Package ‘mable’

July 28, 2018

Title Maximum Approximate Bernstein Likelihood Estimation

Version 1.0

Date 2018-07-11

Author Zhong Guan [aut, cre]

Maintainer Zhong Guan <`zguan@iu.edu`>

Depends R (>= 3.0.0)

Suggests graphics, mixtools, stats

Description Fit raw or grouped continuous data from a population with a smooth density on unit interval by an approximate Bernstein polynomial model which is a mixture of certain beta distributions and find maximum approximate Bernstein likelihood estimator of the unknown coefficients. Consequently, maximum likelihood estimates of the unknown density, distribution functions, and more can be obtained. If the support of the density is not the unit interval then transformation can be applied. This is an implementation of the methods proposed by the author this package published in the Journal of Nonparametric Statistics: Guan (2016) <doi:10.1080/10485252.2016.1163349> and Guan (2017) <doi:10.1080/10485252.2017.1374384>.

License GPL (>= 3)

NeedsCompilation yes

Repository CRAN

Date/Publication 2018-07-27 22:20:16 UTC

R topics documented:

`mable-package` *MABLE: Maximum Approximate Bernstein Likelihood Estimation*

Description

Fit raw or grouped data from a nonparametric density f on $[0,1]$ by an approximate Bernstein polynomial model $f_m(x;p)=\sum_{i=0}^m p_i\beta_{mi}(x)$, where β_{mi} is the density of beta distribution with shapes $(i+1, m-i+1)$, $i=0,\ldots,m$, and find maximum approximate Bernstein likelihood estimator of $p=(p_0,\ldots,p_m)$. If the support of f is $[a,b]$ then the data x_i are transformed to $(x_i-a)/(b-a)$.

Details

Package:	mable
Type:	Package
Version:	1.0
Date:	2018-07-11
License: GPL (>= 3)	

Functions:
`bern.approx`: compute the Bernstein polynomial approximate of density or distribution function.
`mable`: fit raw data by the approximate Bernstein polynomial model with a given degree m.
`mable.optim`: choose an optimal model degree m and find the maximum approximate Bernstein likelihood estimator of p based on raw data.
`mable.optim.group`: choose an optimal model degree m and find the maximum approximate Bernstein likelihood estimator of p based on grouped data.
`plotmable`: plot the maximum approximate Bernstein likelihood estimate of density function f or distribution function F.

Author(s)

Zhong Guan <zguan@iu.edu>

References

Guan, Z. (2016) Efficient and robust density estimation using Bernstein type polynomials. *Journal of Nonparametric Statistics*, 28(2):250-271. https://doi.org/10.1080/10485252.2016.1163349

Guan, Z. (2017) Bernstein polynomial model for grouped continuous data. *Journal of Nonparametric Statistics*, 29(4):831-848. https://doi.org/10.1080/10485252.2017.1374384

