Package: bsplinePsd (via r-universe)

September 14, 2024

Type Package

Title Bayesian Nonparametric Spectral Density Estimation Using B-Spline Priors

Version 0.6.0

Date 2018-10-10

Author Matthew C. Edwards [aut, cre], Renate Meyer [aut], Nelson Christensen [aut]

Maintainer Matthew C. Edwards <matt.edwards@auckland.ac.nz>

Description Implementation of a Metropolis-within-Gibbs MCMC algorithm to flexibly estimate the spectral density of a stationary time series. The algorithm updates a nonparametric B-spline prior using the Whittle likelihood to produce pseudo-posterior samples and is based on the work presented in Edwards, M.C., Meyer, R. and Christensen, N., Statistics and Computing (2018). <doi.org/10.1007/s11222-017-9796-9>.

License GPL (>= 3)

Imports Rcpp (>= 0.12.5), splines (>= 3.2.3)

LinkingTo Rcpp

RoxygenNote 6.0.1

Repository https://mcthedwards.r-universe.dev

RemoteUrl https://github.com/mcthedwards/bsplinepsd

RemoteRef HEAD

RemoteSha c3a7ef1a9f32374787b6560db4ed529fe2547b40

Contents

bsplinePsd-package *Bayesian Nonparametric Spectral Density Estimation Using B-Spline Priors*

Description

Implementation of a Metropolis-within-Gibbs MCMC algorithm to flexibly estimate the spectral density of a stationary time series. The algorithm updates a nonparametric B-spline prior using the Whittle likelihood to produce pseudo-posterior samples.

Details

The function `gibbs_bspline` is an implementation of the (serial version of the) MCMC algorithm presented in Edwards et al. (2018). This algorithm uses a nonparametric B-spline prior to estimate the spectral density of a stationary time series and can be considered a generalisation of the algorithm of Choudhuri et al. (2004), which used the Bernstein polynomial prior. A Dirichlet process prior is used to find the weights for the B-spline densities used in the finite mixture and a seperate and independent Dirichlet process prior used to place knots. The algorithm therefore allows for a data-driven choice of the number of knots/mixture components and their locations.

Author(s)

Matthew C. Edwards, Renate Meyer, Nelson Christensen

Maintainer: Matthew C. Edwards <matt.edwards@auckland.ac.nz>

References

Edwards, M. C., Meyer, R., and Christensen, N. (2018), Bayesian nonparametric spectral density estimation using B-spline priors, *Statistics and Computing*, <https://doi.org/10.1007/s11222-017-9796-9>.

Choudhuri, N., Ghosal, S., and Roy, A. (2004), Bayesian estimation of the spectral density of a time series, *Journal of the American Statistical Association*, 99(468):1050–1059.

dbspline *Generate a B-spline density basis of any degree*

Description

This function generates a B-spline density basis of any degree.

Usage

```
dbspline(x, knots, degree = 3)
```

Arguments

x	numeric vector for which the B-spline densities are to be generated
knots	knots used to generate the B-spline densities
degree	positive integer specifying the degree of the B-spline densities (default is 3 for cubic B-splines)

Details

splineDesign is used to generate a B-spline basis of any degree. Each B-spline is then normalised to become a B-spline density using analytical integration. Note that the two boundary knots (0 and 1) are each coincident `degree` + 1 times.

Value

matrix of the B-spline density basis

See Also

splineDesign

Examples

```
## Not run:

# Generate basis functions
set.seed(1)
x = seq(0, 1, length = 256)
knots = sort(c(0, runif(10), 1))
basis = dbspline(x, knots)

# Plot basis functions
plot(x, basis[1, ], type = "l", ylim = c(min(basis), max(basis)),
     ylab = expression(b[3](x)), main = "Cubic B-spline Density Basis Functions")
for (i in 2:nrow(basis)) lines(x, basis[i, ], col = i)

## End(Not run)
```

---

gibbs_bspline — *Metropolis-within-Gibbs sampler for spectral inference of a stationary time series using a B-spline prior*

Description

This function updates the B-spline prior using the Whittle likelihood and obtains samples from the pseudo-posterior to infer the spectral density of a stationary time series.

