Package: msBP (via r-universe)

September 17, 2024

Type Package

Title Multiscale Bernstein Polynomials for Densities

Version 1.4-1

Date 2023-08-22

Author Antonio Canale

Maintainer Antonio Canale <antonio.canale@unipd.it>

Description Performs Bayesian nonparametric multiscale density estimation and multiscale testing of group differences with multiscale Bernstein polynomials (msBP) mixtures as in Canale and Dunson (2016).

License GPL-2

Suggests R.rsp

VignetteBuilder R.rsp

NeedsCompilation yes

Repository CRAN

Date/Publication 2023-08-23 00:00:13 UTC

Contents

`msBP-package` *Bayesian nonparametric density estimation via Multiscale Bernstein Polynomials (msBP)*

Description

Performs Bayesian nonparametric multiscale density estimation and multiscale testing of group differences with multiscale Bernstein polynomials (msBP) mixtures as in Canale and Dunson (2016).

Author(s)

Antonio Canale <antonio.canale@unipd.it>

References

Canale, A. and Dunson, D. B. (2016), "Multiscale Bernstein polynomials for densities", Statistica Sinica, 26(3), 1175-1195.

Canale, A. (2017), "msBP: An R Package to Perform Bayesian Nonparametric Inference Using Multiscale Bernstein Polynomials Mixtures". Journal of Statistical Software, 78(6), 1-19.

`galaxy` *Galaxy velocities*

Description

Dataset with the velocities of the 82 galaxies reported by Roeder (1990)

Usage

```
data(galaxy)
```

Format

A data frame with 82 observations and a single variable reporting the speed of galaxies (km/second)

Source

Roeder, K. (1990) Density estimation with confidence sets exemplified by superclusters and voids in the galaxies, Journal of the American Statistical Association, 85: 617-624.

References

Escobar, M.D. and West, M. (1995) Bayesian Density Estimation and Inference Using Mixtures. Journal of the American Statistical Association, 90: 577-588.

Examples

```
data(galaxy)
str(galaxy)
```

`msBP.compute.prob` *Compute binary tree of probabilities*

Description

Compute the binary tree of probabilities using the multiscale stick-breaking process of Canale and Dunson (2016).

Usage

```
msBP.compute.prob(msBPtree, root = TRUE)
```

Arguments

`msBPtree`	An object of the class msBPTree
`root`	logical. if the root needs to be considered (default) or it should be cut (fixing $S_{01} = 0$)

Details

Compute a binary tree of weights. The general weights for node h of scale s, is

$$\pi_{s,h} = S_{s,h} \prod_{r<s} (1 - S_{r,g_{shr}}) T_{shr}$$

where $g_{shr} = \lceil h/2^{s-r} \rceil$ and $T_{shr} = R_{r,g_{shr}}$ if $(r+1, g_{shr+1})$ is the right daughter of node (r, g_{shr}), or $T_{shr} = 1 - R_{r,g_{shr}}$ if $(r+1, g_{shr+1})$ is the left daughter of (r, g_{shr}). An object of the msBPTree class is basically a list containing two objects of the class `binaryTree`: the S tree (representing the stoping probabilities) and the R tree (representing the proceed-right probabilities).

Value

An object of the class `msbpTree`.

References

Canale, A. and Dunson, D. B. (2016), "Multiscale Bernstein polynomials for densities", Statistica Sinica, 26(3), 1175-1195.

Canale, A. (2017), "msBP: An R Package to Perform Bayesian Nonparametric Inference Using Multiscale Bernstein Polynomials Mixtures". Journal of Statistical Software, 78(6), 1-19.

See Also

`msBP.rtree`

Examples

```
S <-structure(list( T = list(1/8,c(1/3,1/3), c(1/4,1/4,1/4,1/4),
 rep(1,8)), max.s=3), class  = "binaryTree")
R <-structure(list( T = list(1/2,c(1/2,1/2), c(1/4,1/2,1/2,1/2),
 rep(1,8)), max.s=3), class  = "binaryTree")
RS <-structure(list(S = S, R = R), class  = "msbpTree")
probabilities <- msBP.compute.prob(RS)
```

`msBP.Gibbs` *Gibbs sampling for density estimation for msBP model*

Description

Gibbs sampling for Markov Chain Motecarlo sampling from the posterior distribution of an msBP model.