Examples

```
# Beta distribution
```

```
n<-50; m0<-2; m1<-20
set.seed(1234567)
x<-rbeta(n,7,5)
res<-mable.optim(x, m0, m1)
op<-par(mfrow=c(2,2), mar=c(4,4,2,1))
plot(m0:m1, res$Llik, type="b", xlab="m", ylab="Loglikelihook l(m)")
segments(m<-res$opti.m, 0, m, res$llik, lty=2, col=2)
plot((m0+1):m1, lr<-res$lr, type="b", xlab="m",
                         ylab="LR for Change-Point: R(m)")
segments(m, 0, m, max(lr), lty=2, col=2)
plot(y<-seq(0,1,len=512), dbeta(y,7,5), type="l", xlab="x",
            ylab="f(x)", ylim=c(0, 3.1))
plotmable(res, lty=2, col=2)
lines(density(x, n=512, from=0, to=1), lty=4, col=4)
legend(0, 2.7, bty="n", cex=.8, lty=c(1,2,4),col=c(1,2,4),
           c("TRUE: f",expression(paste("MABLE:",hat(f))),
                 expression(paste("KDE:",tilde(f)))))
plot(y, pbeta(y,7,5), type="l", xlab="x", ylab="F(x)",
                main="CDF of beta(7,5)")
plotmable(res, density=FALSE, lty=2,col=2)
lines(ecdf(x), do.points =FALSE, verticals = TRUE, lty=4, col=4)
legend(0, 1, bty="n", cex=.8, lty=c(1,2,4),col=c(1,2,4),
     c("TRUE: F", expression(paste("MABLE:",hat(F))),
                            expression(paste("ECDF:",F[n]))))
par(op)

# Normal distribution
n<-30; m0<-2; m1<-30
set.seed(13579)
x<-rnorm(n); a<--4; b<-4
res<-mable.optim(x, m0, m1, a, b)
op<-par(mfrow=c(2,2), mar=c(4,4,2,1))
plot(m0:m1, res$Llik, type="b", xlab="m", ylab="Loglikelihook l(m)")
segments(m<-res$opti.m, 0, m, res$llik, lty=2, col=2)
plot((m0+1):m1, lr<-res$lr, type="b", xlab="m",
                              ylab="LR for Change-Point:  R(m)")
segments(m, 0, m, max(lr), lty=2, col=2)
plot(y<-seq(a,b,len=512), dnorm(y), type="l", xlab="x", ylab="f(x)")
plotmable(res, lty=2, col=2)
lines(density(x, n=512, from=a, to=b), lty=4, col=4)
legend(-4, .4, bty="n", cex=.8, lty=c(1,2,4),col=c(1,2,4),
    c("TRUE: f", expression(paste("MABLE:",hat(f))),
                       expression(paste("KDE:",tilde(f)))))
plot(y, pnorm(y), type="l", xlab="x", ylab="F(x)")
plotmable(res, density=FALSE, lty=2, col=2)
lines(ecdf(x), do.points =FALSE, verticals = TRUE, lty=4, col=4)
legend(-4, 1, bty="n", cex=.8, lty=c(1,2,4),col=c(1,2,4),
    c("TRUE: F", expression(paste("MABLE:",hat(F))),
                             expression(paste("ECDF:",F[n]))))
par(op)
```

`bern.approx` *Bernstein Polynomial Approximation*

Description

Returns Bernstein polynomial approximation $f_m(x; p)$ or $F_m(x; p)$

Usage

```
bern.approx(x, p, a=0, b=1, density = TRUE)
```

Arguments

`x`	a vector of values in [0,1] at which the Bernstein polynomial approximation f_m of F_m are evaluated.
`p`	the coefficients of the Bernstein polynomial approximation.
`a`	left endpoint of supporting interval
`b`	right endpoint of supporting interval
`density`	logical; whether a density or a distribution function to be calculated. Default is TRUE.

Details

The $m + 1$ components of p must be nonnegative.

Value

A vector of $f_m(x; p)$ or $F_m(x; p)$ values at x

Author(s)

Zhong Guan <zguan@iu.edu>

References

Guan, Z. (2016) Efficient and robust density estimation using Bernstein type polynomials. *Journal of Nonparametric Statistics*, 28(2): 250-271.

Guan, Z. (2017) Bernstein polynomial model for grouped continuous data. *Journal of Nonparametric Statistics*, 29(4): 831-848.

See Also

`mable.em`

Examples

```
# classical Bernstein polynomial
a<--4; b<-4; m<-200
x<-seq(a,b,len=512)
u<-(0:m)/m
p<-dnorm(a+(b-a)*u)
plot(x, dnorm(x), type="l")
lines(x, (b-a)*bern.approx(x, p, a, b)/(m+1), lty=2, col=2)
legend(a, dnorm(0), lty=1:2, col=1:2, c(expression(f(x)==phi(x)),
                expression(B^{f}*(x))))
```

`chicken.embryo` *Chicken Embryo Data*

Description

The chicken embryo dataset which contains number of days and the corresponding frequencies.

Usage

`data(chicken.embryo)`

Format

The format is: List of 2 $ day: int [1:21] 1 2 3 4 5 6 7 8 9 10 ... $ nT : int [1:21] 6 5 11 2 2 3 0 0 0 0 ...

Source

Jassim, E. W., Grossman, M., Koops, W. J. And Luykx, R. A. J. (1996). Multi-phasic analysis of embryonic mortality in chickens. *Poultry Sci.* 75, 464-71.