Usage

```
gibbs_bspline(data, Ntotal, burnin, thin = 1, k.theta = 0.01, MG = 1,
  G0.alpha = 1, G0.beta = 1, LG = 20, MH = 1, H0.alpha = 1,
  H0.beta = 1, LH = 20, tau.alpha = 0.001, tau.beta = 0.001,
  kmax = 100, k1 = 20, degree = 3)
```

Arguments

`data`	numeric vector
`Ntotal`	total number of iterations to run the Markov chain
`burnin`	number of initial iterations to be discarded
`thin`	thinning number (post-processing)
`k.theta`	prior parameter for number of B-spline densities k (proportional to exp(-k.theta*k^2)) in mixture
`MG`	Dirichlet process base measure constant for weights of B-spline densities in mixture (> 0)
`G0.alpha`, `G0.beta`	parameters of Beta base measure of Dirichlet process for weights of B-spline densities in mixture (default is Uniform[0, 1])
`LG`	truncation parameter of Dirichlet process in stick breaking representation for weights of B-spline densities
`MH`	Dirichlet process base measure constant for knot placements of B-spline densities (> 0)
`H0.alpha`, `H0.beta`	parameters of Beta base measure of Dirichlet process for knot placements of B-spline densities (default is Uniform[0, 1])
`LH`	truncation parameter of Dirichlet process in stick breaking representation for knot placements of B-spline densities
`tau.alpha`, `tau.beta`	prior parameters for tau (Inverse-Gamma)
`kmax`	upper bound for number of B-spline densities in mixture
`k1`	starting value for k. If `k1` = NA then a random starting value between `degree` + 2 and `kmax` is selected
`degree`	positive integer specifying the degree of the B-spline densities (default is 3)

Details

The function `gibbs_bspline` is an implementation of the (serial version of the) MCMC algorithm presented in Edwards et al. (2018). This algorithm uses a nonparametric B-spline prior to estimate the spectral density of a stationary time series and can be considered a generalisation of the algorithm of Choudhuri et al. (2004), which used the Bernstein polynomial prior. A Dirichlet process prior is used to find the weights for the B-spline densities used in the finite mixture and a seperate and independent Dirichlet process prior used to place knots. The algorithm therefore allows for a data-driven choice of the number of knots/mixtures and their locations.

Value

A list with S3 class 'psd' containing the following components:

- `psd.median`, `psd.mean` — psd estimates: (pointwise) posterior median and mean
- `psd.p05`, `psd.p95` — 90% pointwise credibility interval
- `psd.u05`, `psd.u95` — 90% uniform credibility interval
- `k`, `tau`, `V`, `Z`, `U`, `X` — posterior traces of model parameters
- `knots.trace` — trace of knot placements
- `ll.trace` — trace of log likelihood
- `pdgrm` — periodogram
- `n` — integer length of input time series

References

Edwards, M. C., Meyer, R., and Christensen, N. (2018), Bayesian nonparametric spectral density estimation using B-spline priors, *Statistics and Computing*, <https://doi.org/10.1007/s11222-017-9796-9>.

Choudhuri, N., Ghosal, S., and Roy, A. (2004), Bayesian estimation of the spectral density of a time series, *Journal of the American Statistical Association*, 99(468):1050–1059.

See Also

plot.psd

Examples

```
## Not run:

set.seed(123456)

# Generate AR(1) data with rho = 0.9
n = 128
data = arima.sim(n, model = list(ar = 0.9))
data = data - mean(data)

# Run MCMC (may take some time)
mcmc = gibbs_bspline(data, 10000, 5000)

require(beyondWhittle)  # For psd_arma() function
freq = 2 * pi / n * (1:(n / 2 + 1) - 1)[-c(1, n / 2 + 1)]  # Remove first and last frequency
psd.true = psd_arma(freq, ar = 0.9, ma = numeric(0), sigma2 = 1)  # True PSD
plot(mcmc)  # Plot log PSD (see documentation of plot.psd)
lines(freq, log(psd.true), col = 2, lty = 3, lwd = 2)  # Overlay true PSD

## End(Not run)
```

plot.psd *Plot method for psd class*

Description

This function plots the log periodogram, log posterior median PSD, and log 90% credible region PSD. The x-axis uses angular frequency and the y-axis is plotted on the log scale. The PSD at the zero frequency is removed from the plot. If the time series is even length, the PSD at the last frequency is also removed from the plot.

Usage

```
## S3 method for class 'psd'
plot(x, legend.loc = "topright", ylog = TRUE, ...)
```

Arguments

x	an object of class psd
legend.loc	location of legend out of "topright" (default), "topleft", "bottomright", "bottomleft". If set to NA then no legend will be produced
ylog	logical value (default is TRUE) to determine if PSD (y-axis) should be on natural log scale
...	other graphical parameters from the plot.default function

Value

plot of the estimate of the (log) PSD

See Also

gibbs_bspline

Examples

```
## Not run:

set.seed(12345)

# Simulate AR(4) data
n = 2 ^ 7
ar.ex = c(0.9, -0.9, 0.9, -0.9)
data = arima.sim(n, model = list(ar = ar.ex))
data = data - mean(data)

# Run MCMC with linear B-spline prior (may take some time)
mcmc = gibbs_bspline(data, 10000, 5000, degree = 1)

# Plot result
```

```
plot(mcmc)

# Plot result on original scale with title
plot(mcmc, ylog = FALSE, main = "Estimate of PSD using the linear B-spline prior")

## End(Not run)
```

Index