Usage

```
msBP.Gibbs(x, a, b, g0 = "normal", g0par=c(0,1), mcmc,
 grid = list(n.points=40, low=0.001, upp=0.999), state=NULL, hyper,
 printing=0, maxScale=5, ...)
```

Arguments

`x`	the observed sample
`a`	scalar a parameter
`b`	scalar b parameter
`g0`	prior guess for the density of `x`. Currently only "normal", "unif", "gamma", and "empirical" are supported. From version 1.1 random paramters are also allowed (only with `g0="normal"`).
`g0par`	additional scalar parameters for `g0`. If "normal" corresponds to mean and standard deviation, if "uniform" to upper and lower bounds, if "gamma" to shape and rate parameters. If "empirical" this value is not used. From version 1.1 random paramters are also allowed (only with `g0="normal"`).
`mcmc`	a list giving the MCMC parameters. It must include the following integers: `nb` giving the number of burn-in iterations, `nrep` giving the total number of iterations (including `nb`), and `ndisplay` giving the multiple of iterations to be displayed on screen while the `C++` routine is running (a message will be printed every `ndisplay` iterations).
`grid`	a list giving the parameters for plotting the posterior mean density over a finite grid. It must include the following values: `low` and `upp` giving the lower and upper bound respectively of the grid and `n.points`, an integer giving the number of points of the grid
`state`	a list giving the current value of the parameters. This list is used if the current analysis is the continuation of a previous analysis or if we want to start the MCMC algorithm from some particular value of the parameters.

hyper	a list containing the values of the hyperparameters for a and b or for the parameters of the prior guess (only if g0="normal") . It must contains hyperprior, a list of three logical values determining if hyperpriors for a, b and g0 are used (TRUE) or if a, b, or g0 are fixed (FALSE), and hyperpar a list containing the hyperparameters for the hyperprior distributions: beta, gamma, delta, lambda, mu0, kappa0, alpha0, and beta0. See details. gridB is a grid of values for which the prior (and posterior) for b is evaluated with a Griddy Gibbs approach (Ritter and Tanner, 1992). See details.
printing	Vector of integers if the internal C++ function need to print what is doing
maxScale	maximum scale of the binary trees.
...	additional arguments.

Details

Before calling the proper C++ subrouting the function center the sample on an initial guess for the density of the data. If g0 = 'empirical' the data are transformed so that the expctation of the msBP prior is centered on the kernel density estimate of x.

The algorithm consists of two primary steps: (i) allocate each observation to a multiscale cluster, conditionally on the values of the weights (see also msBP.postCluster); (ii) update the weights, conditionally on the cluster allocations. All the procedure is written in C++ and additional R scripts are used to pre- and post-process the data and the output.

If hyper$hyperpriors$a or hyper$hyperpriors$b is true, additional hyperpriors for a and b are assumed. Specifically the algorithm implements $a \sim Ga(\beta, \gamma)$ and $b \sim Ga(\delta, \lambda)$. For the former parameter the full conditional posterior distribution is available in closed form, i.e.

$$a|- \sim Ga\left(\beta + 2^{s'+1} - 1, \gamma - \sum_{s=0}^{s'}\sum_{h=1}^{2^s} \log(1 - S_{s,h})\right),$$

while for the latter its full conditional posterior is proportional to

$$\frac{b^{\delta-1}}{B(b,b)^{2^{s+1}-1}} \exp\left\{b\left(\sum_{s=0}^{s'}\sum_{h=1}^{2^s} \log\{R_{s,h}(1 - R_{s,h})\} - \lambda\right)\right\},$$

where s' is the maximum occupied scale and $B(p, q)$ is the Beta function. To sample from the latter distribution, a griddy Gibbs approach over the grid defined by hyper$hyperpar$gridB is used. See Ritter and Tanner (1992). From Version 1.1, if hyper$hyperpriors$base=TRUE and g0="normal" additional hyperpriors for the parameter of the centering normal density are assumed. Specifically the model is

$$y = \Phi(x; \mu, \sigma^2)$$

$$(\mu, \sigma^2) \sim N(\mu; \mu_0, \kappa_0\sigma^2)\text{I-Ga}(\sigma^2; \alpha_0, \beta_0)$$

and an addtional step simulating the values of μ and σ^2 from their conditional posterior distribution is added to the Gibbs sampler of Canale and Dunson (2016). Specifically, a Metropolis-Hastings step with proposal equal to the prior is implemented.