References

Kuurman, W. W., Bailey, B. A., Koops, W. J. And Grossman, M. (2003). A model for failure of a chicken embryo to survive incubation. *Poultry Sci.* 82, 214-22.

Guan, Z. (2017) Bernstein polynomial model for grouped continuous data. *Journal of Nonparametric Statistics*, 29(4):831-848.

Examples

```
data(chicken.embryo)
```

`mable.em`	*Fit raw data with the Bernstein polynomial model with a given degree m*

Description

It returns the MABLE of p and the log-likelihood.

Usage

```
mable.em(x, m, a = 0, b = 1, maxit = 50000, eps = 1e-09)
```

Arguments

`x`	raw sample data
`m`	a given model degree
`a`	left endpoint of supporting interval
`b`	right endpoint of supporting interval
`maxit`	maximum number of iterations
`eps`	convergence criterion for iteration

Details

It returns the MABLE of p and the log-likelihood. The MABLE of the coefficients p, the mixture proportions, are obtained using EM algorithm.

Value

`phat`	MABLE of p, the coefficients of Bernstein polynomial
`llik`	log-likelihood
`support`	supporting interval (a,b)

Author(s)

Zhong Guan <zguan@iu.edu>

References

Guan, Z. (2016) Efficient and robust density estimation using Bernstein type polynomials. *Journal of Nonparametric Statistics*, 28(2):250-271.

See Also

`mable.optim`

Examples

```
## Old Faithful Data
  library(mixtools)
  x<-faithful$eruptions
  a<-0;  b<-7
  u<-seq(0,1,len=512); v<-(b-a)*u+a
  mu<-c(2,4.5); sig<-c(1,1)
  pmixem<-normalmixEM(x,.5,mu, sig)
  y1<-pmixem$lambda[1]*dnorm(v,pmixem$mu[1], pmixem$sigma[1])
     +pmixem$lambda[2]*dnorm(v,pmixem$mu[2],pmixem$sigma[2])
  mhat<-94 # preselected optimal degree m
  res<-mable.em(x, m=mhat, a, b, maxit=2000, eps=1.0e-4)
  hist(x, breaks=seq(0,7.5,len=20), xlim=c(0,7), ylim=c(0,.7),
      probability =TRUE, xlab="t", ylab="f(t)", col ="light grey",
      main="Histogram and Density of Duration of Old Faithful")
  lines(density(x, bw = "nrd0", adjust = 1), lty=4, col = 4, lwd=2)
  lines(v, bern.approx(u, p=res$phat)/(b-a), lty=1, col = 1, lwd=2)
  lines(v, y1, lty=5, col=3, lwd=2)
  legend(6,.4, lty=c(5,1,4), col=c(3,1,4), lwd=2, bty="n",
      c(expression(hat(f)[P](t)), expression(hat(f)[B](t)),
      expression(hat(f)[K](t))))
```

`mable.em.group` *Fit grouped data with the Bernstein polynomial model with a given degree m*

Description

It returns the MABLE of p and the log-likelihood.

Usage

```
mable.em.group(x, breaks, m, a=0, b=1, maxit = 50000,
                          eps = 1.0e-9)
```

Arguments

`x`	vector of frequencies
`breaks`	end points of class intervals that partition [0,1] after transformation
`m`	a given model degree m
`a`	left endpoint of supporting interval
`b`	right endpoint of supporting interval
`maxit`	maximum number of iterations
`eps`	convergence criterion for iteration

Details

The MABLE of the coefficients p, the mixture proportions, are obtained using EM algorithm.

Value

`phat`	MABLE of p, the coefficients of Bernstein polynomial
`llik`	log-likelihood
`support`	supporting interval (a,b)

Author(s)

Zhong Guan <zguan@iu.edu>

References

Guan, Z. (2017) Bernstein polynomial model for grouped continuous data. *Journal of Nonparametric Statistics*, 29(4):831-848.

