Package 'densEstBayes'

July 22, 2025

Version 1.0-2.2

Date 2021-08-19

Title Density Estimation via Bayesian Inference Engines

Maintainer Matt P. Wand <matt.wand@uts.edu.au>

Description Bayesian density estimates for univariate continuous random samples are provided using the Bayesian inference engine paradigm. The engine options are: Hamiltonian Monte Carlo, the no U-turn sampler, semiparametric mean field variational Bayes and slice sampling. The methodology is described in Wand and Yu (2020) <doi:10.48550/arXiv.2009.06182>.

License GPL (>= 2)

LazyData true

Biarch true

Depends R (>= 3.5.0)

Imports MASS, nlme, Rcpp, methods, rstan, rstantools

LinkingTo BH, Rcpp, RcppArmadillo, RcppEigen, RcppParallel, StanHeaders, rstan

SystemRequirements GNU make

NeedsCompilation yes

Author Matt P. Wand [aut, cre] (ORCID: <https://orcid.org/0000-0003-2555-896X>)

Repository CRAN

Date/Publication 2023-03-31 06:43:36 UTC

Contents

`checkChains` *Check Markov chain Monte Carlo samples*

Description

Facilitates a graphical check of the Markov chain Monte Carlo samples ("chains") corresponding to vertical slices of the log-density function estimate at five representative abscissa values.

Usage

```
checkChains(fitObject,locations = "equally-spaced")
```

Arguments

`fitObject`	`densEstBayes()` fit object.
`locations`	Character string that specifies the locations of the vertical slices for the chains to be displayed. The options are: "equally-spaced" for 5 equally-spaced locations over the range of the data. "hexiles" for the locations being the sample hexiles of the data. The default value is "equally-spaced".

Details

A graphic is produced that summarises the Markov chain Monte Carlo samples ("chains") corresponding to vertical slices of the log-density function estimate at five representative abscissae. If the location is specified to be "equally-spaced" and the range.x array of the density estimate is the interval [a,b] then the abscissae are a + j(b-a)/6 for j=1,2,3,4,5. If the location is specified to be "hexiles" then the abscissae are the sample hexiles of the data. The columns of the graphic are the following summaries of each chain: (1) trace (time series) plot, (2) lag-1 plot in which each chain value is plotted against its previous value and (3) sample autocorrelation function plot as produced by the R function acf().

Author(s)

Matt P. Wand <matt.wand@uts.edu.au>

Examples

```
library(densEstBayes) ; data(OldFaithful2011)

# Obtain a density estimate for the `OldFaithful2011' data:

dest <- densEstBayes(OldFaithful2011)

# Obtain a graphic for checking the chains:

checkChains(dest)
```

`densEstBayes` *Density estimation via Bayesian inference engines*

Description

Construction of a high quality density estimate from a random sample is a fundamental problem in Statistics. This function delivers solutions to this problem by calling upon Bayesian inference engine algorithms with roots in the Machine Learning literature. Both stochastic and faster deterministic algorithms are provided. The stochastic options are Hamiltonian Monte Carlo and the no U-turn sampler (through the Stan inference engine) and slice sampling. The deterministic option is semiparametric mean field variational Bayes. The last of these options provides a very quick Bayesian density estimate, whereas the other options take longer to compute but, according to extensive simulation testing, are highly accurate. The essence of the approach is conversion of the density estimation problem to a Bayesian Poisson mixed model problem via binning on a fine grid. An attractive by-product of the binning approach is the ability to handle very large sample sizes. The number of bins, rather than the sample size, is the limiting factor.

Usage

```
densEstBayes(x,method,verbose,control)
```

Arguments

`x`	Vector containing a continuous univariate data set.
`method`	Character string for specifying the method to be used: "HMC" = Hamiltonian Monte Carlo, "NUTS" = the no U-turn sampler, "slice" = slice sampling, "SMFVB" = semiparametric mean field variational Bayes.
`verbose`	Boolean variable for specifying whether or not progress messages are printed to the console. The default is TRUE.
`control`	Function for controlling Markov chain Monte Carlo sample sizes and other specifications.