Value

A list containing

density	A list containing postMeanDens, the posterior mean density estimate evaluated over xDens and postLowDens and postUppDens, the lower and upper pointwise 95% credible bands,
mcmc	A list containing the MCMC chains: dens is a matrix (nrep-nb) times n.grid, a and b are the vectors with the MCMC chains for the two parameters (if hyperprior was TRUE), scale is a matix where each column is a MCMC chain of the total mass for each scale, R and S, are matrices where each column in the tree2vec form of the corresponding trees, weights is a matrix where each column is the tree2vec form of the corresponding tree of weights, s and h are matrices where each column is the MCMC chain for the node labels for a subject.
postmean	A list containing posterior means over the MCMC samples of a, b, and of all binary trees
fit	A list containing the LPML, mean and median of the log CPO.

References

Canale, A. and Dunson, D. B. (2016), "Multiscale Bernstein polynomials for densities", Statistica Sinica, 26(3), 1175-1195.

Canale, A. (2017), "msBP: An R Package to Perform Bayesian Nonparametric Inference Using Multiscale Bernstein Polynomials Mixtures". Journal of Statistical Software, 78(6), 1-19.

Ritter C., Tanner M. (1992). "Facilitating the Gibbs Sampler: the Gibbs Stopper and the Griddy-Gibbs Sampler." Journal of the American Statistical Association, 87, 861-868.

See Also

msBP.postCluster

Examples

```
## Not run:
data(galaxy)
galaxy <- data.frame(galaxy)
speeds <- galaxy$speed/1000
set.seed(1)
#with fixed g0 and random a, b
fit.msbp.1 <- msBP.Gibbs(speeds, a = 10, b = 5, g0 = "empirical",
 mcmc=list(nrep = 10000, nb = 5000, ndisplay = 1000),
 hyper=list(hyperprior=list(a = TRUE, b = TRUE, g0 = FALSE),
 hyperpar=list(beta=5,gamma = 1,delta = 1,lambda = 1)),
 printing = 0, maxS = 7, grid = list(n.points = 150, low = 5, upp = 38))

#with random a, b and hyperparameters of g0
fit.msbp.2 <- msBP.Gibbs(speeds, a = 10, b=5, g0 = "normal",
 mcmc=list(nrep = 10000, nb = 5000, ndisplay = 1000),
 hyper=list(hyperprior = list(a = TRUE, b = TRUE, g0 = TRUE),
  hyperpar = list(beta = 50, gamma = 5, delta = 10, lambda = 1,
```

```
 gridB = seq(0, 20, length = 30),
 mu0 = 21, kappa0 = 0.1, alpha0 = 1, beta0 = 20)),
 printing = 0, maxS = 7, grid = list(n.points = 150, lo w= 5, upp = 38))

hist(speeds, prob=TRUE,br=10, ylim=c(0,0.23), main="", col='grey')
points(fit.msbp.1$density$postMeanDens~fit.msbp.1$density$xDens, ty='l', lwd=2)
points(fit.msbp.1$density$postUppDens~fit.msbp.1$density$xDens, ty='l',lty=2, lwd=2)
points(fit.msbp.1$density$postLowDens~fit.msbp.1$density$xDens, ty='l',lty=2, lwd=2)

hist(speeds, prob=TRUE,br=10, ylim=c(0,0.23), main="", col='grey')
points(fit.msbp.2$density$postMeanDens~fit.msbp.2$density$xDens, ty='l', lwd=2)
points(fit.msbp.2$density$postUppDens~fit.msbp.2$density$xDens, ty='l',lty=2, lwd=2)
points(fit.msbp.2$density$postLowDens~fit.msbp.2$density$xDens, ty='l',lty=2, lwd=2)


## End(Not run)
```

`msBP.nrvTrees` *Nesting of the sample through the tree*

Description

Compute the path of each subject in the binary tree of weights and returns 3 tree: the `n` tree, the `r` tree, and the `v` tree (see values

Usage

```
msBP.nrvTrees(sh, maxS = max(sh[,1]))
```

Arguments

`sh`	A matrix with 2 columns and a number of rows equal to the sample size denoting the scale and node labels of each unit
`maxS`	Upper bound for the scale

Value

A list containing tree objects of the class `binaryTree`. `n` is the tree containing at each node the number of subjects allocated to that node, `r` is the tree containing at each node the number of subjects that went right at that node, and `v` is the tree containing at each node the number of subjects that passed through that node.

References

Canale, A. and Dunson, D. B. (2016), "Multiscale Bernstein polynomials for densities", Statistica Sinica, 26(3), 1175-1195.

Canale, A. (2017), "msBP: An R Package to Perform Bayesian Nonparametric Inference Using Multiscale Bernstein Polynomials Mixtures". Journal of Statistical Software, 78(6), 1-19.