Examples

```
# Chicken Embryo Data
data(chicken.embryo)
a<-0; b<-21
t<-((a:b)-a)/(b-a)
day<-chicken.embryo$day
nT<-chicken.embryo$nT
Day<-rep(day,nT)
res<-mable.em.group(x=nT, t, m=13)
phat<-res$phat; mhat<-13
xx<-seq(0,1,len=100)
fb<-bern.approx(x=xx, p=phat)/(b-a)
fk<-density(x=rep((0:20)+.5, nT), bw="sj", n=101, from=a, to=b)
hist(Day, breaks=seq(a,b,  length=12), freq=FALSE,
         main="Histogram and Density Estimates")
lines(a+(b-a)*xx, fb, lty=2, col=2)
lines(fk, lty=3, col=3)
legend(14, .2, lty=1:3, c("Histogram", "MABLE", "Kernel"),
                           bty="n", col=1:3)
```

`mable.optim` *Fit raw data with the Bernstein polynomial model with optimal degree m*

Description

It also returns the MABLE of p and the log-likelihood.

Usage

```
mable.optim(x, m0, m1, a = 0, b = 1, maxit = 50000,
            eps = 1e-09, tini = 1e-04)
```

Arguments

`x`	raw sample data
`m0`	specify a starting point for choosing an optimal model degree
`m1`	specify an end point for choosing an optimal model degree
`a`	left endpoint of supporting interval
`b`	right endpoint of supporting interval
`maxit`	maximum number of iterations
`eps`	convergence criterion for iteration
`tini`	a tiny positive number to keep initial p in the interior of the simplex in each iteration

Details

An optimal model degree is selected as the change-point of the increments of log-likelihood, log likelihood ratios, for $m \in \{m_0, m_0+1, \ldots, m_1\}$. For each m, the MABLE of the coefficients p, the mixture proportions, are obtained using EM algorithm.

Value

`Llik`	log-likelihoods evaluated at $m \in \{m_0, \ldots, m_1\}$
`phat`	MABLE of p, the coefficients of Bernstein polynomial of the selected optimal degree m
`llik`	log-likelihood at optimal degree m
`opti.m`	the selected optimal degree m
`lr`	likelihood ratios for change-points evaluated at $m \in \{m_0, \ldots, m_1\}$
`support`	supporting interval (a,b)

Note

Since the Bernstein polynomial model of degree m is nested in the model of degree $m+1$, the maximum likelihood is increasing in m. The change-point method is used to choose an optimal degree m.

Author(s)

Zhong Guan <zguan@iu.edu>

References

Guan, Z. (2016) Efficient and robust density estimation using Bernstein type polynomials. *Journal of Nonparametric Statistics*, 28(2):250-271.