Details

The crux of the Bayesian density estimation approach is to bin the input data on a fine equally-spaced grid and treat the bin counts as response data in a Bayesian Poisson mixed model-based penalized splines model. Section 8 of Eilers and Marx (1996) provides details on couching the density estimation problem as a Poisson penalized splines problem.

Fitting and inference for the resultant Bayesian Poisson mixed model is carried out using one of four possible approaches as specified by the `method` argument; one of which is deterministic ("SMFVB") and three of which are stochastic ("HMC", "NUTS" and "slice").

The settings `method = "HMC"` and `method = "NUTS"` correspond, respectively, to the Markov chain Monte Carlo approaches Hamiltonian Monte Carlo (e.g. Neal, 2011) and the no-U-turn sampler (Hoffman and Gelman, 2014) and fitting and inference is performed using the Stan Bayesian inference engine via the `rstan` package (Stan Development Team, 2017).

The setting `method = "slice"` corresponds to slice sampling (e.g. Neal, 2003).

The setting `method = "SMFVB"` uses a semiparametric mean field variational Bayes approach as summarised in Algorithm 1 of Luts and Wand (2015). Comparatively speaking, this method leads to a very quick density estimate but at some accuracy and convergence reliability costs.

Value

An object of class `densEstBayes`, which is a list with the following components:

`method`	the value of `method`.
`range.x`	a numerical vector with 2 entries such that range.x[1] is the lower limit and range.x[2] is the upper limit of the interval over which the density estimate was computed.
`intKnots`	a numerical vector containing the locations of the interior knots used in the cubic O'Sullivan spline basis
`determFitObj`	if `method` is "SMFVB" then `determFitObj` is a list with the following components: mu.q.betau: numerical vector containing the mean vector of the Multivariate Normal approximate posterior density function of the linear component coefficients (first two entries) and penalised spline coefficients (remainining entries). Sigma.q.betau: numerical matrix containing the covariance matrix of the Multivariate Normal approximate posterior density function of linear and penalised spline coefficients vector. kappa.q.sigsq: numerical scalar containing the shape parameter of the Inverse Gamma approximate posterior density function of the variance parameter in the mixed model-based penalised spline model. lambda.q.sigsq: numerical scalar containing the scale parameter of the Inverse Gamma approximate posterior density function of the variance parameter in the mixed model-based penalised spline model. If `method` is "HMC", "NUTS" or "slice" then `determFitObj` is returned as NULL.
`stochaFitObj`	if `method` is "HMC", "NUTS" or "slice" then `stochaFitObj` is a list with the following components:

	betauMCMC: numerical matrix containing Markov chain Monte Carlo draws from the posterior distribution of the basis function coefficients, with rows corresponding to the linear component coefficients (first two rows) and penalised spline coefficients (remainining rows) and columns corresponding to the samples. sigsqMCMC: numerical vector containing Markov chain Monte Carlo draws from the posterior distribution of the variance parameter in the mixed model-based penalised spline model. If `method` is "SMFVB" then `stochaFitObj` is returned as NULL.
`xname`	A character string that matches the `x` object. For example, if the call to `densEstBayes()` is `dest <- densEstBayes(alexandrina)` then `xname` is "alexandrina".
`sampHexTran`	Sample hexiles of the transformed input data to be used for possible checking Markov chains Monte Carlo samples.

Author(s)

Matt P. Wand <matt.wand@uts.edu.au>

References

Eilers, P.H.C. and Marx, B.D. (1996). Flexible smoothing with B-splines and penalties (with discussion). *Statistical Science*, **11**, 89-121.

Hoffman, M.D. and Gelman, A. (2014). The no-U-turn sampler: adaptively setting path lengths in Hamiltonian Monte Carlo. *Journal of Machine Learning Research*, **15**, 1593-1623.

Minka, T. (2005). Divergence measures and message passing. *Microsoft Research Technical Report Series*, **MSR-TR-2005-173**, 1-17.

