two parameters of interest, μ and σ^2, respectively.

Usage

```
data(musigma2)
```

Format

The par.sim data frame contains the following columns:

mu The population mean.

sigma2 The population variance.

The stat.sim and stat.obs data frames contain the following columns:

mean The sample mean.

var The logarithm of the sample variance.

The post.mu and post.sigma2 data frames contain the following columns:

x the coordinates of the points where the density is estimated.

y the posterior density values.

Details

The prior of σ^2 is an inverse χ^2 distribution with one degree of freedom. The prior of μ is a normal distribution with variance of σ^2. For this simple example, the closed form of the posterior distribution is available.

Source

The observed statistics are the mean and variance of the sepal of *Iris setosa*, estimated from part of the `iris` data.

The data were collected by Anderson, Edgar.

References

Anderson, E. (1935). The irises of the Gaspe Peninsula, *Bulletin of the American Iris Society*, **59**, 2-5.

See Also

`abc`, `cv4abc`

`plot.abc` *Diagnostic plots for ABC*

Description

A plotting utile for quick visualization of the quality of an ABC analysis from an object of class `"abc"` generated with methods `"loclinear"` or `"neuralnet"` (see `abc` for details). Four plots are currently available: a density plot of the prior distribution, a density plot of the posterior distribution, a scatter plot of the Euclidean distances as a function of the parameter values, and a Normal Q-Q plot of the residuals from the regression.

Usage

```
## S3 method for class 'abc'
plot(x, param, subsample = 1000, true = NULL, file = NULL,
postscript = FALSE, onefile = TRUE, ask =
!is.null(deviceIsInteractive()), ...)
```

Arguments

`x`	an object of class `"abc"` generated with methods `"loclinear"` or `"neuralnet"` (see `abc` for details).
`param`	a vector or matrix of parameter values from the simulations that were used in the original call to `abc`.
`subsample`	the number of rows (simulations) to be plotted. Rows are randomly selected from `param`.

`true` a vector of true parameter values, if known. Vertical lines are drawn at these values.

`file` a character string giving the name of the file. See `postscript` for details on accepted file names. If `NULL` (the default) plots are printed to the null device (e.g. `X11`). If not `NULL` plots are printed on a `pdf` device. See also `postscript`.

`postscript` logical; if `FALSE` (default) plots are printed on a `pdf` device, if `TRUE` on a `postscript` device.

`onefile` logical, if `TRUE` (the default) allow multiple figures in one file. If `FALSE`, generate a file name containing the page number for each page. See `postscript` for further details.

`ask` logical; if `TRUE` (the default), the user is asked before each plot, see `par(ask=.)`.

`...` other parameters passed to `plot`.

Details

In order to use this function, one of the regression correction methods had to be used in the original call to `abc`, i.e. `"loclinear"` or `"neuralnet"` (see `abc` for details). Four plots are printed for each parameter. (i) A density plot of the prior distribution. (ii) A density plot of the posterior distribution using the regression correction (red thick lines) and, for reference, using the simple rejection method (black fine lines). The prior distribution (in the posterior distributions' range) is also displayed (dashed lines). (iii) A scatter plot of the log Euclidean distances as a function of the true parameter values. Points corresponding to the accepted simulations are displayed in red. (iv) A Normal Q-Q plot of the residuals from the regression, thus from `lsfit` when method was `"loclinear"`, and from `nnet` when method was `"neuralnet"` in the original `abc`.

For plots (i) and (iii) not the whole data but a subsample is used, the size of which can be is given by `subsample`. This is to avoid plots that may take too much time to print.

If a parameter transformation was applied in the original call to `abc`, the same transformations are applied to the parameters for plotting (on plots (i)-(iii)).

See Also

`abc`, `hist.abc`, `summary.abc`

Examples

```
## see ?abc for examples
```

`plot.cv4abc` *Cross-validation plots for ABC*

Description

Plotting method for cross-validation ABC objects. Helps to visually evaluate the quality of the estimation and/or the effect of the tolerance level.

Usage

```
## S3 method for class 'cv4abc'
plot(x, exclude = NULL, log = NULL, file = NULL,
postscript = FALSE, onefile = TRUE, ask =
!is.null(deviceIsInteractive()), caption = NULL, ...)
```

Arguments

`x` an object of class `"cv4abc"`.