See Also

mable.em

Examples

```
# Vaal Rive Flow Data
data(Vaal.Flow)
x<-Vaal.Flow
a<-0
b<-3000
res<-mable.optim(x, m0=2, m1=40, a, b)
m<-res$opti.m
p<-res$phat
op<-par(mfrow=c(1,2))
layout(rbind(c(1, 2), c(3, 3)))
plot(2:40, Lk<-res$Llik, type="b", pch=20,  xlab="m",
                      ylab="Loglikelihook  \u2113(m)")
segments(m,-10, m, Lk[m], lty=2, col=2)
text(m+5.3,45, expression(paste(hat(m)==19)),pos=4)
arrows(m+5, 44, m,41, length = .07, angle = 20, code = 2)
plot(3:40, lr<-res$lr, type="b", pch=20, xlab="m",
                    ylab="LR for Change-Point: R(m)")
segments(m,-1, m, lr[m-2], lty=2, col=2)
text(m+5.1,3.5, expression(hat(m)==19),pos=4)
arrows(m+5, 3.0, m,-1, length = .07, angle = 20, code = 2)
op<-par(lwd=2)
hist(x, prob=TRUE, xlim=c(a,b), ylim=c(0,.0022), breaks=100*(0:30),
  main="Histogram and Densities of the Annual Flow of Vaal River",
border="dark grey",lwd=1,xlab="x", ylab="f(x)", col ="light grey")
lines(density(x, bw = "nrd0", adjust = 1), lty=4, col = 4)
lines(y<-seq(a, b, length=100), dlnorm(y, mean(log(x)),
                   sqrt(var(log(x)))), lty=2, col=2)
lines(y, bern.approx((y-a)/(b-a), p)/(b-a), col=1)
legend(1500, .0015, lty=c(1,2, 4), col=c(1,2, 4), bty="n",
c(expression(paste("MABLE:",hat(f)[B](x))),
        expression(paste("Log-Normal:",hat(f)[P](x))),
               expression(paste("KDE:",hat(f)[K](x)))))
par(op)

## Not run:
# Old Faithful Data
  library(mixtools)
  x<-faithful$eruptions
  a<-0; b<-7
  u<-seq(0,1,len=512); v<-(b-a)*u+a
  mu<-c(2,4.5); sig<-c(1,1)
  pmixem<-normalmixEM(x,.5,mu, sig)
  y1<-pmixem$lambda[1]*dnorm(v,pmixem$mu[1], pmixem$sigma[1])
      +pmixem$lambda[2]*dnorm(v,pmixem$mu[2],pmixem$sigma[2])
  x0<-(x-a)/(b-a); m0<-2; m1<-300
  res<-mable.optim(x0, m0, m1, maxit=2000, eps=1.0e-4)
```

```
  m<-res$opti.m; p<-res$phat
  op<-par(mfrow=c(1,2))
  layout(rbind(c(1, 2), c(3, 3)))
  plot(m0:m1, res$Llik, type="b", pch=20,  xlab="m",
       ylab="Loglikelihook  \u2113(m)")
  segments(m,-10, m, res$Llik[m], lty=2, col=2)
  text(m+20,100, expression(paste(hat(m)==94)),pos=4)
  arrows(m+24, 95, m,70, length = .07, angle = 20, code = 2)
  plot((m0+1):m1, res$lr, type="b", pch=20, xlab="m",
                         ylab="Likelihood Ratio:  R(m)")
  segments(m-2,-10, m-2, res$lr[m-2], lty=2, col=2)
  text(m+20,40, expression(hat(m)==94),pos=4)
  arrows(m+20, 30, m,-7, length = .07, angle = 20, code = 2)
  hist(x, breaks=seq(0,7.5,len=20), xlim=c(0,7), ylim=c(0,.7),
     prob =TRUE,xlab="t", ylab="f(t)", col ="light grey",
     main="Histogram and Density of
               Duration of Eruptions of Old Faithful")
  lines(density(x, bw = "nrd0", adjust = 1), lty=4, col = 4, lwd=2)
  lines(v, bern.approx(u, p)/(b-a), lty=1, col = 1, lwd=2)
  lines(v, y1, lty=5, col=3, lwd=2)
  legend(6,.4, lty=c(5,1,4), col=c(3,1,4), lwd=2, bty="n",
       c(expression(hat(f)[P](t)),expression(hat(f)[B](t)),
                  expression(hat(f)[K](t))))
  par(op)

## End(Not run)
```

`mable.optim.group` *Fit grouped data with the Bernstein polynomial model with an optimal degree m*

Description

It also returns the MABLE of p and the log-likelihood.

Usage

```
mable.optim.group(x, breaks, m0, m1, a=0, b=1,
                              maxit = 50000, eps = 1e-09)
```

Arguments

`x`	vector of frequencies
`breaks`	end points of class intervals that partition [0,1] after transformation
`m0`	specify a starting point for choosing an optimal model degree
`m1`	specify an end point for choosing an optimal model degree
`a`	left endpoint of supporting interval
`b`	right endpoint of supporting interval

`maxit`	maximum number of iterations
`eps`	convergence criterion for iteration

Details

An optimal model degree is selected as the change-point of the increments of log-likelihood, log likelihood ratios, for $m \in \{m_0, m_0 + 1, \ldots, m_1\}$. For each m, the MABLE of the coefficients p, the mixture proportions, are obtained using EM algorithm.

Value

`Llik`	log-likelihoods evaluated at $m \in \{m_0, \ldots, m_1\}$
`phat`	MABLE of p, the coefficients of Bernstein polynomial of the selected optimal degree m
`llik`	log-likelihood at optimal degree m
`opti.m`	the selected optimal degree m
`lr`	likelihood ratios for change-points evaluated at $m \in \{m_0, \ldots, m_1\}$
`support`	supporting interval (a,b)

Author(s)

Zhong Guan <zguan@iu.edu>

References

Guan, Z. (2017) Bernstein polynomial model for grouped continuous data. *Journal of Nonparametric Statistics*, 29(4):831-848.