Luts, J. and Wand, M.P. (2015). Variational inference for count response semiparametric regression. *Bayesian Analysis*, **10**, 991-1023.

Neal, R. (2003). Slice sampling (with discussion). *The Annals of Statistics*, **31**, 705-767.

Neal, R. (2011). MCMC using Hamiltonian dynamics. In *Handbook of Markov Chain Monte Carlo*, eds. S. Brooks, A. Gelman, G.L. Jones and X.-L. Meng. Boca Raton, Florida: Chapman & Hall/CRC Press.

Stan Development Team. (2020). *Stan Modeling Language User's Guide and Reference Manual*, https://mc-stan.org.

Wainwright, M.J. and Jordan, M.I. (2008). Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, **1**, 1-305.

Wand, M.P. and Yu, J.C.F. (2020). Density estimation via Bayesian inference engines. Submitted.

Examples

```
library(densEstBayes) ; set.seed(1)
x <- rMarronWand(1000,8)
dest <- densEstBayes(x,method = "SMFVB")
plot(dest) ; rug(x,col = "dodgerblue",quiet = TRUE)
xg <- seq(-3,3,length = 1001)
trueDensg <- dMarronWand(xg,8)
lines(xg,trueDensg,col = "indianred3")
```

```
if (require("rstan"))
{
   dest <- densEstBayes(x,method = "NUTS")
   plot(dest) ; rug(x,col = "dodgerblue")
   xg <- seq(-3,3,length = 1001)
   trueDensg <- dMarronWand(xg,8)
   lines(xg,trueDensg,col = "indianred3")
}
```

`densEstBayes.control` *Controlling density estimation via Bayesian inference engines*

Description

Function for optional use in calls to `densEstBayes()` to control convergence values and other specifications for density estimation using Bayesian inference engines.

Usage

```
densEstBayes.control(range.x = NULL,numBins = 401,numBasis = 50,sigmabeta = 1e5,
                     ssigma = 1000,convToler = 1e-5,maxIter = 500,nWarm = NULL,
                     nKept = NULL,nThin = 1,msgCode = 1)
```

Arguments

`range.x`	A two-component array such that the density estimate is obtained over the interval between range.x[1] and range.x[2]. The default value of range.x[1] is 1.05*min(x)-0.05*max(x) and the default value of range.x[2] is 1.05*max(x)-0.05*min(x).
`numBins`	The number of bins used for binning of the data. The default is 401.
`numBasis`	The number of cubic O'Sullivan spline basis functions. The default is 50.
`sigmabeta`	The prior standard deviation of the fixed effects coefficients corresponding to the linear component of the fit. The default is 100000.
`ssigma`	The prior scale parameter of the standard deviation parameter. The default is 1000.
`convToler`	The convergence tolerance value used for determination of convergence of when the method is semiparametric mean field variational Bayes. Convergence is deemed to have occured when the relative change in the q-density expectation of the reciprocal variance parameter is below convToler. The default is 0.00001.
`maxIter`	The maximum number of iterations when the method is mean field variational Bayes. The default is 500.
`nWarm`	The size of the Markov chain Monte Carlo warmup, a positive integer, when the method is Markov chain Monte Carlo. The default for method = "HMC" and method = "NUTS" is 1000 and the default for method = "slice" is 100.

nKept	The size of the kept Markov chain Monte Carlo samples, a positive integer, when the method is Markov chain Monte Carlo. The default is 1000.
nThin	The thinning factor for the kept Markov chain Monte Carlo samples, a positive integer, when the method is Markov chain Monte Carlo. The default is 1.
msgCode	Code for the nature of progress messages printed to the screen when the method is either "HMC", "NUTS" or "sl ice". If msgCode=0 then no progress messages are printed. If msgCode=1 then a messages printed each time approximately each 10% of the sampling is completed. If msgCode=2 then a messages printed each time approximately each 10% of the sampling is completed. The default is 1.

Value

A list containing values of each of the fifteen control parameters, packaged to supply the `control` argument to `densEstBayes`. The values for `densEstBayes.control` can be specified in the call to `densEstBayes`.

