Package ‘OneStep’

January 15, 2021

Type Package

Title One-Step Estimation

Version 0.9.2

Author Alexandre Brouste, Darel Noutsa Mieniedou, Christophe Dutang

Maintainer Christophe Dutang <christophe.dutang@ensimag.fr>

Contact Alexandre Brouste, Darel Noutsa Mieniedou, Christophe Dutang
   <onestep@univ-lemans.fr>

Description Provide principally an eponymic function that numerically computes the Le Cam one-step estimator which is asymptotically efficient (see e.g. L. Le Cam (1956) <https://projecteuclid.org/euclid.bsmsp/1200501652>) and can be computed faster than the maximum likelihood estimator for large observation samples.

License GPL (>= 2)

Encoding UTF-8

Depends fitdistrplus, numDeriv, parallel

Suggests actuar

NeedsCompilation no

Repository CRAN

Date/Publication 2021-01-15 21:50:02 UTC

R topics documented:

OneStep-package ......................................................... 2
benchonestep ............................................................ 3
onestep ................................................................. 4

Index 9
Description

Provide principally an eponymous function that numerically computes the Le Cam one-step estimator which is asymptotically efficient (see e.g. L. Le Cam (1956) <https://projecteuclid.org/euclid.bsmsp/1200501652>) and can be computed faster than the maximum likelihood estimator for large observation samples.

Details

The DESCRIPTION file:

Package: OneStep
Type: Package
Title: One-Step Estimation
Version: 0.9.2
Author: Alexandre Brouste, Darel Noutsa Mieniedou, Christophe Dutang
Maintainer: Christophe Dutang <christophe.dutang@ensimag.fr>
Contact: Alexandre Brouste, Darel Noutsa Mieniedou, Christophe Dutang <onestep@univ-lemans.fr>
Description: Provide principally an eponymous function that numerically computes the Le Cam one-step estimator which is asymptotically efficient and can be computed faster than the maximum likelihood estimator for large observation samples.
License: GPL (&ge; 2)
Encoding: UTF-8
Depends: fitdistrplus, numDeriv, parallel
Suggests: actuar

Index of help topics:

benchonestep Performing benchmark of one-step MLE against other methods
onestep Executing Le Cam's one-step estimation
OneStep-package One-Step Estimation

Author(s)

Alexandre Brouste, Darel Noutsa Mieniedou, Christophe Dutang
Maintainer: Christophe Dutang <christophe.dutang@ensimag.fr>

References


See Also

See fitdistrplus for classic MLE, MME,...
benchonestep  Performing benchmark of one-step MLE against other methods

Description

benchonestep performs a benchmark of one-step MLE against other methods on a given dataset. benchonestep.replicate repeats several times the procedure: data random generation and benchmark through benchonestep.

Usage

benchonestep(data, distr, methods, init, weights=NULL,...)
benchonestep.replicate(nsample, nbsimu, distr, methods=NULL, echo=FALSE, ncpus=1,...)

Arguments

data A numeric vector of length n
distr A character string "name" naming a distribution for which the corresponding density function dname and the corresponding distribution function pname must be classically defined.
methods A vector of methods: character among "mme", "mle", "onestep" (can be abbreviated).
init A named list for the initial guess method.
weights An optional vector of weights to be used in the fitting process. Should be NULL or a numeric vector with strictly positive integers (typically the number of occurrences of each observation). If non-NULL, weighted MLE is used, otherwise ordinary MLE.
... unused for benchonestep; true parameters passed to rdistr for benchonestep.replicate
nsample a numeric for the sample size.
nbsimu a numeric for the replication number.
echo a logical to display or not some traces of benchmarking.
cpus Number of processes to be used in parallel operation: typically one would fix it to the number of available CPUs.

Value

A matrix with estimate and time in seconds per method for benchonestep; an array with estimates, times, errors in seconds per method, per simulation for benchonestep.replicate.

Author(s)

Alexandre Brouste, Darel Noutsa Mieniedou, Christophe Dutang
References


Examples

```r
n <- 1000
set.seed(1234)

x <- rbeta(n, 3, 2)
benchonestep(x, "beta", c("mle", "one"))
```

Description

Executing Le Cam’s one-step estimation based on Le Cam (1956) and Kamatani and Uchida (2015).

Usage

```r
onestep(data, distr, method, init, weights = NULL, 
keepdata = TRUE, keepdata.nb=100, control=list(), ...) 
```

Arguments

data A numeric vector of length n
distr A character string "name" naming a distribution for which the corresponding density function dname and the corresponding distribution function pname must be classically defined.
method A character string coding for the fitting method: "closed formula" for explicit one-step and "numeric" for numeric computation. The default method is the "closed formula".
init A named list for the user initial guess estimation.
weights an optional vector of weights to compute the final likelihood. Should be NULL or a numeric vector with strictly positive integers (typically the number of occurrences of each observation).
keepdata a logical. If TRUE, dataset is returned, otherwise only a sample subset is returned.
keepdata.nb When keepdata=FALSE, the length (>1) of the subset returned.
control a list of control parameters. Currently, param_t is used when the characteristic function is needed, delta is used when the subsample of size n*delta is randomly selected for the initial guess in the generic Le Cam’s one step method.
... further arguments passe to mledist in case it is used.
The Le Cam one-step estimation procedure is based on an initial sequence of guess estimators and a Fisher scoring step or a single Newton step on the loglikelihood function. For the user, the function onestep chooses automatically the best procedure to be used. The function OneStep presents internally several procedures depending on whether the sequence of initial guess estimators is in a closed form or not, and on whether the score and the Fisher information matrix can be elicited in a closed form. "Closed formula" distributions are treated with explicit score and Fisher information matrix (or Hessian matrix). For all other distributions, if the density function is well defined, the numerical computation (NumDeriv) of the Newton step in Le Cam's one-step is proposed with an initial sequence of guess estimators which is the sequence of maximum likelihood estimators computed on a subsample.

onestep returns an object of class "onestep" inheriting from "fitdist". So, it is a list with the following components:

- **estimate**
  - the parameter estimates.