`exclude` a vector of row indices indicating which rows should be excluded from plotting. Useful when the prior distribution has a long tail.

`log` character vector of the same length as the number of parameters in the `"cv4abc"` object. Allows plotting on a log scale. Possible values are "" (normal scale) and "xy" (log scale for both the x and y axis). "x" and "y" are possible as well, but not of any interest here. Negative values are set to `NA` and there is a warning.

`file` a character string giving the name of the file. See `postscript` for details on accepted file names. If `NULL` (the default) plots are printed to the null device (e.g. `X11`). If not `NULL` plots are printed on a `pdf` device. See also `postscript`.

`postscript` logical; if `FALSE` (default) plots are printed on a `pdf` device, if `TRUE` on a `postscript` device.

`onefile` logical, if `TRUE` (the default) allow multiple figures in one file. If `FALSE`, generate a file name containing the page number for each page. See `postscript` for further details.

`ask` logical; if `TRUE` (the default), the user is asked before each plot, see `par(ask=.)`.

`caption` captions to appear above the plot(s); `character` vector of valid graphics annotations, see `as.graphicsAnnot`. By default, `parnames` from `x` are extracted (see `abc`). Can be set to `""` or `NA` to suppress all captions.

`...` other parameters passed to `plot`.

Details

Different tolerance levels are plotted with `heat.colors`. Smaller the tolerance levels correspond to "more red" points.

See Also

`cv4abc`, `abc`

Examples

```
## see ?cv4abc for examples
```

plot.cv4postpr *Barplot of model misclassification*

Description

Displays a barplot of either the proportion of simulations classified to any of the models or the mean misclassification probabilities of models for all tolerance levels in the "cv4postpr" object.

Usage

```
## S3 method for class 'cv4postpr'
plot(x, probs = FALSE, file = NULL, postscript
= FALSE, onefile = TRUE, ask = !is.null(deviceIsInteractive()), caption
= NULL, ...)
```

Arguments

x	an object of class "cv4postpr".
probs	logical, if TRUE the mean posterior model probabilities are plotted. If FALSE the frequencies of the simulations classified to the different models (default).
file	a character string giving the name of the file. See postscript for details on accepted file names. If NULL (the default) plots are printed to the null device (e.g. X11). If not NULL plots are printed on a pdf device. See also postscript.
postscript	logical; if FALSE (default) plots are printed on a pdf device, if TRUE on a postscript device.
onefile	logical, if TRUE (the default) allow multiple figures in one file. If FALSE, generate a file name containing the page number for each page. See postscript for further details.
ask	logical; if TRUE (the default), the user is asked before each plot, see par(ask=.).
caption	captions to appear above the plot(s); character vector of valid graphics annotations, see as.graphicsAnnot. Can be set to "" or NA to suppress all captions.
...	other parameters passed to barplot.

Details

Model are distinguised with different intensities of the gray colour. The first model in alphabetic order has the darkest colour. If the classification of models is perfect (so that the frequency (or probability) of each model is zero for all but the correct model) each bar has a single colour of its corresponding model.

See Also

cv4postpr, summary.cv4postpr

Examples

```
## see ?cv4postpr for examples
```

postr

`postpr` *Estimating posterior model probabilities*

Description

Model selection with Approximate Bayesian Computation (ABC).

Usage

```
postpr(target, index, sumstat, tol, subset = NULL, method, corr=TRUE,
kernel="epanechnikov", numnet = 10, sizenet = 5, lambda =
c(0.0001,0.001,0.01), trace = TRUE, maxit = 500, ...)
```

Arguments

`target` a vector of the observed summary statistics.

`index` a vector of model indices. It can be character or numeric and will be coerced to factor. It must have the same length as `sumstat` to indicate which row of `sumstat` belong to which model.

`sumstat` a vector, matrix or data frame of the simulated summary statistics.

`tol` numeric, the required proportion of points nearest the target values (tolerance), or a vector of the desired tolerance values. If a vector is given

`subset` a logical expression indicating elements or rows to keep. Missing values in `index` and/or `sumstat` are taken as `FALSE`.

`method` a character string indicating the type of simulation required. Possible values are `"rejection"`, `"mnlogistic"`, `"neuralnet"`. See `Details`.

`corr` logical, if `TRUE` (default) posterior model probabilities are corrected for the number of simulations performed for each model. If equal number of simulations are available for all models, `corr` has no effect.