Author(s)

Matt P. Wand <matt.wand@uts.edu.au>

References

Wand, M.P. and Yu, J.C.F. (2020). Density estimation via Bayesian inference engines. Submitted.

Examples

```
library(densEstBayes) ; set.seed(1)
x <- rMarronWand(1000,8)
dest <- densEstBayes(x,method = "SMFVB")
plot(dest) ; rug(x,col="dodgerblue",quiet = TRUE)
xg <- seq(-3,3,length = 1001)
trueDensg <- dMarronWand(xg,8)
lines(xg,trueDensg,col = "indianred3")

# Now modify some of the control values:

destControlled <- densEstBayes(x,method = "SMFVB",control = densEstBayes.control(numBins = 201,
                           numBasis = 35,sigmabeta = 1000,ssigma = 100,convToler = 1e-4))
plot(destControlled) ; rug(x,col = "dodgerblue",quiet = TRUE)
lines(xg,trueDensg,col = "indianred3")
```

---

densEstBayesVignette *Display the package's vignette.*

Description

The vignette of the densEstBayes package is displayed using the default PDF file browser. It provides a detailed description of use of the package for Bayesian density estimation.

Usage

```
densEstBayesVignette()
```

Author(s)

Matt P. Wand <matt.wand@uts.edu.au>

Examples

```
densEstBayesVignette()
```

---

dMarronWand *Marron and Wand density function*

Description

Returns ordinates of a member of the family of Normal Mixture density functions devised in Marron and Wand (1992).

Usage

```
dMarronWand(x,densNum,drv)
```

Arguments

x	Vector of abscissae values.
densNum	An integer between 1 and 15 that specifies the density function according to Table 1 of Marron and Wand (1992).
drv	Either -1,0,1,2 depending on the order of the derivative required, with drv = -1 corresponding to the anti-derivative. The default value is drv = 0.

Author(s)

Matt P. Wand <matt.wand@uts.edu.au>

References

Marron, J.S. and Wand, M.P. (1992). Exact mean integrated squared error. *The Annals of Statistics*, **20**, 712-736.

Examples

```
library(densEstBayes)
xg <- seq(-3,3,length = 1001)
densg <- dMarronWand(xg,8)
plot(xg,densg,type = "l")
```

incomeUK *Incomes of United Kingdom citizens*

Description

The incomeUK numeric vector has 7,201 incomes of United Kingdom citizens for the year 1975. The data have been divided by average income.

Usage

```
data(incomeUK)
```

Format

Each entry of the numeric vector is the ratio of the income of a British citizen and the average income of the sample.

Source

The Economic and Social Research Council Data Archive at the University of Essex, United Kingdom, Family Expenditure Survey, Annual Tapes, 1968-1983, Department of Employment, Statistics Division, Her Majesty's Stationery Office, London.

Examples

```
library(densEstBayes)
data(incomeUK)
hist(incomeUK,breaks = 100,col = "gold")

# Obtain and plot ordinary density estimate:

dest <- densEstBayes(incomeUK,method = "SMFVB",
                     control = densEstBayes.control(range.x = c(0,4)))
plot(dest,xlab = "income (multiple of average income)")
rug(incomeUK,col = "dodgerblue",quiet = TRUE)

# Now obtain and plot improved density estimate using log transformation:

destlogScale <- densEstBayes(log(incomeUK),method = "SMFVB",
                          control = densEstBayes.control(range.x = c(-3,1.2)))
plotVecs <- plot(destlogScale,plotIt = FALSE)
xLogScaleg <- plotVecs$xg
densLowLogScaleg <- plotVecs$densLowg
densEstLogScaleg <- plotVecs$densEstg
densUppLogScaleg <- plotVecs$densUppg
xg <- exp(xLogScaleg)
densLowg <- densLowLogScaleg/xg
densEstg <- densEstLogScaleg/xg
densUppg <- densUppLogScaleg/xg
```

```
plot(0,xlim = range(xg),ylim = range(c(densLowg,densUppg)),type = "n",bty = "l",
     xlab = "income (multiple of average income)",ylab = "density")
polygon(c(xg,rev(xg)),c(densLowg,rev(densUppg)),
        col="palegreen",border = FALSE)
lines(xg,densEstg,col = "darkgreen",lwd = 2)
abline(0,0,col = "slateblue")
rug(incomeUK,col = "dodgerblue",quiet = TRUE)
```