- **method**
  - the character string coding for the fitting method: "closed formula" for closed-form MLE or closed-form one-step, "numeric" for numeric computation of the one-step estimation.

- **sd**
  - the estimated standard errors, NA if numerically not computable or NULL if not available.

- **cor**
  - the estimated correlation matrix, NA if numerically not computable or NULL if not available.

- **vcov**
  - the estimated variance-covariance matrix, NULL if not available.

- **loglik**
  - the log-likelihood.

- **aic**
  - the Akaike information criterion.

- **bic**
  - the the so-called BIC or SBC (Schwarz Bayesian criterion).

- **n**
  - the length of the data set.

- **data**
  - the data set.

- **distname**
  - the name of the distribution.

- **dots**
  - the list of further arguments passed in . . . to be used .

- **convergence**
  - an integer code for the convergence: 0 indicates successful convergence (from explicit formula or not), 10 indicates an error.

- **discrete**
  - the input argument or the automatic definition by the function to be passed to functions gofstat, plotdist and cdfcomp.

- **weights**
  - the vector of weights used in the estimation process or NULL.

Generic functions inheriting from "fitdist" objects:

- **print**
  - The print of a "onestep" object shows few traces about the fitting method and the fitted distribution.
summary The summary provides the parameter estimates of the fitted distribution, the log-likelihood, AIC and BIC statistics and when the maximum likelihood is used, the standard errors of the parameter estimates and the correlation matrix between parameter estimates.

plot The plot of an object of class "onestep" returned by fitdist uses the function plotdist. An object of class "onestep" or a list of objects of class "onestep" corresponding to various fits using the same data set may also be plotted using a cdf plot (function cdfcomp), a density plot (function denscomp), a density Q-Q plot (function qqcomp), or a P-P plot (function ppcomp).

logLik Extracts the estimated log-likelihood from the "onestep" object.
vcov Extracts the estimated var-covariance matrix from the "onestep" object.
coef Extracts the fitted coefficients from the "onestep" object.

Author(s)
Alexandre Brouste, Christophe Dutang, Darel Nouts Mieniedou

References

See Also
See Also as mledist and fitdist in fitdistrplus.

Examples
n <- 1000
set.seed(1234)

#1. Gamma
theta <- c(2, 3)
o.sample <- rgamma(n, shape=theta[1], rate=theta[2])

#Default method
onestep(o.sample, "gamma")

#User initial sequence of guess estimator
# See : Ye and Chen (2017)
qtmp <- sum(o.sample*log(o.sample))-sum(log(o.sample))*mean(o.sample)
alphastar <- sum(o.sample)/qtmp
betastar <- qtmp/length(o.sample)
thetastar <- list(shape=alphastar, rate=1/betastar)
onestep(o.sample, "gamma", init=thetastar)

# Numerical method (for comparison)
onestep(o.sample, "gamma", method="numeric")

## 2. Beta
theta <- c(0.5, 1.5)
o.sample <- rbeta(n, shape1=theta[1], shape2=theta[2])
onestep(o.sample, "beta")

## 3. Cauchy
theta <- c(2, 3)
o.sample <- rcauchy(n, location=theta[1], scale=theta[2])
onestep(o.sample, "cauchy")

# User initial sequence of guess estimator
# See Koutrouvelis (1982).

t <- 1/4
Phi <- mean(exp(1i*t*o.sample))
S <- Re(Phi)
Z <- Im(Phi)
thetastar <- list(location=atan(Z/S)/t, scale=-log(sqrt(S^2+Z^2))/t)
onestep(o.sample, "cauchy", init=thetastar)

## Chi2
theta <- 5
o.sample <- rchisq(n, df=theta)
onestep(o.sample, "chisq")

# User initial sequence of guess estimator
# See Grenander (1965).

p <- n^(2/7)
k <- floor(n^(3/5))
Dstar <- sort(o.sample)
Dk1 <- Dstar[(1+k):n]
Dk2 <- Dstar[1:(n-k)]
sigma.star <- 1/2*sum((Dk1+Dk2)*(Dk1-Dk2)^(-p))/sum((Dk1-Dk2)^(-p))+2
onestep(o.sample, "chisq", init=list(df=sigma.star))
#Negative Binomial

theta <- c(1, 5)
o.sample <- rnbinom(n, size=theta[1], mu=theta[2])
onestep(o.sample, "nbinom")

#Generic (dweibull2)

theta <- c(0.8, 3)
dweibull2 <- function(x, shape, scale, log=FALSE)
  dweibull(x = x, shape = shape, scale = scale, log = log)
o.sample <- rweibull(n, shape = theta[1], scale = 1/theta[2])
onestep(o.sample, "weibull2", method="numeric",
  init=list(shape=1, scale=1))
Index

* distribution
  benchonestep, 3
  onestep, 4
* package
  OneStep-package, 2

benchonestep, 3

cdfcomp, 5, 6

denscomp, 6

fitdist, 6

fitdistrplus, 2

gofstat, 5

mledist, 4, 6

OneStep (OneStep-package), 2
onestep, 4
OneStep-package, 2

plotdist, 5, 6

ppcomp, 6

qqcomp, 6