Examples

```
sh <- cbind(c(2,2,2,3,3,3,3,3,3,3), c(1,2,2,1,2,3,4,5,6,7))
nrv.trees <- msBP.nrvTrees(sh)
plot(nrv.trees$n)
```

`msBP.pdf` *Random probability density function and random cumulative distribution function from a msBP*

Description

Compute the density and the cumulative distribution functions of a random density drawn from an msBP(a,b) process

Usage

```
msBP.pdf(weights, n.points, y)
msBP.cdf(weights, n.points, log, y)
```

Arguments

`weights`	An object of the class binaryTree containing probability weights
`n.points`	Length of the grid over (0,1) in which calculate the value of the random density
`log`	Logical. TRUE for computing the log-cdf
`y`	Vector of values in which the random density is evaluated. If used, `n.points` is not considered.

Value

Vector of size `n.points` or `length(y)`

References

Canale, A. and Dunson, D. B. (2016), "Multiscale Bernstein polynomials for densities", Statistica Sinica, 26(3), 1175-1195.

Canale, A. (2017), "msBP: An R Package to Perform Bayesian Nonparametric Inference Using Multiscale Bernstein Polynomials Mixtures". Journal of Statistical Software, 78(6), 1-19.

See Also

`msBP.rtree`

Examples

```
prob <-structure(list( T = list(0.15, c(0.05,0.05), c(0.05,0.2,0.1,0.1),
  c(0,0,0.3,0,0,0,0,0) ), max.s=3), class  = "binaryTree")
density <- msBP.pdf(prob, 100)
probability <- msBP.cdf(prob, 100)
par(mfrow=c(1,2))
plot(density$dens~density$y, ty='l', main = "pdf")
plot(probability$prob~density$y, ty='l', main = "cdf")
```

`msBP.postCluster` *Posterior cluster allocation*

Description

Perform the posterior multiscale cluster allocation conditionally on a tree of weights. See Algorithm 1 in Canale and Dunson (2016).

Usage

```
msBP.postCluster(y, weights)
```

Arguments

`y`	the sample of individials to be allocated to binary tree structure
`weights`	the binary tree of weights (summing to one). An object of the class msBPTree

Details

conditionally on the weights contained in `weights`, each subject in `y` is allocated to a multiscale cluster using Algorithm 1 of Canale and Dunson (2016). It relies on a multiscale modification of the slice sampler of Kalli et al. (2011).

Value

a matrix with `length(y)` row and two columns, denoting the scale and node within the scale, respectively.

References

Canale, A. and Dunson, D. B. (2016), "Multiscale Bernstein polynomials for densities", Statistica Sinica, 26(3), 1175-1195.

Canale, A. (2017), "msBP: An R Package to Perform Bayesian Nonparametric Inference Using Multiscale Bernstein Polynomials Mixtures". Journal of Statistical Software, 78(6), 1-19.

Kalli, M., Griffin, J., and Walker, S. (2011), "Slice sampling mixture models," Statistics and Computing, 21, 93-105.

See Also

msBP.Gibbs

Examples

```
set.seed(1)
y <- rbeta(30, 5, 1)
weights <-structure(list(
 T = list(0, c(0,0.10), c(0.0,0,0.3,0.6)), max.s=2),
 class  = 'binaryTree')
sh <- msBP.postCluster(y, weights)
clus.size <- msBP.nrvTrees(sh)$n
plot(clus.size)
```

---

msBP.rsample *Random numbers from a random msBP densty*

Description

Random numbers generation from a random density drawn from a msBP process.

Usage

```
msBP.rsample(n, msBPtree)
```

Arguments

n	Size of the sample to be generated
msBPtree	An object of the class msBPtree

Value

A vector containing the random sample

References

Canale, A. and Dunson, D. B. (2016), "Multiscale Bernstein polynomials for densities", Statistica Sinica, 26(3), 1175-1195.

Canale, A. (2017), "msBP: An R Package to Perform Bayesian Nonparametric Inference Using Multiscale Bernstein Polynomials Mixtures". Journal of Statistical Software, 78(6), 1-19.

See Also

msBP.rtree

Examples

```
rand.tree <- msBP.rtree(50,2, 4)
rand.samp <- msBP.rsample(50, rand.tree)
hist(rand.samp, prob=TRUE)
prob <- msBP.compute.prob(rand.tree)
density <- msBP.pdf(prob, 100)
points(density$dens~density$y, ty='l', col=4)
```

---

msBP.rtree *Random msBP tree*

Description

Draw a random tree from the msBP process

Usage

```
msBP.rtree(a, b, max.s = 10)
```

Arguments

a	Scalar parameter
b	Scalar parameter
max.s	Maximum depth of the random trees

Value

An object of the class msBPTree

References

Canale, A. and Dunson, D. B. (2016), "Multiscale Bernstein polynomials for densities", Statistica Sinica, 26(3), 1175-1195.