`OldFaithful2011` *Intervals between geyser eruptions*

Description

The `OldFaithful2011` array has time interval between eruptions (minutes) for all 3,507 eruptions of the Old Faithful Geyser (Yellowstone National Park, U.S.A.) in the year 2011.

Usage

```
data(OldFaithful2011)
```

Format

Each entry of the array is the time interval in minutes between each geyser eruption in chronological order.

Source

The Geyser Observation and Study Association (www.geyserstudy.org).

Examples

```
library(densEstBayes) ; data(OldFaithful2011)
hist(OldFaithful2011,breaks = 25,col = "gold",
     xlab = "time interval between geyser eruptions (minutes)")
dest <- densEstBayes(OldFaithful2011,method="SMFVB")
plot(dest,xlab = "time interval between geyser eruptions (minutes)")
rug(OldFaithful2011,col = "dodgerblue",quiet = TRUE)
```

plot.densEstBayes *Plot the Bayesian density estimate from a* densEstBayes() *fit*

Description

The estimated density function obtained via densEstBayes is plotted.

Usage

```
## S3 method for class 'densEstBayes'
plot(x,plotIt=TRUE,credLev=0.95,gridSize=1001,varBand=TRUE,
                          shade=TRUE,estCol="darkgreen",varBandCol=NULL,
                    axisCol="slateblue",add=FALSE,lwd=2,xlab=NULL,ylab=NULL,...)
```

Arguments

x	A densEstBayes() fit object.
plotIt	Boolean flag: TRUE = plot the density estimate and return nothing (the default) FALSE = do not plot the density estimate and, instead, return a list containing an equally-spaced grid of abscissae values and three grids of ordinate values corresponding to the estimate and lower and uppwer limits of pointwise (100*credLev)% sets.
credLev	The number between 0 and 1 such that the credible interval band has (100*credLev)% approximate pointwise coverage. The default value is 0.95.
gridSize	The number of grid points used to display the density estimate curve and the pointwise credible interval band. The default value is 1001.
varBand	Boolean flag: TRUE = add a pointwise approximate (100*credLev)% credible set variability band (the default) FALSE = only plot the density estimate, without a variability band.
shade	Boolean flag: TRUE = display the variability band using shading (the default) FALSE = display the variability band using dashed curves.
estCol	Colour of the density estimate curve. The default value is "darkgreen".
varBandCol	Colour of the pointwise credible interval variability band. If shade=TRUE then the default value is "palegreen". If shade=FALSE then the default value is estCol.
axisCol	colour of the horizontal axis. The default value is "slateblue".
add	Boolean flag: TRUE = add the density estimate curve(s) to an existing plot FALSE = create a new plot for display of the density estimate (the default).
lwd	A positive integer indicating the line width of plotted curves. The default value is 2.

xlab	A character string spedifying the horizontal axis label.
ylab	A character string specifying the vertical axis label.
...	Place-holder for other graphical parameters.

Value

If plotIt=TRUE then a plot is produced on the current device and no numerical values are returned. If plotIt=FALSE then a list is returned with the following components:

xg	numerical vector of abscissae values
densEstg	numerical vector of ordinate values corresponding to the density estimate
densLowg	numerical vector of ordinate values corresponding to the lower limits of the pointwise approximate (100*credLev)% credible set variability band
densUppg	numerical vector of ordinate values corresponding to the upper limits of the pointwise approximate (100*credLev)% credible set variability band

Author(s)