`kernel` a character string specifying the kernel to be used when method is `"mnlogistic"` or `"neuralnet"`. Defaults to "epanechnikov". See `density` for details.

`numnet` the number of neural networks when method is `"neuralnet"`. It corresponds to the number of times the function `nnet` is called.

`sizenet` the number of units in the hidden layer. Can be zero if there are no skip-layer units.

`lambda` a numeric vector or a single value indicating the weight decay when method is `"neuralnet"`. By default, 0.0001, 0.001, or 0.01 is randomly chosen for the each of the networks. See `nnet` for more details.

`trace` logical, `TRUE` switches on tracing the optimization of `nnet` (applies when method is `"neuralnet"`).

`maxit` numeric, the maximum number of iterations. Defaults to 500. See also `nnet`.

`...` other arguments passed on from `nnet`.

Details

The function computes the posterior model probabilities. Simulations have to be performed with at least two distinct models. When method is `"rejection"`, the posterior probability of a given model is approximated by the proportion of accepted simulations given this model. This approximation holds when the different models are a priori equally likely, and the same number of simulations is performed for each model. When method is `"mnlogistic"` the posterior model probabilities are estimated using a multinomial logistic regression as implemented in the function `multinom` from the package `nnet`. When method is `"neuralnet"`, neural networks are used to predict the probabilities of models based on the observed statistics using `nnet`. This method can be useful if many summary statistics are used.

Names for the summary statistics are strongly recommended. Names can be supplied as colnames to `sumstat` (and target). If no names are supplied S1, S2, ... to summary statistics will be assigned to parameters and the user will be warned.

Value

An object of `class` `"postpr"`, containing the following components:

`pred` a vector of model probabilities when method is `"mnlogistic"` or `"neuralnet"`.

`values` the vector of model indices in the accepted region using the rejection method.

`weights` vector of regression weights when method is `"mnlogistic"` or `"neuralnet"`.

`ss` summary statistics in the accepted region.

`call` the original call.

`na.action` a logical vector indicating the elements or rows that were excluded, including both `NA`/`NaN`'s and elements/rows selected by `subset`

`method` a character string indicating the method used, i.e. `"rejection"`, `"mnlogistic"` or `"neuralnet"`.

`corr` logical, if `TRUE` the posterior model probabilities are corrected for the number of simulations performed for each model.

`nmodels` the number of simulations performed for each model a priori.

`models` a character vector of model names (a priori).

`numstat` the number of summary statistics used.

`names` a list of two elements: `model` contains the model names, and `statistics.names` the names of the summary statistics.

Author(s)

Katalin Csillery, Olivier Francois and Michael Blum with some initial code from Mark Beaumont (`http://www.rubic.rdg.ac.uk/~mab/`).

References

Beaumont, M.A. (2008) Joint determination of topology, divergence time, and immigration in population trees. In *Simulation, Genetics, and Human Prehistory* (Matsumura, S., Forster, P. and Renfrew, C., eds) McDonald Institute for Archaeological Research

See Also

summary.postpr

Examples

```
data(human)
## five R objects are loaded. See ?human for details.

## the two summary statistics: mean and variance of Tajima's D over 50
## loci
par(mfcol = c(1,2))
boxplot(tajima.sim[,1]~models, main=names(tajima.sim)[1])
boxplot(tajima.sim[,2]~models, main=names(tajima.sim)[2])

## model selection with ABC for the three populations, representing
## three continents

## in Africa, population expansion is the most supported model
africa <- postpr(tajima.obs["Hausa",], models, tajima.sim, tol=.01,
method="mnlogistic")
summary(africa)

## in Europe and Asia, population bottleneck is the most supported model
europe <- postpr(tajima.obs["Italian",], models, tajima.sim, tol=.01,
method="mnlogistic")
summary(europe)
asia <- postpr(tajima.obs["Chinese",], models, tajima.sim, tol=.01,
method="mnlogistic")
summary(asia)

ss <- cbind(runif(1000),rt(1000,df=20))
postpr(target=c(3), index=c(rep("norm",500),rep("t",500)),
sumstat=ss[,1], tol=.1, method="rejection")
```

---

summary.abc *Summaries of posterior samples generated by ABC algortithms*

Description

Calculates simple summaries of posterior samples: the minimum and maximum, the weighted mean, median, mode, and credible intervals.