Examples

```
# Chicken Embryo Data
data(chicken.embryo)
m0<-2; m1<-30; a<-0; b<-21
t<-((a:b)-a)/(b-a);
day<-chicken.embryo$day; nT<-chicken.embryo$nT;
Day<-rep(day,nT)
res<-mable.optim.group(x=nT, t, m0, m1)
p<-res$phat; m<-res$opti.m
op<-par(mfrow=c(1,2), lwd=2)
layout(rbind(c(1, 2), c(3, 3)))
plot(m0:m1, Lk<-res$Llik, type="b", xlab="m",
      ylab=expression(paste("Loglikelihook ",  "\u2113", "(m)")))
segments(m,-130, m, Lk[m], lty=2, col=2)
text(m+5,-119, expression(paste(hat(m)==13)),pos=4, cex=1.5)
arrows(m+5.5, -119.5, m,-121.4, length =.07, angle =20, code=2)
plot((m0+1):m1, lr<-res$lr, type="b", xlab="m", ylim=c(0, max(lr)),
     ylab="Likelihood Ratio: R(m)")
segments(m,-10, m, lr[m-2], lty=2, col=2)
```

```
text(m+3,5, expression(hat(m)==13),pos=4, cex=1.5)
arrows(m+3.45, 4.3, m,-.7, length = .07, angle = 20, code = 2)
u<-seq(0,1,len=100); y<-a+(b-a)*u
fb<-bern.approx(u, p=res$phat)/(b-a)
fk<-density(x=rep((0:20)+.5, nT), bw="sj", n=101, from=a, to=b)
hist(Day, breaks=seq(a,b,  length=12), freq=FALSE,
        main="Histogram and Density Estimates")
lines(y, fb, lty=2, col=2)
lines(fk, lty=3, col=3)
legend(14, .2, lty=1:3, c("Histogram", "MABLE", "Kernel"),
               bty="n", col=1:3)
par(op)
```

`plotmable` *Plot Maximum Approximate Bernstein Likelihood Estimates of f or F*

Description

Takes an output of `mable.em()` or `mable.optim()` and returns various graphical output for the nonparametric estimate.

Usage

```
plotmable(mable.fit, density=TRUE, nx=512, add = TRUE, ...)
```

Arguments

`mable.fit`	mable object containing $\hat{p}$ and (a,b) of `mable.em()` or `mable.optim()`.
`density`	logical; whether a density or a distribution function to be calculated. Default is TRUE.
`nx`	number of points to plot.
`add`	logical; if TRUE only add to an existing plot. Default is TRUE.
`...`	other arguments of `plot()`.

Author(s)

Zhong Guan <zguan@iu.edu>

References

Guan, Z. (2016) Efficient and robust density estimation using Bernstein type polynomials. *Journal of Nonparametric Statistics*, 28(2): 250-271.

Guan, Z. (2017) Bernstein polynomial model for grouped continuous data. *Journal of Nonparametric Statistics*, 29(4): 831-848.

See Also

mable.em, mable.optim, mable.em.group, mable.optim.group

Examples

```
# standard normal
a<--4; b<-4;
x<-rnorm(30)
bfit<-mable.em(x, m=20, a, b)
# density estimate
plot(xx<-seq(a,b,len=512), dnorm(xx), type="l", xlab="x", ylab="f(x)")
plotmable(bfit, lty=2,col=2)
# cdf estimate
plot(xx<-seq(a,b,len=512), pnorm(xx), type="l", xlab="x", ylab="F(x)")
plotmable(bfit, density=FALSE, lty=2,col=2)
```

`Vaal.Flow` *Vaal River Annual Flow Data*

Description

The annual flow data of Vaal River at Standerton as given by Table 1.1 of Linhart and Zucchini (1986) give the flow in millions of cubic metres.

Usage

```
data(Vaal.Flow)
```

Format

The format is: int [1:65] 222 1094 452 1298 882 988 276 216 103 490 ...

References

Linhart, H., and Zucchini, W. (1986), *Model selection*, Wiley Series in Probability and Mathematical Statistics: Applied Probability and Statistics, New York: John Wiley \& Sons Inc.

Examples

```
data(Vaal.Flow)
## maybe str(Vaal.Flow) ; plot(Vaal.Flow) ...
```

Index