Matt P. Wand <matt.wand@uts.edu.au>

Examples

```
library(densEstBayes) ; data(OldFaithful2011)

# Obtain a density estimate for the `OldFaithful2011' data:

dest <- densEstBayes(OldFaithful2011,method = "SMFVB")

# Plot the density estimate using default settings:

plot(dest,xlab = "time interval between geyser eruptions (minutes)")
rug(jitter(OldFaithful2011,amount=0.2),col = "dodgerblue")

# Plot the density estimate with some user-specified settings:

plot(dest,credLev = 0.999,estCol = "purple",
     varBandCol = "pink",axisCol = "navy",
     xlab = "time interval between geyser eruptions (minutes)")
rug(jitter(OldFaithful2011,amount= 0.2),col = "darkkhaki")

# Plot the density estimate as a black and white line plot:

plot(dest,estCol = "black",shade = FALSE,axisCol = "black",
     xlab = "time interval between geyser eruptions (minutes)")
rug(jitter(OldFaithful2011,amount = 0.2))
```

predict.densEstBayes *Obtain the Bayesian density estimate from a* densEstBayes() *fit at new abscissae*

Description

Values of the estimated density function, obtained via densEstBayes, are computed for new abscissae.

Usage

```
## S3 method for class 'densEstBayes'
predict(object,newdata,cred.fit = FALSE,credLev = 0.95,...)
```

Arguments

object	A densEstBayes() fit object.
newdata	A numerical vector of abscissae values.
cred.fit	Boolean flag: TRUE = compute the lower and upper limits of (100*credLev)% pointwise credible intervals at 'newdata', FALSE = do not computer credible interval limits (the default).
credLev	number between 0 and 1 such that the credible interval band has (100*credLev)% approximate coverage. The default value is 0.95.
...	A place-holder for other prediction parameters.

Value

A list is returned with the following components:

fit	numerical vector of ordinate values corresponding to the density estimate.
credLow.fit	numerical vector of ordinate values corresponding to the lower limits of the pointwise approximate (100*credLev)% credible set variability band.
credUpp.fit	numerical vector of ordinate values corresponding to the upper limits of the pointwise approximate (100*credLev)% credible set variability band.

Author(s)

Matt P. Wand <matt.wand@uts.edu.au>

Examples

```
library(densEstBayes) ; data(OldFaithful2011)

# Obtain a density estimate for the `OldFaithful2011' data:

dest <- densEstBayes(OldFaithful2011,method = "SMFVB")

# Plot the density estimate:

plot(dest,xlab = "time interval between geyser eruptions (minutes)")
rug(jitter(OldFaithful2011,amount = 0.2),col = "dodgerblue")

# Obtain predictions at 60,70,80,90,100 and 110 seconds and
# add to them plot:

newdataVec <- seq(60,110,by = 10)
predictObj <- predict(dest,newdata = newdataVec,cred.fit = TRUE)
print(predictObj$fit)
points(newdataVec,predictObj$fit,col = "blue")

# Print and add to the plot the lower and upper limits of
# the pointwise 95% credible intervals:

print(predictObj$credLow.fit)
print(predictObj$credUpp.fit)
points(newdataVec,predictObj$credLow.fit,col = "red")
points(newdataVec,predictObj$credUpp.fit,col = "red")
```

---

rMarronWand *Marron and Wand random sample*

Description

Returns a random sample from a member of the family of Normal Mixture density functions devised in Marron and Wand (1992).

Usage

```
rMarronWand(n,densNum)
```

Arguments

n	The sample size, which is a positive integer.
densNum	An integer between 1 and 15 that specifies the density function according to Table 1 of Marron and Wand (1992).

Author(s)

Matt P. Wand <matt.wand@uts.edu.au>

References

Marron, J.S. and Wand, M.P. (1992). Exact mean integrated squared error. *The Annals of Statistics*, **20**, 712-736.

Examples

```
library(densEstBayes)
x <- rMarronWand(1000,8)
hist(x,breaks = 50,col = "gold")
dest <- densEstBayes(x,method = "SMFVB")
plot(dest) ; rug(x,col = "dodgerblue",quiet = TRUE)
```

Index