Usage

```
## S3 method for class 'abc'
summary(object, unadj = FALSE, intvl = .95, print = TRUE,
digits = max(3, getOption("digits")-3), ...)
```

Arguments

object — an object of class "abc".

unadj — logical, if TRUE it forces to plot the unadjusted values when method is "loclinear" or "neuralnet".

intvl — size of the symmetric credible interval.

print — logical, if TRUE prints messages. Mainly for internal use.

digits — the digits to be rounded to. Can be a vector of the same length as the number of parameters, when each parameter is rounded to its corresponding digits.

... — other arguments passed to density.

Details

If method is "rejection" in the original call to abc, posterior means, medians, modes and percentiles defined by intvl, 95 by default (credible intervals) are calculated. If a regression correction was used (i.e. method is "loclinear" or "neuralnet" in the original call to abc) the weighted posterior means, medians, modes and percentiles are calculated.

To calculate the mode, parameters are passed on from density.default. Note that the posterior mode can be rather different depending on the parameters to estimate the density.

Value

The returned value is an object of class "table". The rows are,

Min.	minimun
Lower perc.	lower percentile
Median	or weighted median
Mean	or weighted mean
Mode	or weighted mode
Upper perc.	upper percentile
Max.	maximum

See Also

abc, hist.abc, plot.abc

Examples

```
## see ?abc for examples
```

summary.cv4abc *Calculates the cross-validation prediction error*

Description

This function calculates the prediction error from an object of class "cv4abc" for each parameter and tolerance level.

Usage

```
## S3 method for class 'cv4abc'
summary(object, print = TRUE, digits = max(3,
getOption("digits")-3), ...)
```

Arguments

object	an object of class "abc".
print	logical, if TRUE prints messages. Mainly for internal use.
digits	the digits to be rounded to. Can be a vector of the same length as the number of parameters, when each parameter is rounded to its corresponding digits.
...	other arguments passed to density.

Details

The prediction error is calculated as $\frac{\sum((\theta^*-\theta)^2)}{Var(\theta^*)}$, where θ is the true parameter value and θ^* is the estimated parameter value.

Value

The returned value is an object of class "table", where the columns correspond to the parameters and the rows to the different tolerance levels.

See Also

cv4abc, plot.cv4abc

Examples

```
## see ?cv4abc for examples
```

summary.cv4postpr *Confusion matrix and misclassification probabilities of models*

Description

This function calculates the confusion matrix and the mean misclassification probabilities of models from an object of class "cv4postpr".

Usage

```
## S3 method for class 'cv4postpr'
summary(object, probs = TRUE, print = TRUE, digits =
max(3, getOption("digits")-3), ...)
```

Arguments

object	an object of class "cv4postpr".
probs	logical, if TRUE (default), mean posterior model probabilities are returned.
print	logical, if TRUE prints the mean models probabilities.
digits	the digits to be rounded to.
...	other arguments.

Value

If probs=FALSE a matrix with the frequencies of the simulations classified to the different models (the confusion matrix). If probs=TRUE, a list with two components:

conf.matrix	The confusion matrix.
probs	The mean model misclassification probabilities.

See Also

cv4postpr, plot.cv4postpr

Examples

```
## see ?cv4postpr for examples
```

summary.postpr	*Posterior model probabilities and Bayes factors*

Description

This function extracts the posterior model probabilities and calculates the Bayes factors from an object of class "postpr".

Usage

```
## S3 method for class 'postpr'
summary(object, rejection = TRUE, print = TRUE, digits
= max(3, getOption("digits")-3), ...)
```

Arguments

object	an object of class "postpr".
rejection	logical, if method is "mnlogistic" or "neuralnet", should the approximate model probabilities based on the rejection method returned.
print	logical, if TRUE prints the mean models probabilities.
digits	the digits to be rounded to.
...	other arguments.

Value

A list with the following components if method="rejection":

Prob	an object of class table of the posterior model probabilities.
BayesF	an object of class table with the Bayes factors between pairs of models.

A list with the following components if method is "mnlogistic" or "neuralnet" and rejection is TRUE:

rejection	a list with the same components as above
mnlogistic	a list with the same components as above

See Also

postpr

Examples

```
## see ?postpr for examples
```

Index