Canale, A. (2017), "msBP: An R Package to Perform Bayesian Nonparametric Inference Using Multiscale Bernstein Polynomials Mixtures". Journal of Statistical Software, 78(6), 1-19.

See Also

msBP.rsample, msBP.compute.prob

Examples

```
msBP.rtree(2, 2, 4)
```

`msBP.test` *Multiscale testing of group differences*

Description

Performs multiscale hypothesis testing of difference in the distribution of two groups using msBP prior.

Usage

```
msBP.test(y, a, b, group, priorH0 = 0.5,
  mcmc, maxScale = 5, plot.it = FALSE, ...)
```

Arguments

`y`	The pooled sample of observations
`a, b`	Parameters of the msBP prior
`group`	Vector of size `length(y)` with 0 and 1 denoting the group membership.
`priorH0`	Prior gues for the probability of H0
`mcmc`	a list giving the MCMC parameters. It must include the following integers: `nb` giving the number of burn-in iterations, `nrep` giving the total number of iterations (including `nb`)., and `ndisplay` giving the multiple of iterations to be displayed on screen while the MCMC is running (a message will be printed every `ndisplay` iterations).
`maxScale`	maximum scale of the binary trees.
`plot.it`	logical. If TRUE a plot of the posterior mean probability of H0 is produced
`...`	additional arguments to be passed.

Value

a list containing

`Ps`	a matrix with `maxScale` rows and `mcmc$nrep` columns with the MCMC draws of posterior probabilities of H0 for each scale
`Ps`	the posterior mean probabilities of H0 for each scale.

References

Canale, A. and Dunson, D. B. (2016), "Multiscale Bernstein polynomials for densities", Statistica Sinica, 26(3), 1175-1195.

Canale, A. (2017), "msBP: An R Package to Perform Bayesian Nonparametric Inference Using Multiscale Bernstein Polynomials Mixtures". Journal of Statistical Software, 78(6), 1-19.

Examples

```
set.seed(1)
y <- runif(100)
g <- c(rep(0,50), rep(1,50))
mcmc <- list(nrep = 5000, nb = 1000, ndisplay = 500)
## Not run:
test.res <- msBP.test(y, 5, 1, g, mcmc=mcmc, plot.it = TRUE)

## End(Not run)
```

`msBP.tree` *Creating an* `msBPTree`

Description

Create an object of the class `msBPTree`

Usage

```
msBP.tree(max.s = 10)
```

Arguments

`max.s`	Maximum depth of the binary tree

Details

An object of the class `msbpTree` is a list of 5 elements that represent a draw from a msBP(a,b) prior as introduced by Canale and Dunson (2016). The first two elements are the trees of the stopping and descending-to-the-right probabilities, respectively. Both are object of the class `binaryTree`. The third and fourth argument are the hyperparameters of the msBP prior, namely `a` and `b`. The last value is an integer with the maximum depth of both the trees.

Value

An object of the class `msBPTree` with zero at all nodes and `a=b=NULL`.

References

Canale, A. and Dunson, D. B. (2016), "Multiscale Bernstein polynomials for densities", Statistica Sinica, 26(3), 1175-1195.

Canale, A. (2017), "msBP: An R Package to Perform Bayesian Nonparametric Inference Using Multiscale Bernstein Polynomials Mixtures". Journal of Statistical Software, 78(6), 1-19.

See Also

`msBP.rtree`

Examples

```
tree <- msBP.tree(2)
```

`tree2vec` *Conversions between tree and vector*

Description

Convert a binary tree object into a vector and _vice versa_

Usage

```
tree2vec(tree)
vec2tree(vec)
```

Arguments

`tree`	An object of the class binaryTree
`vec`	A vector of numbers. It must have size 2^s - 1, with s an integer.

Details

An object of the class binaryTree is a binary tree containing at each node a value.

Value

A vector of size $2^{D+1} - 1$, where D is the depth of the binary tree, or a binary tree with depth $\log_2(\texttt{length(vec)} + 1)$.

Examples

```
tree <- vec2tree(1:(2^5 - 1))
vector <- tree2vec(tree)
```

Index