Package 'abcel'

May 7, 2026

Title Empirical Likelihood-Based Approximate Bayesian Computation

Version 1.0

Description
Empirical likelihood-based approximate Bayesian Computation. Approximates the required posterior using empirical likelihood and estimated differential entropy. This is achieved without requiring any specification of the likelihood or estimating equations that connects the observations with the underlying parameters. The procedure is known to be posterior consistent. More details can be found in Chaudhuri, Ghosh, and Kim (2024) <doi:10.1002/SAM.11711>.

License GPL-2

Encoding UTF-8

RoxygenNote 7.3.3

Imports MASS, emplik, methods, FNN, corpcor

NeedsCompilation yes

Author Nicholas Chua [aut],
Riddhimoy Ghosh [aut],
Sanjay Chaudhuri [aut, cre]

Maintainer Sanjay Chaudhuri <schaudhuri2@unl.edu>

Repository CRAN

Date/Publication 2025-11-21 14:40:02 UTC

Contents

abcel *Empirical Likelihood-based Approximate Bayesian Computation*

Description

Perform empirical likelihood-based posterior approximation for ABC problems.

Usage

```
abcel(
  data.obs,
  theta0,
  rep,
  m,
  n,
  burn_in,
  data.func,
  summ.func,
  prior.func,
  proposal.func = NULL,
  transform.func = NULL,
  fixed.summ.num = TRUE,
  print_interval = 1000,
  plot_interval = 0,
  which_plot = NULL,
  k = NULL,
  ...
)
```

Arguments

data.obs	Observed data.
theta0	Initial parameter values.
rep	number of MCMC runs.
m	number of generated samples statistics.
n	size of the data.
burn_in	size of the burn-in.
data.func	This function is used for generating the data.
summ.func	This function is used for generating the statistics on the data.
prior.func	Prior function.
proposal.func	This function is used to propose a new theta given the old thetas. This function needs to include the ellipsis argument.
transform.func	The function is used to transform the theta that will be fed to the generate data function. This function also needs to include the ellipsis argument. The default argument is the identity function.

fixed.summ.num	If the number of the summary statistics remains fixed from one MCMC step to another. (Defaults to TRUE).
print_interval	Fixed interval of iterations to print the results, if it's set to NULL, it will not print.
plot_interval	Fixed interval of iterations to plot the results, if it's set to NULL, there will be no plots.
which_plot	vector of parameters to plot.
k	order of the nearest neighbor to be used for k-NN based differential entropy estimation.
...	Args for transformation and proposal functions.

Details

By default, the sampler performs an adaptive random walk on the parameters with N(0,initial.cov) as its proposal density for the first initial_interval steps. Subsequently, the variance of the proposal Normal density will be updated based on the previously sampled parameters.

When defining a new proposal or a new transformation, be sure to insert the ellipsis argument.

Value

A list with the following components:

size: Size of the data.

logpost: The logarithm of the posterior numerator.

theta: A matrix of sampled theta values.

differential_entropy: A vector of computed differential entropies.

acceptance_rate: Stepwise acceptance rates.

References

Chaudhuri, S., Ghosh, S., & Pham, K. C. (2020). On an Empirical Likelihood-Based Solution to the Approximate Bayesian Computation Problem. Statistical Analysis and Data Mining, Vol 17(5):e11711.

Examples

```
## Not run:
 data.gk1<-function(theta,n=1000){
   p<-runif(n,min=0,max=1)
   h<-(1-exp(-theta[3]*qnorm(p)))/(1+exp(-theta[3]*qnorm(p)))
   x<-theta[1]+theta[2]*(1+0.8*h)*(1+qnorm(p)^2)^theta[4]*qnorm(p)
   return(x)
 }
 summary.gk1<-function(obs){
   summary=c(mean(obs),quantile(obs,p=c(.25,.5,.75)))
   return(summary)
 }
 prior.gk1<-function(theta){
```

```
  pr=c(dunif(theta[1],min=0,max=10,log=TRUE),
       dunif(theta[2],min=0,max=10,log=TRUE),
       dunif(theta[3],min=0,max=10,log=TRUE),
       dunif(theta[4],min=0,max=10,log=TRUE)
  )
  return(pr)
}
# run the algorithm
# parameters
rep<-100
m<-40
n<-1000
## target theta and the observed data
theta.t<-c(3,1,2,.5)
pr.t <- prior.gk1(theta.t)
d=length(theta.t)
data.obs<-data.gk1(theta.t,n=n)
initial.cov=diag(d)
diag(initial.cov)=c(1,1,1,1)*10^(-7)
# initialize mean and variance for the initial theta
marginal.mean.A<-3.003707
marginal.mean.B<-1.012046
marginal.mean.g<-2.017939
marginal.mean.k<-0.4894453
mean0<-matrix(0,4,1)
mean0[1,]<-marginal.mean.A
mean0[2,]<-marginal.mean.B
mean0[3,]<-marginal.mean.g
mean0[4,]<-marginal.mean.k
marginal.var.A<-0.0002019947
marginal.var.B<-0.0009072782
marginal.var.g<-0.004954367
marginal.var.k<-0.001020683
initial.cov<-matrix(0,4,4)
initial.cov[1,1]<-marginal.var.A
initial.cov[2,2]<-marginal.var.B
initial.cov[3,3]<-marginal.var.g
initial.cov[4,4]<-marginal.var.k
theta0<-MASS::mvrnorm(1,mean0,initial.cov/100)
abcel <- function(data.obs=data.obs,
                  theta0=theta0,
                  rep=rep,
                  m=m,
                  n=n,
                  burn_in=burn_in,
                  data.func=data.gk1,
                  summ.func=summary.gk1,
                  prior.func=prior.gk1,
                  print_interval=1000,
                  plot_interval=0,
                  which_plot=NULL)

## End(Not run)
```

entEst *New Entropy Estimator*

Description

Entropy estimator based on k-nearest neighbors.

Usage

```
entEst(summaries, k = NULL, terms = NULL, wts = NULL)
```

Arguments

summaries	data matrix with summary statistics as rows and samples as columns (d x m)
k	user-chosen k, or NULL
terms	user-supplied terms, or NULL
wts	user-supplied weights, or NULL

Value

Estimated differential entropy.

Examples

```
x=matrix(rnorm(2500),nrow=100)
entropy.true=.5*25*log(2*pi*exp(1))
entropy=entEst(t(x))
```

Index